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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
22 February 2018 (22.02.2018)

WIPO | PCT

(10) International Publication Number
WO 2018/035110 A1

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

(21) International Application Number:
PCT/US2017/046922

(22) International Filing Date:
15 August 2017 (15.08.2017)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
62/375,821 16 August 2016 (16.08.2016) US
15/439,044 22 February 2017 (22.02.2017) US

(63) Related by continuation (CON) or continuation-in-part (CIP) to earlier application:
US 15/439,044 (CON)
Filed on 22 February 2017 (22.02.2017)

(71) Applicant: **EBAY INC.** [US/US]; 2145 Hamilton Avenue, San Jose, California 95125 (US).

(72) Inventors: **LIU, Mingkuan**; 4932 Sandy Lane, San Jose, California 95124 (US). **ZHANG, Hao**; 996 Meridian Avenue, Apt 80, San Jose, California 95126 (US). **LIU, Xianjing**; 216 Bret Harte Ct, Santa Clara, California 95050 (US). **LU, Alan Qing**; 2065 Hamilton Avenue, San Jose, California 95125 (US).

(74) Agent: **SCHEER, Bradley W.** et al.; Schwegman Lundberg, & Woessner, P.A., P.O. Box 2938, Minneapolis, Minnesota 55402 (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, 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, JO, 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 of 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,

(54) Title: SEARCH OF PUBLICATION CORPUS WITH MULTIPLE ALGORITHMS

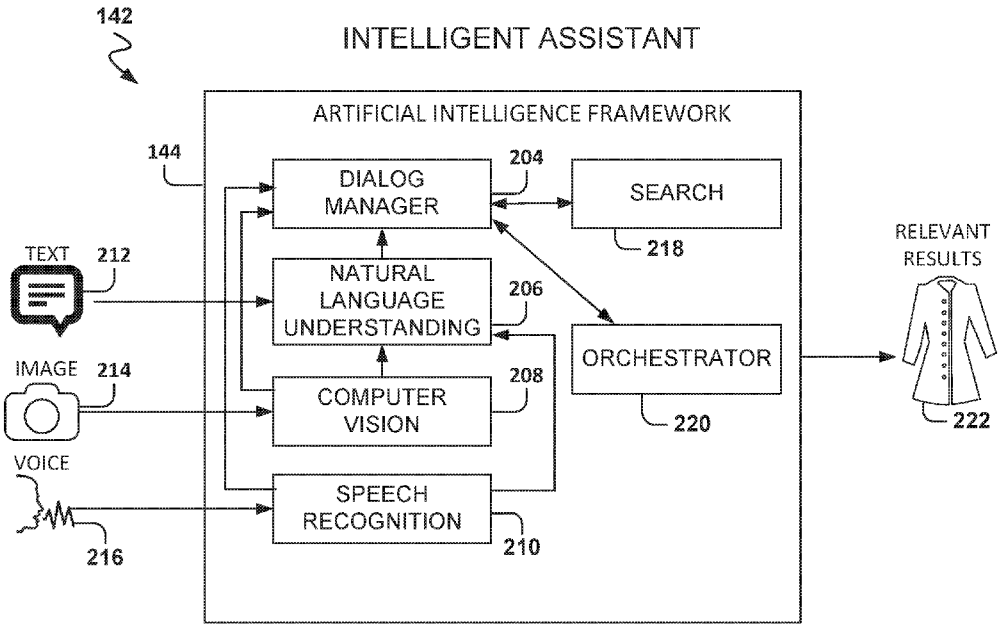


FIG. 2

(57) Abstract: Embodiments of the present disclosure relate generally to indexing with multiple algorithms to improve search results of a large corpus.

WO 2018/035110 A1

[Continued on next page]

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))*

SEARCH OF PUBLICATION CORPUS WITH MULTIPLE ALGORITHMS

RELATED APPLICATION(S)

[0001] This application claims the benefit of U.S. Application No. 15/439,044, filed February 22, 2017, which claims the benefit of U.S. Provisional Application No. 62/375,821, filed August 16, 2016, which are hereby incorporated by reference herein in their entireties.

TECHNICAL FIELD

[0002] Embodiments of the present disclosure relate generally to indexing with multiple algorithms to improve search results of a large corpus.

BACKGROUND

[0003] Present indexing techniques index a corpus based on one algorithm or another. This results in search results that ignore the advantages of different indexing algorithms and fail to be comprehensive.

BRIEF DESCRIPTION OF THE DRAWINGS

[0004] Various ones of the appended drawings merely illustrate example embodiments of the present disclosure and cannot be considered as limiting its scope.

[0005] **FIG. 1** is a block diagram illustrating a networked system, according to some example embodiments.

[0006] **FIG. 2** is a block diagram illustrating operation of the intelligent assistant, according to some example embodiments.

[0007] **FIG. 3** illustrates features of an artificial intelligence (AI) framework, according to some example embodiments.

[0008] **FIG. 4** is a diagram illustrating a service architecture according to some example embodiments.

[0009] **FIG. 5** illustrates a flow diagram of a method for training semantic machine learned models.

[0010] **FIG. 6** illustrates a flow diagram of a method for using a forward semantic search index to find closest matches in a publication corpus in response to a user search.

[0011] **FIG. 7** illustrates a block diagram of a method for making and using a reverse semantic search index to find closest matches in a publication corpus in response to a user search.

[0012] **FIG. 8** illustrates a more detailed version of the block diagram **FIG. 7** of a method for making and using a reverse semantic search index to find closest matches in a publication corpus in response to a user search.

[0013] **FIG. 9** illustrates a flow diagram of a method for trimming candidate publications to improve efficiency of finding closest matches in a publication corpus in response to a user search.

[0014] **FIG. 10** illustrates a flow diagram of a method for using multiple types of reverse indexes to more comprehensively find closest matches in a publication corpus in response to a user search.

[0015] **FIG. 11** illustrates an offline training process for search based on semantic vectorization.

[0016] **FIG. 12** illustrates a runtime search process for search based on semantic vectorization.

[0017] **FIG. 13** illustrates a query to publication retrieval process where multiple levels of relevance filtration/ranking are performed.

[0018] **FIG. 14** is a block diagram illustrating a process of generating semantic forward and reverse indexes.

[0019] **FIG. 15** is a block diagram illustrating an example of a software architecture that may be installed on a machine, according to some example embodiments.

[0020] The headings provided herein are merely for convenience and do not necessarily affect the scope or meaning of the terms used.

DETAILED DESCRIPTION

[0021] The description that follows includes systems, methods, techniques, instruction sequences, and computing machine program products that embody illustrative embodiments of the disclosure. In the following description, for the purposes of explanation, numerous specific details are set forth in order to provide an understanding of various embodiments of the inventive subject matter. It will be evident, however, to those skilled in the art, that embodiments of the inventive subject matter may be practiced without these specific details. In general, well-known instruction instances, protocols, structures, and techniques are not necessarily shown in detail.

[0022] Some embodiments generate a plurality of candidate publications in the publication corpus in response to a search query, comprising identifying multiple pluralities of candidate publications via multiple indexes that are mutually independent. This results in search results that leverage the different advantages of different indexing algorithms, such that the search results are comprehensive.

[0023] **FIG. 1** is a block diagram illustrating a networked system, according to some example embodiments. With reference to **FIG. 1**, an example embodiment of a high-level client-server-based network architecture 100 is shown. A networked system 102, in the example forms of a network-based publication system, provides server-side functionality via a network 104 (e.g., the Internet or wide area network (WAN)) to one or more client devices 110. One or more portions of network 104 may be an ad hoc network, an intranet, an extranet, a virtual private network (VPN), a local area network (LAN), a wireless LAN (WLAN), a wide area network (WAN), a wireless WAN (WWAN), a metropolitan area network (MAN), a portion of the Internet, a portion of the Public Switched Telephone Network (PSTN), a cellular telephone network, a wireless network, a WiFi network, a WiMax network, another type of network, or a combination of two or more such networks. The client device 110 may comprise, but is not limited to, a mobile phone, desktop computer, laptop, portable digital assistants (PDAs), smart phones, tablets, ultra books, netbooks, laptops, multi-

processor systems, microprocessor-based or programmable consumer electronics, game consoles, set-top boxes, or any other communication device that a user may utilize to access the networked system 102. In some embodiments, the client device 110 may comprise a display module (not shown) to display information (e.g., in the form of user interfaces). In further embodiments, the client device 110 may comprise one or more of a touch screens, accelerometers, gyroscopes, cameras, microphones, global positioning system (GPS) devices, and so forth. The client device 110 may be a device of a user that is used to perform a transaction involving digital publications within the networked system 102. In one embodiment, the networked system 102 is a network-based publication system that responds to requests for listings or publications and publishes publications. Examples of publications are item listings for sale.

[0024] Each of the client device 110 may include one or more applications (also referred to as "apps") such as, but not limited to, a web browser, messaging application, electronic mail (email) application, an e-commerce site application (also referred to as a marketplace application), and the like. **FIG. 1** illustrates, for example, a web client 112 (e.g., a browser, such as the Internet Explorer® browser developed by Microsoft® Corporation of Redmond, Washington State), an application 114, and a programmatic client (not shown) executing on the client device 110. The web client 112 may access an intelligent personal assistant system 142 via the web interface supported by the web server 122. Similarly, the programmatic client accesses the various services and functions provided by the intelligent personal assistant system 142 via the programmatic interface provided by the API server 120.

[0025] In some embodiments, if the e-commerce site application is included in a given one of the client device 110, then this application is configured to locally provide the user interface and at least some of the functionalities with the application configured to communicate with the networked system 102, on an as needed basis, for data or processing capabilities not locally available (e.g., access to a publication database of items available for sale, to authenticate a user, to verify a method of payment). Conversely if the e-commerce site application is not included in the client device 110, the client device 110 may use its web browser

to access the e-commerce site (or a variant thereof) hosted on the networked system 102.

[0026] One or more users 106 may be a person, a machine, or other means of interacting with the client device 110. In example embodiments, the user 106 is not part of the network architecture 100, but may interact with the network architecture 100 via the client device 110 or other means. For instance, the user provides input (e.g., touch screen input or alphanumeric input) to the client device 110 and the input is communicated to the networked system 102 via the network 104. In this instance, the networked system 102, in response to receiving the input from the user 106, communicates information to the client device 110 via the network 104 to be presented to the user. In this way, the user 106 can interact with the networked system 102 using the client device 110.

[0027] An application program interface (API) server 120 and a web server 122 are coupled to, and provide programmatic and web interfaces respectively to, one or more application servers 140. The application server 140 host an intelligent personal assistant system 142, which includes an artificial intelligence framework 144, each of which may comprise one or more modules or applications and each of which may be embodied as hardware, software, firmware, or any combination thereof.

[0028] The application server 140 is, in turn, shown to be coupled to one or more database servers 124 that facilitate access to one or more information storage repositories or databases 126. In an example embodiment, the databases 126 are storage devices that store information to be posted (e.g., publications or listings) to a publication system (not shown). The databases 126 may also store digital publication information in accordance with example embodiments.

[0029] Additionally, a third-party application 132, executing on third-party servers 130, is shown as having programmatic access to the networked system 102 via the programmatic interface provided by the API server 120. For example, the third-party application 132, utilizing information retrieved from the networked system 102, supports one or more features or functions on a website hosted by the third party. The third-party website, for example, provides one or more functions that are supported by the relevant applications of the networked system 102.

[0030] Further, while the client-server-based network architecture **100** shown in **FIG. 1** employs a client-server architecture, the present inventive subject matter is of course not limited to such an architecture, and could equally well find application in a distributed, or peer-to-peer, architecture system, for example. The intelligent personal assistant system 142 can also be implemented as a standalone system, which does not necessarily have networking capabilities.

[0031] **FIG. 2** is a diagram illustrating the operation of the intelligent assistant 142, according to some example embodiments. Conventionally, one cannot really perform a search using natural language. For example, today's online shopping is impersonal, unidirectional, and keyword-based, which is inadequate for buyers to express their exact intent, and is much harder compared to shopping in a store with the assistance of a helpful salesperson.

[0032] Example embodiments present a personal assistant, also referred to as an intelligent assistant, that supports human-like dialog with the user, enabling delivery of more relevant and personalized search results. The natural, human-like experience increases the likelihood that the user will reuse the intelligent assistant for future searches.

[0033] The artificial intelligence framework 144 understands the user and the available publications (e.g., inventory) to respond to natural-language queries and has the ability to deliver incremental improvements in anticipating and understanding the user and the user's needs.

[0034] The artificial intelligence framework (AIF) 144 includes a dialogue manager 204, natural language understanding (NLU) 206, computer vision 208, speech recognition 210 module, search module 218, and orchestrator 220. The AIF 144 is able to receive different kinds of inputs, such as text input 212, image input 214, voice input 216, or any combination of the three, to generate relevant results 222. As used herein, the AIF 144 includes a plurality of services (e.g., NLU 206, computer vision 208) that are implemented by corresponding servers, and the terms service or server may be utilized to identify the service and the corresponding server.

[0035] A natural language understanding (NLU) 206 unit processes natural language text input 212, both formal and informal language, detects the intent of the text, and extracts (e.g., parses) useful information, such as objects of interest and their attributes. The natural language text input 212 can thus be transformed into a structured query using rich information from additional knowledge to enrich the query even further. This information is then passed on to the dialog manager 204 through the orchestrator 220 for further actions with the user or with the other components in the overall system. The structured and enriched query is also consumed by the search 218 for improved matching. The text input 212 may be a query for a publication, a refinement to a previous query (e.g., size of shoes).

[0036] The computer vision 208 takes an image as an input and performs image recognition to identify characteristics of the image (e.g., object the user wants to search for), which are then transferred to the NLU 206 for processing. Similarly, the speech recognition 210 takes the speech 216 as an input and performs language recognition to convert speech to text, which is then transferred to the NLU 206 for processing.

[0037] The NLU 206 determines the object, the aspects associated with the object, how to create the search interface input, and how to generate the response. For example, the AIF 144 may ask questions to the user to clarify what the user is looking for. This means that the AIF 144 not only generates results, but also may create a series of interactive operations to get to relevant results 222.

[0038] For example, in response to the query, "Can you find me a pair of red Nike shoes?" the AIF 144 may generate the following parameters: <intent:shopping, statement-type:question, dominant-object:shoes, target:self, color:red, brand:nike>. To the query, "I am looking for a pair of sunglasses for my wife," the NLU may generate <intent:shopping, statement-type:statement, dominant-object:sunglasses, target:wife, target-gender:female>. In other example embodiments, digital embedding of image or voice data can be used directly, or in addition to text extraction, for search.

[0039] The dialogue manager 204 is the module that analyzes the query of a user to extract meaning, and determines if there is a question that needs to be asked in order to refine the query, before sending the query to the search 218. The

dialogue manager 204 uses the current communication in context of any previous communication between the user and the artificial intelligence framework 144. The questions are automatically generated dependent on the combination of the accumulated knowledge (e.g., provided by a knowledge graph) and what the search 218 can extract out of the inventory. The dialogue manager 204 creates a response for the user. For example, if the user says, "hello," the dialogue manager 204 generates a response, "Hi, my name is bot." The dialogue manager 204 pulls predetermined responses from a database or table that correlates the input "hello" to the predetermined responses, or generate a response from a machine-learned model.

[0040] The orchestrator 220 coordinates the interactions between the other services within the artificial intelligence framework 144. More details are provided below regarding the interactions of the orchestrator 220 with other services with reference to **FIG. 5**.

[0041] **FIG. 3** illustrates the features of the artificial intelligence framework (AIF) 144, according to some example embodiments. The AIF 144 is able to interact with several input channels 304, such as native commerce applications, chat applications, social networks, browsers, etc. In addition, the AIF 144 understands intent 306 expressed by the user. For example, the intent 306 may include a user looking for a good deal, a user looking for a gift, a user on a mission to buy a specific product, a user looking for suggestions, etc.

[0042] Further, the AIF 144 performs proactive data extraction 310 from multiple sources, such as social networks, email, calendar, news, market trends, etc. The AIF 144 knows or has access to user details 312, such as user preferences, desired price ranges, sizes, affinities, etc. The AIF 144 facilitates a plurality of services within the service network, such as search, recommendations, user profile / preference learning, checkout, etc. An output 308 may include search results, recommendations, checkout confirmation, etc.

[0043] The AIF 144 is an intelligent and friendly system that understands the user's intent (e.g., targeted search, compare, shop, browse), mandatory publication parameters (e.g., product, product category, item), optional publication parameters (e.g., aspects of the item, color, size, occasion), as well as

implicit information (e.g., geo location, personal preferences, age, gender). The AIF 144 responds with a well-designed response in natural language.

[0044] For example, the AIF 144 may process input queries, such as: "Hey! Can you help me find a pair of light pink shoes for my girlfriend please? With heels. Up to $200. Thanks;" "I recently searched for a men's leather jacket with a classic James Dean look. Think almost Harrison Ford's in the new Star Wars movie. However, I'm looking for quality in a price range of $200-300. Might not be possible, but I wanted to see!;" or "I'm looking for a black Northface Thermoball jacket."

[0045] Instead of a hardcoded system, the AIF 144 provides a configurable, flexible interface with machine learning capabilities for ongoing improvement. The AIF 144 supports a publication system that provides ease of use, value (e.g., connecting the user to the things that the user wants), intelligence (e.g., knowing and learning from the user and the user behavior to recommend the right publications), convenience (e.g., offering a plurality of user interfaces), and efficiency (e.g., saves the user time and money).

[0046] **FIG. 4** is a diagram illustrating a service architecture environment 400 according to some embodiments. The service architecture environment 400 presents various views of the service architecture in order to describe how the service architecture may be deployed on various data centers or cloud services. The service architecture environment 400 represents a suitable environment for implementation of the embodiments described herein.

[0047] A service architecture 402 represents how a cloud architecture typically appears to a user, developer, and so forth. The service architecture 402 is generally an abstracted representation of the actual underlying architecture implementation, represented in the other views of **FIG. 1**. For example, the service architecture 402 comprises a plurality of layers that represent different functionality and/or services associated with the service architecture 402.

