



(51) International Patent Classification:
G10L 15/22 (2006.01)

(21) International Application Number:
PCT/US2024/028136

(22) International Filing Date:
07 May 2024 (07.05.2024)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
18/333,041 12 June 2023 (12.06.2023) US

(71) Applicant: **AMAZON TECHNOLOGIES, INC.**
[US/US]; PO Box 81226, Seattle, Washington 98108-1226 (US).

(72) Inventors: **PETERSON, Michael Thomas**; 410 Terry Avenue North, Seattle, Washington 98109-5210 (US). **FU, Gengshen**; 410 Terry Avenue North, Seattle, Washing-

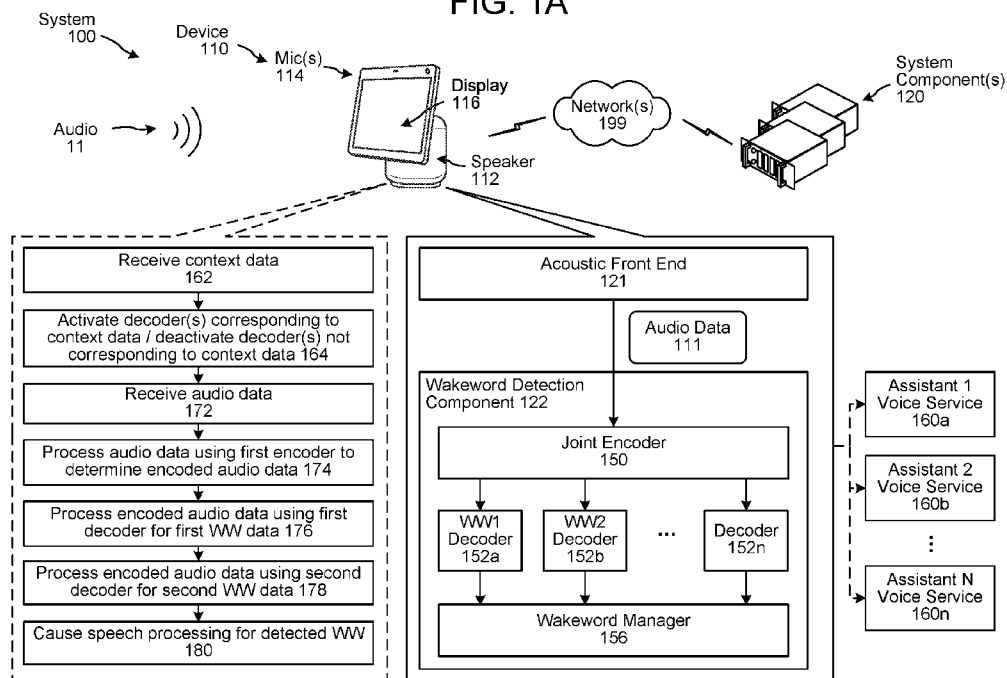
ton 98109-5210 (US). **CHALLENGER, Aaron**; 410 Terry Avenue North, Seattle, Washington 98109-5210 (US). **CHEN, Rong**; 410 Terry Avenue North, Seattle, Washington 98109-5210 (US). **JACQUES, Cody**; 410 Terry Avenue North, Seattle, Washington 98109-5210 (US). **BRADSTREET, Stefan M**; 410 Terry Avenue North, Seattle, Washington 98109-5210 (US).

(74) Agent: **BARZILAY, Ilan**; Pierce Atwood LLP, 100 Summer Street, BOSTON, Massachusetts 02110 (US).

(81) Designated States (unless otherwise indicated, for every kind of national protection available): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CV, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IQ, IR, IS, IT, JM, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, MG, MK, MN, MU, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH,

(54) Title: AUDIO DETECTION

FIG. 1A



(57) Abstract: A device is configured to detect multiple different wakewords. A device may operate a joint encoder that operates on audio data to determine encoded audio data. The device may operate multiple different decoders which process the encoded audio data to determine if a wakeword is detected. Each decoder may correspond to a different wakeword. The decoders may use fewer computing resources than the joint encoder, allowing for the device to more easily perform multiple wakeword processing. Enabling / disabling wakeword(s) may involve the reconfiguring of a wakeword detector to add / remove data for respective decoder(s). Specific decoders may be activated / deactivated depending on device context, thereby efficiently managing device resources.

TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS,
ZA, ZM, ZW.

(84) Designated States (*unless otherwise indicated, for every kind of regional protection available*): ARIPO (BW, CV, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SC, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, ME, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

AUDIO DETECTION

CROSS-REFERENCE TO RELATED APPLICATION DATA

[0001] This application claims the benefit of and priority to U.S. Patent Application No. 18/333,041, filed June 12, 2023, and entitled “AUDIO DETECTION,” in the names of Michael Thomas Peterson, *et al.* The above patent application is herein incorporated by reference in its entirety.

BACKGROUND

[0002] Speech recognition systems have progressed to the point where humans can interact with computing devices using their voices. Such systems employ techniques to identify the words spoken by a human user based on the various qualities of a received audio input. Speech recognition combined with natural language understanding processing techniques enable speech-based user control of a computing device to perform tasks based on the user’s spoken commands. Speech recognition and natural language understanding processing techniques may be referred to collectively or separately herein as speech processing. Speech processing may also involve converting a user’s speech into text data which may then be provided to various text-based software applications.

[0003] Speech processing may be used by computers, hand-held devices, telephone computer systems, kiosks, and a wide variety of other devices to improve human-computer interactions.

BRIEF DESCRIPTION OF DRAWINGS

[0004] For a more complete understanding of the present disclosure, reference is now made to the following description taken in conjunction with the accompanying drawings.

[0005] FIGS. 1A-1B are conceptual diagrams illustrating components and operations of a multi-assistant system for performing multiple wakeword detection, according to embodiments of the present disclosure.

[0006] FIG. 2A is a diagram of a component configured to detect multiple wakewords, according to embodiments of the present disclosure.

[0007] FIG. 2B is a diagram showing multiple wakewords which may be combined to wake a device, according to embodiments of the present disclosure.

[0008] FIG. 2C-2F are diagrams showing sound decoders which may be activated at different times, according to embodiments of the present disclosure.

[0009] FIG. 2G illustrates an example processing of data output by a sound decoder, according to embodiments of the present disclosure.

5 [0010] FIGS. 3A-3B are signal flow diagrams illustrating configuring a device to determine multiple wakewords, according to embodiments of the present disclosure.

[0011] FIG. 4 is a diagram illustrating using context data to activate / deactivate keyword decoder(s), according to embodiments of the present disclosure.

10 [0012] FIG. 5A is a conceptual diagram illustrating components that may be included in a first example implementation of the multi-assistant system, according to embodiments of the present disclosure.

[0013] FIG. 5B is a conceptual diagram illustrating components that may be included in a second example implementation of the multi-assistant system, according to embodiments of the present disclosure.

15 [0014] FIG. 6 is a conceptual diagram of components of a system to detect if input audio data includes system directed speech, according to embodiments of the present disclosure.

[0015] FIG. 7 is a conceptual diagram of an automatic speech recognition component, according to embodiments of the present disclosure.

20 [0016] FIG. 8 is a conceptual diagram of how natural language processing is performed, according to embodiments of the present disclosure.

[0017] FIG. 9 is a conceptual diagram of how natural language processing is performed, according to embodiments of the present disclosure.

[0018] FIG. 10 is a conceptual diagram of components of a dialog manager component, according to embodiments of the present disclosure.

25 [0019] FIG. 11 is a schematic diagram of an illustrative architecture in which sensor data is combined to recognize one or more users according to embodiments of the present disclosure.

[0020] FIG. 12 is a schematic diagram of an illustrative architecture in which sensor data is combined to determine the presence and/or location of one or more users according to embodiments of the present disclosure.

[0021] FIG. 13 is a conceptual diagram of text-to-speech components according to
5 embodiments of the present disclosure.

[0022] FIG. 14 is a block diagram conceptually illustrating example components of a device, according to embodiments of the present disclosure.

[0023] FIG. 15 is a block diagram conceptually illustrating example components of a system, according to embodiments of the present disclosure.

10 [0024] FIG. 16 illustrates an example of a computer network for use with the overall system, according to embodiments of the present disclosure.

DETAILED DESCRIPTION

[0025] Speech processing systems and speech generation systems have been combined with other services to create virtual “assistants” that a user can interact with using natural language
15 inputs such as speech, text inputs, or the like. The assistant can leverage different computerized voice-enabled technologies. Automatic speech recognition (ASR) is a field of computer science, artificial intelligence, and linguistics concerned with transforming audio data associated with speech into text representative of that speech. Similarly, natural language understanding (NLU)
20 is a field of computer science, artificial intelligence, and linguistics concerned with enabling computers to derive meaning from text input containing natural language. ASR and NLU are often used together as part of a speech processing system, sometimes referred to as a spoken language understanding (SLU) system. Text-to-speech (TTS) is a field of computer science concerning transforming textual and/or other data into audio data that is synthesized to resemble human speech. Natural language generation (NLG) is a computing process that produces natural
25 language output (for example, text of words) that can be used to select words to interact with a user as part of an exchange between a user and a system. ASR, NLU, TTS, and NLG may be used together as part of a speech-processing system. The virtual assistant can leverage the speech-processing system as well as additional applications and/or skills to perform tasks for and/or on behalf of the user. A system capable of performing speech processing may also be

capable of performing acoustic-event detection (AED). AED is a field of computer science and artificial intelligence that relates to processing audio data representing a sound, such as a non-speech sound, to determine when and if a particular acoustic event is represented in the audio data.

5 **[0026]** To avoid having a device continue to perform processing on detected audio data, and to direct the processing of such audio data, a speech controlled device may be configured to detect a wakeword that may be spoken by a user to spur a system to perform further processing. As used herein, a wakeword includes sounds used to represent a word, less than a word, or more than one word, such as a wake phrase including more than one word, which, when identified by a
10 machine, transitions a device from a first state to a second state for the purpose of performing more processing on the audio data. For example, the second state can involve sending audio data to a speech processing component, or the like, after a user says “Alexa” as a precursor to further speech that the user intends to be processed by a speech processing system. In another example the second state can involve sending a notification to a device (or other action) in response to
15 detecting a particular sound in the audio data.

[0027] Wakeword detection is an example of keyword detection, which is where a device / system is configured to compare incoming audio to stored data representing one or more words (e.g., keywords) to determine if the incoming audio corresponds to the stored data. If it does, the system may perform an action associated with the detection. For example, in the example of
20 detection of a wakeword, the system may begin sending audio data to later components for speech processing or the like. Keyword detection differs from other speech processing techniques, such as ASR, in that keyword detection can be faster and may not necessarily attempt to make a transcription of speech (like ASR) but rather only outputs an indication of whether the keyword was detected, leaving it to later components to perform the specific
25 action(s) associated with detection of the keyword. Keyword detection may be performed for situations where quick recognition of a word (and operation corresponding thereto) is desired, such as in the case of “waking” a device following detection of a wakeword.

[0028] A device and/or a system may thus be configured to process audio data to determine if properties of the audio data correspond to properties associated with an acoustic event.
30 Examples of acoustic events include a doorbell ringing, a microwave oven beeping, a dog

barking, a window pane breaking, and/or a door closing. The device / system may be configured to perform some action in response to detection of an acoustic event such as sending a notification, starting an alarm, logging the event, or the like.

[0029] Some sound-controlled devices can provide access to more than one speech- or other sound-processing system, where each sound-processing system may provide services associated with a different virtual assistant. In such multi-assistant systems, a speech-processing system may be associated with its own wakeword. Upon detecting a representation of a wakeword in an utterance, the device may send audio data representing the utterance to the corresponding speech-processing system. Additionally, or in the alternative, information about which wakeword is spoken may be used to invoke a particular “personality” of the system and be used as context for AED, NLU, TTS, NLG, or other components of the system. Thus, to enable access to multi-assistant systems, a device can be configured to recognize more than one wakeword. To do so, a device can be configured with wakeword component(s) capable of analyzing incoming speech for each of the wakewords of the corresponding system(s) enabled to process an utterance for the particular device. Wakeword detection may also be enabled on one device and following detection of the wakeword another device may wake. For example, a wakeword may be detected by earbuds / a headset resulting in waking of a phone, or other configurations.

[0030] For one or two wakewords, this may be technically feasible. But the expanding availability of different systems, each accessible through its own wakeword, makes expanding beyond a few wakewords, or desired detectable sounds, technically complex. One reason for this is that wakeword detectors can be computing resource intensive both in terms of incremental computer storage used to store such wakeword detectors and memory / processor usage to operate the wakeword detectors at runtime. If a device is limited in terms of its computing resources, it may be impractical to operate many different wakeword detectors at the same time.

[0031] Offered is, among other things, an improved technical solution to the problem of near-simultaneous detection of different sounds. Although the techniques herein are described in the context of wakewords, they may also be performed with other keywords whose detection causes the system to perform some operation different from a device “wake”, other sounds such as acoustic events, or even other processing that may use encoded audio data in a form as

described herein. Wakeword / keyword detection is split between two layers of components: the first layer is a joint encoder which processes input audio data into feature vectors representing the acoustic units of the input audio data; the second layer is a plurality of decoders—each of which corresponds to a particular wakeword—that can process the feature vectors to determine if the particular wakeword of the decoder is detected. The joint encoder may be trained on thousands or millions of words, representing one or more spoken languages, as well as other non-word sounds made by people (e.g., footsteps, crying, laughing, etc.), other types of animals (e.g., dog, cat, etc.), or innate objects (glass breaking, smoke alarm, unusual automobile noises, etc.), so that it may be flexible for new wakewords and other sounds that may be desired for detection in the future. When a user enables recognition of a new sound, a new decoder may be enabled for the device and the existing joint encoder may still be used. Each decoder may use significantly fewer computing resources than the joint encoder, allowing a device to add and operate multiple decoders in a near-simultaneous (e.g., simultaneous or partially in parallel so as to reduce latency) manner. Thus a device may be able to recognize many different sounds at the same time without overwhelming the computing resources of the device.

[0032] To carefully manage computing resource usage, however, particularly in the face of many such decoders, the system may be configured to activate / deactivate particular decoders depending on system context. This may result in significant resource savings as a sound detection component may otherwise operate in an “always on” configuration, thus potentially using computing resources to detect sound(s) that may not be relevant to the particular operating context of the device. For example, if certain sound(s) are associated with certain user(s) the system may only activate the corresponding decoder(s) for those sound(s) if an associated user is within the same room as a detection device. In another example, if certain sound(s) are associated with a certain device condition (e.g., a keyword of “stop” is enabled when content is being output) the corresponding decoder(s) for those sound(s) may be deactivated when the device condition is not present. In another example, if certain sound(s) are associated with operation of a certain application / skill, the corresponding decoder(s) for those sound(s) may only be activated when the certain application / skill is being operated with respect to a detecting device. In another example, if multiple devices are capable of capturing audio in a particular environment, only one of those devices may activate the corresponding decoder(s) to be detected in the environment.

[0033] To activate and deactivate the corresponding decoder(s) as desired, the decoders may be configured as separate models that may be dynamically loaded and operated by the device. In another configuration, the decoder(s) may comprise portions of a model (e.g., graph sections) that may be dynamically loaded and operated depending on operating context of the device.

5 [0034] These and other features of the disclosure are provided as examples, and maybe used in combination with each other and/or with additional features described herein.

[0035] The system may be configured to incorporate user permissions and may only perform activities disclosed herein if approved by a user. As such, the systems, devices, components, and techniques described herein may be configured to restrict processing where appropriate and only
10 process user information in a manner that ensures compliance with all appropriate laws, regulations, standards, and the like. The system and techniques can be implemented on a geographic basis to ensure compliance with laws in various jurisdictions and entities in which the components of the system and/or user are located.

[0036] In some configurations, a single speech-controlled device can be configured to
15 operate with a single assistant system. For example, an Amazon Echo device would be configured to operate with the Amazon assistant voice service / speech processing system and a Google Home device would be configured to operate with the Google assistant voice service / speech processing system. In some configurations, certain devices can allow a user to select which wakeword the device would recognize (e.g., “Alexa,” “Echo,” “Computer,” etc.), and
20 those wakewords would result only in speech being processed by the same system. In some configurations, devices can be configured to detect multiple wakewords and to send the resulting audio data to a different system depending on the spoken wakeword. For example, the same device would send audio data to an Amazon system if the device detected the wakeword “Alexa” but would send audio data to a Facebook system if the device detected the wakeword “Ok
25 Portal.” Similarly, devices may be configured to detect multiple wakewords that may correspond to the same system, but to different “personalities” or skill configurations within that system. For example, a wakeword of “Hey <Celebrity Name>” may send audio data to an Amazon system but the system would respond (for example using NLG and TTS voice) of the particular celebrity. Each such celebrity may correspond to its own assistant NLG and voice
30 service within a particular speech processing system. In some configurations, a device may be

used by multiple users (for example a device in a family home) where each user has his/her own set of wakewords that the user operates with respect to the device. For example, one user may be configured to use wakeword 1 (for system/assistant 1) and wakeword 2 (for system/assistant 2) while another user may use wakeword 1 (for system/assistant 1) and wakeword 3 (for system/assistant 2). This may be due to a variety of circumstances, such as wakeword 2 and wakeword 3 being associated with different permissions and/or actions of system/assistant 2, personal preference of the users, or some other circumstance.

[0037] A device's wakeword detector may, in some implementations, process input audio data and output a signal when a representation of a wakeword is detected; for example, the wakeword detector may output a logic 0 when no wakeword is detected, transitioning to a logic 1 if/when a wakeword is detected. The wakeword detector may output the wakeword detection signal to client software and/or other components of a system. For example, in some implementations, a wakeword detector may output a wakeword detection signal to an acoustic front end (AFE), in response to which the AFE may begin generating and streaming audio data for processing by the system 100. In some implementations, the wakeword detector may output other metadata upon detecting a wakeword. The other metadata may include a confidence score associated with the detection, fingerprinting (e.g., whether the audio data included a fingerprint signal on the portion representing the wakeword to indicate that the wakeword was output from a media device during, for example, a commercial or other mass media event), and/or other metrics.

[0038] The wakeword detector of the device may process the audio data, representing the audio, to determine whether speech is represented therein. The device may use various techniques to determine whether the audio data includes speech. In some examples, the device may apply voice-activity detection (VAD) techniques. Such techniques may determine whether speech is present in audio data based on various quantitative aspects of the audio data, such as the spectral slope between one or more frames of the audio data; the energy levels of the audio data in one or more spectral bands; the signal-to-noise ratios of the audio data in one or more spectral bands; or other quantitative aspects. In other examples, the device may implement a classifier configured to distinguish speech from background noise. The classifier may be implemented by techniques such as linear classifiers, support vector machines, and decision trees. In still other examples, the device may apply hidden Markov model (HMM) or Gaussian

mixture model (GMM) techniques to compare the audio data to one or more acoustic models in storage, which acoustic models may include models corresponding to speech, noise (e.g., environmental noise or background noise), or silence. Still other techniques may be used to determine whether speech is present in audio data.

5 **[0039]** Wakeword detection can be performed without performing linguistic analysis, textual analysis, or semantic analysis. Instead, the audio data, representing the audio, is analyzed to determine if specific characteristics of the audio data match preconfigured acoustic waveforms, audio signatures, or other data corresponding to a wakeword.

10 **[0040]** Thus, the wakeword detection component may compare audio data to stored data to detect a wakeword. One approach for wakeword detection applies general large vocabulary continuous speech recognition (LVCSR) systems to decode audio signals, with wakeword searching being conducted in the resulting lattices or confusion networks. Another approach for wakeword detection builds HMMs for each wakeword and non-wakeword speech signals, respectively. The non-wakeword speech includes other spoken words, background noise, etc.

15 There can be one or more HMMs built to model the non-wakeword speech characteristics, which are named filler models. Viterbi decoding is used to search the best path in the decoding graph, and the decoding output is further processed to make the decision on wakeword presence. This approach can be extended to include discriminative information by incorporating a hybrid DNN-HMM decoding framework. In another example, the wakeword detection component may be

20 built on deep neural network (DNN) / recursive neural network (RNN) structures directly, without HMM being involved. Such an architecture may estimate the posteriors of wakewords with context data, either by stacking frames within a context window for DNN, or using RNN. Follow-on posterior threshold tuning or smoothing is applied for decision making. Other techniques for wakeword detection, may also be used.

25 **[0041]** One technical challenge to the growing availability of multiple wakewords / assistant voice services / detectable sounds is that each time a system added a new sound to be detected, it would also need to add a new sound detector (such as a new wakeword detector) that a device would be capable of implementing. As many devices (particularly legacy devices that have fewer computing resources but should be enabled to detect new wakewords in some

30 embodiments) are not capable of operating multiple legacy sound detectors at once, addition of a

new wakeword or other sound detector may involve retraining of a complete detection model that could take in input audio data and recognize all the available sound profiles for the particular device / system, depending on system configuration. (For example, a particular device that may not be configured to recognize all such wakewords, a component such as a wakeword manager or the like may simply suppress / ignore any detections of wakewords that a user has not enabled, thus preventing a device from waking if it detects an instance of a word that the user has not indicated should be used to wake the device.) As can be appreciated, this becomes technically challenging as new sounds / wakewords become available, either through the introduction of new speech processing systems, introduction of new assistant voice personalities, introduction of branded or custom voice assistants (for example, one voice assistant for a particular hotel chain, another for a particular theme park, another for a particular store, another for an automobile, etc.).

[0042] The offered solution of dividing sound detection between a joint encoder (which may be used for all sound) and individual sound-specific decoders allows only a new decoder to be trained each time a new sound is to be detected. The existing joint encoder (and any other existing decoders) could be reused without necessitating a retraining of a complete end-to-end new sound detection model that operate in input audio data to detect all the new sound. Further, this division of sound detection allows the system to activate / deactivate certain decoders to preserve computing resources depending on the operating context of the device.

[0043] FIG. 1A is a conceptual diagram illustrating components of a multi-assistant system 100 with default assistant fallback, according to embodiments of the present disclosure. The system 100 may include a device 110, such as the speech controlled device 110 pictured, in communication with one or more remote system component(s) 120 via one or more computer networks 199. The device 110 may include various components for input and output such as one or more displays 116, one or more speakers 112, one or more microphones, 114, etc. In some implementations, the system 100 may detect various gestures (e.g., physical movements) that indicate the system 100 is to receive an input such as audio 11. The system 100 may respond to the user by various means including synthesized speech (e.g., emitted by the speaker 112) conveying a natural language message. Various components of the system 100 as described with reference to FIG. 1A (as well as with reference to other figures herein) may reside in the device 110, and/or the system component(s) 120. In some implementations, various components of the

system 100 may be shared, duplicated, and/or divided between, the device 110, other devices 110, and/or the system component(s) 120. In certain implementations system component(s) 120 may be associated with a cloud service. In other implementations system component(s) 120 may be associated with a home server or other device that resides proximate to a user, thus allowing many operations to happen without the user's data being provided to external components. In other implementations operations of system component(s) 120 may be split between cloud servers / home servers, and/or other components.

[0044] As noted above, the device 110 may provide the user with access to one or more virtual assistants. A virtual assistant may be configured with certain functionalities that it can perform for and/or on behalf of the user. A virtual assistant may further be configured with certain identifying characteristics that may be output to the user to indicate which virtual assistant is receiving input (e.g., listening via the microphone 114), processing (e.g., performing ASR, NLU, and/or executing actions), and/or providing output (e.g., generating output using NLG, speaking via TTS). A user may invoke a particular virtual assistant by, for example, speaking a wakeword associated with that virtual assistant. The system 100 may determine which virtual assistant is to handle the utterance, and process the utterance accordingly; for example, by sending data representing the utterance to the particular speech processing system corresponding to the virtual assistant as illustrated in FIG. 5A and/or processing the utterance using a configuration corresponding to the virtual assistant as illustrated in FIG. 5B.

[0045] FIG. 1A illustrates various components of the system 100 that may be configured to make a determination regarding the presence of one or more wakewords in a received utterance. The system 100 may include an acoustic front end (AFE) 121 that may receive the voice data from the microphone 114 and generate audio data 111 for processing by downstream components (e.g., ASR, NLU, etc.). The AFE 121 may also provide a representation of the utterance to one or more wakeword detectors 122. The AFE 121 may include processing to filter the captured speech. For example, the AFE 121 may perform echo cancelation, noise suppression, beamforming, high- and/or low-pass filtering, etc. The AFE 121 may output both raw audio data and audio data processed using one or more of the aforementioned techniques. The AFE 121 may stream the audio data 111 to a voice activity detector (VAD), the wakeword detector(s) 122, or other components. Although described as a wakeword detection component, the wakeword detection component 122 may also be configured to detect keywords or other

actionable non-word sounds using the encoder / decoder configurations as described herein. Further, although showing certain operations in a particular order, the functions of FIG. 1A, and of any other figures described herein may occur in any order unless explicitly stated otherwise.

[0046] The device 110 may receive (162) context data corresponding to the operational context of the device. Such context data may include a variety of different kinds of data the device may use to determine which sound decoder(s) to activate (and/or deactivate) during operation of the device. Examples of such context data are discussed below in reference to FIG. 4. The device may then activate (164) one or more decoders that correspond to the particular context data / operating context of the device. The device may also deactivate one or more decoders that do not correspond to the particular context data / operating context of the device.

[0047] The wakeword detector 122 can receive (172) the audio data 111 from the AFE 121 and process it to detect the presence of one or more wakewords / keywords. The wakeword detector 122 may be a hardware or software component. For example, the wakeword detector 122 may be executable code that may run without external knowledge of other components. As described above, the wakeword detector 122 may have a two layer architecture. First the wakeword detector 122 may include a joint encoder 150 that processes (174) the input audio data to determine encoded audio data, which may include a plurality of feature vectors representing the input audio data. The encoded audio data may then be sent to the plurality of wakeword decoders 152a through 152n depending on how many are enabled for the particular device and which decoder(s) 152 have been activated pursuant to above step 164. Each of the decoders 152 may be configured to process encoded audio data in the form output by the joint encoder 150. Thus the device 110 may process (176) the encoded audio data using an activated first decoder (e.g., 152a) (which may be associated with a first assistant voice service, e.g., assistant 1 voice service 160a) configured to detect a representation of a first wakeword. The device 110 may also process (178) the encoded audio data using an activated second decoder (e.g., 152b) (which may be associated with a second assistant voice service, e.g., assistant 2 voice service 160b) configured to detect a representation of a second wakeword. The device 110 may also process the encoded audio data using one or more further activated decoders associated with different respective wakewords and/or assistant 2 voice service(s) 160. The device 110, however, may not process the encoded data using one or more deactivated decoders, for example associated with

wakewords / keywords that may not be relevant to the particular operating context of the device as described herein.

[0048] For illustration purposes, a wakeword may be shown as associated with its own assistant voice service, but multiple wakewords may also be associated with a same assistant voice service. For example, a first decoder 152a that detects a first wakeword and second decoder 152b that detects a second wakeword may both be associated with the same assistant voice service (e.g., assistant 1 voice service 160a) depending on system configuration.

[0049] Each decoder may output wakeword decision data to a wakeword manager 156 which may determine, based on the wakeword decision data, if a particular wakeword was represented in the input audio data. While the wakeword manager 156 is illustrated as part of the wakeword detection component 122, it may also be configured as a separate component. The device 110 may then cause (180) speech processing to be performed based on the detected wakeword. For example, by sending audio data 111 (or other data) representing the utterance to the assistant voice service 160 associated with the detected wakeword.

[0050] The above system allows fewer computing resources to be used when adding a wakeword to a system 100 generally and when enabling a new wakeword for a particular device 110 specifically. For example, a system 100 may be capable of working with 10 different wakewords, but only a subset those may be enabled for a particular device 110 at any particular point in time and still further, only a further subset may be actively processing for a particular device 110 depending on its operating context. For example, a first device 110a may be configured to recognize four different wakewords but a second device 110b may be configured to recognize three different wakewords, only two of which are enabled for the first device 110a. And the first device 110a may activate / deactivate its wakewords depending on operating context. Under one previous system configuration, system 100 may have trained a single wakeword model that is capable of recognizing all 10 wakewords and that model may be distributed to many different devices 110, but each device 110 may only cause an action to be performed in response to detection of one of its respective enabled wakewords and may simply ignore any detections of wakewords that are not enabled for the respective device. But in the meantime the device 110 would have spent computing resources operated a large wakeword model for wakewords that were not relevant to it. Each time the system wishes to add a new

possible wakeword, this would enable retraining a large model capable of detect all 11 wakewords at the same time. Such a model may be large and require extensive computing resources by the system 100 to train and deliver to devices and may require extensive computing resources by a device 110 to operate at runtime. Further, this process would need to be repeated
5 each time a new wakeword is enabled by the system (e.g., for wakewords 12, 13, and so forth) to ensure that even legacy devices are capable of detecting new wakewords. Training an individual new wakeword model that can operate on audio data 111 to detect a new wakeword each time may be impractical as it would require a device 110 to run, practically simultaneously, multiple wakeword models on audio data 111. Such operations would be beyond the computing resource
10 capability of many devices 110, making the solution impractical.

[0051] By dividing wakeword detection into the joint encoder 150 and the individual wakeword decoders 152, it allows the heaviest computing operations (e.g., converting the audio data 111 into encoded audio data representing acoustic units) to be handled by a single large model (e.g., the joint encoder 150) and the relatively lightweight (in terms of computing
15 resources) detection of wakewords using the encoded audio data to be handled by as many different smaller models (e.g., decoders 152) as needed for the wakewords operable with the system / enabled with respect to the device 110. Further, the device 110 may activate / deactivate individual decoder(s) 152 as called for by the operating context of the device.

[0052] To enable detection of a new wakeword by a device 110, the system may simply send
20 a device 110 a new decoder that is associated with the new wakeword the device 110 wishes to enable. Practically speaking, however, this may require the device 110 to have sufficient processing capabilities for the device to incorporate the new decoder 152x within its wakeword detection component 122, which may already be configured to operate the joint encoder 150 and other decoders 152a-152n. This incorporation of a new decoder into an existing wakeword
25 detection component 122 may be more processing than a device 110 (particularly one with limited computing capabilities) is capable of performing. Thus, it may be more efficient for the system 100 to construct wakeword component data that incorporates the joint encoder 150 and all the wakewords the device will be configured to detect (including the new wakeword and the device's previous wakewords)

[0053] One example of enabling a new wakeword with respect to a device 110 is shown in FIG. 1B. The operations of FIG. 1B may happen prior to those shown in FIG. 1A (for example, to enable one of the wakewords of FIG. 1A) or may happen after those shown in FIG. 1A (for example to enable a new wakeword after the operations of FIG. 1A). As shown in FIG. 1B a device 110 may be capable of operating with a particular set of wakewords. This may include only a single wakeword, multiple wakewords, etc. depending on device configuration. The device 110 may detect a request to enable a new wakeword. As illustrated in FIG. 1B, this is referred to as the first wakeword. This request may correspond to speech from a user, for example “Alexa” turn on “Sam,” where “Sam” corresponds to a new wakeword. Alternatively the user may operate an application on a companion device (e.g., smartphone, tablet, etc.) to enable a new wakeword, for example using the Amazon Alexa application on a phone, tablet, etc. to enable a new wakeword as selected, for example, from a menu of available wakewords, voice assistants, etc. The device 110 (or companion device) may send (132) a request to system component(s) 120 to enable the first wakeword (WW). The wakeword may be one that is already enabled by the system component(s) 120, such that the system component(s) 120 has already trained a decoder to detect a representation of the first wakeword using the kind of encoded audio data output by the joint encoder 150.

[0054] The system component(s) 120 may receive (134) the request to enable the first wakeword with respect to the first device. The system may determine (136) that the device 110 is already configured to detect a number of other wakewords, including a second wakeword. This may include the system component(s) 120 reviewing profile data associated with the device 110, one or more users of the device 110, etc. to determine which wakeword(s) are enabled with respect to the device 110. The system component(s) 120 may then determine (138) first data of the joint encoder 152, determine (140) second data of the first decoder 152X for the first (e.g., new) wakeword, and determine (142) third data for the second decoder 152b for the second (e.g., currently operable) wakeword that the device 110 can already recognize. The system component(s) 120 may also determine further data representing decoders for any other wakewords the device is capable of recognizing (or wishes to newly recognize, understanding that this process of adding wakeword capability can happen for a single wakeword at a time or for multiple wakewords at a time). The system component(s) 120 can then determine (144) first wakeword component data that includes the first data, second data, third data, and data for any

other decoders of wakewords for the device 110. The system component(s) 120 can then send (146) the first wakeword component data to the device 110. The device 110 can then receive and install (148) the first wakeword component data. Enabling the first wakeword component data may involve deleting previous data associated with the device's wakeword detection component 122 and installing the first wakeword component data with regard to the device's wakeword detection component 122 so that the wakeword detection component 122 may, going forward, using the first wakeword component data (when called for by the context data) to process audio data to detect one or more wakewords.

[0055] In another embodiment the system component(s) 120 may also identify (for example using profile data, or the like) a second device associated with the first device 110. This second device may, for example, be another device in the same home, a device in another location but associated with a same user profile, or the like. The system component(s) 120 may then send, to the second device, the same first wakeword component data as was sent to the first device to allow the second device to also detect the new first wakeword. Or the system component(s) 120 may construct different wakeword component data that includes the first data for the joint encoder 152, the second data for the decoder of the first wakeword and data for different decoders (such as those enabled for the second device) and may send that different wakeword component data to the second device. In this way a user may indicate that a new wakeword should be detectable for one device and the system may (based on user preferences / permissions, etc.) also enable a second device to detect the new wakeword.

[0056] As noted in FIG. 1B, the system component(s) 120 may incorporate information about a device's current list of enabled wakewords when determining and sending updated wakeword component data. In one embodiment the system component(s) 120 may not necessarily engage in such specific customization and may simply create new wakeword component data including data for the joint encoder 150 and for a larger group of decoders, each corresponding to different wakewords, where that group of wakewords may actually include more wakewords than those a user has specifically selected to be operable with regard to device 110. This may be done for several reasons, including allowing a device 110 to more quickly enable recognition of a new wakeword. For example, if a wakeword detection component 122 is configured with the joint encoder 150 and decoders 152 for ten different wakewords, if a user only wishes to enable two of those wakewords, the wakeword manager 156 may be configured to

ignore detections of the eight non-enabled wakewords, which will thus result in no operational change by the device should a non-enabled wakeword be detected. In this situation, if a user later decides to enable a third wakeword, this may simply involve a settings change in the wakeword manager 156, which may be significantly faster than retraining new wakeword component data (for example as shown in FIG. 1B) each time. Thus the new wakeword would be available for use in seconds, rather than making the user wait a longer period of time, which may be undesirable. If a user then wishes to disable the third wakeword, the settings of the wakeword manager 156 may then be changed to ignore any detections of the third wakeword. Which in turn would be less resource intensive than re-installing new wakeword component data that did not include the decoder for the third wakeword. Given that the decoders 152 are significantly smaller than the joint encoder 150, and require fewer computing resources to operate (as explained below) it may result in an improved customer experience to have extra decoders installed on a user device and be ready to go once enabled (e.g., involve a settings change in the wakeword manager 156) than have to retrain a wakeword detection component 122 each time a wakeword is enabled / disabled.

[0057] Further operations of wakeword detection are illustrated in FIG. 2A. As shown, one or more microphones 114 may capture audio of an utterance. The data from the acoustic front end (AFE) 121 may process the microphone data into processed audio data 111. The audio data 111 may be audio data that has been digitized into frames representing time intervals for which the AFE determines a number of values, called features, representing the qualities of the audio data, along with a set of those values, called a feature vector, representing the features/qualities of the audio data within the frame. In at least some embodiments, audio frames may be 10 ms each, 25 ms each, or some other length. Many different features may be determined, as known in the art, and each feature may represent some quality of the audio that may be useful for ASR processing. A number of approaches may be used by an AFE to process the audio data, such as mel-frequency cepstral coefficients (MFCCs), perceptual linear predictive (PLP) techniques, logarithmic filter-bank energies (LFBs), neural network feature vector techniques, linear discriminant analysis, semi-tied covariance matrices, or other approaches known to those of skill in the art.

[0058] Audio frames may be created in a sliding window approach where one frame overlaps in time with another frame. For example, one 25 ms audio frame may include 5 ms of data that

overlap with a previous frame and 5 ms of data that overlap with a next frame. Many such arrangements are possible. This sliding window approach is shown in FIG. 2A. The individual audio frames may make up the audio data 111 that is processed by the joint encoder 150. The joint encoder 150 may input a number of the frames at a time and process them to output
5 encoded audio data 211, representing a feature vector of what acoustic units have been detected within that group of frames. For example, the joint encoder may include a trained model that is configured to input 34 frames of audio data, process that audio data, and output encoded audio data 211, which may include a 100-dimensional feature vector representing which acoustic units
10 were detected in those 34 frames. Such acoustic units may include subword units such as phonemes, diaphonemes, tonemes, phones, diphones, triphones, senons, or the like, depending on system configuration. Further, as can be appreciated, the number of frames of input audio data processed at a time by the joint encoder 150, and/or the dimensional size of the feature vector / encoded audio data 211 output by the joint encoder 150 are also configurable.

[0059] As the encoded audio data 211 output by the joint encoder 150 may be processed by
15 multiple different decoders 152, including potentially decoders for wakewords that will be enabled by the system component(s) 120 long after the joint encoder is trained / deployed, the encoder may be trained on a large vocabulary of words (e.g., 5,000 words) to allow for many different acoustic units to be operable with regard to the encoder, not just those that may correspond to the limited number of wakewords the device 110 may be enabled to recognize at a
20 specific moment in time. As a result, the joint encoder 150 may be relatively large and computing resource intensive.

[0060] Each individual decoder 152, however, may only be trained to determine whether one word (or a small group of words) is represented in captured audio by processing the encoded audio data 211. Thus, the individual decoders may be relatively small and not as intensive in
25 terms of computing resource usage compared to the joint encoder 150. Although in certain configurations joint encoder 150 may be included in wakeword detection component 122, in other configurations joint encoder 150 may be outside wakeword detection component 122.

[0061] As a result, the encoded audio data 211, that resulted from relatively high cost encoder operations, may be sent to multiple decoders for relatively low cost decoding /
30 wakeword detection operations, at only a small increase over the computing resources needed for

the encoding, which had to be performed anyway. Thus, adding the ability for a device to detect a new wakeword may be done at a relatively low incremental computing cost.

[0062] The tables below show examples of the relative cost operations of the joint encoder 150 versus the decoder(s) 152. For example, in one configuration the joint encoder 150 comprises a convolutional neural network (CNN) with the configuration as shown in Table 1:

	Input				Kernel			Striding		Pooling		DivM	Multiplications per frame	#weights	Receptive field		Stride		Layer type
1	34	64	1		7	5	96	1	1	2	3	1	201,600.00	3,360.00	1	1	1	1	CNN
2	14	20	96		5	3	128	3	1	1	2	0.16666667	552,960.00	184,320.00	8	7	2	3	CNN
3	4	9	128		2	4	128	1	1	1	1	0.16666667	131,072.00	131,072.00	16	16	6	6	CNN
4	3	6	128		2	3	160	1	1	1	1	0.16666667	81,920.00	122,880.00	22	34	6	6	CNN
5	2	4	160		2	4	160	1	1	1	1	0.16666667	34,133.33	204,800.00	28	46	6	6	FC
6	1	1	160		1	1	500	1	1	1	1	0.16666667	13,333.33	80,000.00	34	64	6	6	FC
7	1	1	500		1	1	100	1	1	1	1	0.16666667	8,333.33	50,000.00	34	64	6	6	FC

Table 1

As shown, this example of the joint encoder 150 has seven layers, with four CNN layers followed by three fully connected (FC) layers. 34 frames of audio data are input into the first layer and processed as noted by successive layers. A significant number of multiplications per frame may occur in the earlier layers of the joint encoder 150 in order to perform the processing necessary to eventually arrive at the encoded audio data 211. This shows why implementing multiple such encoders may be impossible for devices 110 that have limited available computing resources for such operations.

[0063] As can be appreciated, the joint encoder 150 combined with a decoder 152 may be considered as separated layers of an end-to-end operation. Thus, a first layer of a decoder 152 may be considered to be layer eight of a wakeword detection component 122 if the joint encoder 150 has seven layers. This notation is used to illustrate the examples of decoders 152 below.

