



(51) International Patent Classification:

G06F 16/93 (2019.01)

G06F 16/338 (2019.01)

G06F 16/332 (2019.01)

(21) International Application Number:

PCT/GB2023/051519

(22) International Filing Date:

12 June 2023 (12.06.2023)

(25) Filing Language:

English

(26) Publication Language:

English

(30) Priority Data:

63/351,534

13 June 2022 (13.06.2022)

US

(71) Applicant: EIGEN TECHNOLOGIES LTD. [GB/GB];

60 Queen Victoria Street, London Greater London EC4N 4TR (GB).

(72) Inventors: BENINCASA, Gregorio Benedetto; Eigen Technologies Ltd., 60 Queen Victoria Street, London Greater London EC4N 4TR (GB). KONSTANTINOV, Momchil Preslavov; Eigen Technologies Ltd., 60 Queen Victoria Street, London Greater London EC4N 4TR (GB). KARLIN, Josh; Eigen Technologies Ltd., 60 Queen Victoria Street, London Greater London EC4N 4TR (GB).

(74) Agent: CLARK, Jane; The Shard, 32 London Bridge Street, London Greater London SE1 9SG (GB).

(81) Designated States (unless otherwise indicated, for every kind of national protection available): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CV, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IQ, IR, IS, IT, JM, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, MG, MK, MN, MU, MW, MX, MY, MZ, NA,

(54) Title: METHODS AND SYSTEMS FOR TRANSFORMING AND RETRIEVING INFORMATION FROM DOCUMENT DATA USING MACHINE LEARNING

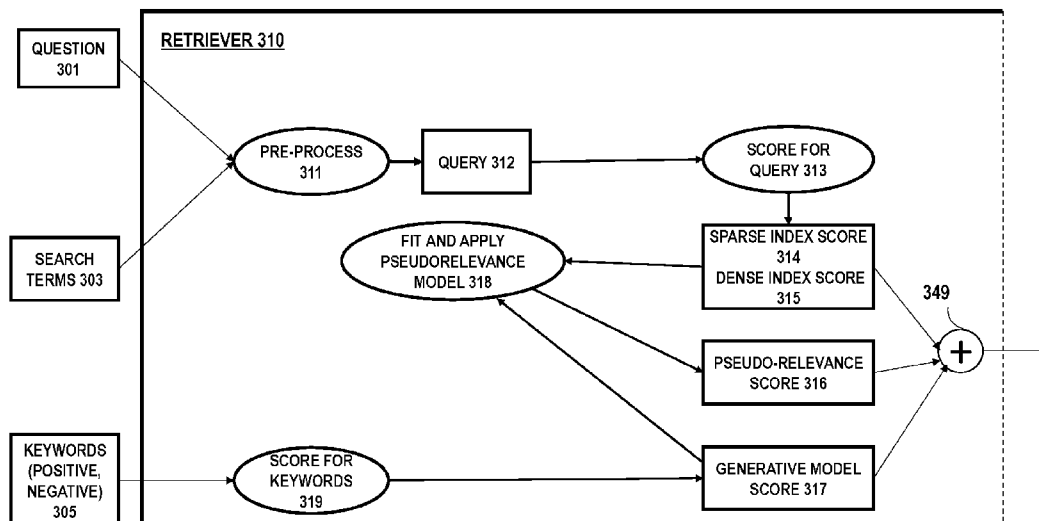


FIG. 3

(57) Abstract: The present disclosure provides methods and systems for querying insight information. The method comprises: providing different input fields on a UI for receiving a query input for searching insight information in a document, the input fields comprise a first input field for receiving one or more questions in natural language and a second input field for receiving one or more search terms; receiving the query comprising a question or a search term; identifying, using one or more machine learning algorithm trained models, one or more sections of the document each comprising a chunk of texts relevant to the query input, and rendering the one or more sections in the UI with the chunk of texts annotated with a visual indicator; receiving a user feedback data indicating an acceptance or rejection of the chunk of texts; and improving at least one of the trained models using the user feedback data.

NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO,
RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH,
TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS,
ZA, ZM, ZW.

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

Published:

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

METHODS AND SYSTEMS FOR TRANSFORMING AND RETRIEVING INFORMATION FROM DOCUMENT DATA USING MACHINE LEARNING

CROSS-REFERENCE TO RELATED APPLICATION

[0001] This application claims priority to U.S. Provisional Application No. 63/351,534 filed on June 13, 2022, the content of which is incorporated herein in its entirety.

BACKGROUND

[0002] There has been methods and platforms for document processing, information retrieval and data management. For instance, data extraction or information extraction methods may be utilized to extract structured data from unstructured or semi- structured electronic document text, then a user may be allowed to search the data in the document such as based on keyword match. However, current document data extraction and retrieval methods may require manual or handcrafted rules in the workflows without automation. Additionally, existing methods may lack the capability of insight querying with sufficient accuracy.

SUMMARY

[0003] Recognized herein is a need for transforming document data into insights and useful information and automating the compilation of information from documents with improved degrees of accuracy. The present disclosure provides systems and methods that can address the above needs and various drawbacks of the conventional systems or methods, including those recognized above. Systems and methods of the present disclosure may automate the workflows for transforming data trapped in documents into insights thereby allowing users to perform insight querying with improved accuracy. In particular, the present disclosure provides an interactive interface for users to search through and ask questions of the documents, and a backend system for extracting information (knowledge or insights) from the document data and retrieving the relevant information pertinent to the user's query with improved accuracy.

[0004] Systems and methods herein may allow users to search through one or more documents by querying insights or ask intelligent questions. In some cases, the provided systems and methods may utilize deep learning models to understand a user's intent (e.g., desired information) without being intrusive and/or transform the document data into insights and retrieving the relevant insights taking into account the user's knowledge. The deep learning models herein may continuously learn and improve based on user feedback data or user inputs collected during the insights retrieval process thereby improving the accuracy of the delivered information subject to the user's needs and computational budget. Additionally, the system herein may allow a user to specify the desired or interested information with improved flexibility.

For instance, a user may be allowed to provide an input query in various forms such as search terms, an intelligent question, positive or negative keywords or any combination of the above in order to specify the desired information.

[0005] In an aspect of the present disclosure, a method for querying insight information from a document is provided. The method comprises: providing, via a user interface, a plurality of different input fields for receiving a query input for searching insight information in a document, where the plurality of input fields comprise a first input field for receiving one or more questions in natural language and a second input field for receiving one or more search terms; receiving the query input submitted via the user interface, where the query input comprises a question or a search term; identifying, using one or more trained models executed at the backend computing system, one or more sections of the document each comprising a chunk of texts relevant to the query input, and rendering the one or more sections within the user interface with the chunk of texts annotated with a visual indicator; receiving a user feedback data via the user interface indicating an acceptance or rejection of the chunk of texts; and improving at least one of the trained models using the user feedback data. The one or more models are trained using machine learning algorithms.

[0006] Another aspect of the present disclosure provides a non-transitory computer readable medium comprising machine executable code that, upon execution by one or more computer processors, implements any of the methods above or elsewhere herein.

[0007] In some embodiments, the method further comprises. generating, at the backend computing system, a query based at least in part on the question. In some cases, the chunk of texts comprise an answer to the question. In some instances, the answer is annotated with a first visual indicator and the remaining texts in the chunk of texts are annotated with a second visual indicator. In some instances, the method further comprises generating a unified query by tokenizing at least the question and the unified query is used to identify one or more relevant sections. For example, the answer is identified by processing the one or more relevant sections using transformer model, and the transformer model is fine-tuned using a small amount of training datasets.

[0008] In some embodiments, the method further comprises generating a unified query by tokenizing the question or the search term. In some cases, the method further comprises computing a sparse index score based at least in part on a similarity between the tokenized query and the one or more sections of the document retrieved by a sparse index. In some instances, the method further comprises computing a dense index score based at least in part on a similarity between a representation of the query and a representation of each section of the document

retrieved by a dense index. In some instances, a pseudo-relevance score is generated based at least in part on the sparse index score and the dense index score and the method may further comprise generating a combined score by aggregating the sparse index score, the dense index score and the pseudo-relevance score using one of the one or more trained models. In some examples, the method further comprises retraining the one of the one or more trained models when the user feedback data comprises a number of relevant sections exceeding a predetermined threshold.

[0009] In some embodiments, the plurality of different input fields further comprises a third input field for receiving a positive keyword and a fourth input field for receiving a negative keyword. In some cases, the query input further comprises a positive keyword or a negative keyword. In some instances, the positive keyword or the negative keyword is utilized to determine a relevance score for at least one of the one or more sections. In some cases, the user interface comprises a first panel displaying a set of sections ranked based at least in part on the combined score, and a second panel rendering at least one of the set of sections with the chunk of texts annotated with a visual indicator, and wherein the second panel displays one or more graphical elements for receiving the user feedback data.

[0010] In some embodiments, the user interface is a web-based user interface. In some embodiments, the document comprises unstructured data. In some embodiments, the document is processed by the backend computing system for building a document dataset and indexes. In some embodiments, the method further comprises processing the one or more sections of the document and the query input by a large language model to output a generated answer.

[0011] Another aspect of the present disclosure provides a system comprising one or more computer processors and computer memory coupled thereto. The computer memory comprises machine executable code that, upon execution by the one or more computer processors, implements any of the methods above or elsewhere herein.

[0012] Additional aspects and advantages of the present disclosure will become readily apparent to those skilled in this art from the following detailed description, wherein only illustrative embodiments of the present disclosure are shown and described. As will be realized, the present disclosure is capable of other and different embodiments, and its several details are capable of modifications in various obvious respects, all without departing from the disclosure. Accordingly, the drawings and description are to be regarded as illustrative in nature, and not as restrictive.

INCORPORATION BY REFERENCE

[0013] All publications, patents, and patent applications mentioned in this specification are herein incorporated by reference to the same extent as if each individual publication, patent, or patent application was specifically and individually indicated to be incorporated by reference. To the extent publications and patents or patent applications incorporated by reference contradict the disclosure contained in the specification, the specification is intended to supersede and/or take precedence over any such contradictory material.

BRIEF DESCRIPTION OF THE DRAWINGS

[0014] The novel features of the invention are set forth with particularity in the appended claims. A better understanding of the features and advantages of the present invention will be obtained by reference to the following detailed description that sets forth illustrative embodiments, in which the principles of the invention are utilized, and the accompanying drawings (also “Figure” and “FIG.” herein), of which:

[0015] FIG. 1 schematically shows a platform in which the method and system herein can be implemented.

[0016] FIG. 2 schematically illustrates a system for document analysis and information retrieval, in accordance with some embodiments of the present disclosure.

[0017] FIG. 3 and **FIG. 4** shows various components and operations of an information extraction and retrieval system, in accordance with some embodiments of the present disclosure.

[0018] FIGs. 5-10 show examples of user interface (UI) provided by a frontend component of the system for insight query, in accordance with some embodiments of the present disclosure.

DETAILED DESCRIPTION

[0019] While various embodiments of the invention have been shown and described herein, it will be obvious to those skilled in the art that such embodiments are provided by way of example only. Numerous variations, changes, and substitutions may occur to those skilled in the art without departing from the invention. It should be understood that various alternatives to the embodiments of the invention described herein may be employed.

[0020] The present disclosure provides methods, systems and platforms for document processing, information and insight extraction and retrieval. Systems and methods herein may beneficially augment a user’s analysis and understanding of document content by incorporating a variety of machine learning (ML) techniques such as machine learning (ML) for augmented intelligence, and other techniques such as heuristics injection and knowledge collection. The platform may advantageously maximize and/or optimize the use of a user’s knowledge while minimize the computational budget by improving the interaction between Human and Machine, incorporating Human Computer Interaction techniques, machine learning techniques (e.g.,

supervised, unsupervised, semi-supervised, trial design), knowledge base construction techniques and the like.

[0021] In some embodiments, methods and systems herein may allow for fast semantic search by computing one or more similarity scores generated with lightweight models. The systems and platform herein may be capable of efficiently and effectively retrieving and extracting information from a universe of documents (e.g., raw documents). In some cases, the raw documents may comprise unstructured or semi-structured electronic document text. Unstructured text documents may contain “free text” in which the underlying information is mainly captured in the words themselves. The unstructured document texts may include, for example, open text, images, that have no predetermined organization or design. Semi-structured text may capture a significant portion of the information in the position, layout and format of the text but the information within has no structure. The systems herein may be capable of extracting information and retrieving insights from the raw documents by converting the raw document texts into structured data (e.g., document datasets, indexes) then retrieving insight needed or desired by a user with machine learning techniques.

[0022] Methods and systems herein may improve the performance of insight retrieval by dynamically constructing the machine learning algorithm trained models using user feedback data. For instance, the system and platform herein may train models to retrieve and extract information from a set of documents and a set of unseen documents by dynamically constructing models for retrieval and extraction, employ models benchmarks and model competition during training, improve the model training process and inform users when a model is properly trained, and/or use weak supervision for training large size models. Additionally, the models provided by the systems herein may adapt to a user’s need by continuously learning from user’s feedback or user’s interaction with the system during the insight retrieval process. For instance, the user feedback data collected by the system may comprise clickthrough data (e.g., how quickly a user responds to a system suggested answer, how many passages/answers identified by the system as relevant are confirmed (e.g., clicked on) or ignored by the user, etc.), or comprise user input indicative of the relevant information to the user in a given document (e.g., system identified relevant information which the user may be interested in and the user may accept or reject the system identified relevant information or answer). The user feedback collected by the system can comprise various other data such as whether the user used a system suggested search term and/or question for inputting a query.

[0023] The provided systems and methods may collect extracted knowledge or information for further improving insight querying. The extracted knowledge or information may be managed

and maintained in knowledge bases (e.g., object model with classes, sub-classes, instances or other structures for storing structured and unstructured information). The systems and methods may employ various other suitable document processing techniques such as summarization, document diffing, coreference resolution and relation extraction, template filing, normalization of extracted fields and the like. The provided methods and systems can be implemented in various scenarios such as in cloud or an on-premises environment.

[0024] Various aspects of the systems and methods described herein may be applied to any applications where document analysis and intelligent information retrieval is involved. For instance, in industries such as banking, financial services, legal services, corporates, insurance, technology and various others, the amount of lengthy articles, regulatory documents, emails, and news articles being published are growing thus increasing the need to efficiently consume all the textual information a user needs to understand. The systems and methods provided herein may computationally extract and retrieve knowledge and information from complex and unstructured text to meet a user's intent with improved accuracy. Insights or knowledge may be extracted and retrieved for various purposes such as for analyzing legal contract or service agreement, regulatory reporting and compliance, and can be utilized in a wide range of industries as described above. It shall be understood that different aspects of the invention can be appreciated individually, collectively or in combination with each other.

[0025] As utilized herein, terms "component," "system," "interface," "unit" and the like are intended to refer to a computer-related entity, hardware, software (e.g., in execution), algorithm, and/or firmware. For example, a component can be a processor, a process running on a processor, an object, an executable, a program, a storage device, and/or a computer. By way of illustration, an application running on a server and the server can be a component. One or more components can reside within a process, and a component can be localized on one computer and/or distributed between two or more computers.

[0026] Further, these components can execute from various computer readable media having various data structures stored thereon. The components can communicate via local and/or remote processes such as in accordance with a signal having one or more data packets (e.g., data from one component interacting with another component in a local system, distributed system, and/or across a network, e.g., the Internet, a local area network, a wide area network, etc. with other systems via the signal).

[0027] As another example, a component can be an apparatus with specific functionality provided by mechanical parts operated by electric or electronic circuitry; the electric or electronic circuitry can be operated by a software application or a firmware application executed by one or

more processors; the one or more processors can be internal or external to the apparatus and can execute at least a part of the software or firmware application. As yet another example, a component can be an apparatus that provides specific functionality through electronic components without mechanical parts; the electronic components can include one or more processors therein to execute software and/or firmware that confer(s), at least in part, the functionality of the electronic components. In some cases, a component can emulate an electronic component via a virtual machine, e.g., within a cloud computing system.