[0048] An experience service layer 404 represents a logical grouping of services and features from an end user's point of view, built across different client platforms, such as applications running on a platform (e.g., mobile phone,

desktop), web based presentation (e.g., mobile web, desktop web browser), and so forth. The experience service layer 404 includes rendering user interfaces and providing information to the client platform so that appropriate user interfaces can be rendered, capturing client input, and so forth. In the context of a marketplace, examples of services that would reside in this layer include, for example, a home page (e.g., home view), view publication, search/view search results, shopping cart, buying user interface and related services, selling user interface and related services, after sale experiences (e.g., posting a transaction, feedback), and so forth. In the context of other systems, the experience service layer 404 incorporates those end user services and experiences that are embodied by the system.

[0049] An API layer 406 contains APIs which allow interaction with business process and core layers. This allows third party development against the service architecture 402 and allows third parties to develop additional services on top of the service architecture 402.

[0050] A business process service layer 408 is where the business logic resides for the services provided. In the context of a marketplace, this is where services such as user registration, user sign in, listing creation and publication, add to shopping cart, place an offer, checkout, send invoice, print labels, ship item, return item, and so forth are implemented. The business process service layer 408 also orchestrates between various business logic and data entities and thus represents a composition of shared services. The business processes in this layer can also support multi-tenancy in order to increase compatibility with some cloud service architectures.

[0051] A data entity service layer 410 enforces isolation around direct data access and contains the services upon which higher level layers depend. Thus, in the marketplace context, this layer comprises underlying services like order management, financial institution management, user account services, and so forth. The services in this layer typically support multi-tenancy.

[0052] An infrastructure service layer 412 comprises those services that are not specific to the type of service architecture being implemented. Thus, in the context of a marketplace, the services in this layer are services that are not specific or unique to a marketplace. Thus, functions like cryptographic functions, key

management, CAPTCHA, authentication and authorization, configuration management, logging, tracking, documentation and management, and so forth reside in this layer.

[0053] Embodiments of the present disclosure will typically be implemented in one or more of these layers, such as the AIF 144, as well as the orchestrator 220 and the other services of the AIF 144.

[0054] A data center 414 is a representation of various resource pools 416 along with their constituent scale units. The data center 414 representation illustrates the scaling and elasticity that comes with implementing the service architecture 402 in a cloud computing model. The resource pool 416 is comprised of server (or compute) scale units 420, network scale units 418 and storage scale units 422. A scale unit is a server, network, or storage unit that is the smallest unit capable of deployment within the data center. The scale units 418, 420, and 422 allow for more capacity to be deployed or removed as the need increases or decreases.

[0055] The network scale unit 418 contains one or more networks (e.g., network interface units) that can be deployed. The networks can include, for example virtual LANs. The compute scale unit 420 typically comprise a unit (e.g., server) that contains a plurality processing units, such as processors. The storage scale unit 422 contains one or more storage devices such as disks, storage attached networks (SAN), network attached storage (NAS) devices, and so forth. These are collectively illustrated as SANs in the description below. Each SAN may comprise one or more volumes, disks, and so forth.

[0056] The remainder of FIG. 4 illustrates another example of a service architecture environment 400. This view is more hardware focused and illustrates the resources underlying the more logical architecture in the other views of FIG. 4. A cloud computing architecture typically has a plurality of servers or other systems 424, 426. These servers comprise a plurality of real or virtual servers. Thus the server 424 comprises server 1 along with virtual servers 1A, 1B, 1C and so forth.

[0057] The servers are coupled to or interconnected by one or more networks such as network A 428 or network B 430. The servers are also coupled to a plurality of storage devices, such as SAN 1 (436), SAN 2 (438) and so forth. SANs are typically coupled to the servers through a network such as SAN access A 432 or SAN access B 434.

[0058] The compute scale units 420 are typically some aspect of servers 424 or 426, like processors, and other hardware associated therewith. The network scale units 418 typically include, or at least utilize, the illustrated networks A (428) and B (432). The storage scale units typically include some aspect of SAN 1 (436) or SAN 2 (438). Thus, the logical service architecture 402 can be mapped to the physical architecture.

[0059] Services and other implementation of the embodiments described herein will run on the servers or virtual servers and utilize the various hardware resources to implement the disclosed embodiments.

[0060] In various embodiments a machine-learned model encodes a search query or a portion of a search query into a continuous, dimensional vector space where semantic level similar sequences will have closer representation in this vector space. This machine-learned model approach can automatically capture the deep latent semantic meaning of a publication title, a search query, a portion of a publication title, or a portion of a search query and project its semantic level meaning into a shared multi-dimensional vector space. Such a model can be adapted to encode additional information such as user profile and session context on the input side, and other publication attributes on the inventory side.

[0061] Deep learning has recently shown much promise in Natural Language Processing (NLP). NLP researchers in this area are trying various ways to encode a sequence of symbols (e.g., phrases, sentences, paragraphs, and documents) into a multi-dimensional vector space, called semantic space. Semantic level similar sequences will have closer representation in this multi-dimensional space. Research in this area has led to an adoption of vector space representations of sentences instead of just words. Generally, phrases or sentences better define the contextual information rather than a single word. In various embodiments, research in sentence embedding is leveraged to

recommend publications of a seller on a publication system.

[0062] In an example embodiment, a machine-learned model is used to embed the deep latent semantic meaning of a given publication title and project it to a shared semantic vector space. A vector space can be referred to as a collection of objects called vectors. Vectors spaces can be characterized by their dimension, which specifies the number of independent directions in the space. A semantic vector space can represent phrases and sentences and can capture semantics for NLP tasks. In further embodiments, a semantic vector space can represent audio sounds, phrases, or music; video clips; and images and can capture semantics for NLP tasks.

[0063] In various embodiments, machine learning is used to maximize the similarity between the source (X), for example, a publication title, and the target (Y), the search query. A machine-learned model may be based on deep neural networks (DNN) or convolutional neural networks (CNN). The DNN is an artificial neural network with multiple hidden layers of units between the input and output layers. The DNN can apply the deep learning architecture to recurrent neural networks. The CNN is composed of one or more convolution layers with fully connected layers (such as those matching a typical artificial neural network) on top. The CNN also uses tied weights and pooling layers. Both the DNN and CNN can be trained with a standard backpropagation algorithm.

[0064] **FIG. 5** illustrates a flow diagram of a method for training semantic machine learned models.

[0065] When a machine-learned model is applied to mapping a specific <source, target> pair, the parameters for machine-learned Source Model and machine-learned Target Model are optimized so that relevant <source, target> pair has closer vector representation distance. The following formula can be used to compute the minimum distance.

$$SrcMod^*, TgtMod^* = \underset{k \text{ in all training pairs}}{\operatorname{argmin}} \sum \| SrcVec^k - TgtVec^k \|$$

Where,

ScrSeq = a source sequence;

TgtSeq = a target sequence;

SrcMod = source machine-learned model;

TgtMod = target machine-learned model;

SrcVec = a continuous vector representation for a source sequence (also referred to the semantic vector of the source); and

TgtVec = a continuous vector representation for a target sequence (also referred to as semantic vector of the target).

[0066] The source machine-learned model encodes the source sequence into a continuous vector representation. The target machine-learned model encodes the target sequence into a continuous vector representation. In an example embodiment, the vectors each have approximately 100 dimensions.

[0067] In other embodiments, any number of dimensions may be used. In example embodiments, the dimensions of the semantic vectors are stored in a KD tree structure. The KD tree structure can be referred to a space-partitioning data structure for organizing points in a KD space. The KD tree can be used to perform the nearest-neighbor lookup. Thus, given a source point in space, the nearest-neighbor lookup may be used to identify the closest point to the source point.

[0068] In **FIG. 5**, the left column is directed to training a machine learned model on search queries. In operation 510, a training search query is input to a first machine learned model. In operation 520, the training search query is processed by the first machine learned model of the neural network. In operation 530, the first machine learned model outputs a training semantic search vector.

[0069] In **FIG. 5**, the right column is directed to training a machine learned model on publications. In operation 540, a training publication is input to a second machine learned model. In operation 550, the training publication is processed by the second machine learned model of the neural network. In operation 560, the second machine learned model outputs a training publication semantic vector.

[0070] In operation 570, the training semantic search vector and the training publication semantic vector are compared to adjust the first machine learned model and the second machine learned model, such that semantically

similar training search queries and training publications result in respective semantic vectors that are proximate, and such that semantically dissimilar training search queries and training publications result in respective semantic vectors that are not proximate.

[0071] **FIG. 6** illustrates a flow diagram for using a forward semantic search index to find closest matches in a publication corpus in response to a user search.

[0072] In operation 610, a search query is received which is received from a device operated by a user searching for a publication listed in a publication corpus. In operation 620, a semantic search vector is accessed that encodes the search query, such that the semantic search vector represents the semantic meaning of the search query. The semantic search vector is pre-generated and just accessed in one embodiment. In another embodiment, the semantic search vector is generated and then accessed. The semantic search vector is based on the entire search query, multiple terms of the search query, or a single term in the event of short search queries.

[0073] Alternative paths exist for finding the closest matches between search queries and publications. In operation 630, the semantic search vector and publication vector are converted into binary representations. In operation 640, the closest matches are identified between the binary representations via XOR operations (which can be XOR or XNOR operations). In another embodiment, the semantic search vector or the publication vector are formed as binary representations to make the binary conversion superfluous. Alternatively in operation 650, closest matches are identified in a non-binary representation such as a floating point. In one embodiment, operation 650 gives higher quality results as operation 630 is not a lossless conversion. In one embodiment, operations 630-640 have sufficiently good results with a lower (and feasible) latency.

[0074] In operation 660, the closest matches are sent to the device operated by the user,and in some embodiments are displayed on the device. In operation 670, a selection signal is received, which is transmitted from the device operated by the user indicating a selection from the closest matches.

[0075] **FIG. 7** illustrates a block diagram of making and using a reverse

semantic search index to find closest matches in a publication corpus in response to a user search. **FIG. 7** shows word vector training and clustering 710, reverse index generation 720, and run-time recall retrieval 730.

[0076] A word vector based reverse-indexing approach captures various synonyms for a search query but without the costly requirement of maintaining a big synonym dictionary for every possible keyword. Keywords in the search query and publications are converted into semantic vectors. Those semantic vectors are grouped into a finite number of clusters in word vector training and clustering 710 and then used in reverse index generation 720. Based on those clusters of semantic vectors, a list of publications are fetched based on any search query in run-time recall retrieval 730.

[0077] In contrast with **FIG. 6**, in **FIG. 7** the semantic vector is a vector representing a single keyword in a search query or publication. For example, a search query "red namebrand shoes" has a single semantic vector [1.0, 2.4. 3.0]. While it has 3 different word vectors: "red" has a semantic vector of [3.2, 1.1, 0.5], "namebrand" has semantic vector of [-0.3, 2.6, 1.1], and "shoes" has a semantic vector of [1.3, 2.4, 3.5]. In other embodiments, the semantic vector is a vector representing multiple keywords in a search query or publication, but less than the whole search query or publication.

[0078] **FIG. 8** illustrates a more detailed version of the block diagram **FIG. 7** of making and using a reverse semantic search index to find closest matches in a publication corpus in response to a user search. The block diagram has primary components of word semantic vector training and clustering 810, reverse index generation 820, and run-time recall retrieval 830. Each primary block is discussed in turn.

[0079] In the beginning of semantic vector training and clustering 810, a publication corpus is collected 811. A Word2Vec or other tool performs word embedding 812 for every keyword entry in the vocabularies within the search query corpus and publication title corpus to make a word embedding dictionary 813. Clustering technology such as k-means clustering groups those word semantic vectors into a finite number of clusters based on their distance to each other 814. The result is a word vector cluster dictionary 815.

[0080] In the beginning of reverse index generation 820, publication IDs

are accessed 821, and keywords are extracted from the publication 822. For each keyword in the publication title, the encoding word vector value and is accessed and then the cluster ID is accessed 823. The publication ID is added to that cluster ID 824. After all keywords are processed, the word2vec reverse index is finalized 825.

[0081] For example, publication ID 12345 has a title of: "red athletic shoes" with keywords: red, athletic, and shoes. The encoding word vector clusterIDs are C1, C5, and C5. Thus, a word2vec reverse-index includes:

C1: 12345, ... (other PublicationIDs containing a word vector belonging to C1 cluster)

C5: 12345, ... (other PublicationIDs containing a word vector belonging to C5 cluster)

[0082] In the beginning of run-time recall retrieval 830, at search run-time, a search query is received 831. Keywords are extracted from the search query 832. The encoding word vector value and its cluster ID are accessed for each keyword 833. The clusterID is used to access PublicationIDs of publications 834, and the candidate publications are merged 835.

[0083] For example, a search query is "red running shoe" with keywords red, running, and shoe. The encoding word vector cluster IDs are: C1, C5 and C5. The publications are retrieved from in the reverse index for clusterID entries C1 and C5. The resulting PublicationIDs (and any others belonging to cluster IDs C1 and C5) are retrieved and merged.

[0084] **FIG. 9** illustrates a flow diagram of a method for trimming candidate publications to improve the efficiency of finding closest matches in a publication corpus in response to a user search.

[0085] At operation 910, for a keyword among potential search keywords, candidate publications are identified in the publication corpus. At operation 915, this is iterated for other keywords among potential search keywords.

[0086] At operation 920, for a keyword among the publication corpus, candidate publications are identified in the publication corpus. At operation 925, this is iterated for other keywords among potential search keywords.

[0087] At operation 930, the candidate publications are aggregated from operations 910 and 920. At operation 940, candidate publications are trimmed,

based on (i) machine-learned model of keywords to relevant publications or (ii) historic user behavior such as purchase, selection, and/or mouse or other control signal. Such trimming may occur offline.

[0088] At operation 950, a search query is received from a device operated by a user searching for a publication listed in a publication corpus. At operation 960, responsive to the search query, the trimmed aggregated candidate publications are processed for closest matches with the search query. Because of the trimming, finding matches with the search query among the publications is faster.

[0089] **FIG. 10** illustrates a flow diagram of a method for using multiple types of reverse indexes to more comprehensively find closest matches in a publication corpus in response to a user search.

[0090] At operation 1005, a search query is received which was received from a device operated by a user searching for a publication listed in a publication corpus.

[0091] At operation 1010, for a keyword among potential search keywords, candidate publications are identified in the publication corpus with a semantic vector-based reverse index. At operation 1015, this is iterated for other keywords among potential search keywords.

[0092] At operation 1020, for a keyword among potential search keywords, candidate publications are identified in the publication corpus with another reverse index. At operation 1025, this is iterated for other keywords among potential search keywords.

[0093] At operation 1030, candidate publications are aggregated from operations 1010 and 1020. At operation 1040, a search query is received from a device operated by a user searching for a publication listed in a publication corpus. At operation 1050, in response to the search query, the aggregated candidate publications are processed for closest matches with the search query. Because of the multiple indexes of different types, the search results are more comprehensive.

[0094] Other example indexes comprise:

(i) N-gram based which is inclusive in what is indexed, but in some embodiments does not retain semantic meaning.

(ii) Name entity based which has more structure than N-gram, but in some embodiments does not understand semantic meaning.

(iii) Head query based memorization which saves time and is accurate due to the system relying on prior input from the user device. However, because head query based memorization is based on prior input from the user device, head query based memorization has limited coverage, is less adaptive, and does not cover new publications or new publication types that have not been the subject of prior input from the user device.

(iv) A variant of (iii) where and index of head to relevant publications are pre-populated and updated near-real-time.

[0095] **FIG. 11** illustrates an offline training process for search based on semantic vectorization.

[0096] Referring to **FIG. 11**, the source machine-learned model and the target machine-learned model are trained. In a bottom portion, training data flow 1103 accesses training data. In a left column, a search query flow 1123 trains the machine-learned model for search queries. In a right column a publication flow 1124 trains the machine-learned model for publications. Labeled training data pairs (e.g., search query, publication) are provided for training both the first machine-learned model and the second machine-learned model.

[0097] In the bottom portion for the training data flow 1103, training data is accessed. In 1110, a historic user behavior data table is accessed, which has <query, publication> value pairs. The historic user behavior data table may include data such as impression count, click count, watch count, adding to purchase cart count, and purchase count.

[0098] In 1120, out of the <query, publication> value pairs from the historic user behavior data table, relevant pairs are accessed from a pre-determined or selectable time period (e.g., the last 4 weeks).

[0099] In the left column for the search query flow 1123, the search query (Y) is received, shown as an example query 1125 of a user search query of a wireless baby monitor. Word hashing is performed on the search query (Y) with a raw sentence sequence 1130. In situations where there is a very large vocabulary word, hashing is performed on a sub-word unit. In various

embodiments, in 1140 letter 3-gram word hashing is performed.

[00100] In the search query flow 1123, a deep neural network (DNN) is used to extract semantic vector representations of the search query (Y). The DNN uses more than one neural network layer to project input sequences into a semantic vector space. In an example embodiment, a convolution layer 1150, a maximum pooling layer 1160, and a semantic layer 1170 represent neural network layers. Some embodiments also have a binary classifier converting a floating point vector in to a binary representation. A number of nodes (e.g., 100 nodes, 500 nodes, and multiple groups of 500 nodes as shown) may be configured in those neural network layers. In other embodiments, the number of nodes may be changed or configured to a different number, depending on the data size. Keywords and concepts are identified from the search query (Y) using convolution and max-pooling.

[00101] In the right column for the publication flow 1124, a publication (X) is received, shown as an example publication 1126 with a seller publication title of video monitors or a white wireless video baby monitor. For an example embodiment, a publication system is used to list publications and may include millions of publications including example publication 1126. Word hashing may be performed on the target publication with a raw sentence sequence 1130. In situations where there is a very large vocabulary word, hashing is performed on a sub-word unit. In various embodiments, in 1140 letter 3-gram word hashing is performed.

[00102] In the publication flow 1124, a deep neural network (DNN) is used to extract semantic vectors representations of the target publication (X). The DNN uses more than one neural network layer to project input sequences into a semantic vector space. In an example embodiment, the convolution layer 1150, the maximum pooling layer 1160, and the semantic layer 1170 represent neural network layers. Some embodiments also have a binary classifier converting the floating point vector into a binary representation. A number of nodes (e.g., 100 nodes, 500 nodes, and multiple groups of 500 nodes) may be configured in those neural network layers. In other embodiments, the number of nodes may be changed or configured to a different number, depending on the data size. Keywords and concepts are identified from the target publication (X) using

convolution and max-pooling.