One benefit to the offered solution is that a first decoder 152a used to detect a first wakeword need not necessarily have the same construction as second decoder 152b used to detect a second wakeword. As different models may work better to detect different wakewords, different models / neural network arrangements may be used for different wakewords. Thus wakeword 1 decoder 152a may have a different construction from wakeword 2 decoder 152b and so forth. (Certain decoders may also have a similar construction depending on system configuration.)

[0064] Thus, in one configuration a decoder 152 (e.g., 152a) may be a CNN configured as illustrated in Table 2:

	Input			Kernel			Striding		Pooling		DivM	Multiplications per frame	#weights	Receptive field		Stride		Layer type
8	4	1	100	4	1	100	1	1	1	1	0.16666667	6,666.67	40,000.00	34	64	6	6	FC
9	1	1	100	1	1	2	1	1	1	1	0.16666667	33.33	200.00	52	64	6	6	Linear
11	1	1	2											34	64	6	6	Output

Table 2

5 while another decoder (e.g., 152b) may be a convolutional neural network configured as illustrated in Table 3:

	Input			Kernel			Striding		Pooling		DivM	Multiplications per frame	#weights	Receptive field		Stride		Layer type
8	6	1	100	6	1	100	1	1	1	1	0.16666667	10,000.00	60,000.00	34	64	6	6	FC
9	1	1	100	1	1	2	1	1	1	1	0.16666667	33.33	200.00	64	64	6	6	Linear
11	1	1	2											34	64	6	6	Output

Table 3

10

or the other decoder may be a CNN configured as illustrated in Table 4:

	Input			Kernel			Striding		Pooling		DivM	Multiplications per frame	#weights	Receptive field		Stride		Layer type
8	9	1	100	5	1	80	1	1	1	1	0.16666667	6,666.67	40,000.00	34	64	6	6	CNN
9	5	1	80	5	1	40	1	1	1	1	0.16666667	2,666.67	16,000.00	58	64	6	6	FC
10	1	1	40	1	1	2	1	1	1	1	0.16666667	13.33	80.00	82	64	6	6	Linear
11	1	1	2											34	64	6	6	Output

Table 4

15 or may be a CNN configured as illustrated in Table 5:

	Input			Kernel			Striding		Pooling		DivM	Multiplications per frame	#weights	Receptive field		Stride		Layer type
8	12	1	100	6	1	80	1	1	1	1	0.16666667	8,000.00	48,000.00	34	64	6	6	CNN
9	7	1	80	7	1	40	1	1	1	1	0.16666667	3,733.33	22,400.00	64	64	6	6	FC
10	1	1	40	1	1	2	1	1	1	1	0.16666667	13.33	80.00	100	64	6	6	Linear
11	1	1	2											34	64	6	6	Output

Table 5

[0065] The decoder 152 need not necessarily be a CNN. It may be a recurrent neural network (RNN) such as a convolutional recurrent neural network (CRNN). For example, it may be an RNN including a long short-term memory (LSTM) layer as illustrated in Table 6:

	Input			Kernel			Striding		Pooling		DivM	Multiplications per frame	#weights	Receptive field		Stride		Layer type
8	17	1	100	1	1	100	1	1	1	1	0.16666667	174,583.33	80,000.00	34	64	6	6	LSTM
9	17	1	100	1	1	100	1	1	1	1	0.16666667	41,666.67	10,000.00	34	64	6	6	FC
10	17	1	100	1	1	2	1	1	1	1	0.16666667	1,250.00	300.00	34	64	6	6	Linear
11	17	1	2											34	64	6	6	Output

Table 6

or may be an RNN configured as illustrated in Table 7:

	Input			Kernel			Striding		Pooling		DivM	Multiplications per frame	#weights	Receptive field		Stride		Layer type
8	17	1	100	1	1	20	1	1	1	1	0.16666667	8,250.00	9,600.00	34	64	6	6	LSTM
9	17	1	20	1	1	20	1	1	1	1	0.16666667	1,666.67	400.00	34	64	6	6	FC
10	17	1	20	1	1	2	1	1	1	1	0.16666667	250.00	60.00	34	64	6	6	Linear
11	17	1	2											34	64	6	6	Output

Table 7

As can be appreciated, many different decoder constructions may be possible. For example, a decoder may include a finite state transducer (FST), support vector machine (SVM), residual neural network (resnet), other configuration(s) and/or combinations thereof. Certain decoder constructions may be more well suited to short wakewords (e.g. “Ok”) while others may be more well suited to longer wakewords (e.g., “Amelia”). Further, certain decoder constructions may be more well suited for detecting certain combinations of sounds while other decoder constructions may be more well suited for detecting other combinations of sounds.

[0066] Individual decoder(s) 152 may operate as separate models that are activated / deactivated as described herein. In such a configuration, the device 110 may send the encoded audio data 211 to the activated decoder(s) and may not send the encoded audio data 211 to the deactivated decoders(s). Alternatively, the device 110 may send the encoded audio data 211 to all the decoder(s) (for example on a shared bus) and may instruct certain decoder(s) to not operate with regard to the encoded audio data 211. In another example, the individual decoder(s) 152 may operate as a combined model, for example with certain shared layers or other aspects.

In such a configuration, the device 110 may deactivate portions of the model (for example, portions of a graph to be traversed) corresponding to wakewords to be deactivated based on the operating context of the device.

[0067] The joint encoder 150 may also be of any available configuration, though each decoder may be configured to operate on the data type output by that particular configuration of joint encoder 150. The split encoder / decoder configuration allows the system component(s) 120 to configure different types of decoders for different wakewords while only using the single joint encoder configuration. (Different types of joint encoders may also be used, but this may lead to increased system complexity and multiple decoders needed for the same wakeword depending on the different joint encoder configurations, etc. and thus may not be desirable.)

[0068] The individual decoders 152 may operate on the encoded audio data 211 to determine if the particular wakeword of the decoder is detected. Thus the decoders may be trained to process the feature vectors of the encoded audio data 211 to detect if the individual acoustic units corresponding to the respective wakeword of that decoder are represented in the encoded audio data 211. Each decoder 152 may output wakeword decision data 252 corresponding to whether the wakeword was detected. The wakeword decision data 252 may include a two-dimensional data vector with one dimension representing a likelihood that the wakeword was detected and the other dimension representing a likelihood that the wakeword was not detected. (In certain configurations, these likelihoods may add to 1, for example with a likelihood score of detection being 0.7 and the likelihood score of non-detection may be 0.3, but this is left to system design.) The wakeword decision data 252 may also include time data indicating, for example, when a wakeword began, how long it lasted, when it ended, etc. which may be used by downstream components. The wakeword decision data 252 may be output on a frame-by-frame basis or on some other basis depending on system configuration / configuration of the particular decoder 152. A deactivated decoder 152 may not output decision data 252, thus conserving resources of downstream components (such as the wakeword manager 156) that would otherwise operate on decision data 252 that may not be relevant to a device's operating context.

[0069] The wakeword manager 156 may be configured to receive the wakeword decision data 252 and determine, using that wakeword decision data 252, if a wakeword was detected. For example, if any particular wakeword decision data 252 includes a likelihood / score above a

threshold, the wakeword manager 156 may determine that a wakeword was detected. A single threshold may be used or a different threshold may be customized for each decoder 152. Other conditions may also be used by the wakeword manager 156 to determine if a wakeword was detected. For example, a likelihood / score below a threshold, within a certain range, multiple likelihoods / scores within a range within a certain period of time, etc. A single condition may be used or a different condition may be customized for each decoder 152. The wakeword manager 156 may output wakeword data 260 which may include an indication that a wakeword was detected (or not), the identifier of the particular wakeword that was detected, time data related to the particular detected wakeword, etc. The device 110 may then process the wakeword data 260 to determine that a particular wakeword was detected and to cause a speech processing component associated with that particular wakeword to perform processing using the audio data 111 or other audio representing the captured audio. The device 110 may also process the wakeword data 260 to determine to perform some other operation, depending on system configuration.

[0070] As can be appreciated, a wakeword may comprise one or more words. For example, “Alexa” may be considered one wakeword while “Ok Google” may be considered a different wakeword. As such, each may have its own decoder 152 associated with it to allow it to be recognized by device 110. One additional benefit, however, of the present construction, is that it may allow the system 100 to operate using split wakeword configurations more easily. For example, if a user wishes to use (or is required to use) the word “Computer” as a wakeword for a device 110, but also uses the word “computer” in regular conversation, it may be desirable for the user to add a preceding word such as “Ok,” “Hey,” or the like such that only the combination of words “Ok Computer” results in a wake by the device (e.g., acts like a wakeword for purposes of interactions with the system 100) while a regular mention of “computer” will not. But other users may prefer a combination such as “Hey Computer” or “Wakeup Computer.” In another scenario a user may wish to use a single wakeword (e.g., “Alexa”) to activate a first assistant voice service and a combination wakeword (e.g., “Hey Sam”) to activate a second assistant voice service while a different user of the same device may prefer a different combination of wakewords (e.g., “Hi Alexa” to activate the first assistant voice service and “Wakeup Sam” to activate the second assistant voice service). Previous wakeword detector training and operation

did not offer this kind of flexibility due to the difficulties in training as well as the difficulties implementing such operations at runtime.

[0071] The present split configuration of the joint encoder 150 and individual decoders 152, however, may enable such user (or system) configured multiple wakeword detections. In

5 particular, a device 110 may be configured with multiple decoders that are capable not only of detection of what would be considered “primary” wakewords (e.g., “Alexa,”

“[CelebrityNameA],” “[CelebrityNameB],” etc.) but also preceding words (e.g., “Hey,” “Hi,” “Wakeup,” etc.) or even interim words or words to otherwise assist wakeword detection /

assistant voice activation. Such configuration is shown in FIG. 2B. As the individual words are

10 detected (by processing encoded audio data 211 with their respective decoders 152), their

respective wakeword decision data 252 may be sent to the wakeword manager 156. The

wakeword manager 156 may process the wakeword decision data 252, which may include time data for each of the wakewords, to detect if certain words were detected within a certain time of each other, for example, detecting “Hi” right before “Alexa”. If the user configures the device

15 110 to operate using a combination of words in this manner (for example through spoken

command with the system 100 and/or through use of a companion application), the settings of the wakeword manager 156 may be updated accordingly. For example, the wakeword manager

156 may operate a state machine to track detections of wakewords, relative time related thereto, etc. Thus, if two wakewords are detected within a certain time of each other, and that

20 combination corresponds to a wake function with regard to a particular assistant voice service,

the wakeword manager 156 may indicate in the wakeword data 260a that the particular assistant voice service has been invoked, and the device 110 / system may act accordingly, for example by

causing speech processing to be performed using the particular assistant voice service. In this manner many different wakeword combinations may be configured and used to wake a device

25 and invoke assistant voice service(s).

[0072] As noted above, a device 110 may be a group device associated with multiple users.

The device may be associated with a device / group profile. Each user may also be associated with their own user profile. Profile data may be stored in profile storage 570 discussed below.

As part of system customization, one user may be associated with a first set of wakewords while

30 another user may be associated with a second set of wakewords. (Still other users may also be

associated with their own set of wakewords.) The users may also have one or more wakewords

in common. In certain situations, it is desirable to only have a device operate wakeword decoder(s) for the particular user that is in the environment of the device to avoid operating decoder(s) that are not relevant to that particular user.

[0073] Such a configuration of decoders is shown in FIG. 2C. As shown, a device 110 may be enabled to detect a plurality of wakewords. One wakeword, “Alexa,” may be shared among multiple users while others may be specific to certain users. For example, user 1 may have two customized wakewords associated with the user, for example WWa and WWb. In one example, WWa may correspond to a first assistant voice service while WWb may correspond with a second assistant voice service. The device 110 may have previously been enabled to detect these two wakewords (for example as a result of performing some operations to customize operations for user 1). Thus the device 110 may have installed / enabled decoder 152g (corresponding to WWa) and decoder 152h (corresponding to WWb). In this example, user 2 may have three customized wakewords, for example WWc, WWd, and WWe. In one example, WWc may correspond to a first assistant voice profile with the first assistant voice service, WWd may correspond to a second assistant voice profile (e.g., different assistant personality) with the first assistant voice service, while WWe may correspond with the second assistant voice service. The device 110 may have previously been enabled to detect these three wakewords (for example as a result of performing some operations to customize operations for user 2). Thus the device 110 may have installed / enabled decoder 152j (corresponding to WWc), decoder 152k (corresponding to WWd), and decoder 152l (corresponding to WWe). The device 110 may also have other enabled decoder(s) (e.g., decoder 152n and/or others).

[0074] At runtime the device 110 may determine data relevant to the operating context of the device. That data may indicate that only user 2 is within the environment of the device. Thus the device 110 may activate decoders 152d (corresponding to the universal wakeword “Alexa”) and decoders 152k, 152k, and 152l (corresponding to the wakeword associated with user 2). The device 110 may deactivate decoders 152g and 152h (corresponding to user 1) as well as any other decoders. The device 110 may then process audio data as described herein and only use the activated decoders to perform wakeword detection.

[0075] As noted above, wakeword detection is an example of keyword detection. The device 110 may be configured to activate / deactivate decoder(s) corresponding to non-wakeword

keywords in the same manner as described with regard to wakewords. In certain configurations a device 110 may default to decoder(s) being active or inactive. The device 110 may then adjust such defaults as called for by the context data and/or processing herein. Thus, activating or deactivating a decoder may involve either leaving the decoder in its default state or change its default state depending. Such a configuration of decoders is shown in FIG. 2D. As shown the device 110 may be enabled to detect certain keywords under a plurality of operating conditions. One examples of such a keyword may be “stop”, corresponding to decoder 152r. One benefit to enabling such keyword detection by the device 110 is that a user may verbally instruct the device 110 to stop its operation without necessarily preceding that command to stop with a wakeword. Further, if the device 110 coordinates with system component(s) 120 for speech processing and the connection between the device 110 and the system component(s) 120 is somehow interrupted, the device 110 may be able to perform actions in response to a user saying “stop” as a result of keyword detection (using decoder 152r) performed on the device 110 without suffering loss of functionality due to the connection interruption.

[0076] The device 110 may also be enabled to detect keyword(s) corresponding to different applications / skills. For example, the device may be configured (for example using a process described above with regard to FIG. 2B) to detect keywords associated with a particular skill / application. For example, skill 590a may be associated with two keywords, KWA1 and KWA2. The device 110 may enable operation of two decoders associated with those keywords, specifically decoder 152m (for KWA1) and decoder 152n (for KWA2). In another example, skill 590b may be associated with three keywords, KWB1, KWB2, and KWB3. The device 110 may enable operation of three decoders associated with those keywords, specifically decoder 152o (for KWB1), decoder 152p (for KWB2), and decoder 152q (for KWB3). Such keywords may be associated with skill / application specific functionality such as song skip, pause, etc. for a music skill /application or image zoom, rotate, etc. for a photo skill / application, or the like. As the device 110 is operating, the user may invoke skill B, which may run with respect to the device 110 and/or the user. The device 110 may receive context data indicating operation of skill B 590b with respect to the device 110. Thus the device may activate decoders 152o, 152p, and 152q. (Depending on the state of skill A 590a with respect to the device 110, the device may also activate decoder(s) 152m and/or 152n.) In certain situations the device 110 may only operate a subset of skill-associated keywords. For example, a particular keyword may only be

associated with certain processes of a skill. The device 110 may receive context data indicating when the certain processes are active and may activate the corresponding decoder(s) for those processes at the appropriate time and may otherwise leave those decoder(s) deactivated. This configuration allows the device 110 to more quickly respond to specific keywords (by using only keyword detection rather than invoking a more involved speech processing pipeline) while also preserving computing resources by using the divided encoder / decoder approach described herein and while only activating certain decoder(s) depending on the device's operational context. As discussed above in reference to FIG. 2A, each decoder 152 may output respective decision data 252 indicating a probability that the respective keyword was detected. The output of the keyword decoders shown in FIG. 2D may go to a keyword manager 156b which processes the decision data to determine keyword data 260b indicating information about potentially detected keyword(s).

[0077] The device 110 (and/or system component(s) 120) may also be configured with components for acoustic event detection (AED). Such components may include an AED component 573 and AED storage 571, shown in FIG. 5A discussed below. The AED component 573 may be configured to process incoming audio data and compare the incoming data to data stored in AED storage 571 to determine if the incoming audio data matches that of a type of acoustic event such as a doorbell ringing, an appliance beeping, a dog barking, glass breaking, a door closing, a baby crying, water running, an alarm, or any number of system configured and/or user customized sounds. In certain situations AED component 573 may involve its own mechanism for processing audio data and/or comparing processed audio data with data in AED storage 571 to determine whether an acoustic event is represented in capture audio. In certain situations the device 110 may be configured to perform AED using encoded audio data 211 output by joint encoder 150, where individual decoder(s) 152 are associated with particular sounds corresponding to specific acoustic events. Thus the joint encoder / separate decoder configuration may also be used for sounds / wake triggers that are not necessarily spoken words but can be specific sounds such as acoustic events like a glass breaking, a baby crying, a doorbell, etc. This will allow the described joint encoder / separate decoder configuration to be used for AED. Thus the device may be configured with one or more decoders corresponding to a particular acoustic event and may process incoming audio data as described herein to determine if an acoustic event is detected. Detection of such an acoustic event may be represented in

output data 260 and may be processed by the device 110 and/or system component(s) 120 to perform a variety of actions such as notifying another device associated with the user / user profile of device 110, notifying a different device (such as a device associated with a security monitoring service), or take other actions as indicated, for example, in user settings / a user profile corresponding to the particular acoustic event. An example of decoders configured to detect acoustic events are illustrated in FIG. 2E.

[0078] As shown in FIG. 2E, a plurality of AED decoders 152 may be configured to process the encoded audio data 211 to determine if an acoustic event is detected. The decoder data 152 for the acoustic events may be activated / deactivated as described herein. For example, the device may have one decoder 152s configured to process encoded audio data 211 to determine an output probability 252s that a doorbell ringing was part of captured audio; the device may also have another decoder 152t configured to process encoded audio data 211 to determine an output probability 252t that an appliance beeping was part of captured audio, and so forth such that the device 110 may have multiple decoders 152 each corresponding to a respective acoustic event and thus may be able to detected a large variety of different acoustic events using such decoders 152 depending on system configuration. (Such AED decoders 152 may be stored with AED storage 571 and/or elsewhere.) Thus, the device may be configured to perform AED using the structure of the joint encoder 150 / multiple decoders 152.

[0079] Certain combinations of AED decoders may be activated / deactivated in groups. For example, if the system processes context data to determine that a first user's preferences are enabled, then decoders (e.g., 152t and 152u) corresponding to acoustic events relevant to the first user may be activated. If the system processes context data to determine that a second user's preferences are enabled, then decoders (e.g., 152v, 152w, and 152x) corresponding to acoustic events relevant to the second user may be activated. In certain configurations both users' preferences are enabled and thus both users' corresponding acoustic event decoders (and/or WW decoders, keyword decoders, etc.) may be activated. As discussed above in reference to FIG. 2A, each decoder 152 may output respective decision data 252 indicating a probability that the respective acoustic event was detected. The output of the acoustic event decoders shown in FIG. 2E may go to an acoustic event detection manager 156c which processes the decision data to determine AED data 260c indicating information about potentially detected acoustic event(s).

The acoustic event detection manager 156c may be part of the AED component 573 or may be differently configured depending on system configuration.

[0080] In one example, as discussed herein, each decoder 152 comprises data making up a separate model from other decoder(s). In another example, certain decoder(s) 152 may correspond to decoder data that is incorporated in larger models where the decoder data may be activated / deactivated depending on context data as described herein.

[0081] Thus, in certain configurations the device 110 may be configured to operate multiple combined decoder models, such as that shown in FIG. 2F. For example, the device may be configured to operate one wakeword combined decoder model 153a, where individual wakeword decoder data 152 corresponding to specific wakewords are combined in the wakeword combined decoder model 153a and different portions of the decoder data 152 within that model 153a may be activated / deactivated depending on context data as described herein. In another example, the device may be configured to operate another combined model 153b corresponding to detecting non-wakeword keywords, where individual keyword decoder data 152 corresponding to specific keywords are combined in the keyword combined decoder model 153b and different portions of the decoder data 152 within that model 153b may be activated / deactivated depending on context data as described herein. In still another example, the device may be configured to operate another combined model 153c corresponding to detecting acoustic events, where individual keyword decoder data 152 corresponding to specific acoustic events are combined in the AED combined decoder model 153c and different portions of the decoder data 152 within that model 153c may be activated / deactivated depending on context data as described herein. Such a model 153 may output a plurality of probabilities / decision data 252 corresponding to the individual sounds / decoder data 152 incorporated into the model 153.

[0082] As can be appreciated, a variety of such arrangements are possible. For example, in one configuration one combined decoder model 153 may comprise decoder data (e.g., 152g and 152h) corresponding to a plurality of wakewords enabled for a first user (e.g., user 1) while another combined decoder model 153 may comprise decoder data (e.g., 152j, 152k, and 152l) corresponding to a plurality of wakewords enabled for a second user (e.g., user 2), such as illustrated above in reference to FIG. 2C. Such combined models may operate in parallel with individual models (for example, models 153a, 153b, etc. operating in parallel with model 152r).

[0083] Although FIG. 2F shows different managers 156a-156c receiving data from different decoders 152/153, the output data from certain decoder(s) may go to multiple different managers 156. For example, one active word may be considered both a wakeword and a keyword and thus the output data from the decoder 152 configured to detect the word may be sent to 156a and
5 156b. Various other such configurations are possible.

[0084] The descriptions herein focus on the joint encoder / multiple decoder configuration as corresponding to sound detection, where each decoder may correspond to outputting a probability if a particular sound was detected. However, individual decoders 152 may also be configured to process encoded audio data 211 for other purposes. For example, a decoder 152
10 may be configured to output a probability that there is media (e.g., music, audio from a video, etc.) being output in a particular environment. In another example, a decoder 152 may be configured to output a probability that a specific media content item (e.g., a specific commercial, song, or the like) is being presented in an environment. In another example, a decoder 152 may be configured to output a probability that an input is directed at a device, for example along the
15 lines of SDD 585 discussed below. In another example, a decoder 152 may be configured to output a probability that detected audio corresponds to a particular user, for example along the lines of user recognition component 595 discussed below. In another example, a decoder 152 may be configured to output a probability that detected audio corresponds to a user being present in the environment, for example along the lines of presence detection component 594 discussed
20 below. As can be appreciated, a decoder 152 may be configured to process encoded audio data 211 for a variety of purposes depending on system configuration.

[0085] The probability output by a decoder 152 may also be used by a component along with other data to perform various processing. Such processing may be performed by a component that sits between a decoder 152 and output manager 156. Such a component is shown in FIG.
25 2G. As illustrated, an event detector 230 may be configured to output probability data 253 corresponding to a likelihood that a particular event has occurred. The event may correspond to an acoustic event but may also correspond to a different kind of event. The event detector 230 may operate on decision data 252 (such as that output by a decoder 152) but also may be configured to operate on other data 262 which may include a variety of data such as image data,
30 audio data, ASR data 710, NLU results data 985/925, dialog data, user recognition data 1195, user presence data 1295, or any variety of data included herein or otherwise configured to be

processed by the system. Thus, the probability data 253 output by the event detector 230 may incorporate information from other sources beyond decoder(s) 152. The probability data 253 may be sent to one or more manager components 256, which may be similar to manager components 156, in that manager component(s) 256 may be configured to process the probability data to determine if an event has occurred and may output an indication of the event and/or other data relevant to the event as output data 260.

[0086] As noted above, one particular wakeword may be associated with a different assistant voice service. Thus, enabling wakewords as discussed herein may allow a device 110 to interact with a different assistant voice service. Processes enabled by the configurations herein allow a device to more easily reach new assistant voice services for speech processing, etc. For example, a first device 110a, as shown in FIG. 3A, may already be configured to send audio data to a first assistant voice service 160a. The device 110, along with the system component(s) 120, may determine (302) a request for enablement of a new sound for a second assistant voice service 160b. The system component(s) may also determine a request for enablement of new sound detection. Such enablement may be determined, for example, by a user making a spoken request for such enablement / activation and/or operating an application (for example on a tablet device) associated with the system 100 and requesting enablement / activation of the ability to detect the new sound. The system component(s) 120 may then determine (304) a group of sounds (e.g., wakewords, keywords, acoustic events, etc.) currently active for a first device (for example by reviewing profile data associated with first device 110a). This may include a wakeword already detectable by the device 110 to interact with the first assistant voice service 160a. The system component(s) 120 may determine (306) first sound detection component data including joint encoder data and decoder data for all enabled sounds for the first device, which may now include not only the wakeword associated with the first assistant voice service 160a but also the new wakeword associated with the second assistant voice service 160b and/or any additional sounds. This first wakeword component data may be considered a “wakeword model” in that the data for the joint encoder may be considered the early layers of the wakeword model and the data for the decoder(s) may be considered the later layers of the wakeword model. Alternatively the system 100 may send decoder data 152 corresponding to the new sounds to be enabled without sending new data corresponding joint encoder 150. The system component(s) 120 may send (308) the first sound detection component data to the first device 110 which may then configure (310) the

first sound detection component data for further processing, for example by using it to update / reinstall a wakeword detection component 122. The first device 110a may then process (312) input audio data using the first sound detection component data (as installed / operating on the first device 110a) to detect the new wakeword / sound and then send (314) audio data to the system component(s) 120 / second assistant voice service 160b for processing with regard to the second assistant voice service 160b (which may use components of system component(s) 120 or may be separate therefrom).

[0087] If a user then wishes to enable the first device 110a to interact with still another assistant voice service, the process may be easily repeated as shown in FIG. 3B. The device 110, along with the system component(s) 120, may determine (316) a request for enablement of a further new wakeword, this time for a third assistant voice service 160c. The system component(s) 120 may then determine (318) a group of wakewords currently active for a first device (for example by reviewing profile data associated with first device 110a). This may include a wakeword already detectable by the device 110 to interact with the first assistant voice service 160a as well as the new wakeword (added as discussed above with regard to FIG. 3A) for the second assistant voice service 160b. The system component(s) 120 may determine (320) second sound detection component data including joint encoder data and decoder data for all enabled wakewords for the first device, which may now include not only the wakeword associated with the first assistant voice service 160a and the second assistant voice service 160b but also the further new wakeword associated with the third assistant voice service 160c. The system component(s) 120 may send (322) the second sound detection component data to the first device 110 which may then configure (324) the second sound detection component data for further processing, for example by using it to update / reinstall a wakeword detection component 122. The first device 110a may then process (326) input audio data using the second sound detection component data (as installed / operating on the first device 110a) to detect the further new wakeword and then send (328) audio data to the system component(s) 120 / third assistant voice service 160c for processing with regard to the third assistant voice service 160c (which may use components of system component(s) 120 or may be separate therefrom).

[0088] The system may also repeat the operations for FIG. 3A and/or FIG. 3B with a second device 110b, which may be associated with a different profile and/or different already-enabled wakewords than the first device 110a. Thus the system component(s) 120 may configure

different wakeword component data for different devices, where the wakeword component data sent to a first device may have data for the same joint encoder but different data for the different decoder(s) depending on which wakewords are enabled for the second device 110b relative to the first device 110a. In this manner the split configuration of the joint encoder 150 and individual decoders 152 adds further flexibility to the system component(s) 120.

[0089] In one embodiment the system may, in addition to or in the alternative of creating customized wakeword component data each time a device enables / disables a wakeword, create pre-existing wakeword component data packages ready for deployment upon request. The system may thus create ahead of time, and store for deployment when appropriate, wakeword models (e.g., wakeword component data) corresponding to the top N most popular wakeword combinations. For example, if the top three most popular wakeword (WW) combinations are combination 1: {WW1, WW3, WW5}, combination 2: {WW1, WW2, WW4}, and combination 3: {WW2, WW3, WW5} the system may, even before requested by a device, create three sets of wakeword component data corresponding to each combination. Thus, for combination 1 the system may create wakeword component data 1 (e.g., a wakeword model) including data for the joint encoder as well as data for decoders corresponding to WW1, WW3, and WW5. For combination 2 the system may create wakeword component data 2 (e.g., a wakeword model) including data for the joint encoder as well as data for decoders corresponding to WW1, WW2, and WW4. For combination 3 the system may create wakeword component data 3 (e.g., a wakeword model) including data for the joint encoder as well as data for decoders corresponding to WW2, WW3, and WW5. Then, if device that is configured to detect wakewords 1 and 3 sends a request to enable detection of wakeword 5, the system may simply send that device the previously created and stored wakeword component data 1. While this approach may involve expending computing resources initially in preparing and storing the wakeword component data for the wakeword combinations, doing so may reduce latency experienced at runtime when a device requests a new wakeword to be enabled (or disabled). Rather than needing to configure the wakeword component data in a custom manner for the device, the system may simply send the device the pre-configured wakeword component data, assuming the new combination of wakewords to be enabled for the device is among the combinations for which the system has pre-configured wakeword component data available. Depending on system configuration and available resources, the system may create such pre-configured wakeword component data for

many different wakeword combinations, for example the most popular 50, 100, etc. combinations, thus reducing latency for many users when enabling / disabling wakewords.

[0090] FIG. 4 is a diagram illustrating using context data to activate / deactivate keyword decoder(s), according to embodiments of the present disclosure. As shown, a device 110 receives (162) context data. The context data can take many different forms. For example, context data may include settings and/or profile data or other data which may indicate which sound(s) are to be detected for a particular device / user. The context data may also include audio data, image data, and/or sensor data from one or more sensors of device 110 and/or of other devices (such as those in the environment of the device 110). The context data may include user recognition data 1195, such as that output by a user recognition component 595 described below and/or user presence data 1295, such as that output by a presence detection component 594 described below. Such user recognition data 1195 and/or user presence data 1295 may indicate one or more specific users are in the environment of the device 110 and thus certain sound decoder(s) 152 associated with the user (e.g., as indicated in a user profile) should be activated.

[0091] The context data may also include information corresponding to other device(s) that may be capable of detecting audio in an environment. Thus, if multiple devices are capable of capturing audio in a particular environment, only one of those devices may activate the corresponding decoder(s) to be detected in the environment.

[0092] The context data may include an indicator of one or more skills / applications 590 that are active with respect to the device 110, which thus may cause the device 110 to activate certain sound decoder(s) 152 associated with the skill(s) / application(s) 590.

[0093] The context data may also represent a network connection status of a device 110. For example, if a device loses connection to another device (e.g., system component(s) 120), the device 110 may enable detection of certain sounds to compensate for losing the connection.

[0094] The context data may also include dialog data, such as that corresponding to a dialog manager 572 described below where the dialog data indicates the status of certain skill(s) / application(s) 590 where certain keyword(s) / keyword decoder(s) 152 should be activated / deactivated corresponding to the status. Such dialog data may also include dialog history or

other data which may correspond to activation / deactivation of certain keyword(s) / keyword decoder(s) 152.

[0095] Context data may also include indicators of other active sound(s) and/or detected sound(s). For example, a sound detector(s) for one sound may only be activated for a certain
5 time period after a different sound was detected. The context data may reflect this prior detected sound, thus indicating that the specified sound detector(s) for the one sound should be activated for the certain time period. In such an example, the context data may also include time data.

[0096] Context data may also indicate which primary wakeword is to be used for the device
10 110. For example, if a device is configured to only operate one of a group of primary wakewords (e.g., only one of “Alexa,” “Computer,” “Echo”), the context data may indicate the primary wakeword to be active. The device 110 may then activate the decoder(s) 152 for that wakeword and deactivate the decoder(s) 152 for the non-active primary wakewords.

[0097] Context data may also include SDD result data 642 as described below where the
15 SDD result data 642 indicates the user is directing an input to the system. In response the device 110 may activate one or more keyword detector(s). In an inverse example, the SDD result data 642 may indicate the user is not directing an input to the system. In response the device 110 may deactivate one or more keyword detector(s).

[0098] Context data may also include a state of a device 110. For example, if a nearby
20 device is in a disabled or off state, another device may activate more decoder(s) to allow for detection of sound(s) the disabled / off device might have otherwise detected. In another example, state data may indicate a device is outputting content such as image and/or audio content. In such situations the device 110 (or another, for example, nearby device) may be configured to activate / deactivate certain decoder(s).

[0099] The device 110 may process (402) the context data to determine the operating context
25 of the device 110. For example, the device may process the context data to determine that a user is in the environment of the device, that an application is operating with respect to device 110 and/or the user, that a keyword was recently detected, that a dialog is in a certain state, etc. Context evaluation may happen in different time windows depending on system configuration. For example, the device 110 may determine the operation context every X minutes, seconds, etc.
30 on some regular schedule or may do so when prompted by an event or occurrence.

[0100] The device 110 may determine (404) that the operating context corresponds to first sound(s) (e.g., keyword(s), acoustic events, etc.) such as the sound(s) associated with a user, with an active application / skill, with a dialog, or the like. The device 110 may also determine (406) that the operating context does not correspond to second sound(s), that is that such second sound(s) are not related to the current operating context such that the device 110 need not expend computing resources detecting such second sound(s). The device may activate (408) a first one or more decoder(s) 152 corresponding to the first sound(s) and may deactivate (410) a second one or more decoder(s) 152 corresponding to the second sound(s).

[0101] The activation / deactivation may be done in different ways. In one configuration, each decoder(s) 152 may correspond to its own model that may be provided input data (e.g., encoded audio data 211). In this configuration the device 110 may provide the input data to the activated decoder(s) and may not provide the input data to the deactivated decoder(s). In another example, a plurality of decoder(s) that are enabled for a device 110 are combined into a single model where portions of the combined model correspond to different decoder(s) for different keyword(s). For example, a first portion of the combined model may correspond to a first decoder / keyword (for example a portion of a model graph) while a second portion of the combined model may correspond to a second decoder / keyword (for example a different portion of a model graph). The combined model may be configured to process the encoded audio data 211 to determine output data indicating the probability of whether one or more keywords was detected (e.g., decision data 252a, 252b...252n). To activate the first decoder(s) the device 100 may perform processing using the first portion of the combined model and to deactivate the second decoder(s) the device 110 may not perform processing using the second portion of the combined model.

[0102] In another example, deactivating a decoder 152 may simply instructing the wakeword manager 156 to ignore the output of that decoder, but this option may not meaningfully reduce the use of computing resources.

[0103] The device 110 may receive (172) audio data. Although illustrated as occurring after determining the active / inactive decoder(s), as noted above, the steps may occur in a different order and thus the device may receive (172) the audio data before determining the operating context, determining the active / inactive decoder(s), etc. The device may process (174) the

audio data using a first encoder (e.g., joint encoder 150) to determine encoded audio data 211. The device 110 may process (412) the encoded audio data 211 using the activated first decoder(s) and may refrain from processing (414) the encoded audio data 211 using the deactivated second decoder(s).

5 **[0104]** As can be appreciated, an assistant voice service may be used to access many different functions of a virtual assistant. As such, a particular assistant voice service 160 may correspond to language processing components, skill components, language output components, and/or associated components as described in further detail below with reference to FIGS. 5A and 5B. In various implementations, the components represented by the assistant voice services
10 160 may reside on the device 110 or the system component(s) 120, or divided or shared between the two. In some implementations, the assistant 1 voice service 160a may correspond to one or more components of a first system component(s) 120a and the assistant 2 voice service 160b may correspond to one or more components of a second system component(s) 120b, etc. In some implementations, multiple assistant voice services 160 may correspond to a single system
15 component(s) 120, where configuration for each virtual assistant are stored in a multi-assistant component and used to process a user command and/or request using settings and/or resources corresponding to the determined virtual assistant. In other implementations a first assistant voice service 160a may correspond to an entirely different system component(s) 120a from a second assistant voice service 160b, which may correspond to a system component(s) 120b. Thus, in
20 response to detection of a first wakeword associated with the first assistant voice service 160a, the device 110 may send audio data to a speech processing component associated with first system component(s) 120a and/or first assistant voice service 160a. But in response to detection of a second wakeword associated with the second assistant voice service 160b, the device 110 may send audio data to a speech processing component associated with second system
25 component(s) 120b and/or second assistant voice service 160b.

[0105] FIG. 5A is a conceptual diagram illustrating components that may be included in a first example implementation of the multi-assistant system 100, according to embodiments of the present disclosure. In the implementation of the system 100 shown in FIG. 5A, components corresponding to a first virtual assistant (e.g., the assistant 1 voice service 160a) may be divided
30 and/or shared between the device 110 and a first system component(s) 120a, components corresponding to a second virtual assistant may be divided and/or shared between the device 110

and a second system component(s) 120b, etc. The respective system component(s) 120 may be separate and distinct from each other. Data from the device 110 corresponding to a first virtual assistant (e.g., an utterance to be handled by the first virtual assistant) may be sent to the first system component(s) 120a, and data from the device corresponding to a second virtual assistant may be sent to the second system component(s) 120b, etc.

[0106] The system 100 may operate using various components as described in FIG. 5A. The various components may be located on same or different physical devices. Communication between various components may occur directly or across a network(s) 199. The device 110 may include audio capture component(s), such as a microphone or array of microphones of a device 110, captures audio 11 and creates corresponding audio data. Once speech is detected in audio data representing the audio 11, the device 110 may determine if the speech is directed at the device 110 / system component(s) 120. In at least some embodiments, such determination may be made using a wakeword detection component 122. The wakeword detection component 122 may be configured to detect various wakewords as described herein. In at least some examples, each wakeword may correspond to a name of a different digital assistant. For example, detection of the wakeword “Alexa” by the wakeword detector 122 may result in sending audio data to assistant 1 voice service 160a / system component(s) 120a for processing while detection of the wakeword “Computer” by the wakeword detector may result in sending audio data to assistant 2 voice service 160b / system component(s) 120b for processing. The system may have a separate wakeword and system for different skills / systems (e.g., “Dungeon Master” for a game play assistant 3 voice service 160c / system component(s) 120c) and/or such skills / systems may be coordinated by one or more skill(s) 590 of one or more system component(s) 120. In another example, input to the system may be in form of text data 513, for example as a result of a user typing an input into a user interface of device 110. Other input forms may include indication that the user has pressed a physical or virtual button on device 110, the user has made a gesture, etc.

[0107] For example, the device 110 / system component(s) 120 may use a system directed detector 585 to determine if a user input is directed at the system. The system directed input detector 585 may be configured to determine whether an input to the system (for example speech, a gesture, etc.) is directed to the system or not directed to the system (for example directed to another user, etc.). The system directed input detector 585 may work in conjunction

with the wakeword detector 122. If the system directed input detector 585 determines an input is directed to the system, the device 110 may “wake” and begin sending captured data for further processing (for example, processing audio data using the language processing 592, processing captured image data using an image processing component or the like). If data is being
5 processed the device 110 may indicate such to the user, for example by activating or changing the color of an illuminated output (such as a light emitting diode (LED) ring), displaying an indicator on a display (such as a light bar across the display), outputting an audio indicator (such as a beep) or otherwise informing a user that input data is being processed. If the system directed input detector 585 determines an input is not directed to the system (such as a speech or
10 gesture directed to another user) the device 110 may discard the data and take no further action for processing purposes. In this way the system 100 may prevent processing of data not directed to the system, thus protecting user privacy. As an indicator to the user, however, the system may output an audio, visual, or other indicator when the system directed input detector 585 is determining whether an input is potentially device directed. For example, the system may output
15 an orange indicator while considering an input, and may output a green indicator if a system directed input is detected. Other such configurations are possible. Further details regarding the system directed input detector 585 are included below with regard to FIG. 6.

[0108] Following detection of a wakeword, button press, or other indication to begin receiving input, the device 110 may “wake” and begin generating and processing audio data 111
20 representing the audio 11. The audio data 111 may include data corresponding to the wakeword; in other embodiments, the portion of the audio corresponding to the wakeword may be removed prior to downstream processing of the audio data 111 (e.g., ASR and/or NLU). In the case of touch input detection or gesture based input detection, the audio data may not include a wakeword.

[0109] Upon receipt by the system 100, the audio data 111 may be sent to an orchestrator component 530. The orchestrator component 530 may include memory and logic that enables the orchestrator component 530 to transmit various pieces and forms of data to various components of the system, as well as perform other operations as described herein.

