August 2021

Method And System For Automatically Generating Artificial Intelligence Powered Prototypes

CAROLINA BARCENAS, Ms
Visa

HARISHKUMAR MAJITHIYA, Mr
Visa

SHIZHE MA,
Visa

PRASAD KUDTARKAR, Mr
Visa

SUGANDH SACHDEV, Mr
Visa

See next page for additional authors

Inventor(s)
CAROLINA BARCENAS, Ms; HARISHKUMAR MAJITHIYA, Mr; SHIZHE MA,; PRASAD KUDTARKAR, Mr; SUGANDH SACHDEV, Mr; and ROSHNI SAMUEL Ms

METHOD AND SYSTEM FOR AUTOMATICALLY GENERATING ARTIFICIAL INTELLIGENCE POWERED PROTOTYPES

VISA

INVENTORS:

CAROLINA BARCENAS, HARISHKUMAR MAJITHIYA, SHIZHE MA, PRASAD KUDTARKAR, SUGANDH SACHDEV, ROSHNI SAMUEL

TECHNICAL FIELD

[0001] This disclosure relates to a method and a system for automatically generating Artificial Intelligence (AI) powered prototypes.

BACKGROUND

[0002] AI research is a relatively new field with limited support for integration with traditional applications. Research findings are difficult to convey to an end-user of an application without a software solution. Generating end-to-end AI solutions involve iteration through an entire Software Development Life Cycle (SDLC) for each solution. This is time consuming and requires a high amount of computing resources.

[0003] To overcome the above-mentioned shortcomings, the present invention provides automatic generation of prototype applications using Al models.

BRIEF DESCRIPTION OF THE DRAWINGS

[0004] Additional advantages and details are explained in greater detail below with reference to the exemplary embodiments that are illustrated in the accompanying schematic figures, in which:

[0005] **Fig.1a** shows an exemplary architecture for automatically generating AI powered prototypes in accordance with some embodiments of the present disclosure.

[0006] **Figs.1b-1c** show process flow diagrams for automatically generating AI powered prototypes in accordance with some embodiments of the present disclosure.

[0007] **Fig.2a** shows a block diagram of an AI prototype generation system in accordance with some embodiments of the present disclosure.

[0008] **Fig.2b** shows a workflow of an AI prototype generation system in accordance with some embodiments of the present disclosure.

[0009] **Fig.2c** shows a plurality of components of an AI prototype generation system in accordance with some embodiments of the present disclosure.

[0010] **Fig.3** shows a block diagram illustrating interaction between a plurality of AI models in accordance with some embodiments of the present disclosure.

[0011] **Fig.4** shows a set of configurations for a model, data, and presentation, in accordance with embodiments of the invention.

[0012] **Fig.5** shows a block diagram of an exemplary process for automatically generating AI powered prototypes in accordance with some embodiments of the present disclosure.

[0013] **Fig.6** shows a block diagram of an exemplary computer system for automatically generating AI powered prototypes in accordance with some embodiments of the present disclosure.

DESCRIPTION OF THE DISCLOSURE

[0014] It is to be understood that the present disclosure may assume various alternative variations and step sequences, except where expressly specified to the contrary. It is also to be understood that the specific devices and processes illustrated in the attached drawings and described in the following specification are simply exemplary and non-limiting embodiments or aspects. Hence, specific dimensions and other physical characteristics related to the embodiments or aspects disclosed herein are not to be considered as limiting.

[0015] For purposes of the description hereinafter, the terms “end,” “upper,” “lower,” “right,” “left,” “vertical,” “horizontal,” “top,” “bottom,” “lateral,” “longitudinal,” and derivatives thereof shall relate to the disclosed subject matter as it is oriented in the drawing figures. However, it is to be understood that the disclosed subject matter may assume various alternative variations and step sequences, except where expressly specified to the contrary. It is also to be understood that the specific devices and processes illustrated in the attached drawings, and described in the following specification, are simply exemplary embodiments or aspects of the disclosed subject matter. Hence, specific dimensions and other physical characteristics related to the embodiments or aspects disclosed herein are not to be considered as limiting unless otherwise indicated.

[0016] No aspect, component, element, structure, act, step, function, instruction, and/or the like used herein should be construed as critical or essential unless explicitly described as such. Also, as used herein, the articles “a” and “an” are intended to include one or more items and may be used interchangeably with “one or more” and “at least one.” Furthermore, as used herein, the term “set” is intended to include one or more items (e.g., related items, unrelated items, a combination of related and unrelated items, and/or the like) and may be used interchangeably with “one or more” or “at least one.” Where only one item is intended, the term “one” or similar language is used. Also, as used herein, the terms “has,” “have,” “having,” or the like are intended to be open-ended terms. Further, the phrase “based on” is intended to mean “based at least partially on” unless explicitly stated otherwise.

[0017] It will be apparent that systems and/or methods, described herein, can be implemented in different forms of hardware, software, or a combination of hardware and software. The actual specialized control hardware or software code used to implement these systems and/or methods is not limiting of the implementations. Thus, the operation and behavior of the systems and/or methods are described herein without reference to specific software code, it being understood that software and hardware can be designed to implement the systems and/or methods based on the description herein.