[00103] Finally, at 1180 semantic vector distance between X and Y is used to measure the similarity between the semantic vectors representations of the search query (Y) and the semantic vector representations of publications (X). In an example embodiment, the semantic relevance, represented by the function sim (X, Y), is measured by cosine similarity. In an embodiment with binary representations of the semantic vectors, XOR operations are performed.

[00104] When both the source machine-learned model and the target machine-learned model are trained, the semantic vector representations for all of the search query entries can be pre-computed in advance using the machine-learned model. Additionally, when there is a need to map any new publication from a seller, the semantic vector representation of the publication can be projected into shared semantic vector space with the semantic vectors representations of the publication.

[00105] As indicated above, when the machine-learned model is applied to mapping a specific <source sequence, target sequence> pair, the parameters for machine-learned Source Model and machine-learned Target Model are optimized so that relevant <source, target> pair has closer vector representation distance. The following formula can be used to compute the minimum distance.

$$SrcMod^*, TgtMod^* = \arg\min \sum_{k \text{ in all training pairs}} \| SrcVec^k - TgtVec^k \|$$

Where,

SerSeq = a source sequence;

TgtSeq = a target sequence;

SrcMod = source machine-learned model;

TgtMod = target machine-learned model;

SrcVec = a continuous vector representation for a source sequence (also referred to the semantic vector of the source); and

TgtVec = a continuous vector representation for a target sequence (also referred to as semantic vector of the target).

[00106] The source machine-learned model encodes the source sequence into a continuous vector representation. The target machine-learned model

encodes the target sequences into a continuous vector representations. In an example embodiment, the vectors each have approximately 100 dimensions. Alternatively, XOR operations can be performed on the binary representations of the semantic search vector and the publication vectors.

[00107] The results of the training process are two optimized neural network models - one the query model, another the publicationTitle model).

[00108] When a relevant pair of <query, publicationTitle> is input to the <query_model, publicationTitle_model> respectively, the inferenced semantic vector pair <query_semantic_vector, publicationTitle_semantic_vector> will have a closer distance.

[00109] When a non-relevant pair of <query, publicationTitle> is input to the <query_model, publicationTitle_model> respectively, the inferenced semantic vector pair <query_semantic_vector, publicationTitle_semantic_vector> will have a larger distance.

[00110] **FIG. 12** illustrates a runtime search process for search based on semantic vectorization. The left column is directed to the addition of a publication to the publication corpus. The right column is directed to searching for a publication in a publication corpus, in response to a search query transmitted from a device operated by a user.

[00111] Addition of a publication to the publication corpus is covered generally at 1202 to 1212. At 1202, a new publication X is added to the publication website or other storage. At 1204, the information of the new publication X is parsed into its components, such as title, description, and aspects. At 1206, sequence semantic embedding (SSE) encodes publication titles (e.g., title keywords for publications being listed) and category tree paths into semantic vector representations as <source sequence, target sequence> pairs. In various embodiments, SSE is used to encode a sequence of symbols (e.g., a phrase, a sentence or a paragraph) into a continuous, dimensional vector space where semantic level similar sequences will have closer representation in this semantic vector space. This SSE approach can automatically capture the deep latent semantic meaning of a publication title and project its semantic level meaning into a shared multi-dimensional vector space.

[00112] At 1208, a semantic vector is generated for publication X, applying

the SSE approach to the remainder of publication X, such as the description and aspects. At 1210, publication X is inserted into one or more forward indexes. At 1212, all live publication vectors are stored in memory to reduce access times.

[00113] Searching for a publication in a publication corpus, in response to a search query transmitted from a device operated by a user, is generally covered at 1230 to 1242. At 1230, a search query Y is received from a user device. At 1232, sequence semantic embedding (SSE) query elements into semantic vector representations as <source sequence, target sequence> pairs. At 1234, a semantic vector is generated for search query Y, applying the SSE approach to the search query Y.

[00114] At 1220, L0 filtering is performed using both candidate publications X from and on search query Y. L0 filtering is shown in more detail at FIG. 13. At 1236, semantic vector distance between X and Y is used to measure the similarity between the semantic vectors representations of the search query (Y) and the semantic vector representations of publications (X). In an example embodiment, the semantic relevance, represented by the function sim (X, Y), is measured by cosine similarity. In an embodiment with binary representations of the semantic vectors, XOR operations are performed.

[00115] At 1238, the best matched publication X to the search query Y is identified, according to the shortest distance in semantic space between semantic vectors of candidate publications X and the semantic vector of search query Y.

[00116] Certain publication updates impact the search index. The resulting search index update is performed similarly to a removal and a subsequent addition.

[00117] **FIG. 13** illustrates a query to publication retrieval process where multiple levels of relevance filtration/ranking are performed.

[00118] At 1302, a search query is received from a user device. At 1304, the search query is processed according to one or more algorithms, such as artificial intelligence (AI), named entity recognition (NER), or natural language understanding (NLU).

[00119] At 1310, L0 filtration is performed. Responsive to the processed search query from 1304, various components assist with gathering candidate

publications to match against the search query.

[00120] At 1312, semantic vector category mapping maps a search query to most relevant categories. Similar publications are placed in the same category. For example, a cellphone is listed under the "smartphone" category; while a book is listed under the "book" category. By applying the categorization block, search engine performance is much faster and returns more relevant results to the user. For example, if a user search query is "Smartphone X development," without applying the categorization block, the search engine can search against a large quantity (e.g., 600 million+) of publications for those keywords and might return some publications that are "Smartphone X" instead of a book about "Smartphone X development." After applying categorization, the "Category classification" block knows that this search query is about a book, and thus guides the search engine to only search a small quantity (e.g., around 100K) of publications under the book category for those keywords and return more relevant publications about the books of "Smartphone X development." This block thus significantly improves search engine's speed and accuracy.

[00121] At 1314, a semantic word2vector index is applied to a reverse semantic search index to extend query expansion. By adding search words that are semantically related to query words, the recall set of candidate publications is increased.

[00122] At 1316, query rewrite increases a search engine's recall set by capturing as many synonyms as possible. For example, if a search query is "sneakers for men" and a publication has a publication title of "running shoes," then without query rewrite, the publication will not be returned to the user because the publication title does not contain any keywords in the search query. So a synonym dictionary captures synonyms for keywords. For example, "sneakers," "running shoes," and "athletic shoes" are in a same synonym dictionary entry as having the same semantic meaning. "Query Rewrite" re-writes/expands the raw search query into multiple possible search queries according to the synonym dictionary. For example, the "sneakers for men" is rewritten/expanded into three search queries: "sneakers for men," "running shoes for men," "athletic shoes for men." The search engine returns publications which match with any of these rewrittten/expanded search queries. By applying

this "Query Rewrite" block, a publication with the publication title "running shoes" is returned to user even if its title does not contain any keywords in the raw search query.

[00123] At 1318, candidate publications are gathered into a reverse index publication list partitioned by category. The reverse index is generated from categorization service 1312, semantic word2vec reverse index 1314, and query rewrite 1316. At 1320, a top number (e.g., top 1000) of candidate publications are selected from the reverse index of 1318. A publication has a publicationID, an L0 score, and a BM25 score. The ranking is based on, for example, L0 score and BM25 score.

[00124] L1 ranking 1330 is performed after L0 filtration 1310. At 1332, a forward index is generated of semantic vectors for the candidate publications from 1320. The semantic vector mapping assists with mapping a query to a title at the sentence level and with ranking. In the semantic vector forward-index case of a collection of key-value pairs, the "key" is the PublicationID of the publication, and the "value" is the semantic vector. Examples with a limited number of semantic vector dimensions are:

Publication123: [0.2, 1.5, -3.2]

Publication456: [0.5, 3.2, -2.1]

[00125] In one example, the semantic vector is a 200 bit floating point vector. Other examples have a different number of bits, and/or different representations such as fixed point. In one example, the semantic vector is generated from the machine learned model trained in the publication flow 1124 of FIG. 11.

At 1334, the query from 1304 is vectorized for comparison with the publication semantic vectors. In one example, the semantic vector is a 200 bit floating point vector. Other examples have a different number of bits, and/or different representations such as fixed point. In one example, the semantic vector is generated from the machine learned model trained in the search query flow 1123 of FIG. 11.

[00126] L2 aspect filtration 1340 is performed after L1 ranking 1330. At 1342, a forward aspect index 1342 extends beyond aspects provide by seller to find the most relevant candidate publications. Additional aspects are found from

the publication via NER (name entity recognition).

[00127] In the publication aspect forward-index case of a collection of key-value pairs, the "key" is the PublicationID, and the "value" is some aspect associated with the publication. An example is:

Publication123: [color_red, type_shoes, brand_shoebrand, ….]

[00128] L3 final ranking 1350 is performed after L2 aspect filtration 1340. The L3 ranking module 1352 is a machine-learned model that takes inputs such as historic user behavior data, L0 score, BM25 score, L1 score, reputation score, value, publication condition, user profile, session context, and demotion signal. Finally, at 1360, the most relevant N publications are returned to the user device, based on L3 ranking 1350.

[00129] **FIG. 14** is a block diagram illustrating the process of generating, from the publications, semantic forward and reverse indexes used in connection with other figures. The forward indexes have key-value pairs with the PublicationID as the key. The reverse indexes have key-value pairs with the PublicationID as the value.

[00130] The semantic forward and reverse indexes originate from data of publications of a publication corpus 1402, such as title, category, images, aspects, publication party such as a seller, etc. The publication data 1402 is processed by a NER (name entity recognition service) 1412 that extracts from publication entities information such as color, brand, type; an image recognition service 1416 that converts images to semantic vectors and text; and a semantic vectorization service 1414 that converts publications to a semantic vector.

[00131] Reverse indexes 1422 of collections of key-value pairs include:

[00132] (i) N-gram reverse index 1424 where the "key" is the keyword, and the value is a list of PublicationIDs of publications containing this keyword. A simple example with the keyword "red" is:

[00133] red: [publication123, publication456,…]

[00134] (ii) Entity-reverse index 1426 where the "key" is the entity word, and the value is a list of PublicationIDs containing of publications this entity word. A simple example with an aspect of the publication is:

[00135] type_shoes: [publication123, publication456,….]

[00136] (iii) Word2Vec-reverse index 1428 where the "key" is the word

vector's clusterID, and the value is a list of PublicationIDs which contain a keyword whose word-vector is this clusterID. A simple example is:

[00137] word_vector_cluster#123: [publication342, publication456,....]

[00138] Reverse index 1429 combines n-gram reverse index 1424, entity-reverse index 1426, and word2Vec-reverse index 1428. Reverse index 1429 is partitioned by publication category.

[00139] The forward indexes 1432 of collections of key-value pairs include:

[00140] (i) Forward semantic vector index 134 for publications in the publication corpus. Each publication in the publication corpus is represented by a semantic vector, such a vector of 200 bit floating point value.

[00141] (ii) Forward image index 1436 derived from publication images.

[00142] (iii) Forward aspect index 1438 derived from publication aspects.

EXAMPLE MACHINE ARCHITECTURE AND MACHINE-READABLE MEDIUM

[0100] **FIG. 15** is a block diagram illustrating components of a machine **1500**, according to some example embodiments, able to read instructions from a machine-readable medium (e.g., a machine-readable storage medium) and perform any one or more of the methodologies discussed herein. Specifically, **FIG. 15** shows a diagrammatic representation of the machine 1500 in the example form of a computer system, within which instructions 1510 (e.g., software, a program, an application, an applet, an app, or other executable code) for causing the machine 1500 to perform any one or more of the methodologies discussed herein may be executed. For example, the instructions 1510 may cause the machine 1500 to execute the flow diagrams of **FIGS. 5-10**. Additionally, or alternatively, the instructions 1510 may implement the servers associated with the services and components of the **FIGS. 1- 14**, and so forth. The instructions 1510 transform the general, non-programmed machine 1500 into a particular machine 1500 programmed to carry out the described and illustrated functions in the manner described.

[0101] In alternative embodiments, the machine 1500 operates as a standalone device or may be coupled (e.g., networked) to other machines. In a

networked deployment, the machine 1500 may operate in the capacity of a server machine or a client machine in a server-client network environment, or as a peer machine in a peer-to-peer (or distributed) network environment. The machine 1500 may comprise, but not be limited to, a switch, a controller, a server computer, a client computer, a personal computer (PC), a tablet computer, a laptop computer, a netbook, a set-top box (STB), a personal digital assistant (PDA), an entertainment media system, a cellular telephone, a smart phone, a mobile device, a wearable device (e.g., a smart watch), a smart home device (e.g., a smart appliance), other smart devices, a web appliance, a network router, a network switch, a network bridge, or any machine capable of executing the instructions 1510, sequentially or otherwise, that specify actions to be taken by the machine 1500. Further, while only a single machine 1500 is illustrated, the term "machine" shall also be taken to include a collection of machines 1500 that individually or jointly execute the instructions 1510 to perform any one or more of the methodologies discussed herein.

[0102] The machine 1500 may include processors 1504, memory/storage 1506, and I/O components 1518, which may be configured to communicate with each other such as via a bus 1502. In an example embodiment, the processors 1504 (e.g., a Central Processing Unit (CPU), a Reduced Instruction Set Computing (RISC) processor, a Complex Instruction Set Computing (CISC) processor, a Graphics Processing Unit (GPU), a Digital Signal Processor (DSP), an Application Specific Integrated Circuit (ASIC), a Radio-Frequency Integrated Circuit (RFIC), another processor, or any suitable combination thereof) may include, for example, a processor 1508 and a processor 1512 that may execute the instructions 1510. The term "processor" is intended to include multi-core processors that may comprise two or more independent processors (sometimes referred to as "cores") that may execute instructions contemporaneously. Although **FIG. 15** shows multiple processors 1504, the machine 1500 may include a single processor with a single core, a single processor with multiple cores (e.g., a multi-core processor), multiple processors with a single core, multiple processors with multiples cores, or any combination thereof.

[0103] The memory/storage 1506 may include a memory 1514, such as a main memory, or other memory storage, and a storage unit 1516, both accessible to the processors 1504 such as via the bus 1502. The storage unit 1516 and memory 1514 store the instructions 1510 embodying any one or more of the methodologies or functions described herein. The instructions 1510 may also reside, completely or partially, within the memory 1514, within the storage unit 1516, within at least one of the processors 1504 (e.g., within the processor's cache memory), or any suitable combination thereof, during execution thereof by the machine 1500. Accordingly, the memory 1514, the storage unit 1516, and the memory of the processors 1504 are examples of machine-readable media.

[0104] As used herein, "machine-readable medium" means a device able to store or carry instructions and data temporarily or permanently and may include, but is not limited to, random-access memory (RAM), read-only memory (ROM), buffer memory, flash memory, optical media, magnetic media, cache memory, other types of storage (e.g., Erasable Programmable Read-Only Memory (EEPROM)), and/or any suitable combination thereof. The term "machine-readable storage medium" should be taken to include a single medium or multiple media (e.g., a centralized or distributed database, or associated caches and servers) able to store the instructions 1510. The term "machine-readable storage medium" shall also be taken to include any medium, or combination of multiple media, that is capable of storing instructions (e.g., instructions 1510) for execution by a machine (e.g., machine 1500), such that the instructions, when executed by one or more processors of the machine (e.g., processors 1504), cause the machine to perform any one or more of the methodologies described herein. Accordingly, a "machine-readable storage medium" refers to a single storage apparatus or device, as well as "cloud-based" storage systems or storage networks that include multiple storage apparatus or devices. The term machine readable medium includes a machine readable medium and a transmission medium or signals.

[0105] The I/O components 1518 may include a wide variety of components to receive input, provide output, produce output, transmit information, exchange information, capture measurements, and so on. The specific I/O components 1518 that are included in a particular machine will depend on the type of machine. For

example, portable machines such as mobile phones will likely include a touch input device or other such input mechanisms, while a headless server machine will likely not include such a touch input device. It will be appreciated that the I/O components 1518 may include many other components that are not shown in **FIG. 15**. The I/O components 1518 are grouped according to functionality merely for simplifying the following discussion, and the grouping is in no way limiting. In various example embodiments, the I/O components 1518 may include output components 1526 and input components 1528. The output components 1526 may include visual components (e.g., a display such as a plasma display panel (PDP), a light emitting diode (LED) display, a liquid crystal display (LCD), a projector, or a cathode ray tube (CRT)), acoustic components (e.g., speakers), haptic components (e.g., a vibratory motor, resistance mechanisms), other signal generators, and so forth. The input components 1528 may include alphanumeric input components (e.g., a keyboard, a touch screen configured to receive alphanumeric input, a photo-optical keyboard, or other alphanumeric input components), point based input components (e.g., a mouse, a touchpad, a trackball, a joystick, a motion sensor, or other pointing instruments), tactile input components (e.g., a physical button, a touch screen that provides location and/or force of touches or touch gestures, or other tactile input components), audio input components (e.g., a microphone), and the like.

[0106] In further example embodiments, the I/O components 1518 may include biometric components 1530, motion components 1534, environmental components 1536, or position components 1538 among a wide array of other components. For example, the biometric components 1530 may include components to detect expressions (e.g., hand expressions, facial expressions, vocal expressions, body gestures, or eye tracking), measure biosignals (e.g., blood pressure, heart rate, body temperature, perspiration, or brain waves), identify a person (e.g., voice identification, retinal identification, facial identification, fingerprint identification, or electroencephalogram based identification), and the like. The motion components 1534 may include acceleration sensor components (e.g., accelerometer), gravitation sensor components, rotation sensor components (e.g., gyroscope), and so forth. The environmental components 1536 may include,

for example, illumination sensor components (e.g., photometer), temperature sensor components (e.g., one or more thermometers that detect ambient temperature), humidity sensor components, pressure sensor components (e.g., barometer), acoustic sensor components (e.g., one or more microphones that detect background noise), proximity sensor components (e.g., infrared sensors that detect nearby objects), gas sensors (e.g., gas detection sensors to detect concentrations of hazardous gases for safety or to measure pollutants in the atmosphere), or other components that may provide indications, measurements, or signals corresponding to a surrounding physical environment. The position components 1538 may include location sensor components (e.g., a Global Position System (GPS) receiver component), altitude sensor components (e.g., altimeters or barometers that detect air pressure from which altitude may be derived), orientation sensor components (e.g., magnetometers), and the like.

