(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
01 November 2018 (01.11.2018)

W!P O | PCT

(10) International Publication Number
WO 2018/195875 Al

(51) International Patent Classification:
G06F 17/30 (2006.01)

(21) International Application Number:
PCT/CN2017/082253

(22) International Filing Date:
27 April 2017 (27.04.2017)

(25) Filing Language: English

(26) Publication Langiiage: English

(71) Applicant: MICROSOFT TECHNOLOGY LICENSING, LLC [US/US]; One Microsoft Way, Redmond, Washington 98052 (US).

(72) Inventor; and
(71) Applicant *(for US only)*: WU, Xianchao [CN/JP]; One Microsoft Way, Redmond, Washington 98052 (US).

(74) Agent: NTD PATENT & TRADEMARK AGENCY LIMITED; 10th Floor, Tower C, Beijing Global Trade Center, 36 North Third Ring Road East, Dongcheng District, Beijing 100013 (CN).

(81) Designated States *(unless otherwise indicated, for every kind cf 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, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, 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, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind cf regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, 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, 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).

Declarations under Rule 4.17:
— *cf inventorship (Rule 4.1 7(iv))*

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

(54) **Title:** GENERATING QUESTION-ANSWER PAIRS FOR AUTOMATED CHATTING

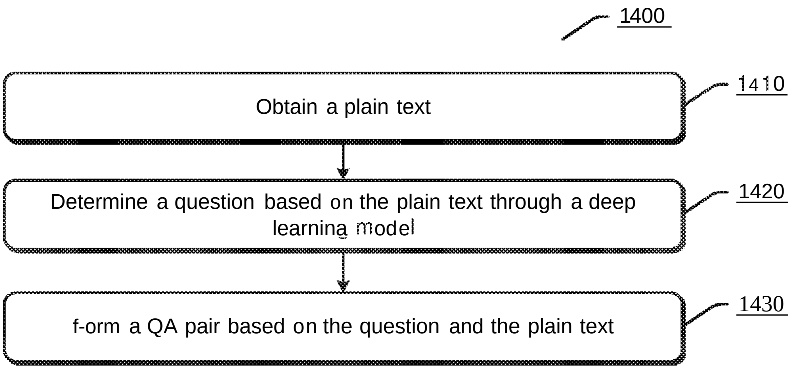

1400
1410
Obtain a plain text
1420
Determine a question based on the plain text through a deep learning model
1430
f-orm a QA pair based on the question and the plain text

FIG 14

(57) **Abstract:** The present disclosure provides method and apparatus for generating question-answer (QA) pairs for automated chatting. A plain text may be obtained. A question may be determined based on the plain text through a deep learning model. A QA pair may be formed based on the question and the plain text.

GENERATING QUESTION-ANSWER PAIRS FOR AUTOMATED CHATTING

BACKGROUND

[0001] Artificial Intelligence (AI) chatbot is becoming more and more popular, and is being applied in an increasing number of scenarios. The chatbot is designed to simulate people's conversation, and may chat with users by text, speech, image, etc. Generally, the chatbot may scan for keywords within a message input by a user or apply natural language processing on the message, and provide a response with the most matching keywords or the most similar wording pattern to the user. The chatbot may be constructed based on a set of question-answer (QA) pairs that can facilitate the chatbot to determine the response to the message input by the user.

SUMMARY

[0002] This Summary is provided to introduce a selection of concepts that are further described below in the Detailed Description. It is not intended to identify key features or essential features of the claimed subject matter, nor is it intended to be used to limit the scope of the claimed subject matter.

[0003] Embodiments of the present disclosure propose method and apparatus for generating question-answer (QA) pairs for automated chatting. A plain text may be obtained. A question may be determined based on the plain text through a deep learning model. A QA pair may be formed based on the question and the plain text.

[0004] It should be noted that the above one or more aspects comprise the features hereinafter fully described and particularly pointed out in the claims. The following description and the drawings set forth in detail certain illustrative features of the one or more aspects. These features are only indicative of the various ways in which the principles of various aspects may be employed, and this disclosure is intended to include all such aspects and their equivalents.

BRIEF DESCRIPTION OF THE DRAWINGS

[0005] The disclosed aspects will hereinafter be described in connection with the appended drawings that are provided to illustrate and not to limit the disclosed aspects.

[0006] FIG. 1 illustrates an exemplary application scenario of a chatbot according to an embodiment.

[0007] FIG. 2 illustrates an exemplary chatbot system according to an embodiment.

[0008] FIG. 3 illustrates an exemplary chat window according to an embodiment.

[0009] FIG.4 illustrates an exemplary process for generating QA pairs according to an embodiment.

[0010] FIG.5 illustrates an exemplary process for generating QA pairs through a Learning-to-Rank (LTR) model according to an embodiment.

[0011] FIG.6 illustrates an exemplary matching between a plain text and a reference QA pair according to an embodiment.

[0012J FIG.7 illustrates an exemplary process for training a recurrent neutral network which is for determining similarity scores according to an embodiment.

[0013] FIG.8 illustrates an exemplary GRIT process according to an embodiment.

[0014] FIG.9 illustrates an exemplary process for applying a recurrent neutral network for determining similarity scores according to an embodiment.

[0015] FIG. 10 illustrates an exemplary process for generating QA pairs through a Neutral Machine Translation (NMT) model according to an embodiment.

[0016] FIG. 11 illustrates an exemplary structure of an NMT model according to an embodiment.

[0017] FIG. 12 illustrates an exemplary process for generating a question through a Dynamic Memory Network (DMN) model according to an embodiment.

[0018] FIG. 13 illustrates exemplary user interfaces according to an embodiment.

[0019] FIG. 14 illustrates a flowchart of an exemplary method for generating QA pairs for automated chatting according to an embodiment.

[0020] FIG. 15 illustrates an exemplary apparatus for generating QA pairs for automated chatting according to an embodiment.

[0021] FIG. 16 illustrates an exemplary apparatus for generating QA pairs for automated chatting according to an embodiment.

DETAILED DESCRIPTION

[0022] The present disclosure will now be discussed with reference to several example implementations. It is to be understood that these implementations are discussed only for enabling those skilled in the art to better understand and thus implement the embodiments of the present disclosure, rather than suggesting any limitations on the scope of the present disclosure.

[0023] AI chat system, e.g., AI chatbot, is tending to be one of the most impressive directions in the AI field in recent years. Conversation, through voice, text, etc., is discovered as a unified entrance to a number of products or applications. For example, E-commerce online shopping may customize general chatbots to fit individual shops that are selling clothes, shoes, cameras, cosmetics, etc., and supply online and in-time conversation-style consumer services. Through this multiple-round conversation, consumers' questions can be answered and the consumers' orders may be expected to be received consequently. In addition, the consumers' detailed requests can be clarified step-by-step during the conversation. This type of consumer service is more user-friendly compared with traditional search engines which are designed for a single-round question-answering service. On the other hand, search engines can be further taken as a background "toolkit" to help making the chatbot's responses to be more accurate and more diverse.

[0024] Conventional methods for constructing chatbot may obtain a set of QA pairs from QA style websites, e.g., Yahoo Answers, Lineq, Zhihu, etc., and use the set of QA pairs to construct a chatbot. However, since these conventional methods lack effective technical means for obtaining QA pairs from a large-scale of plain texts automatically, they are limited to use QA pairs from the QA style websites to construct the chatbot. In other words, these conventional methods cannot construct a chatbot based on plain texts automatically and effectively. Accordingly, it is difficult for these conventional methods to construct chatbots for a lot of domains or companies, since these domains or companies only have a number of plain texts but have no QA pairs. Herein, plain texts may refer to non-QA-style texts, such as, product descriptions, user comments, etc. A plain text may contain one single sentence or a plurality of sentences.

[0025] Embodiments of the present disclosure propose to generate QA pairs from plain texts automatically. Accordingly, chatbots may also be constructed based on the plain texts. Deep learning techniques in conjunction with natural language processing techniques may be adopted in the embodiments. For example, the embodiments may determine a question based on a plain text through the deep learning techniques, and further form a QA pair based on the question and the plain text. In this way, a set of QA pairs may be generated from a plurality of plain texts. The deep learning techniques may comprise Learning-to-Rank (LTR) algorithm. Neutral Machine Translation (NMT) technique, Dynamic Memory Network (DMN) technique, etc.

[0026] According to the embodiments of the present disclosure, a chatbot may be constructed for a specific domain or for a specific company, as long as plain texts of this domain or company are given. The deep learning techniques may help extracting rich information included in plain texts. Consequently, questions can be built for the "rich information". Through constructing chatbots based on a large-scale of plain texts, knowledge from vaiious domains can be used for enriching responses provided by the chatbots.

[0027] FIG.l illustrates an exemplary application scenario 100 of a chatbot according to an embodiment.

[0028] In FIG.l, a network 110 is applied for interconnecting among a terminal device 120 and a chatbot server 130.

[0029] The network 110 may be any type of networks capable of interconnecting network entities. The network 110 may be a single network or a combination of various networks. In terms of coverage range, the network 110 may be a Local Area Network (LAN), a Wide Area Network (WAN), etc. In terms of carrying medium, the network 110 may be a wireline network, a wireless network, etc. In terms of data switching techniques, the network 110 may be a circuit switching network, a packet switching network, etc.

[0030] The terminal device 120 may be any type of electronic computing devices capable of connecting to the network 110, assessing servers or websites on the network 110, processing data or signals, etc. For example, the terminal device 120 may be a desktop computer, a laptop, a tablet, a smart phone, etc. Although only one terminal device 120 is shown in FIG.l, it should be appreciated that a different number of terminal devices may connect to the network 110.

[0031] The terminal device 120 may include a chatbot client 122 which may-provide automated chatting service for a user. In some implementations, the chatbot client 122 may interact with the chatbot server 130. For example, the chatbot client 122 may transmit messages input by the user to the chatbot server 130, and receive responses associated with the messages from the chatbot server 130. However, it should be appreciated that, in other implementations, instead of interacting with the chatbot server 130, the chatbot client 122 may also locally generate responses to messages input by the user.

[0032] The chatbot server 130 may connect to or incorporate a chatbot database 140. The chatbot database 140 may comprise information that can be used by the chatbot server 130 for generating responses.

[0033] It should be appreciated that all the network entities shown in FIG.l are exemplary, and depending on specific application requirements, any other network entities may be involved in the application scenario 100.

[0034] FIG.2 illustrates an exemplary chatbot system 200 according to an embodiment.

[0035] The chatbot system 200 may comprise a user interface (UI) 210 for presenting a chat window. The chat window may be used by the chatbot for interacting with a user.

[0036] The chatbot system 200 may comprise a core processing module 220. The core processing module 220 is configured for, during operation of the chatbot, providing processing capabilities through cooperation with other modules of the chatbot system 200.

[0037] The core processing module 220 may obtain messages input by the user in the chat window, and store the messages in the message queue 232. The messages may be in various multimedia forms, such as, text, speech, image, video, etc.

[0038] The core processing module 220 may process the messages in the message queue 232 in a first-in-first-out manner. The core processing module 220 may invoke processing units in an application program interface (API) module 240 for processing various forms of messages. The API module 240 may comprise a text processing unit 242, a speech processing unit 244, an image processing unit 246, etc.

[0039] For a text message, the text processing unit 242 may perform text understanding on the text message, and the core processing module 220 may further determine a text response.

[0040] For a speech message, the speech processing unit 244 may perform a speech-to-text conversion on the speech message to obtain text sentences, the text processing unit 242 may perform text understanding on the obtained text sentences, and the core processing module 220 may further determine a text response. If it is determined to provide a response in speech, the speech processing unit 244 may perform a text-to-speech conversion on the text response to generate a corresponding speech response.

[0041] For an image message, the image processing unit 246 may perform image recognition on the image message to generate corresponding texts, and the core processing module 220 may further determine a text response. In some cases, the image processing unit 246 may also be used for obtaining an image response based on the text response.

[0042] Moreover, although not shown in FIG.2, the API module 240 may also comprise any other processing units. For example, the API module 240 may comprise a video processing unit for cooperating with the core processing module 220 to process a video message and determine a response.

[0043] The core processing module 220 may determine responses through an index database 250. The index database 250 may comprise a plurality of index items that can be retrieved by the core processing module 220 as responses. The index items in the index database 250 may be classified into a pure chat index set 252 and a QA pair index set 254. The pure chat index set 252 may comprise index items that are prepared for free chatting between users and the chatbot, and may be established with data from social networks. The index items in the pure chat index set 252 may or may not be in a form of question-answer pair. A question-answer pair may also be referred to as message-response pair. The QA pair index set 254 may comprise QA pairs generated based on plain texts through methods according to the embodiments of the present disclosure.

[0044] The chatbot system 200 may comprise a QA pair generating module 260. The QA pair generating module 260 may be used for generating QA pairs based on plain texts according to the embodiments of the present disclosure. The generated QA pairs may be indexed in the QA pair index set 254

[0045] The responses determined by the core processing module 220 may be provided to a response queue or response cache 234. For example, the response cache 234 may ensure that a sequence of responses can be displayed in a pre-defined time stream. Assuming that, for a message, there are no less than two responses determined by the core processing module 220, then a time-delay setting for the responses may be necessary. For example, if a message input by the player is "Did you eat your breakfast?", two responses may be determined, such as, a first response "Yes, I ate bread" and a second response "How about you? Still feeling hungry?". In this case, through the response cache 234, the chatbot may ensure that the first response is provided to the player immediately. Further, the chatbot may ensure that the second response is provided in a time delay, such as 1 or 2 seconds, so that the second response will be provided to the player 1 or 2 seconds after the first response. As such, the response cache 234 may manage the to-be-sent responses and appropriate timing for each response.