[0028] In some embodiments, methods and systems herein may provide various functions that can be implemented or accessed via web application program interfaces (APIs), a Software Development Kit (SDK), web-based integrated development environment (IDE) and the like. Various components of the system herein may be seamlessly integrated into a third-party platform or system via customized Software Development Kit (SDK) or APIs. For instance, intelligent information extraction and retrieval as well as document processing modules may be provided via open-ended integration with a full suite of APIs and plugins thereby allowing for convenient and seamless system integrations into any third-party systems.

[0029] In some embodiments, the platform may be a no-code user-friendly platform that requires only a small amount of training datasets to deliver highly accurate results across a wide range of document types and data formats. The platform herein may train models to be able to extract relevant information relating to a user input from documents. The data extraction models may be trained on a relatively small number of documents, and still provide accurate outcomes when used to analyze documents. For example, the platform may provide customized models using limited training datasets (e.g., 2-50 examples) to fine tune the models to extract and retrieve information from any document, for any data, any user and any use case. This greatly reduces the time and effort required by a user before the system can commence useful data extraction. The present disclosure also provides a user-friendly user interface (UI) which allows user to specify the information needed in natural language-based question, using search terms, positive or negative keywords of any combination of the above with improved flexibility. The UI may also allow the user to interact with the extracted information and collect user feedback data related to the relevancy of the extracted information which user feedback data may be utilized by the system to further improve the information retrieval models in an automated fashion.

[0030] In some embodiments, the system herein may provide improved flexibility for a user to provide input via a user interface to specify the desired or interested information or to retrieve information from one or more documents. The system herein may allow a user to provide an insight query input in various formats or types. For instance, a user may be allowed to provide

search terms, an intelligent question, positive or negative keywords or any combination of the above in order to specify the desired information. The system may process the user input in the one or more types or input channels in respective processes and identify the relevant information by aggregating a plurality of similarity and relevancy scores as well as the real-time user feedback data in a unique process.

[0031] The output provided by the system may comprise the relevant information and/or answer in response to the user input querying the information. The term “relevant information” as utilized herein may generally refer to target of information to satisfy the user’s information needs or texts in one or more sections from the original document (e.g., highlight salient passages) relevant to the user’s query such that the user can make informed decisions based on such information. For instance, the system output may comprise relevant pieces of information which the user may rely on to determine/satisfy some criterion. The term “answer” as utilized herein may refer to a word, short phrase or a span of texts that directly answer a question specified in the query input. In some cases, an “answer” may not be provided if there is no question specified in the query input, when the question cannot be answered by extractive texts from the original document, and/or when the information collected by the retriever is insufficient to answer the provided question.

[0032] In some cases, the system may employ different back-end processes or models to address the different types/formats of queries (e.g., intelligent question, search terms, positive or negative keywords, etc.) to enable accurate extraction of “answer” (words/short phrases) and relevant “sections” (e.g., paragraphs/clauses) from diverse text, enabling documents to be turned into actionable data. In the present context, words and short phrases may be considered “answer” and are extracted in a question-answer process or utilizing a question-answering model (reader engine). Longer chunks of text may be considered “sections”, and may be extracted in a section extraction process. In some cases, document data may also be extracted from semi-structured data such as tables in a table extraction process. In some cases, the various different models for extracting the answer and relevant passage/section may generally refer to data extraction models but with different input features or network architecture.

[0033] When the raw documents are processed by the system herein for data extraction, the document may be tokenized and may be processed to build document datasets. In some cases, indexes are generated for the datasets. In some cases, the indexes may comprise sparse index each may index or correspond to an element such as a token, a word, a passage or other data structures for efficient lookup. It should be noted that any other suitable data structures may be used for the lookup of the elements. In some cases, the indexes may also comprise a dense index

each may correspond to or index elements such as a representation of textual items (e.g., word embeddings of a chunk of texts). For example, a word embedding (a real-valued vector that encodes the meaning of the word such that the words that are closer in the vector space are expected to be similar in meaning) may be indexed and a dense index may be created. A chunk may, for example, be a sentence, a section, a passage, a paragraph, a clause, or a sub-clause of a document. The chunks may be defined such that they correspond to the format of the document. For example, chunks may not carry over from one part of the document, such as a chapter or sub-chapter, to another or from one paragraph to another. In some cases, chunks of text may be terminated by a full stop. In some cases, at least some chunks may contain multiple words. In some embodiments, a representation of textual items may be indexed and a dense index may be created. For example, a dense index for a section may correspond to the representation or word embeddings of the section. The term “section” as utilized herein may be sentence-level, or longer sections (e.g., paragraphs). In some cases, the sparse index and dense index may be built for both the raw document and input query for information retrieval. Alternatively, the sparse index and dense index may be built only for the document not the query. Details about building the sparse index and dense index are described later herein.

[0034] The term “question” as utilized herein, may generally refer to a question in natural language (e.g., “what is the governing law?”, “what is the operational risk event predicted for file x being delayed from source system A?”) unless the context suggests otherwise. The system herein may provide an “answer” to a question or multiple most likely answers to a question. The answer may be a piece of text in one or more sections or passages which answers the question. For example, the extracted answer may be presented at the user interface by rendering the corresponding characters (text) of the original document and automatically highlighting the corresponding characters in one or more sections of the original document. In some cases, in addition to the texts indicative of the answer to the question, the insight query results may also comprise annotated relevant information (e.g., highlight, underline or any visual indicator) that may or may not encompassing the answer.

[0035] In some cases, an answer may comprise a string of text, generated by a generative Large Language Model (LLM). For instance, an answer may be generated by a generative LLM based on one or more passages identified in response to a user’s question. The answer may or may not be exactly the original texts that are processed by the LLM to form the answer. In some cases, the answer may be generated in natural language where the original texts may serve as supporting evidence. In some examples, the answer (generated answer) may be presented at the user interface by rendering the corresponding characters (text), preceded by the string “Generated:” and automatically highlighting the corresponding one or more sections of the original document

as supporting evidence for the generated answer. By displaying both the generated answer and the supporting evidence (original texts), a user may conveniently visualize and verify the authenticity or authority of the generated answer.

[0036] The term “search term” as utilized herein may generally refer to query terms unless the context suggests otherwise. The one or more search terms may be used to search the relevant information such as based on intent match, phrase match, semantic matching, broad match or others. In some cases, the system may employ one or more algorithms or rules to interpret the query terms and provide results that match the search intent (e.g., results may not contain the exact term). For instance, latent semantic indexing, co-occurring terms and synonyms or other natural language processing techniques may be utilized to provide the search results. In some cases, the search results may comprise the search terms in any order, case insensitive, a word as a truncated search term (e.g., “governed”, “governing”, “govern” are identified with “govern”), or a term has a similar semantic meaning of the search term (e.g., search results containing “salary” or “compensation” may be returned for search term of “remuneration”).

[0037] The term “positive keywords” or “negative keywords” as utilized herein may be treated as soft constraints for determining relevancy of query results. In some cases, the positive keywords or negative keywords may not form part of the query. In some cases, the system may use the presence of a keyword in a section (if positive keyword is specified) to determine a relevancy of the section. Additionally, the system may use presence of a keyword in a section to determine the section as “non-relevant” if the keyword is specified as a negative keyword. In some cases, absence of a keyword in a section may also be used to determine whether the section is relevant or non-relevant. In some cases, one or more sections may be first identified as relevant based on the question and/or search terms then further refined based on the keywords. For example, one or more sections may be identified in response to a query constructed based on the question and/or search terms, and if a positive keyword is matched with a word in an identified section, the section may be labeled as relevant. Similarly, if a negative keyword is matched with a word in an identified section, the section may be labeled as irrelevant. In some cases, the match may not be exact literal match. A match of the positive keywords or negative keywords may be determined using any suitable algorithms. For example, the match may be determined based on regex (regular expression) features where pattern matching may be utilized to search for particular strings of characters rather than constructing multiple, literal search queries.

[0038] FIG. 1 schematically shows a platform 100 in which the method and system herein can be implemented. A platform 100 may include one or more user devices 101-1, 101-2, 101-3, a server 120, a system 121, one or more third-party systems 130, and a storage unit 111, 123. Each

of the components 101-1, 101-2, 101-3, 111, 120, 121, 123, 130 may be operatively connected to one another via a network 110 or any type of communication link that allows transmission of data from one component to another.

[0039] The system 121 may be configured to permit users to perform insight query or information search through one or more documents. The system 121 may include a plurality of functional components such as retriever engine, reader engine, recommending system, user interface module, model creation and management system, and/or various others described elsewhere herein. Details about the various functional components are described later herein.

[0040] In some cases, the system 121 may be configured to train and develop a plurality of predictive models (e.g., RNN, CNN, GAN, classifiers, etc.) consistent with the methods and functions described herein. The system 121 may be configured to perform one or more operations and provide one or more features consistent with those disclosed with respect to FIGs. 2-10. For example, the system 121 may provide pre-trained models and may fine tune the pre-trained models with custom or private datasets to provide customized models. For instance, the system may initialize encoder-decoder model with pre-trained encoder and/or decoder checkpoints (e.g., Bidirectional Encoder Representations from Transformers (BERT), Generative Pre-trained Transformer 2 (GPT-2)) to skip the costly pre-training. In some cases, the system 121 may further comprise binary and multi-category classification models to optimize the ranking of the retrieval results generated by the system, to enrich the user experience and improve the accuracy of the retrieval performance based on the user's intent/knowledge. In some embodiments, the generative models and/or the classification models may be transformer-based models which may be fine-tuned using custom datasets. The datasets may be generated manually such as by manual labelling, collected by the system (e.g., user clickthrough data) or generated automatically or semi-automatically by a labeling system. In some cases, custom datasets may be utilized to fine-tune a preliminary model (e.g., pretrained model). In some cases, insights extracted by the system or newly collected user feedback data may be used to retrain or update a predictive model. Details about the transformer model, artificial intelligence (AI) engines and the training methods are described later herein.

[0041] In some embodiments, the system may be implemented in a cloud-based platform. For example, the front-end of the system may be implemented as a web application using the framework (e.g., Django Python) hosted on an Elastic Cloud Compute (EC2) instance on Amazon Web Services (AWS). The backend of the system may be implemented as serverless compute service such as hosted on AWS Lambda as a serverless compute service running a web framework for developing RESTful APIs (e.g., FastAPI). This may beneficially allow for a large

scale implementation of the system. For instance, the backend system (e.g., AWS Lambda) may partition a separate (e.g., 10GB RAM) compute service for each independent document submission, allowing for a large number of concurrent submissions.

[0042] The system 121 may be implemented anywhere within the platform, and/or outside of the platform 100. In some embodiments, system 121 may be implemented on server 120. In other embodiments, a portion of the system 121 may be implemented on the user device. Additionally, a portion of system 121 may be implemented on the third-party system 130. Alternatively or in addition to, a portion of the system 121 may be implemented in one or more storage units (e.g., knowledge base, data lakes, databases) 111, 123. The system 121 may be implemented using software, hardware, or a combination of software and hardware in one or more of the above-mentioned components within the platform. Details about the plurality of the components of the system 121 are described later herein.

[0043] In some embodiments, a user 103-1, 103-2 may be associated with one or more user devices 101-1, 101-2, 101-3. User device 101-1, 101-2, 101-3 may be a computing device configured to perform one or more operations consistent with the disclosed embodiments. Examples of user devices may include, but are not limited to, laptop or notebook computers, desktop computers, mobile devices, smartphones/cell phones, wearable device (e.g., smartwatches), tablets, personal digital assistants (PDAs), media content players, television sets, video gaming station/system, virtual reality systems, augmented reality systems, microphones, or any electronic device capable of analyzing, receiving (e.g., receiving user input indicating accept, reject or select a system identified relevant information or answer, user input for conducting insight querying, user input for modifying ruleset, etc.), providing or displaying certain types of data (e.g., rendering of a GUI displaying query results, highlighting relevant information and/or answer, rendering document, etc.) to a user. The user device may be portable. In some cases, the user device may be located remotely from a human user, and the user can control the user device using wireless and/or wired communications. The user device can be any electronic device with a display.

[0044] User device 101-1, 101-2, 101-3 may include one or more processors that are capable of executing non-transitory computer readable media that may provide instructions for one or more operations consistent with the disclosed embodiments. The user device may include one or more memory storage devices comprising non-transitory computer readable media including code, logic, or instructions for performing the one or more operations. The user device may include software applications that allow the user to search or query information in one or more documents (e.g., software application provided by third-party server 130), and/or software

applications provided by the system 121 that allow the user device to communicate with and transfer data between server 120, the system 121, and/or the storage unit (e.g., knowledge base or database 111).

[0045] The user device 101-1, 101-2, 101-3 may include a communication unit, which may permit the communications with one or more other components in the platform 100. In some instances, the communication unit may include a single communication module, or multiple communication modules. In some instances, the user device may be capable of interacting with one or more components in the platform 100 using a single communication link or multiple different types of communication links.

[0046] User devices 101-1, 101-2, 101-3 may include a display. The display may be a screen. The display may or may not be a touchscreen. The display may be a light-emitting diode (LED) screen, OLED screen, liquid crystal display (LCD) screen, plasma screen, or any other type of screen. The display may be configured to show a user interface (UI) or a graphical user interface (GUI) rendered through an application (e.g., via an application programming interface (API) executed on the user device). The GUI may display, for example, a user portal with various features such as document upload, query input field, preview of system identified relevant information, relevant sections, extractive answer, and the like. The user device may also be configured to display webpages and/or websites on the Internet. One or more of the web pages/websites may be hosted by server 120, the third-party system 130 and/or rendered by the system 121.

[0047] In some cases, users may utilize the user devices to interact with the system 121 or the third-party system 130 by way of one or more software applications (i.e., client software) running on and/or accessed by the user devices, wherein the user devices and the system 121 or the third-party system 130 may form a client-server relationship. For example, the user devices may run dedicated mobile applications or software applications for accessing the client portal provided by the system 121 or the third-party system 130. The software applications for managing the platform (e.g., admin portal), document processing, and for conducting insight query may be different applications. Alternatively or additionally, the client application may comprise different interfaces/modes for a user to modify/specify heuristics for determining relevancy, perform insight query and view query result, select, reject or accept system identified relevant information, sections or answers, to manage the AI engine or handcrafted rules, respectively.

[0048] In some cases, the client software (i.e., software applications installed on the user devices 101-1, 101-2, 101-3) may be available either as downloadable software or mobile applications for various types of computer devices. Alternatively, the client software can be implemented in a

combination of one or more programming languages and markup languages for execution by various web browsers. For example, the client software can be executed in web browsers that support JavaScript and HTML rendering, such as Chrome, Mozilla Firefox, Internet Explorer, Safari, and any other compatible web browsers. The various embodiments of client software applications may be compiled for various devices, across multiple platforms, and may be optimized for their respective native platforms.

[0049] In some cases, the provided platform may generate one or more graphical user interfaces (GUIs). The GUIs may be rendered on a display screen on a user device 101-1, 101-2, 101-3. A GUI is a type of interface that allows users to interact with electronic devices through graphical icons and visual indicators such as secondary notation, as opposed to text-based interfaces, typed command labels or text navigation. The actions in a GUI are usually performed through direct manipulation of the graphical elements. In addition to computers, GUIs can be found in hand-held devices such as MP3 players, portable media players, gaming devices and smaller household, office and industry equipment. The GUIs may be provided in software, a software application, a mobile application, a web browser, or the like. The GUIs may be displayed on a user device (e.g., desktop computers, laptops or notebook computers, mobile devices, smart phones, personal digital assistants (PDAs), and tablets).

[0050] User devices may be associated with one or more users. In some embodiments, a user may be associated with a unique user device. Alternatively, a user may be associated with a plurality of user devices. A user may be registered with the platform. In some cases, for a registered user, user profile data may be stored in a database (e.g., database 123) along with a user ID uniquely associated with the user. The user profile data may include, for example, user names, user ID, identity, business field, contact information, historical data, and various others. In some cases, a registered user may be permitted to share or publish exported insight information with other users or store the insight information in a storage space provided by the system.