[0107] Communication may be implemented using a wide variety of technologies. The I/O components 1518 may include communication components 1540 operable to couple the machine 1500 to a network 1532 or devices 1520 via a coupling 1524 and a coupling 1522, respectively. For example, the communication components 1540 may include a network interface component or other suitable device to interface with the network 1532. In further examples, the communication components 1540 may include wired communication components, wireless communication components, cellular communication components, Near Field Communication (NFC) components, Bluetooth® components (e.g., Bluetooth® Low Energy), Wi-Fi® components, and other communication components to provide communication via other modalities. The devices 1520 may be another machine or any of a wide variety of peripheral devices (e.g., a peripheral device coupled via a USB).

[0108] Moreover, the communication components 1540 may detect identifiers or include components operable to detect identifiers. For example, the communication components 1540 may include Radio Frequency Identification (RFID) tag reader components, NFC smart tag detection components, optical reader components (e.g., an optical sensor to detect one-dimensional bar codes such as Universal Product Code (UPC) bar code, multi-dimensional bar codes

such as Quick Response (QR) code, Aztec code, Data Matrix, Dataglyph, MaxiCode, PDF417, Ultra Code, UCC RSS-2D bar code, and other optical codes), or acoustic detection components (e.g., microphones to identify tagged audio signals). In addition, a variety of information may be derived via the communication components 1540, such as location via Internet Protocol (IP) geo-location, location via Wi-Fi® signal triangulation, location via detecting an NFC beacon signal that may indicate a particular location, and so forth.

[0109] In various example embodiments, one or more portions of the network 1532 may be an ad hoc network, an intranet, an extranet, a virtual private network (VPN), a local area network (LAN), a wireless LAN (WLAN), a wide area network (WAN), a wireless WAN (WWAN), a metropolitan area network (MAN), the Internet, a portion of the Internet, a portion of the Public Switched Telephone Network (PSTN), a plain old telephone service (POTS) network, a cellular telephone network, a wireless network, a Wi-Fi® network, another type of network, or a combination of two or more such networks. For example, the network 1532 or a portion of the network 1532 may include a wireless or cellular network and the coupling 1524 may be a Code Division Multiple Access (CDMA) connection, a Global System for Mobile communications (GSM) connection, or another type of cellular or wireless coupling. In this example, the coupling 1524 may implement any of a variety of types of data transfer technology, such as Single Carrier Radio Transmission Technology (1xRTT), Evolution-Data Optimized (EVDO) technology, General Packet Radio Service (GPRS) technology, Enhanced Data rates for GSM Evolution (EDGE) technology, third Generation Partnership Project (3GPP) including 3G, fourth generation wireless (4G) networks, Universal Mobile Telecommunications System (UMTS), High Speed Packet Access (HSPA), Worldwide Interoperability for Microwave Access (WiMAX), Long Term Evolution (LTE) standard, others defined by various standard-setting organizations, other long range protocols, or other data transfer technology.

[0110] The instructions 1510 may be transmitted or received over the network 1532 using a transmission medium via a network interface device (e.g., a network interface component included in the communication components 1540)

and utilizing any one of a number of well-known transfer protocols (e.g., hypertext transfer protocol (HTTP)). Similarly, the instructions 1510 may be transmitted or received using a transmission medium via the coupling 1522 (e.g., a peer-to-peer coupling) to the devices 1520. The term "transmission medium" shall be taken to include any intangible medium that is capable of storing, encoding, or carrying the instructions 1510 for execution by the machine 1500, and includes digital or analog communications signals or other intangible media to facilitate communication of such software. A transmission medium is an embodiment of a machine readable medium.

[00143] Certain embodiments are described herein as including logic or a number of components, modules, or mechanisms. Modules may constitute either software modules (e.g., code embodied on a machine-readable medium) or hardware modules. A "hardware module" is a tangible unit capable of performing certain operations and may be configured or arranged in a certain physical manner. In various example embodiments, one or more computer systems (e.g., a standalone computer system, a client computer system, or a server computer system) or one or more hardware modules of a computer system (e.g., a processor or a group of processors) may be configured by software (e.g., an application or application portion) as a hardware module that operates to perform certain operations as described herein.

[00144] In some embodiments, a hardware module may be implemented mechanically, electronically, or any suitable combination thereof. For example, a hardware module may include dedicated circuitry or logic that is permanently configured to perform certain operations. For example, a hardware module may be a special-purpose processor, such as a Field-Programmable Gate Array (FPGA) or an Application Specific Integrated Circuit (ASIC). A hardware module may also include programmable logic or circuitry that is temporarily configured by software to perform certain operations. For example, a hardware module may include software executed by a general-purpose processor or other programmable processor. Once configured by such software, hardware modules become specific machines (or specific components of a machine) uniquely tailored to perform the configured functions and are no longer general-purpose

processors. It will be appreciated that the decision to implement a hardware module mechanically, in dedicated and permanently configured circuitry, or in temporarily configured circuitry (e.g., configured by software) may be driven by cost and time considerations.

[00145] Accordingly, the phrase "hardware module" should be understood to encompass a tangible entity, be that an entity that is physically constructed, permanently configured (e.g., hardwired), or temporarily configured (e.g., programmed) to operate in a certain manner or to perform certain operations described herein. As used herein, "hardware-implemented module" refers to a hardware module. Considering embodiments in which hardware modules are temporarily configured (e.g., programmed), each of the hardware modules need not be configured or instantiated at any one instance in time. For example, where a hardware module comprises a general-purpose processor configured by software to become a special-purpose processor, the general-purpose processor may be configured as respectively different special-purpose processors (e.g., comprising different hardware modules) at different times. Software accordingly configures a particular processor or processors, for example, to constitute a particular hardware module at one instance of time and to constitute a different hardware module at a different instance of time.

[00146] Hardware modules can provide information to, and receive information from, other hardware modules. Accordingly, the described hardware modules may be regarded as being communicatively coupled. Where multiple hardware modules exist contemporaneously, communications may be achieved through signal transmission (e.g., over appropriate circuits and buses) between or among two or more of the hardware modules. In embodiments in which multiple hardware modules are configured or instantiated at different times, communications between such hardware modules may be achieved, for example, through the storage and retrieval of information in memory structures to which the multiple hardware modules have access. For example, one hardware module may perform an operation and store the output of that operation in a memory device to which it is communicatively coupled. A further hardware module may then, at a later time, access the memory device to retrieve and process the stored output. Hardware modules may also initiate

communications with input or output devices, and can operate on a resource (e.g., a collection of information).

[00147] The various operations of example methods described herein may be performed, at least partially, by one or more processors that are temporarily configured (e.g., by software) or permanently configured to perform the relevant operations. Whether temporarily or permanently configured, such processors may constitute processor-implemented modules that operate to perform one or more operations or functions described herein. As used herein, "processor-implemented module" refers to a hardware module implemented using one or more processors.

[00148] Similarly, the methods described herein may be at least partially processor-implemented, with a particular processor or processors being an example of hardware. For example, at least some of the operations of a method may be performed by one or more processors or processor-implemented modules. Moreover, the one or more processors may also operate to support performance of the relevant operations in a "cloud computing" environment or as a "software as a service" (SaaS). For example, at least some of the operations may be performed by a group of computers (as examples of machines including processors), with these operations being accessible via a network (e.g., the Internet) and via one or more appropriate interfaces (e.g., an Application Program Interface (API)).

[00149] The performance of certain of the operations may be distributed among the processors, not only residing within a single machine, but deployed across a number of machines. In some example embodiments, the processors or processor-implemented modules may be located in a single geographic location (e.g., within a home environment, an office environment, or a server farm). In other example embodiments, the processors or processor-implemented modules may be distributed across a number of geographic locations.

[00150] The modules, methods, applications and so forth described herein are implemented in some embodiments in the context of a machine and an associated software architecture. The sections below describe representative software architecture(s) and machine (e.g., hardware) architecture that are suitable for use with the disclosed embodiments.

[00151] Software architectures are used in conjunction with hardware architectures to create devices and machines tailored to particular purposes. For example, a particular hardware architecture coupled with a particular software architecture will create a mobile device, such as a mobile phone, tablet device, or so forth. A slightly different hardware and software architecture may yield a smart device for use in the "internet of things." While yet another combination produces a server computer for use within a cloud computing architecture. Not all combinations of such software and hardware architectures are presented here as those of skill in the art can readily understand how to implement the invention in different contexts from the disclosure contained herein.

[00152] The following numbered examples are embodiments.

[00153] 1. A computer-implemented method comprising:

accessing a search query of one or more keywords originating from a user device, the search query searching for a relevant set of publications in a publication corpus;

generating a plurality of candidate publications in the publication corpus in response to the search query, the generating comprising identifying multiple pluralities of candidate publications via multiple indexes that are mutually independent, including:

using a first index of the multiple indexes, for at least one keyword of the one or more keywords of the search query, identifying a first plurality of candidate publications in the publication corpus that correspond to a word vector based on a first machine-learned model that projects the at least one keyword into a semantic vector space; and

using a second index of the multiple indexes, for at least one keyword of the one or more keywords of the search query, identifying a second plurality of candidate publications in the publication corpus; and

causing display, on the user device, of a list of the plurality of candidate publications out of the first plurality of candidate publications and the second plurality of candidate publications, as the relevant set of publications in the publication corpus.

[00154] 2. The computer-implemented method of example 1, wherein at least one of the multiple indexes is a forward index.

[00155] 3. The computer-implemented method of example 1, wherein at least one of the multiple indexes is a reverse index.

[00156] 4. The computer-implemented method of any one of examples 1 to 3, further comprising generating the first index of the multiple indexes.

[00157] 5. The computer-implemented method of any one of examples 1 to 4, further comprising generating the second index of the multiple indexes.

[00158] 6. The computer-implemented method of any one of examples 1 to 5, wherein the search query includes non-textual input that includes any of user profile data, session context data, and at least any of audio input, video input, or image input.

[00159] 7. The computer-implemented method of any one of examples 1 to 6, further comprising after the causing the display, receiving a selection signal originating from the user device, the selection signal indicating a selection from the list of the plurality of candidate publications.

[00160] 8. A system comprising:

one or more processors and executable instructions accessible on a computer-readable medium that, when executed by the one or more processors, configure the one or more processors to at least perform operations comprising:

accessing a search query of one or more keywords originating from a user device, the search query searching for a relevant set of publications in a publication corpus;

generating a plurality of candidate publications in the publication corpus in response to the search query, the generating comprising identifying multiple pluralities of candidate publications via multiple indexes that are mutually independent, including:

using a first index of the multiple indexes, for at least one keyword of the one or more keywords of the search query, identifying a first plurality of candidate publications in the publication corpus that correspond to a word vector based on a first machine-learned model that projects the at least one keyword into a semantic vector space; and

using a second index of the multiple indexes, for at least one keyword of the one or more keywords of the search query, identifying a second plurality of candidate publications in the publication corpus; and

causing display, on the user device, of a list of the plurality of candidate publications out of the first plurality of candidate publications and the second plurality of candidate publications, as the relevant set of publications in the publication corpus.

[00161] 9. The system of example 8, wherein at least one of the multiple indexes is a forward index.

[00162] 10. The system of example 8, wherein at least one of the multiple indexes is a reverse index.

[00163] 11. The system of any one of examples 8 to 10, wherein the operations further comprise generating the first index of the multiple indexes.

[00164] 12. The system of any one of examples 8 to 11, wherein the operations further comprise generating the second index of the multiple indexes.

[00165] 13. The system of any one of examples 8 to 12, wherein the search query includes non-textual input that includes any of user profile data, session context data, and at least any of audio input, video input, or image input.

[00166] 14. The system of any one of examples 8 to 13, wherein the operations further comprise after the causing the display, receiving a selection signal originating from the user device, the selection signal indicating a selection from the list of the plurality of candidate publications.

[00167] 15. A machine-readable storage medium storing instructions that, when executed by one or more processors of a machine, cause the machine to at least perform operations comprising:

accessing a search query of one or more keywords originating from a user device, the search query searching for a relevant set of publications in a publication corpus;

generating a plurality of candidate publications in the publication corpus in response to the search query, the generating comprising identifying multiple pluralities of candidate publications via multiple indexes that are mutually independent, including:

using a first index of the multiple indexes, for at least one keyword of the one or more keywords of the search query, identifying a first plurality of candidate publications in the publication corpus that correspond to a word vector based on a first machine-learned model that projects the at least one keyword into a semantic vector space; and

using a second index of the multiple indexes, for at least one keyword of the one or more keywords of the search query, identifying a second plurality of candidate publications in the publication corpus; and

causing display, on the user device, of a list of the plurality of candidate publications out of the first plurality of candidate publications and the second plurality of candidate publications, as the relevant set of publications in the publication corpus.

[00168] 16. The machine-readable storage medium of example 15, wherein at least one of the multiple indexes is a forward index.

[00169] 17. The machine-readable storage medium of example 15, wherein at least one of the multiple indexes is a reverse index.

[00170] 18. The machine-readable storage medium of any one of examples 15 to 17, wherein the operations further comprise generating the first index of the multiple indexes.

[00171] 19. The machine-readable storage medium of any one of examples 15 to 18, wherein the operations further comprise generating the second index of the multiple indexes.

[00172] 20. The machine-readable storage medium of any one of examples 15 to 19, wherein the search query includes non-textual input that includes any of user profile data, session context data, and at least any of audio input, video input, or image input.

[00173] 21. A machine-readable medium carrying instructions that, when executed by one or more processors of a machine, cause the machine to carry out the method of any one of examples 1 to 7.

[00174] Throughout this specification, plural instances may implement components, operations, or structures described as a single instance. Although individual operations of one or more methods are illustrated and described as separate operations, one or more of the individual operations may be performed

concurrently, and nothing requires that the operations be performed in the order illustrated. Structures and functionality presented as separate components in example configurations may be implemented as a combined structure or component. Similarly, structures and functionality presented as a single component may be implemented as separate components. These and other variations, modifications, additions, and improvements fall within the scope of the subject matter herein.

[00175] Although an overview of the inventive subject matter has been described with reference to specific example embodiments, various modifications and changes may be made to these embodiments without departing from the broader scope of embodiments of the present disclosure. Such embodiments of the inventive subject matter may be referred to herein, individually or collectively, by the term "invention" merely for convenience and without intending to voluntarily limit the scope of this application to any single disclosure or inventive concept if more than one is, in fact, disclosed.

[00176] The embodiments illustrated herein are described in sufficient detail to enable those skilled in the art to practice the teachings disclosed. Other embodiments may be used and derived therefrom, such that structural and logical substitutions and changes may be made without departing from the scope of this disclosure. The Detailed Description, therefore, is not to be taken in a limiting sense, and the scope of various embodiments is defined only by the appended claims, along with the full range of equivalents to which such claims are entitled.

[00177] As used herein, the term "or" may be construed in either an inclusive or exclusive sense. Moreover, plural instances may be provided for resources, operations, or structures described herein as a single instance. Additionally, boundaries between various resources, operations, modules, engines, and data stores are somewhat arbitrary, and particular operations are illustrated in a context of specific illustrative configurations. Other allocations of functionality are envisioned and may fall within a scope of various embodiments of the present disclosure. In general, structures and functionality presented as separate resources in the example configurations may be implemented as a combined structure or resource. Similarly, structures and

functionality presented as a single resource may be implemented as separate resources. These and other variations, modifications, additions, and improvements fall within a scope of embodiments of the present disclosure as represented by the appended claims. The specification and drawings are, accordingly, to be regarded in an illustrative rather than a restrictive sense.

INDUSTRIAL APPLICABILITY

[00178] The invention disclosed herein has a wide range of applications, such as artificial intelligence and machine learning.

CLAIMS

1. A computer-implemented method comprising:

accessing a search query of one or more keywords originating from a user device, the search query searching for a relevant set of publications in a publication corpus;

generating a plurality of candidate publications in the publication corpus in response to the search query, the generating comprising identifying multiple pluralities of candidate publications via multiple indexes that are mutually independent, including:

using a first index of the multiple indexes, for at least one keyword of the one or more keywords of the search query, identifying a first plurality of candidate publications in the publication corpus that correspond to a word vector based on a first machine-learned model that projects the at least one keyword into a semantic vector space; and

using a second index of the multiple indexes, for at least one keyword of the one or more keywords of the search query, identifying a second plurality of candidate publications in the publication corpus; and

causing display, on the user device, of a list of the plurality of candidate publications out of the first plurality of candidate publications and the second plurality of candidate publications, as the relevant set of publications in the publication corpus.

2. The computer-implemented method of claim 1, wherein at least one of the multiple indexes is a forward index.

3. The computer-implemented method of claim 1, wherein at least one of the multiple indexes is a reverse index.

4. The computer-implemented method of claim 1, further comprising:

generating the first index of the multiple indexes.

5. The computer-implemented method of claim 1, further comprising:

generating the second index of the multiple indexes.

6. The computer-implemented method of claim 1, wherein the search query includes non-textual input that includes any of user profile data, session context data, and at least any of audio input, video input, or image input.

7. The computer-implemented method of claim 1, further comprising:

after the causing the display, receiving a selection signal originating from the user device, the selection signal indicating a selection from the list of the plurality of candidate publications.

8. A system comprising:

one or more processors and executable instructions accessible on a computer-readable medium that, when executed by the one or more processors, configure the one or more processors to at least perform operations comprising:

accessing a search query of one or more keywords originating from a user device, the search query searching for a relevant set of publications in a publication corpus;

generating a plurality of candidate publications in the publication corpus in response to the search query, the generating comprising identifying multiple pluralities of candidate publications via multiple indexes that are mutually independent, including:

using a first index of the multiple indexes, for at least one keyword of the one or more keywords of the search query, identifying a first plurality of candidate publications in the publication corpus that correspond to a word vector based on a first machine-learned model that projects the at least one keyword into a semantic vector space; and

using a second index of the multiple indexes, for at least one keyword of the one or more keywords of the search query, identifying a second plurality of candidate publications in the publication corpus; and

causing display, on the user device, of a list of the plurality of candidate publications out of the first plurality of candidate publications and the second plurality of candidate publications, as the relevant set of publications in the publication corpus.