[0046] The responses in the response queue or response cache 234 may be further transferred to the user interface 210 such that the responses can be displayed to the user in the chat window.

[0047] It should be appreciated that all the elements shown in the chatbot system 200 in FIG.2 are exemplary, and depending on specific application requirements, any shown elements may be omitted and any other elements may be involved in the chatbot system 200.

[0048] FIG.3 illustrates an exemplary chat window 300 according to an embodiment. The chat window 300 may comprise a presentation area 310, a control area 320 and an input area 330. The presentation area 310 displays messages and responses in a chat flow. The control area 320 includes a plurality of virtual buttons for the user to perform message input settings. For example, the user may select to make a voice input, attach image files, select emoji symbols, make a short-cut of the current screen, etc. through the control area 320. The input area 330 is used for the user to input messages. For example, the user may type text through the input area 330. The chat window 300 may further comprise a virtual button 340 for confirming to send input messages. If the user touches the virtual button 340, the messages input in the input area 330 may be sent to the presentation area 310.

[0049] It should be noted that all the elements and their layout shown in FIG.3 are exemplary. Depending on specific application requirements, the chat window in FIG.3 may omit or add any elements, and the layout of the elements in the chat window in FIG.3 may also be changed in various manners.

[0050] FIG.4 illustrates an exemplary process 400 for generating QA pairs according to an embodiment. The process 400 may be performed by, such as, the QA pair generating model 260 shown in FIG.2.

[0051] A plurality of plain texts 410 may be obtained. The plain texts 410 may be crawled from a website of a content source, e.g., a company. The plain texts 410 may also be received in plain text documents provided by the content source. In some implementations, the plain texts 410 are relating to a specific domain or a specific company for which a chatbot is desired to be constructed.

[0052] The plain texts 410 may be provided to a deep learning model 420. The deep learning model 420 may determine questions 430 based on the plain texts 410. Various techniques may be adopted in the deep learning model 420. For example, the deep learning model 420 may comprise at least one of a LTR model 422, a NMT model 424 and a DMN model 426. Any one or any combination of the LTR model 422, the NMT model 424 and the DMN model 426 may be used for generating questions 430 based on the plain texts 410.

[0053] The LTR model 422 may find questions for a plain text from a reference QA database. The reference QA database may comprise a plurality of reference <question, answer> QA pairs. A reference QA pair may also be referred to as an existing QA pair, which is obtained from QA websites or through any known approaches. A ranking algorithm in the LTR model 422 may take a plain text and reference QA pairs in the reference QA database as inputs, and compute similarity scores between the plain text and each reference QA pair through at least one of word matching and latent semantic matching. For example, the ranking algorithm may compute a first matching score between the plain text and a reference question in each reference QA pair and a second matching score between the plain text and a reference answer in the reference QA pair, and then obtain a similarity score of the reference QA pair based on the first matching score and the second matching score. In this way, the ranking algorithm may obtain a set of similarity scores of reference QA pairs in the reference QA database compared to the plain text, and then rank the reference QA pairs based on the similarity scores. A reference question in a top-ranked reference QA pair may be selected as a question for the plain text.

[0054] The NMT model 424 may generate a question based on a plain text in a sequence-to-sequence approach. For example, if the plain text is provided to the NMT model 424 as an input, then the question may be output by the NMT model 424. In other words, the plain text may be translated by the NMT model 424 into the question directly.

[0055] The DMN model 426 may generate a question based on a plain text through capturing latent semantic relations in the plain text. That is, the DMN model

426 may reason out the question for a list of sentences in the plain text automatically. For example, the DMN model 426 may capture latent semantic relations among the list of sentences in the plain text automatically to determine whether to use or ignore a sentence or words in a sentence during generating the question. In an implementation, the DMN model 426 may take a result from the NMT model 424 as a priori input, so as to further improve quality of the question finally generated. It should be appreciated that the NMT model 424 may provide a local optimization, while the DMN model 426 may provide a global optimization since it is strong at multi-turn "reasoning". Moreover, in an implementation, the DMN model 426 may also use one or more candidate questions generated by the LTR model 422 to further improve quality of the question finally generated.

[0056] Upon determining questions for plain texts through the deep learning model 420, a plurality of QA pairs may be formed and added into a <question, plain texi> pair database 440. For example, for a plain text, a QA pair may be formed based on the plain text and a question determined for the plain text, where the plain text is added in an answer part of the QA pair. The <question, plain text> pair database 440 may be further used for establishing the QA pair index set 254 shown in FIG.2.

[0057] FIG.5 illustrates an exemplary process 500 for generating QA pairs through a LTR model according to an embodiment.

[0058] The process 500 may be performed for generating QA pairs for a plain text 510.

[0059] According to the process 500, a plurality of QA pairs may be obtained from QA websites 520. The QA websites 520 may be any QA style websites, e.g., Yahoo Answers, Lineq, Zhihu, etc.

[0060] The QA pairs obtained from the QA websites 520 may be used as reference QA pairs 530. Each reference QA pair may contain a reference question 532 and a reference answer 534.

[0061] At 540, a reference QA pair-plain text matching may be applied on the plain text 510 and the reference QA pairs 530. The reference QA pair-plain text matching at 540 may perform a matching process between the plain text 510 and the reference QA pairs 530 through, such as, word matching and/or latent semantic matching. The word matching may refer to a character, word or phrase level comparison between a plain text and a reference QA pair so as to find shared/matched words. The latent semantic matching may refer to a comparison in a dense vector space between a plain text and a reference QA pair so as to find semanticaily related words. It should be appreciated that, in this disclosure, the use of the terms "word", "character" and "phrase" may be interchanged among each other. For example, if the term "word" is used in an expression, this term may also be interpreted as "character" or "phrase".

[0062] In an implementation, a question-plain text matching model 542 and an answer-plain text matching model 544 may be adopted in the reference QA pair-plain text matching 540. The question-plain text matching model 542 may compute a matching score, *S(question, plain text),* between the plain text 510 and a reference question in a reference QA pair. The answer-plain text matching model 544 may compute a matching score, *S(answer, plain text),* between the plain text 510 and a reference answer in the reference QA pair. The question-plain text matching model 542 and the answer-plain text matching model 544 will be further discussed later.

[0063] At 550, the matching score obtained by the question-plain text matching model 542 and the matching score obtained by the answer-plain text matching model 544 may be combined so as to obtain a similarity score, *S (<question, answef>, plain text),* for the reference QA pair. The similarity score may be computed through:

$$\begin{aligned} &S(<question, answer>, plain\ text) \\ &\quad = \lambda * S(question, plain\ text) + (1 - \lambda) * S(answer, plain\ text) \quad \textbf{Equation (1)} \end{aligned}$$

where λ is a hyper-parameter and λ G **[0, 1]**.

[0064] Through performing the reference QA pair-plain text matching at 540 and the combining at 500 for each of the reference QA pairs 530, similarity scores of these reference QA pairs 530 compared to the plain text 510 may be obtained respectively. Thus, these reference QA pairs 530 may be ranked at 560 based on the similarity scores.

[0065] At 570, a reference question in a top-ranked reference QA pair may be selected as a question for the plain text 510.

[0066] A <question, plain text> pair may be formed based on the selected question and the plain text 510, and added into a <question, plain text> pair database 580. Question-plain text pairs in the <question, plain text> pair database 580 may be construed as QA pairs generated through the LTR model according to the embodiments of the present disclosure.

[0067] It should be appreciated that, in some implementations, more than one question-plain text may be generated for the plain text 510. For example, at 570, two or more reference questions in two or more top-ranked reference QA pairs may be selected as questions for the plain text 510, and thus two or more question-plain text pairs may be formed based on the selected questions and the plain text 510.

[0068] FIG.6 illustrates an exemplary matching 600 between a plain text and a reference QA pair according to an embodiment. The matching 600 may be implemented by the reference QA pair-plain text matching 540 shown in FIG.5.

[0069] An exemplary plain text 610 may be: *For meaningful words, that should be considered as "Manma". This happened with my child.* An exemplary reference QA pair 620 may comprise a reference question and a reference answer. The reference question may be: *What are the most frequently speaking words when new horn babies begin to talk?* The reference answer may be: *Is Mama, Manma, Papa or alike? When the baby begin to recognize something, should be manma or alike.*

[0070] Block 630 shows an exemplary matching between the plain text 610 and the reference question in the reference QA pair 620. For example, the term "words" in the plain text 610 is found matching the term "words" in the reference question, and the term "child" in the plain text 610 is found latent-semantically matching the phrase "new born babies" in the reference question.

[0071] Block 640 shows an exemplary matching between the plain text 610 and the reference answer in the reference QA pair 620. For example, the term "Manma" in the plain text 610 is found matching the term "Manma" in the reference answer, the term "considered" in the plain text 610 is found latent-semantically matching the term "recognize" in the reference answer, and the term "child" in the plain text 610 is found latent-semanticaily matching the term "baby" in the reference answer.

[0072] Next, the question-plain text matching model 542 shown in FIG.5 will be discussed in details.

[0073] A Gradient Boosting Decision Tree (GBDT) may be adopted for the question-plain text matching model 542. The GBDT may take a plain text and reference questions in a plurality of reference QA pairs as inputs, and output similarity scores of the reference questions compared to the plain text.

[0074] In an implementation, a feature in the GBDT may be based on a language model for information retrieval. This feature may evaluate relevance between a plain text ***q*** and a reference question ***Q*** through:

$$P(q|Q) = \mathbf{I}_{w\in q}[(1-\lambda)P_{ml}(w|Q) + \lambda P_{ml}(w\backslash C)] \qquad \text{Equation (2)}$$

where $\boldsymbol{P_m i(w\backslash Q\sim)}$ is the maximum likelihood of word w estimated from ***Q,*** and $P_m i(w\backslash C)$ is a smoothing item that is computed as the maximum likelihood estimation in a large-scale corpus ***C*****.** The smoothing item avoids zero probability, which stems from those words appearing in the plain text ***q*** but not in the reference question ***Q*****.** λ is a parameter that acts as a trade-off between the likeliliood and the smoothing item, where λ **G** [0, 1] . This feature works well when there are a number of words overlapped between the plain text and the reference question.

[0075] In an implementation, a feature in the GBDT may be based on a translation-based language model. This feature may learn word-to-word and/or phrase-to-phrase translation probability from, such as, reference questions or reference QA pairs, and may incorporate the learned information into the maximum likelihood. Given a plain text ***q*** and a reference question ***Q,*** the translation-based language model may be defined as:

$$Ptr_b(q|Q) = \prod_{w\in q}[(1-\lambda)P_{mx}(w|Q) + \lambda P_m i(w|C)] \qquad \textbf{Equation} \text{ (3)}$$

where

$$P_{mx}(w\backslash Q) = \alpha P_{ml}(w\backslash Q) + \beta P_{ir}(w\, iQ) \qquad \text{Equation (4)}$$

$$Ptr(w|Q) = \sum_{v\in Q} P_{tp}(w|v)P_{ml}(v\backslash \mathbf{Q}) \qquad \text{Equation (5)}$$

[0076] Here λ, a and β are parameters satisfying λ **G** [0, 1] and **a** + β = 1 . $P_{tp}(w|v)$ is a translation probability from word v in ***Q*** to word w in q. $P_{tr}(.)$, ¾.) and $P_{tr}b(.)$ are similarity functions constructed step-by-step by using $\boldsymbol{P_{lp}(.)}$ and $\boldsymbol{P_{ml}(.)}$.

[0077] In an implementation, a feature in the GBDT may be an edit distance between a plain text and a reference question in a word or character level.

[0078] In an implementation, a feature in the GBDT may be a maximum subsequence ratio between a plain text and a reference question.

[0079] In an implementation, a feature in the GBDT may be a cosine similarity score from a recurrent neural network containing Gated Recurrent Units (GRUs). The cosine similarity score may be an evaluation for similarity between a plain text and a reference question. The recurrent neural network will be discussed in connection with FIG.7 to FIG.9 below.

[0080] FIG.7 illustrates an exemplary process 700 for training a recurrent neutral network which is for determining similarity scores according to an embodiment.

[0081] Training data may be input in an embedding layer. The training data may comprise an answer, a good question and a bad question. The good question may be semantically related to the answer, while the bad question may be not semantically related to the answer. Assuming that an answer is *"For meaningful words, that should be considered as 'Ahanmi'. This happened with my child"*, then a good question may be " *What are the most frequently speaking words when new horn babies begin to talk?"*, and a bad question may be *"What is the difference between the languages of children and adults?"*. The embedding layer may map the input training data into respective dense vector representations.

[0082] A hidden layer may use GRIT to process the vectors from the embedding layer, e.g., vector of the answer, vector of the good question and vector of the bad question. It should be appreciated that there may be one or more hidden layers in the recurrent neural network. Here, the hidden layer may also be referred to as a recurrent hidden layer.