[0051] A server 120 may access and execute the system 121 to perform one or more processes consistent with the disclosed embodiments. In certain configurations, the system may be software stored in memory accessible by a server (e.g., in memory local to the server or remote memory accessible over a communication link, such as the network). Thus, in certain aspects, the system(s) may be implemented as one or more computers, as software stored on a memory device accessible by the server, or a combination thereof

[0052] In some embodiments, one or more systems or components of the present disclosure are implemented as a containerized application (e.g., application container or service containers).

The application container provides tooling for applications and batch processing such as web servers with Python or Ruby, JVMs, or Hadoop or HPC tooling. The various functions performed by the system such as document processing, retriever-reader pipelines, generating ruleset for further modifying AI predictions, model manage system for training a predictive model, executing a trained model, updating and retraining a model using user feedback data and the like may be implemented in software, hardware, firmware, embedded hardware, standalone hardware, application specific-hardware, or any combination of these. The system, and techniques described herein may be realized in digital electronic circuitry, integrated circuitry, specially designed ASICs (application specific integrated circuits), computer hardware, firmware, software, and/or combinations thereof. These systems, devices, and techniques may include implementation in one or more computer programs that are executable and/or interpretable on a programmable system including at least one programmable processor, which may be special or general purpose, graphics processing unit (GPU), coupled to receive data and instructions from, and to transmit data and instructions to, a storage system, at least one input device, and at least one output device. These computer programs (also known as programs, software, software applications, or code) may include machine instructions for a programmable processor, and may be implemented in a high-level procedural and/or object-oriented programming language, and/or in assembly/machine language. As used herein, the terms “machine-readable medium” and “computer-readable medium” refer to any computer program product, apparatus, and/or device (such as magnetic discs, optical disks, memory, or Programmable Logic Devices (PLDs)) used to provide machine instructions and/or data to a programmable processor.

[0053] The third-party system 130 can be any existing platforms or systems that provide document processing, data management and the like. For example, the third-party system may provide software applications to process and analyze document and the insight retrieval functions may be integrated into the applications running on the third-party system 130. In some cases, the third-party system may be hosted by any entity, organization that is interested in the data generated by the system herein. For example, the third-party system may utilize the data for various purposes or via privacy-preserving techniques such as, but not limited to, federated learning. In some cases, the third-party system may be in direct communication with the system 121 such that the document processing, information retrieval and the like may be integrated into the third-party application such as via an API.

[0054] In some cases, the server 120 may also be configured to store, retrieve, and/or analyze data and information stored in one or more of the storage unit (e.g., knowledge base, databases). The data and information may include converted document dataset, indexes (e.g., sparse index, dense index, etc.), extracted information or insight (e.g., locations and offset of tokens

identifying start and end of relevant information or answer), user feedback data, query input data, data about a predictive model (e.g., parameters, model architecture, training dataset, performance metrics, threshold, etc.), and the like. While FIG. 1 illustrates the server as a single server, in some embodiments, multiple devices may implement the functionality associated with a server.

[0055] A server may include a web server, an enterprise server, or any other type of computer server, and can be computer programmed to accept requests (e.g., HTTP, or other protocols that can initiate data transmission) from a computing device (e.g., user device) and to serve the computing device with requested data. In addition, a server can be a broadcasting facility, such as free-to-air, cable, satellite, and other broadcasting facility, for distributing data. A server may also be a server in a data network (e.g., a cloud computing network).

[0056] A server may include known computing components, such as one or more processors, one or more memory devices storing software instructions executed by the processor(s), and data. A server can have one or more processors and at least one memory for storing program instructions. The processor(s) can be a single or multiple microprocessors, field programmable gate arrays (FPGAs), or digital signal processors (DSPs) capable of executing particular sets of instructions. Computer-readable instructions can be stored on a tangible non-transitory computer-readable medium, such as a hard disk, a CD-ROM (compact disk-read only memory), and MO (magneto-optical), a DVD-ROM (digital versatile disk-read only memory), a DVD RAM (digital versatile disk-random access memory), or a semiconductor memory. Alternatively, the methods can be implemented in hardware components or combinations of hardware and software such as, for example, ASICs, special purpose computers, or general purpose computers.

[0057] Network 110 may be a network that is configured to provide communication between the various components illustrated in FIG. 1. The network may be implemented, in some embodiments, as one or more networks that connect devices and/or components in the network layout for allowing communication between them. For example, user device 101-1, 101-2, 101-3 third-party system 130, server 120, system 121, and storage units 111, 123 may be in operable communication with one another over network 110. Direct communications may be provided between two or more of the above components. The direct communications may occur without requiring any intermediary device or network. Indirect communications may be provided between two or more of the above components. The indirect communications may occur with aid of one or more intermediary devices or networks. For instance, indirect communications may utilize a telecommunications network. Indirect communications may be performed with aid of one or more routers, communication towers, satellites, or any other intermediary device or network. Examples of types of communications may include, but are not limited to:

communications via the Internet, Local Area Networks (LANs), Wide Area Networks (WANs), Bluetooth, Near Field Communication (NFC) technologies, networks based on mobile data protocols such as General Packet Radio Services (GPRS), GSM, Enhanced Data GSM Environment (EDGE), 3G, 4G, 5G or Long Term Evolution (LTE) protocols, Infra-Red (IR) communication technologies, and/or Wi-Fi, and may be wireless, wired, or a combination thereof. In some embodiments, the network may be implemented using cell and/or pager networks, satellite, licensed radio, or a combination of licensed and unlicensed radio. The network may be wireless, wired, or a combination thereof.

[0058] User device 101-1, 101-2, 101-3, third-party system 130, server 120, or system 121, may be connected or interconnected to one or more storage units (e.g., databases, knowledge bases) 111, 123. The databases may be one or more memory devices configured to store structured data. Additionally, the databases may also, in some embodiments, be implemented as a computer system with a storage device. The database may make it easy to access ORC, Parquet, Avro or any other types of data stored in object store. In one aspect, the databases may be used by components of the network layout to perform one or more operations consistent with the disclosed embodiments. One or more local databases, and cloud databases of the platform may utilize any suitable database techniques. For instance, structured query language (SQL) or “NoSQL” database may be utilized for storing the document data, indexes, data generated by a predictive model such as extracted insight (e.g., relevant section, information, answer, etc.), and the like. Some of the databases may be implemented using various standard data-structures, such as an array, hash, (linked) list, struct, structured text file (e.g., XML), table, JavaScript Object Notation (JSON), NOSQL and/or the like. Such data-structures may be stored in memory and/or in (structured) files. In another alternative, an object-oriented database may be used. Object databases can include a number of object collections that are grouped and/or linked together by common attributes; they may be related to other object collections by some common attributes. Object-oriented databases perform similarly to relational databases with the exception that objects are not just pieces of data but may have other types of functionality encapsulated within a given object. In some embodiments, the database may include a graph database that uses graph structures for semantic queries with nodes, edges and properties to represent and store data. Also, the database may be implemented as a mix of data structures, objects, and relational structures. Databases may be consolidated and/or distributed in variations through standard data processing techniques. Portions of databases, e.g., tables, may be exported and/or imported and thus decentralized and/or integrated.

[0059] The storage unit may comprise knowledge bases utilized to store complex structured and unstructured information generated and retrieved by the system. The knowledge base may be an

object model with classes, subclasses and instances for storing, for example, user feedback data, extracted information and various other data and information as described elsewhere herein.

[0060] In some cases, the platform 100 may construct the database for fast and efficient data retrieval, query and delivery. For example, the system 121 may provide customized algorithms to extract, transform, and load (ETL) the data. In some embodiments, the system 121 may construct the databases using proprietary database architecture or data structures to provide an efficient database model that is adapted to large scale databases, is easily scalable, is efficient in query and data retrieval, or has reduced memory requirements in comparison to using other data structures.

[0061] In some embodiments, the one or more storage systems 123, 111, which may be configured for storing or retrieving relevant data as described elsewhere herein. In some cases, the system 121 may source data or otherwise communicate (e.g., via the one or more networks 110) with one or more external systems or data sources 111 (e.g., document storage), and third party system 130. In some instances, the system 121 may retrieve data from the storage systems 111, 123 which are in communication with the one or more external systems (e.g., external document management system, etc.) or third-party systems 130 (e.g., industry or company proprietary systems, etc.).

[0062] In some cases, the storage systems can store algorithms or ruleset utilized by one or more methods disclosed herein. In certain embodiments, one or more of the databases may be co-located with the server, may be co-located with one another on the network, or may be located separately from other devices. One of ordinary skills will recognize that the disclosed embodiments are not limited to the configuration and/or arrangement of the database(s). In some cases, data stored in the knowledge base, databases or external databases can be utilized or accessed by a variety of applications through application programming interfaces (APIs). Access to the database may be authorized at per API level, per data level (e.g., type of data), per application level or according to other authorization policies.

[0063] Although particular computing devices are illustrated and networks described, it is to be appreciated and understood that other computing devices and networks can be utilized without departing from the spirit and scope of the embodiments described herein. In addition, one or more components of the network layout may be interconnected in a variety of ways, and may in some embodiments be directly connected to, co-located with, or remote from one another, as one of ordinary skill will appreciate.

[0064] Various aspects of the present disclosure may be applied to any of the particular applications set forth below or for any other types of applications or systems. Systems or methods of the present disclosure may be employed in a standalone manner, or as part of a

package. The system may also allow for an easy and flexible integration of the various personalization features into any existing third-party website or platforms. For instance, the system may provide a plurality of options such as raw application programming interface (API), Plugins, SDK, Google Tag Manager and the like for integrating the AI-based outputs (e.g., extracted information, relevant sections, answer to a question, etc.) to a third-party platform. For example, the system may create various API endpoints for rendering frontend elements and code injection. One or more features (e.g., insight query, document processing, etc.) of the system may be integrated to a third-party application (e.g., company's proprietary software, document management system, etc.). For instance, the system may include a family of plugins, extensions, modules and scripts that facilitate development and integration of the document analysis, insight retriever-reader and services into third-party platforms.

[0065] FIG. 2 schematically illustrates a system 200 for document processing and information retrieval, in accordance with embodiments of the present disclosure. The system 200 can be the same as the system 121 as described in FIG. 1.

[0066] As described above, the system may be implemented on a cloud platform system (e.g., including a server or serverless) that is in communication with one or more user systems/devices via a network. The cloud platform system may be configured to provide the aforementioned functionalities to the users via one or more user interface 210. The user interface may comprise a graphical user interfaces (GUIs), which may include, without limitation, web-based GUIs, client-side GUIs, or any other GUI as described above. For example, a user may upload document and perform insight query via a web-based-GUIs or within a web browser.

[0067] A graphical user interface (GUI) is a type of interface that allows users to interact with electronic devices through graphical icons and visual indicators such as secondary notation, as opposed to text-based interfaces, typed command labels or text navigation. The actions in a GUI are usually performed through direct manipulation of the graphical elements. In addition to computers, GUIs can be rendered in hand-held devices such as mobile devices, MP3 players, portable media players, gaming devices and smaller household, office and industry equipment. The GUIs may be provided in a software, a software application, a web browser, etc. The GUIs may be displayed on a user device or user system (e.g., mobile device, personal computers, personal digital assistants, cloud computing system, etc.). The GUIs may be provided through a mobile application or web application.

[0068] In some cases, the graphical user interface (GUI) or user interface may be provided on a display. The display may or may not be a touchscreen. The display may be a light-emitting diode

(LED) screen, organic light-emitting diode (OLED) screen, liquid crystal display (LCD) screen, plasma screen, or any other type of screen.

[0069] The system 200 may comprise a front-end component 220 allowing users to access the system. In some cases, the user interface (UI) 210 or a graphical user interface (GUI) may be rendered through an application (e.g., via an application programming interface (API) executed on the user device or the user system, or on the cloud) of the front-end component 220.

[0070] The frontend component 220 may receive user input in one or more forms such as question, search term, positive keyword and/or negative keyword and present a search result such as a structured data output comprising relevant information with respect to the question, search terms and/or keywords, or “answers” to the question. The user input can include any other query information such as specifying data types to be sorted (e.g., dates, amounts, etc.). The frontend component 220 may also comprising a UI module for rendering one or more GUIs such as a GUI for uploading document(s), GUI for receiving the user query input in a plurality of input channels (e.g., question, search term, positive keyword and/or negative keyword), GUI for displaying the query result and GUI for receiving and collecting user feedback data. Examples of the user interface, user input fields and output of the system presented within a GUI are described later herein.

[0071] In some cases, the frontend component 220 may allow a user to upload one or more documents (electronic document) to be analyzed by the system for information retrieval. For example, the UI may provide a landing page where the user can upload one or more documents into the platform to start a session for document processing and information retrieval.

[0072] The frontend component 220 may be in communication with a backend component 230. In some cases, the user-uploaded document may be processed by the document dataset builder and indexer 237 of the system 200 to build document dataset and build indexes. Although the document dataset builder and indexer 237 is shown as a component separate from the backend component, one or more components of the document dataset builder and indexer 237 can be an integral part of the backend component.

[0073] In some embodiments, the document dataset builder and indexer 237 may process an uploaded document to build a document dataset by extracting textual data, decomposing the uploaded document, and preprocessing the text. In some cases, a user-uploaded document may be converted, as necessary, from the format in which it is uploaded, for example PDF, to a raw text or html/DOCX representation etc. (in which HTML or other metadata tags may, in some cases, be used to represent document structure, formatting etc.). The term “original document” as utilized herein may refer to the converted document, i.e. after such conversion has been applied

(e.g. in raw text, HTML or DOCX form, as opposed to a pre-converted PDF). The textual data may be extracted using any suitable algorithm. For example, if the uploaded document is PDF or image file, it may be converted to text data using OCR (Optical Character Recognition) techniques.

[0074] The dataset building may further comprise document decomposition. The document dataset builder and indexer 237 may utilize any suitable document analysis techniques to tokenize the document. For example, the document dataset builder and indexer 237 may section the document by merging sentences or utilization of heuristics over pixel values in the PDF to identify ‘chunks’ of text or more advanced document segmentation techniques that account for both the visual and the layout information in the document. The document dataset builder and indexer 237 may merge the chunks to form sections. A chunk may, for example, be a sentence, a paragraph, a clause, or a sub-clause of a document. The chunks may be defined such that they correspond to the format of the document. For example, chunks may not carry over from one part of the document, such as a chapter or sub-chapter, to another or from one paragraph to another. In some cases, chunks of text may be terminated by a full stop. In some cases, chunks may contain multiple words. A section formed by the merging of chunks may correspond to a paragraph, sentence, clause etc. depending on the precise form of tokenization applied.

[0075] The dataset building may also comprise text preprocessing. The text preprocessing may comprise a variety of operations such as stopwords removal, stripping non alphanumeric characters, stripping multiple whitespaces, stripping punctuation, stripping HTML tags (e.g., ...), lower casing, stripping numeric (for creating the sparse index), stemming the text (for creating sparse index) and/or other operations.

[0076] The document dataset builder and indexer 237 may further process the document datasets to build indexes. The indexes may comprise sparse index and dense index. In some embodiments, building sparse index may comprise traversing the preprocessed sections and obtain information about the occurrences of each word in each passage/section. In some cases, building sparse index may comprise collecting token statistics to compute term frequency-inverse document frequency (TFIDF) values for each token in each passage. The sparse index may be used to determine the saliency the word/term in a passage. For example, the sparse index may be computed based on a count of occurrences of a term in the portion of the passage and an inverse document frequency (IDF) weight. The IDF weight may represent how common or rare a word is across the sections in the corpus. For instance, the closer it is to 0, the more common a word is. In an example, the metric (IDF) may be calculated by taking the total number of sections,

dividing it by the number of sections that contain a word, and calculating the logarithm. It should be noted that any other suitable methods can be utilized to calculate the IDF.