[0110] The orchestrator component 530 may send the audio data 111 to a language
30 processing component 592. The language processing component 592 (sometimes also referred

to as a spoken language understanding (SLU) component) includes an automatic speech recognition (ASR) component 550 and a natural language understanding (NLU) component 560. The ASR component 550 may transcribe the audio data 111 into text data. The ASR component 550 may receive the audio data 111 in a streaming fashion; that is, the ASR component 550 may begin receiving and/or processing the audio data 111 as it is generated by the system 100 and without necessarily waiting for the user to stop speaking, release a button, or otherwise indicate an end of speech. The text data output by the ASR component 550 represents one or more than one (e.g., in the form of an N-best list) ASR hypotheses representing speech represented in the audio data 111. The ASR component 550 interprets the speech in the audio data 111 based on a similarity between the audio data 111 and pre-established language models. For example, the ASR component 550 may compare the audio data 111 with models for sounds (e.g., acoustic units such as phonemes, triphones, senons, phones, etc.) and sequences of sounds to identify words that match the sequence of sounds of the speech represented in the audio data 111. The ASR component 550 sends the text data generated thereby to an NLU component 560, via, in some embodiments, the orchestrator component 530. The text data sent from the ASR component 550 to the NLU component 560 may include a single top-scoring ASR hypothesis or may include an N-best list including multiple top-scoring ASR hypotheses. An N-best list may additionally include a respective score associated with each ASR hypothesis represented therein. The ASR component 550 is described in greater detail below with regard to FIG. 7.

[0111] The speech processing system 592 may further include a NLU component 560. The NLU component 560 may receive the text data from the ASR component. The NLU component 560 may attempts to make a semantic interpretation of the phrase(s) or statement(s) represented in the text data input therein by determining one or more meanings associated with the phrase(s) or statement(s) represented in the text data. The NLU component 560 may determine an intent representing an action that a user desires be performed and may determine information that allows a device (e.g., the device 110, the system component(s) 120, a skill component 590, a skill system component(s) 525, etc.) to execute the intent. For example, if the text data corresponds to “play the 5th Symphony by Beethoven,” the NLU component 560 may determine an intent that the system output music and may identify “Beethoven” as an artist/composer and “5th Symphony” as the piece of music to be played. For further example, if the text data corresponds to “what is the weather,” the NLU component 560 may determine an intent that the

system output weather information associated with a geographic location of the device 110. In another example, if the text data corresponds to “turn off the lights,” the NLU component 560 may determine an intent that the system turn off lights associated with the device 110 or the user 5. However, if the NLU component 560 is unable to resolve the entity—for example, because the entity is referred to by anaphora such as “this song” or “my next appointment”—the speech processing system 592 can send a decode request to another speech processing system 592 for information regarding the entity mention and/or other context related to the utterance. The speech processing system 592 may augment, correct, or base results data upon the audio data 111 as well as any data received from the other speech processing system 592.

10 **[0112]** The NLU component 560 may return NLU results data 985/925 (which may include tagged text data, indicators of intent, etc.) back to the orchestrator 530. The orchestrator 530 may forward the NLU results data to a skill component(s) 590. If the NLU results data includes a single NLU hypothesis, the NLU component 560 and the orchestrator component 530 may direct the NLU results data to the skill component(s) 590 associated with the NLU hypothesis. If 15 the NLU results data 985/925 includes an N-best list of NLU hypotheses, the NLU component 560 and the orchestrator component 530 may direct the top scoring NLU hypothesis to a skill component(s) 590 associated with the top scoring NLU hypothesis. The system may also include a post-NLU ranker (not shown) which may incorporate other information to rank potential interpretations determined by the NLU component 560. The NLU component 560, 20 post-NLU ranker and other components are described in greater detail below with regard to FIGS. 8 and 9.

[0113] A skill component may be software running on the system component(s) 120 that is akin to a software application. That is, a skill component 590 may enable the system component(s) 120 to execute specific functionality in order to provide data or produce some 25 other requested output. As used herein, a “skill component” may refer to software that may be placed on a machine or a virtual machine (e.g., software that may be launched in a virtual instance when called). A skill component may be software customized to perform one or more actions as indicated by a business entity, device manufacturer, user, etc. What is described herein as a skill component may be referred to using many different terms, such as an action, bot, 30 app, or the like. The system component(s) 120 may be configured with more than one skill component 590. For example, a weather service skill component may enable the system

component(s) 120 to provide weather information, a car service skill component may enable the system component(s) 120 to book a trip with respect to a taxi or ride sharing service, a restaurant skill component may enable the system component(s) 120 to order a pizza with respect to the restaurant's online ordering system, etc. A skill component 590 may operate in conjunction
5 between the system component(s) 120 and other devices, such as the device 110, in order to complete certain functions. Inputs to a skill component 590 may come from speech processing interactions or through other interactions or input sources. A skill component 590 may include hardware, software, firmware, or the like that may be dedicated to a particular skill component 590 or shared among different skill components 590.

10 **[0114]** A skill support system component(s) 525 may communicate with a skill component(s) 590 within the system component(s) 120 and/or directly with the orchestrator component 530 or with other components. A skill support system component(s) 525 may be configured to perform one or more actions. An ability to perform such action(s) may sometimes be referred to as a "skill." That is, a skill may enable a skill support system component(s) 525 to
15 execute specific functionality in order to provide data or perform some other action requested by a user. For example, a weather service skill may enable a skill support system component(s) 525 to provide weather information to the system component(s) 120, a car service skill may enable a skill support system component(s) 525 to book a trip with respect to a taxi or ride sharing service, an order pizza skill may enable a skill support system component(s) 525 to order a pizza
20 with respect to a restaurant's online ordering system, etc. Additional types of skills include home automation skills (e.g., skills that enable a user to control home devices such as lights, door locks, cameras, thermostats, etc.), entertainment device skills (e.g., skills that enable a user to control entertainment devices such as smart televisions), video skills, flash briefing skills, as well as custom skills that are not associated with any pre-configured type of skill.

25 **[0115]** The system component(s) 120 may be configured with a skill component 590 dedicated to interacting with the skill support system component(s) 525. Unless expressly stated otherwise, reference to a skill, skill device, or skill component may include a skill component 590 operated by the system component(s) 120 and/or skill operated by the skill support system component(s) 525. Moreover, the functionality described herein as a skill or skill may be
30 referred to using many different terms, such as an action, bot, app, or the like. The skill 590 and or skill support system component(s) 525 may return output data to the orchestrator 530.

[0116] The system component(s) 120 includes a language output component 593. The language output component 593 includes a natural language generation (NLG) component 579 and a text-to-speech (TTS) component 580. The NLG component 579 can generate text for purposes of TTS output to a user. For example, the NLG component 579 may generate text corresponding to instructions corresponding to a particular action for the user to perform. The NLG component 579 may generate appropriate text for various outputs as described herein. The NLG component 579 may include one or more trained models configured to output text appropriate for a particular input. The text output by the NLG component 579 may become input for the TTS component 580 (e.g., output text data 1010 discussed below). Alternatively or in addition, the TTS component 580 may receive text data from a skill 590 or other system component for output.

[0117] The NLG component 579 may include a trained model. The NLG component 579 generates text data 1010 from dialog data received (e.g., by a dialog manager) such that the output text data 1010 has a natural feel and, in some embodiments, includes words and/or phrases specifically formatted for a requesting individual. The NLG may use templates to formulate responses. And/or the NLG system may include models trained from the various templates for forming the output text data 1010. For example, the NLG system may analyze transcripts of local news programs, television shows, sporting events, or any other media program to obtain common components of a relevant language and/or region. As one illustrative example, the NLG system may analyze a transcription of a regional sports program to determine commonly used words or phrases for describing scores or other sporting news for a particular region. The NLG may further receive, as inputs, a dialog history, an indicator of a level of formality, and/or a command history or other user history such as the dialog history.

[0118] The NLG system may generate dialog data based on one or more response templates. Further continuing the example above, the NLG system may select a template in response to the question, "What is the weather currently like?" of the form: "The weather currently is \$weather_information\$." The NLG system may analyze the logical form of the template to produce one or more textual responses including markups and annotations to familiarize the response that is generated. In some embodiments, the NLG system may determine which response is the most appropriate response to be selected. The selection may, therefore, be based on past responses, past questions, a level of formality, and/or any other feature, or any other

combination thereof. Responsive audio data representing the response generated by the NLG system may then be generated using the text-to-speech component 580.

[0119] The TTS component 580 may generate audio data (e.g., synthesized speech) from text data using one or more different methods. Text data input to the TTS component 580 may come from a skill component 590, the orchestrator component 530, or another component of the system. In one method of synthesis called unit selection, the TTS component 580 matches text data against a database of recorded speech. The TTS component 580 selects matching units of recorded speech and concatenates the units together to form audio data. In another method of synthesis called parametric synthesis, the TTS component 580 varies parameters such as frequency, volume, and noise to create audio data including an artificial speech waveform. Parametric synthesis uses a computerized voice generator, sometimes called a vocoder.

[0120] The system 100 (either on device 110, system component(s) 120, or a combination thereof) may include profile storage for storing a variety of information related to individual users, groups of users, devices, etc. that interact with the system. As used herein, a “profile” refers to a set of data associated with a user, group of users, device, etc. The data of a profile may include preferences specific to the user, device, etc.; input and output capabilities of the device; internet connectivity information; user bibliographic information; subscription information, as well as other information.

[0121] The profile storage 570 may include one or more user profiles, with each user profile being associated with a different user identifier / user profile identifier. Each user profile may include various user identifying data. Each user profile may also include data corresponding to preferences of the user. Each user profile may also include preferences of the user and/or one or more device identifiers, representing one or more devices of the user. For instance, the user account may include one or more IP addresses, MAC addresses, and/or device identifiers, such as a serial number, of each additional electronic device associated with the identified user account. When a user logs into to an application installed on a device 110, the user profile (associated with the presented login information) may be updated to include information about the device 110, for example with an indication that the device is currently in use. Each user profile may include identifiers of skills that the user has enabled. When a user enables a skill, the user is providing the system component(s) 120 with permission to allow the skill to execute with

respect to the user's natural language user inputs. If a user does not enable a skill, the system component(s) 120 may not invoke the skill to execute with respect to the user's natural language user inputs.

[0122] The profile storage 570 may include one or more group profiles. Each group profile may be associated with a different group identifier. A group profile may be specific to a group of users. That is, a group profile may be associated with two or more individual user profiles. For example, a group profile may be a household profile that is associated with user profiles associated with multiple users of a single household. A group profile may include preferences shared by all the user profiles associated therewith. Each user profile associated with a group profile may additionally include preferences specific to the user associated therewith. That is, each user profile may include preferences unique from one or more other user profiles associated with the same group profile. A user profile may be a stand-alone profile or may be associated with a group profile.

[0123] The profile storage 570 may include one or more device profiles. Each device profile may be associated with a different device identifier. Each device profile may include various device identifying information. Each device profile may also include one or more user identifiers, representing one or more users associated with the device. For example, a household device's profile may include the user identifiers of users of the household.

[0124] FIG. 5B is a conceptual diagram illustrating components that may be included in a second example implementation of the multi-assistant system 100, according to embodiments of the present disclosure. In the implementation of the system 100 shown in FIG. 5B, components corresponding to multiple virtual assistants may be incorporated as part of a same system component(s) 120. For example, a system may allow access to different celebrity personality assistants (e.g., assistant 1 voice service 160a, assistant 2 voice service 160b, etc.) within the same system component(s) 120. Various operations of the voice assistants may be divided and/or shared between the device 110 and the system component(s) 120. Processing of data by components of the system component(s) 120 may be performed according to a determined virtual assistant using configuration data maintained by the multi-assistant component 515. For example, data received by the system component(s) 120 for processing may include an assistant identifier or other metadata that the components of the system component(s) 120 may use to

process the data in a manner consistent with the determined virtual assistant. Thus, the system component(s) 120 may perform language processing using models, entity libraries, etc.

corresponding to the determined virtual assistant, and may generate synthesized speech using voice parameters corresponding to the determined virtual assistant, etc. As can be appreciated,

5 configurations of FIG. 5A and 5B may be used together, for example with certain assistant voice services being part of a same system and certain assistant voice services being part of separate respective systems.

[0125] In some implementations, the system component(s) 120 may have and/or interface with skill components dedicated to a particular virtual assistant. For example, the device 110
10 and/or system component(s) 120 may include first assistant skills 591, which may include skill components 590 as previously described. The device 110 and/or system component(s) 120 may further include second assistant skills 592 including skill components 595a, 595b, and 595c (collectively, “skill components 592”). In some implementations, the skill components 592 may be the same as or similar to the skill components 590. In some implementations, the first
15 assistant skills 591 may provide different functionality than the second assistant skills. In some implementations, the first assistant skills 591 may be general-purpose skills (e.g., shopping, weather, music, etc.) not specific to a particular domain or device, while the second assistant skills 592 may be specialized to a certain domain, device, service, etc., such as for controlling operations of the smart TV 110g. Some of the skill components 590 may interface with one or
20 more skill support system component(s) 525; similarly, some of the skill components 592 may interface with one or more skill support system component(s) 526. In some implementations, the system 100 may have other skills and/or skill components that are common to more than one virtual assistant.

[0126] Other components of the system 100 illustrated in FIG. 5B, such as the language
25 processing components 592, language output components 593, profile storage 570, and/or orchestrator component 530 may be the same as or similar to the corresponding components as described with respect to FIG. 5A.

[0127] Configuration and operation of the system directed input detector 585 is illustrated in FIG. 6. As shown in FIG. 6, the system directed input detector 585 may include a number of
30 different components. First, the system directed input detector 585 may include a voice activity

detector (VAD) 620. The VAD 620 may operate to detect whether the incoming audio data 111 includes speech or not. The VAD output 621 may be a binary indicator. Thus, if the incoming audio data 111 includes speech, the VAD 620 may output an indicator 621 that the audio data 111 does includes speech (e.g., a 1) and if the incoming audio data 111 does not includes speech, the VAD 620 may output an indicator 621 that the audio data 111 does not includes speech (e.g., a 0). The VAD output 621 may also be a score (e.g., a number between 0 and 1) corresponding to a likelihood that the audio data 111 includes speech. The VAD 620 may also perform start-point detection as well as end-point detection where the VAD 620 determines when speech starts in the audio data 111 and when it ends in the audio data 111. Thus the VAD output 621 may also include indicators of a speech start point and/or a speech endpoint for use by other components of the system. (For example, the start-point and end-points may demarcate the audio data 111 that is sent to the speech processing component 240.) The VAD output 621 may be associated with a same unique ID as the audio data 111 for purposes of tracking system processing across various components.

[0128] The VAD 620 may operate using a variety of VAD techniques, including those described above with regard to VAD operations performed by device 110. The VAD may be configured to be robust to background noise so as to accurately detect when audio data actually includes speech or not. The VAD 620 may operate on raw audio data 111 such as that sent by device 110 or may operate on feature vectors or other data representing the audio data 111. For example, the VAD 620 may take the form of a deep neural network (DNN) and may operate on a single feature vector representing the entirety of audio data 111 received from the device or may operate on multiple feature vectors, for example feature vectors representing frames of audio data where each frame covers a certain amount of time of audio data (e.g., 25 ms). The VAD 620 may also operate on other data 681 that may be useful in detecting voice activity in the audio data 111. For example, the other data 681 may include results of anchored speech detection where the system takes a representation (such as a voice fingerprint, reference feature vector, etc.) of a reference section of speech (such as speech of a voice that uttered a previous command to the system that included a wakeword) and compares a voice detected in the audio data 111 to determine if that voice matches a voice in the reference section of speech. If the voices match, that may be an indicator to the VAD 620 that speech was detected. If not, that may be an indicator to the VAD 620 that speech was not detected. (For example, a representation may be

taken of voice data in the first input audio data which may then be compared to the second input audio data to see if the voices match. If they do (or do not) that information may be considered by the VAD 620.) The VAD 620 may also consider other data when determining if speech was detected. The VAD 620 may also consider speaker ID information (such as may be output by a user recognition component), directionality data that may indicate what direction (relative to the capture device 110) the incoming audio was received from. Such directionality data may be received from the device 110 and may have been determined by a beamformer or other component of device 110. The VAD 620 may also consider data regarding a previous utterance which may indicate whether the further audio data received by the system is likely to include speech. Other VAD techniques may also be used.

[0129] If the VAD output 621 indicates that no speech was detected the system (through orchestrator 530 or some other component) may discontinue processing with regard to the audio data 111, thus saving computing resources that might otherwise have been spent on other processes (e.g., ASR for the audio data 111, etc.). If the VAD output 621 indicates that speech was detected, the system may make a determination as to whether the speech was or was not directed to the speech-processing system. Such a determination may be made by the system directed audio detector 640. The system directed audio detector 640 may include a trained model, such as a DNN, that operates on a feature vector which represent certain data that may be useful in determining whether or not speech is directed to the system. To create the feature vector operable by the system directed audio detector 640, a feature extractor 630 may be used. The feature extractor 630 may input ASR results 710 which include results from the processing of the audio data 111 by the speech recognition component 550. For privacy protection purposes, in certain configurations the ASR results 710 may be obtained from a language processing component 592 / ASR component 550 located on device 110 or on a home remote component as opposed to a language processing component 592 / ASR component 550 located on a cloud or other remote system component(s) 120 so that audio data 111 is not sent remote from the user's home unless the system directed input detector component 585 has determined that the input is system directed. Though this may be adjusted depending on user preferences / system configuration.

[0130] The ASR results 710 may include an N-best list of top scoring ASR hypotheses and their corresponding scores, portions (or all of) an ASR lattice/trellis with scores, portions (or all

of) an ASR search graph with scores, portions (or all of) an ASR confusion network with scores, or other such ASR output. As an example, the ASR results 710 may include a trellis, which may include a raw search graph as scored during ASR decoding. The ASR results 710 may also include a lattice, which may be a trellis as scored that has been pruned to remove certain hypotheses that do not exceed a score threshold or number of hypotheses threshold. The ASR results 710 may also include a confusion network where paths from the lattice have been merged (e.g., merging hypotheses that may share all or a portion of a same word). The confusion network may be a data structure corresponding to a linear graph that may be used as an alternate representation of the most likely hypotheses of the decoder lattice. The ASR results 710 may also include corresponding respective scores (such as for a trellis, lattice, confusion network, individual hypothesis, N-best list, etc.)

[0131] The ASR results 710 (or other data 691) may include other ASR result related data such as other features from the ASR system or data determined by another component. For example, the system may determine an entropy of the ASR results (for example a trellis entropy or the like) that indicates a how spread apart the probability mass of the trellis is among the alternate hypotheses. A large entropy (e.g., large spread of probability mass over many hypotheses) may indicate the ASR component 550 being less confident about its best hypothesis, which in turn may correlate to detected speech not being device directed. The entropy may be a feature included in other data 691 to be considered by the system directed audio detector 640.

[0132] The system may also determine and consider ASR decoding costs, which may include features from Viterbi decoding costs of the ASR. Such features may indicate how well the input acoustics and vocabulary match with the acoustic models 753 and language models 754. Higher Viterbi costs may indicate greater mismatch between the model and the given data, which may correlate to detected speech not being device directed. Confusion network feature may also be used. For example, an average number of arcs (where each arc represents a word) from a particular node (representing a potential join between two words) may measure how many competing hypotheses there are in the confusion network. A large number of competing hypotheses may indicate that the ASR component 550 is less confident about the top hypothesis, which may correlate to detected speech not being device directed. Other such features or data from the ASR results 710 may also be used as other data 691.

[0133] The ASR results 710 may be represented in a system directed detector (SDD) feature vector 631 that can be used to determine whether speech was system-directed. The feature vector 631 may represent the ASR results 710 but may also represent audio data 111 (which may be input to feature extractor 630) or other information. Such ASR results may be helpful in determining if speech was system-directed. For example, if ASR results include a high scoring single hypothesis, that may indicate that the speech represented in the audio data 111 is directed at, and intended for, the device 110. If, however, ASR results do not include a single high scoring hypothesis, but rather many lower scoring hypotheses, that may indicate some confusion on the part of the speech recognition component 550 and may also indicate that the speech represented in the audio data 111 was not directed at, nor intended for, the device 110.

[0134] The ASR results 710 may include complete ASR results, for example ASR results corresponding to all speech between a startpoint and endpoint (such as a complete lattice, etc.). In this configuration the system may wait until all ASR processing for a certain input audio has been completed before operating the feature extractor 630 and system directed audio detector 640. Thus the system directed audio detector 640 may receive a feature vector 631 that includes all the representations of the audio data 111 created by the feature extractor 630. The system directed audio detector 640 may then operate a trained model (such as a DNN) on the feature vector 631 to determine a score corresponding to a likelihood that the audio data 111 includes a representation of system-directed speech. If the score is above a threshold, the system directed audio detector 640 may determine that the audio data 111 does include a representation of system-directed speech. The SDD result 642 may include an indicator of whether the audio data includes system-directed speech, a score, and/or some other data.

[0135] The ASR results 710 may also include incomplete ASR results, for example ASR results corresponding to only some speech between a between a startpoint and endpoint (such as an incomplete lattice, etc.). In this configuration the feature extractor 630 / system directed audio detector 640 may be configured to operate on incomplete ASR results 710 and thus the system directed audio detector 640 may be configured to output an SSD result 642 that provides an indication as to whether the portion of audio data processed (that corresponds to the incomplete ASR results) corresponds to system directed speech. The system may thus be configured to perform ASR at least partially in parallel with the system directed audio detector 640 to process ASR result data as it is ready and thus continually update an SDD result 642.

Once the system directed input detector 585 has processed enough ASR results and/or the SDD result 642 exceeds a threshold, the system may determine that the audio data 111 includes system-directed speech. Similarly, once the system directed input detector 585 has processed enough ASR results and/or the SDD result 642 drops below another threshold, the system may
5 determine that the audio data 111 does not include system-directed speech.

[0136] The SDD result 642 may be associated with a same unique ID as the audio data 111 and VAD output 621 for purposes of tracking system processing across various components.

[0137] The feature extractor 630 may also incorporate in a feature vector 631 representations of other data 691. Other data 691 may include, for example, word embeddings from words
10 output by the speech recognition component 550 may be considered. Word embeddings are vector representations of words or sequences of words that show how specific words may be used relative to other words, such as in a large text corpus. A word embedding may be of a different length depending on how many words are in a text segment represented by the word embedding. For purposes of the feature extractor 630 processing and representing a word
15 embedding in a feature vector 631 (which may be of a fixed length), a word embedding of unknown length may be processed by a neural network with memory, such as an LSTM (long short term memory) network. Each vector of a word embedding may be processed by the LSTM which may then output a fixed representation of the input word embedding vectors.

[0138] Other data 691 may also include, for example, NLU output from the natural language
20 560 component may be considered. Thus, if natural language output data 985/925 indicates a high correlation between the audio data 111 and an out-of-domain indication (e.g., no intent classifier scores from ICs 864 or overall domain scores from recognizers 863 reach a certain confidence threshold), this may indicate that the audio data 111 does not include system-directed speech. Other data 691 may also include, for example, an indicator of a user/speaker as output
25 by a user recognition component (not shown). Thus, for example, if the user recognition component does not indicate the presence of a known user, or indicates the presence of a user associated with audio data 111 that was not associated with a previous utterance, this may indicate that the audio data 111 does not include system-directed speech. The other data 691 may also include an indication that a voice represented in audio data 111 is the same (or
30 different) as the voice detected in previous input audio data corresponding to a previous utterance. The other data 691 may also include directionality data, for example using

beamforming or other audio processing techniques to determine a direction / location of a source of detected speech and whether that source direction / location matches a speaking user. The other data 691 may also include data indicating that a direction of a user's speech is toward a device 110 or away from a device 110, which may indicate whether the speech was system directed or not.

[0139] Other data 691 may also include image data 623. For example, if image data is detected from one or more devices that are nearby to the device 110 (which may include the device 110 itself) that captured the audio data being processed using the system directed input detector (585), the image data may be processed to determine whether a user is facing an audio capture device for purposes of determining whether speech is system-directed as further explained below.

[0140] Other data 691 may also include dialog history data. For example, the other data 691 may include information about whether a speaker has changed from a previous utterance to the current audio data 111, whether a topic of conversation has changed from a previous utterance to the current audio data, how NLU results from a previous utterance compare to NLU results obtained using the current audio data 111, other system context information. The other data 691 may also include an indicator as to whether the audio data 111 was received as a result of a wake command or whether the audio data 111 was sent without the device 110 detecting a wake command (e.g., the device 110 being instructed by remote system component(s) 120 and/or determining to send the audio data without first detecting a wake command).

[0141] Other data 691 may also include information from the user profile 570.

[0142] Other data 691 may also include direction data, for example data regarding a direction of arrival of speech detected by the device, for example a beam index number, angle data, or the like. If second audio data is received from a different direction than first audio data, then the system may be less likely to declare the second audio data to include system-directed speech since it is originating from a different location.

[0143] Other data 691 may also include acoustic feature data such as pitch, prosody, intonation, volume, or other data descriptive of the speech in the audio data 111. As a user may use a different vocal tone to speak with a machine than with another human, acoustic feature information may be useful in determining if speech is device-directed.

[0144] Other data 691 may also include an indicator that indicates whether the audio data 111 includes a wakeword. For example, if a device 110 detects a wakeword prior to sending the audio data 111 to the remote system component(s) 120, the device 110 may send along an indicator that the device 110 detected a wakeword in the audio data 111. In another example, the remote system component(s) 120 may include another component that processes incoming audio data 111 to determine if it includes a wakeword. If it does, the component may create an indicator indicating that the audio data 111 includes a wakeword. The indicator may then be included in other data 691 to be incorporated in the feature vector 631 and/or otherwise considered by the system directed audio detector 640.

[0145] Other data 691 may also include device history data such as information about previous operations related to the device 110 that sent the audio data 111. For example, the other data 691 may include information about a previous utterance that was just executed, where the utterance originated with the same device 110 as a current utterance and the previous utterance was within a certain time window of the current utterance. Device history data may be stored in a manner associated with the device identifier (which may also be included in other data 691), which may also be used to track other information about the device, such as device hardware, capability, location, etc.

[0146] The other data 681 used by the VAD 620 may include similar data and/or different data from the other data 691 used by the feature extractor 630. The other data 681 / 691 may thus include a variety of data corresponding to input audio from a previous utterance. That data may include acoustic data from a previous utterance, speaker ID / voice identification data from a previous utterance, information about the time between a previous utterance and a current utterance, or a variety of other data described herein taken from a previous utterance. A score threshold (for the system directed audio detector 640 and/or the VAD 620) may be based on the data from the previous utterance. For example, a score threshold (for the system directed audio detector 640 and/or the VAD 620) may be based on acoustic data from a previous utterance.

[0147] The feature extractor 630 may output a single feature vector 631 for one utterance / instance of input audio data 211. The feature vector 631 may consistently be a fixed length, or may be a variable length vector depending on the relevant data available for particular audio data 111. Thus, the system directed audio detector 640 may output a single SDD result 642 per utterance / instance of input audio data 211. The SDD result 642 may be a binary indicator.

Thus, if the incoming audio data 111 includes system-directed speech, the system directed audio detector 640 may output an indicator 642 that the audio data 111 does includes system-directed speech (e.g., a 1) and if the incoming audio data 111 does not includes system-directed speech, the system directed audio detector 640 may output an indicator 642 that the audio data 111 does not system-directed includes speech (e.g., a 0). The SDD result 642 may also be a score (e.g., a number between 0 and 1) corresponding to a likelihood that the audio data 111 includes system-directed speech. Although not illustrated in FIG. 6, the flow of data to and from the system directed input detector 585 may be managed by the orchestrator 530 or by one or more other components.

[0148] The trained model(s) of the system directed audio detector 640 may be trained on many different examples of SDD feature vectors that include both positive and negative training samples (e.g., samples that both represent system-directed speech and non-system directed speech) so that the DNN and/or other trained model of the system directed audio detector 640 may be capable of robustly detecting when speech is system-directed versus when speech is not system-directed.

[0149] A further input to the system directed input detector 585 may include output data from TTS component 580 to avoid synthesized speech output by the system being confused as system-directed speech spoken by a user. The output from the TTS component 580 may allow the system to ignore synthesized speech in its considerations of whether speech was system directed. The output from the TTS component 580 may also allow the system to determine whether a user captured utterance is responsive to the TTS output, thus improving system operation.

[0150] The system directed input detector 585 may also use echo return loss enhancement (ERLE) and/or acoustic echo cancellation (AEC) data to avoid processing of audio data generated by the system.

[0151] As shown in FIG. 6, the system directed input detector 585 may simply user audio data to determine whether an input is system directed (for example, system directed audio detector 640 may output an SDD result 642). This may be true particularly when no image data is available (for example for a device without a camera). If image data 623 is available, however, the system may also be configured to use image data 623 to determine if an input is system directed. The image data 623 may include image data captured by device 110 and/or

image data captured by other device(s) in the environment of device 110. The audio data 111, image data 623 and other data 681 may be timestamped or otherwise correlated so that the system directed input detector 585 may determine that the data being analyzed all relates to a same time window so as to ensure alignment of data considered with regard to whether a particular input is system directed. For example, the system directed input detector 585 may determine system directedness scores for every frame of audio data / every image of a video stream and may align and/or window them to determine a single overall score for a particular input that corresponds to a group of audio frames / images.

[0152] Image data 623 along with other data 681 may be received by feature extractor 635.

The feature extractor may create one or more feature vectors 636 which may represent the image data 623 / other data 681. In certain examples, other data 681 may include data from image processing component which may include information about faces, gesture, etc. detected in the image data 623. For privacy protection purposes, in certain configurations any image processing / results thereof may be obtained from an image processing component located on device 110 or on a home remote component as opposed to an image processing component located on a cloud or other remote system component(s) 120 so that image data 623 is not sent remote from the user's home unless the system directed input detector component 585 has determined that the input is system directed. Though this may be adjusted depending on user preferences / system configuration.

[0153] The feature vector 636 may be passed to the user detector 625. The user detector 625 (which may use various components / operations of image processing component, user recognition component, etc.) may be configured to process image data 623 and/or feature vector 636 to determine information about the user's behavior which in turn may be used to determine if an input is system directed. For example, the user detector 625 may be configured to determine the user's position / behavior with respect to device 110 / system 100. The user detector 625 may also be configured to determine whether a user's mouth is opening / closing in a manner that suggests the user is speaking. The user detector 625 may also be configured to determine whether a user is nodding or shaking his/her head. The user detector 625 may also be configured to determine whether a user's gaze is directed to the device 110, to another user, or to another object. For example, the use detector 625 may include, or be configured to use data from, a gaze detector. The user detector 625 may also be configured to determine gestures of the

user such as a shoulder shrug, pointing toward an object, a wave, a hand up to indicate an instruction to stop, or a fingers moving to indicate an instruction to continue, holding up a certain number of fingers, putting a thumb up, etc. The user detector 625 may also be configured to determine a user's position / orientation such as facing another user, facing the device 110, whether their back is turned, etc. The user detector 625 may also be configured to determine relative positions of multiple users that appear in image data (and/or are speaking in audio data 111 which may also be considered by the user detector 625 along with feature vector 631), for example which users are closer to a device 110 and which are farther away. The user detector 625 (and/or other component) may also be configured to identify other objects represented in image data and determine whether objects are relevant to a dialog or system interaction (for example determining if a user is referring to an object through a movement or speech).

[0154] The user detector 625 may operate one or more models (e.g., one or more classifiers) to determine if certain situations are represented in the image data 623. For example, the user detector 625 may employ a visual directedness classifier that may determine, for each face detected in the image data 623 whether that face is looking at the device 110 or not. For example, a light-weight convolutional neural network (CNN) may be used which takes a face image cropped from the result of the face detector as input and output a [0,1] score of how likely the face is directed to the camera or not. Another technique may include to determine a three-dimensional (3D) landmark of each face, estimate the 3D angle of the face and predict a directness score based on the 3D angle.

[0155] The user detector 625 (or other component(s) such as those in an image processing) may be configured to track a face in image data to determine which faces represented may belong to a same person. The system may use IOU based tracker, a mean-shift based tracker, a particle filter based tracker or other technique.

[0156] The user detector 625 (or other component(s) such as those in a user recognition component) may be configured to determine whether a face represented in image data belongs to a person who is speaking or not, thus performing active speaker detection. The system may take the output from the face tracker and aggregate a sequence of face from the same person as input and predict whether this person is speaking or not. Lip motion, user ID, detected voice data, and other data may be used to determine whether a user is speaking or not.

[0157] The system directed image detector 650 may then determine, based on information from the user detector 625 as based on the image data whether an input relating to the image data is system directed. The system directed image detector 650 may also operate on other input data, for example image data including raw image data 623, image data including feature data 636 based on raw image data, other data 681, or other data. The determination by the system directed image detector 650 may result in a score indicating whether the input is system directed based on the image data. If no audio data is available, the indication may be output as SDD result 642. If audio data is available, the indication may be sent to system directed detector 670 which may consider information from both system directed audio detector 640 and system directed image detector 650. The system directed detector 670 may then process the data from both system directed audio detector 640 and system directed image detector 650 to come up with an overall determination as to whether an input was system directed, which may be output as SDD result 642. The system directed detector 670 may consider not only data output from system directed audio detector 640 and system directed image detector 650 but also other data/metadata corresponding to the input (for example, image data / feature data 636, audio data / feature data 631, image data 623, audio data 111, or the like discussed with regard to FIG. 6. The system directed detector 670 may include one or more models which may analyze the various input data to make a determination regarding SDD result 642.

[0158] In one example the determination of the system directed detector 670 may be based on “AND” logic, for example determining an input is system directed only if affirmative data is received from both system directed audio detector 640 and system directed image detector 650. In another example the determination of the system directed detector 670 may be based on “OR” logic, for example determining an input is system directed if affirmative data is received from either system directed audio detector 640 or system directed image detector 650. In another example the data received from system directed audio detector 640 and system directed image detector 650 are weighted individually based on other information available to system directed detector 670 to determine to what extend audio and/or image data should impact the decision of whether an input is system directed.

[0159] The system directed input detector 585 may also receive information from a wakeword component 122. For example, an indication that a wakeword was detected (e.g., WW data 260) may be considered by the system directed input detector 585 (e.g., by system directed

audio detector 640, system directed detector 670, etc.) as part of the overall consideration of whether a system input was device directed. Detection of a wakeword may be considered a strong signal that a particular input was device directed.

[0160] If an input is determined to be system directed, the data related to the input may be sent to downstream components for further processing (e.g., to language processing 592). If an input is determined not to be system directed, the system may take no further action regarding the data related to the input and may allow it to be deleted. In certain configurations, to maintain privacy, the operations to determine whether an input is system directed are performed by device 110 (or home server(s) 120) and only if the input is determined to be system directed is further data (such as audio data 111 or image data 623) sent to a remote system component(s) 120 that is outside a user's home or other direct control.

[0161] FIG. 7 is a conceptual diagram of an ASR component 550, according to embodiments of the present disclosure. The ASR component 550 may interpret a spoken natural language input based on the similarity between the spoken natural language input and pre-established language models 754 stored in an ASR model storage 752. For example, the ASR component 550 may compare the audio data with models for sounds (e.g., subword units or phonemes) and sequences of sounds to identify words that match the sequence of sounds spoken in the natural language input. Alternatively, the ASR component 550 may use a finite state transducer (FST) 755 to implement the language model functions.

[0162] When the ASR component 550 generates more than one ASR hypothesis for a single spoken natural language input, each ASR hypothesis may be assigned a score (e.g., probability score, confidence score, etc.) representing a likelihood that the corresponding ASR hypothesis matches the spoken natural language input (e.g., representing a likelihood that a particular set of words matches those spoken in the natural language input). The score may be based on a number of factors including, for example, the similarity of the sound in the spoken natural language input to models for language sounds (e.g., an acoustic model 753 stored in the ASR model storage 752), and the likelihood that a particular word, which matches the sounds, would be included in the sentence at the specific location (e.g., using a language or grammar model 754). Based on the considered factors and the assigned confidence score, the ASR component 550 may output an ASR hypothesis that most likely matches the spoken natural language input,

or may output multiple ASR hypotheses in the form of a lattice or an N-best list, with each ASR hypothesis corresponding to a respective score.

[0163] The ASR component 550 may include a speech recognition engine 758. The ASR component 550 receives audio data 111 (for example, received from a local device 110 having processed audio detected by a microphone by an acoustic front end (AFE) or other component). The speech recognition engine 758 compares the audio data 111 with acoustic models 753, language models 754, FST(s) 755, and/or other data models and information for recognizing the speech conveyed in the audio data. The audio data 111 may be audio data that has been digitized (for example by an AFE) into frames representing time intervals for which the AFE determines a number of values, called features, representing the qualities of the audio data, along with a set of those values, called a feature vector, representing the features/qualities of the audio data within the frame. In at least some embodiments, audio frames may be 10ms each. Many different features may be determined, as known in the art, and each feature may represent some quality of the audio that may be useful for ASR processing. A number of approaches may be used by an AFE to process the audio data, such as mel-frequency cepstral coefficients (MFCCs), perceptual linear predictive (PLP) techniques, neural network feature vector techniques, linear discriminant analysis, semi-tied covariance matrices, or other approaches known to those of skill in the art. In some cases, feature vectors of the audio data may arrive at the supporting system component(s) 120 encoded, in which case they may be decoded by the speech recognition engine 758 and/or prior to processing by the speech recognition engine 758.

[0164] In some implementations, the ASR component 550 may process the audio data 111 using the ASR model 750. The ASR model 750 may be, for example, a recurrent neural network such as an RNN-T. An example RNN-T architecture is illustrated in FIG. 7. The ASR model 750 may predict a probability ($y|x$) of labels $y = (y_1, \dots, y_u)$ given acoustic features $x = (x_1, \dots, x_t)$. During inference, the ASR model 750 can generate an N-best list using, for example, a beam search decoding algorithm. The ASR model 750 may include an encoder 712, a prediction network 720, a joint network 730, and a softmax 740. The encoder 712 may be similar or analogous to an acoustic model (e.g., similar to the acoustic model 753 described below), and may process a sequence of acoustic input features to generate encoded hidden representations. The prediction network 720 may be similar or analogous to a language model (e.g., similar to the language model 754 described below), and may process the previous output label predictions,

and map them to corresponding hidden representations. The joint network 730 may be, for example, a feed forward neural network (NN) that may process hidden representations from both the encoder 712 and prediction network 720, and predict output label probabilities. The softmax 740 may be a function implemented (e.g., as a layer of the joint network 730) to normalize the predicted output probabilities.

[0165] The speech recognition engine 758 may process the audio data 111 with reference to information stored in the ASR model storage 752. Feature vectors of the audio data 111 may arrive at the system component(s) encoded, in which case they may be decoded prior to processing by the speech recognition engine 758.

[0166] The speech recognition engine 758 attempts to match received feature vectors to language acoustic units (e.g., phonemes) and words as known in the stored acoustic models 753, language models 754, and FST(s) 755. For example, audio data 111 may be processed by one or more acoustic model(s) 753 to determine acoustic unit data. The acoustic unit data may include indicators of acoustic units detected in the audio data 111 by the ASR component 550. For example, acoustic units can consist of one or more of phonemes, diaphonemes, tonemes, phones, diphones, triphones, or the like. The acoustic unit data can be represented using one or a series of symbols from a phonetic alphabet such as the X-SAMPA, the International Phonetic Alphabet, or Initial Teaching Alphabet (ITA) phonetic alphabets. In some implementations a phoneme representation of the audio data can be analyzed using an n-gram based tokenizer. An entity, or a slot representing one or more entities, can be represented by a series of n-grams.

[0167] The acoustic unit data may be processed using the language model 754 (and/or using FST 755) to determine ASR data 710. The ASR data 710 can include one or more hypotheses. One or more of the hypotheses represented in the ASR data 710 may then be sent to further components (such as the NLU component 560) for further processing as discussed herein. The ASR data 710 may include representations of text of an utterance, such as words, subword units, or the like.

[0168] The speech recognition engine 758 computes scores for the feature vectors based on acoustic information and language information. The acoustic information (such as identifiers for acoustic units and/or corresponding scores) is used to calculate an acoustic score representing a likelihood that the intended sound represented by a group of feature vectors matches a language

phoneme. The language information is used to adjust the acoustic score by considering what sounds and/or words are used in context with each other, thereby improving the likelihood that the ASR component 550 will output ASR hypotheses that make sense grammatically. The specific models used may be general models or may be models corresponding to a particular domain, such as music, banking, etc.