[0083] An output layer may compute a margin between similarity of <answer, good question> and similarity of <answer, bad question>, and maximize the margin. If the similarity of <answer, good question> is below the similarity of <answer, bad question>, a distance between these two types of similarity may be taken as an error and back propagated to the hidden layer and the embedding layer. In an implementation, the process in the output layer may be expressed as:

$$max\{Q, cos(answer, good\ question) - \boldsymbol{cos(answer,}\ bad\ question)\} \qquad \textbf{Equation}\ (6)$$

where *cosianswer, good question)* denotes a cosine similarity score between the answer and the good question, and *cos(answer, bad question)* denotes a cosine similarity score between the answer and the bad question.

[0084] FIG.8 illustrates an exemplary GRU process 800 according to an embodiment. The GRU process 800 may be implemented in the hidden layer shown in FIG.7.

[0085] An input vector for the GRU process may be obtained from an embedding layer or a previous hidden layer. The input vector may also be referred to as input sequence, word sequence, etc.

[0086] The GRU process is a type of bidirectional encoding process applied on the input vector. There are two directions in the GRU process, e.g., a left-to-right forward direction and a right-to-left backward direction. The GRU process may involves a plurality of GRU units which take an input vector x and a previous step vector $h_{t\cdot\backslash}$ as inputs and output a next step vector h_t.

[0087] Internal mechanism of the GRU process may be defined by the following equations:

$$z_t = \sigma_g (W^{(z)} \mathbf{x}_t + U^{(z)} h_{t-1} + b^{(\mathbf{z})})$$ **E.quation (7)**

$$r_t = \sigma_{\mathbf{g}} (W^{(r)} \mathbf{x}_t + U^{(r)} /h_{t-,} + b^{(r)})$$ **Equation** (8)

$$\tilde{h}_t = \sigma_h \{W^{(h)} x_t + U^{(\mathbf{ft})} (\mathbf{r}_t \circ h_{t}\mathbf{...i}) + b^{(h)})$$ Equation (9)

$$h_t = z_t \circ h_{t-1} + (1 - z_t) \circ \tilde{h}_t$$ **Equation (10)**

where x_t is an input vector, h_t is an output vector, ¾ is an update gate vector, r_t is a reset gate vector, $\sigma_g$is from a sigmoid function, σ_h is from a hyperbolic function, ° is an element-wise product, and $h_0 = 0$. Moreover, $W^{(z)}$, ¥ $^{r)}$, $W^{(h)}$, $U^{(z>}$, $U^{(r)}$, $U^{(h;}$ are parameter matrices, and $h^{(z)}$, $h^{(r>}$, $h^{(n'}$ are parameter vectors. Here, $W^{(z)}, W^{(r)}, W^{(h)}$ G $R^{n_H \times n_I}$; and $\upsilon^{(z)}, \upsilon^{(r)}, \upsilon^{(h)} \in R^{n«\times n«}$, n_H denoting a dimension of a hidden layer, and *iii* denoting a dimension of the input vector. For example, in Equation (7), $W^{(z)}$ is a matrix that projects the input vector $x,$ into a vector space, $iJ^{(\bar{\imath})}$ is a matrix that projects the recurrent hidden layer $h_{t\cdot\backslash}$ into a vector space, and $b^{(z)}$ is a bias vector that determines a relative position of the target vector z_t. Similarly, in Equations (8) and (9), $W^{(r)}$, $U^{(r)}$, $b^{(r)}$ and $W^{(h)}$, $U^{(h)}$, $b^{(h)}$ function in the same way as $W^{(z)}$, $U^{z)}$ and $b^{z)}$.

[0088] Block 810 in FIG.8 shows an exemplary detailed structure of a GRU unit, where x is an input vector for the GRU unit, and h is an output vector for the GRU unit. The GRU unit may be expressed as:

$$k_t^j = z_t^j h_{t-1}^j + (1 - z_t^j) \tilde{h}_t^j$$ Equation **(11)**

where j is a word index in the input vector x. Processes in both the left-to-right forward direction and the right-to-left backward direction may follow Equation (11).

[0089] FIG.9 illustrates an exemplary process 900 for applying a recurrent neutral network for determining similarity scores according to an embodiment. The recurrent neutral network may have been trained through the process 700 shown in FIG.7.

[0090] A plain text and a reference question may be input in an embedding layer. The embedding layer may map the input plain text and reference question into respective dense vector representations.

[0091] A hidden layer may use GRU to process the vectors from the embedding layer, i.e., vector of the plain text and vector of the reference question. It should be appreciated that there may be one or more hidden layers in the recurrent neural network.

[0092] An output layer may compute and output a cosine similarity score between the plain text and the reference question, e.g., *cos (plain text, reference question).* The cosine similaiity score may be used as a feature in the GBDT for the question-plain text matching model 542.

[0093] Next, the answer-plain text matching model 544 shown in FIG.5 will be discussed in details.

[0094] A GBDT may be adopted for the answer-plain text matching model 544. The GBDT may compute a similaiity score of a reference answer in a plurality of reference QA pairs compared to a plain text.

[0095] In an implementation, a feature in the GBDT may be based on an edit distance in a word level between a plain text and a reference answer.

[0096] In an implementation, a feature in the GBDT may be based on an edit distance in a character level between a plain text and a reference answer. For example, for Asian languages such as Chinese and Japanese, similarity computation may be on a character basis.

[0097] In an implementation, a feature in the GBDT may be based on an accumulated Word2vec similarity score, such as a cosine similarity score, between a plain text and a reference answer. Generally, Word2vee similarity computation may project words into a dense vector space and then compute a semantic distance between two words through applying cosine function on two vectors corresponding to the two words. The Word2vec similarity computation may alleviate a sparseness problem caused by word matching. In some implementations, before computing a Word2vec similarity score, a high frequency phrase table may be used for pre-processing the plain text and the reference answer, e.g., pre-combining high frequency n-grams words in the plain text and the reference answer. The following Equations (12) and (13) may be adopted in the computing of the Word2vec similarity score.

$$Sim_1 = \Sigma_{w\ in\ p}{}^{iain}{}_{te}\mathbf{xt}(\text{Word2vec}(w,\ \tfrac{3}{4})) \qquad \textbf{Equation (12)}$$

where v_x is a word or phrase in the reference answer and makes Word2vec(¾>, v) the maximum among all words or phrases v in the reference answer.

$$Sim_2 = \Sigma_{v\ in\ ref} arena{:}\ {}_a\mathbf{nswer}(\mathrm{Word2vec}(\boldsymbol{w_x}, v)) \quad \textbf{Equation (13)}$$

where W_x is a word or phrase in the plain text and makes Word2vec(w, v) the maximum among all words or phrases W in the plain text.

[0098] In an implementation, a feature in the GBDT may be based on a BM25 score between a plain text and a reference answer. BM25 score is a frequently used similarity score in information retrieval. BM25 may be a bag-of-words retrieval function, and may be used here for ranking a set of reference answers based on plain text words appearing in each reference answer, regardless of inter-relationship, e.g., relative proximity, between plain text words within a reference answer. BM25 may be not a single function, and may actually comprise a group of scoring functions with respective components and parameters. An exemplary function is given as follows.

[0099] For a plain text Q containing keywords $q_i, \ldots, q_n$, a BM25 score of a reference answer D may be:

$$Score\ (D,Q) = \sum_{i=1}^{n} \mathrm{IDF}(q_i) \cdot \frac{f(q_i,D)\cdot(k_1+1)}{f(q_i,D)+k_1\cdot(1-b+b\cdot\frac{|D|}{avgdl})} \quad \text{Equation (14)}$$

Here,

- $f(q_i,D)$ is a term frequency of word q_i in the reference answer D, where $f(qi_{>}D)=^{n}$ if q_i occurs n ($n \geqslant 1$) times in D, or otherwise /`(¾., D)= 0;
- \D\ is the number of words in the reference answer D;
- *avgdl* is an average length of reference answers in a reference answer set M (D EM);
- k_1 and b are free parameters, such as, k_1 —1.2 and b —0.75;
- ® IDF(^,) is an inverse document frequency (IDF) weight of plain text word q_i. !!)F(q_i, M) = log(N/|d E M and $q_i \in$ ***d*** \) , where N is the total number of reference answers in the reference answer set M, e.g., N=𝕄I. Moreover, \d ∈ ***M*** and q_i G d \ is the number of reference answers where the word q_i appears.

[00100] Through Equation (14), a BM25 score of a reference answer may be computed based on a plain text.

[00101] FIG. 10 illustrates an exemplary process 1000 for generating QA pairs

through a NMT model according to an embodiment.

[00102] According to the process 1000, a plurality of QA pairs may be obtained from QA websites 1002. The QA websites 1002 may be any QA style websites, e.g., Yahoo Answers, Lineq, Zhihu, etc.

[00103] The QA pairs obtained from the QA websites 520 may be used as training QA pairs 1004. Each training QA pair may contain a question and an answer.

[00104] At 1006, the training QA pairs 1004 may be used for training a NMT model 1008. The NMT model 1008 may be configured for generating a question based on an input answer in a sequence-to-sequence approach. In other words, the input answer may be translated by the NMT model 1008 into the output question directly. Thus, each of the training QA pairs 1004 may be used as a pair of training data for training the NMT model 1008. An exemplary structure of the NMT model 1008 will be discussed later in connection with FIG.l 1.

[00105] After the NMT model 1008 is trained, the NMT model 1008 may be used for generating questions for plain texts. For example, if a plain text 1010 is input into the NMT model 1008, the NMT model 1008 may output a generated question 1012 corresponding to the plain text 1010.

[00106] A <question, plain text> pair may be formed based on the generated question 1012 and the plain text 1010, and added into a <question, plain text> pair database 1014. Question-plain text pairs in the <question, plain text> pair database 1014 may be construed as QA pairs generated through the NMT model 1008 according to the embodiments of the present disclosure.

[00107] FIG. 11 illustrates an exemplary structure 1100 of an NMT model according to an embodiment. The NMT model may comprise an embedding layer, an internal semantic layer, a hidden recurrent layer, and an output layer.

[00108] At the embedding layer, bidirectional recurrent operations may be applied on an input sequence, such as, a plain text, so as to obtain source vectors. There are two directions involved in the bidirectional recurrent operations, e.g., left-to-right and right-to-left. In an implementation, the bidirectional recurrent operations may be based on a GRU process and follow Equations (7)-(10). The embedding layer may also be referred to as "encoder" layer. The source vectors may be denoted by temporal annotation h_j, where $j=l$, 2, …, T_x, and T_x is the length of the input sequence, e.g., the number of words in the input sequence.

[00109] At **the** internal semantic layer, an attention mechanism may be implemented. **A** context vector c_i may be computed based on a set of temporal annotations $\boldsymbol{h_j}$ and may be taken as a temporal dense representation **of** the current input sequence. The context vector $c_:$ may be computed as a weighted sum of the temporal annotations h_j as follows:

$$c_i = \sum_{j=1}^{T_x} aij\, h_j$$

Equatiori (15)

[00110] The weight «$_{;j}$ for each $h_{,-}$ may also be **referred** to as "attention" weight, and may be computed by a softmax function:

$$a\ ij = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}$$

Equation (16)

where e_{ij} = a(s,;...$_1$, $\boldsymbol{h_j}$) is an alignment model which scores how **well** inputs around a position ***j*** and an output at position $_i$ match **with** each other. The alignment score is between a pervious hidden state $s_{j\text{-}1}$ and the j**-th** temporal annotation h_j of the input sequence. The probability $\alpha_{:j}$ reflects importance of ¾$_j$ with respect to **the** previous hidden state Si_{-1} **in** deciding the next hidden state $s_;$ and simultaneously generating **the** next word y$_i$. The internal semantic layer implements an attention mechanism through applying the weight ¾$_{;j}$.

[00111] At the hidden recurrent layer, hidden states ¾ **for** an output sequence are determined through **a** unidirectional recurrent operation, such as, a left-to-right **GRU** process. The computation of $s_;$ also follows Equations (?)-(1os.

[00112] **At the** output layer, word prediction for **the** next word y_i may be determined as follows:

$$P(y\ _i|y_1, -\ , yi_{-1}, x) = g(y\,i_{-1}, s_i, c_i)$$

Equation **(17)**

where si is from the hidden recurrent layer, c,- is from the internal semantic layer. Here, $g(.)$ function is a nonlinear, potentially multi-layered function that outputs probabilities of **the** next candidate words **in the** output sequence. The output layer may also be referred to as a "decoder" layer.

[00113] Through the above exemplary structure, **the NMT** model may generate a question for a plain text through picking up "information-rich" words and changing the words **into** interrogative words. Through implementing **the** attention mechanism **in** the internal semantic layer, relations between an "information-rich" word and corresponding interrogative words may be captured. In other words, the attention mechanism in the NMT model may be used for determining a pattern of a question, e.g., which word in the plain text may be set a question and what interrogative word may be used in the question. Taking the sentences shown in FIG. 6 as an example, the interrogative word "what" may be determined as relating to the word "Manma" in an answer. Moreover, it should be appreciated that it may be meaningless if only these two words are considered. Thus, the NMT model may apply recurrent operations on the input sequence in the embedding layer and/or on the output sequence in the hidden recurrent layer, such that context information for each word in the input sequence and/or for each word in the output sequence may be obtained and applied during determining the output sequence.

[00114] FIG. 12 illustrates an exemplary process 1200 for generating a question through a DMN model according to an embodiment.