[0077] The dense index may be used to retrieve the representation (semantic meaning) of a section or any other data structure. Each section may be indexed and may be retrieved by a dense index. As an example, a dense index may correspond to a dense representation of a passage/section such as a tensor of floats computed to capture the semantic meaning of the section. Any suitable metrics may be utilized for determining the semantic meaning of a passage. For example, a dense index may be computed by iterating through the tokens for each section, and if the TFIDF value of the token is below a threshold, then prune the corresponding term, otherwise keep the term. The threshold may be fixed (e.g., 0.15) or dynamic. In some cases, the threshold may be dynamically determined from the distribution of TFIDF values such as the mean of the TFIDF values throughout the entire document or across one or more sections. Next, for each of the preserved words, the word's embedding vector may be retrieved and scaled the values by the word's TFIDF value. The scaled word embedding vectors may then be stacked up to form a matrix (e.g., matrix of size `len_pruned_passage` x `word_embedding_dimension`). The dense index may comprise a set of these matrices. Each section or passage may have a dense index.

[0078] In some cases, the document dataset and indexes for one or more uploaded documents may be built by the document dataset builder and indexer 237 and stored in the storage 240. The storage 240 may store various other information or data produced or collected during the information retrieval process.

[0079] Upon a user provided input via the frontend component to specify interested information in the document (i.e., insight query), the backend component 230 may process the user input and return the information retrieval result to the frontend component. For example, the backend component may identify relevant texts in one or more sections in the original document containing answer to the question (if the user input query comprises a question) or containing relevant information and pass such information to the frontend, the original documents and the relevant information with visual indicator, as well as texts indicative of answer to the user question, may be rendered at the UI 210 at the front end 220 of the system.

[0080] In some embodiments, the backend component 230 may comprise a retriever-reader paradigm to locate the information in the one or more documents. In some cases, the backend component 230 may comprise a plurality of components such as a retriever engine 231 capable of ranking the one or more sections of the one or more documents by degree of relevance based on the user input query and the document indexes built as described above. The user input query

may be received through one or more input channels or in one or more types (e.g., question, search terms, positive keywords, negative keywords, etc.). In some embodiments, the retriever engine 231 may comprise a plurality of models (e.g., generative model, pseudo-relevance model, retriever score aggregator, etc.) for computing a plurality of scores using the sparse index of the document dataset and the user input, the dense index of the of the document dataset and the user input, using the positive and negative keywords and the document dataset, respectively, as well as assigning scores indicative of probability of relevance of each section, and/or combining the plurality of scores to compute an “information retrieval score” (e.g., IR_score) for each section. The score may indicate the degree of relevancy. In some cases, the IR_score may be further aggregated with a score based on an answer identified to question to generate a final relevance score. Details about the information retrieval (IR) pipeline of the retriever engine and the operations and models of the retriever engine are described with respect to FIGs. 3-4.

[0081] In the case when the user input comprises a question received through the question input channel, the retriever engine 231 may pass the top ranked sections or most relevant sections (ranked based on the aggregated IR_score) to the reader engine 233. The reader engine 233 may further process the top ranked sections to identify the “answer” to the question in each section. The reader engine 233 may comprise a deep neural network trained to extract or retrieve a span/piece of text (e.g., start and end locations of tokens and token offset for the answer, location of an answer in a table, coordinates of the answer on a page) from a section, a passage or a page which contains answer to the question. In some cases, the reader engine 233 may not be trained to answer a given question. Instead, the reader engine may be trained to understand/interpret the question and identify a piece of texts in the one or more sections (passed by the retriever engine) that contains the answer to the question. In some cases, the reader engine 233 may be an extractive-type question answering system that is trained to extract answer from the original document. As described above, the reader engine may output a score based on the answer identified in a section which is further combined with the IR_score associated with the corresponding section to generate a final relevance score. In some cases, the reader engine 233 may also perform continuous learning based on user feedback data.

[0082] In some cases, the reader engine may further comprise a generative model such as a generative Large Language Model (LLM) which is trained to generate new text as an answer to a question based on the piece of texts identified from the original document. Details about the reader engine are described later herein.

[0083] In some embodiments, the backend component 230 may further comprise a recommender engine 235. In some cases, the recommender engine 235 may comprise a lightweight model

trained to improve the ranking of the sections (based on the information retrieval scores). For instance, the model may be trained to optimize the interpolation of the information retrieval scores generated by the retriever engine. In some cases, the input features for the recommender model may comprise the plurality of scores from the information retrieval pipeline (e.g., sparse index score, dense index score, generative model score, pseudo relevance score, etc.) and the output may be a combined/improved IR_score. In some embodiments, the recommender model may be a classifier trained on the clickthrough data of the user indicative whether a given section is relevant. For example, the training datasets may be paired datasets or labeled data generated based on a user clicking on one or more sections indicative of relevancy. This beneficially improves the relevancy score by taking into account the user knowledge or feedback collected during the process.

[0084] In some cases, one or more systems or components of the system (e.g., frontend component, backend component) may be implemented as a containerized application (e.g., application container or service containers) as described elsewhere herein. The application container may provide tooling for applications and batch processing, such as web servers with Python or Ruby, JVMs, or even Hadoop or HPC tooling. For instance, the frontend of the system may be implemented as a web application using the framework (e.g., Django Python) hosted on an Elastic Cloud Compute (EC2) instance on Amazon Web Services (AWS). The backend of the system may be implemented as serverless compute service such as hosted on AWS Lambda as a serverless compute service running a web framework for developing RESTful APIs (e.g., FastAPI). This may beneficially allow for a large-scale implementation of the system. In some cases, the backend system (e.g., AWS Lambda) may partition a separate (e.g., 10GB RAM) compute service for each independent document(s) submission and/or session, allowing for a large number of concurrent submissions.

[0085] In some cases, one or more functions or operations consist with the methods described herein can be provided as software application that can be deployed as a cloud service, such as in a web services model. A cloud-computing resource may be a physical or virtual computing resource (e.g., virtual machine). In some embodiments, the cloud-computing resource is a storage resource (e.g., Storage Area Network (SAN), Network File System (NFS), or Amazon S3.RTM.), a network resource (e.g., firewall, load-balancer, or proxy server), an internal private resource, an external private resource, a secure public resource, an infrastructure-as-a-service (IaaS) resource, a platform-as-a-service (PaaS) resource, or a software-as-a-service (SaaS) resource. Hence, in some embodiments, a cloud-computing service provided may comprise an IaaS, PaaS, or SaaS provided by private or commercial (e.g., public) cloud service providers.

[0086] It should be noted that methods and systems of the present disclosure may utilize any type of machine learning algorithms, architectures or approaches. The machine learning algorithm may comprise one or more of the following: a support vector machine (SVM), a naïve Bayes classification, a linear regression, a quantile regression, a logistic regression, a random forest, a neural network, convolutional neural network (CNN), recurrent neural network (RNN), a gradient-boosted classifier or regressor, or another supervised or unsupervised machine learning algorithm (e.g., generative adversarial network (GAN), Cycle-GAN, etc.). It should be noted that the systems and methods here may utilize any suitable statistical or neural models including, but not limited to, factor graphs, probabilistic graphical models, neural networks and the like.

[0087] FIG. 3 and FIG. 4 shows examples various components and operations of the information extraction and retrieval system, in accordance with embodiments of the present disclosure. The retriever 310, recommender 330 and reader 320 can be the same as the retriever engine, reader engine and recommender engine as described above. In some embodiments, the system may allow users to specify the information of interest/need via one or more input channels or types. In some cases, the one or more input channels may comprise question 301, search terms 303 and positive and negative keywords 305. A user may choose to provide the query input using one or more of the multiple channels. As described above, the “question” channel 301 may permit a user to input a question in natural language (e.g., “what is the governing law?”, “what is the operational risk event predicted for file x being delayed from source system A?”). In some cases, the question may be processed to determine whether it is a valid question. A valid question may be the one that can be answered by identifying a span/piece of texts in the original document (e.g., “what is the governing law?”). An invalid question may be one that no extractive answer can be identified (e.g., “is the agreement construed under French law?”). Alternatively, the question may be processed regardless the validity. The system may return “no answer” if no extractive answer is identified. In some cases, the system may comprise a generative model such as a generative LLM to generate answer texts based on the extractive answer. For example, the system may return a generated answer text (e.g., not exactly the same as the original texts) based on the piece of texts provided by the reader engine. The question input 301 may be fed to both the retriever 310 and the reader 320. The question input 301 may be fed to the reader 320 and used by the reader to identify an answer in one or more relevant sections/passages. The question input fed into the retriever 310 may be pre-processed 311 to be tokenized into words and may form part of a unified query.

[0088] The search term input 303 may be query terms. The one or more search terms may be used to search the relevant information such as based on intent match, phrase match, semantic matching, broad match or others. In some cases, the system may employ one or more algorithms

or rules to interpret the query terms and provide results that match the search intent (e.g., results may not contain the exact term). For instance, latent semantic indexing, co-occurring terms and synonyms or other natural language processing techniques may be utilized to provide the search results. In some cases, the search terms may be treated regardless the order (e.g., *caput mundi* and *mundi caput* will result in the search terms {caput mundi}), and/or may be case insensitive. In some cases, the search term may be treated as word that is truncated to the root (e.g., “governed”, “governing”, “govern” are identified with “govern”). This beneficially provides a more robust search result by avoiding OCR errors or spelling errors which may occur at the end of a word (e.g., “gorvennnned” is treated as “govern”). In some cases, the search term may be treated for searching a term has a similar semantic meaning of the search term (e.g., search results containing “salary”, “compensation” will be returned for search term of “remuneration”) rather than the literal exact match.

[0089] In some embodiments, the search term 303 and the question 301 may be pre-processed 311 by the retriever to generate a unified query for search. The unified query may be generated based on the question, the search terms or a combination of both depending on the availability. For instance, if the user input query includes only a question, the unified query is constructed based on the question only. In some cases, the question 301 and the search terms 303 may be tokenized into words and combined into a single, unified query 312 for search. In some cases, generating the query 312 may comprise de-duplicate terms in the question and the search terms. The unified query 312 may be used to score the sections/passages of the one or more uploaded documents with both the sparse index and the dense index 313. For instance, the unified query may be processed for index construction (i.e., building sparse index and dense index for the unified query) for retrieving the query and representation of the query. Alternatively, no index is built for the query. The representation of the query may be compared against the representation of sections of the documents to calculate the similarity score where the section and representation of the sections are retrieved by the sparse index and the dense index respectively.

[0090] The keywords input 305 may comprise positive keywords, negative keywords or a combination of both. In some cases, the keywords may be used in a form of distant supervision. For example, the keywords may be used to build “weak” or “soft” labels via aggregation of the labelling functions, indicator functions or signals, such as by majority vote or by a generative model built on top of the labelling functions (e.g., utilizing the labels as emission probability features). In some cases, the keywords may be used as “weak” or “soft” labels for determining relevancy of query results. In some cases, the keywords may not be used to construct the unified query. In some cases, the system may use the presence of a keyword in a section (positive keyword or negative keyword) to determine a relevancy of the section. The section may be

identified as a search result based on the question and/or search terms. For example, one or more sections may be identified as relevant based on question and/or search terms, and if a positive keyword is matched with a word in a given section, the section may be labeled as relevant. Similarly, if a negative keyword is matched with a word in a given section, the section may be labeled as irrelevant. The match of the positive keywords or negative keywords may be determined using any suitable algorithms. In some cases, the match may not be exact literal match. For example, the match may be determined based on regex (regular expression) features wherein pattern matching may be utilized to search for particular strings of characters rather than constructing multiple, literal search queries. In some cases, the keywords may be used to construct indicator functions that may be used to generate training datasets to train a generative model. Alternatively or additionally, the keyword may be used as scoring function (e.g., emission probability matrix). In some cases, the keywords may not be used to form the unified query but a score outputted by the generative model may be aggregated with the sparse index score and dense index score to generate a combined relevancy score for a section.

[0091] The keywords 305 may be inputted in various formats such as comma-separated lists of words. In some cases, the system may allow users to specify different spellings of a keyword, conjunctions and disjunctions of words and the like by separating the keywords with "|" or any other suitable symbols, using a DSL (domain specific language) and the like. In some cases, a user may be required to provide at least one keyword for each type (positive or negative) in order for the keywords to have an effect on the search result. Alternatively, a user may be required to provide a keyword for either type.

[0092] The retriever 310 may comprise a plurality of models to compute a plurality of scores and aggregate the plurality of scores to output an information retrieval score (IR_score) in a unique process. The retriever pipeline may comprise searches based on the sparse and dense indexes, models trained following a pairwise protocol in a learning-to-rank framework, relevancy predictions by generative models trained using the keywords, and an aggregation of the plurality of scores to return a single information retrieval score.

[0093] As shown in the example, the retriever 310 may perform sparse index scoring based on the sparse index corresponding to the unified query 313 and the sparse index corresponding to the section/passage. The sparse index score 314 may be a lexical match score measuring a lexical match between the query terms and each section terms based on the sparse index. The sparse index score for each section may be computed by finding all matches between the query terms and section terms (after the preprocessing of both the query terms and the section terms) and then weighing each match by how important the matched term is in the corpus of sections. Any

suitable lexical match retrievers such as BM25 or other methods such as Boolean Retrieval and statistical language models may be utilized. In some case, the retriever may utilize the BM25 as the lexical-match score to determine the similarity (degree of relevancy). The lexical match may advantageously provide improved efficiency. With summation over exact matches, scoring of each query term only goes to documents that contain matching terms. This can be done efficiently using inverted list indexing. The lexical-match score can also be conveniently applied to n-grams.

[0094] The retriever 310 may also compute a dense index score 315 to measure a similarity between the query word and each section/passage. The dense index score may be a semantic similarity score. The query word may be processed to generate a representation such as a tensor of floats which is computed along with the representations of sections stored in the dense index to compute the similarity. In some cases, a dense index may be created corresponding to the representation that can be used for lookup of the representation. Alternatively, the system may not generate a dense index for retrieving the query representation. In some cases, a dense index score may be generated corresponding to an RV score (multivariate generalization of the squared Pearson correlation coefficient) between the query word embedding matrix and word embedding matrix of each passage. This beneficially allows for the use of RV score for fast semantic search with lightweight models. The query matrix (e.g., tensor of floats or dense index) may be constructed similarly to the passage matrix (e.g., dense index for the passage) except that the terms may not be pruned in the unified query.

[0095] The retriever 310 may also compute a generative model score 317. The generative model score 317 may be computed by an unsupervised probabilistic model which interprets the presence of positive and negative keywords in any given section into a probability that the section is relevant. The generative model score 317 may be the output of a generative model. As described above, the generative model may produce weak labels using the keywords 305.

[0096] The keywords 319 may be used to construct an indicator function that may return, for each passage, a statement of whether the keyword is found in the passage (a match of the keyword). In some cases, the indicator function may comprise user-defined rules indicating whether a section is relevant or irrelevant. A user may be permitted to specify different kinds of heuristics or rules to indicate whether a section is likely be relevant or irrelevant. A user may be permitted to specify any other heuristics such as types of data (e.g., date, type of entity, addresses) and the like. Alternatively, the positive/negative keywords received via the user interface may be used by the system to construct the indicator function automatically without requiring a user to define the rules explicitly. For instance, for each positive keyword, if a match

is determined, the indicator function may return a Relevant label. For each negative keyword, if a match is determined, an Irrelevant label may be returned for the given passage. Such observation may then be fed to an unsupervised generative model as the model's emission features (i.e., emission probabilities). In some cases, the retriever may further assign a prior probability of relevance for each passage. The generative model may then be fitted to the entire universe of passages across all uploaded documents. The generative model may be applied to make inference to obtain a posterior probability distribution over the probability of relevance of each passage 317. For example, the output of the generative model may comprise a generative model score 317 in the range [0, 1] assigned for each passage.