9. The system of claim 8, wherein at least one of the multiple indexes is a forward index.

10. The system of claim 8, wherein at least one of the multiple indexes is a reverse index.

11. The system of claim 8, wherein the operations further comprise:

generating the first index of the multiple indexes.

12. The system of claim 8, wherein the operations further comprise:

generating the second index of the multiple indexes.

13. The system of claim 8, wherein the search query includes non-textual input that includes any of user profile data, session context data, and at least any of audio input, video input, or image input.

14. The system of claim 8, wherein the operations further comprise:

after the causing the display, receiving a selection signal originating from the user device, the selection signal indicating a selection from the list of the plurality of candidate publications.

15. A machine-readable storage medium storing instructions that, when executed by one or more processors of a machine, cause the machine to at least perform operations comprising:

accessing a search query of one or more keywords originating from a user device, the search query searching for a relevant set of publications in a publication corpus;

generating a plurality of candidate publications in the publication corpus in response to the search query, the generating comprising identifying multiple pluralities of candidate publications via multiple indexes that are mutually independent, including:

using a first index of the multiple indexes, for at least one keyword of the one or more keywords of the search query, identifying a first plurality of candidate publications in the publication corpus that correspond to a word vector based on a first machine-learned model that projects the at least one keyword into a semantic vector space; and

using a second index of the multiple indexes, for at least one keyword of the one or more keywords of the search query, identifying a second plurality of candidate publications in the publication corpus; and

causing display, on the user device, of a list of the plurality of candidate publications out of the first plurality of candidate publications and the second plurality of candidate publications, as the relevant set of publications in the publication corpus.

16. The machine-readable storage medium of claim 15, wherein at least one of the multiple indexes is a forward index.

17. The machine-readable storage medium of claim 15, wherein at least one of the multiple indexes is a reverse index.

18. The machine-readable storage medium of claim 15, wherein the operations further comprise:

generating the first index of the multiple indexes.

19. The machine-readable storage medium of claim 15, wherein the operations further comprise:

generating the second index of the multiple indexes.

20. The machine-readable storage medium of claim 15, wherein the search query includes non-textual input that includes any of user profile data, session context data, and at least any of audio input, video input, or image input.

21. A machine-readable medium carrying instructions that, when executed by one or more processors of a machine, cause the machine to carry out the method of any one of claims 1 to 7.

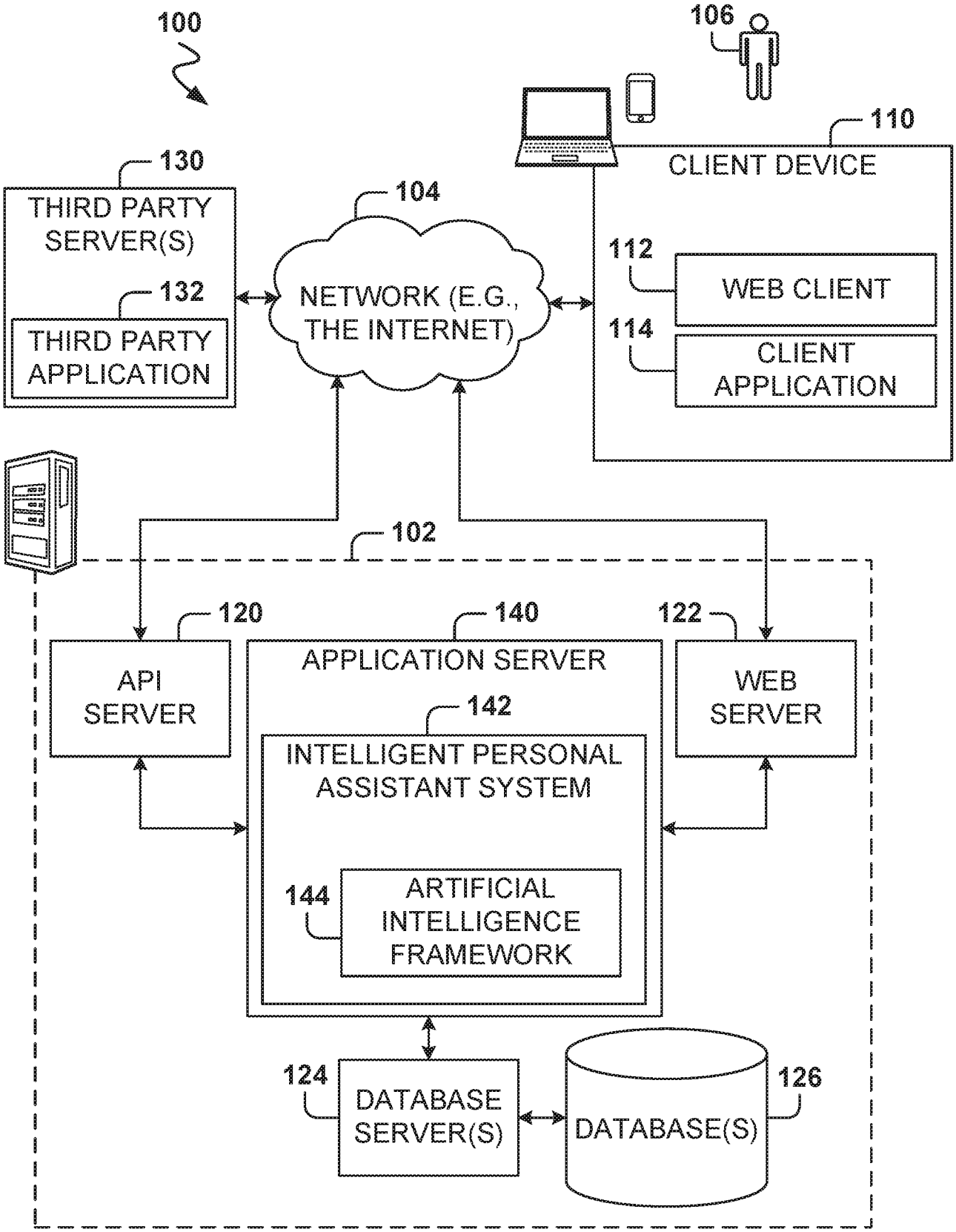

100
106
110
CLIENT DEVICE
130
THIRD PARTY SERVER(S)
104
NETWORK (E.G., THE INTERNET)
112
WEB CLIENT
132
THIRD PARTY APPLICATION
114
CLIENT APPLICATION
102
120
140
122
API SERVER
APPLICATION SERVER
WEB SERVER
142
INTELLIGENT PERSONAL ASSISTANT SYSTEM
144
ARTIFICIAL INTELLIGENCE FRAMEWORK
124
DATABASE SERVER(S)
126
DATABASE(S)

FIG. 1

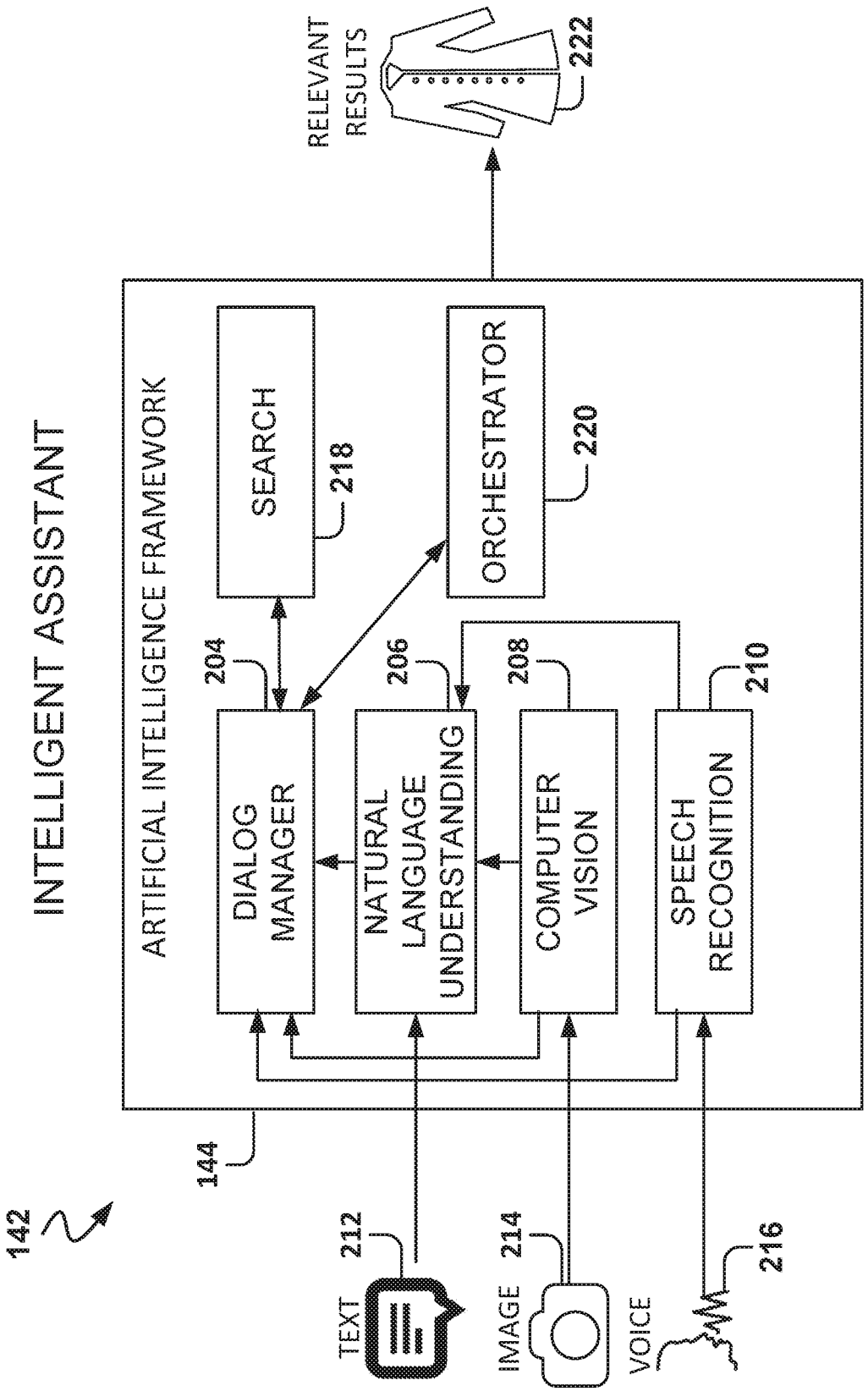

INTELLIGENT ASSISTANT
142
ARTIFICIAL INTELLIGENCE FRAMEWORK
144
DIALOG MANAGER
204
NATURAL LANGUAGE UNDERSTANDING
206
COMPUTER VISION
208
SPEECH RECOGNITION
210
SEARCH
218
ORCHESTRATOR
220
RELEVANT RESULTS
222
TEXT
212
IMAGE
214
VOICE
216

FIG. 2

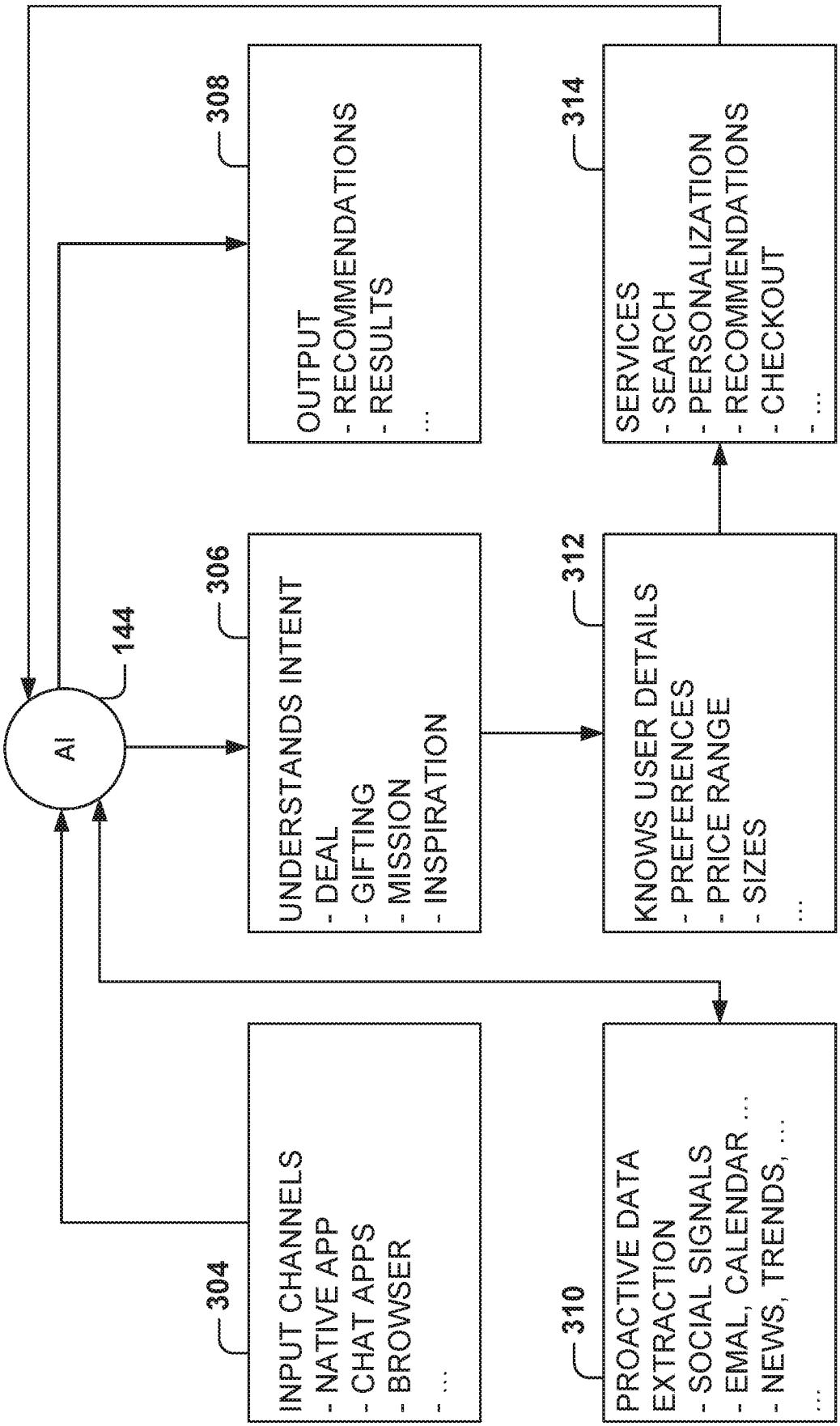

AI
144
INPUT CHANNELS
- NATIVE APP
- CHAT APPS
- BROWSER
- ...
304
UNDERSTANDS INTENT
- DEAL
- GIFTING
- MISSION
- INSPIRATION
306
OUTPUT
- RECOMMENDATIONS
- RESULTS
- ...
308
KNOWS USER DETAILS
- PREFERENCES
- PRICE RANGE
- SIZES
- ...
312
SERVICES
- SEARCH
- PERSONALIZATION
- RECOMMENDATIONS
- CHECKOUT
- ...
314
PROACTIVE DATA EXTRACTION
- SOCIAL SIGNALS
- EMAL, CALENDAR ...
- NEWS, TRENDS, ...
- ...
310

FIG. 3

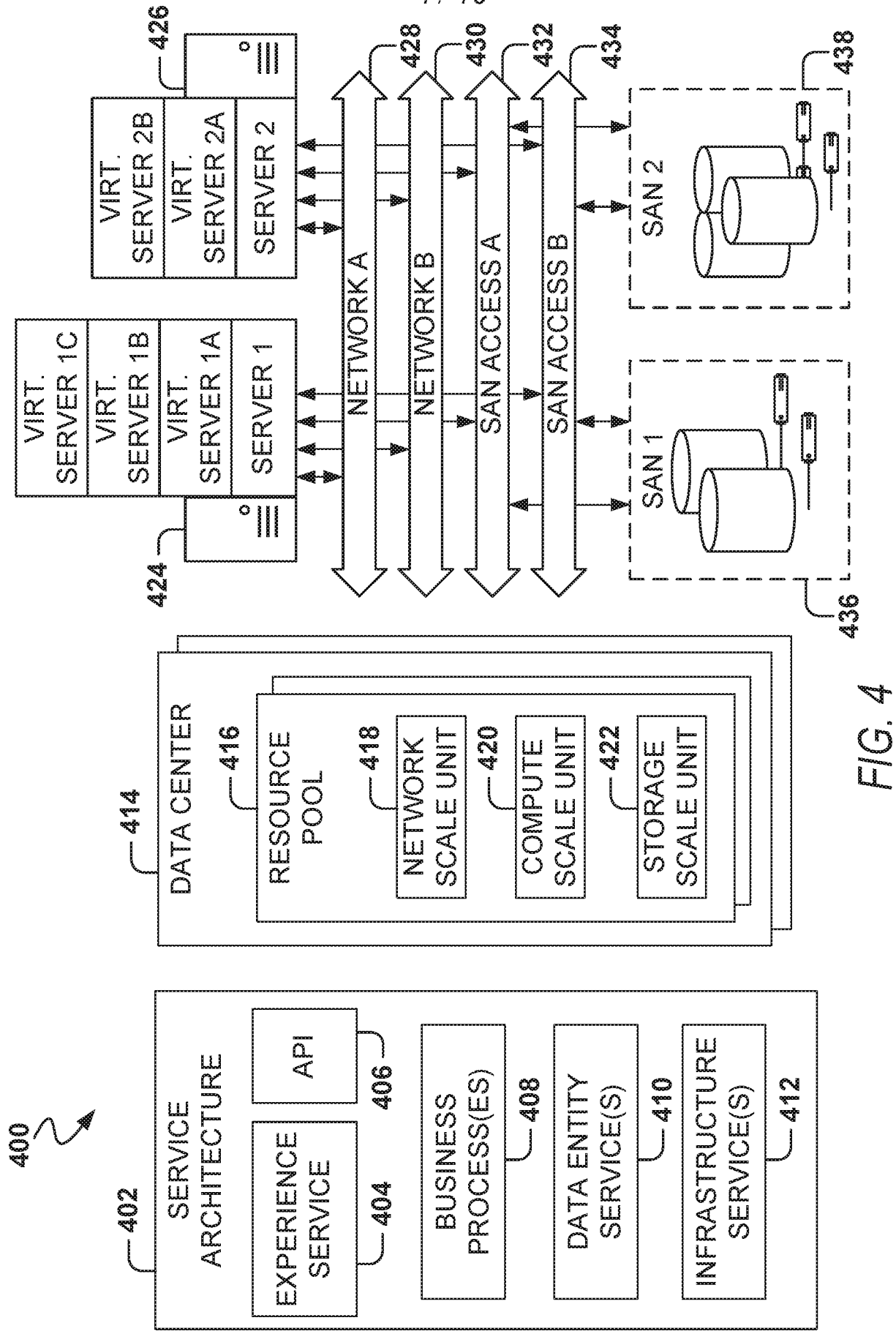

400
402
SERVICE ARCHITECTURE
EXPERIENCE SERVICE
404
API
406
BUSINESS PROCESS(ES)
408
DATA ENTITY SERVICE(S)
410
INFRASTRUCTURE SERVICE(S)
412
414
DATA CENTER
416
RESOURCE POOL
418
NETWORK SCALE UNIT
420
COMPUTE SCALE UNIT
422
STORAGE SCALE UNIT
424
VIRT. SERVER 1C
VIRT. SERVER 1B
VIRT. SERVER 1A
SERVER 1
426
VIRT. SERVER 2B
VIRT. SERVER 2A
SERVER 2
NETWORK A
428
NETWORK B
430
SAN ACCESS A
432
SAN ACCESS B
434
SAN 1
436
SAN 2
438