[00115] As shown in FIG. 12, a DMN model 1210 may be used for generating a question for a plain text. A <question, plain text> pair may be formed based on the generated question and the plain text, and added into a <question, plain text> pair database. Question-plain text pairs in the <question, plain text> pair database may be construed as QA pairs generated through the NMT model 1210 according to the embodiments of the present disclosure. As shown in FIG. 12, the DMN model 1210 may cooperate with a LTR model 1220 and a NMT model 1230 to generate a question. However, it should be appreciated that, in other implementations, either or both of the LTR model 1220 and the NMT model 1230 may be omitted from the process 1200.

[00116] The DMN model 1210 may take a plain text and context information of the plain text as inputs, where a question is intended to generate for the plain text, and the context information may refer to one or more plain texts previously input to the DMN model 1210. For example, a plain text S_9 may be input through a current plain text model 1242, and a sequence of sentences S_1 to $S\$$ in the context information may¬ be input through an input module 1244. The DMN model 1210 may also take one or more ranked candidate questions $C_\backslash$ to C_5 as inputs, which are determined by the LTR model 1220 based on the plain text ¾ and a set of reference QA pairs 1222. Moreover, the DMN model 1210 may take a priori question q_1 as an input, which is generated by the NMT model 1230 based on the plain text ¾ . A generated question q_2 for the plain text ¾ may be output by a question generation module 1252. It should be appreciated

that, when training the DMN model 1210, a training question obtained through any existing approaches and/or artificially checking for an input plain text may be set in the question generation module 1252.

[00117] Next, exemplary processes in modules of the DMN model 1210 will be discussed in details.

[00118] At the input module 1244, a sequence of sentences Si to S_8 in the context information may be processed. Each sentence is ended with "</s>" to denote the ending of one sentence. Ail the eight sentences may be concatenated together to form a word sequence having T words, from W_l to WT- A bidirectional GRU encoding may be applied on the word sequence. For the left-to-right direction or the right-to-left direction, at each time step t, the DMN model 1210 may update its hidden state as h_t — GRU($L[w_t]$, h_{t}-i), where L is an embedding matrix, and w_t is a word index of the t-th word in the word sequence. Thus, a resulting representation vector for a sentence is a combination of two vectors and each vector is from one direction. Internal mechanism of the GRU may follow Equations (7) to (10). These equations may also be abbreviated as h_t = $GRU(x_t, h_t$-i).

[00119] In addition to encoding the word sequence, a positional encoding with bidirectional GRU may also be applied so as to represent "facts" of the sentences. The facts may be computed as $f_t = GRU_{l2r}(L[S_t], f_{t-1}) + GRU_{r2l}(L[S_t], f_t$-ι), where $I2r$ denotes left-to-right, $r2l$ denotes right-to-left, S_t is an embedding expression of a current sentence, and $/_{-1}$, f_t are facts of a former sentence and the current sentence respectively. As shown in FIG. 12, facts/ $_I$ to f_8 are obtained for the eight sentences in the context information.

[00120] At the current plain text module 1242, the encoding for the current plain text S_9 is a simplified version of the input module 1244, where there is only one sentence o be processed in the current plain text module 1242. The processing by the current plain text module 1242 is similar with the input module 1244. Assuming that there are TQ words in the current plain text, hidden states at the time step t may be computed as q_t = $[GRUi_{2r}(L[W_t^Q], q_t$_i), $GRU_{r2l}(L[W_t^Q], q_{t-1})]$, where L is an embedding matrix, and W_t^Q is a word index of the t-th word in the current plain text. A fact $/_9$ may be obtained for the current plain text S_9 in the current plain text module 1242.

[00121] The DMN model 1210 may comprise a ranked candidate questions module

1246. At the ranked candidate questions module 1246, the DMN model **1210** may compute hidden state and facts for one or more ranked candidate questions in the same way as the input module 1244. As an example, FIG. 12 shows five candidate questions c_1 to C_5, and five facts cf_1 to cf_5 are obtained for these candidate questions.

[00122] Although not shown, the DMN model 1210 may also compute a fact f_p for the priori question q_1 generated by the NMT model 1230 in the same way as the current plain text module 1242.

[00123] The DMN model 1210 may comprise an attention mechanism module and an episodic memory module. The episodic memory module may include a recurrent network, and the attention mechanism module may be based on a gating function. The attention mechanism module may be separated from or incorporated in the episodic memory module.

[00124] According to a conventional computing process, the episodic memory module and the attention mechanism module may cooperate to update episodic memory in an iteration way. For each pass i, the gating function of the attention mechanism module may take a fact/', a previous memory vector m^{r1}, and a current plain text S as inputs, to compute an attention gate $g_t^i = G[f^i, rn^{l\sim 1}, S]$. To compute the episode e^l for pass i, a GRU over a sequence of inputs, e.g., a list of facts f ', weighted by the gates g^i may be applied. Then the episodic memory vector may be computed as $m^i = GRU(e^i, m^{i-1})$. Initially, m^0 is equal to a vector expression of the current plain text S. The episode vector that is given to a question generation module may be the final state m^x of the GRU. The following Equation (18) is for updating hidden states of the GRU at a time step t, and the following Equation (19) is for computing the episode.

$$h_t^i = g_t^i \mathrm{GRU}(f_t, h_{t-1}^i) + (1 - g_t^i) h_{t-1}^i \qquad \text{Equation (18)}$$

$$e^i = h_{T_C}^i \qquad \textbf{Equation (19)}$$

where T_c is the number of input sentences.

[00125] According to the embodiment of the present disclosure, the processing in an attention mechanism module 1248 and an episodic memory module 1250 in the DMN model 1210 further takes the ranked candidate questions and the priori question into account. As shown in FIG. 12, besides the input module 1244 and the current plain text module 1242, the attention mechanism module 1248 also obtains inputs from the ranked candidate questions module 1246 and the NMT module 1230. Thus, the attention gate may be computed as $\boldsymbol{g} \backslash = \boldsymbol{G} \backslash \boldsymbol{f}^{\,i}, \boldsymbol{m}^{l\sim1}, \boldsymbol{s}_{<_9}, \boldsymbol{q}_1, cf^{\,l}, m^{x+l\sim1}]$, where $\boldsymbol{cf}^{\,l}$ denotes the facts from the ranked candidate responses, and $m^{x+1\text{-}1}$ is a memory vector computed for the ranked candidate questions and the priori question. Accordingly, the recurrent network in the episodic memory module 1250 further comprises a computing process of memories $\text{ï}\Pi^{x\div1}$ to $\boldsymbol{m}^{x+y}$ for the ranked candidate questions and the priori question. For example, e_1^{x+l} to $\boldsymbol{e}_5^{x+l}$ in FIG. 12 correspond to the ranked candidate questions, and e_6^{x+i} in FIG. 12 corresponds to the priori question. Outputs from the episodic memory module 1250 to the question generation module 1252 include at least $\boldsymbol{m}^{x}$ and $\boldsymbol{m}^{x+y}$.

[00126] The question generation module 1252 may be used for generating a question. A GRIT decoder may be adopted in the question generation module 1252, and an initial state of the GRU decoder may be initialized to be the last memory vector $a_0 = \backslash\boldsymbol{m}^{x}, m^{x\div y}]$. At a time step t, the GRU decoder may take the current plain text f_9, a last hidden state a_{t--1}, and a previous output $\boldsymbol{y}_t\text{-}i$ as inputs, and then compute a current output as:

$$y_t = softmax(W\wedge\ ^{a,} a_t)'' \qquad \textbf{Equation}\ (20)$$

where $a_t = GRU([y_{t-1}, f_9], a_{t-1})$, and $W^{(a>}$ is a weight matrix by training.

[00127] The last generated word may be concatenated to the question vector at each time step. The generated output by the question generation module 1252 may be trained with a cross-entropy error classification of a correct sequence attached with a "</s>" tag at the end of the sequence.

[00128] The generated question output from the question generation module 1252 may be used for forming a QA pair together with the current plain text.

[00129] It should be appreciated that all the modules, equations, parameters and processes discussed above in connection with FIG. 12 are exemplary, and the embodiments of the present disclosure are not limited to any details in the discussion.

[00130] FIG. 13 illustrates exemplary user interfaces according to an embodiment. The user interfaces in FIG. 13 may be shown to a client, e.g., a company requiring a chatbot provision service, when the client is assessing, such as, a corresponding URL. These user interfaces may be used by the client for building a new chatbot or updating an existing chatbot.

[00131] As shown in the user interface 1310, block 1312 indicates that this user interface is used for adding websites or plain text files. At block 1314, the client may add, delete or edit URLs of websites. At block 1316, the client may upload a plain text file.

[00132] The user interface 1320 is triggered by an operation of the client in the user interface 1310. Block 1322 shows a list of QA pairs generated from plain texts in the websites or the plain text file input by the client. The client may choose to build a new chatbot at block 1324, or update an existing chatbot at block 1326.

[00133] The user interface 1330 shows a chat window with a newly-built chatbot or a newly-updated chatbot that is obtained through an operation of the client in the user interface 1320. As shown in the user interface 1330, the chatbot may provide responses based on the generated QA pairs shown in block 1322.

[00134] It should be appreciated that the user interfaces in FIG. 13 are exemplary, and the embodiments of the present disclosure are not limited to any forms of user interface.

[00135] FIG. 14 illustrates a flowchart of an exemplary method 1400 for generating QA pairs for automated chatting according to an embodiment.

[00136] At 1410, a plain text may be obtained.

[00137] At 1420, a question may be determined based on the plain text through a deep learning model.

[00138] At 1430, a QA pair may be formed based on the question and the plain text.

[00139] In an implementation, the deep learning model may comprise at least one of a LTR model, a NMT model and a DMN model.

[00140] In an implementation, the deep learning model may comprise a LTR model, and the LTR model may be for computing a similarity score between the plain text and a reference QA pair through at least one of word matching and latent semantic matching. In an implementation, the similarity score may be computed through: computing a first matching score between the plain text and a reference question in the reference QA pair; computing a second matching score between the plain text and a reference answer in the reference QA pair; and combining the first matching score and the second matching score to obtain the similarity score. In an implementation, the first matching score and the second matching score may be computed through GBDT.

[00141] In an implementation, the determining the question at 1420 may comprise: computing similarity scores of a plurality of reference QA pairs compared to the plain text through the LTR model; and selecting a reference question in an reference QA pair having the highest similarity score as the question.

[00142] In an implementation, the deep learning model may comprise a NMT model, and the NMT model may be for generating the question based on the plain text in a sequence-to-sequence approach, the plain text being as an input sequence, the question being as an output sequence. In an implementation, the NMT model may comprise an attention mechanism for determining a pattern of the question. In an implementation, the NMT model may comprise at least one of: a first recurrent process for obtaining context information for each word in the input sequence; and a second recurrent process for obtaining context information for each word in the output sequence.

[00143] In an implementation, the deep learning model may comprise a DMN model, and the DMN model may be for generating the question based on the plain text through capturing latent semantic relations in the plain text.

[00144] In an implementation, the deep learning model may comprise a LTR model, and the DMN model may comprise an attention mechanism, the attention mechanism taking at least one candidate question as an input, the at. least one candidate question being determined by the LTR model based on the plain text.

[00145] In an implementation, the deep learning model may comprise a NMT model, and the DMN model may comprise an attention mechanism, the attention mechanism taking a reference question as an input, the reference question being determined by the NMT model based on the plain text.

[00146] In an implementation, the deep learning model may comprise at least one of a LTR model and a NMT model, and the DMN model may compute memory vectors based at least on: at least one candidate question and/or a reference question, the at least one candidate question being determined by the LTR model based on the plain text, the reference question being determined by the NMT model based on the plain text.

[00147] It should be appreciated that the method 1400 may further comprise any steps/processes for generating QA pairs for automated chatting according to the embodiments of the present disclosure as mentioned above.

[00148] FIG. 15 illustrates an exemplary apparatus 1500 for generating QA pairs for automated chatting according to an embodiment.

[00149] The apparatus 1500 may comprise: a plain text obtaining module 1510, for obtaining a plain text; a question determining module 1520, for determining a question based on the plain text through a deep learning model; and a QA pair forming module 1530, for forming a QA pair based on the question and the plain text.

[00150] In an implementation, the deep learning model may comprise at least one of a LTR model, a NMT model and a DMN model.

[00151] In an implementation, the deep learning model may comprise a LTR model, and the LTR model may be for computing a similarity score between the plain text and a reference QA pair through at least one of word matching and latent semantic matching. In an implementation, the similarity score may be computed through: computing a first matching score between the plain text and a reference question in the reference QA pair; computing a second matching score between the plain text and a reference answer in the reference QA pair; and combining the first matching score and the second matching score to obtain the similarity score.

[00152] In an implementation, the deep learning model may comprise a NMT model, and the NMT model may be for generating the question based on the plain text in a sequence-to-sequence approach, the plain text being as an input sequence, the question being as an output sequence. In an implementation, the NMT model may comprise at least one of: a first recurrent process for obtaining context information for each word in the input sequence; and a second recurrent process for obtaining context information for each word in the output sequence.