[0018] Some non-limiting embodiments or aspects are described herein in connection with thresholds. As used herein, satisfying a threshold may refer to a value being greater than the threshold, more than the threshold, higher than the threshold, greater than or equal to the threshold, less than the threshold, fewer than the threshold, lower than the threshold, less than or equal to the threshold, equal to the threshold, etc.

[0019] The present invention discloses a method and a system for automatically generating AI powered prototypes. Here, Al inferencing methodologies are combined with dynamic configuration driven web applications that demonstrate a value of the Al model. The present invention provides a configuration driven framework for serving different Al models, which allows a user to view results of a model and/or a product. Additionally, a feedback mechanism may be used to gather information from the user and the results to further train a model. The present invention is particularly advantageous as it enables visualizations of data to be accessible through multiple mediums of communication, namely website and mobile applications. Also, time required for development process of Al prototypes is significantly reduced such as to an estimate of three days, using use-case driven scalability, configuration driven visualizations, and highly scalable databases for multiple-tenant support.

[0020] **Fig.1a** shows exemplary architecture for automatically generating AI powered prototypes in accordance with some embodiments of the present disclosure.

[0021] As shown in **Fig.1a**, the architecture 100 may include an AI prototype generation system 101 and a user 103. The AI prototype generation system 101 may be implemented on a personal computer, or a web server. The AI prototype generation system 101 may comprise one or more processors, a memory, a model store, a data store, a model configuration engine, a data configuration engine, an orchestration engine, and a user interface (not shown in figure). The model store may store a plurality of AI models 105. The data store may store training data for training one or more of the plurality of AI models 105. The model configuration engine may configure input features and data sources for training each of the plurality of AI models

105. The model configuration engine may also configure a cost function for each of the plurality of AI models 105, and aggregation of data sets to form the model inputs. Further, the data configuration engine may perform aggregation of model inputs and model outputs to form one or more searchable tables of sample data for each of the plurality of AI models 105. Particularly, the data configuration engine may perform joining of data sources to form the model inputs, and joining of output data from a first model and other data for visualization. The data configuration engine may also be configured to receive one or more categories of filtering criteria for populating the one or more searchable tables of sample data for at least one of the AI models. Further, the orchestration engine may configure an order of operation of the plurality of AI models 105, with an output of one or more first AI models 105 being used as an input for one or more second AI models 105. Further, the AI prototype generation system 101 may comprise a visualization engine (not shown in figure) for receiving one or more filtering criteria; and searching the one or more searchable tables of the sample data to provide a performance metric for the sample data corresponding to the one or more filtering criteria.

[0022] The one or more processors of the AI prototype generation system 101 may retrieve the plurality of AI models 105 from the model store and may provide the plurality of AI models 105 on the user interface. Further, the one or more processors may receive, via the user interface, a selection of a set of the AI models 105 from the user 103. Further, the one or more processors may receive, via the user interface, a first user input 107 for the model configuration engine. Particularly, the first user input 107 may include identification of the input features and the data sources for training each of the plurality of AI models 105 in the received set of the AI models 105. Further, the one or more processors may receive, via the user interface, receive a second user input 109 for the data configuration engine. Particularly, the second user input 109 may include a selection of one or more rules for aggregating model inputs and model outputs to form one or more searchable tables of sample data for at least one of the AI models 105. The one or more rules may include one or more categories of filtering criteria for populating the one or more searchable tables. In response to the user inputs, the one or more processors may identify the model inputs in the data sources as defined by the input features and defining a link between the model inputs and the AI model. Further, the one or more processors may create the one or more searchable tables having one or more columns corresponding to the one or more categories. Thereafter, the one or more processors may receive a third user input 111 for the orchestration engine, including an order of operation of the set of the AI models 105, with an output of one or more first AI models 105 being used as an input for one or more second AI models 105. Further, the one or more processors may generate an AI ensemble 113 that comprises the set of AI models 105. An interactive software service may be provided to allow the user 103 to specify segments of data via the one or more filtering criteria to visualize a performance of the AI ensemble 113 for a particular segment of sample data. Upon generating the AI ensemble 113, the one or more processors may operate on the model inputs using the AI ensemble 113 to obtain model outputs; and populate the one or more searchable tables according to the one or more rules.

[0023] **Fig. 1b** shows a process flow diagram for the generation of the prototype applications. The process may be initiated by accessing one or more data sources, which may be retrieved from an existing database or collected in real-time by one or more data collection devices. The one or more data sources may then be used, in conjunction with one or more Machine-Learning (ML) algorithms, to train an AI model. The ML algorithms may include supervised learning algorithms, unsupervised learning algorithms, and reinforcement learning algorithms. The ML algorithms may include, but not limited to, linear regression, logistic regression, decision tree, Support Vector Machine (SVM), Naive Bayes, k- Nearest Neighbors (kNN), K-Means, Random Forest, dimensionality reduction algorithms, and gradient boosting algorithms. The trained model may be deployed to the AI prototype generation system 101 that performs data classification based on model configurations set by the user 103. Model configurations may include model inputs, the one or more data sources used by the model, and settings for organizing data from the one or more data sources into one or more categories. The AI prototype generation system 101 may then output a visualization of data processed by the model. In some embodiments, the user 103 may select an option to only view visualization data from a particular category of the one or more categories. In some embodiments, one or more models may be used to generate the visualization data. **Fig. 1c** shows some key features of the AI prototype generation system 101, including an exemplary visualization of data and an exemplary means of grouping data such as aggregating into one or more categories.