FIG. 4

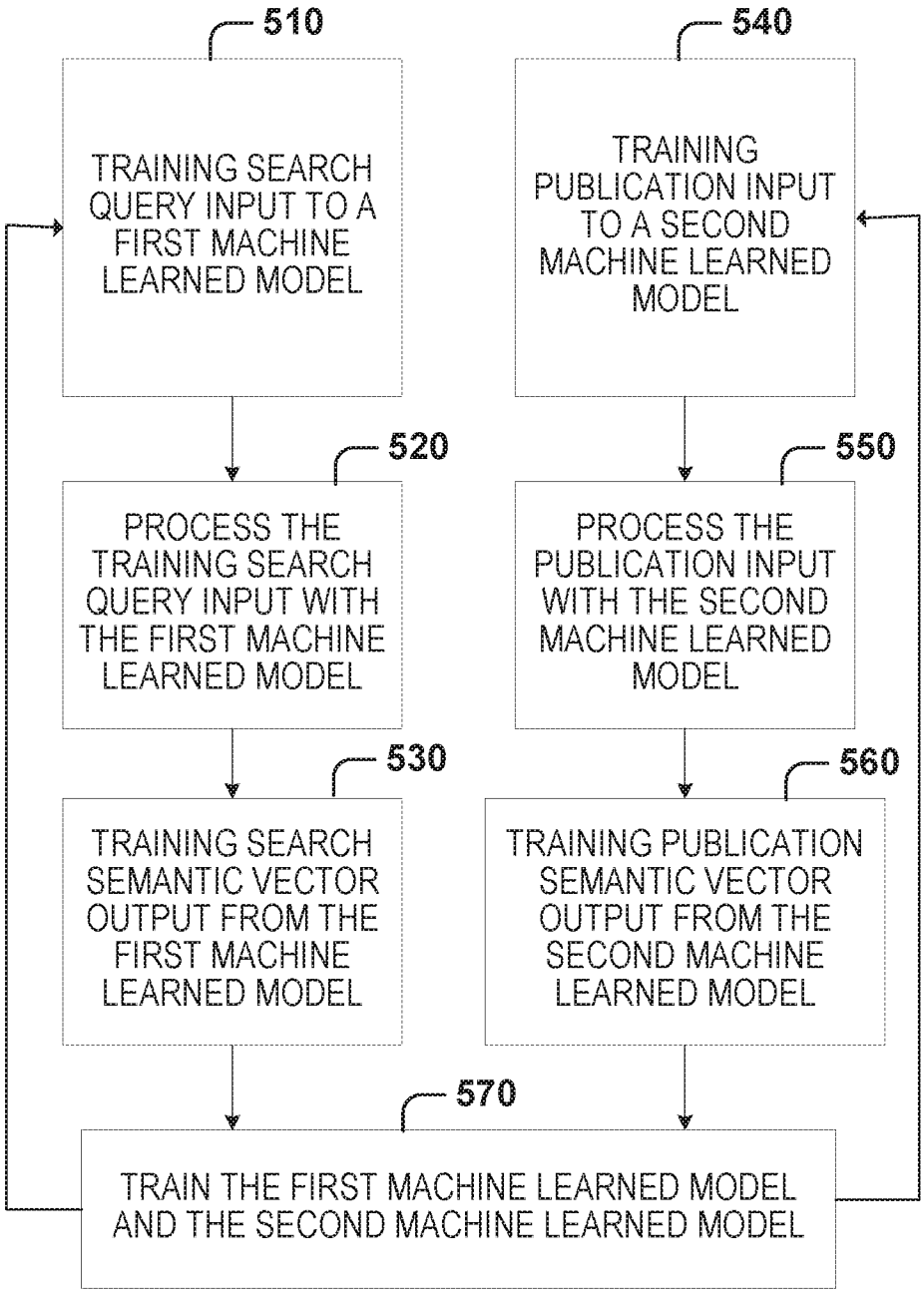

510
TRAINING SEARCH QUERY INPUT TO A FIRST MACHINE LEARNED MODEL
540
TRAINING PUBLICATION INPUT TO A SECOND MACHINE LEARNED MODEL
520
PROCESS THE TRAINING SEARCH QUERY INPUT WITH THE FIRST MACHINE LEARNED MODEL
550
PROCESS THE PUBLICATION INPUT WITH THE SECOND MACHINE LEARNED MODEL
530
TRAINING SEARCH SEMANTIC VECTOR OUTPUT FROM THE FIRST MACHINE LEARNED MODEL
560
TRAINING PUBLICATION SEMANTIC VECTOR OUTPUT FROM THE SECOND MACHINE LEARNED MODEL
570
TRAIN THE FIRST MACHINE LEARNED MODEL AND THE SECOND MACHINE LEARNED MODEL

FIG. 5

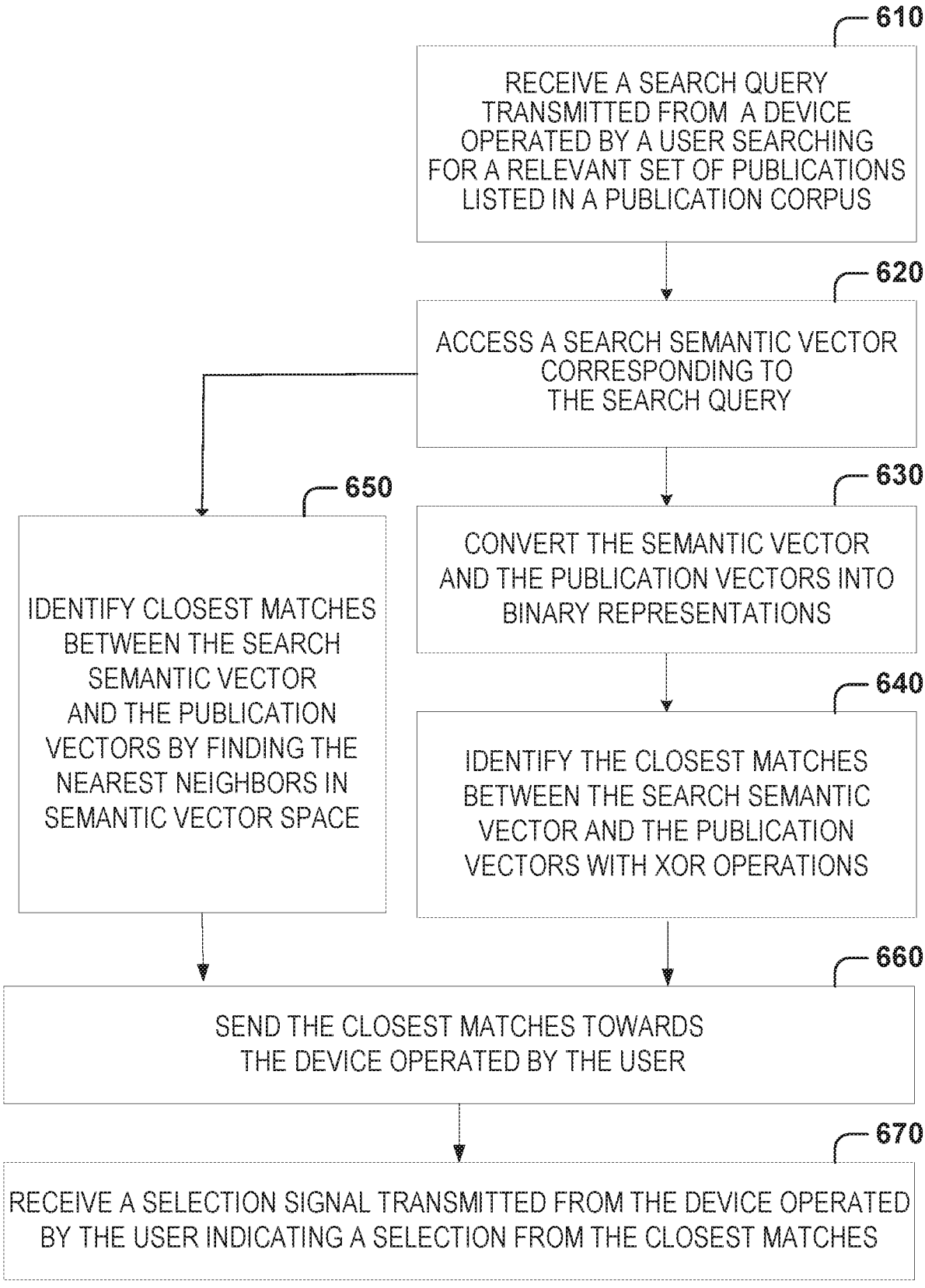

610
RECEIVE A SEARCH QUERY TRANSMITTED FROM A DEVICE OPERATED BY A USER SEARCHING FOR A RELEVANT SET OF PUBLICATIONS LISTED IN A PUBLICATION CORPUS
620
ACCESS A SEARCH SEMANTIC VECTOR CORRESPONDING TO THE SEARCH QUERY
650
IDENTIFY CLOSEST MATCHES BETWEEN THE SEARCH SEMANTIC VECTOR AND THE PUBLICATION VECTORS BY FINDING THE NEAREST NEIGHBORS IN SEMANTIC VECTOR SPACE
630
CONVERT THE SEMANTIC VECTOR AND THE PUBLICATION VECTORS INTO BINARY REPRESENTATIONS
640
IDENTIFY THE CLOSEST MATCHES BETWEEN THE SEARCH SEMANTIC VECTOR AND THE PUBLICATION VECTORS WITH XOR OPERATIONS
660
SEND THE CLOSEST MATCHES TOWARDS THE DEVICE OPERATED BY THE USER
670
RECEIVE A SELECTION SIGNAL TRANSMITTED FROM THE DEVICE OPERATED BY THE USER INDICATING A SELECTION FROM THE CLOSEST MATCHES

FIG. 6

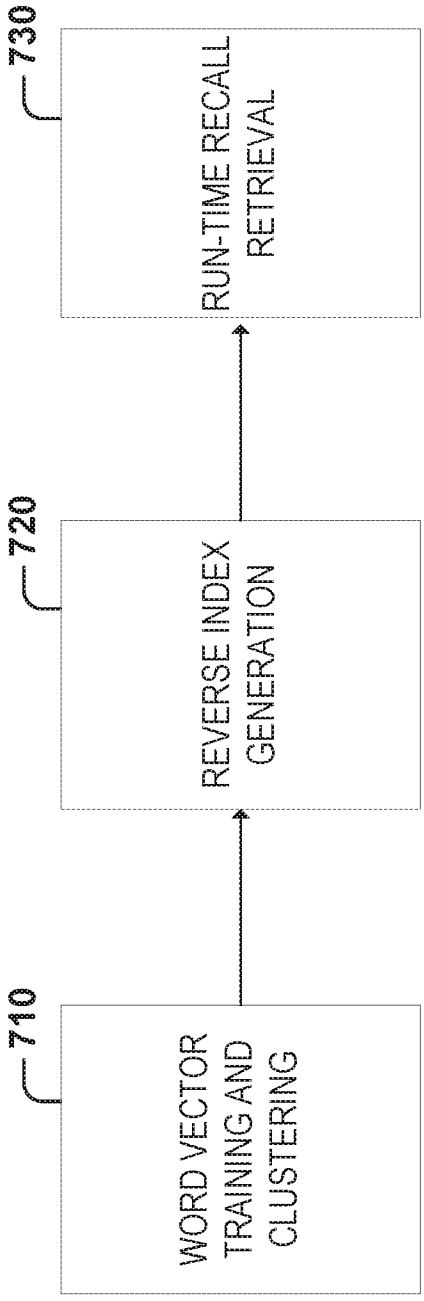

710
WORD VECTOR TRAINING AND CLUSTERING
720
REVERSE INDEX GENERATION
730
RUN-TIME RECALL RETRIEVAL

FIG. 7

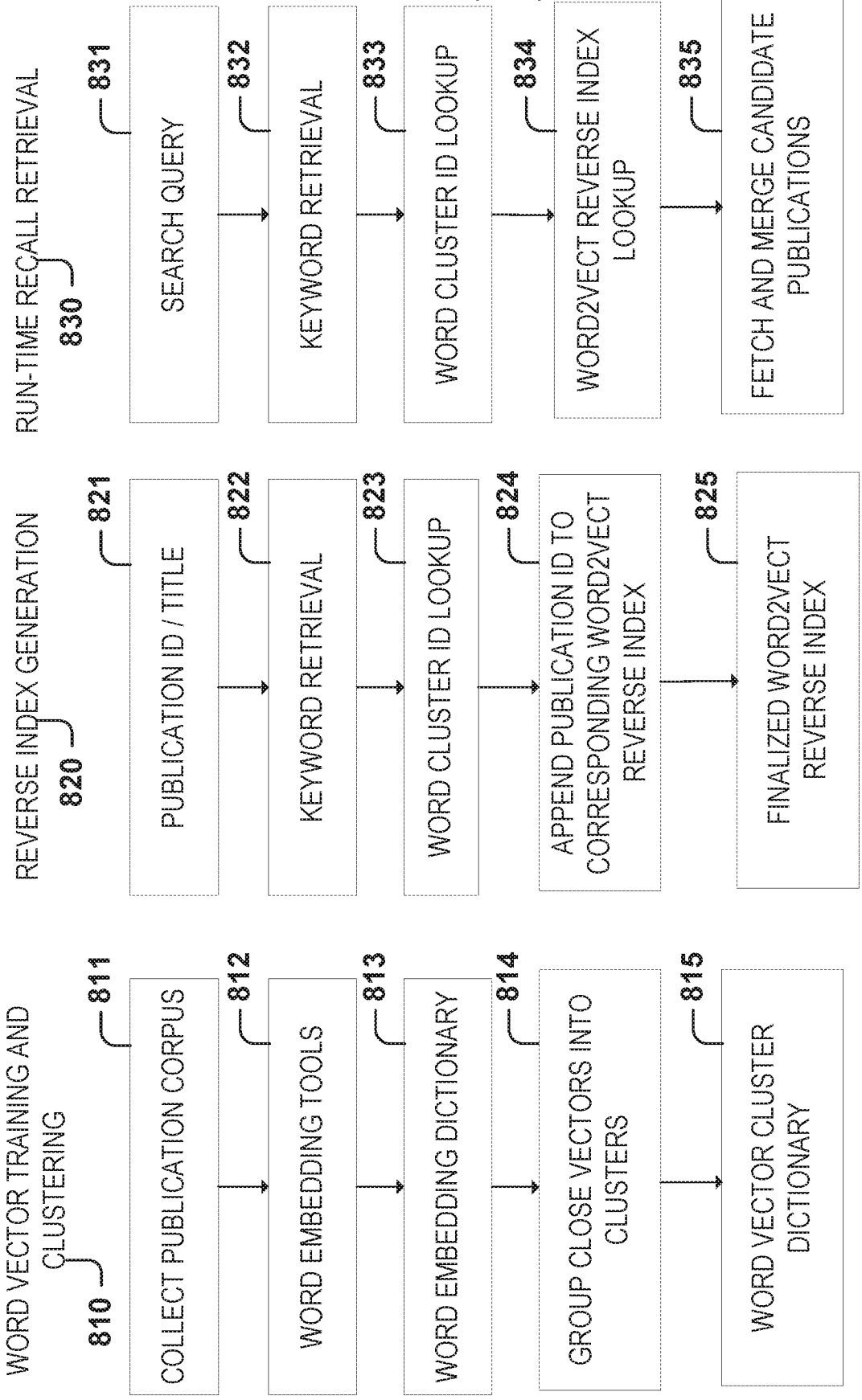

WORD VECTOR TRAINING AND CLUSTERING
810
811
COLLECT PUBLICATION CORPUS
812
WORD EMBEDDING TOOLS
813
WORD EMBEDDING DICTIONARY
814
GROUP CLOSE VECTORS INTO CLUSTERS
815
WORD VECTOR CLUSTER DICTIONARY
REVERSE INDEX GENERATION
820
821
PUBLICATION ID / TITLE
822
KEYWORD RETRIEVAL
823
WORD CLUSTER ID LOOKUP
824
APPEND PUBLICATION ID TO CORRESPONDING WORD2VECT REVERSE INDEX
825
FINALIZED WORD2VECT REVERSE INDEX
RUN-TIME RECALL RETRIEVAL
830
831
SEARCH QUERY
832
KEYWORD RETRIEVAL
833
WORD CLUSTER ID LOOKUP
834
WORD2VECT REVERSE INDEX LOOKUP
835
FETCH AND MERGE CANDIDATE PUBLICATIONS