[00153] In an implementation, the deep learning model may comprise a DMN model, and the DMN model may be for generating the question based on the plain text through capturing latent semantic relations in the plain text. In an implementation, the deep learning model may comprise at least one of a LTR model and a NMT model, and the DMN model may comprise an attention mechanism, the attention mechanism taking at least one candidate question and/or a reference question as an input, the at least one candidate question being determined by the LTR model based on the plain text, the reference question being determined by the NMT model based on the plain text. In an implementation, the deep learning model may comprise at least one of a LTR model and a NMT model, and the DMN model may compute memory vectors

based at least on: at least one candidate question and/or a reference question, the at least one candidate question being determined by the LTR model based on the plain text, the reference question being determined by the NMT model based on the plain text.

[00154] Moreover, the apparatus 1500 may also comprise any other modules configured for performing any operations of the methods for generating QA pairs for automated chatting according to the embodiments of the present disclosure as mentioned above.

[00155] FIG. 16 illustrates an exemplary apparatus 1600 for generating QA pairs for automated chatting according to an embodiment.

[00156] The apparatus 1600 may comprise at least one processor 1610. The apparatus 1600 may further comprise a memory 1620 that is connected with the processor 1110. The memory 1620 may store computer-executable instructions that, when executed, cause the processor 1610 to perform any operations of the methods for generating QA pairs for automated chatting according to the embodiments of the present disclosure as mentioned above.

[00157] The embodiments of the present disclosure may be embodied in a non-transitory computer-readable medium. The non-transitory computer-readable medium may comprise instructions that, when executed, cause one or more processors to perform any operations of the methods for generating QA pairs for automated chatting according to the embodiments of the present disclosure as mentioned above.

[00158] It should be appreciated that all the operations in the methods described above are merely exemplary, and the present disclosure is not limited to any operations in the methods or sequence orders of these operations, and should cover all other equivalents under the same or similar concepts.

[00159] It should also be appreciated that all the modules in the apparatuses described above may be implemented in various approaches. These modules may be implemented as hardware, software, or a combination thereof. Moreover, any of these modules may be further functionally divided into sub-modules or combined together.

[00160] Processors have been described in connection with various apparatuses and methods. These processors may be implemented using electronic hardware, computer software, or any combination thereof. Whether such processors are implemented as hardware or software will depend upon the particular application and overall design constraints imposed on the system. By way of example, a processor, any portion of a processor, or any combination of processors presented in the present disclosure may be implemented with a microprocessor, microcontroller, digital signal processor (DSP), a field-programmable gate array (FPGA), a programmable logic device (PLD), a state machine, gated logic, discrete hardware circuits, and other suitable processing components configured to perform the various functions described throughout the present disclosure. The functionality of a processor, any portion of a processor, or any combination of processors presented in the present disclosure may be implemented with software being executed by a microprocessor, microcontroller, DSP, or other suitable platform.

[00161] Software shall be construed broadly to mean instructions, instruction sets, code, code segments, program code, programs, subprograms, software modules, applications, software applications, software packages, routines, subroutines, objects, threads of execution, procedures, functions, etc. The software may reside on a computer-readable medium. A computer-readable medium may include, by way of example, memory such as a magnetic storage device (e.g., hard disk, floppy disk, magnetic strip), an optical disk, a smart card, a flash memory device, random access memory (RAM), read only memory (ROM), programmable ROM (PROM), erasable PROM (EPROM), electrically erasable PROM (EEPROM), a register, or a removable disk. Although memory is shown separate from the processors in the various aspects presented throughout the present disclosure, the memory may be internal to the processors (e.g., cache or register).

[00162] The previous description is provided to enable any person skilled in the art to practice the various aspects described herein. Various modifications to these aspects will be readily apparent to those skilled in the art, and the generic principles defined herein may be applied to other aspects. Thus, the claims are not intended to be limited to the aspects shown herein. All structural and functional equivalents to the elements of the various aspects described throughout the present disclosure that are known or later come to be known to those of ordinary skill in the art are expressly incorporated herein by reference and are intended to be encompassed by the claims.

WHAT IS CLAIMED IS:

1. A method for generating question-answer (QA) pairs for automated chatting, comprising:

obtaining a plain text;

determining a question based on the plain text through a deep learning model; and

forming a QA pair based on the question and the plain text.

2. The method of claim 1, wherein

the deep learning model comprises a Learning-to-Rank (LTR) model, and

the LTR model is for computing a similarity score between the plain text and a reference QA pair through at least one of word matching and latent semantic matching.

3. The method of claim 2, wherein the similarity score is computed through:

computing a first matching score between the plain text and a reference question in the reference QA pair;

computing a second matching score between the plain text and a reference answer in the reference QA pair; and

combining the first matching score and the second matching score to obtain the similarity score.

4. The method of claim 3, wherein the first matching score and the second matching score are computed through Gradient Boosting Decision Tree (GBDT).

5. The method of claim 1, wherein the deep learning model comprises a Learning-to-Rank (LTR) model, and the determining the question comprises:

computing similarity scores of a plurality of reference QA pairs compared to the plain text through the LTR model; and

selecting a reference question in an reference QA pair having the highest similarity score as the question.

6. The method of claim 1, wherein

the deep learning model comprises a Neutral Machine Translation (NMT) model, and

the NMT model is for generating the question based on the plain text in a sequence-to-sequence approach, the plain text being as an input sequence, the question being as an output sequence.

7. The method of claim 6, wherein the NMT model comprises an attention mechanism for determining a pattern of the question.

8. The method of claim 6, wherein the NMT model comprises at least one of:

a first recurrent process for obtaining context information for each word in the input sequence; and

a second recurrent process for obtaining context information for each word in the output sequence.

9. The method of claim 1, wherein

the deep learning model comprises a Dynamic Memory Network (DMN) model, and

the DMN model is for generating the question based on the plain text through capturing latent semantic relations in the plain text.

10. The method of claim 9, wherein

the deep learning model comprises a Learning-to-Rank (LTR) model, and

the DMN model comprises an attention mechanism, the attention mechanism taking at least one candidate question as an input, the at least one candidate question being determined by the LTR model based on the plain text.

11. The method of claim 9, wherein

the deep learning model comprises a Neutral Machine Translation (NMT) model, and the DMN model comprises an attention mechanism, the attention mechanism taking a reference question as an input, the reference question being determined by the NMT model based on the plain text.

12. The method of claim 9, wherein

the deep learning model comprises a Learning-to-Rank (LTR) model and a Neutral Machine Translation (NMT) model, and

the DMN model computes memory vectors based at least on: at least one candidate quesiion and/or a reference question, the at least one candidate question being determined by the LTR model based on the plain text, the reference question being determined by the NMT model based on the plain text.

13. An apparatus for generating question-answer (QA) pairs for automated chatting, comprising:

a plain text obtaining module, for obtaining a plain text;

a question determining module, for determining a question based on the plain text through a deep learning model; and

a QA pair forming module, for forming a QA pair based on the question and the plain text.

14. The apparatus of claim 13, wherein

the deep learning model comprises a Learning-to-Rank (LTR) model, and

the LTR model is for computing a similarity score between the plain text and a reference QA pair through at least one of word matching and latent semantic matching.

15. The apparatus of claim 14, wherein the similarity score is computed through:

computing a first matching score between the plain text and a reference question in the reference QA pair;

computing a second matching score between the plain text and a reference answer in the reference QA pair; and

combining the first matching score and the second matching score to obtain the similarity score.

16. The apparatus of claim 13, wherein

the deep learning model comprises a Neutral Machine Translation (NMT) model, and

the NMT model is for generating the question based on the plain text in a sequence-to-sequence approach, the plain text being as an input sequence, the question being as an output sequence.

17. The apparatus of claim 16, wherein the NMT model comprises at least one of:

a first recurrent process for obtaining context information for each word in the input sequence; and

a second recurrent process for obtaining context information for each word in the output sequence.

18. The apparatus of claim 13, wherein

the deep learning model comprises a Dynamic Memory Network (DMN) model, and

the DMN model is for generating the question based on the plain text through capturing latent semantic relations in the plain text.

19. The apparatus of claim 18, wherein

the deep learning model comprises at least one of a Learning-to-Rank (LTR) model and a Neutral Machine Translation (NMT) model, and

the DMN model comprises an attention mechanism, the attention mechanism taking at least one candidate question and/or a reference question as an input, the at least one candidate question being determined by the LTR model based on the plain text, the reference question being determined by the NMT model based on the plain text.

20. The apparatus of claim 18, wherein

the deep learning model comprises at least one of a Learning-to-Rank (LTR) model and a Neutral Machine Translation (NMT) model, and

the DMN model computes memory vectors based **at** least **on:** at least one candidate question and/or a reference question, the at least one candidate question being determined by **the** LTR model based **on** the plain **text,** the reference question being determined by the NMT model based on the plain text.

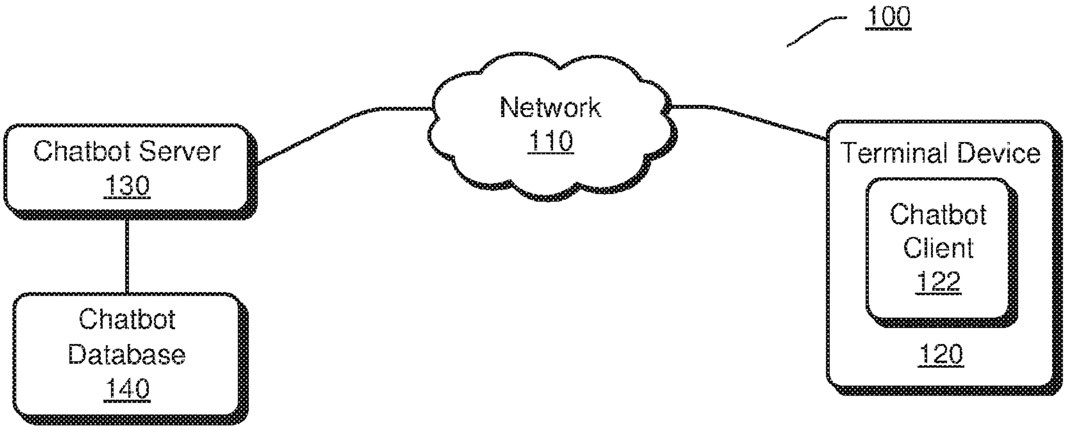

100
Network
110
Chatbot Server
130
Chatbot
Database
140
Terminal Device
Chatbot
Client
122
120

FIG 1

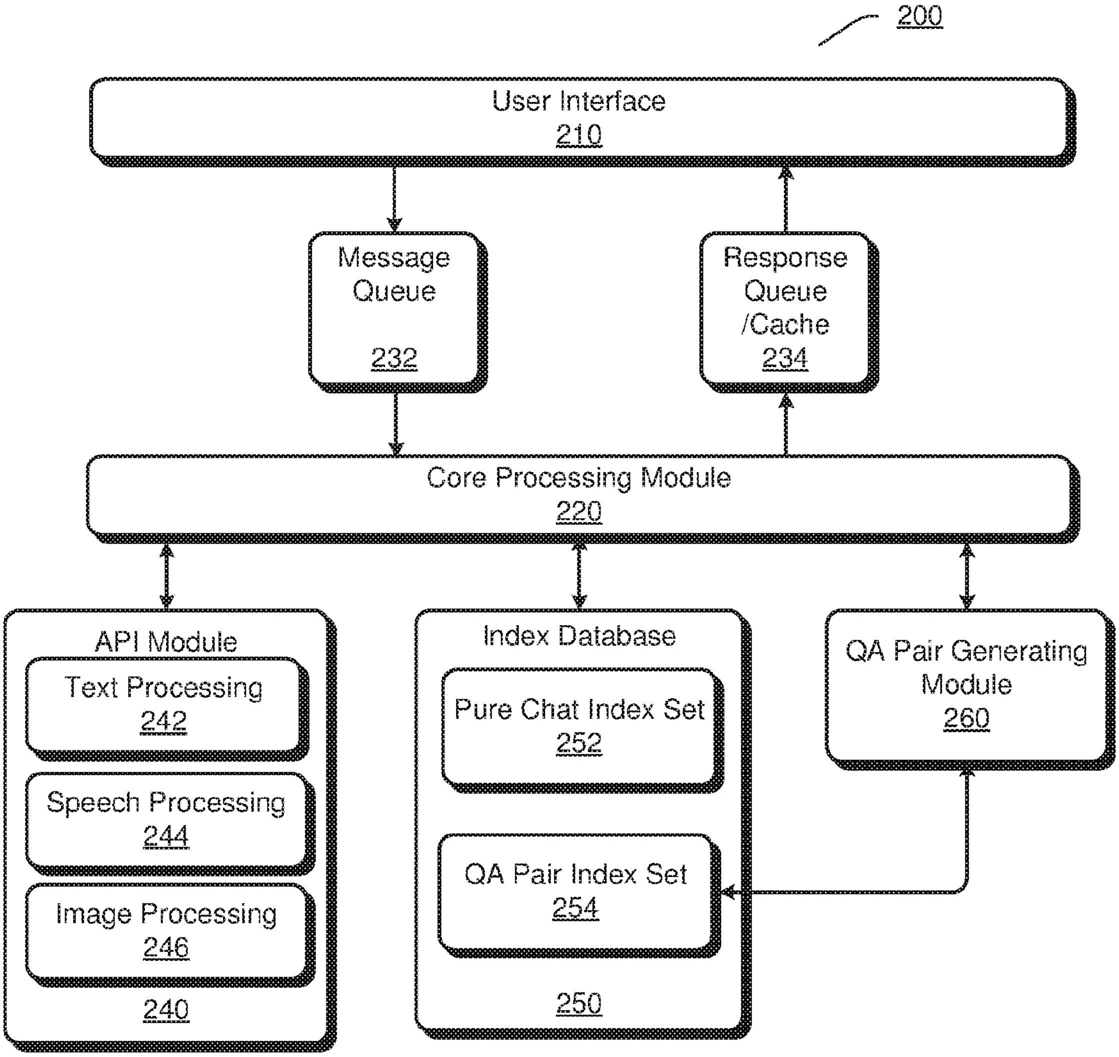
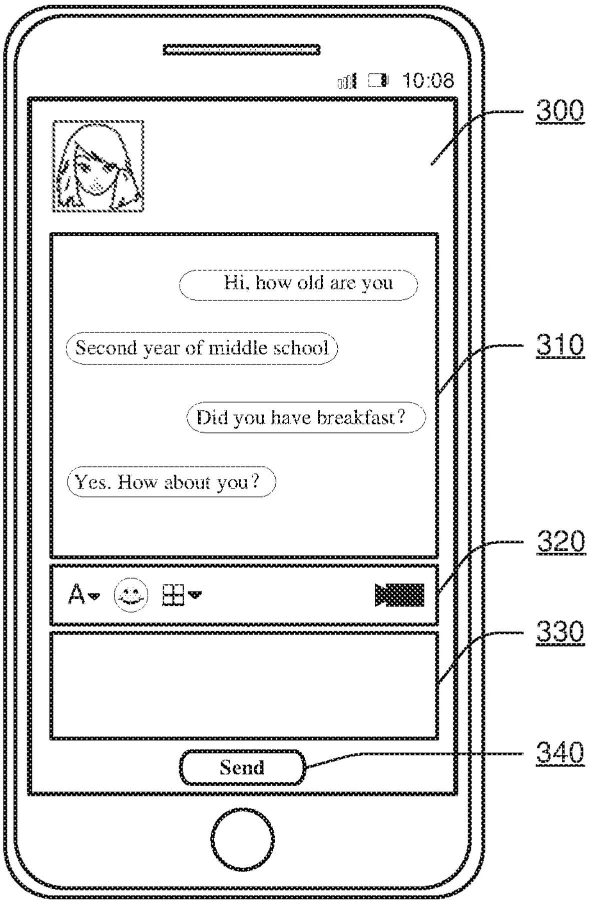