[0024] **Fig.2a** shows a block diagram of an AI prototype generation system 101 in accordance with some embodiments of the present disclosure. In some implementations, AI prototype generation system 101 may include an I/O interface 201, at least a processor 202, a memory 203 and modules 204. The I/O interface 201 may be configured to receive one or more inputs from the user 103. As an example, the I/O interface 201 may receive input from the user 103 comprising a selection of a set of AI models from a plurality of AI models 105. As an example, the I/O interface 201 may receive input from the user 103 comprising input features and data sources for training each AI model in the set of AI models 105. As an example, the I/O interface 201 may receive input from the user 103 comprising a selection of one or more rules for aggregating model inputs and model outputs to form one or more searchable tables of sample data for at least one of the AI models 105 in the set of AI models 105. As an example, the I/O interface 201 may receive input from the user 103 comprising an order of operation of the set of AI models 105, with an output of one or more first AI models 105 being used as an input for one or more second AI models 105. Here, the I/O interface 201 may be a user interface. The at least processor 202 may be coupled with the I/O interface 201 to receive user inputs, execute commands, and process the input data with the help of the modules 204 and the memory 203 to automatically generate AI prototypes. The memory 203 may store data received through the I/O interface 201, modules 204, and the at least processor 202.

[0025] In some embodiments, the modules 204 may include, for example, an AI research module 205, a model deployment module 213, a management module 221, an execution module 231, a data generation module 243, a presentation module 245, and other modules 257. The other modules 257 may be used to perform various miscellaneous functionalities of the AI prototype generation system 101. It will be appreciated that aforementioned modules 204 may be represented as a single module or a combination of different modules. Furthermore, a person of ordinary skill in the art will appreciate that in an implementation, the one or more modules 204 may be stored in the memory 203, without limiting the scope of the disclosure. The said modules 204 when configured with the functionality defined in the present disclosure will result in a novel hardware.

[0026] In an embodiment, the AI research module 205 may comprise a data store 207, a plurality of AI models 209 and a plurality of AI algorithms 211. The data store 207 may store training data for training one or more of the plurality of AI models 209. The AI research module 205 may receive, via the I/O interface 201, a selection of the set of the AI models 209 from the plurality of AI models 209, which may be input to the model deployment module 213 for further processing.

[0027] In an embodiment, the model deployment module 213 may comprise a collection module 215, a feature preparation module 217, a scoring engine 219, and an aggregation module 221. The collection module 215 may collect data from external resources such as web resources and/or user input. The feature preparation module 217 may configure input features and data sources for training each of the plurality of AI models 209 in the set of the AI models. The aggregation module 221 may include one or more categories used to filter data elements in the collection of data. The aggregation module 221 may perform aggregation of model inputs and model outputs to form one or more searchable tables of sample data for each of the plurality of AI models.

[0028] In an embodiment, the management module 221 may comprise a model store 223, a configuration store 225, an algorithms manager 227, and a schema manager 229. The model store 223 may comprise a plurality of AI models 105. The configuration store 225 may comprise one or more model configurations for the plurality of AI models 209. The configuration store 225 may include input features and data sources for training each of the plurality of AI models 209. The configuration store 225 may also include one or more cost functions that can be used in training each of the plurality of AI models 209. Further, the algorithm manager may comprise one or more algorithms, and the schema manager 229 may comprise data schema. The management module 221 may be configured to receive and store one or more models, one or more algorithms, and data schema from a user 103 and/or a third-party computer. Further, the management module 221 may be configured to manage different models and algorithms generated as output from one or more sources such as research studies. The management module 221 may input data to the execution module 231 for further processing.

[0029] In an embodiment, the execution module 231 may comprise a workflow module 233, an automated AI solution module 235, an alerts module 237, a dynamic application creation module 239, and an AI visualization module 241. The dynamic application creation module 239 may create Application Programming Interface (API). The workflow module 233 and the automated AI solution module 235 may be configured to process input data from the management module 221 to generate output data, which may be input to the presentation module 245. Particularly, the execution module 231 may be configured to receive business logic, data, and user input which can be used as one or more additional inputs for the AI model. The execution module 231 may provide a dynamic workflow-based approach for an automatic execution of the models and algorithms and may further provide additional inputs to the execution of a model in the form of data, business logic, and/or user inputs. The execution module 231 may further be in communication with data generation module 243 and presentation module 245.

[0030] In an embodiment, the data generation module 243 may receive the results data from the execution module 231 and may generate static intelligence data and real-time intelligence data. In some embodiments, data generation module 243 may perform training of the plurality of AI models 209. The data generation module 243 may also be in communication with presentation module 245.

[0031] In an embodiment, the presentation module 245 may comprise a transformation module 247, a Representational State Transfer (REST) services module, one or more application interfaces 251, a case management module 253, and a notification module 255. The transformation module 247 may transform output data from the AI model to a graphical format, and/or a table format, for generating reports. The REST services module 249 may include REST services such as Java and Spring framework to be utilized for dynamically generating API endpoints. The case management module 253 may analyze data outputted by the plurality of artificial intelligence models to determine a possibility of fraud. The one or more application interfaces 251 may comprise a web application interface, or a mobile application interface for displaying data to the user 103. The presentation module 245 may be configured to enable access to the data of the data store 207 and/or the results data from the execution module 231. The presentation module 245 may provide access to the user 103 in the form of an application such as a mobile application.