FIG. 8

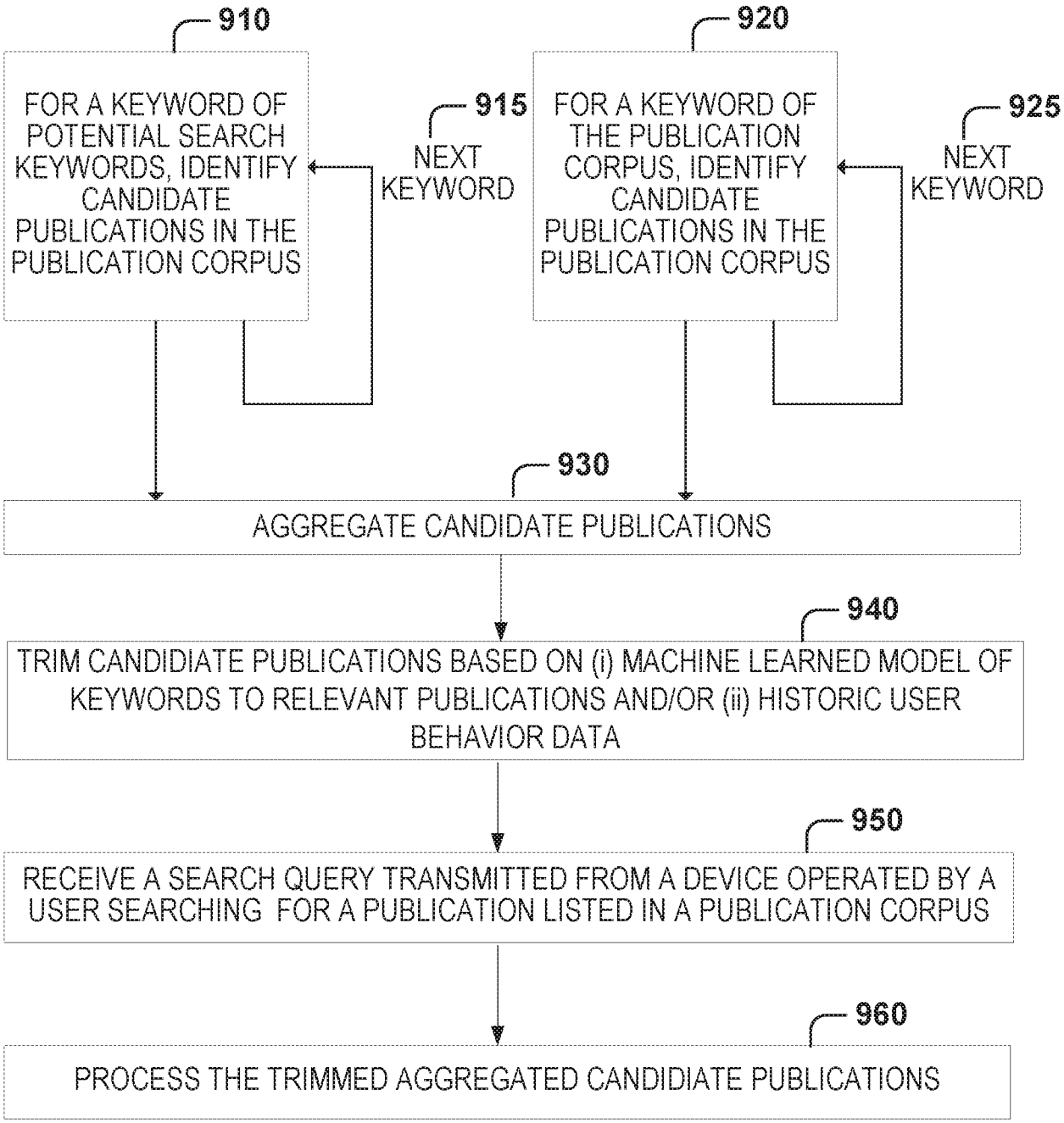

910
FOR A KEYWORD OF POTENTIAL SEARCH KEYWORDS, IDENTIFY CANDIDATE PUBLICATIONS IN THE PUBLICATION CORPUS
915
NEXT KEYWORD
920
FOR A KEYWORD OF THE PUBLICATION CORPUS, IDENTIFY CANDIDATE PUBLICATIONS IN THE PUBLICATION CORPUS
925
NEXT KEYWORD
930
AGGREGATE CANDIDATE PUBLICATIONS
940
TRIM CANDIDIATE PUBLICATIONS BASED ON (i) MACHINE LEARNED MODEL OF KEYWORDS TO RELEVANT PUBLICATIONS AND/OR (ii) HISTORIC USER BEHAVIOR DATA
950
RECEIVE A SEARCH QUERY TRANSMITTED FROM A DEVICE OPERATED BY A USER SEARCHING FOR A PUBLICATION LISTED IN A PUBLICATION CORPUS
960
PROCESS THE TRIMMED AGGREGATED CANDIDIATE PUBLICATIONS

FIG. 9

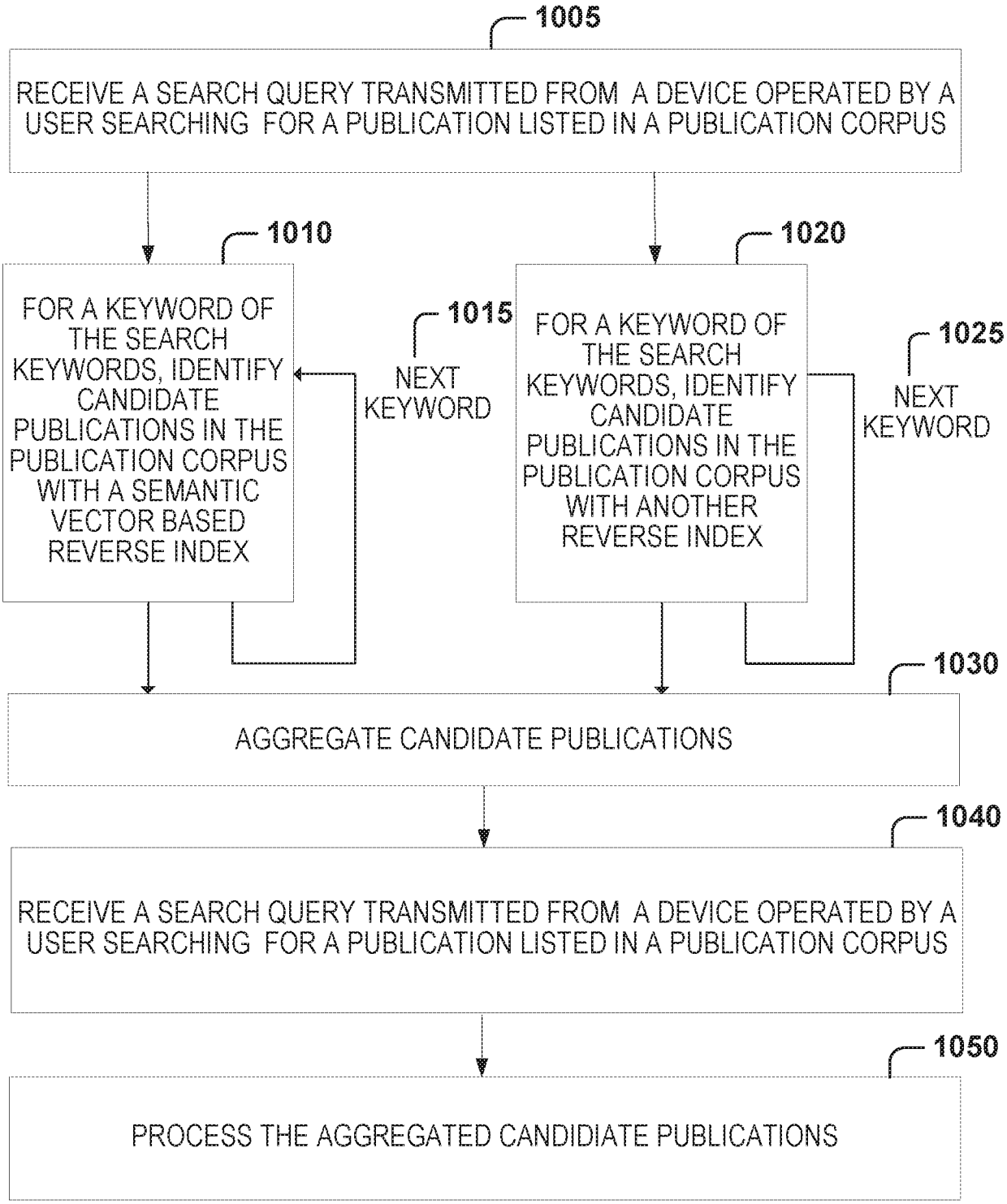

1005
RECEIVE A SEARCH QUERY TRANSMITTED FROM A DEVICE OPERATED BY A USER SEARCHING FOR A PUBLICATION LISTED IN A PUBLICATION CORPUS
1010
FOR A KEYWORD OF THE SEARCH KEYWORDS, IDENTIFY CANDIDATE PUBLICATIONS IN THE PUBLICATION CORPUS WITH A SEMANTIC VECTOR BASED REVERSE INDEX
1015
NEXT KEYWORD
1020
FOR A KEYWORD OF THE SEARCH KEYWORDS, IDENTIFY CANDIDATE PUBLICATIONS IN THE PUBLICATION CORPUS WITH ANOTHER REVERSE INDEX
1025
NEXT KEYWORD
1030
AGGREGATE CANDIDATE PUBLICATIONS
1040
RECEIVE A SEARCH QUERY TRANSMITTED FROM A DEVICE OPERATED BY A USER SEARCHING FOR A PUBLICATION LISTED IN A PUBLICATION CORPUS
1050
PROCESS THE AGGREGATED CANDIDIATE PUBLICATIONS

FIG. 10

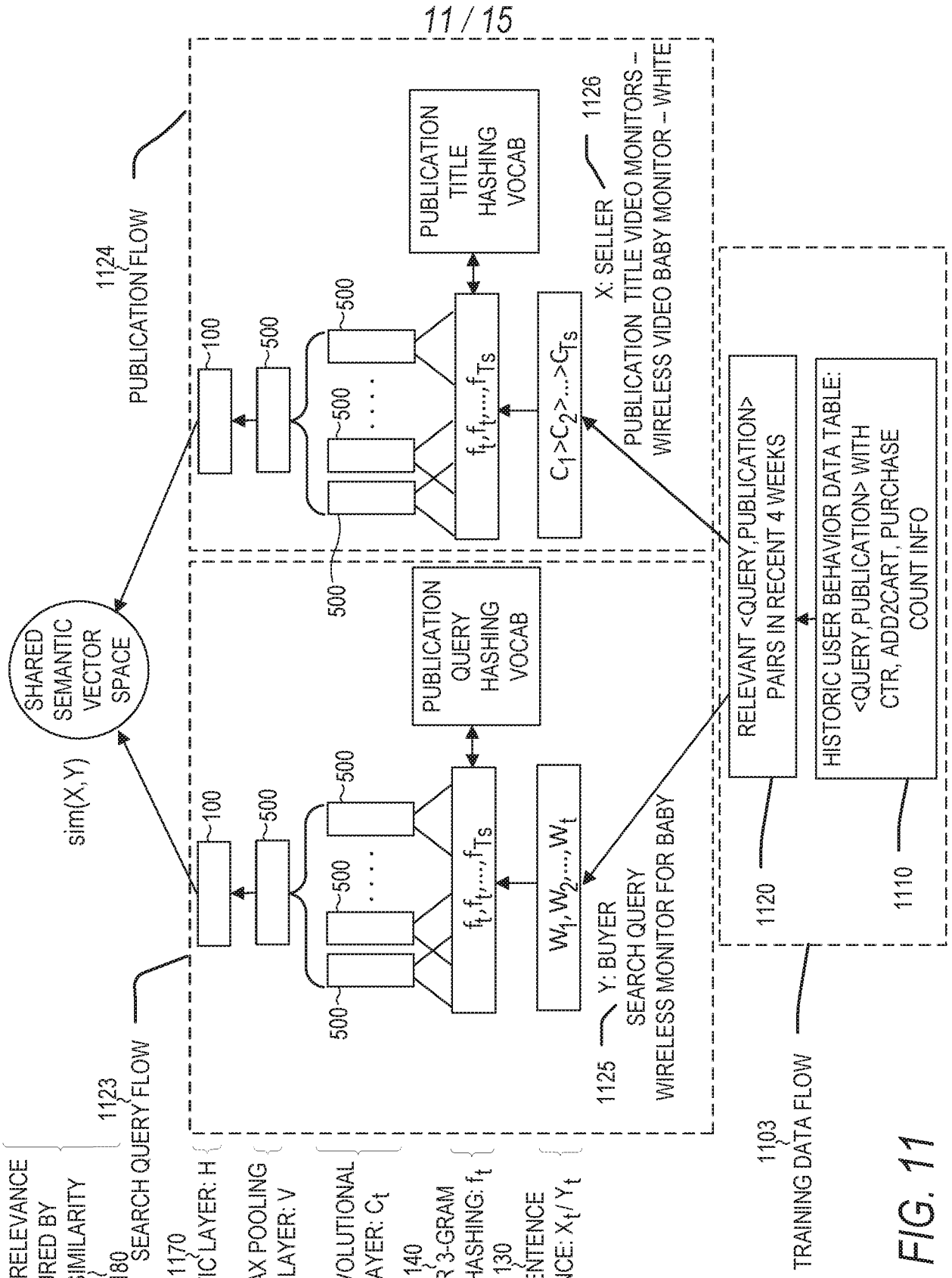

SEMANTIC RELEVANCE MEASURED BY COSINE SIMILARITY
1180
SEARCH QUERY FLOW
1123
SEMANTIC LAYER: H
1170
MAX POOLING LAYER: V
1160
CONVOLUTIONAL LAYER: C_t
1150
LETTER 3-GRAM WORD HASHING: f_t
1140
RAW SENTENCE SEQUENCE: X_t/ Y_t
1130
sim(X, Y)
SHARED SEMANTIC VECTOR SPACE
PUBLICATION FLOW
1124
100
500
PUBLICATION QUERY HASHING VOCAB
PUBLICATION TITLE HASHING VOCAB
$W_1, W_2, ..., W_t$
$f_t, f_t, ..., f_{Ts}$
$C_1 > C_2 > ... > G_{Ts}$
Y: BUYER SEARCH QUERY WIRELESS MONITOR FOR BABY
1125
X: SELLER PUBLICATION TITLE VIDEO MONITORS – WIRELESS VIDEO BABY MONITOR – WHITE
1126
RELEVANT <QUERY,PUBLICATION> PAIRS IN RECENT 4 WEEKS
1120
HISTORIC USER BEHAVIOR DATA TABLE: <QUERY,PUBLICATION> WITH CTR, ADD2CART, PURCHASE COUNT INFO
1110
TRAINING DATA FLOW
1103

FIG. 11

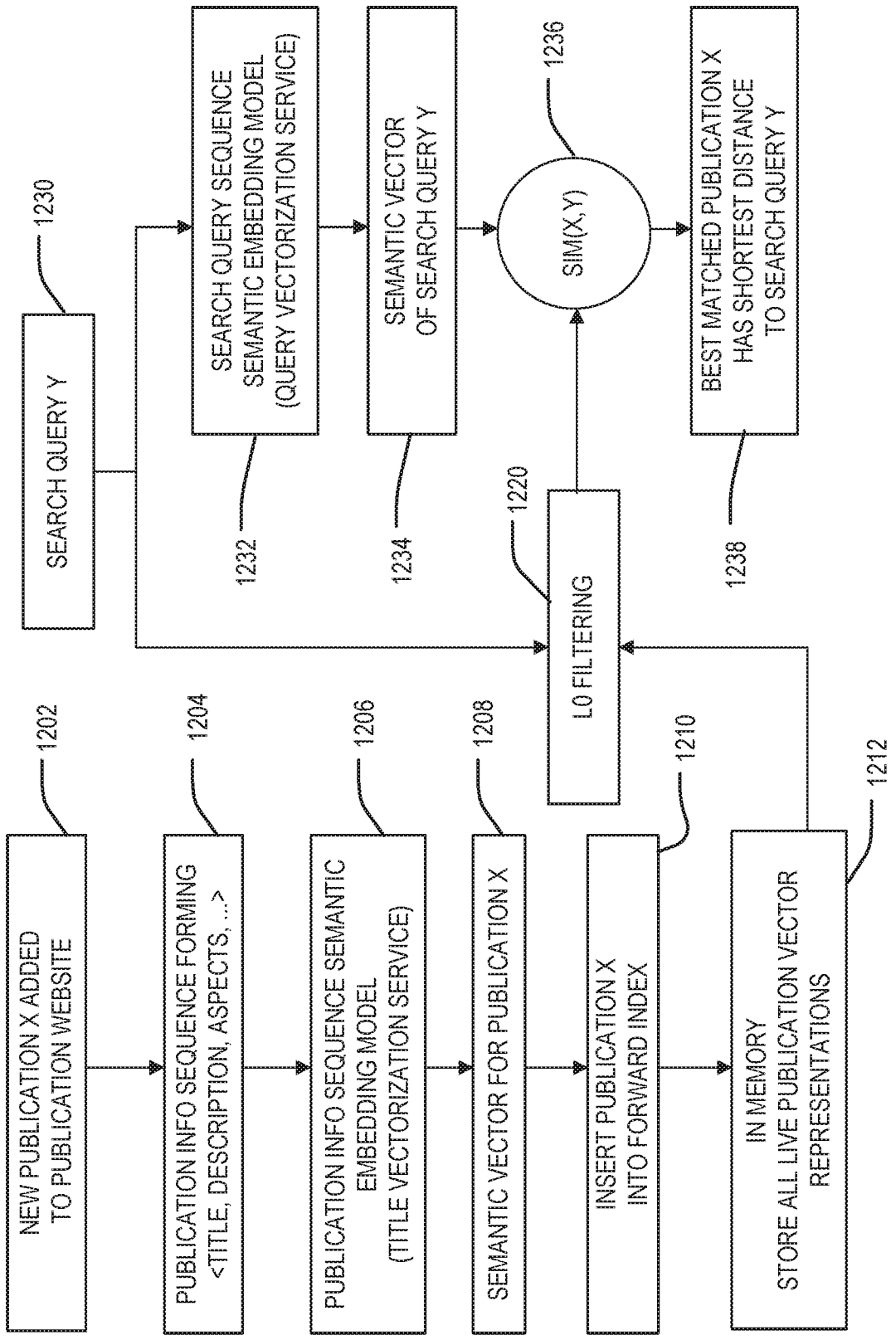

NEW PUBLICATION X ADDED TO PUBLICATION WEBSITE
1202
PUBLICATION INFO SEQUENCE FORMING <TITLE, DESCRIPTION, ASPECTS, ...>
1204
PUBLICATION INFO SEQUENCE SEMANTIC EMBEDDING MODEL (TITLE VECTORIZATION SERVICE)
1206
SEMANTIC VECTOR FOR PUBLICATION X
1208
INSERT PUBLICATION X INTO FORWARD INDEX
1210
IN MEMORY STORE ALL LIVE PUBLICATION VECTOR REPRESENTATIONS
1212
L0 FILTERING
1220
SEARCH QUERY Y
1230
SEARCH QUERY SEQUENCE SEMANTIC EMBEDDING MODEL (QUERY VECTORIZATION SERVICE)
1232
SEMANTIC VECTOR OF SEARCH QUERY Y
1234
SIM(X,Y)
1236
BEST MATCHED PUBLICATION X HAS SHORTEST DISTANCE TO SEARCH QUERY Y
1238