200
User Interface
210
Message
Queue
232
Response
Queue
/Cache
234
Core Processing Module
220
API Module
Text Processing
242
Speech Processing
244
Image Processing
246
240
Index Database
Pure Chat Index Set
252
QA Pair Index Set
254
250
QA Pair Generating
Module
260

FIG 2



Hi, how old are you

Second year of middle school

Did you have breakfast?

Yes. How about you?

FIG 3

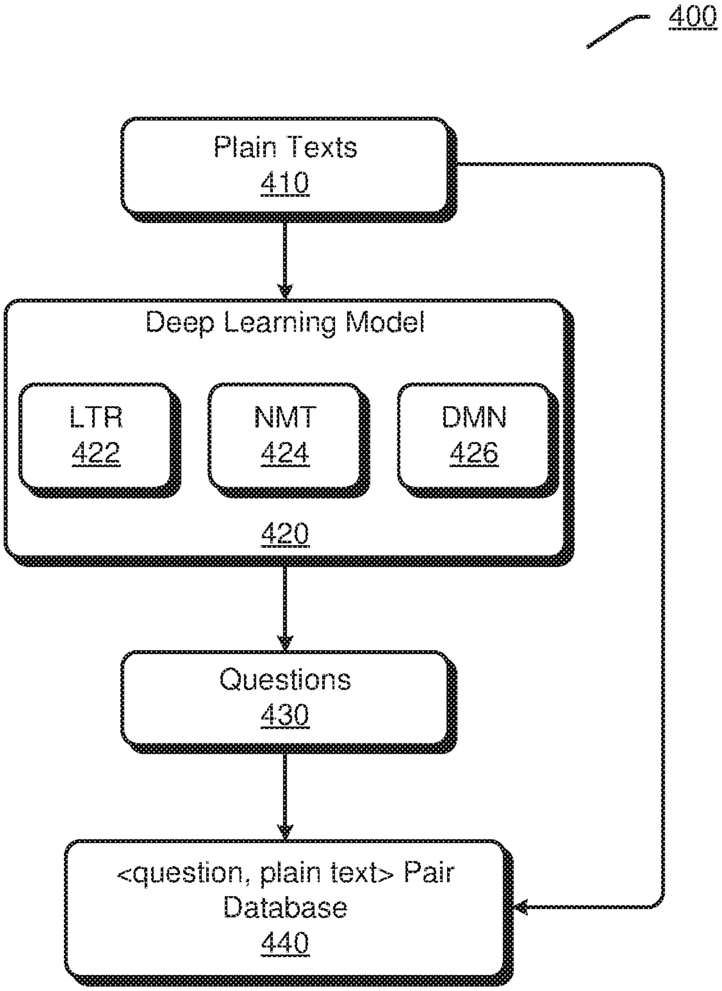

400
Plain Texts
410
Deep Learning Model
LTR
422
NMT
424
DMN
426
420
Questions
430
<question, plain text> Pair
Database
440

FIG 4

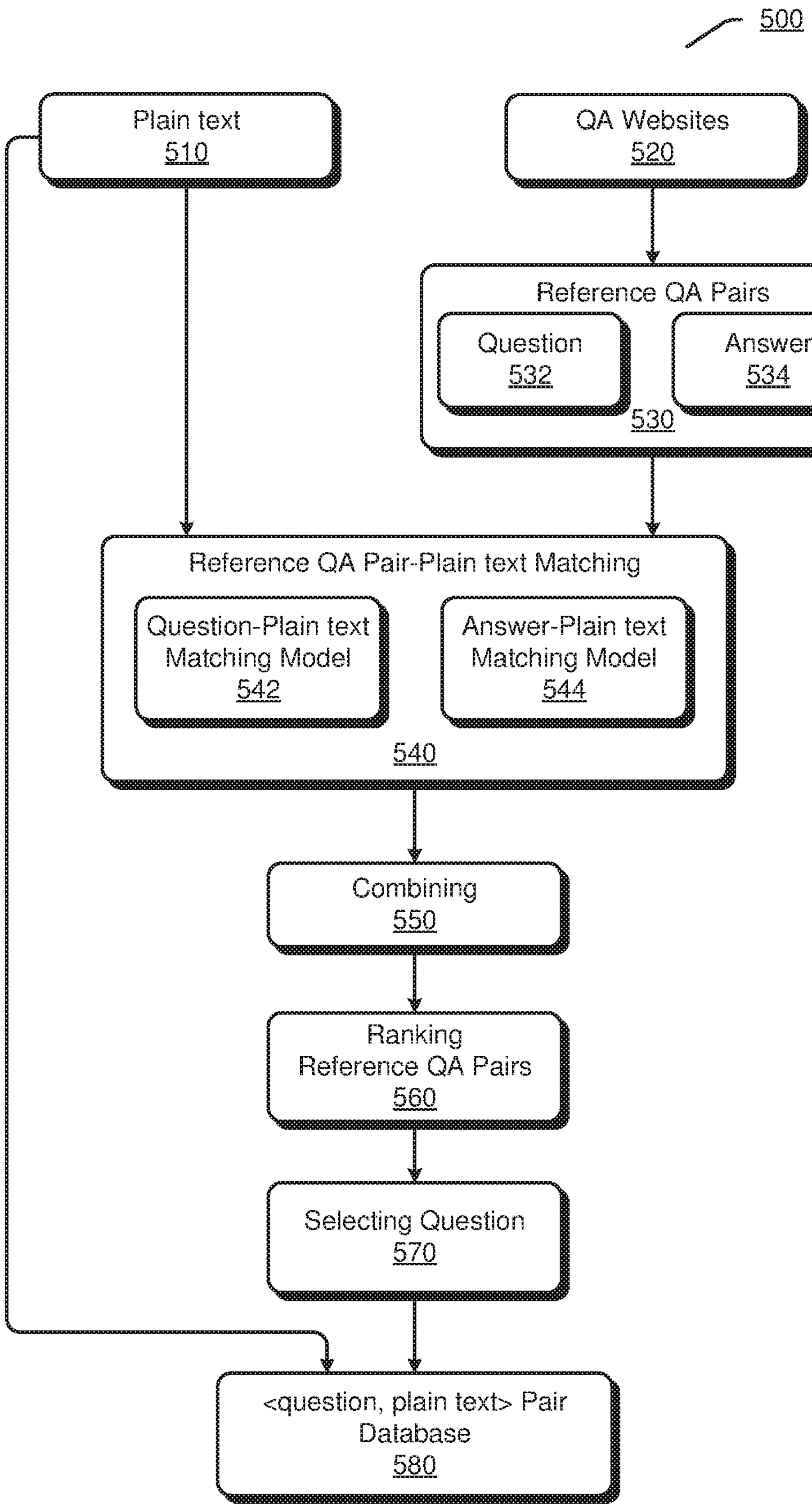
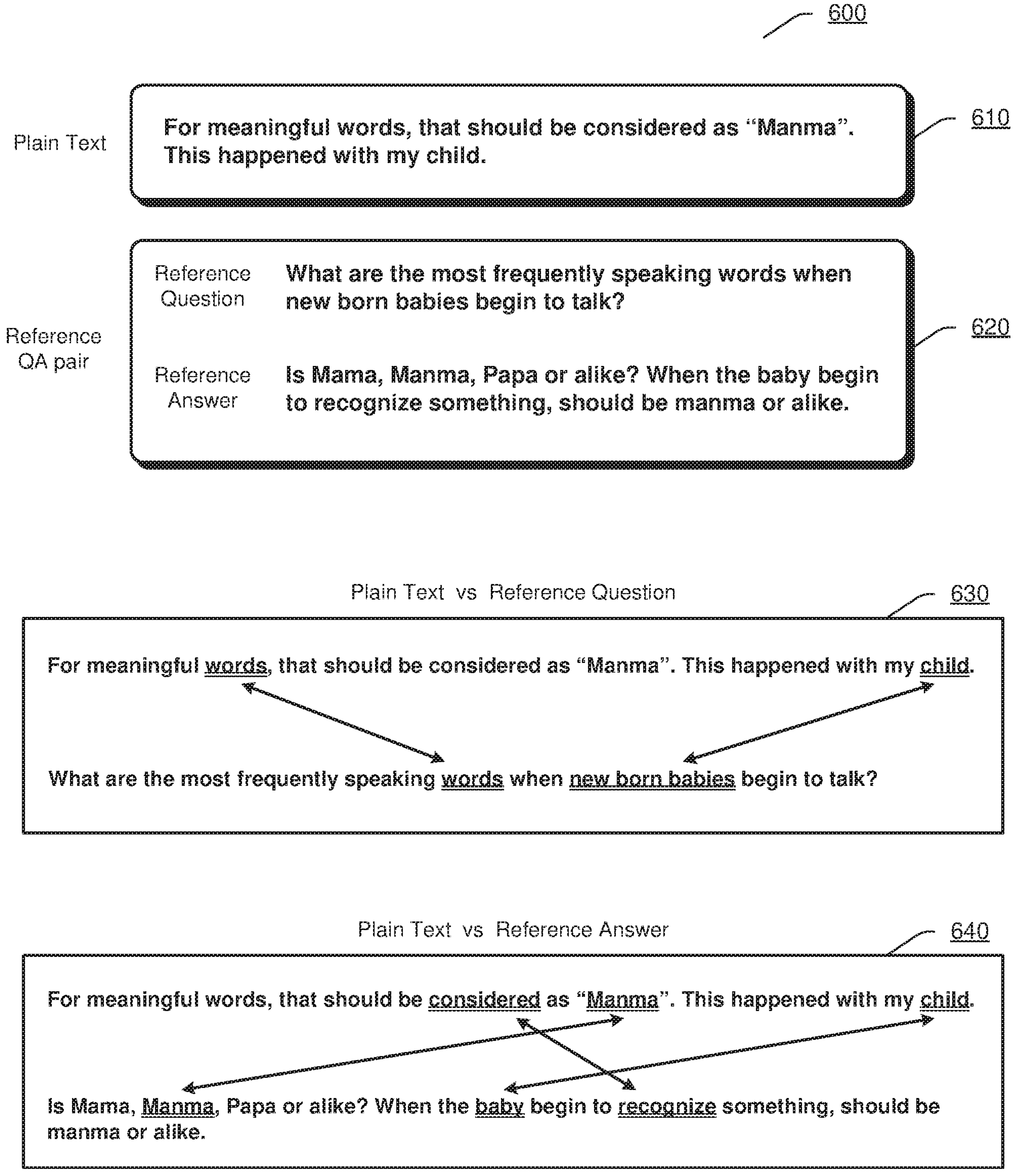

500
Plain text
510
QA Websites
520
Reference QA Pairs
Question
532
Answer
534
530
Reference QA Pair-Plain text Matching
Question-Plain text Matching Model
542
Answer-Plain text Matching Model
544
540
Combining
550
Ranking Reference QA Pairs
560
Selecting Question
570
<question, plain text> Pair Database
580

FIG 5



For meaningful words, that should be considered as "Manma". This happened with my child.