[0032] In some embodiments, the displayed data may be data outputted by the AI model of the plurality of AI models 209 that has undergone one or more data processing techniques. In some embodiments, at least one data processing technique may include aggregating output data based on one or more categories and displaying only output data that corresponds to one of the one or more categories.

[0033] **Fig.2b** shows a workflow of an AI prototype generation system 101 in accordance with some embodiments of the present disclosure. The workflow may include a model development process operated by a modeler. Here, the modeler may generate the trained AI models 209 and corresponding model configurations and may be input to the execution module 231. Upon receiving the trained AI models 209, the execution module 231 may apply one or more configurations and business logic to the functionality of the trained AI models 209 for generating output data, which may be input to the presentation module 245. Thereafter, the presentation module 245 may process the output data to be displayed to the user 103.

[0034] In some embodiments, the model development process may include model development software and serving engines associated with ML software library, which may include, but not limited to, Tensorflow, Pytorch, Caffe, and Keras. Further, Big Data tools may be utilized for processing the data. The Big Data tools may include, but not limited to, Hadoop, Spark, Hive for Data Ingestion. Embodiments may also utilize NVIDIA drivers and tools.

[0035] **Fig.2c** shows a plurality of components of an AI prototype generation system 101 in accordance with some embodiments of the present disclosure. The AI prototype generation system 101 may further comprise a model configuration extractor 259, a model metadata file, a model definition classifier 261, an application component builder 263, a model features logic file, an application pipeline builder 265, a components repository, and a ML prototype builder 267.

[0036] Deployable model configuration like parameters and weights may be typically stored in a binary file. For automatically generating the AI powered protypes, it may become necessary that the system understands the structure of model configuration.

[0037] In an embodiment, the model configuration extractor 259 may provide algorithm to extract the model configuration for multiple libraries and model architectures. Utilizing the algorithm, extracted information may be converted into single uniform data structure which may be used to build the prototype component. The result from the algorithm may be fed to another algorithm for building an execution component, which may be input to the components repository. As shown in **Fig.2c,** the model configuration extractor 259 may comprise a metadata APIs wrapper, a model configuration parser, and an execution component builder.

[0038] After the model training process, the modeler may extract the model into library specific format. Some of the libraries may provide uniform APIs to read the files. As an example, Tensorflow may provide Tensorflow serving APIs to read model metadata. This sub-component may read configuration depending on the model meta data type utilizing APIs provided by the libraries. In an embodiment, custom code may be written to extract the information from the response. For the libraries not providing uniform APIs to read the files, a general-purpose code may be implemented in the metadata APIs wrapper for reading the format.

[0039] Once the model configuration has been read and available, the model configuration parser may read the information to build executables which may be put into the ML prototype 269. As an example, weights provided in the metadata may be fed into a specific Javascript Object Notion (JSON) format to be executed while developing the ML prototype 269.

[0040] Utilizing the data and code from the metadata APIs wrapper and the model configuration parser, the execution component builder may build model execution component which may be then fed into the component’s repository. The ID for the components may be created utilizing the model and project configurations.

[0041] In the AI prototype generation system 101, the metadata file may store information extracted by the model configuration extractor 259 and may make it available for other system components to consume. The metadata file may be a key entry point for building other components that helps with faster prototyping process.

[0042] Executing the model in a prototype environment may require understanding the schema, which is taken care by the model configuration extractor 259, and semantics of the model. In an embodiment, the model definition classifier 261 may help building a file that contains model features logic and may develop an application by generating both backend components and front-end components required to build the model features. The model definition classifier 261 may comprise an automated feature classifier, one or more training code parsing algorithms, a manual feature classifier, a classifier web interface, an input components builder, and a feedback components builder.

[0043] In an embodiment, the automated feature classifier may read the model metadata file and may automatically classify features of the model into the categories. the automated feature classifier may help identifying categorical features versus calculated features, uncover patterns like one-hot encoding. For classifying the features, the automated feature classifier may receive sample input data files. The automated feature classifier may significantly reduce development time for generating the AI powered prototypes. The automated feature classifier may build data schema that can be fed into the I/O interface 201. In an alternative embodiment, the use of the automated feature classifier may be optional for developing the ML prototypes 269.

[0044] In an embodiment, the training code parsing algorithm may help read the semantic of the features. Particularly, the training code parsing algorithm may provide more details about feature semantic, may confirm understanding of the automated feature classifier, and may provide schema which may be fed into the user interface.

[0045] In an embodiment, the manual feature classifier may classify the features which cannot be classified using the automated classification and training code parsing algorithms. A JSON template may be utilized in the manual feature classifier for classifying the features.

[0046] In an embodiment, the classifier web interface may display outcome from the automated feature classifier and the training code parsing algorithm. The classifier web interface may allow modelers to review decisions inferred from the algorithms and make modifications accordingly. The classifier web interface may also provide the user interface to configure the manual feature classifier.