[0169] The speech recognition engine 758 may use a number of techniques to match feature vectors to phonemes, for example using Hidden Markov Models (HMMs) to determine probabilities that feature vectors may match phonemes. Sounds received may be represented as paths between states of the HMM and multiple paths may represent multiple possible text matches for the same sound. Further techniques, such as using FSTs, may also be used.

[0170] The speech recognition engine 758 may use the acoustic model(s) 753 to attempt to match received audio feature vectors to words or subword acoustic units. An acoustic unit may be a senone, phoneme, phoneme in context, syllable, part of a syllable, syllable in context, or any other such portion of a word. The speech recognition engine 758 computes recognition scores for the feature vectors based on acoustic information and language information. The acoustic information is used to calculate an acoustic score representing a likelihood that the intended sound represented by a group of feature vectors match a subword unit. The language information is used to adjust the acoustic score by considering what sounds and/or words are used in context with each other, thereby improving the likelihood that the ASR component 550 outputs ASR hypotheses that make sense grammatically.

[0171] The speech recognition engine 758 may use a number of techniques to match feature vectors to phonemes or other acoustic units, such as diphones, triphones, etc. One common technique is using Hidden Markov Models (HMMs). HMMs are used to determine probabilities that feature vectors may match phonemes. Using HMMs, a number of states are presented, in which the states together represent a potential phoneme (or other acoustic unit, such as a triphone) and each state is associated with a model, such as a Gaussian mixture model or a deep belief network. Transitions between states may also have an associated probability, representing a likelihood that a current state may be reached from a previous state. Sounds received may be represented as paths between states of the HMM and multiple paths may represent multiple possible text matches for the same sound. Each phoneme may be represented by multiple

potential states corresponding to different known pronunciations of the phonemes and their parts (such as the beginning, middle, and end of a spoken language sound). An initial determination of a probability of a potential phoneme may be associated with one state. As new feature vectors are processed by the speech recognition engine 758, the state may change or stay the same, based on the processing of the new feature vectors. A Viterbi algorithm may be used to find the most likely sequence of states based on the processed feature vectors.

[0172] The probable phonemes and related states / state transitions, for example HMM states, may be formed into paths traversing a lattice of potential phonemes. Each path represents a progression of phonemes that potentially match the audio data represented by the feature vectors. One path may overlap with one or more other paths depending on the recognition scores calculated for each phoneme. Certain probabilities are associated with each transition from state to state. A cumulative path score may also be calculated for each path. This process of determining scores based on the feature vectors may be called acoustic modeling. When combining scores as part of the ASR processing, scores may be multiplied together (or combined in other ways) to reach a desired combined score or probabilities may be converted to the log domain and added to assist processing.

[0173] The speech recognition engine 758 may also compute scores of branches of the paths based on language models or grammars. Language modeling involves determining scores for what words are likely to be used together to form coherent words and sentences. Application of a language model may improve the likelihood that the ASR component 550 correctly interprets the speech contained in the audio data. For example, for an input audio sounding like “hello,” acoustic model processing that returns the potential phoneme paths of “H E L O”, “H A L O”, and “Y E L O” may be adjusted by a language model to adjust the recognition scores of “H E L O” (interpreted as the word “hello”), “H A L O” (interpreted as the word “halo”), and “Y E L O” (interpreted as the word “yellow”) based on the language context of each word within the spoken utterance.

[0174] FIGS. 8 and 9 illustrates how the NLU component 560 may perform NLU processing. FIG. 8 is a conceptual diagram of how natural language processing is performed, according to embodiments of the present disclosure. And FIG. 9 is a conceptual diagram of how natural language processing is performed, according to embodiments of the present disclosure.

[0175] FIG. 8 illustrates how NLU processing is performed on text data. The NLU component 560 may process text data including several ASR hypotheses of a single user input. For example, if the ASR component 550 outputs text data including an n-best list of ASR hypotheses, the NLU component 560 may process the text data with respect to all (or a portion of) the ASR hypotheses represented therein.

[0176] The NLU component 560 may annotate text data by parsing and/or tagging the text data. For example, for the text data “tell me the weather for Seattle,” the NLU component 560 may tag “tell me the weather for Seattle” as an <OutputWeather> intent as well as separately tag “Seattle” as a location for the weather information.

[0177] The NLU component 560 may include a shortlister component 850. The shortlister component 850 selects skills that may execute with respect to ASR output data 710 input to the NLU component 560 (e.g., applications that may execute with respect to the user input). The ASR output data 710 (which may also be referred to as ASR data 710) may include representations of text of an utterance, such as words, subword units, or the like. The shortlister component 850 thus limits downstream, more resource intensive NLU processes to being performed with respect to skills that may execute with respect to the user input.

[0178] Without a shortlister component 850, the NLU component 560 may process ASR output data 710 input thereto with respect to every skill of the system, either in parallel, in series, or using some combination thereof. By implementing a shortlister component 850, the NLU component 560 may process ASR output data 710 with respect to only the skills that may execute with respect to the user input. This reduces total compute power and latency attributed to NLU processing.

[0179] The shortlister component 850 may include one or more trained models. The model(s) may be trained to recognize various forms of user inputs that may be received by the system component(s) 120. For example, during a training period skill system component(s) 525 associated with a skill may provide the system component(s) 120 with training text data representing sample user inputs that may be provided by a user to invoke the skill. For example, for a ride sharing skill, a skill system component(s) 525 associated with the ride sharing skill may provide the system component(s) 120 with training text data including text corresponding to “get me a cab to [location],” “get me a ride to [location],” “book me a cab to [location],” “book

me a ride to [location],” etc. The one or more trained models that will be used by the shortlister component 850 may be trained, using the training text data representing sample user inputs, to determine other potentially related user input structures that users may try to use to invoke the particular skill. During training, the system component(s) 120 may solicit the skill system component(s) 525 associated with the skill regarding whether the determined other user input structures are permissible, from the perspective of the skill system component(s) 525, to be used to invoke the skill. The alternate user input structures may be derived by one or more trained models during model training and/or may be based on user input structures provided by different skills. The skill system component(s) 525 associated with a particular skill may also provide the system component(s) 120 with training text data indicating grammar and annotations. The system component(s) 120 may use the training text data representing the sample user inputs, the determined related user input(s), the grammar, and the annotations to train a model(s) that indicates when a user input is likely to be directed to / handled by a skill, based at least in part on the structure of the user input. Each trained model of the shortlister component 850 may be trained with respect to a different skill. Alternatively, the shortlister component 850 may use one trained model per domain, such as one trained model for skills associated with a weather domain, one trained model for skills associated with a ride sharing domain, etc.

[0180] The system component(s) 120 may use the sample user inputs provided by a skill system component(s) 525, and related sample user inputs potentially determined during training, as binary examples to train a model associated with a skill associated with the skill system component(s) 525. The model associated with the particular skill may then be operated at runtime by the shortlister component 850. For example, some sample user inputs may be positive examples (e.g., user inputs that may be used to invoke the skill). Other sample user inputs may be negative examples (e.g., user inputs that may not be used to invoke the skill).

[0181] As described above, the shortlister component 850 may include a different trained model for each skill of the system, a different trained model for each domain, or some other combination of trained model(s). For example, the shortlister component 850 may alternatively include a single model. The single model may include a portion trained with respect to characteristics (e.g., semantic characteristics) shared by all skills of the system. The single model may also include skill-specific portions, with each skill-specific portion being trained with respect to a specific skill of the system. Implementing a single model with skill-specific portions

may result in less latency than implementing a different trained model for each skill because the single model with skill-specific portions limits the number of characteristics processed on a per skill level.

[0182] The portion trained with respect to characteristics shared by more than one skill may be clustered based on domain. For example, a first portion of the portion trained with respect to multiple skills may be trained with respect to weather domain skills, a second portion of the portion trained with respect to multiple skills may be trained with respect to music domain skills, a third portion of the portion trained with respect to multiple skills may be trained with respect to travel domain skills, etc.

[0183] Clustering may not be beneficial in every instance because it may cause the shortlister component 850 to output indications of only a portion of the skills that the ASR output data 710 may relate to. For example, a user input may correspond to “tell me about Tom Collins.” If the model is clustered based on domain, the shortlister component 850 may determine the user input corresponds to a recipe skill (e.g., a drink recipe) even though the user input may also correspond to an information skill (e.g., including information about a person named Tom Collins).

[0184] The NLU component 560 may include one or more recognizers 863. In at least some embodiments, a recognizer 863 may be associated with a skill system component(s) 525 (e.g., the recognizer may be configured to interpret text data to correspond to the skill system component(s) 525). In at least some other examples, a recognizer 863 may be associated with a domain such as smart home, video, music, weather, custom, etc. (e.g., the recognizer may be configured to interpret text data to correspond to the domain).

[0185] If the shortlister component 850 determines ASR output data 710 is potentially associated with multiple domains, the recognizers 863 associated with the domains may process the ASR output data 710, while recognizers 863 not indicated in the shortlister component 850’s output may not process the ASR output data 710. The “shortlisted” recognizers 863 may process the ASR output data 710 in parallel, in series, partially in parallel, etc. For example, if ASR output data 710 potentially relates to both a communications domain and a music domain, a recognizer associated with the communications domain may process the ASR output data 710 in parallel, or partially in parallel, with a recognizer associated with the music domain processing the ASR output data 710.

[0186] Each recognizer 863 may include a named entity recognition (NER) component 862. The NER component 862 attempts to identify grammars and lexical information that may be used to construe meaning with respect to text data input therein. The NER component 862 identifies portions of text data that correspond to a named entity associated with a domain, associated with the recognizer 863 implementing the NER component 862. The NER component 862 (or other component of the NLU component 560) may also determine whether a word refers to an entity whose identity is not explicitly mentioned in the text data, for example “him,” “her,” “it” or other anaphora, exophora, or the like.

[0187] Each recognizer 863, and more specifically each NER component 862, may be associated with a particular grammar database 876, a particular set of intents/actions 874, and a particular personalized lexicon 886. The grammar databases 876, and intents/actions 874 may be stored in an NLU storage 873. Each gazetteer 884 may include domain / skill-indexed lexical information associated with a particular user and/or device 110. For example, a Gazetteer A (884a) includes skill-indexed lexical information 886aa to 886an. A user’s music domain lexical information might include album titles, artist names, and song names, for example, whereas a user’s communications domain lexical information might include the names of contacts. Since every user’s music collection and contact list is presumably different. This personalized information improves later performed entity resolution.

[0188] An NER component 862 applies grammar information 876 and lexical information 886 associated with a domain (associated with the recognizer 863 implementing the NER component 862) to determine a mention of one or more entities in text data. In this manner, the NER component 862 identifies “slots” (each corresponding to one or more particular words in text data) that may be useful for later processing. The NER component 862 may also label each slot with a type (e.g., noun, place, city, artist name, song name, etc.).

[0189] Each grammar database 876 includes the names of entities (i.e., nouns) commonly found in speech about the particular domain to which the grammar database 876 relates, whereas the lexical information 886 is personalized to the user and/or the device 110 from which the user input originated. For example, a grammar database 876 associated with a shopping domain may include a database of words commonly used when people discuss shopping.

[0190] A downstream process called entity resolution (discussed in detail elsewhere herein) links a slot of text data to a specific entity known to the system. To perform entity resolution, the NLU component 560 may utilize gazetteer information (884a-884n) stored in an entity library storage 882. The gazetteer information 884 may be used to match text data (representing a portion of the user input) with text data representing known entities, such as song titles, contact names, etc. Gazetteers 884 may be linked to users (e.g., a particular gazetteer may be associated with a specific user's music collection), may be linked to certain domains (e.g., a shopping domain, a music domain, a video domain, etc.), or may be organized in a variety of other ways.

[0191] Each recognizer 863 may also include an intent classification (IC) component 864.

An IC component 864 parses text data to determine an intent(s) (associated with the domain associated with the recognizer 863 implementing the IC component 864) that potentially represents the user input. An intent represents to an action a user desires be performed. An IC component 864 may communicate with a database 874 of words linked to intents. For example, a music intent database may link words and phrases such as "quiet," "volume off," and "mute" to a <Mute> intent. An IC component 864 identifies potential intents by comparing words and phrases in text data (representing at least a portion of the user input) to the words and phrases in an intents database 874 (associated with the domain that is associated with the recognizer 863 implementing the IC component 864).

[0192] The intents identifiable by a specific IC component 864 are linked to domain-specific (i.e., the domain associated with the recognizer 863 implementing the IC component 864) grammar frameworks 876 with "slots" to be filled. Each slot of a grammar framework 876 corresponds to a portion of text data that the system believes corresponds to an entity. For example, a grammar framework 876 corresponding to a <PlayMusic> intent may correspond to text data sentence structures such as "Play {Artist Name}," "Play {Album Name}," "Play {Song name}," "Play {Song name} by {Artist Name}," etc. However, to make entity resolution more flexible, grammar frameworks 876 may not be structured as sentences, but rather based on associating slots with grammatical tags.

[0193] For example, an NER component 862 may parse text data to identify words as subject, object, verb, preposition, etc. based on grammar rules and/or models prior to recognizing named entities in the text data. An IC component 864 (implemented by the same recognizer 863

as the NER component 862) may use the identified verb to identify an intent. The NER component 862 may then determine a grammar model 876 associated with the identified intent. For example, a grammar model 876 for an intent corresponding to <PlayMusic> may specify a list of slots applicable to play the identified “object” and any object modifier (e.g., a prepositional phrase), such as {Artist Name}, {Album Name}, {Song name}, etc. The NER component 862 may then search corresponding fields in a lexicon 886 (associated with the domain associated with the recognizer 863 implementing the NER component 862), attempting to match words and phrases in text data the NER component 862 previously tagged as a grammatical object or object modifier with those identified in the lexicon 886.

[0194] An NER component 862 may perform semantic tagging, which is the labeling of a word or combination of words according to their type / semantic meaning. An NER component 862 may parse text data using heuristic grammar rules, or a model may be constructed using techniques such as Hidden Markov Models, maximum entropy models, log linear models, conditional random fields (CRF), and the like. For example, an NER component 862 implemented by a music domain recognizer may parse and tag text data corresponding to “play mother’s little helper by the rolling stones” as {Verb}: “Play,” {Object}: “mother’s little helper,” {Object Preposition}: “by,” and {Object Modifier}: “the rolling stones.” The NER component 862 identifies “Play” as a verb based on a word database associated with the music domain, which an IC component 864 (also implemented by the music domain recognizer) may determine corresponds to a <PlayMusic> intent. At this stage, no determination has been made as to the meaning of “mother’s little helper” or “the rolling stones,” but based on grammar rules and models, the NER component 862 has determined the text of these phrases relates to the grammatical object (i.e., entity) of the user input represented in the text data.

[0195] An NER component 862 may tag text data to attribute meaning thereto. For example, an NER component 862 may tag “play mother’s little helper by the rolling stones” as: {domain} Music, {intent} <PlayMusic>, {artist name} rolling stones, {media type} SONG, and {song title} mother’s little helper. For further example, the NER component 862 may tag “play songs by the rolling stones” as: {domain} Music, {intent} <PlayMusic>, {artist name} rolling stones, and {media type} SONG.

[0196] The shortlister component 850 may receive ASR output data 710 output from the ASR component 550 or output from the device 110b (as illustrated in FIG. 9). The ASR component 550 may embed the ASR output data 710 into a form processable by a trained model(s) using sentence embedding techniques as known in the art. Sentence embedding results in the ASR output data 710 including text in a structure that enables the trained models of the shortlister component 850 to operate on the ASR output data 710. For example, an embedding of the ASR output data 710 may be a vector representation of the ASR output data 710.

[0197] The shortlister component 850 may make binary determinations (e.g., yes or no) regarding which domains relate to the ASR output data 710. The shortlister component 850 may make such determinations using the one or more trained models described herein above. If the shortlister component 850 implements a single trained model for each domain, the shortlister component 850 may simply run the models that are associated with enabled domains as indicated in a user profile associated with the device 110 and/or user that originated the user input.

[0198] The shortlister component 850 may generate n-best list data 915 representing domains that may execute with respect to the user input represented in the ASR output data 710. The size of the n-best list represented in the n-best list data 915 is configurable. In an example, the n-best list data 915 may indicate every domain of the system as well as contain an indication, for each domain, regarding whether the domain is likely capable to execute the user input represented in the ASR output data 710. In another example, instead of indicating every domain of the system, the n-best list data 915 may only indicate the domains that are likely to be able to execute the user input represented in the ASR output data 710. In yet another example, the shortlister component 850 may implement thresholding such that the n-best list data 915 may indicate no more than a maximum number of domains that may execute the user input represented in the ASR output data 710. In an example, the threshold number of domains that may be represented in the n-best list data 915 is ten. In another example, the domains included in the n-best list data 915 may be limited by a threshold a score, where only domains indicating a likelihood to handle the user input is above a certain score (as determined by processing the ASR output data 710 by the shortlister component 850 relative to such domains) are included in the n-best list data 915.

[0199] The ASR output data 710 may correspond to more than one ASR hypothesis. When this occurs, the shortlister component 850 may output a different n-best list (represented in the n-best list data 915) for each ASR hypothesis. Alternatively, the shortlister component 850 may output a single n-best list representing the domains that are related to the multiple ASR

5 hypotheses represented in the ASR output data 710.

[0200] As indicated above, the shortlister component 850 may implement thresholding such that an n-best list output therefrom may include no more than a threshold number of entries. If the ASR output data 710 includes more than one ASR hypothesis, the n-best list output by the shortlister component 850 may include no more than a threshold number of entries irrespective
10 of the number of ASR hypotheses output by the ASR component 550. Alternatively or in addition, the n-best list output by the shortlister component 850 may include no more than a threshold number of entries for each ASR hypothesis (e.g., no more than five entries for a first ASR hypothesis, no more than five entries for a second ASR hypothesis, etc.).

[0201] In addition to making a binary determination regarding whether a domain potentially
15 relates to the ASR output data 710, the shortlister component 850 may generate confidence scores representing likelihoods that domains relate to the ASR output data 710. If the shortlister component 850 implements a different trained model for each domain, the shortlister component 850 may generate a different confidence score for each individual domain trained model that is run. If the shortlister component 850 runs the models of every domain when ASR output data
20 710 is received, the shortlister component 850 may generate a different confidence score for each domain of the system. If the shortlister component 850 runs the models of only the domains that are associated with skills indicated as enabled in a user profile associated with the device 110 and/or user that originated the user input, the shortlister component 850 may only generate a different confidence score for each domain associated with at least one enabled skill. If the
25 shortlister component 850 implements a single trained model with domain specifically trained portions, the shortlister component 850 may generate a different confidence score for each domain whose specifically trained portion is run. The shortlister component 850 may perform matrix vector modification to obtain confidence scores for all domains of the system in a single instance of processing of the ASR output data 710.

[0202] N-best list data 915 including confidence scores that may be output by the shortlister component 850 may be represented as, for example:

Search domain, 0.67

Recipe domain, 0.62

5 Information domain, 0.57

Shopping domain, 0.42

As indicated, the confidence scores output by the shortlister component 850 may be numeric values. The confidence scores output by the shortlister component 850 may alternatively be binned values (e.g., high, medium, low).

10 [0203] The n-best list may only include entries for domains having a confidence score satisfying (e.g., equaling or exceeding) a minimum threshold confidence score. Alternatively, the shortlister component 850 may include entries for all domains associated with user enabled skills, even if one or more of the domains are associated with confidence scores that do not satisfy the minimum threshold confidence score.

15 [0204] The shortlister component 850 may consider other data 920 when determining which domains may relate to the user input represented in the ASR output data 710 as well as respective confidence scores. The other data 920 may include usage history data associated with the device 110 and/or user that originated the user input. For example, a confidence score of a domain may be increased if user inputs originated by the device 110 and/or user routinely invoke
20 the domain. Conversely, a confidence score of a domain may be decreased if user inputs originated by the device 110 and/or user rarely invoke the domain. Thus, the other data 920 may include an indicator of the user associated with the ASR output data 710, for example as determined by the user recognition component 595.

[0205] The other data 920 may be character embedded prior to being input to the shortlister
25 component 850. The other data 920 may alternatively be embedded using other techniques known in the art prior to being input to the shortlister component 850.

[0206] The other data 920 may also include data indicating the domains associated with skills that are enabled with respect to the device 110 and/or user that originated the user input. The shortlister component 850 may use such data to determine which domain-specific trained

models to run. That is, the shortlister component 850 may determine to only run the trained models associated with domains that are associated with user-enabled skills. The shortlister component 850 may alternatively use such data to alter confidence scores of domains.

[0207] As an example, considering two domains, a first domain associated with at least one enabled skill and a second domain not associated with any user-enabled skills of the user that originated the user input, the shortlister component 850 may run a first model specific to the first domain as well as a second model specific to the second domain. Alternatively, the shortlister component 850 may run a model configured to determine a score for each of the first and second domains. The shortlister component 850 may determine a same confidence score for each of the first and second domains in the first instance. The shortlister component 850 may then alter those confidence scores based on which domains is associated with at least one skill enabled by the present user. For example, the shortlister component 850 may increase the confidence score associated with the domain associated with at least one enabled skill while leaving the confidence score associated with the other domain the same. Alternatively, the shortlister component 850 may leave the confidence score associated with the domain associated with at least one enabled skill the same while decreasing the confidence score associated with the other domain. Moreover, the shortlister component 850 may increase the confidence score associated with the domain associated with at least one enabled skill as well as decrease the confidence score associated with the other domain.

[0208] As indicated, a user profile may indicate which skills a corresponding user has enabled (e.g., authorized to execute using data associated with the user). Such indications may be stored in the profile storage 570. When the shortlister component 850 receives the ASR output data 710, the shortlister component 850 may determine whether profile data associated with the user and/or device 110 that originated the command includes an indication of enabled skills.

[0209] The other data 920 may also include data indicating the type of the device 110. The type of a device may indicate the output capabilities of the device. For example, a type of device may correspond to a device with a visual display, a headless (e.g., displayless) device, whether a device is mobile or stationary, whether a device includes audio playback capabilities, whether a device includes a camera, other device hardware configurations, etc. The shortlister component

850 may use such data to determine which domain-specific trained models to run. For example, if the device 110 corresponds to a displayless type device, the shortlister component 850 may determine not to run trained models specific to domains that output video data. The shortlister component 850 may alternatively use such data to alter confidence scores of domains.

5 **[0210]** As an example, considering two domains, one that outputs audio data and another that outputs video data, the shortlister component 850 may run a first model specific to the domain that generates audio data as well as a second model specific to the domain that generates video data. Alternatively the shortlister component 850 may run a model configured to determine a score for each domain. The shortlister component 850 may determine a same confidence score
10 for each of the domains in the first instance. The shortlister component 850 may then alter the original confidence scores based on the type of the device 110 that originated the user input corresponding to the ASR output data 710. For example, if the device 110 is a displayless device, the shortlister component 850 may increase the confidence score associated with the domain that generates audio data while leaving the confidence score associated with the domain
15 that generates video data the same. Alternatively, if the device 110 is a displayless device, the shortlister component 850 may leave the confidence score associated with the domain that generates audio data the same while decreasing the confidence score associated with the domain that generates video data. Moreover, if the device 110 is a displayless device, the shortlister component 850 may increase the confidence score associated with the domain that generates
20 audio data as well as decrease the confidence score associated with the domain that generates video data.

[0211] The type of device information represented in the other data 920 may represent output capabilities of the device to be used to output content to the user, which may not necessarily be the user input originating device. For example, a user may input a spoken user input
25 corresponding to “play Game of Thrones” to a device not including a display. The system may determine a smart TV or other display device (associated with the same user profile) for outputting Game of Thrones. Thus, the other data 920 may represent the smart TV or other display device, and not the displayless device that captured the spoken user input.

[0212] The other data 920 may also include data indicating the user input originating
30 device’s speed, location, or other mobility information. For example, the device may correspond

to a vehicle including a display. If the vehicle is moving, the shortlister component 850 may decrease the confidence score associated with a domain that generates video data as it may be undesirable to output video content to a user while the user is driving. The device may output data to the system component(s) 120 indicating when the device is moving.

5 **[0213]** The other data 920 may also include data indicating a currently invoked domain. For example, a user may speak a first (e.g., a previous) user input causing the system to invoke a music domain skill to output music to the user. As the system is outputting music to the user, the system may receive a second (e.g., the current) user input. The shortlister component 850 may use such data to alter confidence scores of domains. For example, the shortlister component 850
10 may run a first model specific to a first domain as well as a second model specific to a second domain. Alternatively, the shortlister component 850 may run a model configured to determine a score for each domain. The shortlister component 850 may also determine a same confidence score for each of the domains in the first instance. The shortlister component 850 may then alter
15 the original confidence scores based on the first domain being invoked to cause the system to output content while the current user input was received. Based on the first domain being
invoked, the shortlister component 850 may (i) increase the confidence score associated with the first domain while leaving the confidence score associated with the second domain the same, (ii)
leave the confidence score associated with the first domain the same while decreasing the confidence score associated with the second domain, or (iii) increase the confidence score
20 associated with the first domain as well as decrease the confidence score associated with the second domain.

[0214] The thresholding implemented with respect to the n-best list data 915 generated by the shortlister component 850 as well as the different types of other data 920 considered by the shortlister component 850 are configurable. For example, the shortlister component 850 may
25 update confidence scores as more other data 920 is considered. For further example, the n-best list data 915 may exclude relevant domains if thresholding is implemented. Thus, for example, the shortlister component 850 may include an indication of a domain in the n-best list 915 unless the shortlister component 850 is one hundred percent confident that the domain may not execute the user input represented in the ASR output data 710 (e.g., the shortlister component 850
30 determines a confidence score of zero for the domain).

[0215] The shortlister component 850 may send the ASR output data 710 to recognizers 863 associated with domains represented in the n-best list data 915. Alternatively, the shortlister component 850 may send the n-best list data 915 or some other indicator of the selected subset of domains to another component (such as the orchestrator component 530) which may in turn send the ASR output data 710 to the recognizers 863 corresponding to the domains included in the n-best list data 915 or otherwise indicated in the indicator. If the shortlister component 850 generates an n-best list representing domains without any associated confidence scores, the shortlister component 850 / orchestrator component 530 may send the ASR output data 710 to recognizers 863 associated with domains that the shortlister component 850 determines may execute the user input. If the shortlister component 850 generates an n-best list representing domains with associated confidence scores, the shortlister component 850 / orchestrator component 530 may send the ASR output data 710 to recognizers 863 associated with domains associated with confidence scores satisfying (e.g., meeting or exceeding) a threshold minimum confidence score.

[0216] A recognizer 863 may output tagged text data generated by an NER component 862 and an IC component 864, as described herein above. The NLU component 560 may compile the output tagged text data of the recognizers 863 into a single cross-domain n-best list 940 and may send the cross-domain n-best list 940 to a pruning component 950. Each entry of tagged text (e.g., each NLU hypothesis) represented in the cross-domain n-best list data 940 may be associated with a respective score indicating a likelihood that the NLU hypothesis corresponds to the domain associated with the recognizer 863 from which the NLU hypothesis was output. For example, the cross-domain n-best list data 940 may be represented as (with each line corresponding to a different NLU hypothesis):

[0.95] Intent: <PlayMusic> ArtistName: Beethoven SongName: Waldstein Sonata

[0.70] Intent: <PlayVideo> ArtistName: Beethoven VideoName: Waldstein Sonata

[0.01] Intent: <PlayMusic> ArtistName: Beethoven AlbumName: Waldstein Sonata

[0.01] Intent: <PlayMusic> SongName: Waldstein Sonata

[0217] The pruning component 950 may sort the NLU hypotheses represented in the cross-domain n-best list data 940 according to their respective scores. The pruning component 950

may perform score thresholding with respect to the cross-domain NLU hypotheses. For example, the pruning component 950 may select NLU hypotheses associated with scores satisfying (e.g., meeting and/or exceeding) a threshold score. The pruning component 950 may also or alternatively perform number of NLU hypothesis thresholding. For example, the pruning component 950 may select the top scoring NLU hypothesis(es). The pruning component 950 may output a portion of the NLU hypotheses input thereto. The purpose of the pruning component 950 is to create a reduced list of NLU hypotheses so that downstream, more resource intensive, processes may only operate on the NLU hypotheses that most likely represent the user's intent.

[0218] The NLU component 560 may include a light slot filler component 952. The light slot filler component 952 can take text from slots represented in the NLU hypotheses output by the pruning component 950 and alter them to make the text more easily processed by downstream components. The light slot filler component 952 may perform low latency operations that do not involve heavy operations such as reference to a knowledge base (e.g., 872). The purpose of the light slot filler component 952 is to replace words with other words or values that may be more easily understood by downstream components. For example, if a NLU hypothesis includes the word "tomorrow," the light slot filler component 952 may replace the word "tomorrow" with an actual date for purposes of downstream processing. Similarly, the light slot filler component 952 may replace the word "CD" with "album" or the words "compact disc." The replaced words are then included in the cross-domain n-best list data 960.

[0219] The cross-domain n-best list data 960 may be input to an entity resolution component 970. The entity resolution component 970 can apply rules or other instructions to standardize labels or tokens from previous stages into an intent/slot representation. The precise transformation may depend on the domain. For example, for a travel domain, the entity resolution component 970 may transform text corresponding to "Boston airport" to the standard BOS three-letter code referring to the airport. The entity resolution component 970 can refer to a knowledge base (e.g., 872) that is used to specifically identify the precise entity referred to in each slot of each NLU hypothesis represented in the cross-domain n-best list data 960. Specific intent / slot combinations may also be tied to a particular source, which may then be used to resolve the text. In the example "play songs by the stones," the entity resolution component 970 may reference a personal music catalog, Amazon Music account, a user profile, or the like. The

entity resolution component 970 may output an altered n-best list that is based on the cross-domain n-best list 960 but that includes more detailed information (e.g., entity IDs) about the specific entities mentioned in the slots and/or more detailed slot data that can eventually be used by a skill. The NLU component 560 may include multiple entity resolution components 970 and
5 each entity resolution component 970 may be specific to one or more domains.

[0220] The NLU component 560 may include a reranker 990. The reranker 990 may assign a particular confidence score to each NLU hypothesis input therein. The confidence score of a particular NLU hypothesis may be affected by whether the NLU hypothesis has unfilled slots. For example, if a NLU hypothesis includes slots that are all filled/resolved, that NLU hypothesis
10 may be assigned a higher confidence score than another NLU hypothesis including at least some slots that are unfilled/unresolved by the entity resolution component 970.

[0221] The reranker 990 may apply re-scoring, biasing, or other techniques. The reranker 990 may consider not only the data output by the entity resolution component 970, but may also consider other data 991. The other data 991 may include a variety of information. For example,
15 the other data 991 may include skill rating or popularity data. For example, if one skill has a high rating, the reranker 990 may increase the score of a NLU hypothesis that may be processed by the skill. The other data 991 may also include information about skills that have been enabled by the user that originated the user input. For example, the reranker 990 may assign higher scores to NLU hypothesis that may be processed by enabled skills than NLU hypothesis that may
20 be processed by non-enabled skills. The other data 991 may also include data indicating user usage history, such as if the user that originated the user input regularly uses a particular skill or does so at particular times of day. The other data 991 may additionally include data indicating date, time, location, weather, type of device 110, user identifier, context, as well as other information. For example, the reranker 990 may consider when any particular skill is currently
25 active (e.g., music being played, a game being played, etc.).

[0222] As illustrated and described, the entity resolution component 970 is implemented prior to the reranker 990. The entity resolution component 970 may alternatively be implemented after the reranker 990. Implementing the entity resolution component 970 after the reranker 990 limits the NLU hypotheses processed by the entity resolution component 970 to
30 only those hypotheses that successfully pass through the reranker 990.

[0223] The reranker 990 may be a global reranker (e.g., one that is not specific to any particular domain). Alternatively, the NLU component 560 may implement one or more domain-specific rerankers. Each domain-specific reranker may rerank NLU hypotheses associated with the domain. Each domain-specific reranker may output an n-best list of reranked hypotheses (e.g., 5-10 hypotheses).

[0224] The NLU component 560 may perform NLU processing described above with respect to domains associated with skills wholly implemented as part of the system component(s) 120 (e.g., designated 590 in FIGS. 5A and 5B). The NLU component 560 may separately perform NLU processing described above with respect to domains associated with skills that are at least partially implemented as part of the skill system component(s) 525. In an example, the shortlister component 850 may only process with respect to these latter domains. Results of these two NLU processing paths may be merged into NLU output data 985, which may be sent to a post-NLU ranker 965, which may be implemented by the system component(s) 120.

[0225] The post-NLU ranker 965 may include a statistical component that produces a ranked list of intent/skill pairs with associated confidence scores. Each confidence score may indicate an adequacy of the skill's execution of the intent with respect to NLU results data associated with the skill. The post-NLU ranker 965 may operate one or more trained models configured to process the NLU results data 985, skill result data 930, and the other data 920 in order to output ranked output data 925. The ranked output data 925 may include an n-best list where the NLU hypotheses in the NLU results data 985 are reordered such that the n-best list in the ranked output data 925 represents a prioritized list of skills to respond to a user input as determined by the post-NLU ranker 965. The ranked output data 925 may also include (either as part of an n-best list or otherwise) individual respective scores corresponding to skills where each score indicates a probability that the skill (and/or its respective result data) corresponds to the user input.

[0226] The system may be configured with thousands, tens of thousands, etc. skills. The post-NLU ranker 965 enables the system to better determine the best skill to execute the user input. For example, first and second NLU hypotheses in the NLU results data 985 may substantially correspond to each other (e.g., their scores may be significantly similar), even though the first NLU hypothesis may be processed by a first skill and the second NLU

hypothesis may be processed by a second skill. The first NLU hypothesis may be associated with a first confidence score indicating the system's confidence with respect to NLU processing performed to generate the first NLU hypothesis. Moreover, the second NLU hypothesis may be associated with a second confidence score indicating the system's confidence with respect to NLU processing performed to generate the second NLU hypothesis. The first confidence score may be similar or identical to the second confidence score. The first confidence score and/or the second confidence score may be a numeric value (e.g., from 0.0 to 1.0). Alternatively, the first confidence score and/or the second confidence score may be a binned value (e.g., low, medium, high).

[0227] The post-NLU ranker 965 (or other scheduling component such as orchestrator component 530) may solicit the first skill and the second skill to provide potential result data 930 based on the first NLU hypothesis and the second NLU hypothesis, respectively. For example, the post-NLU ranker 965 may send the first NLU hypothesis to the first skill 590a along with a request for the first skill 590a to at least partially execute with respect to the first NLU hypothesis. The post-NLU ranker 965 may also send the second NLU hypothesis to the second skill 590b along with a request for the second skill 590b to at least partially execute with respect to the second NLU hypothesis. The post-NLU ranker 965 receives, from the first skill 590a, first result data 930a generated from the first skill 590a's execution with respect to the first NLU hypothesis. The post-NLU ranker 965 also receives, from the second skill 590b, second results data 930b generated from the second skill 590b's execution with respect to the second NLU hypothesis.

[0228] The result data 930 may include various portions. For example, the result data 930 may include content (e.g., audio data, text data, and/or video data) to be output to a user. The result data 930 may also include a unique identifier used by the system component(s) 120 and/or the skill system component(s) 525 to locate the data to be output to a user. The result data 930 may also include an instruction. For example, if the user input corresponds to "turn on the light," the result data 930 may include an instruction causing the system to turn on a light associated with a profile of the device (110a/110b) and/or user.

[0229] The post-NLU ranker 965 may consider the first result data 930a and the second result data 930b to alter the first confidence score and the second confidence score of the first

NLU hypothesis and the second NLU hypothesis, respectively. That is, the post-NLU ranker 965 may generate a third confidence score based on the first result data 930a and the first confidence score. The third confidence score may correspond to how likely the post-NLU ranker 965 determines the first skill will correctly respond to the user input. The post-NLU ranker 965 may also generate a fourth confidence score based on the second result data 930b and the second confidence score. One skilled in the art will appreciate that a first difference between the third confidence score and the fourth confidence score may be greater than a second difference between the first confidence score and the second confidence score. The post-NLU ranker 965 may also consider the other data 920 to generate the third confidence score and the fourth confidence score. While it has been described that the post-NLU ranker 965 may alter the confidence scores associated with first and second NLU hypotheses, one skilled in the art will appreciate that the post-NLU ranker 965 may alter the confidence scores of more than two NLU hypotheses. The post-NLU ranker 965 may select the result data 930 associated with the skill 590 with the highest altered confidence score to be the data output in response to the current user input. The post-NLU ranker 965 may also consider the ASR output data 710 to alter the NLU hypotheses confidence scores.

[0230] The orchestrator component 530 may, prior to sending the NLU results data 985 to the post-NLU ranker 965, associate intents in the NLU hypotheses with skills 590. For example, if a NLU hypothesis includes a <PlayMusic> intent, the orchestrator component 530 may associate the NLU hypothesis with one or more skills 590 that can execute the <PlayMusic> intent. Thus, the orchestrator component 530 may send the NLU results data 985, including NLU hypotheses paired with skills 590, to the post-NLU ranker 965. In response to ASR output data 710 corresponding to “what should I do for dinner today,” the orchestrator component 530 may generate pairs of skills 590 with associated NLU hypotheses corresponding to:

Skill 1 / NLU hypothesis including <Help> intent

Skill 2 / NLU hypothesis including <Order> intent

Skill 3 / NLU hypothesis including <DishType> intent

[0231] The post-NLU ranker 965 queries each skill 590, paired with a NLU hypothesis in the NLU output data 985, to provide result data 930 based on the NLU hypothesis with which it is associated. That is, with respect to each skill, the post-NLU ranker 965 colloquially asks the

each skill “if given this NLU hypothesis, what would you do with it.” According to the above example, the post-NLU ranker 965 may send skills 590 the following data:

Skill 1: First NLU hypothesis including <Help> intent indicator

Skill 2: Second NLU hypothesis including <Order> intent indicator

5 Skill 3: Third NLU hypothesis including <DishType> intent indicator

The post-NLU ranker 965 may query each of the skills 590 in parallel or substantially in parallel.

[0232] A skill 590 may provide the post-NLU ranker 965 with various data and indications in response to the post-NLU ranker 965 soliciting the skill 590 for result data 930. A skill 590 may simply provide the post-NLU ranker 965 with an indication of whether or not the skill can
10 execute with respect to the NLU hypothesis it received. A skill 590 may also or alternatively provide the post-NLU ranker 965 with output data generated based on the NLU hypothesis it received. In some situations, a skill 590 may need further information in addition to what is represented in the received NLU hypothesis to provide output data responsive to the user input. In these situations, the skill 590 may provide the post-NLU ranker 965 with result data 930
15 indicating slots of a framework that the skill 590 further needs filled or entities that the skill 590 further needs resolved prior to the skill 590 being able to provided result data 930 responsive to the user input. The skill 590 may also provide the post-NLU ranker 965 with an instruction and/or computer-generated speech indicating how the skill 590 recommends the system solicit further information needed by the skill 590. The skill 590 may further provide the post-NLU
20 ranker 965 with an indication of whether the skill 590 will have all needed information after the user provides additional information a single time, or whether the skill 590 will need the user to provide various kinds of additional information prior to the skill 590 having all needed information. According to the above example, skills 590 may provide the post-NLU ranker 965 with the following:

25 Skill 1: indication representing the skill can execute with respect to a NLU hypothesis including the <Help> intent indicator

Skill 2: indication representing the skill needs to the system to obtain further information

Skill 3: indication representing the skill can provide numerous results in response to the third NLU hypothesis including the <DishType> intent indicator

[0233] Result data 930 includes an indication provided by a skill 590 indicating whether or not the skill 590 can execute with respect to a NLU hypothesis; data generated by a skill 590 based on a NLU hypothesis; as well as an indication provided by a skill 590 indicating the skill 590 needs further information in addition to what is represented in the received NLU hypothesis.

5 [0234] The post-NLU ranker 965 uses the result data 930 provided by the skills 590 to alter the NLU processing confidence scores generated by the reranker 990. That is, the post-NLU ranker 965 uses the result data 930 provided by the queried skills 590 to create larger differences between the NLU processing confidence scores generated by the reranker 990. Without the post-NLU ranker 965, the system may not be confident enough to determine an output in response to
10 a user input, for example when the NLU hypotheses associated with multiple skills are too close for the system to confidently determine a single skill 590 to invoke to respond to the user input. For example, if the system does not implement the post-NLU ranker 965, the system may not be able to determine whether to obtain output data from a general reference information skill or a medical information skill in response to a user input corresponding to “what is acne.”