What are the most frequently speaking words when new born babies begin to talk?

Is Mama, Manma, Papa or alike? When the baby begin to recognize something, should be manma or alike.

For meaningful words, that should be considered as "Manma". This happened with my child.

What are the most frequently speaking words when new born babies begin to talk?

For meaningful words, that should be considered as "Manma". This happened with my child.

Is Mama, Manma, Papa or alike? When the baby begin to recognize something, should be manma or alike.

FIG 6

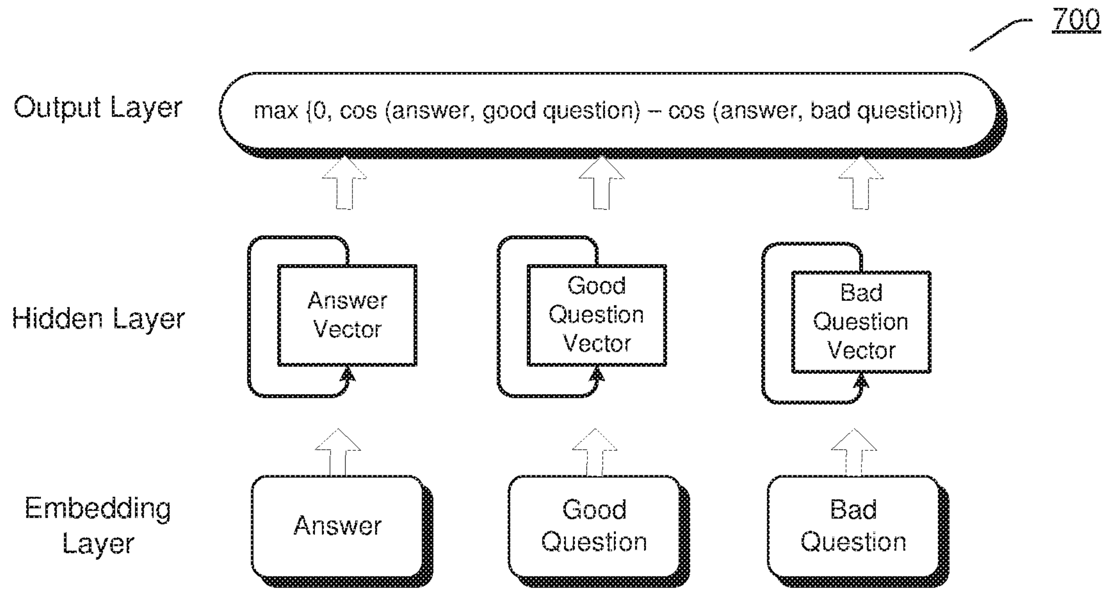


FIG 7

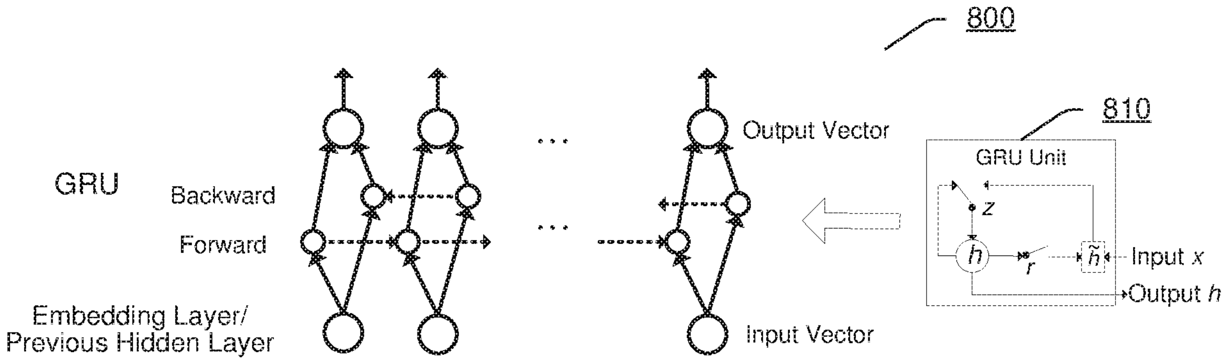


FIG 8

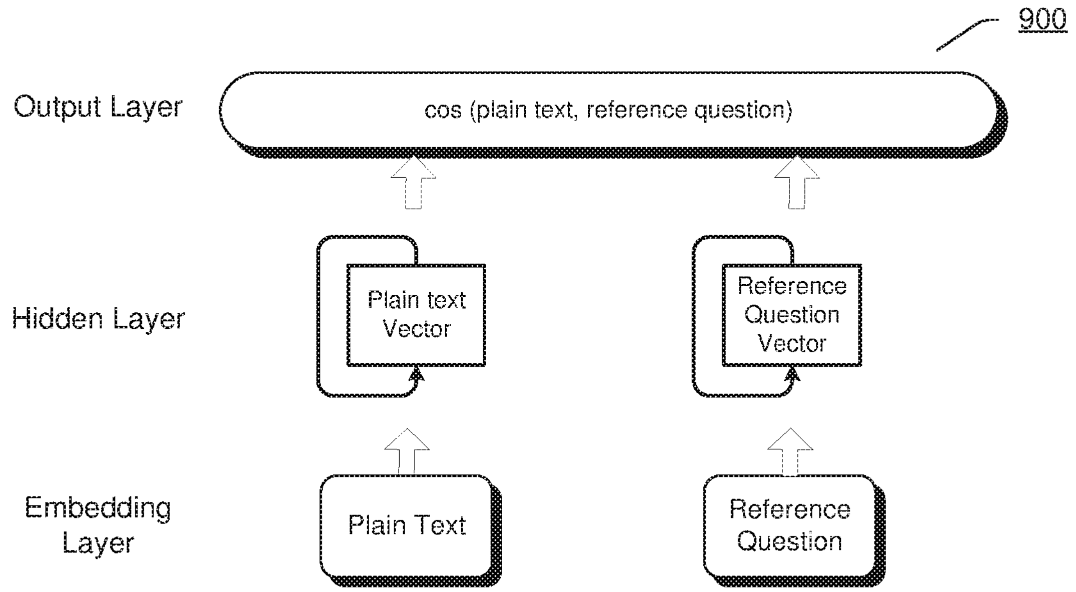


FIG 9

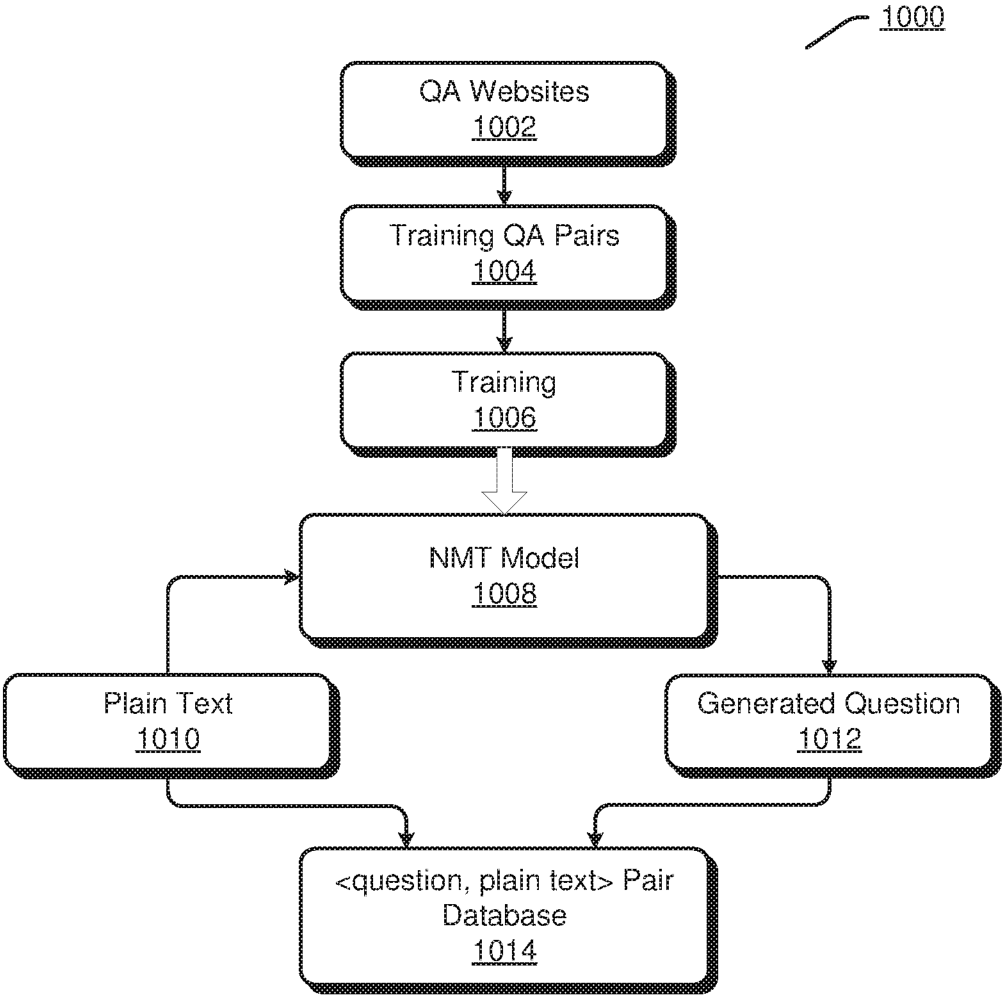

1000
QA Websites
1002
Training QA Pairs
1004
Training
1006
NMT Model
1008
Plain Text
1010
Generated Question
1012
<question, plain text> Pair Database
1014