[0047] After features are identified and classified, user inputs may be necessary for executing the models, which may require building UI/backend components. In an embodiment, the input components builder may utilize model metadata and classifier outputs for building the UI/backend components. The input components builder may also generate output which may be fed into other components such as the automated feature generators, the pipeline builders and the model execution components.

[0048] For continuously improving the models, classification of the outcome from the models' feedback may be required from the users 103. In an embodiment, the feedback components builder may build UI/backend components utilizing business SME knowledge, configurations and the feature classification results. The feedback components builder may build UI/backend components to capture the feedback.

[0049] In an embodiment, the model features logic file may store output from the model definition classifier 261 in a unified data structure. The model features logic file may be in JSON format.

[0050] In an embodiment, the application component builder 263 may build the final application components after model feature components are ready and application features are available. The final application components may be utilized into the prototypes which may be visible to the end users 103. The application component builder 263 may comprise a REST API generator, a custom API container, a UI components builder, a custom UI components container, a portal schema builder, and a collaboration component builder.

[0051] In an embodiment, the REST API generator may create REST services using model metadata, features file, data schema generator outputs and the business SME configuration.

[0052] In an embodiment, the custom API container may facilitate developers to push custom APIs to meet the specific business needs.

[0053] In an embodiment, the UI components builder may create UI Components using model metadata, features file, data schema generator outputs and the business SME configuration.

[0054] In an embodiment, the custom UI components container may facilitate developers to push custom UI components to meet the specific business needs.

[0055] In an embodiment, the portal schema builder may generate schema for the final portal that gets delivered to the user 103. Particularly, portal schema builder may generate data specific schema as well as configuration schema for all the components to work together.

[0056] In an embodiment, the collaboration component builder may generate components which may be pushed to the components repository, for facilitating the multiple users 103 to collaborate for consuming model results and make business decisions.

[0057] After model features are defined and configuration files are created, building the pipeline may be essential. Further, the pipeline needs to be aligned to business objectives. In an embodiment, the application pipeline builder 265 may help building all the components that may fulfill the business objectives. The objectives may be around execution frequency, visualization requirements or data access requirements. Additionally, the application pipeline builder 265 may help in generating the features using uniform data structures. The application pipeline builder 265 may comprise an automated feature generator, a data ingestion tracker, a model results formatter, a data schema generator, a pipeline components builder, and an Extract-Transform-Load (ETL) components builder.

[0058] The algorithm may read model features logic file and given input data storage locations may generate features required for the models to execute. Although, the automated feature generator may not be aware of scheduling needs but may be executed in the sequence such that scheduling is not broken. Further, output from the automated feature generator may be input into the data schema generator and the model execution components, for maintaining uniform structure across the model execution components.

[0059] The models may require data to be ingested from multiple data sources. In an embodiment, the data ingestion tracker may keep track of all the data sources coming into the system and their processing status. This may facilitate the users 103 of the prototype to understand the processing scenario and evaluate the results from the model accordingly.

[0060] In an embodiment, the model results formatter may format the output from the models such that visualization and REST APIs can consume the output. For traditional products, the output from the models may be well defined and documented. However, for ML prototypes 269, the output from the models may be dynamic and may change as the features are added while developing the prototypes. The model results formatter may format model results with a dynamic structure, which may be fed into UI/API components and may explain the behavior of the model.

[0061] In an embodiment, the data schema generator may perform transformations of the output/scores into required format, data aggregation, optimal storage strategies for faster API response time, and integration with real world other data sources that aids value to model results investigation and business decision making. The data schema generator may contribute towards creating data structure for prototype screens and operating on data elements produced by the models.

[0062] In an embodiment, the pipeline components builder may utilize the results from the previous components and may create components that may be fed to the components repository. These blocks of components may be utilized for development of prototype and integrated with real data sources. This may be utilized to perform pre-processing of the data to create the tailored components.

[0063] Development of prototypes may require using SQL, NoSQL and Data lake systems, and different components may be hosted in their relevant environments to serve the business needs of evaluating the model and making business critical decisions. In an embodiment, the ETL components builder may build components to extract, transform, load the data from one system to another with the knowledge of metadata, data schema and feature logic file from previous steps.

[0064] In an embodiment, the components repository may store different type of components such as REST APIs, UIs, model execution, application pipeline generated by various modules of the AI prototype generation system 101. The aforesaid components may be stored in configuration files and/or DB tables for dynamically generating the AI powered prototypes.

[0065] In an embodiment, the ML prototype builder 267 may generate the ML prototypes 269. Here, all the components may be connected to each other to come up with deployable product. The ML prototype builder 267 may combine multiple models in to one entity named as "Project", in the prototype generation system. The ML prototype builder 267 may comprise a dynamic artifacts generator, a ML execution scheduler, a components organizer, a smart alerting system, an authorization workflow builder, and a configuration user interface.

[0066] In an embodiment, the dynamic artifacts generator may build web application artifacts and mobile application artifacts. The artifacts may be stored in configuration files and DB tables.

[0067] In an embodiment, the ML execution scheduler may schedule execution of application pipeline and may integrate with web and mobile application backends.

[0068] In an embodiment, the components organizer may organize components into correct sequence and layout in mobile and web applications. The components organizer may organize the ML execution components to execute in correct sequence to get the correct pipeline for business decision making.