15 [0235] The post-NLU ranker 965 may prefer skills 590 that provide result data 930 responsive to NLU hypotheses over skills 590 that provide result data 930 corresponding to an indication that further information is needed, as well as skills 590 that provide result data 930 indicating they can provide multiple responses to received NLU hypotheses. For example, the post-NLU ranker 965 may generate a first score for a first skill 590a that is greater than the first
20 skill’s NLU confidence score based on the first skill 590a providing result data 930a including a response to a NLU hypothesis. For further example, the post-NLU ranker 965 may generate a second score for a second skill 590b that is less than the second skill’s NLU confidence score based on the second skill 590b providing result data 930b indicating further information is needed for the second skill 590b to provide a response to a NLU hypothesis. Yet further, for
25 example, the post-NLU ranker 965 may generate a third score for a third skill 590c that is less than the third skill’s NLU confidence score based on the third skill 590c providing result data 930c indicating the third skill 590c can provide multiple responses to a NLU hypothesis.

[0236] The post-NLU ranker 965 may consider other data 920 in determining scores. The other data 920 may include rankings associated with the queried skills 590. A ranking may be a
30 system ranking or a user-specific ranking. A ranking may indicate a veracity of a skill from the

perspective of one or more users of the system. For example, the post-NLU ranker 965 may generate a first score for a first skill 590a that is greater than the first skill's NLU processing confidence score based on the first skill 590a being associated with a high ranking. For further example, the post-NLU ranker 965 may generate a second score for a second skill 590b that is less than the second skill's NLU processing confidence score based on the second skill 590b being associated with a low ranking.

[0237] The other data 920 may include information indicating whether or not the user that originated the user input has enabled one or more of the queried skills 590. For example, the post-NLU ranker 965 may generate a first score for a first skill 590a that is greater than the first skill's NLU processing confidence score based on the first skill 590a being enabled by the user that originated the user input. For further example, the post-NLU ranker 965 may generate a second score for a second skill 590b that is less than the second skill's NLU processing confidence score based on the second skill 590b not being enabled by the user that originated the user input. When the post-NLU ranker 965 receives the NLU results data 985, the post-NLU ranker 965 may determine whether profile data, associated with the user and/or device that originated the user input, includes indications of enabled skills.

[0238] The other data 920 may include information indicating output capabilities of a device that will be used to output content, responsive to the user input, to the user. The system may include devices that include speakers but not displays, devices that include displays but not speakers, and devices that include speakers and displays. If the device that will output content responsive to the user input includes one or more speakers but not a display, the post-NLU ranker 965 may increase the NLU processing confidence score associated with a first skill configured to output audio data and/or decrease the NLU processing confidence score associated with a second skill configured to output visual data (e.g., image data and/or video data). If the device that will output content responsive to the user input includes a display but not one or more speakers, the post-NLU ranker 965 may increase the NLU processing confidence score associated with a first skill configured to output visual data and/or decrease the NLU processing confidence score associated with a second skill configured to output audio data.

[0239] The other data 920 may include information indicating the veracity of the result data 930 provided by a skill 590. For example, if a user says "tell me a recipe for pasta sauce," a first

skill 590a may provide the post-NLU ranker 965 with first result data 930a corresponding to a first recipe associated with a five star rating and a second skill 590b may provide the post-NLU ranker 965 with second result data 930b corresponding to a second recipe associated with a one star rating. In this situation, the post-NLU ranker 965 may increase the NLU processing confidence score associated with the first skill 590a based on the first skill 590a providing the first result data 930a associated with the five star rating and/or decrease the NLU processing confidence score associated with the second skill 590b based on the second skill 590b providing the second result data 930b associated with the one star rating.

[0240] The other data 920 may include information indicating the type of device that originated the user input. For example, the device may correspond to a “hotel room” type if the device is located in a hotel room. If a user inputs a command corresponding to “order me food” to the device located in the hotel room, the post-NLU ranker 965 may increase the NLU processing confidence score associated with a first skill 590a corresponding to a room service skill associated with the hotel and/or decrease the NLU processing confidence score associated with a second skill 590b corresponding to a food skill not associated with the hotel.

[0241] The other data 920 may include information indicating a location of the device and/or user that originated the user input. The system may be configured with skills 590 that may only operate with respect to certain geographic locations. For example, a user may provide a user input corresponding to “when is the next train to Portland.” A first skill 590a may operate with respect to trains that arrive at, depart from, and pass through Portland, Oregon. A second skill 590b may operate with respect to trains that arrive at, depart from, and pass through Portland, Maine. If the device and/or user that originated the user input is located in Seattle, Washington, the post-NLU ranker 965 may increase the NLU processing confidence score associated with the first skill 590a and/or decrease the NLU processing confidence score associated with the second skill 590b. Likewise, if the device and/or user that originated the user input is located in Boston, Massachusetts, the post-NLU ranker 965 may increase the NLU processing confidence score associated with the second skill 590b and/or decrease the NLU processing confidence score associated with the first skill 590a.

[0242] The other data 920 may include information indicating a time of day. The system may be configured with skills 590 that operate with respect to certain times of day. For example,

a user may provide a user input corresponding to “order me food.” A first skill 590a may generate first result data 930a corresponding to breakfast. A second skill 590b may generate second result data 930b corresponding to dinner. If the system component(s) 120 receives the user input in the morning, the post-NLU ranker 965 may increase the NLU processing confidence score associated with the first skill 590a and/or decrease the NLU processing score associated with the second skill 590b. If the system component(s) 120 receives the user input in the afternoon or evening, the post-NLU ranker 965 may increase the NLU processing confidence score associated with the second skill 590b and/or decrease the NLU processing confidence score associated with the first skill 590a.

[0243] The other data 920 may include information indicating user preferences. The system may include multiple skills 590 configured to execute in substantially the same manner. For example, a first skill 590a and a second skill 590b may both be configured to order food from respective restaurants. The system may store a user preference (e.g., in the profile storage 570) that is associated with the user that provided the user input to the system component(s) 120 as well as indicates the user prefers the first skill 590a over the second skill 590b. Thus, when the user provides a user input that may be executed by both the first skill 590a and the second skill 590b, the post-NLU ranker 965 may increase the NLU processing confidence score associated with the first skill 590a and/or decrease the NLU processing confidence score associated with the second skill 590b.

[0244] The other data 920 may include information indicating system usage history associated with the user that originated the user input. For example, the system usage history may indicate the user originates user inputs that invoke a first skill 590a more often than the user originates user inputs that invoke a second skill 590b. Based on this, if the present user input may be executed by both the first skill 590a and the second skill 590b, the post-NLU ranker 965 may increase the NLU processing confidence score associated with the first skill 590a and/or decrease the NLU processing confidence score associated with the second skill 590b.

[0245] The other data 920 may include information indicating a speed at which the device 110 that originated the user input is traveling. For example, the device 110 may be located in a moving vehicle, or may be a moving vehicle. When a device 110 is in motion, the system may prefer audio outputs rather than visual outputs to decrease the likelihood of distracting the user

(e.g., a driver of a vehicle). Thus, for example, if the device 110 that originated the user input is moving at or above a threshold speed (e.g., a speed above an average user's walking speed), the post-NLU ranker 965 may increase the NLU processing confidence score associated with a first skill 590a that generates audio data. The post-NLU ranker 965 may also or alternatively
5 decrease the NLU processing confidence score associated with a second skill 590b that generates image data or video data.

[0246] The other data 920 may include information indicating how long it took a skill 590 to provide result data 930 to the post-NLU ranker 965. When the post-NLU ranker 965 multiple skills 590 for result data 930, the skills 590 may respond to the queries at different speeds. The
10 post-NLU ranker 965 may implement a latency budget. For example, if the post-NLU ranker 965 determines a skill 590 responds to the post-NLU ranker 965 within a threshold amount of time from receiving a query from the post-NLU ranker 965, the post-NLU ranker 965 may increase the NLU processing confidence score associated with the skill 590. Conversely, if the
15 post-NLU ranker 965 determines a skill 590 does not respond to the post-NLU ranker 965 within a threshold amount of time from receiving a query from the post-NLU ranker 965, the post-NLU ranker 965 may decrease the NLU processing confidence score associated with the skill 590.

[0247] It has been described that the post-NLU ranker 965 uses the other data 920 to increase and decrease NLU processing confidence scores associated with various skills 590 that the post-NLU ranker 965 has already requested result data from. Alternatively, the post-NLU ranker 965
20 may use the other data 920 to determine which skills 590 to request result data from. For example, the post-NLU ranker 965 may use the other data 920 to increase and/or decrease NLU processing confidence scores associated with skills 590 associated with the NLU results data 985 output by the NLU component 560. The post-NLU ranker 965 may select n-number of top
25 scoring altered NLU processing confidence scores. The post-NLU ranker 965 may then request result data 930 from only the skills 590 associated with the selected n-number of NLU processing confidence scores.

[0248] As described, the post-NLU ranker 965 may request result data 930 from all skills 590 associated with the NLU results data 985 output by the NLU component 560. Alternatively, the system component(s) 120 may prefer result data 930 from skills implemented entirely by the
30 system component(s) 120 rather than skills at least partially implemented by the skill system

component(s) 525. Therefore, in the first instance, the post-NLU ranker 965 may request result data 930 from only skills associated with the NLU results data 985 and entirely implemented by the system component(s) 120. The post-NLU ranker 965 may only request result data 930 from skills associated with the NLU results data 985, and at least partially implemented by the skill system component(s) 525, if none of the skills, wholly implemented by the system component(s) 120, provide the post-NLU ranker 965 with result data 930 indicating either data response to the NLU results data 985, an indication that the skill can execute the user input, or an indication that further information is needed.

[0249] As indicated above, the post-NLU ranker 965 may request result data 930 from multiple skills 590. If one of the skills 590 provides result data 930 indicating a response to a NLU hypothesis and the other skills provide result data 930 indicating either they cannot execute or they need further information, the post-NLU ranker 965 may select the result data 930 including the response to the NLU hypothesis as the data to be output to the user. If more than one of the skills 590 provides result data 930 indicating responses to NLU hypotheses, the post-NLU ranker 965 may consider the other data 920 to generate altered NLU processing confidence scores, and select the result data 930 of the skill associated with the greatest score as the data to be output to the user.

[0250] A system that does not implement the post-NLU ranker 965 may select the highest scored NLU hypothesis in the NLU results data 985. The system may send the NLU hypothesis to a skill 590 associated therewith along with a request for output data. In some situations, the skill 590 may not be able to provide the system with output data. This results in the system indicating to the user that the user input could not be processed even though another skill associated with lower ranked NLU hypothesis could have provided output data responsive to the user input.

[0251] The post-NLU ranker 965 reduces instances of the aforementioned situation. As described, the post-NLU ranker 965 queries multiple skills associated with the NLU results data 985 to provide result data 930 to the post-NLU ranker 965 prior to the post-NLU ranker 965 ultimately determining the skill 590 to be invoked to respond to the user input. Some of the skills 590 may provide result data 930 indicating responses to NLU hypotheses while other skills 590 may providing result data 930 indicating the skills cannot provide responsive data. Whereas

a system not implementing the post-NLU ranker 965 may select one of the skills 590 that could not provide a response, the post-NLU ranker 965 only selects a skill 590 that provides the post-NLU ranker 965 with result data corresponding to a response, indicating further information is needed, or indicating multiple responses can be generated.

5 **[0252]** The post-NLU ranker 965 may select result data 930, associated with the skill 590 associated with the highest score, for output to the user. Alternatively, the post-NLU ranker 965 may output ranked output data 925 indicating skills 590 and their respective post-NLU ranker rankings. Since the post-NLU ranker 965 receives result data 930, potentially corresponding to a response to the user input, from the skills 590 prior to post-NLU ranker 965 selecting one of the
10 skills or outputting the ranked output data 925, little to no latency occurs from the time skills provide result data 930 and the time the system outputs responds to the user.

[0253] If the post-NLU ranker 965 selects result audio data to be output to a user and the system determines content should be output audibly, the post-NLU ranker 965 (or another component of the system component(s) 120) may cause the device 110a and/or the device 110b
15 to output audio corresponding to the result audio data. If the post-NLU ranker 965 selects result text data to output to a user and the system determines content should be output visually, the post-NLU ranker 965 (or another component of the system component(s) 120) may cause the device 110b to display text corresponding to the result text data. If the post-NLU ranker 965 selects result audio data to output to a user and the system determines content should be output
20 visually, the post-NLU ranker 965 (or another component of the system component(s) 120) may send the result audio data to the ASR component 550. The ASR component 550 may generate output text data corresponding to the result audio data. The system component(s) 120 may then cause the device 110b to display text corresponding to the output text data. If the post-NLU ranker 965 selects result text data to output to a user and the system determines content should be
25 output audibly, the post-NLU ranker 965 (or another component of the system component(s) 120) may send the result text data to the TTS component 580. The TTS component 580 may generate output audio data (corresponding to computer-generated speech) based on the result text data. The system component(s) 120 may then cause the device 110a and/or the device 110b to output audio corresponding to the output audio data.

[0254] As described, a skill 590 may provide result data 930 either indicating a response to the user input, indicating more information is needed for the skill 590 to provide a response to the user input, or indicating the skill 590 cannot provide a response to the user input. If the skill 590 associated with the highest post-NLU ranker score provides the post-NLU ranker 965 with result data 930 indicating a response to the user input, the post-NLU ranker 965 (or another component of the system component(s) 120, such as the orchestrator component 530) may simply cause content corresponding to the result data 930 to be output to the user. For example, the post-NLU ranker 965 may send the result data 930 to the orchestrator component 530. The orchestrator component 530 may cause the result data 930 to be sent to the device (110a/110b), which may output audio and/or display text corresponding to the result data 930. The orchestrator component 530 may send the result data 930 to the ASR component 550 to generate output text data and/or may send the result data 930 to the TTS component 580 to generate output audio data, depending on the situation.

[0255] The skill 590 associated with the highest post-NLU ranker score may provide the post-NLU ranker 965 with result data 930 indicating more information is needed as well as instruction data. The instruction data may indicate how the skill 590 recommends the system obtain the needed information. For example, the instruction data may correspond to text data or audio data (i.e., computer-generated speech) corresponding to “please indicate _____.” The instruction data may be in a format (e.g., text data or audio data) capable of being output by the device (110a/110b). When this occurs, the post-NLU ranker 965 may simply cause the received instruction data be output by the device (110a/110b). Alternatively, the instruction data may be in a format that is not capable of being output by the device (110a/110b). When this occurs, the post-NLU ranker 965 may cause the ASR component 550 or the TTS component 580 to process the instruction data, depending on the situation, to generate instruction data that may be output by the device (110a/110b). Once the user provides the system with all further information needed by the skill 590, the skill 590 may provide the system with result data 930 indicating a response to the user input, which may be output by the system as detailed above.

[0256] The system may include “informational” skills 590 that simply provide the system with information, which the system outputs to the user. The system may also include “transactional” skills 590 that require a system instruction to execute the user input.

Transactional skills 590 include ride sharing skills, flight booking skills, etc. A transactional skill 590 may simply provide the post-NLU ranker 965 with result data 930 indicating the transactional skill 590 can execute the user input. The post-NLU ranker 965 may then cause the system to solicit the user for an indication that the system is permitted to cause the transactional skill 590 to execute the user input. The user-provided indication may be an audible indication or a tactile indication (e.g., activation of a virtual button or input of text via a virtual keyboard). In response to receiving the user-provided indication, the system may provide the transactional skill 590 with data corresponding to the indication. In response, the transactional skill 590 may execute the command (e.g., book a flight, book a train ticket, etc.). Thus, while the system may not further engage an informational skill 590 after the informational skill 590 provides the post-NLU ranker 965 with result data 930, the system may further engage a transactional skill 590 after the transactional skill 590 provides the post-NLU ranker 965 with result data 930 indicating the transactional skill 590 may execute the user input.

[0257] In some instances, the post-NLU ranker 965 may generate respective scores for first and second skills that are too close (e.g., are not different by at least a threshold difference) for the post-NLU ranker 965 to make a confident determination regarding which skill should execute the user input. When this occurs, the system may request the user indicate which skill the user prefers to execute the user input. The system may output TTS-generated speech to the user to solicit which skill the user wants to execute the user input.

[0258] FIG. 10 illustrate operations to handle dialog management, according to embodiments of the present disclosure. One or more components shown in FIG. 10 may be part of the dialog manager 572. For example, the dialog manager 572 may include the entity resolver 1070, the focus data component 1016 and the action selector 1018. The dialog manager 572 may work in concert with other language processing components, for example NLU 560, or may operate instead of such components in certain embodiments.

[0259] The system receives input text data 1002 which may be received, for example, by a device or from another component of the system (for example as ASR output data 710). The input text data 1002 may include text corresponding to a user input and metadata indicating further information about the text (such as an identity of the user for example determined by user recognition component 595), an emotional state of the user (for example determined by A

sentiment detection component not shown, etc.). The input text data 1002 may be text representing words, instructions, markup language, speech, or gestures, or may be a vector or other representation of the same. The input text data 1002 may be generated by a user via a keyboard, touchscreen, microphone, camera, or other such input device. In other embodiments, the input text data 1002 is created using ASR, as described above, from audio data received from a user. The system may further receive other input data 1004, which may correspond to a button press, gesture, or other input, such as image data as may interpreted by device 110, system component(s), or other component (for example image processing component, not shown). As described in greater detail below, using the input text data 1002 and/or other input data 1004, the system may determine and output text data 1315 and/or other output data 1008. The system may instead or in addition perform an action based on the input text data 1002 and/or other input data 1004, such as calling one or more APIs 1010.

[0260] An entity resolver 1070 may be used to determine that the input text data 1002 includes a representation of one or more entities, a process that may include named entity recognition (NER) – i.e., determining that the input text data 1002 includes the representation – and entity resolution (ER) – i.e., identifying a meaning or context of the entity, such as associating an identity of a person based on a recognized nickname. An entity may be a person, place, thing, idea, and/or goal; example entities include proper names, nicknames, business names, place names, and/or application names. The entity resolver 1070 may operate similarly to NER component 862 and/or entity resolution component 970 discussed herein with regard to NLU operations.

[0261] In some embodiments, a single entity resolver 1070 is used for more than one domain (i.e., a “cross-domain” entity resolver 1070). Each domain may correspond to one or more dialog models. One or more candidate domains corresponding to the input text data 1070 may be determined; entity resolvers 1070 corresponding to the candidate domains may be used to process the input text data 1070. The dialog focus data 1016 may store the output entities from each candidate domain and may remove unselected entities when an API 1010 is selected or an action to be performed is determined.

[0262] The dialog focus data 1016 may store state data (for example in dialog storage 1030) corresponding to dialog history data (including an intent(s), an entity(ies), etc. corresponding to a

prior turn in the dialog), action history data, and/or other data. In some embodiments, the other components (e.g., the action selector 1018) do not store state data and instead query the dialog focus data 1016 for the state data. The system may send some or all of the dialog focus data 1016 to other systems and/or may receive additional dialog focus data from other systems. In some embodiments, the other components (e.g., the action selector 1018) include a feature-extractor component to extract features from the dialog focus data 1016.

[0263] The dialog focus data 1016 may be graph-based data including a plurality of graph nodes; each graph node may correspond to an item of state data, such as an entity type, entity value, prior API call, and/or user data. The other components, such as the action selector 1018, may access all of the graph nodes of the dialog focus data 1016 or may access only a subset of the graph nodes of the dialog focus data 1016. The dialog focus data 1016 may be any type of storage mechanism and may serve as long-term and/or short term memory for the system, thus enabling tracking of entities, ASR output, TTS output, and other features) at each turn through a dialog. In some embodiments, the dialog focus data 1016 is updated after each turn of dialog with updated dialog focus data; in other embodiments, the dialog focus data 1016 is updated after an end of a dialog is determined.

[0264] The entity resolver 1070 may utilize gazetteer information stored in an entity library storage. The gazetteer information may be used to match text data (representing a portion of the user input) with text data representing known entities, such as song titles, contact names, etc. Gazetteers may be linked to users (e.g., a particular gazetteer may be associated with a specific user's music collection), may be linked to certain skills 590 (e.g., a shopping skill, a music skill, a video skill, etc.), or may be organized in a variety of other ways.

[0265] The entity resolver 1070 may include a ranker component. The ranker component may assign a particular confidence score to each hypothesis input therein. The confidence score of a hypothesis may represent a confidence of the system in the processing performed with respect to the hypothesis. The confidence score of a particular hypothesis may be affected by whether the hypothesis has unfilled slots. For example, if a hypothesis associated with a first skill component includes slots that are all filled/resolved, that NLU hypothesis may be assigned a higher confidence score than another hypothesis including at least some slots that are unfilled/unresolved by the entity resolver 1070.

[0266] The focus data 1016 may store data relevant to a dialog. In various embodiments, the dialog focus data 1016 stores the input text data 1002, other input data 1004, entity data from the entity resolver 1070 and/or action data and dialog data from an action selector 1018. The dialog focus data 1016 may further include additional information, such as location data, user preference data, and environment data. In various embodiments, the dialog focus data 1016 uses an encoder to encode some or all of the received data into one or more feature vectors and a decoder to determine, based on the feature vectors, intent data corresponding to an intent of a user. The dialog focus data 1016 may further include state data that represents prior dialog, actions, or other prior user information or data.

[0267] The action selector 1018 determines an action to be performed in response to the user request, where the action may include calling an API to perform an action and/or presenting an output to the user. The action selector 1018 may include a trained model(s), and may process the dialog focus data 1016. If the action selector 1018 determines to invoke an API call, one or more APIs 1010 may be activated and a corresponding action carried out. If the action selector 1018 determines to present a prompt or other output data to the user, the NLG component 579 may be used to generate the output text data 1315 and/or other output data 1008. In either case, the action selection 1018 may update the dialog focus data 1016 based on the API call or the output presented to the user.

[0268] In some embodiments, the action selector 1018 may process data from the dialog storage 1030 to select one or more skills 590 / skill system component(s) 525 capable of responding to the user request, and present the selected skill to the user using the output text data 1315.

[0269] In some embodiments, the system component(s) 120 may determine past interaction data, such as a satisfaction rating corresponding to skill(s) 590 and store it in the storage 1030.

The satisfaction rating may be based on past interactions between users of the system component(s) 120 and the skill. In some embodiments, the system component(s) 120 may determine a user-specific satisfaction rating corresponding to the skill based on user-specific data, interaction-specific data and/or focus data 1016, such as, user profile data 570 associated with the specific user, location data, past user interactions with the system component(s) 120, past user interactions with the skill 590, user preferences, device type for the device 110 that

received the user request, device type for the device 110 that may output / respond to the user request, device capabilities, dialog state of previous turns in the dialog, and other data.

[0270] In some embodiments, users may provide feedback to the system component(s) 120 or the skill system component(s) 525 indicating the user's satisfaction in the service skill responding to the user request / performing an action in response to the user request. The feedback may be solicited by the system component(s) 120 or the skill system component(s) 525. In some cases, the feedback may be explicit feedback and may be received as a voice input, text input, gestures, or other types of input. In other cases, the feedback may be implicit feedback and may be determined by processing data related to the user interaction, such as, image data representing the user's facial expression, audio data (representing an utterance from the user) indicating a sentiment / emotion of the user, text data (representing the user input) indicating a sentiment of the user, other sounds / speech indicating the user's emotions / feelings / sentiment (e.g., "ugh" indicating frustration), and other types of data.

[0271] The action selector 1018 may select a skill based on various other factors, including but not limited to, a user's age, user cohort (user affinities or other social organizations), location information (e.g., where the user input originated, where the user intends to travel to, etc.), skill location data, intent, intent type, time of day the user input is received, time of year the user input is received, and others.

[0272] The action selector 1018 may select a skill based on skill availability, for example, a reservations at a restaurant may indicate availability or no availability for a particular day. The system component(s) 120 / action selector 1018 may determine the number of diners at a restaurant or the number of takeout orders a restaurant is handling based on the number of user requests the system component(s) 120 routed to the skill. In another example, the system component(s) 120 / action selector 1018 may determine that a number of user requests were routed to a first skill for a first location / geographic area (e.g., a city), and may determine to route the next / future user requests to another skill for that geographic area.

[0273] For a restaurant reservation, the action selector 1018 may select a skill based on a number of people in the user's party, since some restaurants may not make reservations for small parties less than a particular number. The action selector 1018 may select a skill based on person(s) other than the user that may be involved in the user request. For example, if the user

says “Alexa, what should we do tomorrow?” the system component(s) 120 may determine information relating to the other persons in the user’s household to select a skill.

[0274] The action selector 1018 may select a skill based on the dialog state where the dialog state indicates multiple turns / attempts to recommend a skill / service provider, and the action selector 1018 may modify the search criteria (e.g., expand the zip code, expand the time frame, etc.) to select a skill.

[0275] The device 110 and/or the system component(s) 120 may include a user recognition component 595 that recognizes one or more users using a variety of data. As illustrated in FIG. 11, the user recognition component 595 may include one or more subcomponents including a vision component 1108, an audio component 1110, a biometric component 1112, a radio frequency (RF) component 1114, a machine learning (ML) component 1116, and a recognition confidence component 1118. In some instances, the user recognition component 595 may monitor data and determinations from one or more subcomponents to determine an identity of one or more users associated with data input to the device 110 and/or the system component(s) 120. The user recognition component 595 may output user recognition data 1195, which may include a user identifier associated with a user the user recognition component 595 determines originated data input to the device 110 and/or the system component(s) 120. The user recognition data 1195 may be used to inform processes performed by various components of the device 110 and/or the system component(s) 120.

[0276] The vision component 1108 may receive data from one or more sensors capable of providing images (e.g., cameras) or sensors indicating motion (e.g., motion sensors). The vision component 1108 can perform facial recognition or image analysis to determine an identity of a user and to associate that identity with a user profile associated with the user. In some instances, when a user is facing a camera, the vision component 1108 may perform facial recognition and identify the user with a high degree of confidence. In other instances, the vision component 1108 may have a low degree of confidence of an identity of a user, and the user recognition component 595 may utilize determinations from additional components to determine an identity of a user. The vision component 1108 can be used in conjunction with other components to determine an identity of a user. For example, the user recognition component 595 may use data from the vision component 1108 with data from the audio component 1110 to identify what

user's face appears to be speaking at the same time audio is captured by a device 110 the user is facing for purposes of identifying a user who spoke an input to the device 110 and/or the system component(s) 120.

[0277] The overall system of the present disclosure may include biometric sensors that transmit data to the biometric component 1112. For example, the biometric component 1112 may receive data corresponding to fingerprints, iris or retina scans, thermal scans, weights of users, a size of a user, pressure (e.g., within floor sensors), etc., and may determine a biometric profile corresponding to a user. The biometric component 1112 may distinguish between a user and sound from a television, for example. Thus, the biometric component 1112 may incorporate biometric information into a confidence level for determining an identity of a user. Biometric information output by the biometric component 1112 can be associated with specific user profile data such that the biometric information uniquely identifies a user profile of a user.

[0278] The radio frequency (RF) component 1114 may use RF localization to track devices that a user may carry or wear. For example, a user (and a user profile associated with the user) may be associated with a device. The device may emit RF signals (e.g., Wi-Fi, Bluetooth®, etc.). A device may detect the signal and indicate to the RF component 1114 the strength of the signal (e.g., as a received signal strength indication (RSSI)). The RF component 1114 may use the RSSI to determine an identity of a user (with an associated confidence level). In some instances, the RF component 1114 may determine that a received RF signal is associated with a mobile device that is associated with a particular user identifier.

[0279] In some instances, a personal device (such as a phone, tablet, wearable or other device) may include some RF or other detection processing capabilities so that a user who speaks an input may scan, tap, or otherwise acknowledge his/her personal device to the device 110. In this manner, the user may "register" with the system 100 for purposes of the system 100 determining who spoke a particular input. Such a registration may occur prior to, during, or after speaking of an input.

[0280] The ML component 1116 may track the behavior of various users as a factor in determining a confidence level of the identity of the user. By way of example, a user may adhere to a regular schedule such that the user is at a first location during the day (e.g., at work or at school). In this example, the ML component 1116 would factor in past behavior and/or trends in

determining the identity of the user that provided input to the device 110 and/or the system component(s) 120. Thus, the ML component 1116 may use historical data and/or usage patterns over time to increase or decrease a confidence level of an identity of a user.

[0281] In at least some instances, the recognition confidence component 1118 receives determinations from the various components 1108, 1110, 1112, 1114, and 1116, and may determine a final confidence level associated with the identity of a user. In some instances, the confidence level may determine whether an action is performed in response to a user input. For example, if a user input includes a request to unlock a door, a confidence level may need to be above a threshold that may be higher than a threshold confidence level needed to perform a user request associated with playing a playlist or sending a message. The confidence level or other score data may be included in the user recognition data 1195.

[0282] The audio component 1110 may receive data from one or more sensors capable of providing an audio signal (e.g., one or more microphones) to facilitate recognition of a user. The audio component 1110 may perform audio recognition on an audio signal to determine an identity of the user and associated user identifier. In some instances, aspects of device 110 and/or the system component(s) 120 may be configured at a computing device (e.g., a local server). Thus, in some instances, the audio component 1110 operating on a computing device may analyze all sound to facilitate recognition of a user. In some instances, the audio component 1110 may perform voice recognition to determine an identity of a user.

[0283] The audio component 1110 may also perform user identification based on audio data 111 input into the device 110 and/or the system component(s) 120 for speech processing. The audio component 1110 may determine scores indicating whether speech in the audio data 111 originated from particular users. For example, a first score may indicate a likelihood that speech in the audio data 111 originated from a first user associated with a first user identifier, a second score may indicate a likelihood that speech in the audio data 111 originated from a second user associated with a second user identifier, etc. The audio component 1110 may perform user recognition by comparing speech characteristics represented in the audio data 111 to stored speech characteristics of users (e.g., stored voice profiles associated with the device 110 that captured the spoken user input).

[0284] A presence detection component 594 may determine the presence and/or location of one or more users using a variety of data. As illustrated in FIG. 12, the presence detection component 594 may include one or more components, similar to (but potentially separate from) those used in the user recognition component 1295, such as a vision component 1108, an audio component 1110, a biometric component 1112, a radio frequency component 1114, or the like. The presence detection component 594 may use these components to determine the presence of users within an environment. The presence detection component 594 may base its operation on sensor data detected by a variety of devices, for example device such as those shown in FIG. 12 which may provide image data, audio data, RF data or even data from other sensors not expressly shown in FIG. 12 such as a RADAR sensor, LIDAR sensor, proximity sensor, etc.

[0285] Thus, in some instances, the presence detection component 594 may monitor data and determinations from one or more components to determine an identity of a user and/or a location of a user in an environment 1202. The presence detection component 594 may output user presence data 1295 which may indicate the presence of one or more users in an environment. The user presence data 1295 may also indicate a location of the user within the environment if the system has determined such information. The user presence data 1295 may also include a user identifier (e.g., user recognition data 1195) matched with location information as to where the system believes the particular user of the user identifier is located. Such data may rely on processing by the user recognition component 595.

[0286] The location information may include geographic information (such as an address, city, state, country, geo-position (e.g., GPS coordinates), velocity, latitude, longitude, altitude, or the like). The location information may also include a device identifier, zone identifier or environment identifier corresponding to a device / zone / environment the particular user is nearby / within. Output of the presence detection component 594 may be used to inform natural language component 560 processes as well as processing performed by skills 590, routing of output data, permission access to further information, etc. The details of the vision component 1108, the audio component 1110, the biometric component 1112, the radio frequency component 1114, the machine learning component 1216, and the presence confidence component 1218 are provided below following a description of the environment 1202.

[0287] In some instances, the environment 1202 may represent a home or office associated with a user 1220 “Alice” and/or a user 1222 “Bob.” In some instances, the user 1220 “Alice” may be associated with a computing device 1224, such as a smartphone. In some instances, the user 1222 “Bob” may be associated with a radio frequency device 1226, such as a wearable device (e.g., a smartwatch) or an identifier beacon.

[0288] The environment 1202 may include, but is not limited to, a number of devices that may be used to locate a user. For example, within zone 1201(1), the environment 1202 may include an imaging device 1228, an appliance 1230, a voice-controlled device 110a, and a computing device 1234. Within zone 1201(2), the environment 1202 may include a microphone 1236 and a motion sensor 1238. Within zone 1201(3), the environment may include an imaging device 1240, a television 1242, a speaker 1244, a set-top box 1246, a voice-controlled device 110b, a television 1250, and an access point 1252. Within zone 1201(4), the environment 1202 may include an appliance 1254, an imaging device 1256, a speaker 1258, a voice-controlled device 110c, and a microphone 1260.

[0289] Further, in some instances, the presence detection component 594 may have information regarding the layout of the environment 1202, include details regarding which devices are in which zones, the relationship between zones (e.g., which rooms are adjacent), and/or the placement of individual devices within each zone. In some instances, the presence detection component 594 can leverage knowledge of the relationships between zones and the devices within each zone to increase a confidence level of user identity and location as a user moves about the environment 1202. For example, in a case where the user 1222 is in zone 1201(3), and subsequently moves beyond a field of view of the imaging device 1240 into the zone 1201(2), the presence detection component 594 may infer a location and/or identity of the user to determine with a high confidence level (in combination with data from one or more other devices) that any motion detected by the motion sensor 1238 corresponds to movement by the user 1222.

[0290] In some instances, the vision component 1108 may receive data from one or more sensors capable of providing images (e.g., such as the imaging devices 1228, 1240, 1256 and the computing devices 1224 and 1234) or sensors indicating motion (e.g., such as the motion sensor 1238). In some instances, the vision component 1108 can perform facial recognition or image

analysis to determine an identity of a user and to associate that identity with a user profile associated with the user. In some instances, when a user (e.g., the user 1222 “Bob”) is facing the imaging device 1240, the vision component 1108 may perform facial recognition and identify the user 1222 with a high degree of confidence. In some instances, the vision component 1108 may have a low degree of confidence of an identity of a user, and the presence detection component 594 may utilize determinations from additional components to determine an identity and/or location of a user. In some instances, the vision component 1108 can be used in conjunction with other components to determine when a user is moving to a new location within the environment 1202. In some instances, the vision component 1108 can receive data from one or more imaging devices to determine a layout of a zone or room, and/or to determine which devices are in a zone and where they are located. In some instances, data from the vision component 1108 may be used with data from the audio component 1110 to identify what face appears to be speaking at the same time audio is captured by a particular device the user is facing for purposes of identifying a user who spoke an utterance.

[0291] In some instances, the environment 1202 may include biometric sensors that may transmit data to the biometric component 1112. For example, the biometric component 1112 may receive data corresponding to fingerprints, iris or retina scans, thermal scans, weights of users, a size of a user, pressure (e.g., within floor sensors), etc., and may determine a biometric profile corresponding to a user. In some instances, the biometric component 1112 may distinguish between a user and sound from a television, for example. Thus, the biometric component 1112 may incorporate biometric information into a confidence level for determining an identity and/or location of a user. In some instances, the biometric information from the biometric component 1112 can be associated with a specific user profile such that the biometric information uniquely identifies a user profile of a user (for example in conjunction with user recognition component 595).

[0292] In some instances, the radio frequency (RF) component 1114 may use RF localization to track devices that a user may carry or wear. For example, as discussed above, the user 1220 (and a user profile associated with the user) may be associated with a computing device 1224. The computing device 1224 may emit RF signals (e.g., Wi-Fi, Bluetooth®, etc.), which are illustrated as signals 1262 and 1264. As illustrated, the appliance 1254 may detect the signal 1262 and the access point 1252 may detect the signal 1264. In some instances, the access point

1252 and the appliance 1254 may indicate to the RF component 1114 the strength of the signals 1264 and 1262 (e.g., as a received signal strength indication (RSSI)), respectively. Thus, the RF component 1114 may compare the RSSI for various signals and for various appliances and may determine an identity and/or location of a user (with an associated confidence level). In some instances, the RF component 1114 may determine that a received RF signal is associated with a mobile device that is associated with a particular user. In some instances, a device (e.g., the access point 1252) may be configured with multiple antennas to determine a location of a user relative to the device using beamforming or spatial diversity techniques. In such a case, the RF component 1114 may receive an indication of the direction of the user relative to an individual device.

[0293] As illustrated, the appliance 1230 may receive a signal 1266 from the RF device 1226 associated with the user and a user profile, while the access point 1252 may receive a signal 1268. Further, the appliance 1254 can receive a signal 1270 from the RF device 1226. In an example where there is some uncertainty about an identity of the users in zones 1201(3) and 1201(4), the RF component 1114 may determine that the RSSI of the signals 1262, 1264, 1266, 1268, and/or 1270 increases or decreases a confidence level of an identity and/or location of the users, such as the user 1220 and 1224. For example, if an RSSI of the signal 1262 is higher than the RSSI of the signal 1270, the RF component may determine that it is more likely that a user in the zone 1201(4) is the user 1220 than the user 1222. In some instances, a confidence level of the determination may depend on a relative difference of the RSSIs, for example.

[0294] In some instances a voice controlled device 110, or another device proximate to the voice controlled device 110 may include some RF or other detection processing capabilities so that a user who speaks an utterance may scan, tap, or otherwise acknowledge his/her personal device (such as a phone) to a sensing device in the environment 1202. In this manner the user may “register” with the system for purposes of the system determining who spoke a particular utterance. Such a registration may occur prior to, during, or after speaking of an utterance.

[0295] In some instances, the audio component 1110 may receive data from one or more sensors capable of providing an audio signal (e.g., the voice-controlled devices 110a-c, the microphones 1236 and 1260, the computing devices 1224 and 1234, the set-top box 1246) to facilitate locating a user. In some instances, the audio component 1110 may perform audio

recognition on an audio signal to determine an identity of the user and an associated user profile. Further, in some instances, the imaging devices 1228, 1240, and 1256 may provide an audio signal to the audio component 1110. In some instances, the audio component 1110 is configured to receive an audio signal from one or more devices and may determine a sound level or volume of the source of the audio. In some instances, if multiple sources of audio are available, the audio component 1110 may determine that two audio signals correspond to the same source of sound, and may compare the relative amplitudes or volumes of the audio signal to determine a location of the source of sound. In some instances, individual devices may include multiple microphone and may determine a direction of a user with respect to an individual device. In some instances, aspects of the system component(s) 120 may be configured at a computing device (e.g., a local server) within the environment 1202. Thus, in some instances, the audio component 1110 operating on a computing device in the environment 1202 may analyze all sound within the environment 1202 (e.g., without requiring a wake word) to facilitate locating a user.

[0296] Components of a system that may be used to perform unit selection, parametric TTS processing, and/or model-based audio synthesis are shown in FIG. 13. FIG. 13 is a conceptual diagram that illustrates operations for generating synthesized speech using a TTS component 580, according to embodiments of the present disclosure. The TTS component 580 may receive text data 1315 and process it using one or more TTS models 1360 to generate TTS output data 1345 representing synthesized speech. In some embodiments the TTS output data 1345 may represent synthesized speech in the form of, for example, spectrogram data. In other embodiments, the TTS output data 1345 may comprise other data, for example data representing a latent representation (e.g., embedding data) representing synthesized speech. A vocoder 1390 or other component may convert the TTS output data 1345 into output speech audio data 1395, which may represent a time-domain waveform suitable for amplification and output as audio (e.g., from a loudspeaker).

[0297] The TTS component 580 may additionally receive other input data 1325. The other input data 1325 may include, for example, identifiers and/or labels corresponding to a desired speaker identity, voice characteristics, emotion, speech style, etc. desired for the synthesized speech. In some implementations, the other input data 1325 may include text tags or text metadata, that may indicate, for example, how specific words should be pronounced, for example

by indicating the desired output speech quality in tags formatted according to the speech synthesis markup language (SSML) or in some other form. For example, a first text tag may be included with text marking the beginning of when text should be whispered (e.g., <begin whisper>) and a second tag may be included with text marking the end of when text should be whispered (e.g., <end whisper>). The tags may be included in the text data 1315 and/or the other input data 1325 such as metadata accompanying a TTS request and indicating what text should be whispered (or have some other indicated audio characteristic).