FIG 10

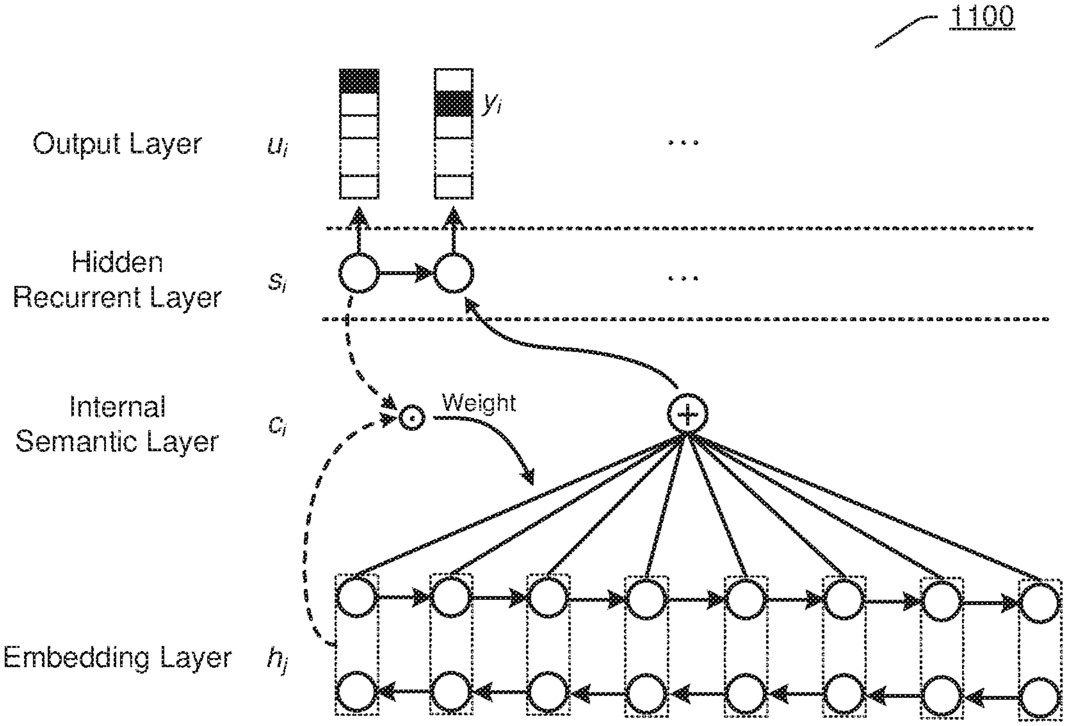

1100
Output Layer
u_i
y_i
...
Hidden Recurrent Layer
s_i
...
Internal Semantic Layer
c_i
Weight
Embedding Layer
h_j

FIG 11

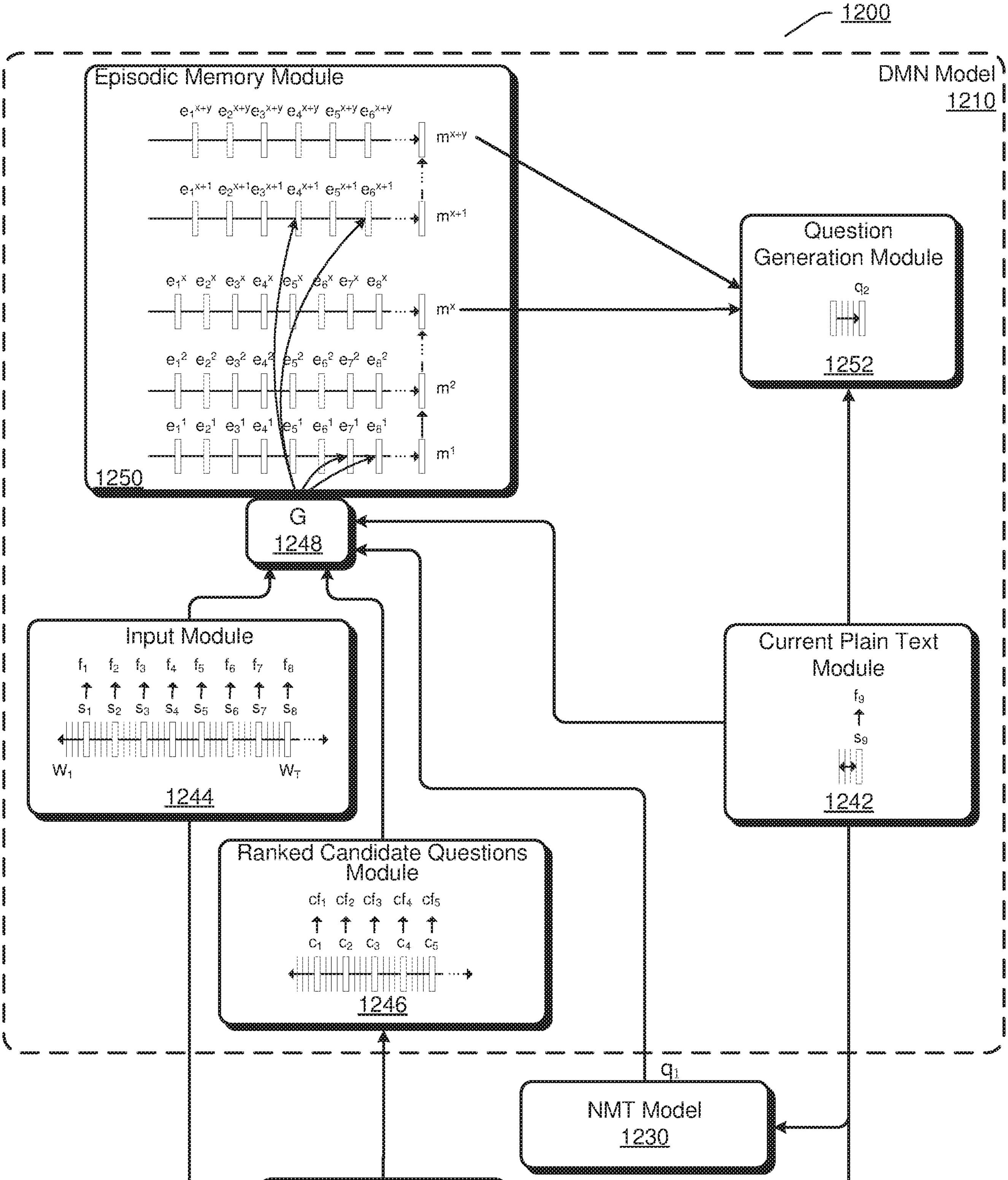
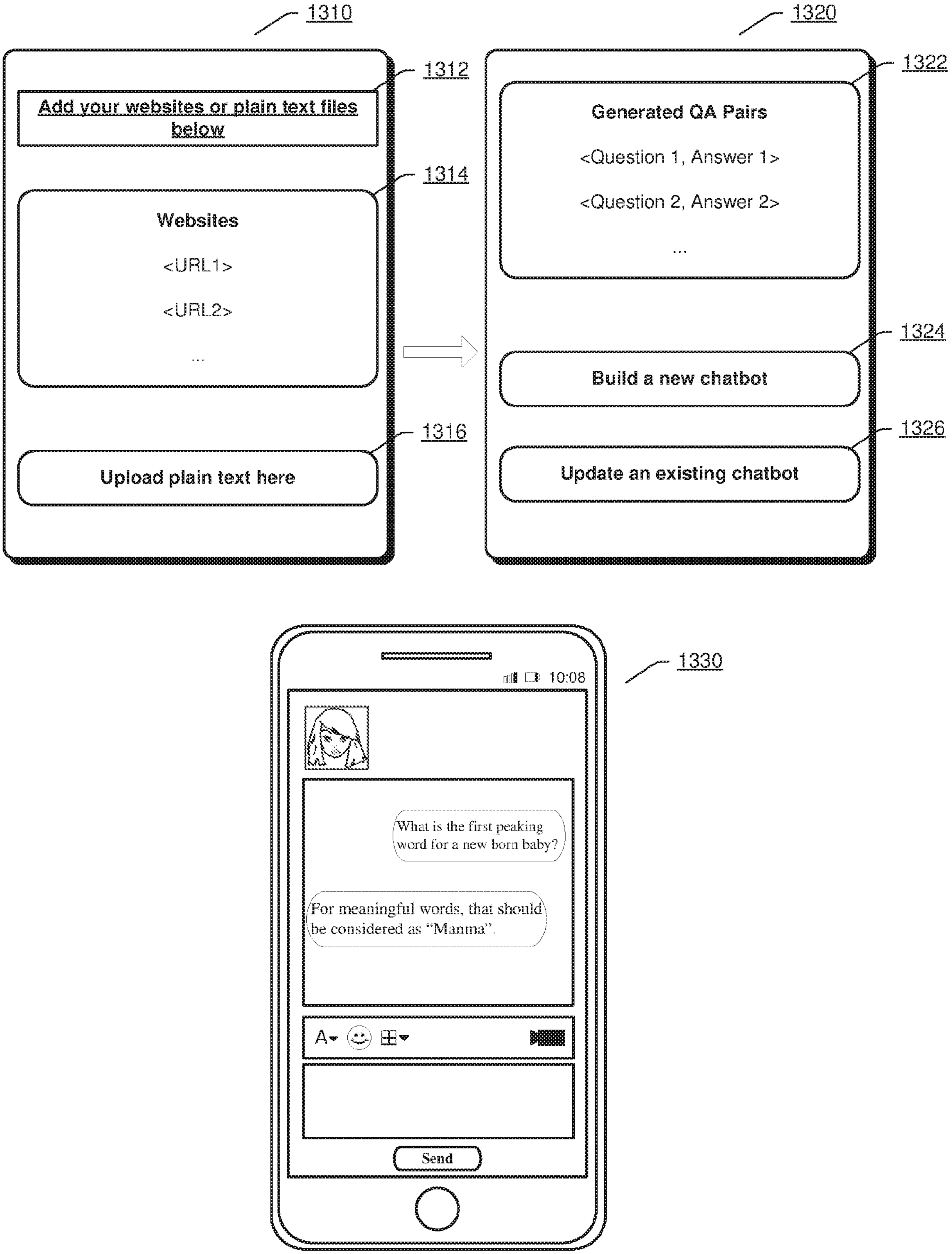

1200
DMN Model
1210
Episodic Memory Module
e_1^{x+y} e_2^{x+y} e_3^{x+y} e_4^{x+y} e_5^{x+y} e_6^{x+y}
m^{x+y}
e_1^{x+1} e_2^{x+1} e_3^{x+1} e_4^{x+1} e_5^{x+1} e_6^{x+1}
m^{x+1}
e_1^x e_2^x e_3^x e_4^x e_5^x e_6^x e_7^x e_8^x
m^x
e_1^2 e_2^2 e_3^2 e_4^2 e_5^2 e_6^2 e_7^2 e_8^2
m^2
e_1^1 e_2^1 e_3^1 e_4^1 e_5^1 e_6^1 e_7^1 e_8^1
m^1
1250
Question Generation Module
q_2
1252
G
1248
Input Module
f_1 f_2 f_3 f_4 f_5 f_6 f_7 f_8
s_1 s_2 s_3 s_4 s_5 s_6 s_7 s_8
W_1
W_T
1244
Current Plain Text Module
f_9
s_9
1242
Ranked Candidate Questions Module
cf_1 cf_2 cf_3 cf_4 cf_5
c_1 c_2 c_3 c_4 c_5
1246
q_1
NMT Model
1230
LTR Model
1220
Reference QA Pairs
1222

FIG 12



What is the first peaking word for a new born baby?

For meaningful words, that should be considered as "Mamma".

FIG 13

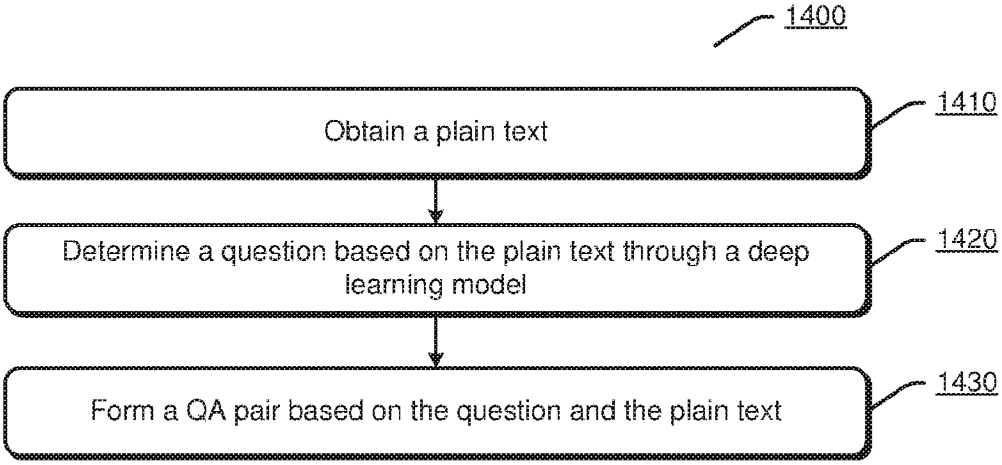

1400
Obtain a plain text
1410
Determine a question based on the plain text through a deep learning model
1420
Form a QA pair based on the question and the plain text
1430

FIG 14

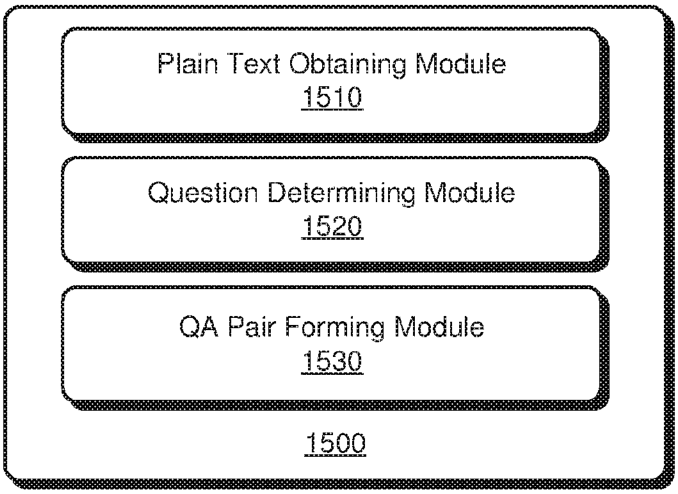


FIG 15

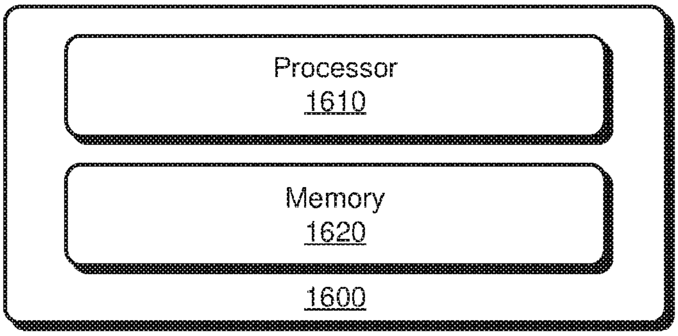


FIG 16

INTERNATIONAL SEARCH REPORT

International application No.

PCT/CN2017/082253

A. CLASSIFICATION OF SUBJECT MATTER

G06F 17/30(2006.01)i

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)

G06F G06N

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)

WPI,EPODOC,CNKI,CNPAT: QA, question, answer, pair, text, chat, deep learning, score, match, reference, LTR, GBDT, NMT, DMN

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	CN 106202301 A (WUHAN TIPDM INTELLIGENT TECHNOLOGY CO., LTD.) 07 December 2016 (2016-12-07) description, paragraphs [0035] to [0059]	1-20
X	CN 106295792 A (BEIJING GUANGNIANWUXIAN TECHNOLOGY CO., LTD.) 04 January 2017 (2017-01-04) description, paragraphs [0032] to [0046]	1-20
A	CN 106528538 A (EMOTIBOT TECHNOLOGIES LTD.) 22 March 2017 (2017-03-22) the whole document	1-20
A	US 2015262066 A1 (HUAWEI TECHNOLOGIES CO., LTD.) 17 September 2015 (2015-09-17) the whole document	1-20

□ 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 **21 December 2017**	Date of mailing of the international search report **24 January 2018**
Name and mailing address of the ISA/CN **STATE INTELLECTUAL PROPERTY OFFICE OF THE P.R.CHINA** **6, Xitucheng Rd., Jimen Bridge, Haidian District, Beijing 100088** **China**	Authorized officer **WANG,Xin**
Facsimile No. **(86-10)62019451**	Telephone No. **(86-10)62413253**

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.

PCT/CN2017/082253

Patent document cited in search report			Publication date (day/month/year)	Patent family member(s)			Publication date (day/month/year)
CN	106202301	A	07 December 2016	None			
CN	106295792	A	04 January 2017	None			
CN	106528538	A	22 March 2017	None			
US	2015262066	A1	17 September 2015	WO	2015139559	A1	24 September 2015
				CN	104933049	A	23 September 2015