[0069] In an embodiment, the smart alerting system may combine output from the multiple models into single alert. This may facilitate users 103 to configure alerts using output from several models.

[0070] In an embodiment, the authorization workflow builder may facilitate building authorization workflow for the prototype to restrict usage of the data by model users 103 and allows to control project level access.

[0071] In an embodiment, the configuration user interface may consist of user interface that may be utilized by data scientist, engineers and business users 103 to configure different components and multiple models in to single "Project" entity.

[0072] In an embodiment, the ML prototypes 269 may be generated from the output of the previous modules or components. The ML prototypes 269 may be a ready to deploy system that may allow users 103 to interact with the model, consume results from the model, give feedback on the model, use it for marketing purposes and extract the components for building proprietary products. The ML prototypes 269 may include, but not limited to, deployable mobile applications and deployable web portals.

[0073] **Fig.3** shows a block diagram illustrating interaction between a plurality of AI models 209 in accordance with some embodiments of the present disclosure. Particularly, an interaction between a fraud model, a card testing model, and a breach detection model is illustrated. Each model may include a separate set of model specifications and configurations. Each model may also be trained using a separate training data set derived from a same collection of data. Each model may be configured with different model inputs, data sources, data schema, and scoring techniques within the AI prototype generation system 101. For each model, the AI prototype generation system 101 may output presentation layers. In some embodiments, the output of one model may be used as a model input to another model, in a cascaded manner.

[0074] **Fig.4** shows a set of model configurations, data configurations, and presentation configurations, in accordance with embodiments of the invention. The model configurations may include model inputs, model aggregates, and data sources and schema. The model aggregates may be one or more categories used to filter data. For example, a possible model aggregate may be identifying card-not-present transactions and card-present transactions in separate categories.

[0075] The data configurations may include output aggregated views and joins. The output aggregated views may be one or more settings for configuring a view for a user 103, such that only data from one or more categories is included. In some embodiments, the data configurations may be used to process data at a merchant-level, an issuer-level, and/or an acquirer level. Further, joins may include settings for including one or more other data stores and/or output data from another model.

[0076] The presentation configurations may include security configurations, visualization configurations, feedback configurations, REST configurations and mobile configurations. The security configurations may be settings to identify users 103 who can access data and/or differentiates data into one or more security levels such as high-security, and low-security, such that the user 103 may need to provide authentication values and/or a pass code to access certain security levels. The visualization configurations may include a specified format such as in a graph or a table format for displaying the data to the user 103. The feedback configurations may include configurations for determining an accuracy of output data. The REST configurations may include settings for utilizing one or more REST APIs. The mobile configurations may include specifications for a mobile application used for displaying the data to the user 103.

[0077] As an example, an acquirer system may utilize an AI prototype to view a list of breached merchants ranked by a priority using a breach detection model. The acquirer system may select first visualization configurations to display a histogram explaining trends for each merchant. The acquirer system may then select a second visualization configurations to see if there is any pattern within merchant-to-merchant interactions. The acquirer system may then select a plurality of merchants flagged by the second visualization configuration and may determine if the selected plurality of merchants is additionally flagged by a card testing model. In some embodiments, the determination may involve inputting the selected plurality of merchants into the card testing model. Alternate visualization configurations may be set, such that the acquirer system can be alerted when a new merchant is detected. The acquirer system may also be provided with a Return On Investment (ROI) score after implementation.

[0078] In another example, an issuer system may choose to utilize an AI prototype to determine a list of Bank Identification Numbers (BINs) that may have been impacted using the card testing model. The issuer system may select a first visualization configuration that displays a list of impacted BINs or a list of impacted channels. The issuer system may also select a second visualization configuration that displays a list of cards impacted due to breached merchants by transmitting the impacted BINs to the breach detection model. The issuer system may additionally select a third visualization configuration for displaying an average fraud score and/or may determine if the impacted cards/channels correlate to a network. Alternate visualization configurations may be set, such that the issuer system may be alerted for card-testing events. The issuer system may also be provided an ROI score after implementation of the prototype.

[0079] **Fig.5** shows a block diagram of an exemplary process for automatically generating AI powered prototypes in accordance with some embodiments of the present disclosure.

[0080] In some embodiments, a plurality of AI models 105 may be provided to a user 103 via a user interface of an AI prototype generation system 101.

[0081] The user 103 may input a first selection for a set of AI models from the provided plurality of AI models 105.

[0082] For each of the set of the AI models, the user 103 may initiate a configuration process by inputting a second selection of identification of input features and data sources for training each of the plurality of AI models 105.

[0083] Further, the user 103 may input a third selection of one or more rules for aggregating model inputs and model outputs to form one or more searchable tables of sample data for at least one of the artificial intelligence models. The rules may include one or more categories of filtering criteria for populating the one or more searchable tables. In some embodiments, the filtering criteria may include a transaction type, a transaction amount range, a transaction risk score. For example, the one or more categories of filtering criteria may be used to differentiate card-not-present transactions from card-present transactions.