[0097] In some cases, the retriever 310 may fit and apply a pseudo-relevance model 318 to the sparse index 314, dense index 315 and generative scores 317 to generate a pseudo-relevance score 316. In some cases, the pseudo-relevance model 316 may be trained using supervised learning and may be a scoring model. In some embodiments, the paired datasets or the labels (relevant/not relevant) used to train the model may not be provided by human. Instead, the labels may be automatically generated from ranking based on the other scores as proxies (e.g., BM25 score, RV similarity score, generative model score, etc.). The pseudo relevance score may indicate answer to the question "What caused the other scores to be high (low)?" and may use the patterns (e.g., rankings based on the BM25 score, RV similarity score, generative model score) it learns to assign scores to sections based on what it thinks makes a section relevant.

[0098] In some cases, the labels (relevant/not relevant) used to train the model may be automatically generated by: ranking the sections/passages according to the following priority: first by the sparse index score (e.g., BM25 score), then by the dense index score (e.g., RV similarity score) and finally by the generative model's predicted probability of relevance (e.g., generative model score). A number (n) of the top ranking passages across all documents (e.g., top ranking 10 passage) may be assumed to be relevant (pseudo relevant) and a number (m) of the bottom ranked passages (e.g., bottom ranking 100 passages) may be assumed irrelevant (pseudo irrelevant). Based on the cluster hypothesis (i.e., closely related documents tend to be of relevance to the same query request), the top-n and bottom-m passages as ranked based on the other scores (e.g., BM25, RV similarity metric, generative model), are likely to be approximately linearly separable in TFIDF space and suitable for training a logistic regression model.

[0099] These passages along with the pseudo labels may form paired datasets (passage, pseudo-label) used for training a logistic regression model. Although the clustering hypothesis justifies the use logistic regression fit on TFIDF features for the top-n/bottom-m passages as a pseudo-relevance model, other types of models may be utilized. In some cases, the logistic regression

model may be trained using bag of words features (e.g., uni-grams and bi-grams) in the pairwise approach to Learning-to-Rank. The bag of words (BoW) features may represent the occurrence of words within a document disregarding the order or structure of the words. The Pseudo-Relevance model once properly trained, may predict, for each passage, the probability of relevance (e.g., pseudo-relevance score 316). The BoW features such as the uni-gram and bi-gram TFIDF vectors can be good representatives of document content supported by the cluster hypothesis.

[00100] The plurality of scores comprising the sparse index score 314, dense index score 315, pseudo-relevance score 316 and generative model score 317 may then be combined 349 to produce a single information retrieval score (e.g., IR_score). In the absence of user feedback, a weighted average may be used to combine the plurality of scores. For example, the retriever 310 may comprise a retriever score aggregator to combine the scores associated with each passage (e.g., BM25 score, RV similarity score, generative model's probability of relevance, and the probability of relevance i.e., Pseudo-Relevance model score) to compute a single information retrieval score

as follows:
$$\text{ir_score} = 0.2 * (\text{bm25_score} + \text{rv_score} + \text{generative_model_score}) + 0.4 * \text{pseudo_relevance_score}.$$
 It should be noted that the information retrieval score can be computed using any suitable formula and/or coefficient to combine the varieties of scores. Alternatively, other methods such as rank-based approaches may be employed to determine the relevancy instead of computing a score.

[00101] In some cases, when the user feedback is available (e.g., user provided clickthrough data by clicking on sections that are relevant), a Relevance Model 331 of the recommender system 330 may be used. The relevance model 331 may be a lightweight model used to optimize the interpolation of the information retrieval scores 333 to improve the relevancy ranking in response to user feedback. The relevance model 331 may be trained to improve the ranking of the sections. For instance, the model may be trained to optimize the interpolation of the information retrieval scores generated by the retriever engine 333. It should be noted that other methods such as ranking-based approach may be used to generate the IR score. For example, IR score may be generated based on the ranks of the other scores such as utilizing Reciprocal-Rank-Fusion. In some cases, the input features for the relevance model 331 may comprise the scores from the information retrieval pipeline (e.g., sparse index score, dense index score, generative model score, pseudo relevance score). In some embodiments, the recommender model may be a classifier trained on the clickthrough data of the user indicative whether a given section is relevant. For example, the training datasets may be paired datasets or labeled data generated based on a user clicking on one or more sections indicative of relevancy.

In some cases, the recommender model may be automatically retrained when positive sections as clicked by a user is greater than a predetermined threshold 348.

[00102] The retriever may identify a number of most relevant sections (e.g., top n sections ranked by IR_score 335) and documents which are most-likely to contain the relevant information based on the IR score 333. In some cases, the relevance score of a document may be defined to be equal to the relevance score of its most relevant section. If the user has not specified a Question in the use input, the top n documents and sections may be presented to the user within the GUI in the order of decreasing relevance score. If the user input comprises a Question, the top n relevant sections of the documents may be passed to the reader 320 for further processing.

[00103] As shown in **FIG. 4**, the reader 320 may comprise an extractive question answering model 321. The extractive question answering model 321 may be a deep neural network trained to extract/retrieve a span/piece of text (e.g., start and end token offset for highlighting the answer) from a section/passage which contains answer to the question. In some cases, the extractive question answering model 321 may be trained to generate an answer to a given question by processing the documents. In some cases, the extractive question answering model 321 may be trained to understand/interpret the question and identify a piece of texts in the one or more sections (passed by the retriever engine) that contains the answer to the question. The extractive question answering model 321 may be an extractive-type question answering system that extract answer from the original document. For example, the extractive question answering model 321 may not return an answer if the question is invalid (e.g., question cannot be answered by identifying spans of texts in the original document).

[00104] In some embodiments, the reader 320 may further comprise a generative Large Language Model (LLM) to generate an answer in natural language. The generative LLM may be running as a service, hosted by the system herein or a third-party system. For example, the service may be hosted by OpenAI, and accessed through the corresponding API. The underlying LLM model may be a GPT-type model, and the API calls may be configured to use the available model (e.g., text-davinci-003 model or gpt-3.5-turbo or any other model that OpenAI or other vendors make available). Alternatively, the LLM service may be built and hosted by the system or platform herein as an integral component.

[00105] In some cases, the input to the generative LLM may comprise the piece of texts identified by the retriever engine from the original document and the question. For example, one or more sections or passages identified by the Retriever may be combined with the natural question in a prompt to the generative LLM. In some cases, the prompt may be specified to

instruct the LLM to generate the answer based only on the identified sections or passages and use the same wording as provided in the passages. For example, the prompt may specify in natural language that "Given these passages [insert retrieved passages here], answer the following question [insert question here]. Only repeat literal text from the provided passages in your answer and do not include any other words." In another example, the prompt may specify the model's "confidence" in its predictions to be a higher confidence (e.g., a lower confidence level resulting in more diverse and unexpected text and a higher confidence level resulting in more conservative and predictable text) such as by setting the temperature parameter of the model to 0, in order to get reproducibility of the answer.

[00106] In some cases, the generated answer from the LLM may be compared to the identified piece of texts or passages provided in the prompt and a substring from each passage provided as the identified answer based on a fuzzy string matching algorithm. The fuzzy string matching algorithm may identify a match of the generated answer at a certain position in a given passage if the number of characters in agreement between the generated answer and the given passage, when generated answer is placed at that position in the given passage, exceeds a given threshold. In some cases, the matched texts may be highlighted or displayed with visual indicator so the user can visually verify the supporting texts. In some cases, when a match cannot be identified then the generated answer may be presented to the user directly, with all of the retrieved passages highlighted as 'supporting evidence.' In some cases, an invalid question for an extractive question answering model may still be returned with a generated answer provided by the LLM. For example, question cannot be answered by identifying spans of texts in the original document may still get an answer generated by the LLM based on paragraphs identified by the retriever engine. Displaying the retrieved paragraphs and/or visual indicator of the supporting evidence may beneficially allow a user to visualize and verify the authenticity or authority of the generated answer.

[00107] In some cases, the extractive question answering model 321 may perform continuous learning based on user feedback data. In some cases, the output 323 of the extractive question answering model 321 may comprise an answer (e.g., comprising start and end token offset for the spans of texts) and a score (e.g., QA_score) indicative of likelihood of an answer. In some embodiments, the extractive question answering model 321 may be a transformer model for processing the question input, document dataset and identifying the answer to the question. As an example, the transformer model may a Bidirectional and Auto-Regressive Transformer (BART) which is a Transformer that combines the Bidirectional Encoder (i.e. BERT like) with an Autoregressive decoder (i.e. GPT like) into one Seq2Seq model. The Bidirectional Encoder Representations from Transformers (BERT) is a transformer language model with a variable

number of encoder layers and self-attention heads. The BERT may be pre-trained on two tasks: language modelling (e.g., tokens may be masked and BERT may be trained to predict them from context) and next sentence prediction (BERT may be trained to predict if a chosen next sentence was probable or not given the first sentence). As a result of the training process, BERT learns contextual embeddings for words. After pre-training, BERT can be finetuned with less resources on smaller datasets such as custom datasets to optimize its performance on specific tasks such as generating opener sentences.

[00108] The extractive question answering model 321 may be built on top of a BERT large whole word masking model with a question answering head on top (e.g., trainable start vector of answer span and end vector of answer span). As an example, the network architecture may comprise a transformer model with a Multi-Head Attention mechanism to compute representations of its input and output without using sequence aligned RNNs or convolution. The model may use stacked self-attention and point-wise, fully connected layers for both the encoder and decoder. For example, the encoder may comprise a stack of N identical layers each with two sub-layers. The first sublayer is a multi-head self-attention mechanism, and the second sub-layer is a simple, position-wise fully connected feed-forward network. In some cases, a residual connection may be employed around each of the two sub-layers, followed by layer normalization. The decoder may comprise a stack of N identical layers. In addition to the two sub-layers in each encoder layer, the decoder may insert a third sub-layer, which performs multi-head attention over the output of the encoder stack. Similar to the encoder, residual connections are employed around each of the sub-layers, followed by layer normalization. In some cases, the self-attention sub-layer in the decoder stack may be modified to prevent positions from attending to subsequent positions. The multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions. An attention function may map a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key. The input consists of queries and keys of dimension d_k , and values of dimension d_v . The multi-head attention may linearly project the queries, keys and values h times with different, learned linear projections to d_k , d_k and d_v dimensions, respectively. On each of these projected versions of queries, keys and values the attention function is performed in parallel, yielding d_v -dimensional output values. These are concatenated and once again projected, resulting in the final values. As an example, the input tokens/output tokens may be converted to embeddings such as vectors of dimension d_{model} . A learned linear transformation and softmax function may be used to convert the decoder output to predicted next-token probabilities. In some

cases, the model for extracting unsupervised features may be refined or tuned as new data is collected from the system.

[00109] The generative LLM may be running as a service, hosted by the system herein or a third-party system. For example, the service may be hosted by OpenAI, and accessed through the corresponding API. The underlying LLM model may be a GPT-type model, and the API calls may be configured to use the available model (e.g., text-davinci-003 model or gpt-3.5-turbo or any other model that OpenAI or other vendors make available). Alternatively, the LLM service may be built and hosted by the system or platform herein as an integral component.

[00110] It should be noted that the example model architecture is for illustration purposes only, people skilled in the art would recognize that various other types of encoder-decoder models can be utilized. For instance, the model may comprise a sparse autoencoder with an RNN (recurrent neural network) architecture, such as LSTM (long-short-term memory) network, may be trained to regenerate the question input and document data. For example, an encoder-decoder LSTM model with encoder and decoder layers may be used to recreate a low-dimensional representation of the input data (i.e., low-dimensional features) to the following model training despite a latent/hidden layer. Once the unsupervised features (e.g., low-dimensional representation) are extracted, the extracted features may be used to train two or more classifiers, which accept the extracted features as inputs and map them to the appropriate class label.

[00111] In some embodiments, the language model may be pre-trained on a large dataset. The complete reader model may be fine-tuned on the dataset comprising question-passage pairs (e.g., ~100k question-passage pairs) for the question answering head. The model's pre-trained checkpoint may be loaded to the system then the model is further trained on the system's Treasure Chest data (e.g., ~25k question-passage pairs). The treasure chest data may be data which have not been used to train the model. As an example, the weighted F1 score on the Treasure Chest test set is about 86% over 46 question "types" (e.g., governing law, agreement date, parties, etc.).

[00112] In some embodiments, the system herein may comprise sockets to integrate the variety of machine learning trained models and/or functional components. In some cases, for time-consuming or computationally expensive models, a socket to a GPU API may be provided. Other less-computationally expensive processes may be implemented as parallel CPU processing steps. A socket may provide a programming construct, an instance, or instantiate, that can make use of suitable protocol to send and receive data. For example, a socket may be Web Socket APIs allow bi-directional, full-duplex communication between clients and servers. The API may not require a new connection to be set up for each message to be sent between clients and servers.

Once the connection is set up the data can be sent and received continuously without any interruption. Sockets such as WebSocket APIs are suitable for Applications with low latency or high throughput requirements. Alternatively or additionally, various functional components of the system herein may be seamlessly integrated into a third-party platform or on-premises environment via customized Software Development Kit (SDK).

[00113] The QA_score, answers identified by the reader 320 and the IR_score 333 generated by the recommender 330 or the retriever 310 may be processed by a relevance score calculator 340 to generate a final relevance score 341. The relevance score calculator 340 may utilize any suitable equation or algorithm to calculate the final relevance score. For instance, the final relevance score associated with a section may be calculated based on both the corresponding IR_score, and the score (e.g., QA_score) indicative of the presence of an answer identified in the section. An example of the method for calculating the final relevance score is the following: $IR_score + (-1)^{has_answer} * QA_score$. The system may then retrieve 342 the top n (e.g., $n=1, 2, 3, 4, 5, 6$, etc.) sections/passages ranked based on the final relevance score as the most relevant sections and display to the user within the GUI 343.

[00114] In some cases, the user may provide feedback 344 which may be used to refine, and retrain the models. The models may continuously learn and improve based on the user feedback data. The user feedback 345 may indicate approval or rejection of relevant information, relevant passage and/or answer identified by the system. For example, a user may be presented a number of top ranked sections within the GUI. The GUI may provide a view of the document in a predefined format (e.g., PDF view), on a first panel allowing the user to navigate the properly rendered document and to annotate (e.g., via highlighting, underline, or using any visual indicator) any span of texts in the document. On a second panel, a number of the top ranked passages (e.g., top 5 most relevant passages) may be shown. In some cases, the number of identified relevant passages to be shown may be configurable. The user may inspect the suggested passages, and a click on any of these passages may redirect the PDF view to the passage in the document.

[00115] The GUI may display both the relevant information and the answer using different visual indicators such that such information is easily identifiable by a user. For example, the relevant information in each of these passages may be rendered in a first color (e.g., yellow) on the PDF view. If a Question is specified in the user input, one or more of the passages may further contain an answer to the question annotated/highlighted in a second color (e.g., blue). It should be noted that any suitable visual indicator (e.g., any color for highlighting, underline,

markup, etc.) may be utilized to annotate the relevant information and/or the answer so long as such information is visually identifiable to a user.

[00116] In some cases, the GUI may provide features for collecting user feedback. For example, graphical elements adjacent to each of the suggested answers (e.g., Confirm/Cancel button) may allow the user to confirm or cancel the proposed annotation. In some cases, the user may also be permitted to annotate the relevant span of text directly if an answer has not been found. The relevance feedback may be stored and updated in the relevance feedback store 346.

[00117] In some cases, the user feedback may also include information based on a user action or user interaction with the identified sections. For example, any passage or answer that has been proposed as relevant but the user has not explicitly confirmed may be treated by the system as not relevant to the user's information needs. Any passage which has not been presented to the user (e.g., not contained within the top 5 ranked passages) may be treated by the system as un-annotated (i.e., neither relevant nor irrelevant). In some cases, user actions such as how quickly or the time takes for a user clicks on the suggested information/answer may also be part of the user feedback indicating the relevancy of the retrieved information.