[0298] The TTS component 580 may include a preprocessing component 1320 that can convert the text data 1315 and/or other input data 1325 into a form suitable for processing by the TTS model 1360. The text data 1315 may be from, for example an application, a skill component (described further below), an NLG component, another device or source, or may be input by a user. The text data 1315 received by the TTS component 580 may not necessarily be text, but may include other data (such as symbols, code, other data, etc.) that may reference text (such as an indicator of a word and/or phoneme) that is to be synthesized. The preprocessing component 1320 may transform the text data 1315 into, for example, a symbolic linguistic representation, which may include linguistic context features such as phoneme data, punctuation data, syllable-level features, word-level features, and/or emotion, speaker, accent, or other features for processing by the TTS component 580. The syllable-level features may include syllable emphasis, syllable speech rate, syllable inflection, or other such syllable-level features; the word-level features may include word emphasis, word speech rate, word inflection, or other such word-level features. The emotion features may include data corresponding to an emotion associated with the text data 1315, such as surprise, anger, or fear. The speaker features may include data corresponding to a type of speaker, such as sex, age, or profession. The accent features may include data corresponding to an accent associated with the speaker, such as Southern, Boston, English, French, or other such accent. Style features may include a book reading style, poem reading style, a news anchor style, a sports commentator style, various singing styles, etc.

[0299] The preprocessing component 1320 may include functionality and/or components for performing text normalization, linguistic analysis, linguistic prosody generation, or other such operations. During text normalization, the preprocessing component 1320 may first process the

text data 1315 and generate standard text, converting such things as numbers, abbreviations (such as Apt., St., etc.), symbols (\$, %, etc.) into the equivalent of written out words.

[0300] During linguistic analysis, the preprocessing component 1320 may analyze the language in the normalized text to generate a sequence of phonetic units corresponding to the input text. This process may be referred to as grapheme-to-phoneme conversion. Phonetic units include symbolic representations of sound units to be eventually combined and output by the system as speech. Various sound units may be used for dividing text for purposes of speech synthesis. In some implementations, the TTS model 1360 may process speech based on phonemes (individual sounds), half-phonemes, di-phones (the last half of one phoneme coupled with the first half of the adjacent phoneme), bi-phones (two consecutive phonemes), syllables, words, phrases, sentences, or other units. Each word may be mapped to one or more phonetic units. Such mapping may be performed using a language dictionary stored by the system, for example in a storage component. The linguistic analysis performed by the preprocessing component 1320 may also identify different grammatical components such as prefixes, suffixes, phrases, punctuation, syntactic boundaries, or the like. Such grammatical components may be used by the TTS component 580 to craft a natural-sounding audio waveform output. The language dictionary may also include letter-to-sound rules and other tools that may be used to pronounce previously unidentified words or letter combinations that may be encountered by the TTS component 580. Generally, the more information included in the language dictionary, the higher quality the speech output.

[0301] The output of the preprocessing component 1320 may be a symbolic linguistic representation, which may include a sequence of phonetic units. In some implementations, the sequence of phonetic units may be annotated with prosodic characteristics. In some implementations, prosody may be applied in part or wholly by a TTS model 1360. This symbolic linguistic representation may be sent to the TTS model 1360 for conversion into audio data (e.g., in the form of Mel-spectrograms or other frequency content data format).

[0302] The TTS component 580 may retrieve one or more previously trained and/or configured TTS models 1360 from the voice profile storage 1385. A TTS model 1360 may be, for example, a neural network architecture that may be described as interconnected artificial neurons or “cells” interconnected in layers and/or blocks. In general, neural network model

architecture can be described broadly by hyperparameters that describe the number of layers and/or blocks, how many cells each layer and/or block contains, what activations functions they implement, how they interconnect, etc. A neural network model includes trainable parameters (e.g., “weights”) that indicate how much weight (e.g., in the form of an arithmetic multiplier) a cell should give to a particular input when generating an output. In some implementations, a neural network model may include other features such as a self-attention mechanism, which may determine certain parameters at run time based on inputs rather than, for example, during training based on a loss calculation. The various data that describe a particular TTS model 1360 may be stored in the voice profile storage 1385. A TTS model 1360 may represent a particular speaker identity and may be conditioned based on speaking style, emotion, etc. In some implementations, a particular speaker identity may be associated with more than one TTS model 1360; for example, with a different model representing a different speaking style, language, emotion, etc. in some implementations, a particular TTS model 1360 may be associated with more than one speaker identity; that is, be able to produce synthesized speech that reproduces voice characteristics of more than one character. Thus a first TTS model 1360a may be used to create synthesized speech for the first speech-processing system component(s)a while a second, different, TTS model 1360b may be used to create synthesized speech for the second speech-processing system component(s)b. In some cases, the TTS model 1360 may generate the desired voice characteristics based on conditioning data received or determined from the text data 1315 and/or the other input data 1325. For example a synthesized voice of the first speech-processing system component(s)a may be different from a synthesized voice of the second speech-processing system component(s)b.

[0303] The TTS component 580 may, based on an indication received with the text data 1315 and/or other input data 1325, retrieve a TTS model 1360 from the voice profile storage 1385 and use it to process input to generate synthesized speech. The TTS component 580 may provide the TTS model 1360 with any relevant conditioning labels to generate synthesized speech having the desired voice characteristics. The TTS model 1360 may generate spectrogram data 1345 (e.g., frequency content data) representing the synthesized speech, and send it to the vocoder 1390 for conversion into an audio signal.

[0304] The TTS component 580 may generate other output data 1355. The other output data 1355 may include, for example, indications or instructions for handling and/or outputting the

synthesized speech. For example, the text data 1315 and/or other input data 1325 may be received along with metadata, such as SSML tags, indicating that a selected portion of the text data 1315 should be louder or quieter. Thus, the other output data 1355 may include a volume tag that instructs the vocoder 1390 to increase or decrease an amplitude of the output speech audio data 1395 at times corresponding to the selected portion of the text data 1315.

Additionally or alternatively, a volume tag may instruct a playback device to raise or lower a volume of the synthesized speech from the device's current volume level, or lower a volume of other media being output by the device (e.g., to deliver an urgent message).

[0305] The vocoder 1390 may convert the spectrogram data 1345 generated by the TTS model 1360 into an audio signal (e.g., an analog or digital time-domain waveform) suitable for amplification and output as audio. The vocoder 1390 may be, for example, a universal neural vocoder based on Parallel WaveNet or related model. The vocoder 1390 may take as input audio data in the form of, for example, a Mel-spectrogram with 80 coefficients and frequencies ranging from 50Hz to 12kHz. The synthesized speech audio data 1395 may be a time-domain audio format (e.g., pulse-code modulation (PCM), waveform audio format (WAV), μ -law, etc.) that may be readily converted to an analog signal for amplification and output by a loudspeaker. The synthesized speech audio data 1395 may consist of, for example, 8-, 16-, or 24-bit audio having a sample rate of 16kHz, 24kHz, 44.1kHz, etc. In some implementations, other bit and/or sample rates may be used.

[0306] Various machine learning techniques may be used to train and operate models to perform various steps described herein, such as user recognition, sentiment detection, image processing, dialog management, etc. Models may be trained and operated according to various machine learning techniques. Such techniques may include, for example, neural networks (such as deep neural networks and/or recurrent neural networks), inference engines, trained classifiers, etc. Examples of trained classifiers include Support Vector Machines (SVMs), neural networks, decision trees, AdaBoost (short for "Adaptive Boosting") combined with decision trees, and random forests. Focusing on SVM as an example, SVM is a supervised learning model with associated learning algorithms that analyze data and recognize patterns in the data, and which are commonly used for classification and regression analysis. Given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that assigns new examples into one category or the other, making it a non-probabilistic binary linear

classifier. More complex SVM models may be built with the training set identifying more than two categories, with the SVM determining which category is most similar to input data. An SVM model may be mapped so that the examples of the separate categories are divided by clear gaps. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gaps they fall on. Classifiers may issue a “score” indicating which category the data most closely matches. The score may provide an indication of how closely the data matches the category.

[0307] In order to apply the machine learning techniques, the machine learning processes themselves need to be trained. Training a machine learning component such as, in this case, one of the first or second models, requires establishing a “ground truth” for the training examples. In machine learning, the term “ground truth” refers to the accuracy of a training set’s classification for supervised learning techniques. Various techniques may be used to train the models including backpropagation, statistical learning, supervised learning, semi-supervised learning, stochastic learning, or other known techniques.

[0308] FIG. 14 is a block diagram conceptually illustrating a device 110 that may be used with the system. FIG. 15 is a block diagram conceptually illustrating example components of a remote device, such as the natural language command processing system component(s), which may assist with ASR processing, NLU processing, etc., and a skill system component(s) 525. A system (120/525) may include one or more servers. A “server” as used herein may refer to a traditional server as understood in a server / client computing structure but may also refer to a number of different computing components that may assist with the operations discussed herein. For example, a server may include one or more physical computing components (such as a rack server) that are connected to other devices / components either physically and/or over a network and is capable of performing computing operations. A server may also include one or more virtual machines that emulates a computer system and is run on one or across multiple devices. A server may also include other combinations of hardware, software, firmware, or the like to perform operations discussed herein. The server(s) may be configured to operate using one or more of a client-server model, a computer bureau model, grid computing techniques, fog computing techniques, mainframe techniques, utility computing techniques, a peer-to-peer model, sandbox techniques, or other computing techniques.

[0309] While the device 110 may operate locally to a user (e.g., within a same environment so the device may receive inputs and playback outputs for the user) the server / system component(s) may be located remotely from the device 110 as its operations may not require proximity to the user. The server / system component(s) may be located in an entirely different location from the device 110 (for example, as part of a cloud computing system or the like) or may be located in a same environment as the device 110 but physically separated therefrom (for example a home server or similar device that resides in a user's home or business but perhaps in a closet, basement, attic, or the like). The server / system component(s) 120 may also be a version of a user device 110 that includes different (e.g., more) processing capabilities than other user device(s) 110 in a home / office. One benefit to the server / system component(s) being in a user's home / business is that data used to process a command / return a response may be kept within the user's home, thus reducing potential privacy concerns.

[0310] Multiple system components (120/525) may be included in the overall system 100 of the present disclosure, such as one or more natural language processing system component(s) 120 for performing ASR processing, one or more natural language processing system component(s) 120 for performing NLU processing, one or more skill system component(s) 525, etc. In operation, each of these systems may include computer-readable and computer-executable instructions that reside on the respective device (120/525), as will be discussed further below.

[0311] Each of these devices (110/120/525) may include one or more controllers/processors (1404/1504), which may each include a central processing unit (CPU) for processing data and computer-readable instructions, and a memory (1406/1506) for storing data and instructions of the respective device. The memories (1406/1506) may individually include volatile random access memory (RAM), non-volatile read only memory (ROM), non-volatile magnetoresistive memory (MRAM), and/or other types of memory. Each device (110/120/525) may also include a data storage component (1408/1508) for storing data and controller/processor-executable instructions. Each data storage component (1408/1508) may individually include one or more non-volatile storage types such as magnetic storage, optical storage, solid-state storage, etc. Each device (110/120/525) may also be connected to removable or external non-volatile memory and/or storage (such as a removable memory card, memory key drive, networked storage, etc.) through respective input/output device interfaces (1402/1502).

[0312] Computer instructions for operating each device (110/120/525) and its various components may be executed by the respective device's controller(s)/processor(s) (1404/1504), using the memory (1406/1506) as temporary "working" storage at runtime. A device's computer instructions may be stored in a non-transitory manner in non-volatile memory (1406/1506), storage (1408/1508), or an external device(s). Alternatively, some or all of the executable instructions may be embedded in hardware or firmware on the respective device in addition to or instead of software.

[0313] Each device (110/120/525) includes input/output device interfaces (1402/1502). A variety of components may be connected through the input/output device interfaces (1402/1502), as will be discussed further below. Additionally, each device (110/120/525) may include an address/data bus (1424/1524) for conveying data among components of the respective device. Each component within a device (110/120/525) may also be directly connected to other components in addition to (or instead of) being connected to other components across the bus (1424/1524).

[0314] Referring to FIG. 14, the device 110 may include input/output device interfaces 1402 that connect to a variety of components such as an audio output component such as a speaker 1412, a wired headset or a wireless headset (not illustrated), or other component capable of outputting audio. The device 110 may also include an audio capture component. The audio capture component may be, for example, a microphone 1420 or array of microphones, a wired headset or a wireless headset (not illustrated), etc. If an array of microphones is included, approximate distance to a sound's point of origin may be determined by acoustic localization based on time and amplitude differences between sounds captured by different microphones of the array. The device 110 may additionally include a display 1416 for displaying content. The device 110 may further include a camera 1418.

[0315] Via antenna(s) 1422, the input/output device interfaces 1402 may connect to one or more networks 199 via a wireless local area network (WLAN) (such as Wi-Fi) radio, Bluetooth, and/or wireless network radio, such as a radio capable of communication with a wireless communication network such as a Long Term Evolution (LTE) network, WiMAX network, 3G network, 4G network, 5G network, etc. A wired connection such as Ethernet may also be supported. Through the network(s) 199, the system may be distributed across a networked

environment. The I/O device interface (1402/1502) may also include communication components that allow data to be exchanged between devices such as different physical servers in a collection of servers or other components.

[0316] The components of the device(s) 110, the natural language command processing system component(s), or a skill system component(s) 525 may include their own dedicated processors, memory, and/or storage. Alternatively, one or more of the components of the device(s) 110, the natural language command processing system component(s), or a skill system component(s) 525 may utilize the I/O interfaces (1402/1502), processor(s) (1404/1504), memory (1406/1506), and/or storage (1408/1508) of the device(s) 110, natural language command processing system component(s), or the skill system component(s) 525, respectively. Thus, the ASR component 550 may have its own I/O interface(s), processor(s), memory, and/or storage; the NLU component 560 may have its own I/O interface(s), processor(s), memory, and/or storage; and so forth for the various components discussed herein.

[0317] As noted above, multiple devices may be employed in a single system. In such a multi-device system, each of the devices may include different components for performing different aspects of the system's processing. The multiple devices may include overlapping components. The components of the device 110, the natural language command processing system component(s), and a skill system component(s) 525, as described herein, are illustrative, and may be located as a stand-alone device or may be included, in whole or in part, as a component of a larger device or system. As can be appreciated, a number of components may exist either on a system component(s) and/or on device 110. For example, language processing 592 (which may include ASR 550), language output 593 (which may include NLG 579 and TTS 580), etc., for example as illustrated in FIGS. 5. Unless expressly noted otherwise, the system version of such components may operate similarly to the device version of such components and thus the description of one version (e.g., the system version or the local version) applies to the description of the other version (e.g., the local version or system version) and vice-versa.

[0318] As illustrated in FIG. 16, multiple devices (110a-110n, 120, 525) may contain components of the system and the devices may be connected over a network(s) 199. The network(s) 199 may include a local or private network or may include a wide network such as the Internet. Devices may be connected to the network(s) 199 through either wired or wireless

connections. For example, a speech-detection device 110a, a smart phone 110b, a smart watch 110c, a tablet computer 110d, a vehicle 110e, a speech-detection device with display 110f, a display / smart television 110g, a washer/dryer 110h, a refrigerator 110i, a microwave 110j, autonomously motile device 110k (e.g., a robot), etc., may be connected to the network(s) 199 through a wireless service provider, over a Wi-Fi or cellular network connection, or the like.

Other devices are included as network-connected support devices, such as the natural language command processing system component(s) 120, the skill system component(s) 525, and/or others. The support devices may connect to the network(s) 199 through a wired connection or wireless connection. Networked devices may capture audio using one-or-more built-in or connected microphones or other audio capture devices, with processing performed by ASR components, NLU components, or other components of the same device or another device connected via the network(s) 199, such as the ASR component 550, the NLU component 560, etc. of the natural language command processing system component(s) 120.

[0319] The concepts disclosed herein may be applied within a number of different devices and computer systems, including, for example, general-purpose computing systems, speech processing systems, and distributed computing environments.

[0320] The materials herein may also be understood in view of the following clauses.

[0321] 1. A computer-implemented method comprising: operating a first device using a first keyword component configured to detect a wakeword in received audio, the first wakeword component comprising a first convolutional neural network (CNN) encoder, first convolutional recurrent neural network (CRNN) decoder data corresponding to a first wakeword, and second convolutional recurrent neural network decoder data corresponding to a second wakeword; receiving user recognition data corresponding to the first device; processing the user recognition data to identify user profile data corresponding to a first user; determining the user profile data indicates the first user is associated with the first wakeword but not the second wakeword; in response to the first user being associated with the first wakeword, causing the first CRNN decoder data to be activated; in response to the first user not being associated with the second wakeword, causing the second CRNN decoder data to be deactivated; receiving input audio data representing an utterance; processing the input audio data using the first CNN encoder to determine first encoded audio data comprising a plurality of feature vectors representing acoustic

units of the utterance; in response to the first CRNN decoder data being activated, processing the first encoded audio data using the first CRNN decoder data to determine a first likelihood the utterance included the first wakeword; in response to the second CRNN decoder data being deactivated, not processing the first encoded audio data using the second CRNN decoder data; and based at least in part on the first likelihood, causing speech processing to be performed using data representing the utterance.

[0322] 2. The computer-implemented method of clause 1, wherein: the first CRNN decoder data corresponds to a first CRNN model; the second CRNN decoder data corresponds to a second CRNN model; processing the first encoded audio data using the first CRNN decoder data comprises, in response to the first CRNN decoder data being activated, sending the first encoded audio data to the first CRNN model for processing; and the first likelihood is output by the first CRNN model.

[0323] 3. The computer-implemented method of clause 1 or 2, wherein: the first CRNN decoder data corresponds to a first portion of a first CRNN model; the second CRNN decoder data corresponds to a second portion of the first CRNN model; the method comprises sending the first encoded audio data to the first CRNN model for processing; and not processing the first encoded audio data using the second CRNN decoder data comprises not activating the second CRNN decoder data during processing of the first encoded audio data using the first CRNN model.

[0324] 4. The computer-implemented method of clause 1, 2, or 3, further comprising: determining a request to configure the first device to detect a third keyword; determining an updated keyword component data comprising: first data representing the first CNN encoder, the first CRNN decoder data, the second CRNN decoder data, and third CRNN decoder data corresponding to the third keyword; and using the updated keyword component data to configure an updated keyword component for operation by the first device.

[0325] 5. A computer-implemented method comprising: determining first data corresponding to a first context of a first device; receiving audio data representing audio detected by the first device; determining the first context corresponds to a first sound corresponding to a first action; processing the audio data using a first encoder to determine first encoded audio data; in response to the first context corresponding to the first sound, activating first decoder data

corresponding to the first sound, wherein the first device corresponds to a plurality of decoder data corresponding to processing output from the first encoder, the plurality of decoder data including the first decoder data corresponding to the first sound and second decoder data corresponding to a second sound; in response to the first decoder data being active, processing
5 the first encoded audio data using the first decoder data to determine second data representing a first likelihood the audio included the first sound; and based at least in part on the second data, causing the first action to be performed.

[0326] 6. The computer-implemented method of clause 5, further comprising: failing to determine that the first context corresponds to the second sound; and in response to the first
10 context not corresponding to the second sound, maintaining the second decoder data in an inactive state.

[0327] 7. The computer-implemented method of clause 6, wherein: the first decoder data corresponds to a first model; the second decoder data corresponds to a second model; processing the first encoded audio data using the first decoder data comprises, in response to the first
15 decoder data being activated, sending the first encoded audio data to the first model for processing; and the first likelihood is output by the first model.

[0328] 8. The computer-implemented method of clause 6 or 7, wherein: the first decoder data corresponds to a first portion of a first model; the second decoder data corresponds to a second portion of the first model; the method comprises sending the first encoded audio data to
20 the first model for processing; and maintaining the second decoder data in an inactive state comprises disabling the second portion of the first model.

[0329] 9. The computer-implemented method of clause 5, 6, 7, or 8, wherein the first data comprises user recognition data and determining the first context corresponds to the first sound comprises: processing the user recognition data to determine a first user is in an environment of
25 the first device; determining user profile data corresponding to the first user; and determining the user profile data indicates the first user is associated with first sound.

[0330] 10. The computer-implemented method of clause 5, 6, 7, 8, or 9, further comprising: determining the first data indicates a first application is operating with respect to the first device; and determining the first context corresponds to the first sound comprises determining the first
30 application is associated with the first sound.

[0331] 11. The computer-implemented method of clause 5, 6, 7, 8, 9, or 10, further comprising: determining the first data indicates a network connection status corresponding to the first device; and determining the first context corresponds to the first sound comprises determining the network connection status is associated with the first sound.

5 **[0332]** 12. The computer-implemented method of clause 5, 6, 7, 8, 9, 10, or 11, wherein: the audio data represents a plurality of audio frames; the method comprises: determining a first portion of the audio data corresponding to a first subset of the plurality of audio frames, processing, by the first encoder, the first portion to determine a first feature vector representing at least one acoustic unit represented by the first subset, determining a second portion of the
10 audio data corresponding to a second subset of the plurality of audio frames, and processing, by the first encoder, the second portion to determine a second feature vector representing at least one acoustic unit represented by the second subset; and the first encoded audio data includes the first feature vector and the second feature vector.

[0333] 13. A system, comprising: at least one processor; and at least one memory comprising
15 instructions that, when executed by the at least one processor, cause the system to: determine first data corresponding to a first context of a first device; receive audio data representing audio detected by the first device; determine the first context corresponds to a first sound corresponding to a first action; process the audio data using a first encoder to determine first encoded audio data; in response to the first context corresponding to the first sound, activate first
20 decoder data corresponding to the first sound, wherein the first device corresponds to a plurality of decoder data corresponding to processing output from the first encoder, the plurality of decoder data including the first decoder data corresponding to the first sound and second decoder data corresponding to a second sound; in response to the first decoder data being active, process the first encoded audio data using the first decoder data to determine second data representing a
25 first likelihood the audio included the first sound; and based at least in part on the second data, cause the first action to be performed.

[0334] 14. The system of clause 13, wherein the at least one memory further comprises instructions that, when executed by the at least one processor, further cause the system to: fail to determine that the first context corresponds to the second sound; and in response to the first

context not corresponding to the second sound, maintain the second decoder data in an inactive state.

[0335] 15. The system of clause 14, wherein: the first decoder data corresponds to a first model; the second decoder data corresponds to a second model; processing the first encoded audio data using the first decoder data comprises, in response to the first decoder data being activated, sending the first encoded audio data to the first model for processing; and the first likelihood is output by the first model.

[0336] 16. The system of clause 14 or 15, wherein: the first decoder data corresponds to a first portion of a first model; the second decoder data corresponds to a second portion of the first model; the at least one memory further comprises instructions that, when executed by the at least one processor, further cause the system to send the first encoded audio data to the first model for processing; and the instructions that cause the system to maintain the second decoder data in an inactive state comprise instructions that, when executed by the at least one processor, cause the system to disable the second portion of the first model.

[0337] 17. The system of clause 13, 14, 15, or 16, wherein the first data comprises user recognition data and wherein the instructions that cause the system to determine the first context corresponds to the first sound comprise instructions that, when executed by the at least one processor, further cause the system to: process the user recognition data to determine a first user is in an environment of the first device; determine user profile data corresponding to the first user; and determine the user profile data indicates the first user is associated with first sound.

[0338] 18. The system of clause 13, 14, 15, 16, or 17, wherein the at least one memory further comprises instructions that, when executed by the at least one processor, further cause the system to: determine the first data indicates a first application is operating with respect to the first device; and determine the first context corresponds to the first sound comprises determining the first application is associated with the first sound.

[0339] 19. The system of clause 13, 14, 15, 16, 17, or 18, wherein the at least one memory further comprises instructions that, when executed by the at least one processor, further cause the system to: determine the first data indicates a network connection status corresponding to the first device; and determine the first context corresponds to the first sound comprises determining the network connection status is associated with the first sound.

[0340] 20. The system of clause 13, 14, 15, 16, 17, 18, or 19, wherein: the audio data represents a plurality of audio frames; the at least one memory further comprises instructions that, when executed by the at least one processor, further cause the system to: determine a first portion of the audio data corresponding to a first subset of the plurality of audio frames, process, by the first encoder, the first portion to determine a first feature vector representing at least one acoustic unit represented by the first subset, determine a second portion of the audio data corresponding to a second subset of the plurality of audio frames, and process, by the first encoder, the second portion to determine a second feature vector representing at least one acoustic unit represented by the second subset; and the first encoded audio data includes the first feature vector and the second feature vector.

[0341] The above aspects of the present disclosure are meant to be illustrative. They were chosen to explain the principles and application of the disclosure and are not intended to be exhaustive or to limit the disclosure. Many modifications and variations of the disclosed aspects may be apparent to those of skill in the art. Persons having ordinary skill in the field of computers and speech processing should recognize that components and process steps described herein may be interchangeable with other components or steps, or combinations of components or steps, and still achieve the benefits and advantages of the present disclosure. Moreover, it should be apparent to one skilled in the art, that the disclosure may be practiced without some or all of the specific details and steps disclosed herein. Further, unless expressly stated to the contrary, features / operations / components, etc. from one embodiment discussed herein may be combined with features / operations / components, etc. from another embodiment discussed herein.

[0342] Aspects of the disclosed system may be implemented as a computer method or as an article of manufacture such as a memory device or non-transitory computer readable storage medium. The computer readable storage medium may be readable by a computer and may comprise instructions for causing a computer or other device to perform processes described in the present disclosure. The computer readable storage medium may be implemented by a volatile computer memory, non-volatile computer memory, hard drive, solid-state memory, flash drive, removable disk, and/or other media. In addition, components of system may be implemented as in firmware or hardware.

[0343] Conditional language used herein, such as, among others, "can," "could," "might," "may," "e.g.," and the like, unless specifically stated otherwise, or otherwise understood within the context as used, is generally intended to convey that certain embodiments include, while other embodiments do not include, certain features, elements and/or steps. Thus, such conditional language is not generally intended to imply that features, elements, and/or steps are in any way required for one or more embodiments or that one or more embodiments necessarily include logic for deciding, with or without other input or prompting, whether these features, elements, and/or steps are included or are to be performed in any particular embodiment. The terms "comprising," "including," "having," and the like are synonymous and are used inclusively, in an open-ended fashion, and do not exclude additional elements, features, acts, operations, and so forth. Also, the term "or" is used in its inclusive sense (and not in its exclusive sense) so that when used, for example, to connect a list of elements, the term "or" means one, some, or all of the elements in the list.

[0344] Disjunctive language such as the phrase "at least one of X, Y, Z," unless specifically stated otherwise, is understood with the context as used in general to present that an item, term, etc., may be either X, Y, or Z, or any combination thereof (e.g., X, Y, and/or Z). Thus, such disjunctive language is not generally intended to, and should not, imply that certain embodiments require at least one of X, at least one of Y, or at least one of Z to each be present.

[0345] As used in this disclosure, the term "a" or "one" may include one or more items unless specifically stated otherwise. Further, the phrase "based on" is intended to mean "based at least in part on" unless specifically stated otherwise.

CLAIMS

WHAT IS CLAIMED IS:

1. A computer-implemented method comprising:
determining first data corresponding to a first context of a first device;
5 receiving audio data representing audio detected by the first device;
determining the first context corresponds to a first sound corresponding to a first action;
processing the audio data using a first encoder to determine first encoded audio data;
in response to the first context corresponding to the first sound, activating first decoder
data corresponding to the first sound, wherein the first device corresponds to a plurality of
10 decoder data corresponding to processing output from the first encoder, the plurality of decoder
data including the first decoder data corresponding to the first sound and second decoder data
corresponding to a second sound;
in response to the first decoder data being active, processing the first encoded audio data
using the first decoder data to determine second data representing a first likelihood the audio
15 included the first sound; and
based at least in part on the second data, causing the first action to be performed.
2. The computer-implemented method of claim 1, further comprising:
failing to determine that the first context corresponds to the second sound; and
20 in response to the first context not corresponding to the second sound, maintaining the
second decoder data in an inactive state.
3. The computer-implemented method of claim 2, wherein:
the first decoder data corresponds to a first model;
25 the second decoder data corresponds to a second model;
processing the first encoded audio data using the first decoder data comprises, in response
to the first decoder data being activated, sending the first encoded audio data to the first model
for processing; and
the first likelihood is output by the first model.
30
4. The computer-implemented method of claim 2 or 3, wherein:

the first decoder data corresponds to a first portion of a first model;
the second decoder data corresponds to a second portion of the first model;
the method comprises sending the first encoded audio data to the first model for
processing; and

5 maintaining the second decoder data in an inactive state comprises disabling the second
portion of the first model.

5. The computer-implemented method of claim 1, 2, 4, or 4, wherein the first data
comprises user recognition data and determining the first context corresponds to the first sound
10 comprises:

 processing the user recognition data to determine a first user is in an environment of the
first device;

 determining user profile data corresponding to the first user; and

 determining the user profile data indicates the first user is associated with first sound.

15

6. The computer-implemented method of claim 1, 2, 3, 4, or 5, further comprising:
 determining the first data indicates a first application is operating with respect to the first
device; and

 determining the first context corresponds to the first sound comprises determining the
20 first application is associated with the first sound.

7. The computer-implemented method of claim 1, 2, 3, 4, 5, or 6, further comprising:
 determining the first data indicates a network connection status corresponding to the first
device; and

25 determining the first context corresponds to the first sound comprises determining the
network connection status is associated with the first sound.

8. The computer-implemented method of claim 1, 2, 3, 4, 5, 6, or 7, wherein:
 the audio data represents a plurality of audio frames;
30 the method comprises:

determining a first portion of the audio data corresponding to a first subset of the plurality of audio frames,

processing, by the first encoder, the first portion to determine a first feature vector representing at least one acoustic unit represented by the first subset,

5 determining a second portion of the audio data corresponding to a second subset of the plurality of audio frames, and

processing, by the first encoder, the second portion to determine a second feature vector representing at least one acoustic unit represented by the second subset; and the first encoded audio data includes the first feature vector and the second feature vector.

10 9. A system, comprising:

at least one processor; and

at least one memory comprising instructions that, when executed by the at least one processor, cause the system to:

15 determine first data corresponding to a first context of a first device;

receive audio data representing audio detected by the first device;

determine the first context corresponds to a first sound corresponding to a first action;

process the audio data using a first encoder to determine first encoded audio data;

20 in response to the first context corresponding to the first sound, activate first decoder data corresponding to the first sound, wherein the first device corresponds to a plurality of decoder data corresponding to processing output from the first encoder, the plurality of decoder data including the first decoder data corresponding to the first sound and second decoder data corresponding to a second sound;

25 in response to the first decoder data being active, process the first encoded audio data using the first decoder data to determine second data representing a first likelihood the audio included the first sound; and

based at least in part on the second data, cause the first action to be performed.

30 10. The system of claim 9, wherein the at least one memory further comprises instructions that, when executed by the at least one processor, further cause the system to:

fail to determine that the first context corresponds to the second sound; and
in response to the first context not corresponding to the second sound, maintain the
second decoder data in an inactive state.

- 5 11. The system of claim 10, wherein:
the first decoder data corresponds to a first model;
the second decoder data corresponds to a second model;
processing the first encoded audio data using the first decoder data comprises, in response
to the first decoder data being activated, sending the first encoded audio data to the first model
10 for processing; and
the first likelihood is output by the first model.
12. The system of claim 10 or 11, wherein:
the first decoder data corresponds to a first portion of a first model;
15 the second decoder data corresponds to a second portion of the first model;
the at least one memory further comprises instructions that, when executed by the at least
one processor, further cause the system to send the first encoded audio data to the first model for
processing; and
the instructions that cause the system to maintain the second decoder data in an inactive
20 state comprise instructions that, when executed by the at least one processor, cause the system to
disable the second portion of the first model.
13. The system of claim 9, 10, 11, or 12, wherein the first data comprises user recognition
data and wherein the instructions that cause the system to determine the first context corresponds
25 to the first sound comprise instructions that, when executed by the at least one processor, further
cause the system to:
process the user recognition data to determine a first user is in an environment of the first
device;
determine user profile data corresponding to the first user; and
30 determine the user profile data indicates the first user is associated with first sound.

14. The system of claim 9, 10, 11, 12, or 13, wherein the at least one memory further comprises instructions that, when executed by the at least one processor, further cause the system to:

5 determine the first data indicates a first application is operating with respect to the first device; and

determine the first context corresponds to the first sound comprises determining the first application is associated with the first sound.

15. The system of claim 9, 10, 11, 12, 13, or 14, wherein the at least one memory further comprises instructions that, when executed by the at least one processor, further cause the system to:

10 determine the first data indicates a network connection status corresponding to the first device; and

15 determine the first context corresponds to the first sound comprises determining the network connection status is associated with the first sound.