FIG. 12

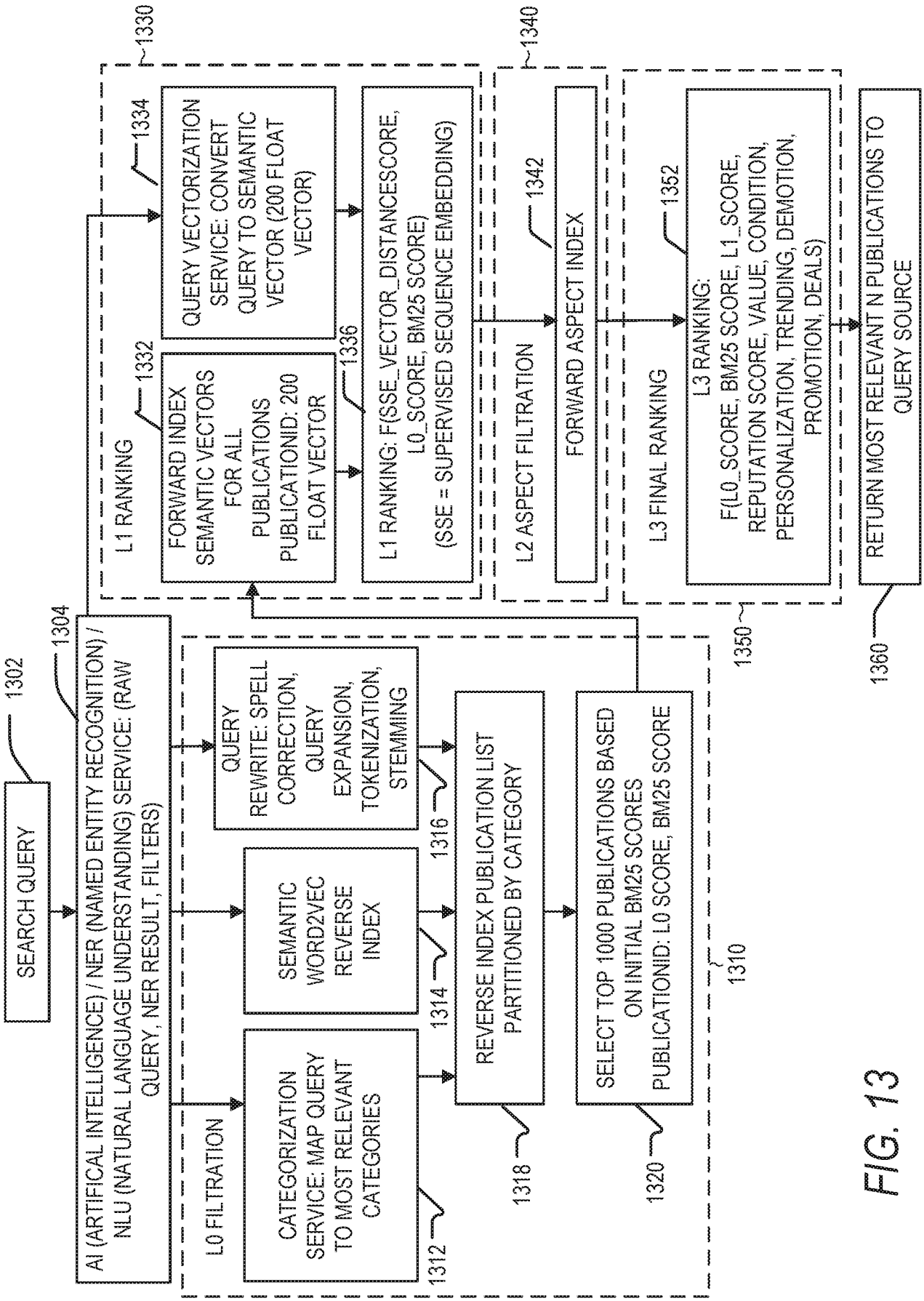

SEARCH QUERY
1302
AI (ARTIFICAL INTELLIGENCE) / NER (NAMED ENTITY RECOGNITION) / NLU (NATURAL LANGUAGE UNDERSTANDING) SERVICE: (RAW QUERY, NER RESULT, FILTERS)
1304
L0 FILTRATION
QUERY REWRITE: SPELL CORRECTION, QUERY EXPANSION, TOKENIZATION, STEMMING
SEMANTIC WORD2VEC REVERSE INDEX
CATEGORIZATION SERVICE: MAP QUERY TO MOST RELEVANT CATEGORIES
1316
1314
1312
REVERSE INDEX PUBLICATION LIST PARTITIONED BY CATEGORY
1318
SELECT TOP 1000 PUBLICATIONS BASED ON INITIAL BM25 SCORES
PUBLICATIONID: L0 SCORE, BM25 SCORE
1320
1310
1350
L1 RANKING
1332
FORWARD INDEX SEMANTIC VECTORS FOR ALL PUBLICATIONS
PUBLICATIONID: 200 FLOAT VECTOR
1336
QUERY VECTORIZATION SERVICE: CONVERT QUERY TO SEMANTIC VECTOR (200 FLOAT VECTOR)
1334
1330
L1 RANKING: F(SSE_VECTOR_DISTANCESCORE, L0_SCORE, BM25 SCORE)
(SSE = SUPERVISED SEQUENCE EMBEDDING)
L2 ASPECT FILTRATION
1342
1340
FORWARD ASPECT INDEX
L3 FINAL RANKING
1352
L3 RANKING:
F(L0_SCORE, BM25 SCORE, L1_SCORE, REPUTATION SCORE, VALUE, CONDITION, PERSONALIZATION, TRENDING, DEMOTION, PROMOTION, DEALS)
RETURN MOST RELEVANT N PUBLICATIONS TO QUERY SOURCE
1360

FIG. 13

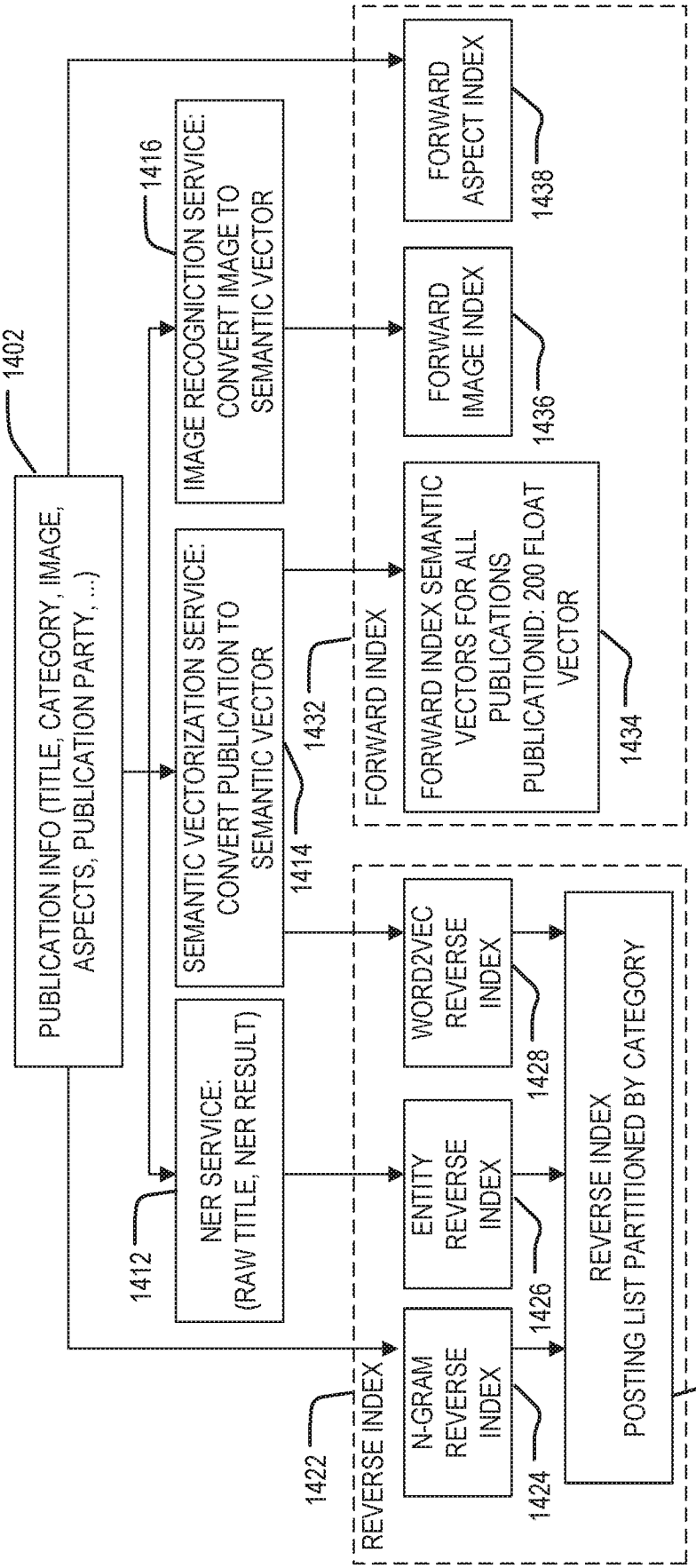

1402
PUBLICATION INFO (TITLE, CATEGORY, IMAGE, ASPECTS, PUBLICATION PARTY, ...)
1412
NER SERVICE: (RAW TITLE, NER RESULT)
1414
SEMANTIC VECTORIZATION SERVICE: CONVERT PUBLICATION TO SEMANTIC VECTOR
1416
IMAGE RECOGNICTION SERVICE: CONVERT IMAGE TO SEMANTIC VECTOR
1422
REVERSE INDEX
N-GRAM REVERSE INDEX
1424
ENTITY REVERSE INDEX
1426
WORD2VEC REVERSE INDEX
1428
REVERSE INDEX POSTING LIST PARTITIONED BY CATEGORY
1429
1432
FORWARD INDEX
FORWARD INDEX SEMANTIC VECTORS FOR ALL PUBLICATIONS
PUBLICATIONID: 200 FLOAT VECTOR
1434
FORWARD IMAGE INDEX
1436
FORWARD ASPECT INDEX
1438

FIG. 14

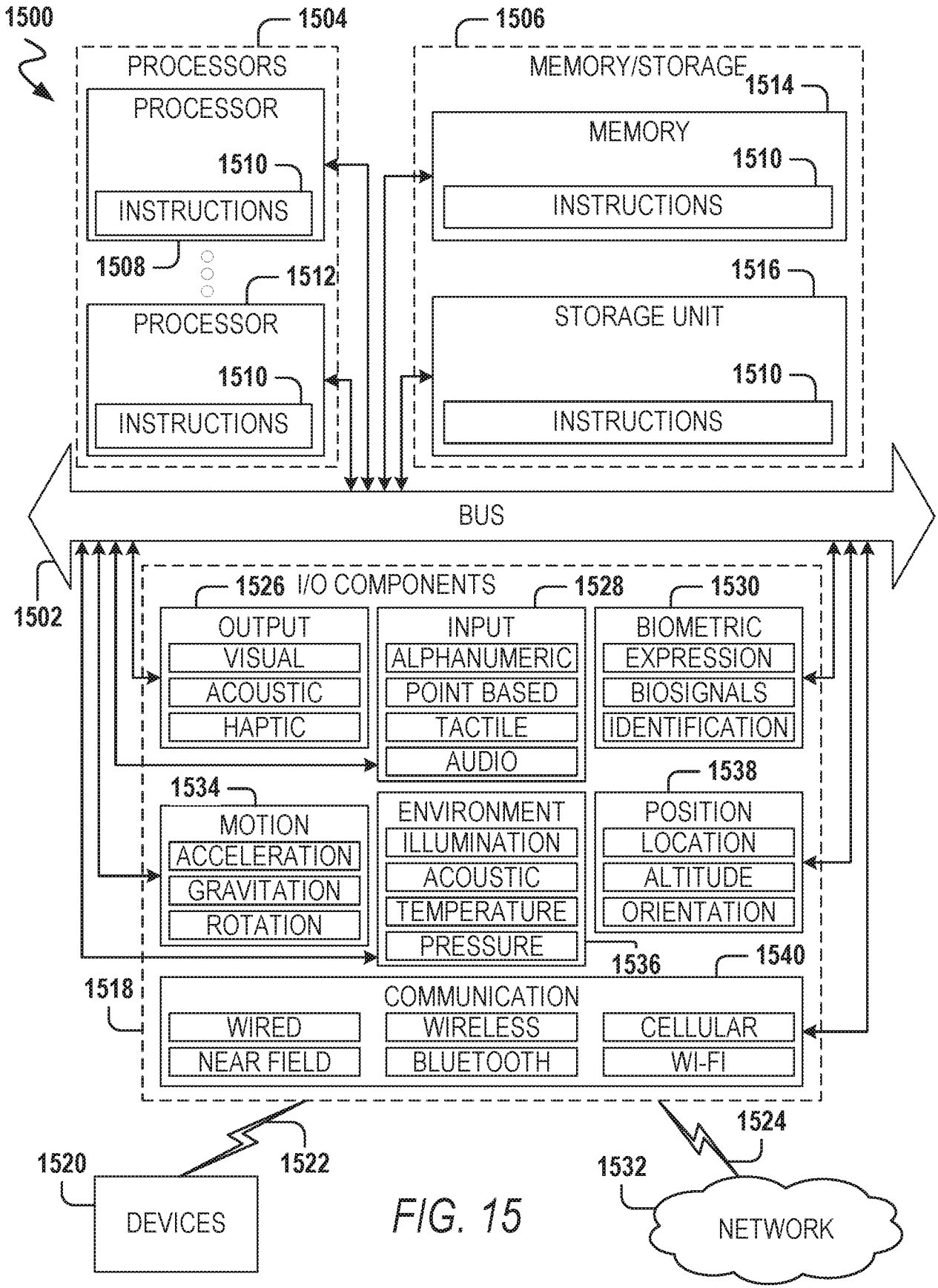

1500
1504
PROCESSORS
PROCESSOR
1510
INSTRUCTIONS
1508
1512
PROCESSOR
1510
INSTRUCTIONS
1506
MEMORY/STORAGE
1514
MEMORY
1510
INSTRUCTIONS
1516
STORAGE UNIT
1510
INSTRUCTIONS
BUS
1502
1526 I/O COMPONENTS
1528
1530
OUTPUT
VISUAL
ACOUSTIC
HAPTIC
INPUT
ALPHANUMERIC
POINT BASED
TACTILE
AUDIO
BIOMETRIC
EXPRESSION
BIOSIGNALS
IDENTIFICATION
1534
MOTION
ACCELERATION
GRAVITATION
ROTATION
ENVIRONMENT
ILLUMINATION
ACOUSTIC
TEMPERATURE
PRESSURE
1538
POSITION
LOCATION
ALTITUDE
ORIENTATION
1536
1540
1518
COMMUNICATION
WIRED
NEAR FIELD
WIRELESS
BLUETOOTH
CELLULAR
WI-FI
1524
1520
1522
1532
DEVICES
NETWORK

FIG. 15

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US 17/46922

A. CLASSIFICATION OF SUBJECT MATTER
IPC(8) - G06F 17/30 (2017.01)
CPC - G06F 17/30864, G06F 17/30321, G06F 17/30613, G06F 17/30961, G06F 17/30067, G06F 17/2247, G06F 17/2241, G06F 17/2735, G06F 17/277, G06F 17/30067, G06F 17/2785, G06F 17/30864, G06F 17/30038, G06F 17/30616, G06F 17/3069

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)
See Search History Document

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
See Search History Document

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
See Search History Document

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2014/0081941 A1 (Bai et al.) 20 March 2014 (20.03.2014), entire document especially paras [0028], [0031], [0034], [0040], [0046], [0053]	1-21
A	US 2009/0204605 A1 (Bai et al.) 13 August 2009 (13.08.2009), entire document	1-21
A	US 2005/0267871 A1 (Marchisio et al.) 01 December 2005 (01.12.2005), entire document	1-21
A	US 2009/0063426 A1 (Crouch et al.) 05 March 2009 (05.03.2009), entire document	1-21
A	US 2007/0088734 A1 (Krishnamurthy et al.) 19 April 2007 (19.04.2007), entire document	1-21

☐ 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	Date of mailing of the international search report
16 October 2017 (16.10.2017)	02 NOV 2017
Name and mailing address of the ISA/US Mail Stop PCT, Attn: ISA/US, Commissioner for Patents P.O. Box 1450, Alexandria, Virginia 22313-1450 Facsimile No. 571-273-8300	Authorized officer: Lee W. Young PCT Helpdesk: 571-272-4300 PCT OSP: 571-272-7774

Form PCT/ISA/210 (second sheet) (January 2015)