[00118] The user feedback collected by the system herein may indicate relevancy of a passage or answer retrieved by the system herein. In some cases, the relevancy may be binary (e.g., relevant or not relevant) or multi-level (e.g., multiclass, multi-label, or grade indicating relevancy level). Such user feedback may be utilized to further tune/train the models for information extraction and models in the retriever-reader pipelines (e.g., generative model, pseudo-relevance model, relevance model, extractive question answering model, etc.) and/or validate the performance of the aforementioned models (e.g., used for model validation).

[00119] The user generated relevance feedback (e.g., passages and/or answers as relevant/correct or irrelevant/incorrect) may be used to improve the ranking of passages and the relevance of the proposed sections. In some cases, the model (e.g., relevance model 331) may be retrained when the positive passages confirmed/annotated by a user is greater than a retrain threshold 348. Positive passages may refer to passages identified by the system as relevant to the user's query input. For example, at least 15 positive passages may be collected and used to train a model to identify sections of relevance. The model may be retrained when at least 5 additional positive sections (e.g., equal to or greater than a retrain threshold of 5 positive passages) have been accumulated in addition to the initial 15 passages. The retraining may be triggered and executed automatically in the backend component without user intervention. The retrained model may be deployed and may make a prediction for each section to update the relevance of that

passage to the user's interest. The prediction may be a probability that the given passage is of interest to the user and the ranking is based on that probability of relevance.

[00120] **FIGs. 5-10** show examples of user interfaces provided by a frontend component of the system, in accordance with some embodiments of the present disclosure. **FIG. 5** shows an example of a GUI for a user to input query for searching information/insight in one or more uploaded documents. As described above, a user may upload one or more documents to be analyzed by the system for information retrieval. For example, the UI may provide a landing page where the user can upload one or more documents into the platform to start a session for information retrieval. In some cases, a user may be permitted to upload a new document, remove a document and the like via the document upload UI. As an example, a user may provide the document by drag/drop, providing link to the document via the UI and may click a graphical element (e.g., graphical button "Upload Documents") displayed within the UI to upload the document. The system may automatically process the document to build document dataset and indexes as described above.

[00121] Once the document is processed, the user may input query for searching information/insight in the one or more uploaded documents via the GUI comprising multiple input channels as illustrated in FIG. 5. The multiple input channels may allow the user to specify the interested/needed information in multiple forms. For example, a user may be allowed to provide an intelligent question 501, search terms 503, positive keyword 505, negative keywords 507 or any combination of the above to specify the desired information. The input channels may permit a user to provide any other types of inputs (e.g., data types such as dates, addresses, etc.) related to a query as described elsewhere herein. In some cases, the system may determine the type of question and select or adapt the backend models according to the type of question accordingly. For instance, the system may utilize more regexes (regular expressions) if the user inputs indicate the data types are dates.

[00122] The GUI may allow a user to provide inputs to the multiple input channels in any suitable forms. For example, the input may be provided via a text field each corresponding to an input channel for direct text inputs as illustrated in FIG. 5. The GUI may or may not display system suggested search terms, questions, or the keywords. For instance, the system may display suggested question or complete a question as the user typing the question.

[00123] The user input may be provided for at least one of the input channels and may be processed by the system as described elsewhere herein. For example, the "question" field 501 may receive a question in natural language (e.g., "what is the governing law?", "what is the operational risk event predicted for file x being delayed from source system A?"). In some cases,

the question may be received by the backend component and processed to determine whether it is a valid question. A valid question may be the one that can be answered by identifying a span/piece of texts in the original document (e.g., “what is the governing law?”). An invalid question may be one that no extractive answer can be identified (e.g., “Is the agreement construed under French law?”) when the reader engine comprises only the extractive model not the LLM. In some cases, the GUI may prompt the user to enter a new question when an invalid question is determined or display “no answer”. The question may be received and processed by the backend component and used to identify an answer regardless the validity. An invalid question may result in “no answer” displayed to the user within the GUI or only the relevant information being highlighted in the original document. In some cases, an invalid question for an extractive question answering model may still be returned with a generated answer provided by the LLM. For example, question cannot be answered by identifying spans of texts in the original document may still get an answer generated by the LLM based on paragraphs identified by the retriever engine.

[00124] The search terms input field 503 may receive query terms inputted by a user. The search terms may be inputted in the text field 503 in various formats such as comma-separated lists of words. In some cases, the positive keyword input field 505 and negative keyword input field 507 may receive the keywords inputted in various formats such as comma-separated lists of words. In some cases, the system may allow users to specify different spellings of a keyword such as by separating the keywords with "|".

[00125] As shown in **FIG. 6**, the user may provide input for at least one of the multiple channels. The user may start a search session by clicking on the “Rank Sections” 601 button. Once the system has completed the search in the processed as described above, the GUI may show the results to the user. In the illustrated example, the GUI may provide a first panel 610 showing the top n ranked passages (e.g., passages 605, 607) as most relevant to the user input query. For example, up to five passages from the uploaded document may be displayed in the first panel 610 as the search result and ranked based on the relevancy. Each passage may contain relevant information identified by the system. Displaying multiple relevant passages in the first panel may allow the user to conveniently inspect and compare multiple passages. A user may be permitted to select any of the passages such as by clicking on a passage 607.

[00126] A view of the document (e.g., PDF view) with the user selected passage may be displayed in a second panel 620. The second panel 620 may render the document in a pre-defined format (e.g., PDF view) and may update a rendering of a portion of the document upon a user selecting a different passage in the first panel 610. The second panel 620 may allow a user to

navigate the rendered document to annotate relevant information (determined by the user) such as by highlighting, any span of texts in the document. It should be noted that any suitable features can be employed for a user to provide or visualize the relevancy information. For example, a heatmap showing degree of presumed relevance may be displayed within the GUI. The user may navigate within a document (e.g., moving pages) and/or navigate across different documents (e.g., clicking on “next document” 627). In some cases, any annotation generated by the system and/or provided by the user may be stored before the next document is rendered. The user may annotate any texts in the rendered document indicating relevant or not relevant and such input may be utilized as user feedback to further improve the models in the backend component as described above. In the second panel 620, the passage identified as relevant 621 may be highlighted in a first color (e.g., yellow). If the user input includes a question, the highlighted one or more passages may further contain an answer to the question annotated/highlighted in a second color (e.g., blue) 623. In some cases, the second panel may provide graphical elements adjacent to each of the suggested answers (e.g., buttons of Confirm/Cancel 625) allowing the user to confirm or cancel the proposed answer annotation.

[00127] FIG. 7 shows another example of the GUI for a user to provide feedback by confirming or rejecting a system identified answer. A passage containing relevant information may be displayed to the user. The relevant information may be highlighted in a first color (e.g., yellow). In some cases, if an answer is identified for a question as part of the user input, the answer may be highlighted/annotated in a second color 703, such that the answer is easily identifiable by the user. In these cases, the highlighted relevant information portion 701 may be longer, for example, a clause of a legal document, referred to a ‘chunks’ of text. In some cases, the relevant information portion may contain the answer portion 703, for example, a word or phrase, that directly answers the question. That is, when a word or phrase is an answer to a question, it may also form part of a chunk of text which is identified as relevant information based on the search terms or a combination of the search terms and the question. The GUI may permit a user to provide a feedback indicating whether the extracted answer 703 is relevant or not by clicking on the button 705 to confirm or cancel the extracted answer. Although the GUI only illustrates the graphical elements for providing feedback regarding the answer, a user may also be allowed to provide feedback regarding the relevancy of the relevant information 701 in a similar manner.

[00128] As shown in FIG. 8, the GUI may allow a user to change or update a search at any point during the process. A search may be updated by modifying a query input (e.g., modifying search terms, questions, keywords, adding/removing an input to an input channel) and/or by uploading/removing a document to be analyzed. For example, the user may click on the “clear

session” button 801 to search for new information across the same one or more documents (e.g., entering a new question) or uploading one or more new documents. In some cases, the “clear session” may trigger a drop-down menu allowing the user to select to modify a query or modify the underlying documents. In some cases, upon receiving the command to clear a session, the learned information for the session may be cleared. Alternatively or additionally, the learned information may be reserved when a user intends to search across multiple sessions.

[00129] In some embodiments, the GUI may allow a user to refine a search at any point during the process. Refining a search may not start a new search session. For instance, instead of clicking on the “clear session” button, the user may modify the search terms, questions, keywords, adding/removing an input to a selected input channel to refine the input query. The refined input query may be applied to the uploaded documents. All the previous annotations generated by the system or provided by the user may be persisted (e.g., stored in a memory/storage, or database). In some cases, the user may not be shown previously identified passages.

[00130] The GUI may allow users to export the annotated information. For example, once a user is satisfied with having annotated a sufficient number of documents/passages, the annotations (including the relevant information and/or answer, or top N ranked relevant sections) may be exported to a file in a selected format (e.g., json file) by clicking on the “export data” button 803. A user may export and save the extracted information such as by specifying the exported document type, the name of the insight, the question number, the type of extraction and the like. In some embodiments, the user may also be permitted to share and publish the insight via the platform such that the insights may be searchable by the user and/or other users. Alternatively or in addition to, the created insight may be added to a knowledge database that may not be shared with other users. In alternative cases, the knowledge base may be shared with other users with access permission.

[00131] The user may provide inputs in one or more of the multiple channels. The user may be required to provide an input in at least one of the input channels. The input query may comprise any combination of the multiple channels. **FIG. 8** shows an example when the input query contains only search terms. In this case, the extracted relevant information may not comprise an answer whereas the extracted relevant information may be based on the search terms only. **FIG. 9** shows an example where the input query contains only a question. In the illustrated example, the extracted information comprises a word or phrase as an answer to a question, which also forms part of a chunk of text which is identified as relevant information based on a unified

query generated according to the question. **FIG. 10** shows an example where the input query comprises both a question and search terms.

[00132] Aspects of the systems and methods provided herein can be embodied in programming. Various aspects of the technology may be thought of as “products” or “articles of manufacture” typically in the form of machine (or processor) executable code and/or associated data that is carried on or embodied in a type of machine readable medium. Machine-executable code can be stored on an electronic storage unit, such as memory (e.g., read-only memory, random-access memory, flash memory) or a hard disk. “Storage” type media can include any or all of the tangible memory of the computers, processors or the like, or associated modules thereof, such as various semiconductor memories, tape drives, disk drives and the like, which may provide non-transitory storage at any time for the software programming. All or portions of the software may at times be communicated through the Internet or various other telecommunication networks. Such communications, for example, may enable loading of the software from one computer or processor into another, for example, from a management server or host computer into the computer platform of an application server. Thus, another type of media that may bear the software elements includes optical, electrical and electromagnetic waves, such as used across physical interfaces between local devices, through wired and optical landline networks and over various air-links. The physical elements that carry such waves, such as wired or wireless links, optical links or the like, also may be considered as media bearing the software. As used herein, unless restricted to non-transitory, tangible “storage” media, terms such as computer or machine “readable medium” refer to any medium that participates in providing instructions to a processor for execution.

[00133] Hence, a machine readable medium, such as computer-executable code, may take many forms, including but not limited to, a tangible storage medium, a carrier wave medium or physical transmission medium. Non-volatile storage media include, for example, optical or magnetic disks, such as any of the storage devices in any computer(s) or the like, such as may be used to implement the databases, etc. shown in the drawings. Volatile storage media include dynamic memory, such as main memory of such a computer platform. Tangible transmission media include coaxial cables; copper wire and fiber optics, including the wires that comprise a bus within a computer system. Carrier-wave transmission media may take the form of electric or electromagnetic signals, or acoustic or light waves such as those generated during radio frequency (RF) and infrared (IR) data communications. Common forms of computer-readable media therefore include for example: a floppy disk, a flexible disk, hard disk, magnetic tape, any other magnetic medium, a CD-ROM, DVD or DVD-ROM, any other optical medium, punch cards paper tape, any other physical storage medium with patterns of holes, a RAM, a ROM, a

PROM and EPROM, a FLASH-EPROM, any other memory chip or cartridge, a carrier wave transporting data or instructions, cables or links transporting such a carrier wave, or any other medium from which a computer may read programming code and/or data. Many of these forms of computer readable media may be involved in carrying one or more sequences of one or more instructions to a processor for execution.

[00134] The present disclosure provides methods and systems for querying insight information. The method comprises: providing different input fields on a UI for receiving a query input for searching insight information in a document, the input fields comprise a first input field for receiving one or more questions in natural language and a second input field for receiving one or more search terms; receiving the query comprising a question or a search term; identifying, using one or more machine learning algorithm trained models, one or more sections of the document each comprising a chunk of texts relevant to the query input, and rendering the one or more sections in the UI with the chunk of texts annotated with a visual indicator; receiving a user feedback data indicating an acceptance or rejection of the chunk of texts; and improving at least one of the trained models using the user feedback data.

[00135] It should be understood from the foregoing that, while particular implementations have been illustrated and described, various modifications can be made thereto and are contemplated herein. It is also not intended that the invention be limited by the specific examples provided within the specification. While the invention has been described with reference to the aforementioned specification, the descriptions and illustrations of the preferable embodiments herein are not meant to be construed in a limiting sense. Furthermore, it shall be understood that all aspects of the invention are not limited to the specific depictions, configurations or relative proportions set forth herein which depend upon a variety of conditions and variables. Various modifications in form and detail of the embodiments of the invention will be apparent to a person skilled in the art. It is therefore contemplated that the invention shall also cover any such modifications, variations and equivalents.

CLAIMS

WHAT IS CLAIMED IS:

1. A method for querying insight information in a document, the method comprising:
 - (a) providing, via a user interface, a plurality of different input fields for receiving a query input for searching insight information in the document, wherein the plurality of input fields comprise a first input field for receiving one or more questions in natural language and a second input field for receiving one or more search terms;
 - (b) receiving the query input submitted via the user interface, wherein the query input comprises a question or a search term;
 - (c) identifying, using one or more machine learning algorithm trained models at the backend computing system, one or more sections of the document each comprising a chunk of texts relevant to the query input, and rendering the one or more sections in the user interface with the chunk of texts annotated with a visual indicator;
 - (d) receiving a user feedback data via the user interface indicating an acceptance or rejection of the chunk of texts; and
 - (e) improving at least one of the trained models using the user feedback data.
2. The method of claim 1, further comprising generating, at the backend computing system, a query based at least in part on the question.
3. The method of claim 1 or 2, wherein the chunk of texts comprise an answer to the question.
4. The method of claim 3, wherein the answer is annotated with a first visual indicator and the remaining texts in the chunk of texts are annotated with a second visual indicator.
5. The method of claim 3, further comprising generating a unified query by tokenizing at least the question and the unified query is used to identify one or more relevant sections.
6. The method of claim 5, wherein the answer is identified by processing the one or more relevant sections using transformer model.
7. The method of claim 6, wherein the transformer model is fine-tuned using a small amount of training datasets.

8. The method of any preceding claim, further comprising generating a unified query by tokenizing the question or the search term.
9. The method of claim 8, further comprising computing a sparse index score based at least in part on a similarity between the tokenized query and the one or more sections of the document retrieved by a sparse index.
10. The method of claim 9, further comprising computing a dense index score based at least in part on a similarity between a representation of the query and a representation of each section of the document retrieved by a dense index.
11. The method of claim 10, further comprising generating a pseudo-relevance score based at least in part on the sparse index score and the dense index score.
12. The method of claim 11, further comprising generating a combined score by aggregating the sparse index score, the dense index score and the pseudo-relevance score using one of the one or more trained models.
13. The method of claim 12, further comprising retraining the one of the one or more trained models when the user feedback data comprises a number of relevant sections exceeding a predetermined threshold.
14. The method of any preceding claim, wherein the plurality of different input fields further comprise a third input field for receiving a positive keyword and a fourth input field for receiving a negative keyword.
15. The method of claim 14, wherein the query input further comprises a positive keyword or a negative keyword.
16. The method of claim 15, wherein the positive keyword or the negative keyword is utilized to determine a relevance score for at least one of the one or more sections.
17. The method of claim 12, wherein the user interface comprises a first panel displaying a set of sections ranked based at least in part on the combined score, and a second panel rendering at least one of the set of sections with the chunk of texts annotated with a visual indicator, and wherein the second panel displays one or more graphical elements for receiving the user feedback data.