FIG. 1A

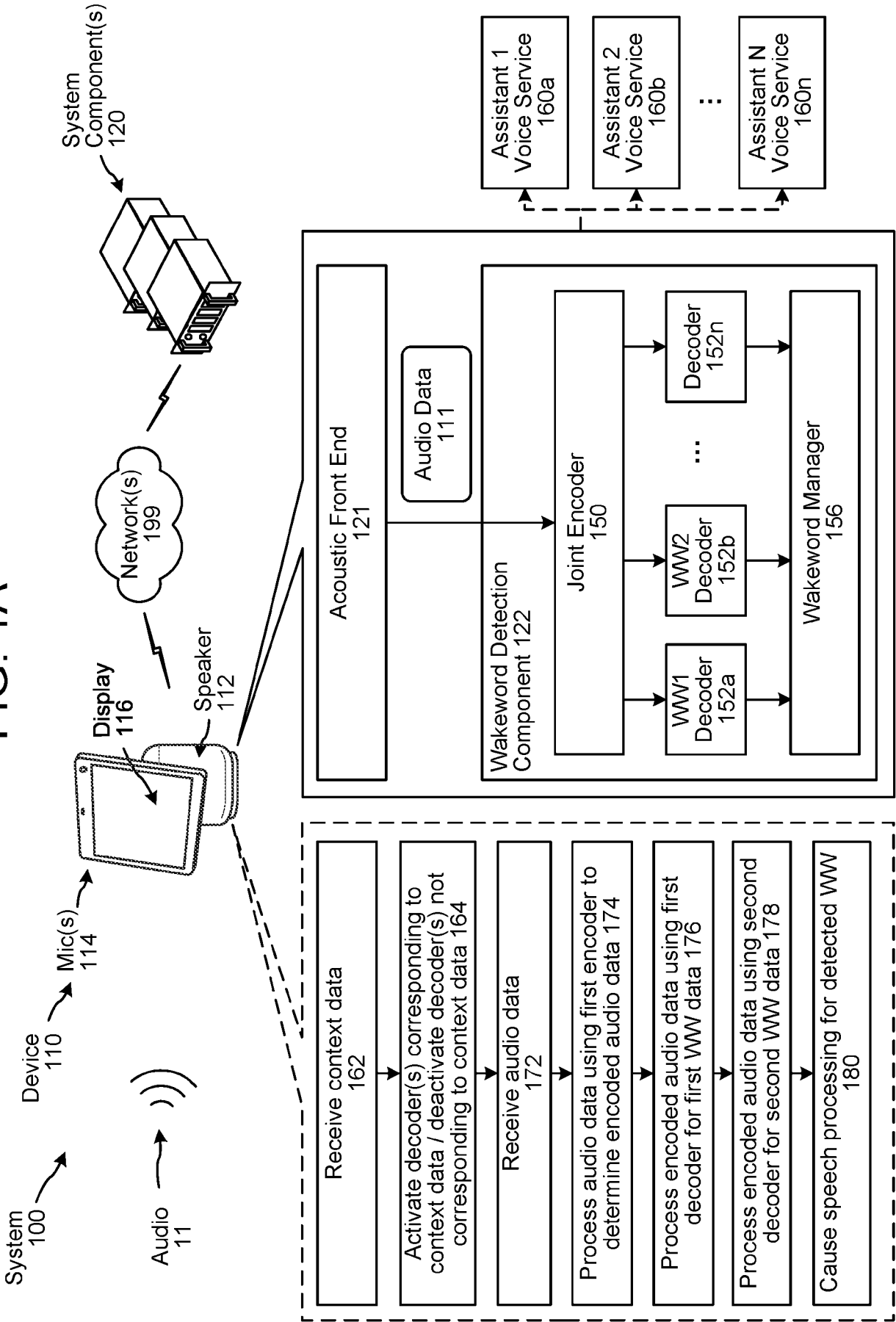


FIG. 1B

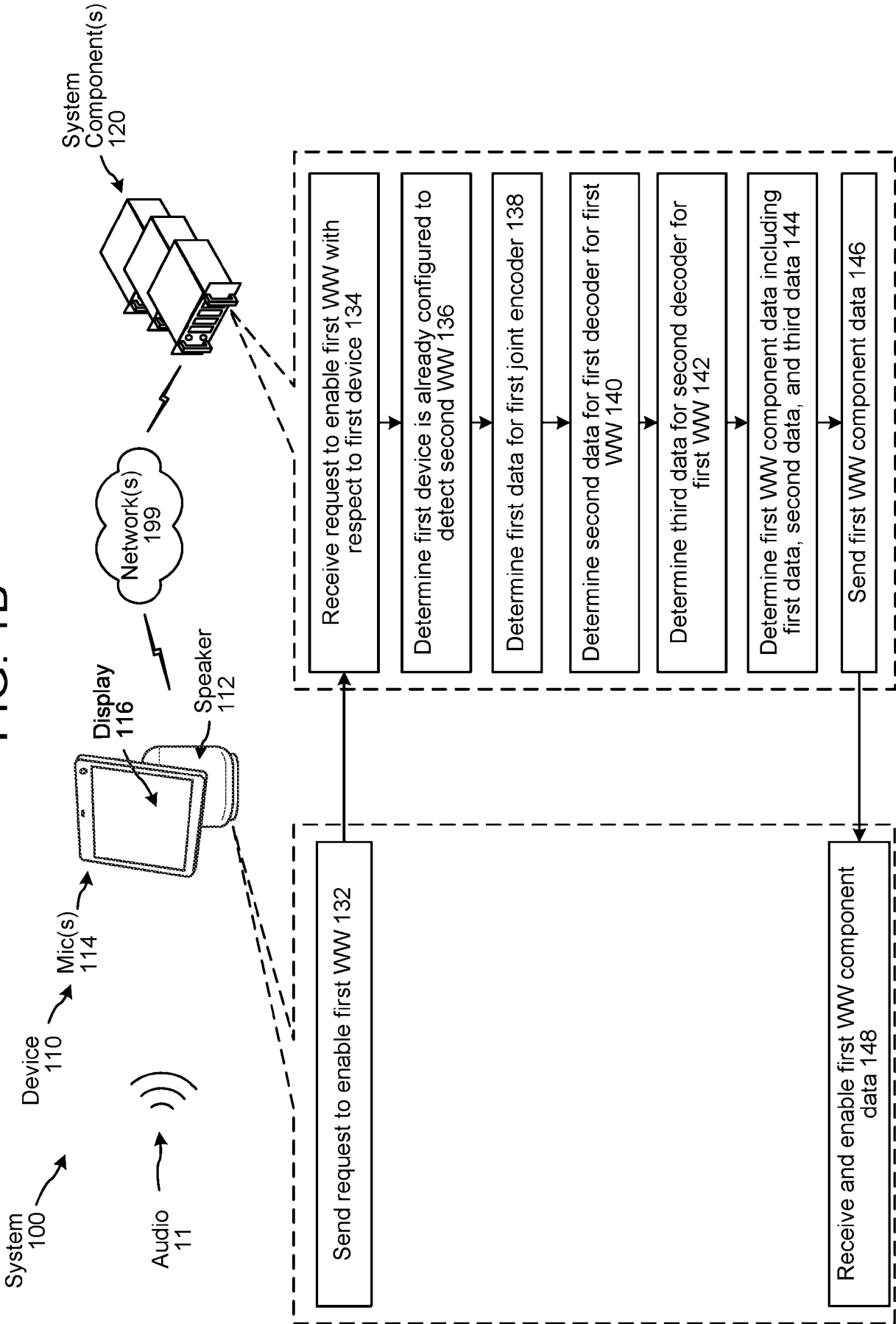


FIG. 2A

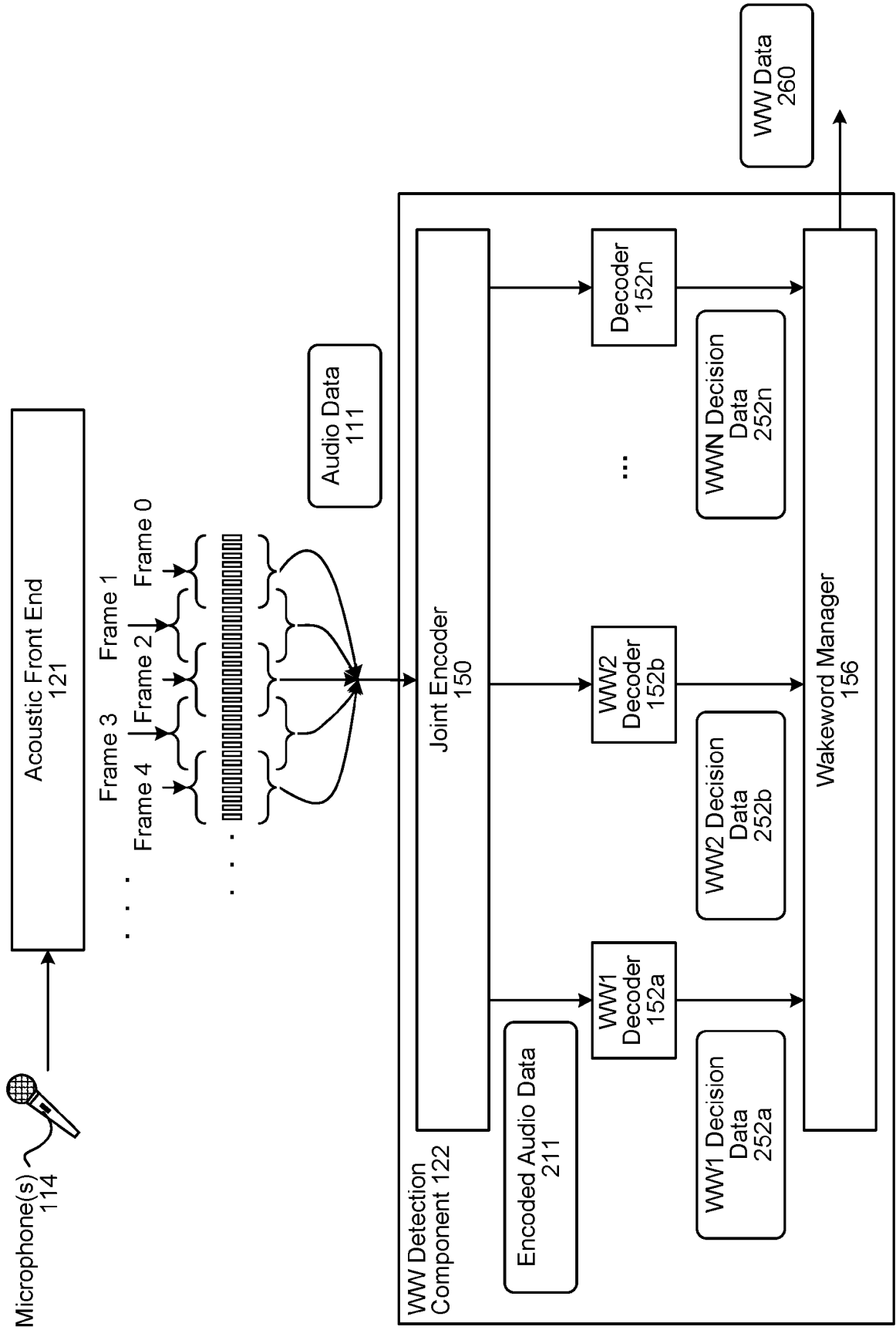


FIG. 2B

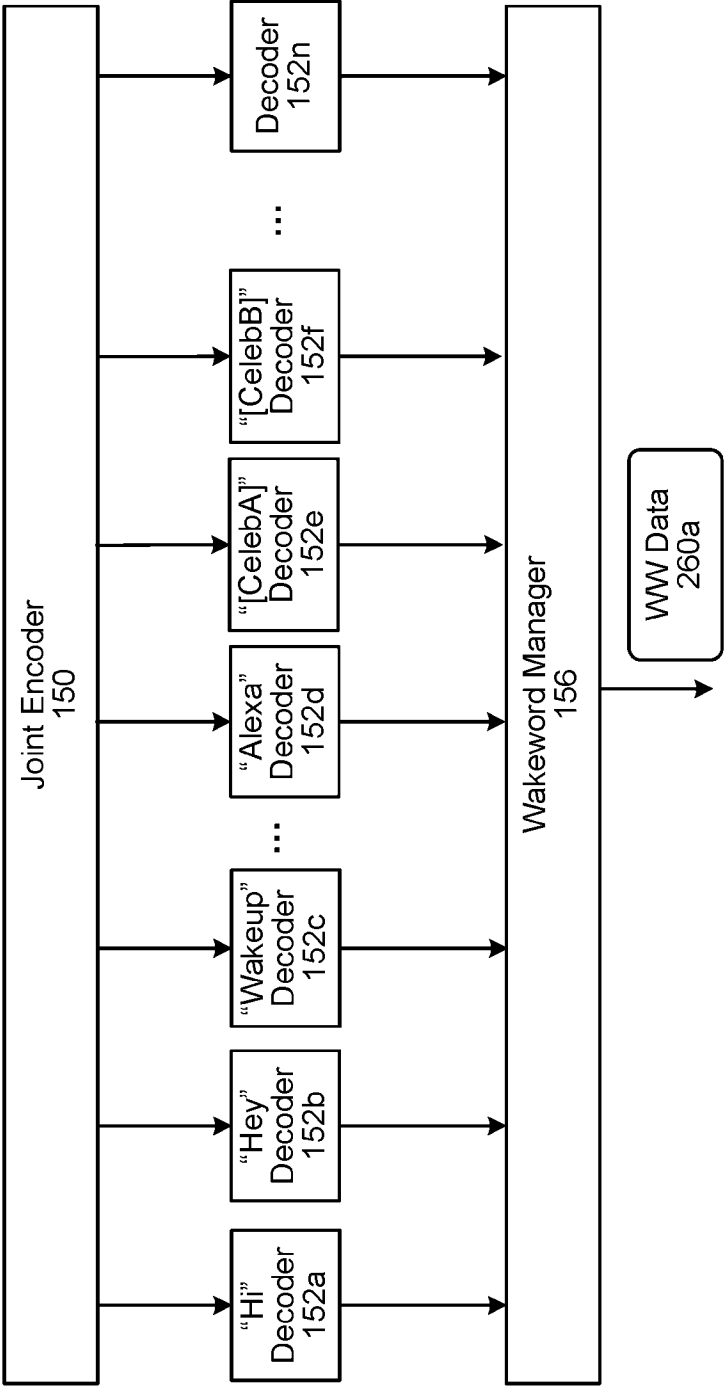


FIG. 2C

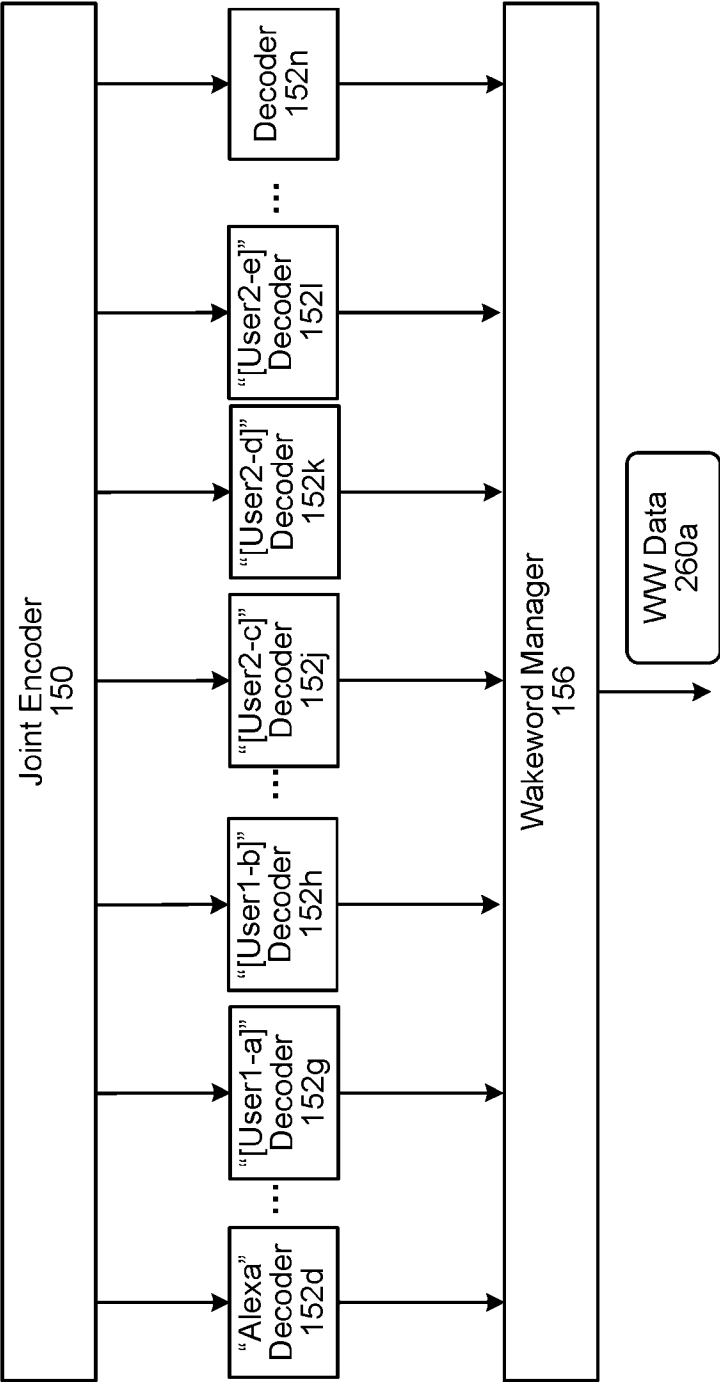


FIG. 2D

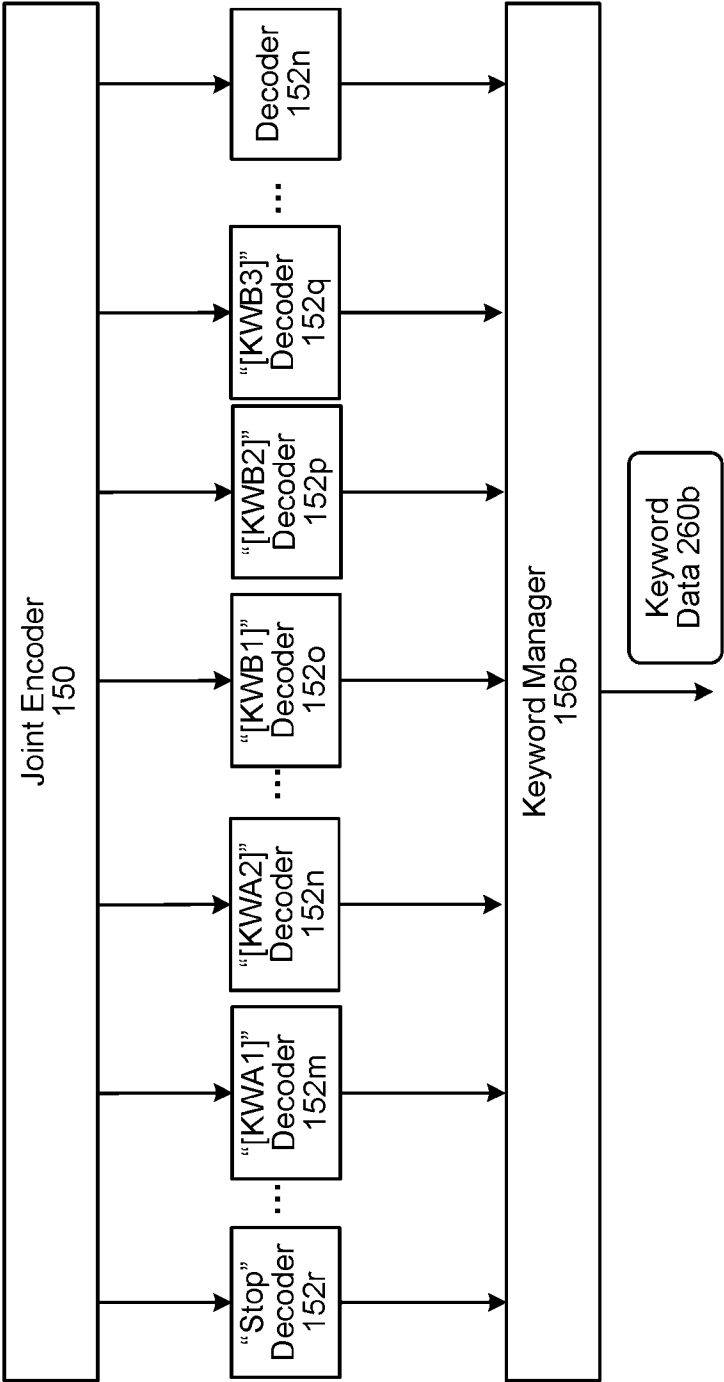


FIG. 2E

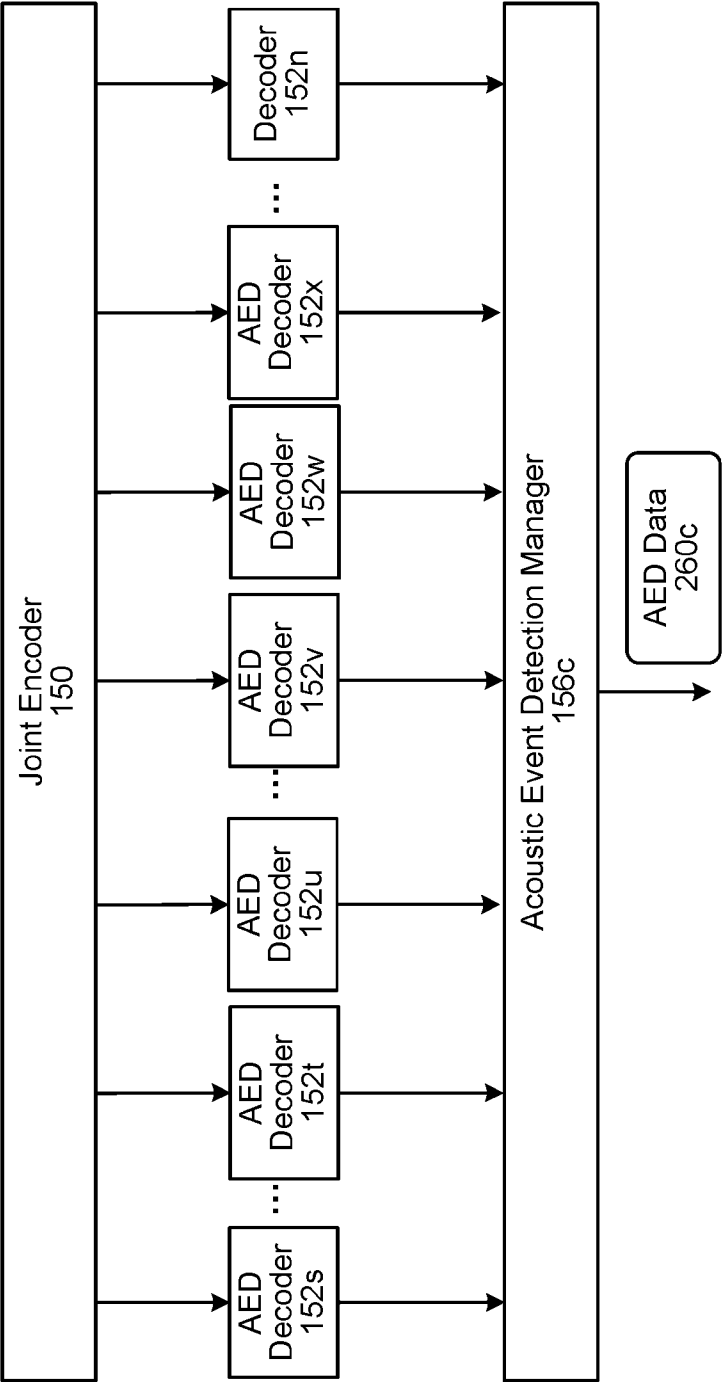


FIG. 2F

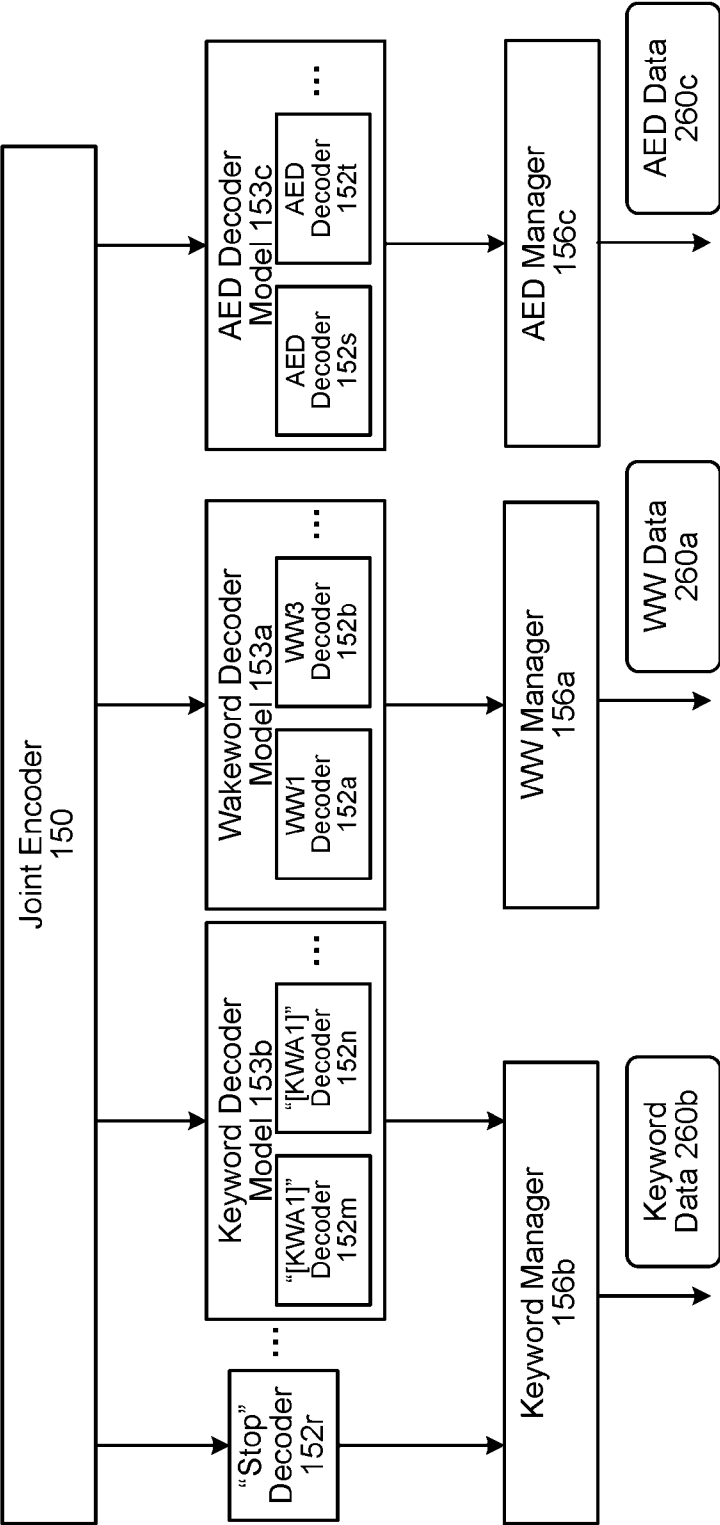


FIG. 2G

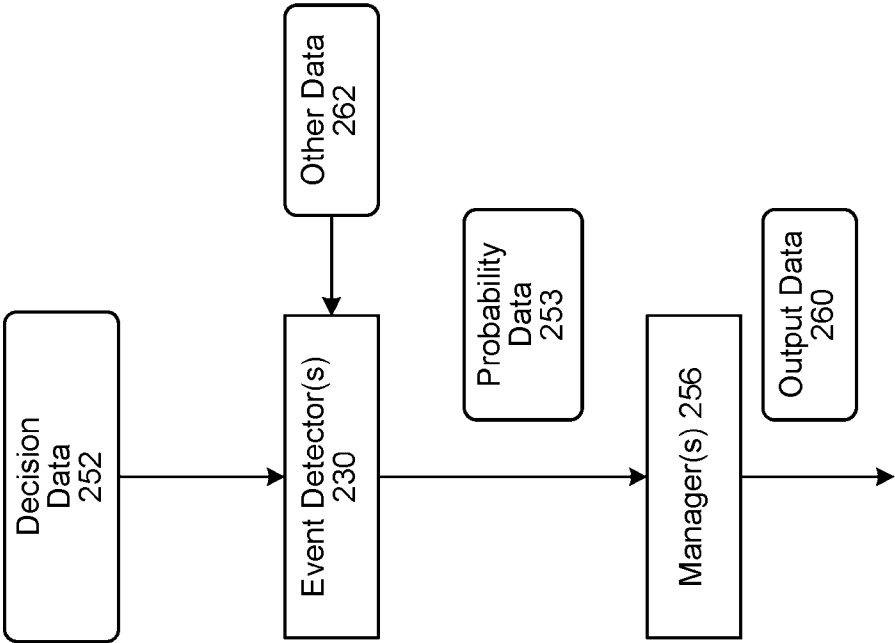


FIG. 3A

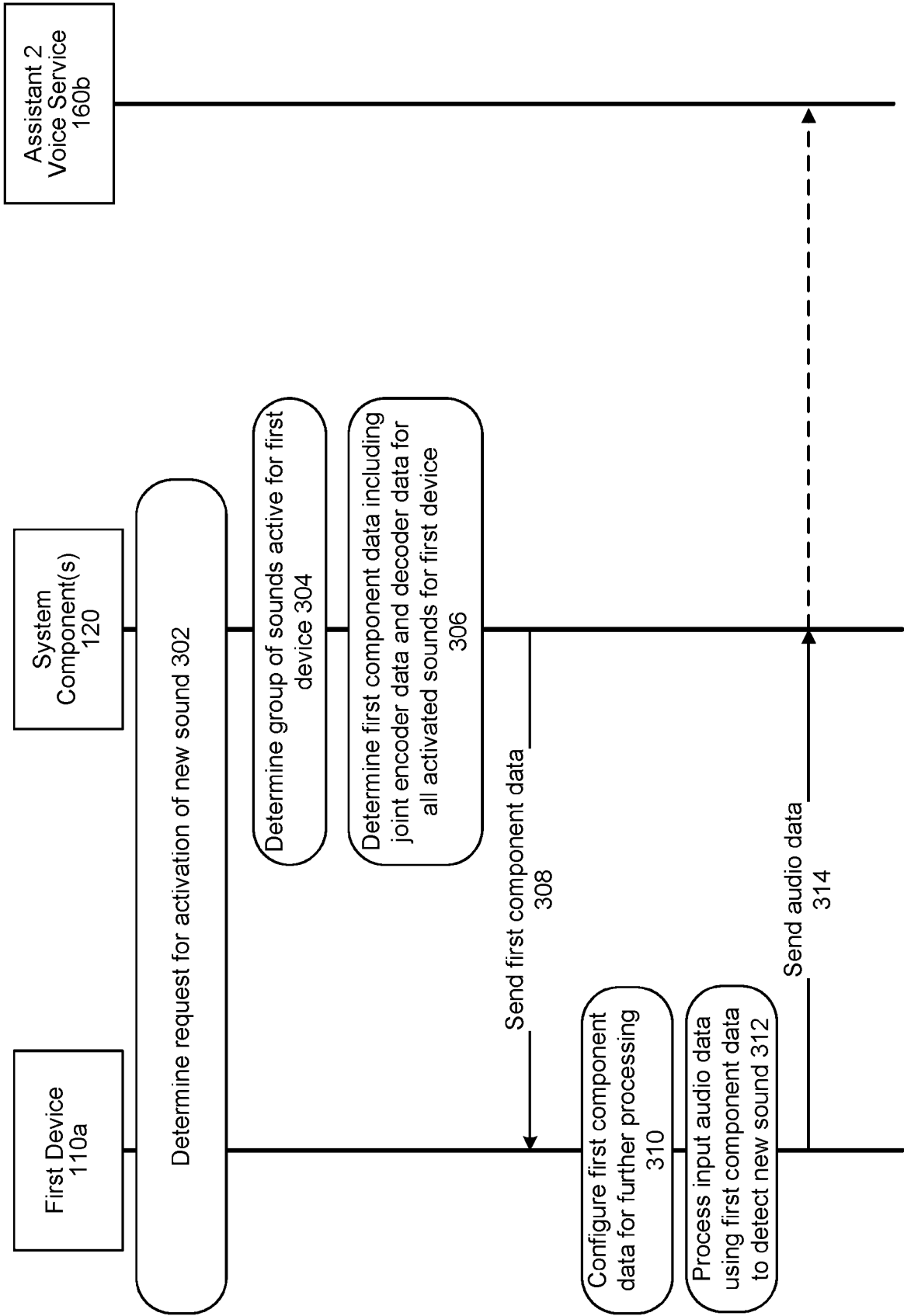


FIG. 3B

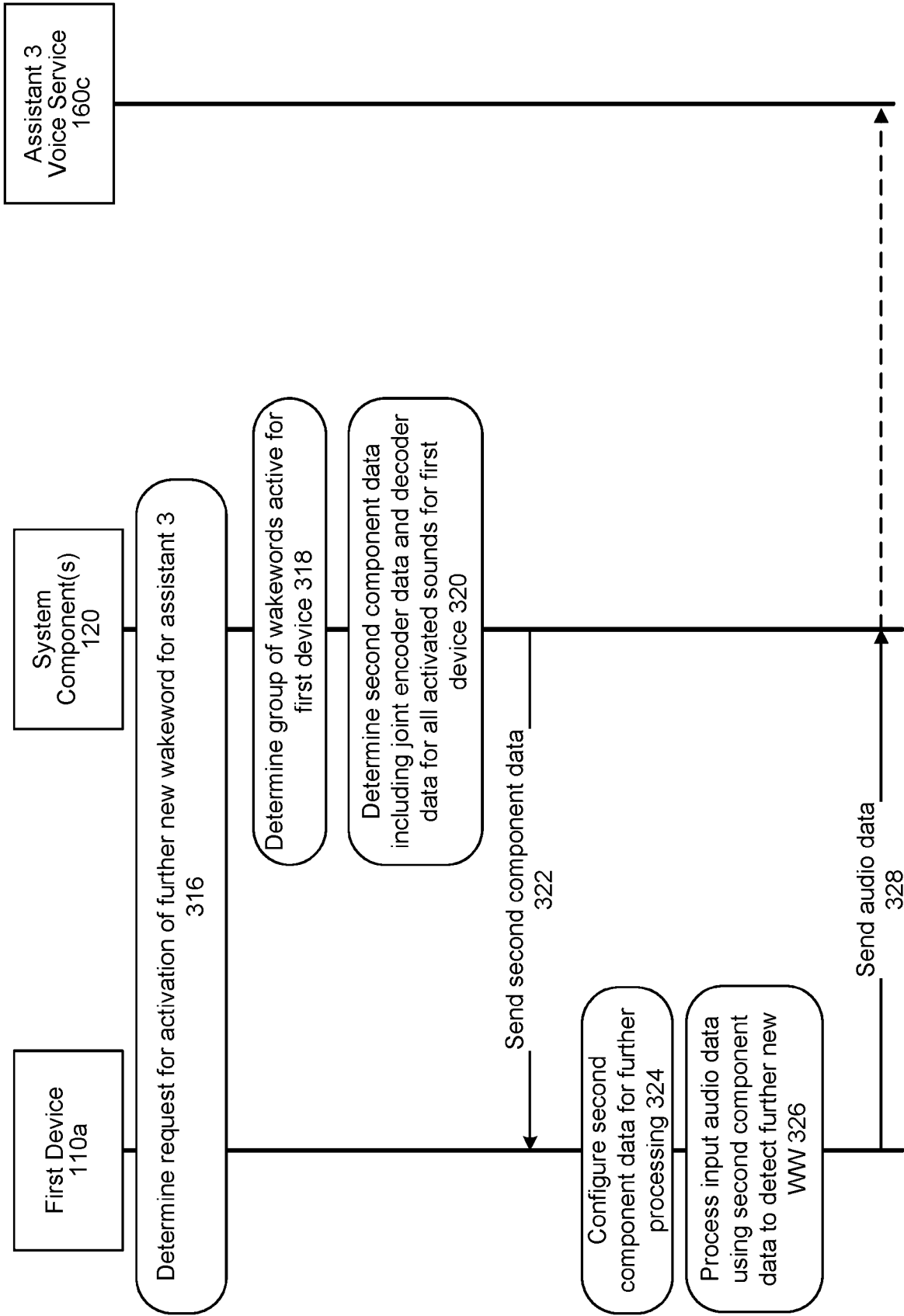


FIG. 4

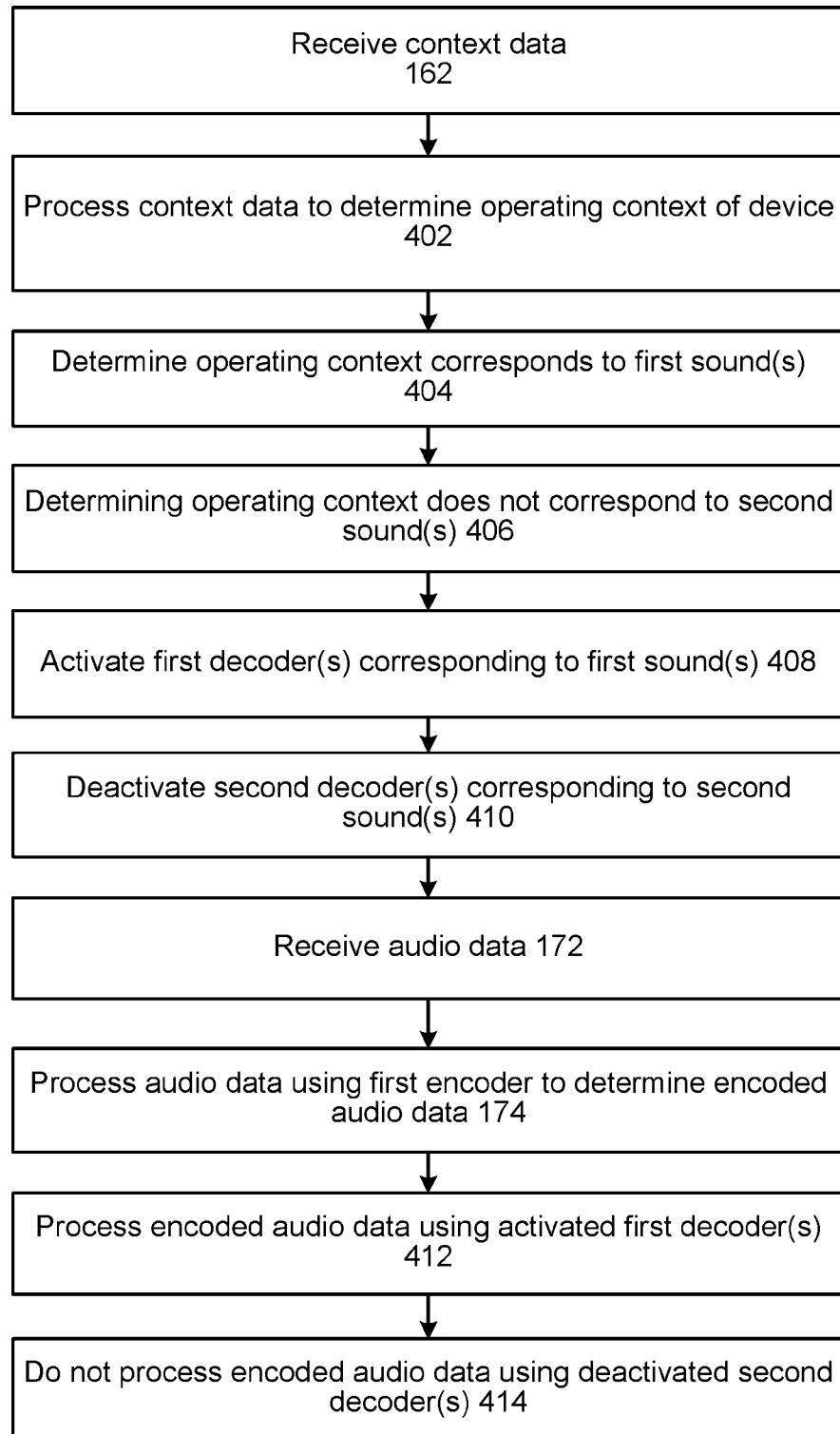


FIG. 5A

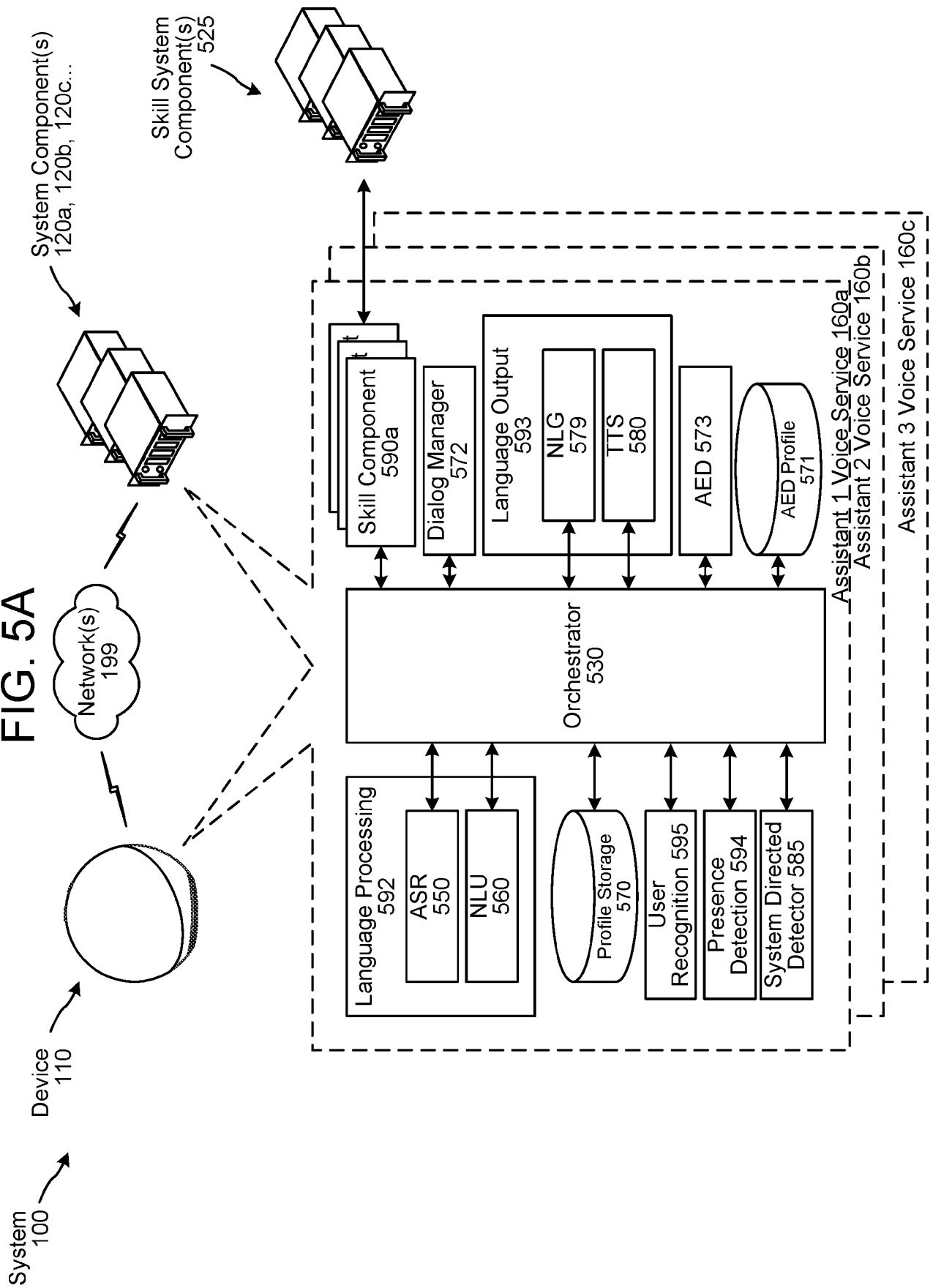


FIG. 5B

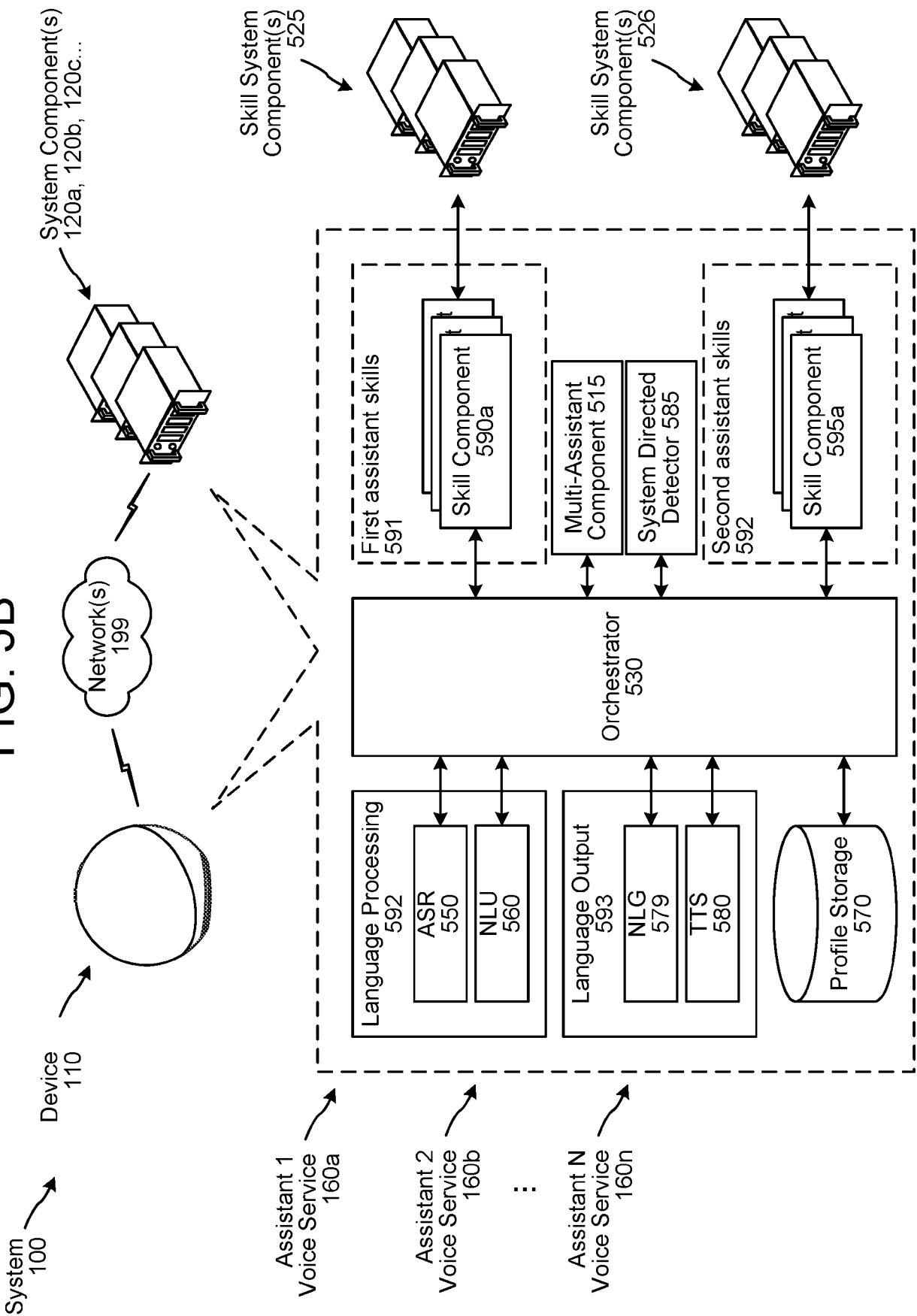


FIG. 6

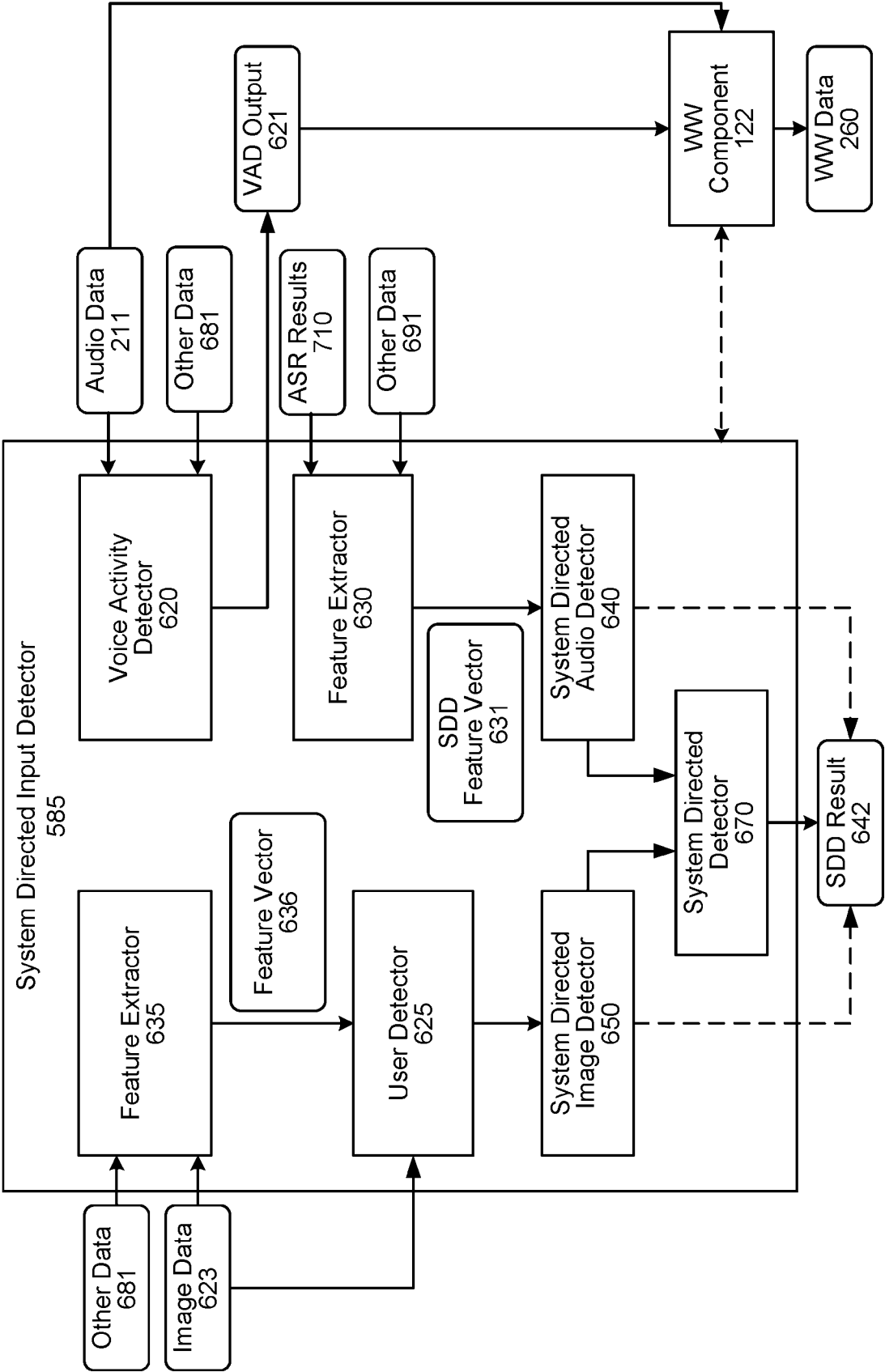


FIG. 7

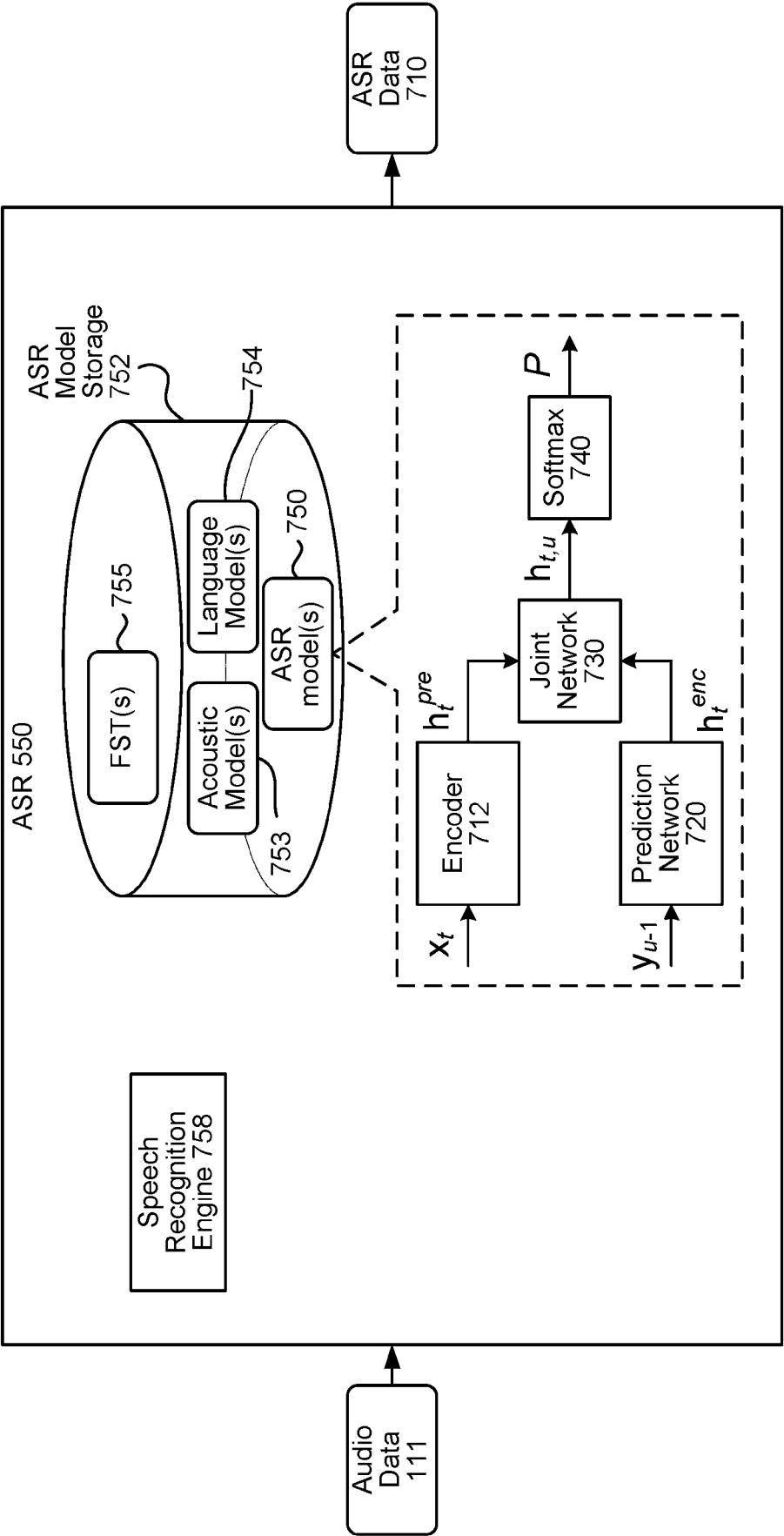


FIG. 8

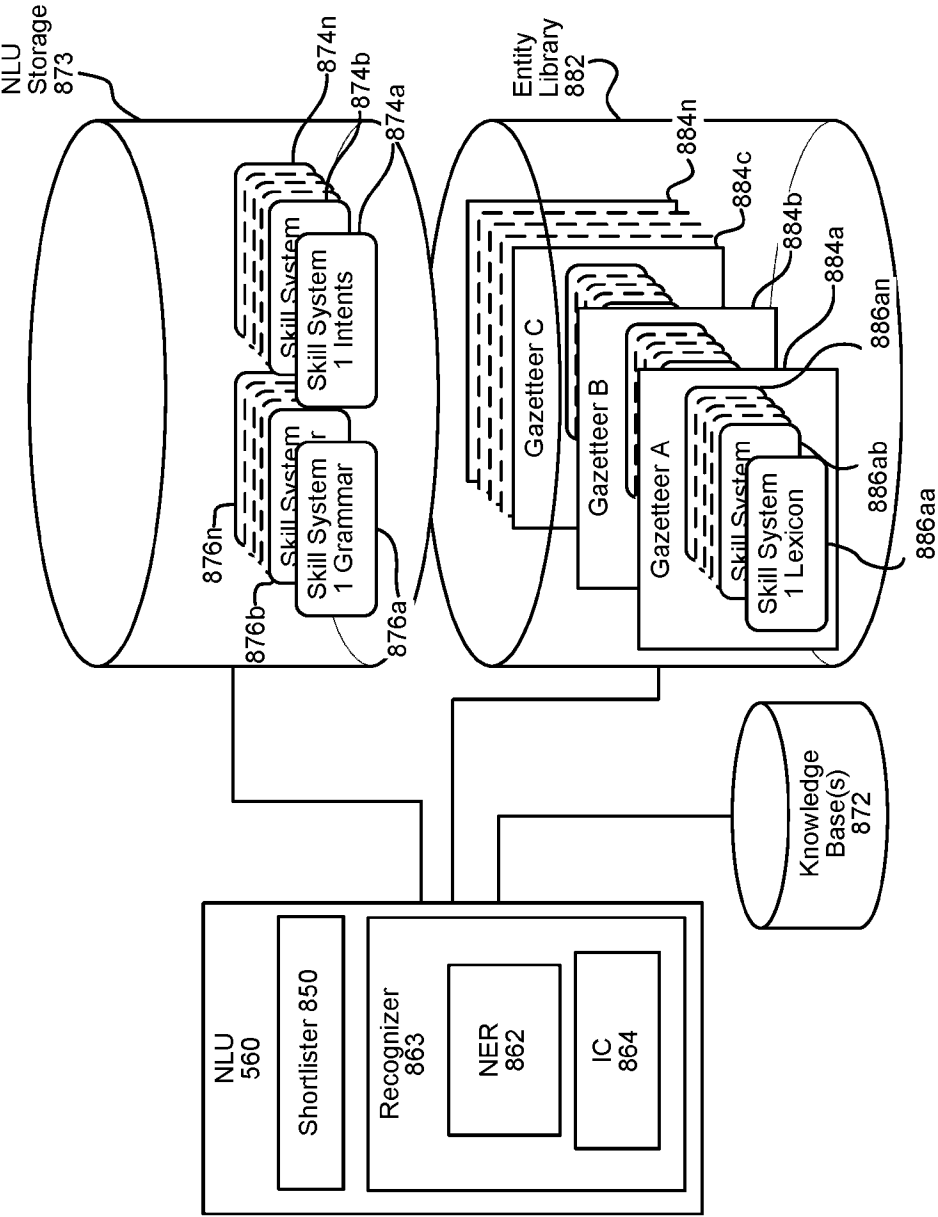


FIG. 9

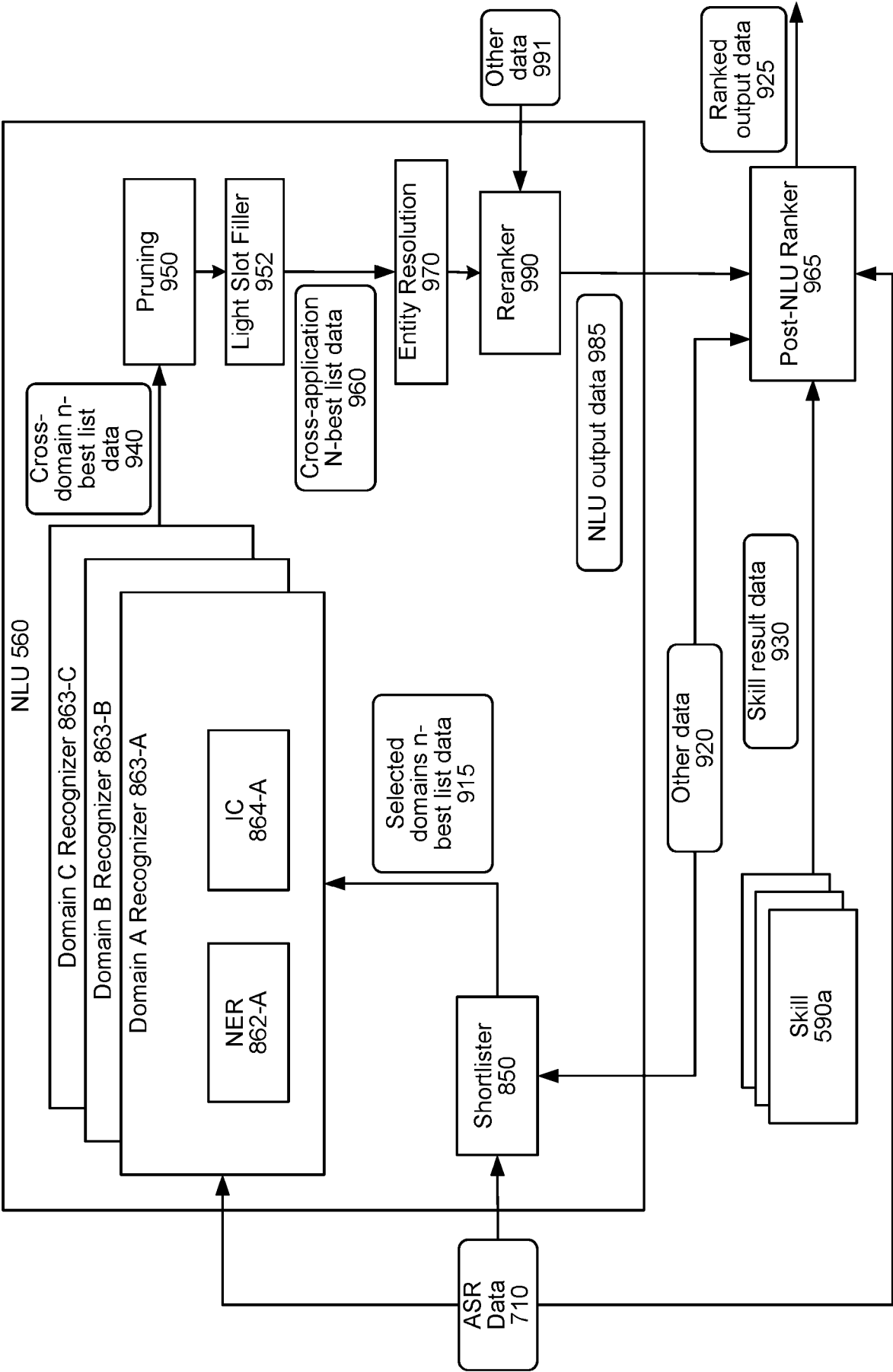


FIG. 10

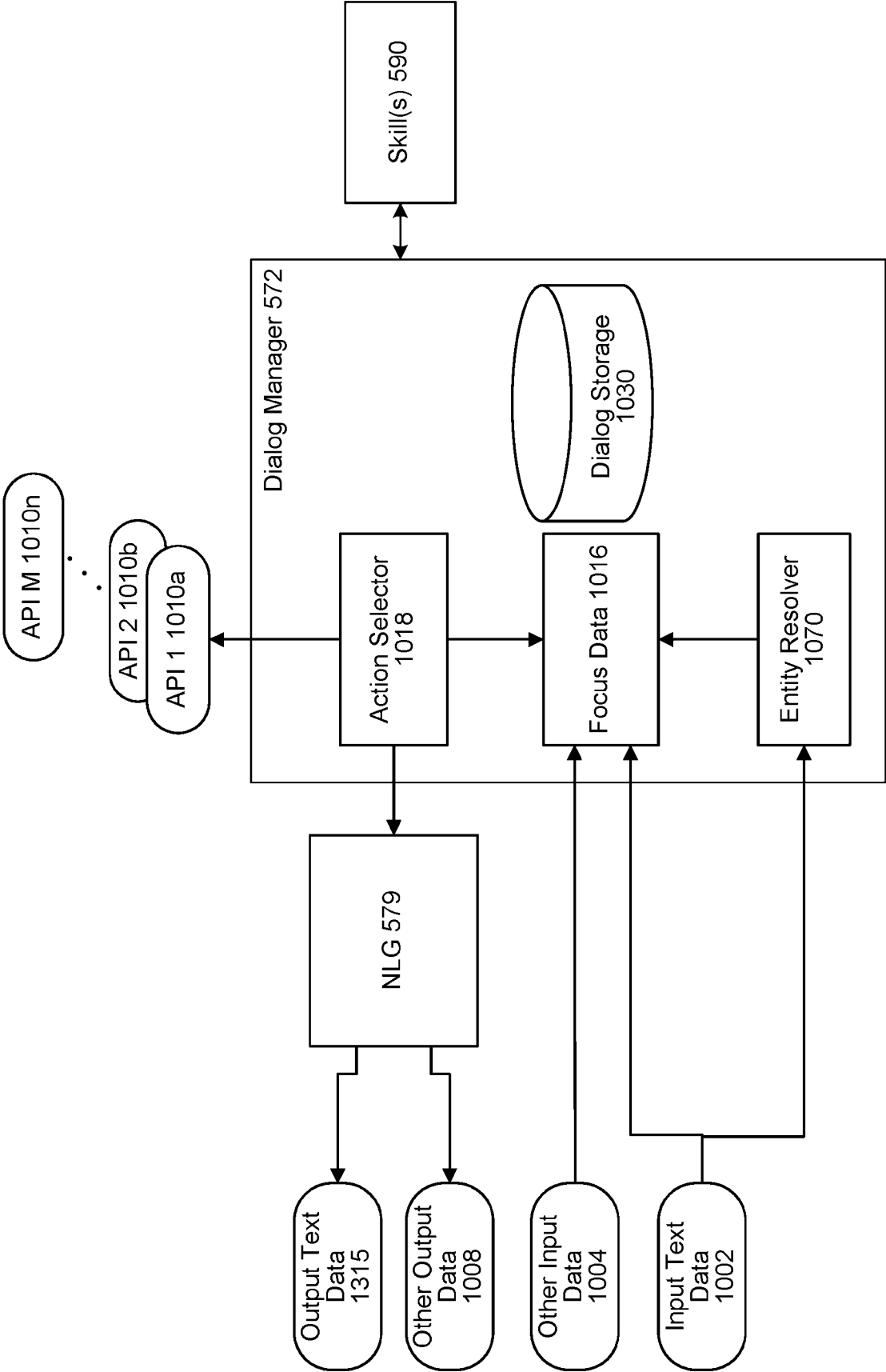


FIG. 11

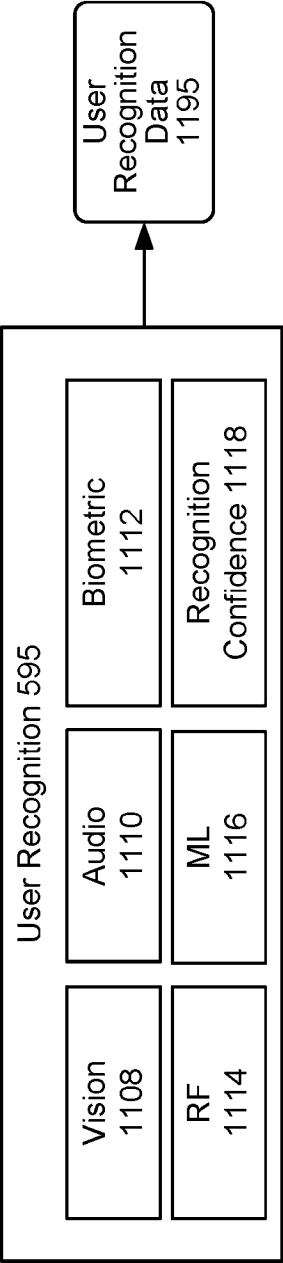


FIG. 12

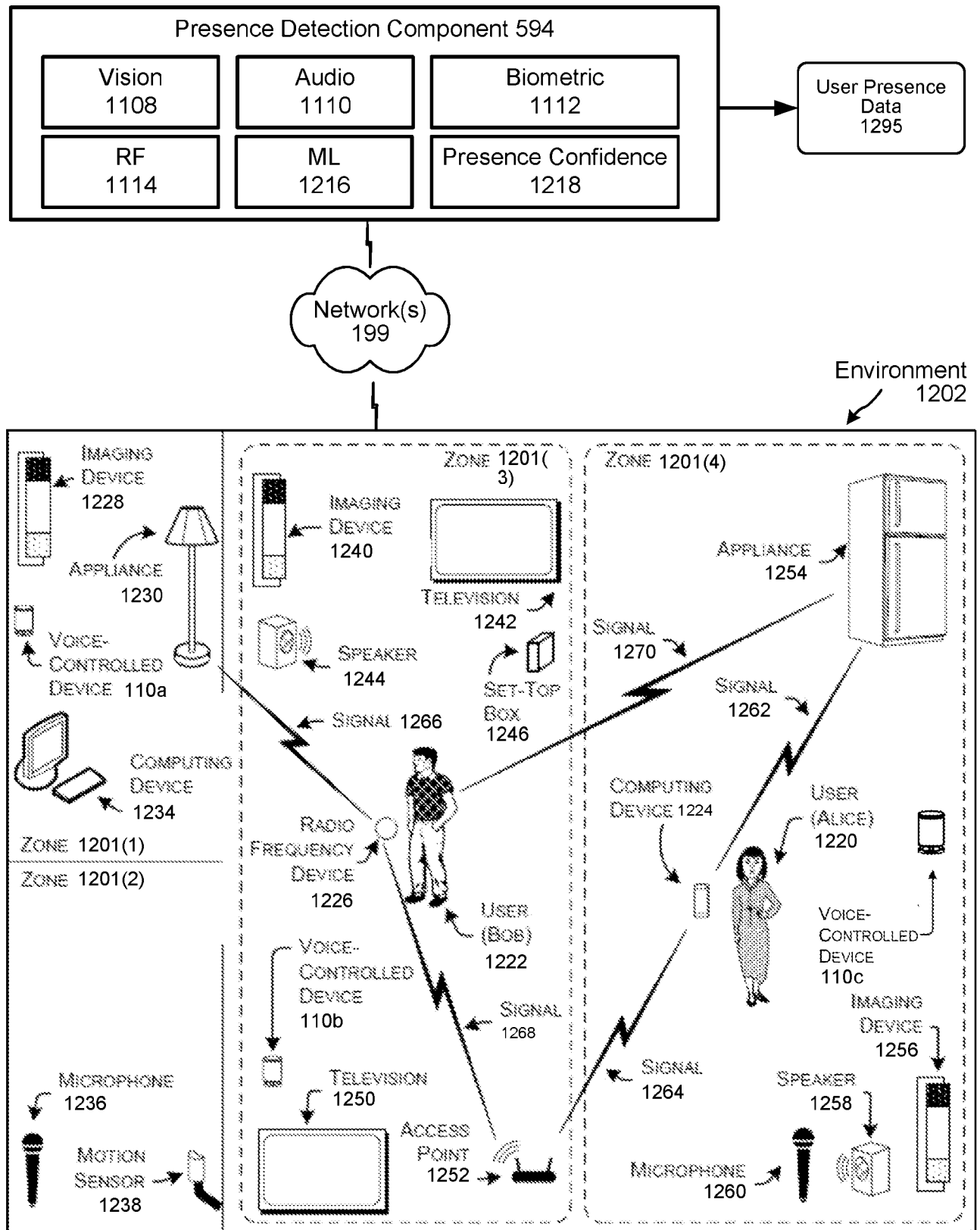


FIG. 13

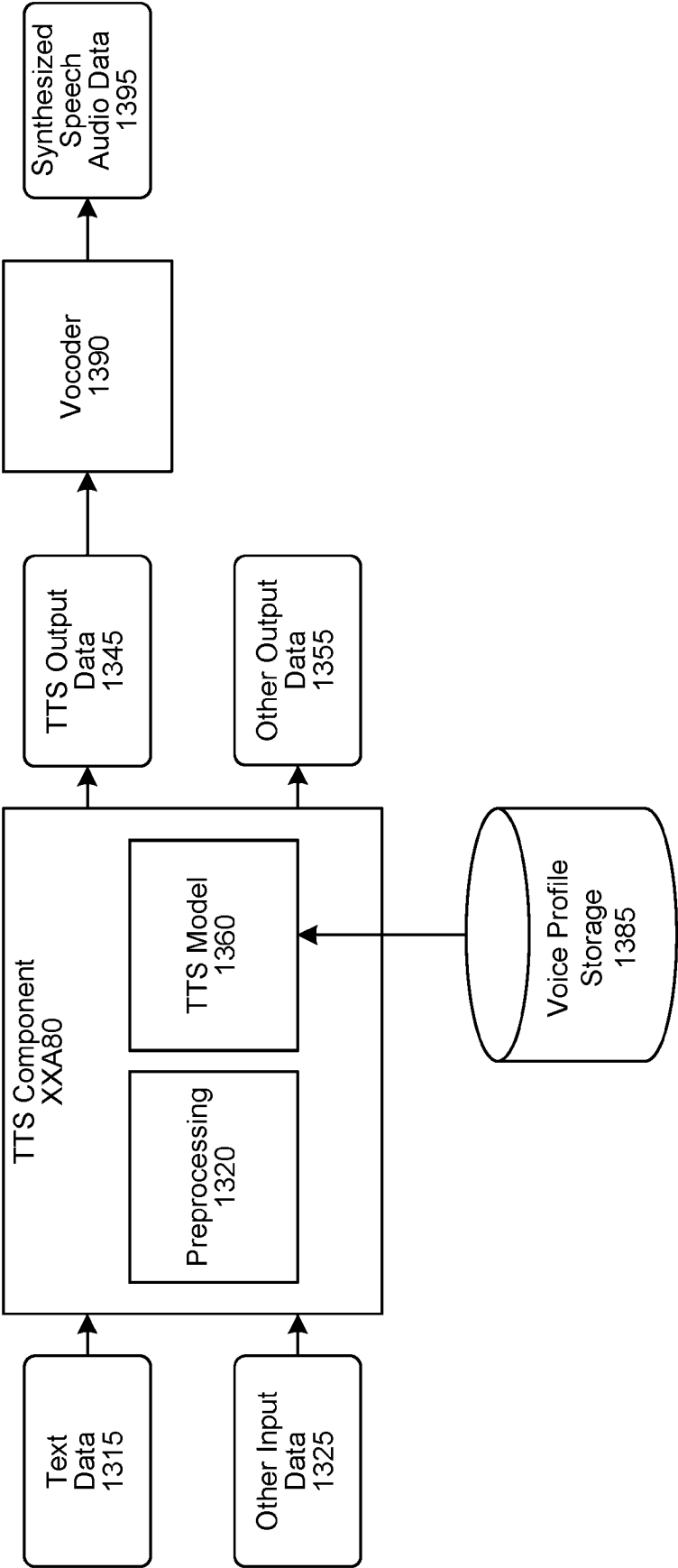


FIG. 14

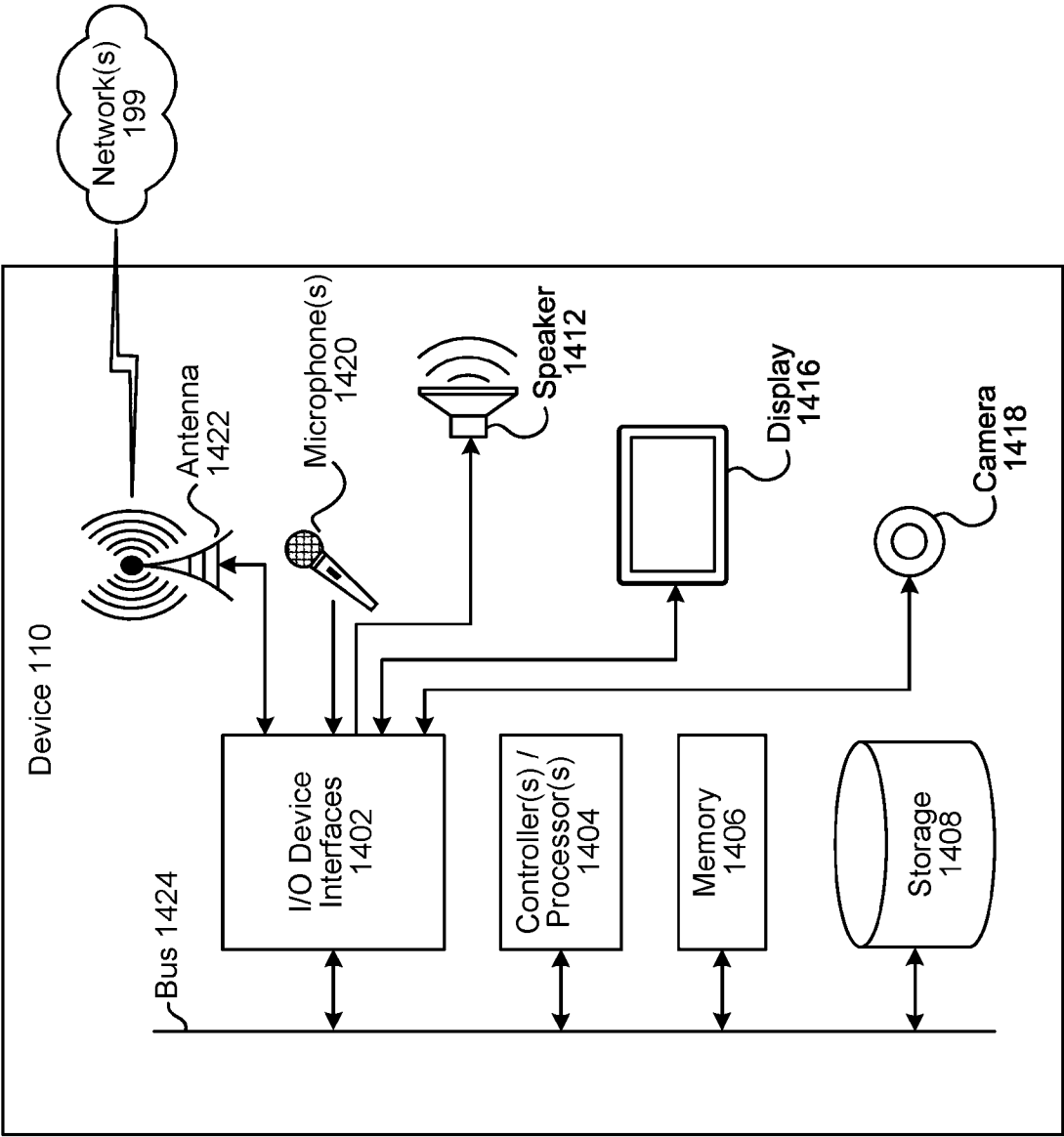


FIG. 15

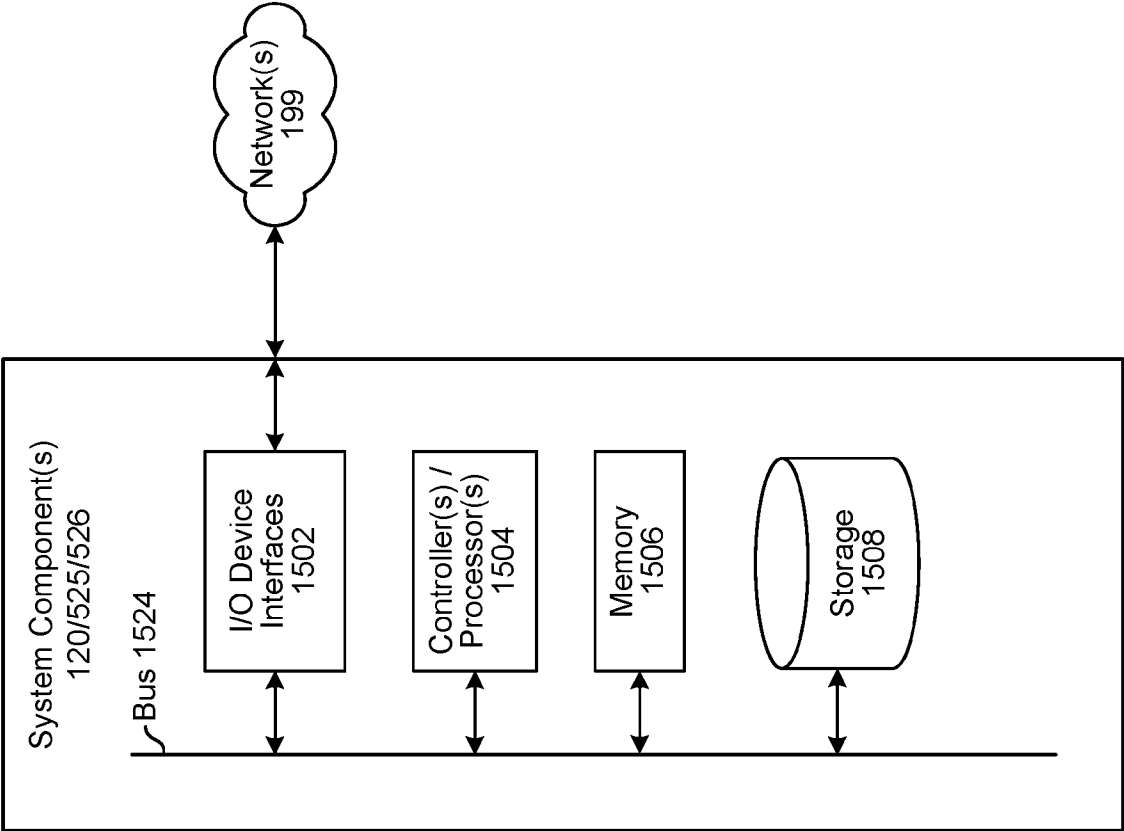
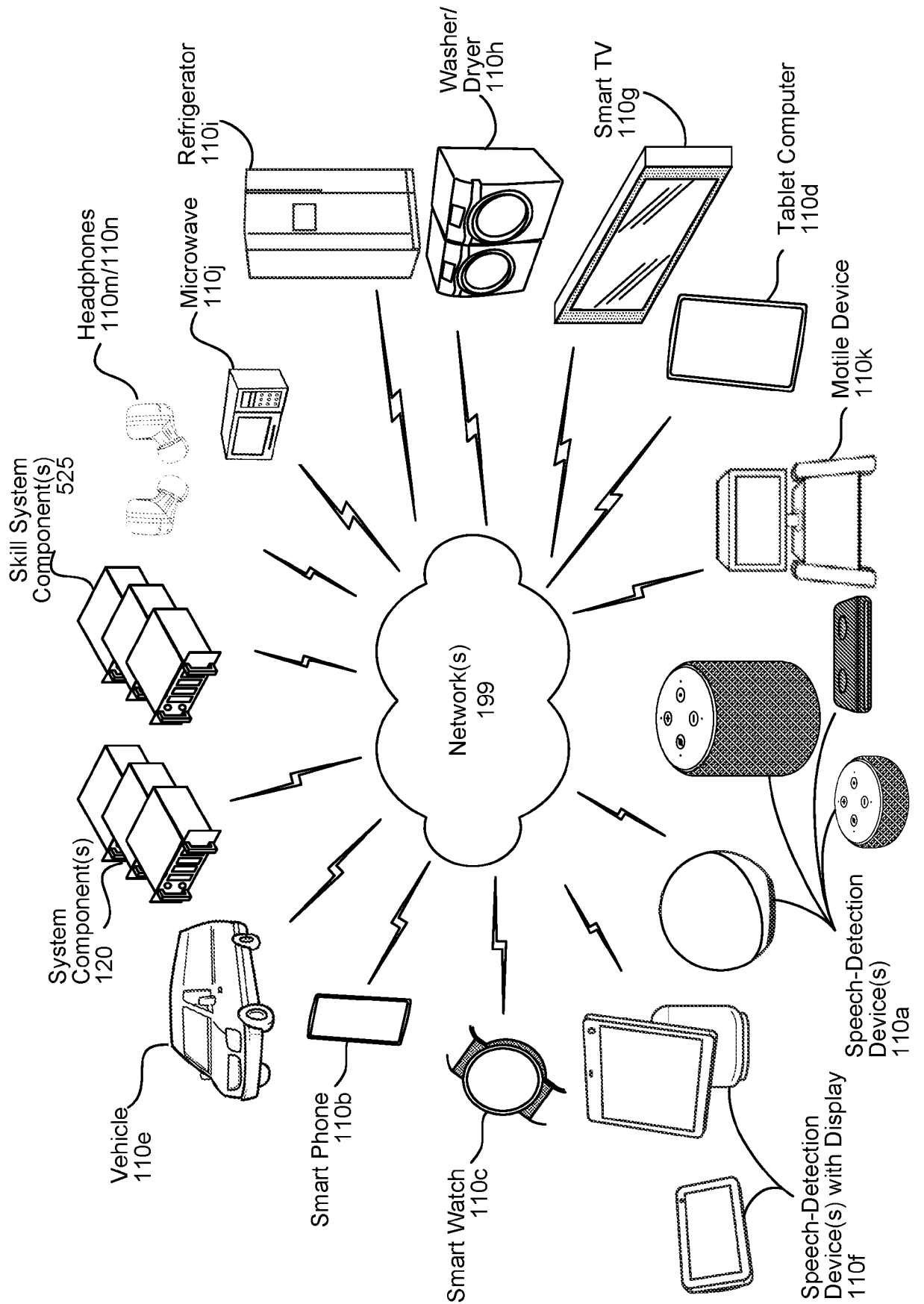


FIG. 16



INTERNATIONAL SEARCH REPORT

International application No
PCT/US2024/028136

A. CLASSIFICATION OF SUBJECT MATTER INV. G10L15/22 ADD.		
According to International Patent Classification (IPC) or to both national classification and IPC		
B. FIELDS SEARCHED		
Minimum documentation searched (classification system followed by classification symbols) G10L		
Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched		
Electronic data base consulted during the international search (name of data base and, where practicable, search terms used) EPO-Internal		
C. DOCUMENTS CONSIDERED TO BE RELEVANT		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2021/090575 A1 (MAHMOOD MUNIR [US] ET AL) 25 March 2021 (2021-03-25) paragraph [0047] - paragraph [0086] figures 2-7 -----	1 - 15
A	US 2015/279360 A1 (MENGIBAR PEDRO J MORENO [US] ET AL) 1 October 2015 (2015-10-01) figure 3 -----	1 - 15
A	US 11 544 504 B1 (FAN XING [US] ET AL) 3 January 2023 (2023-01-03) figure 11 ----- - / - -	1 - 15