[0084] Responsive to the user interface receiving user input, the model inputs in the data sources may be identified as defined by the input features and a link may be defined between the model inputs and the AI model. The link may identify labels used for the input data or a particular section of memory 203, or API calls to retrieve the data. The transfer of data from one model to another model may be accomplished via API calls or by specifying a particular memory 203 location from which the second model can read, e.g., after a trigger command has been sent specifying new data is available.

[0085] Furthermore, one or more searchable tables may be generated, such that each searchable table includes one or more columns corresponding to the one or more categories. The one or more searchable tables may be populated with the model inputs according to the one or more rules.

[0086] The specification of the columns may enable certain data to be populated into the tables for quick analysis in a dashboard. For example, different parameters of a model such as a threshold, may be changed, and new graphs may be presented for the user 103 to determine an optimal threshold. As an example, a certain segment of input or output data may be filtered to identify a performance of the model for a given segment. This may be helpful during product runs, such as for an end user 103, who can review the relative performance and possibly make a final determination for an output label such as a classification. This may be accomplished when the AI model provides a suggested output label. However, if the probability of the output label is not very high, then the sample such as transaction or event may be sent to a user 103 for a final determination.

[0087] The aggregation for inputs and outputs may provide row entries for each sample. Further, statistical properties of the data may be pre-computed according to the configuration.

[0088] The selected models may be trained sequentially or in an iterative fashion. As an example, output of one model may serve as input for another model. As an example, an intermediate version of a model may be used to provide its output as an input to another model, which may then use the data for training. Even though the first model is not completely trained, the results obtained from the first model may be used to partially train the second model.

[0089] The order of the models may take various forms. In an embodiment, an order of operations for the set of AI models may be determined such that an output of one or more first AI models may be used as an input for one or more second AI models. In other words, output of a preceding model may serve as input of a succeeding model. As an example, a first model may provide an output which may serve as an input to a second model. Further, output of the second model may be input to a third model. In alternative embodiment, output of a model may be provided to a plurality of models. As an example, the output of the first model may be input to the second model and the third model.

[0090] Further, an AI ensemble 113 may be generated. The AI ensemble 113 may comprise the set of artificial intelligence models, which may be represented by the prototype (MVP). The AI ensemble 113 may then present desired data 115 to the user 103 based on at least the aggregations.

[0091] **Fig.6** shows a block diagram of an exemplary computer system 600 for automatically generating AI powered prototypes in accordance with some embodiments of the present disclosure. Any of the computer systems mentioned herein may utilize any suitable number of subsystems. Examples of such subsystems are illustrated in **Fig.6** in a computer system 600. In some embodiments, the computer system 600 may include a single computer apparatus, where the subsystems may be components of the computer apparatus. In other embodiments, the computer system 600 may include multiple computer apparatuses, each being a subsystem, with internal components. The computer system 600 may include desktop and laptop computers, tablets, mobile phones and other mobile devices.

[0092] The subsystems shown in **Fig.6** may be interconnected via a system bus. Additional subsystems such as a printer 607, a keyboard 617, a storage device(s) 619, a monitor 613 or a display screen, such as an LED, which may be coupled to a display adapter 623, and others are shown. Peripherals and Input/Output (I/O) devices, which may be coupled to an I/O controller 601, may be connected to the computer system 600 by any number of means known in the art such as I/O port 615 such as USB, FireWire®. For example, the I/O port 615 or external interface 621 such as Ethernet, and Wi-Fi may be used to connect the computer system 600 to a Wide Area Network (WAN) such as the Internet, a mouse input device, or a scanner. The interconnection via the system bus 611 may allow a central processor 605 to communicate with each subsystem and to control the execution of a plurality of instructions from system memory 603 or the storage device(s) 619 or a fixed disk, such as a hard drive, or optical disk, as well as the exchange of information between subsystems. The system memory 603 and/or the storage device(s) 619 may embody a computer readable medium. Another subsystem may be a data collection device 609, such as a camera, microphone, accelerometer, and the like. Any of the data mentioned herein can be output from one component to another component and can be output to the user 103.

[0093] The computer system 600 may include a plurality of the same components or subsystems, such as connected together by external interface 621, by an internal interface, or via removable storage devices that can be connected and removed from one component to another component. In some embodiments, computer systems, subsystem, or apparatuses may communicate over a network. In such instances, one computer may be considered a client and another computer a server, where each may be part of a same computer system 600. Each of the client and the server may include multiple systems, subsystems, or components.

[0094] Aspects of embodiments may be implemented in the form of control logic using hardware circuitry such as an Application Specific Integrated Circuit (ASIC) or Field Programmable Gate Array (FPGA) and/or using computer software with a generally programmable processor in a modular or integrated manner. As used herein, a processor may include a single-core processor, multi-core processor on a same integrated chip, or multiple processing units on a single circuit board or networked, as well as dedicated hardware. Based on the disclosure and teachings provided herein, a person of ordinary skill in the art will know and appreciate other ways and/or methods to implement embodiments of the present disclosure using hardware and a combination of hardware and software.

[0095] Any of the software components or functions described in this application, may be implemented as software code to be executed by a processor using any suitable computer language such as, for example, Java, C++ or Perl using, for example, conventional or object-oriented techniques. The software code may be stored as a series of instructions, or commands on a computer readable medium, such as a Random-Access Memory (RAM), a read only memory (ROM), a magnetic medium such as a hard-drive or a floppy disk, or an optical medium such as a CD-ROM. Any such computer readable medium may reside on or within a single computational apparatus and may be present on or within different computational apparatuses within a system or network.