18. The method of any preceding claim, wherein the user interface is a web-based user interface.
19. The method of any preceding claim, wherein the document comprises unstructured data.
20. The method of any preceding claim, wherein the document is processed by the backend computing system for building a document dataset and indexes.
21. The method of any preceding claim, further comprising processing the one or more sections of the document and the query input by a large language model to output a generated answer.
22. A non-transitory computer-readable storage medium including instructions that, when executed by one or more processors, cause the one or more processors to perform operations comprising:
 - (a) providing, via a user interface, a plurality of different input fields for receiving a query input for searching insight information in the document, wherein the plurality of input fields comprise a first input field for receiving one or more questions in natural language and a second input field for receiving one or more search terms;
 - (b) receiving the query input submitted via the user interface, wherein the query input comprises a question or a search term;
 - (c) identifying, using one or more machine learning algorithm trained models at the backend computing system, one or more sections of the document each comprising a chunk of texts relevant to the query input, and rendering the one or more sections in the user interface with the chunk of texts annotated with a visual indicator;
 - (d) receiving a user feedback data via the user interface indicating an acceptance or rejection of the chunk of texts; and
 - (e) improving at least one of the trained models using the user feedback data.
23. The non-transitory computer-readable storage medium of claim 22, wherein the operations further comprise generating, at the backend computing system, a query based at least in part on the question.
24. The non-transitory computer-readable storage medium of claim 22 or 23, wherein the chunk of texts comprise an answer to the question.

25. The non-transitory computer-readable storage medium of claim 24, wherein the answer is annotated with a first visual indicator and the remaining texts in the chunk of texts are annotated with a second visual indicator.
26. The non-transitory computer-readable storage medium of claim 24 or 25, wherein the operations further comprise generating a unified query by tokenizing at least the question and the unified query is used to identify one or more relevant sections.
27. The non-transitory computer-readable storage medium of any of claims 22 to 26, wherein the answer is identified by processing the one or more relevant sections using transformer model.
28. The non-transitory computer-readable storage medium of any of claims 22 to 27, wherein the plurality of different input fields further comprise a third input field for receiving a positive keyword and a fourth input field for receiving a negative keyword.
29. The non-transitory computer-readable storage medium of any of claims 22 to 28, wherein the document is processed by the backend computing system for building a document dataset and indexes.
30. The non-transitory computer-readable storage medium of any of claims 22 to 29, wherein the operations further comprise processing the one or more sections of the document and the query input by a large language model to output a generated answer.

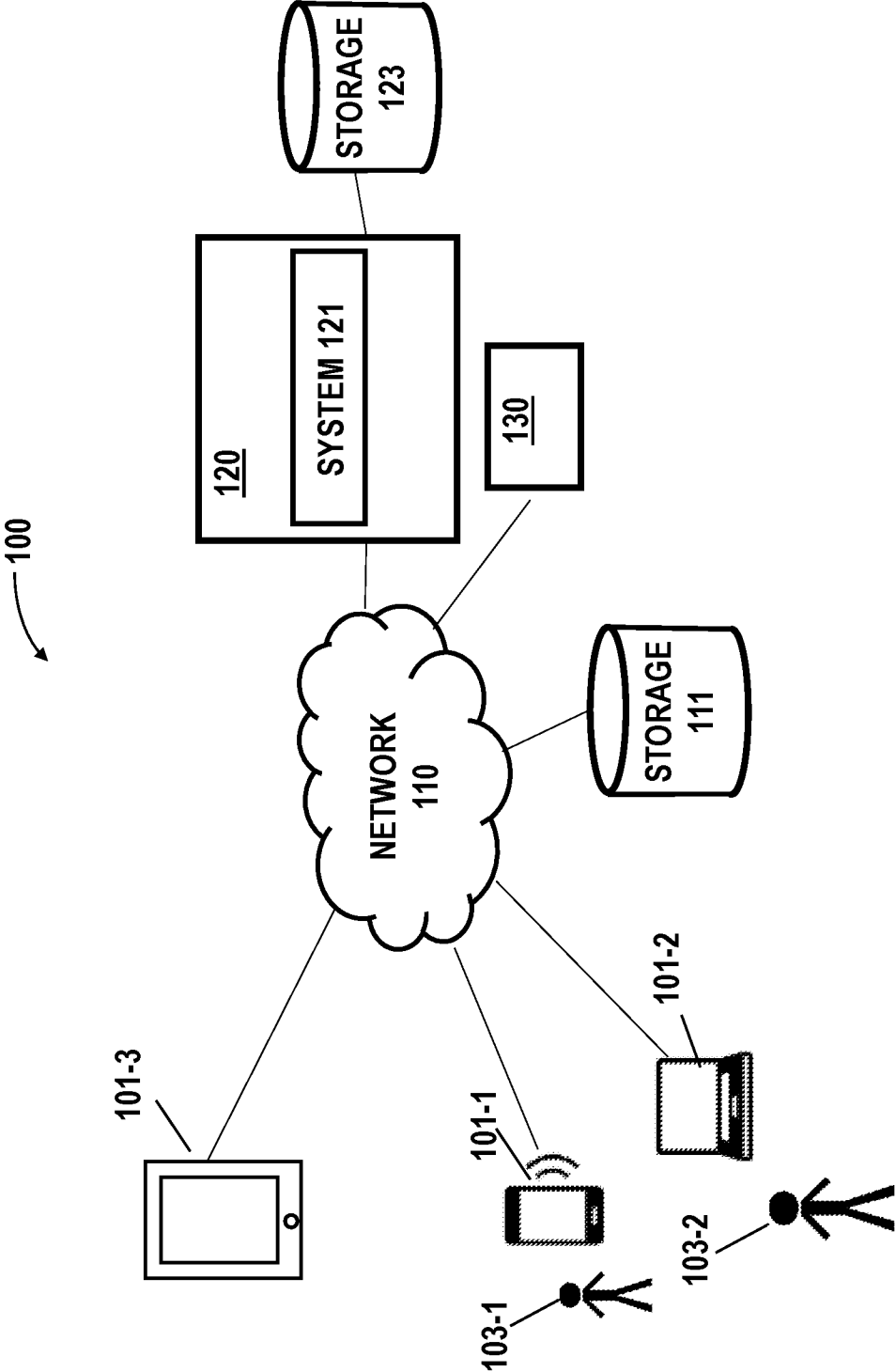


FIG. 1

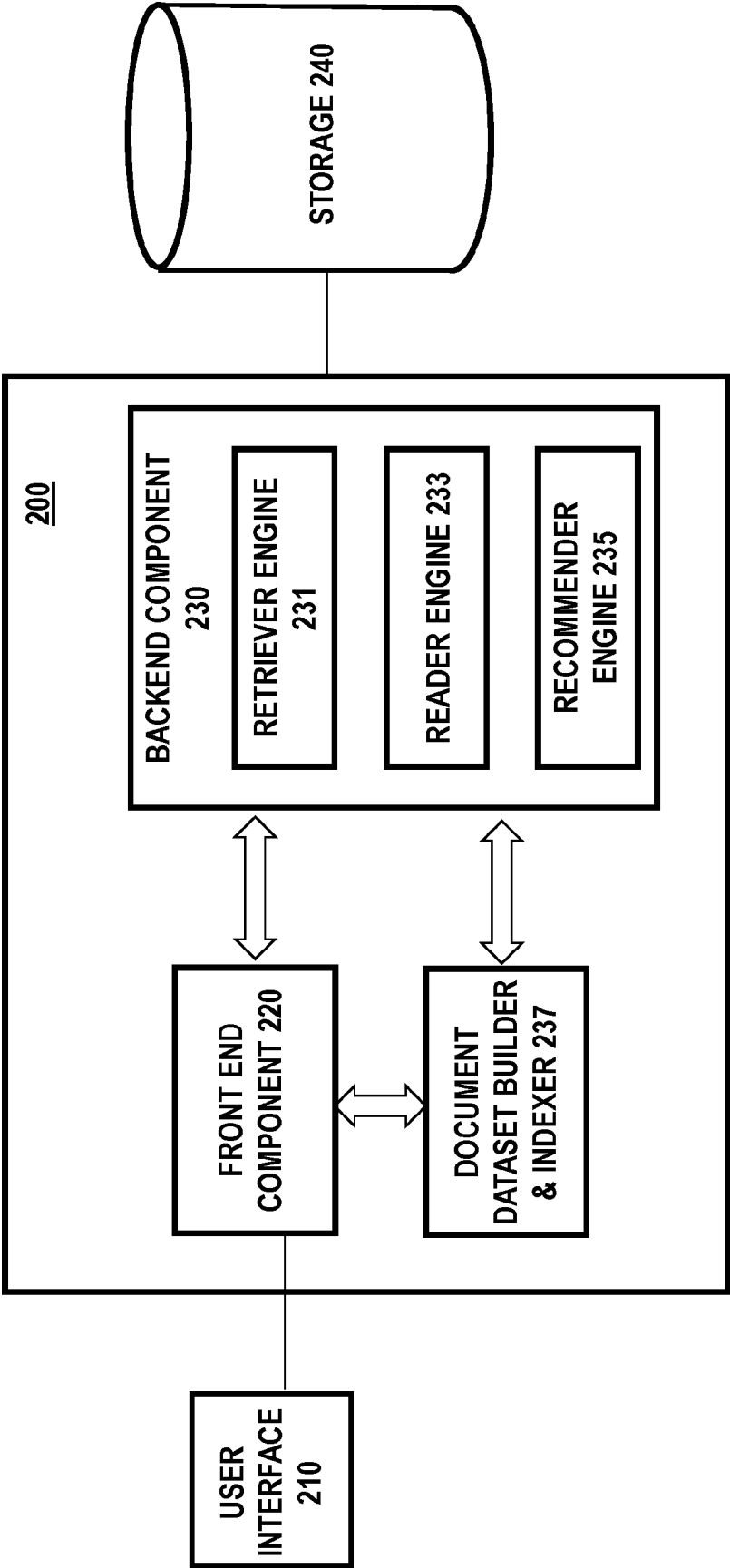


FIG. 2

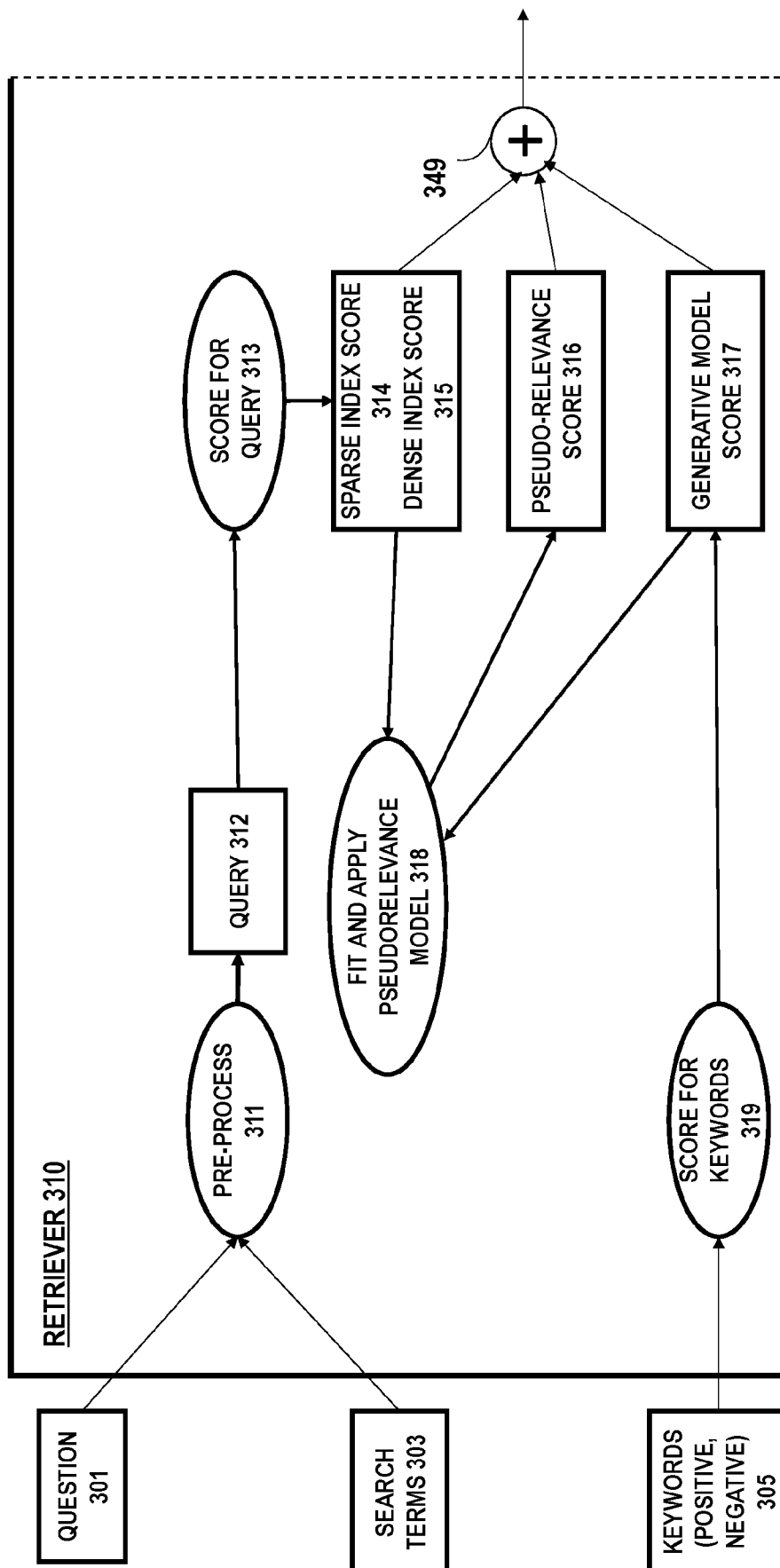
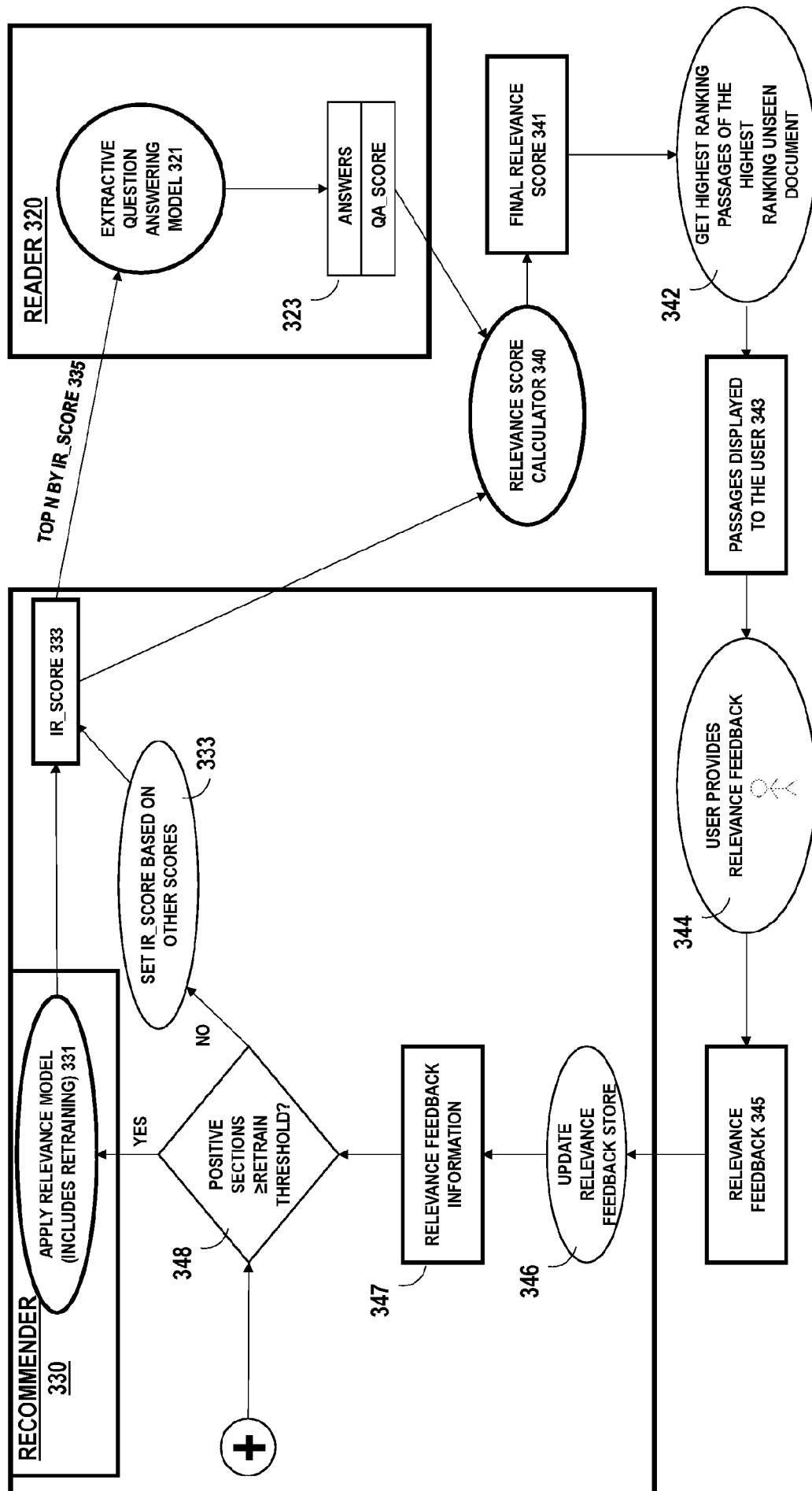