☒ Further documents are listed in the continuation of Box C. ☒ See patent family annex.		
* Special categories of cited documents : "A" document defining the general state of the art which is not considered to be of particular relevance "E" earlier application or patent but published on or after the international filing date "L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) "O" document referring to an oral disclosure, use, exhibition or other means "P" document published prior to the international filing date but later than the priority date claimed "T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention "X" document of particular relevance;; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone "Y" document of particular relevance;; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art "&" document member of the same patent family		
Date of the actual completion of the international search 18 July 2024		Date of mailing of the international search report 20/08/2024
Name and mailing address of the ISA/ European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016		Authorized officer De Ceulaer, Bart

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2024/028136

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	MUNKHDALAI TSENDSUREN ET AL: "Fast Contextual Adaptation with Neural Associative Memory for On-Device Personalized Speech Recognition", ICASSP 2022 - 2022 IEEE INTERNATIONAL CONFERENCE ON ACOUSTICS, SPEECH AND SIGNAL PROCESSING (ICASSP), IEEE, 23 May 2022 (2022-05-23), pages 6632-6636, XP034156811, DOI: 10.1109/ICASSP43922.2022.9747726 [retrieved on 2022-04-27] the whole document -----	1 - 15
A	US 2020/357388 A1 (ZHAO DING [US] ET AL) 12 November 2020 (2020-11-12) figure 1 -----	1 - 15

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/US2024/028136

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2021090575	A1	25-03-2021	NONE
US 2015279360	A1	01-10-2015	NONE
US 11544504	B1	03-01-2023	NONE
US 2020357388	A1	12-11-2020	CN 113874935 A 31-12-2021
		EP 3966808 A1 16-03-2022	
		JP 7417634 B2 18-01-2024	
		JP 2022531615 A 07-07-2022	
		KR 20220008309 A 20-01-2022	
		US 2020357388 A1 12-11-2020	
		WO 2020231522 A1 19-11-2020	