FIG. 3



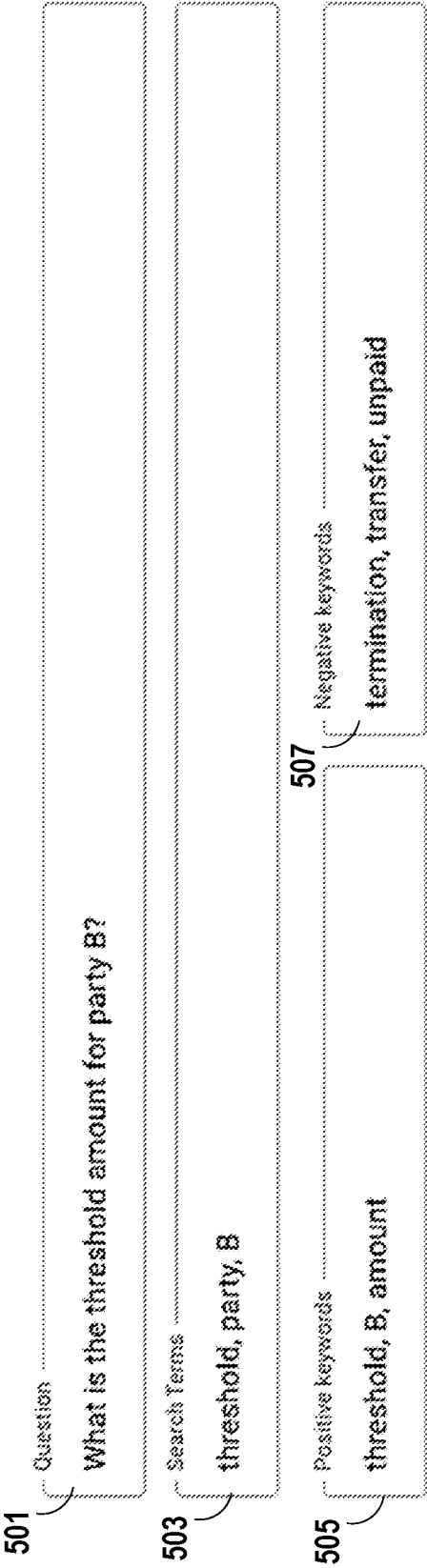
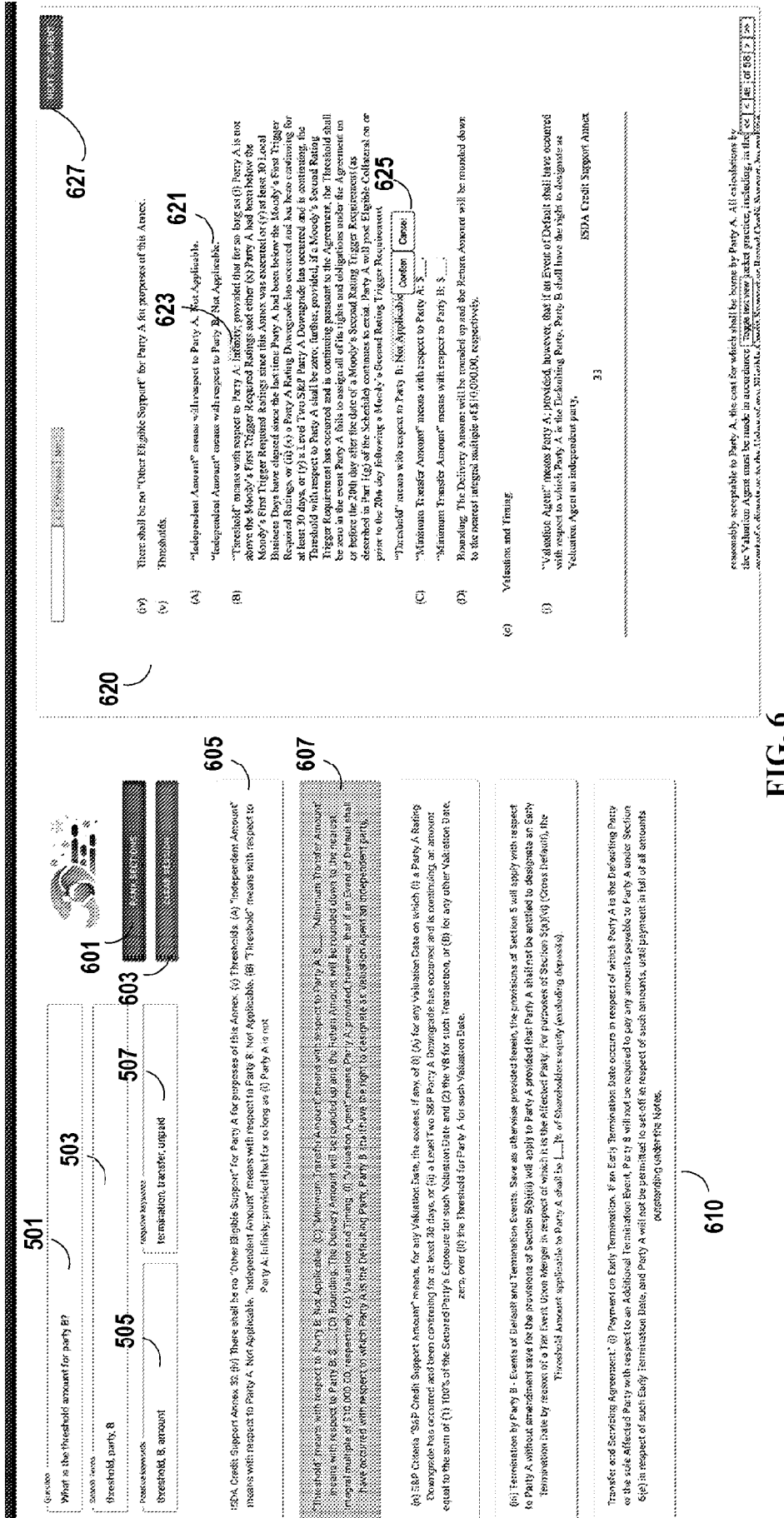


FIG. 5



- 701

or before the 20th day after the date of a Moody's Second Rating Trigger Requirement (as described in Part 1(g) of the Schedule) continues to exist. Party A will post Eligible Collateral on or prior to the 20th day following a Moody's Second Rating Trigger Requirement.

703
- (C)

“Threshold” means with respect to Party B: Not Applicable

705

Confirm

Cancel

“Minimum Transfer Amount” means with respect to Party A: \$ ____.

“Minimum Transfer Amount” means with respect to Party B: \$ ____.
- (D)

Rounding. The Delivery Amount will be rounded up and the Return Amount will be rounded down to the nearest integral multiple of \$10,000.00, respectively.

FIG. 7

PREFACE

EXPORT DATA

803

Citibank Europe PLC, Citigroup Financial Products Inc., Citigroup Energy Inc., Citibank Canada,
FgU Canada LLC, Citibank Japan Ltd., and Citigroup Global Markets Deutschland

AG & Co. KGaA

Section 5(a)(v), None Specified
Section 5(a)(vi), None Specified
Section 5(b)(v), None Specified

29

In relation to Party B for the purpose of:

Section 5(a)(v), Each Loan Party (as defined in the Credit Agreement)

Section 5(a)(vi), None Specified

Section 5(b)(v), None Specified

(b) "Specified Transaction" will have the meaning specified in Section 14 of this Agreement. For purposes of clause (c) of such definition, Specified Transaction includes any securities options, swaps, loans, short sales, any agreement governing the purchase, sale, transfer, exchange or option of a commodity, or any other contractually binding transaction and any other similar transaction now existing or hereafter entered into between Party A (or its Affiliate) and Party B (or its Affiliate of Party B). For this purpose, "commodity" means any tangible or intangible commodity of any type or description, including without limitation power, natural gas, petroleum (and its products and by-products thereof), emissions allowances, precious metals and steel.

(c) The "Cross Default" provisions of Section 5(a)(v) will apply to Party A and will apply to Party

B.

For purposes of Section 5(a)(v), the following provisions apply:

"Specified Indebtedness" shall have the meaning set forth in Section 14 of this Agreement; provided, however, that Specified Indebtedness shall not include deposits received at the close of a party's ordinary banking business.

"Threshold Amount" means

(i) with respect to Party A, 2% of the stockholders' equity of Party A, and

(ii) with respect to Party B, USD 25,000,000;

including the U.S. Dollar equivalent on the date of any default, event of default or other similar condition in even if any obligation stated in another currency.

For purposes of the above, stockholders' equity shall be determined by reference to the relevant party's most recent consolidated quarterly, in the case of a U.S.-incorporated party, balance sheet and shall exclude, in the case of a foreign-incorporated party, legal capital, paid-up

801

Question

Relevant term

Threshold, party B

Positive keywords

Negative keywords

existence of a party's ordinary banking business. "Threshold Amount" means (i) with respect to Party A, 2% of the stockholders' equity of Party A, and (ii) with respect to Party B, USD 25,000,000.

Party A is a Multinational Party and may enter into a Transaction through any of the following offices: New York, London, Singapore and Sydney. Party B is not a Multinational Party. 35 (e) Calculation Agent. The Calculation Agent is Party A, provided, however, that if at any time an Event of Default or Additional Termination Event has occurred with respect to Party A from the Calculation Agent shall be a party selected by Party B. The Alternative Calculation Agent will reasonably acceptable to Party A, with any costs arising from such selection being borne by Party B. The Alternative Calculation Agent will remain the Calculation Agent for so long as such event continues. If Party A cures the

event the Calculation Agent for so long as such event continues. If Party A cures the

bankruptcy of a company. (d) The "Credit Event Upon Merge" provision of Section 5(a)(v) of the Agreement will apply to Party A and will apply to Party B. 36 (e) The "Automatic Early Termination" provision of Section 6(a) will not apply to Party A and will not apply to Party B; provided, however, that with respect to a party, where the Event of Default specified in Section 5(a)(v)(C), (D), (E), (F), (G), (H) or (I) to the extent analogous therein, (B) is governed by a system of law which does not permit termination to take place after the occurrence of the relevant Event of Default, then the

operation of or act on its behalf. (f) (i) Cross Default. "Cross Default" is specified in the Schedule as applying to the party. The occurrence or existence of (i) a default, event of default or other similar condition or event (however described) in respect of such party, any Credit Support Provider of such party or any applicable Specified Entity of such party under one or more agreements or instruments relating to Specified Indebtedness of any of them (individually or collectively) where the aggregate principal amount of such agreements or instruments, either alone or together with the amount, if any, referred to in clause (2) below is not less than the applicable Threshold Amount (as specified in the Schedule), which has resulted in such Specified Indebtedness becoming or increasing payable at such time of being declared, due and payable under such agreements or

as possible to that of the threshold, legal or non-performance provisions. (g) Confirmation Procedures. For each Transaction that Party A and Party B enter into under the terms of the Agreement, Party A shall promptly send to Party B a confirmation setting forth the terms of such Transaction and Party B

FIG. 8

	What is the threshold amount for party B? Search Terms: Possible Keywords: Negative Keywords:
---	---

(iv) There shall be no "Other Eligible Support" for Party A for purposes of this Annex. (v) Thresholds.	"Independent Amount" means with respect to Party A: Not Applicable. "Interdependent Amount" means with respect to Party B: Not Applicable. (B) "Threshold" means with respect to Party A: Infinitely provided that for so long as (i) Party A is not above the Moody's First Trigger Required Ratings and either (x) Party A had been below the Moody's First Trigger Required Ratings since this Annex was executed or (y) at least 30 Local Business Days have elapsed since the last time Party A had been below the Moody's First Trigger Required Ratings; or (ii) (A) a Party A Rating Downgrade has occurred and has been continuing for at least 30 days; or (B) a Level Two S&P Party A Downgrade has occurred and is continuing; the Threshold with respect to Party A shall be zero; further, provided, if a Moody's Second Rating Trigger Requirement has occurred and is continuing pursuant to the Agreement, the Threshold shall be zero in the event Party A fails to assign all of its rights and obligations under the Agreement on or before the 20th day after the date of a Moody's Second Rating Trigger Requirement (as described in Part 1(g) of the Schedule); continues to exist; Party A will post Eligible Collateral on or prior to the 20th day following a Moody's Second Rating Trigger Requirement. "Threshold" means with respect to Party B: Not Applicable. (C) "Minimum Transfer Amount" means with respect to Party A: \$ ____ Confirm Cancel "Minimum Transfer Amount" means with respect to Party B: \$ ____ (D) Rounding. The Delivery Amount will be rounded up and the Return Amount will be rounded down to the nearest integer multiple of \$10,000.00, respectively.
---	---

(c) Valuation and Timing.	(e) "Valuation Agent" means Party A; provided, however, that if an Event of Default shall have occurred with respect to which Party A is the Defeating Party, Party B shall have the right to designate as Valuation Agent an independent party. 33 SDA Credit Support Annex
----------------------------------	--

FIG. 9

[illegible]

FIG. 10

INTERNATIONAL SEARCH REPORT

International application No
PCT/GB2023/051519

A. CLASSIFICATION OF SUBJECT MATTER

INV. G06F16/93 G06F16/332 G06F16/338
ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

G06F

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

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

EPO-Internal

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2018/365324 A1 (FINBERG NOAH [US] ET AL) 20 December 2018 (2018-12-20) paragraphs [0089], [0091], [0125], [0137], [0144], [0147]; claim 16; figures 2,18,25 -----	1-30

☐

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

12 July 2023

Date of mailing of the international search report

20/07/2023

Name and mailing address of the ISA/

European Patent Office, P.B. 5818 Patentlaan 2
NL - 2280 HV Rijswijk
Tel. (+31-70) 340-2040,
Fax: (+31-70) 340-3016

Authorized officer

Hackelbusch, Richard

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No

PCT/GB2023/051519

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2018365324	A1	20-12-2018	NONE