[0096] The illustrated steps are set out to explain the exemplary embodiments shown, and it should be anticipated that ongoing technological development will change the manner in which particular functions are performed. These examples are presented herein for purposes of illustration, and not limitation. Further, the boundaries of the functional building blocks have been arbitrarily defined herein for the convenience of the description. Alternative boundaries can be defined so long as the specified functions and relationships thereof are appropriately performed. Alternatives (including equivalents, extensions, variations, deviations, etc., of those described herein) will be apparent to persons skilled in the relevant art(s) based on the teachings contained herein. Such alternatives fall within the scope and spirit of the disclosed embodiments. Also, the words "comprising," "having," "containing," and "including," and other similar forms are intended to be equivalent in meaning and be open ended in that an item or items following any one of these words is not meant to be an exhaustive listing of such item or items, or meant to be limited to only the listed item or items. It must also be noted that as used herein, the singular forms “a,” “an,” and “the” include plural references unless the context clearly dictates otherwise.

[0097] Finally, the language used in the specification has been principally selected for readability and instructional purposes, and it may not have been selected to delineate or circumscribe the inventive subject matter. Accordingly, the disclosure of the embodiments of the disclosure is intended to be illustrative, but not limiting, of the scope of the disclosure.

METHOD AND SYSTEM FOR AUTOMATICALLY GENERATING ARTIFICIAL INTELLIGENCE POWERED PROTOTYPES

ABSTRACT

The present disclosure relates to a method and a system for automatically generating Artificial Intelligence (AI) powered prototypes. Here, a plurality of AI models is provided in a user interface of the system. Further, a selection of a set of the AI models is received via the user interface. For each of the set of the AI models, identification of input features and data sources is received via the user interface for training each of the plurality of AI models. Further, a selection of one or more rules for aggregating model inputs and model outputs is received via the user interface to form one or more searchable tables of sample data for at least one of the artificial intelligence models. The one or more rules include one or more categories of filtering criteria for populating the one or more searchable tables. In response to the user input, the model inputs in the data sources as defined by the input features are identified and a link between the model inputs and the artificial intelligence model is defined. Further, the one or more searchable tables having one or more columns corresponding to the one or more categories are created. Further, an order of operation of the set of artificial intelligence models is received via the user interface. In the order of operation, an output of one or more first AI models is used as an input for one or more second AI models. Thereafter, an AI ensemble that comprises the set of AI models is generated.

Fig.1a

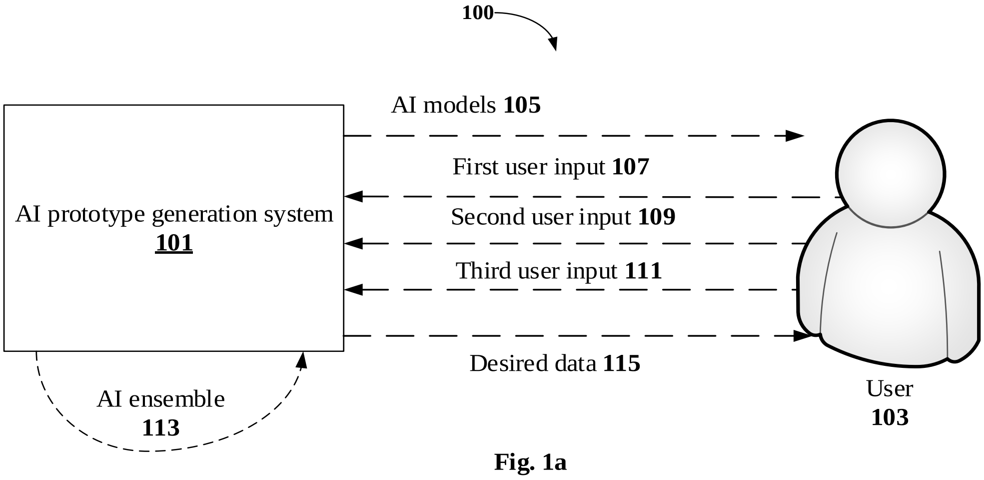


Fig. 1a

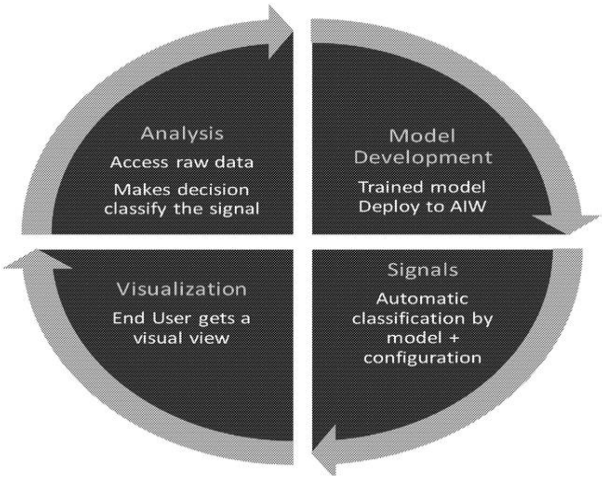


Fig. 1b

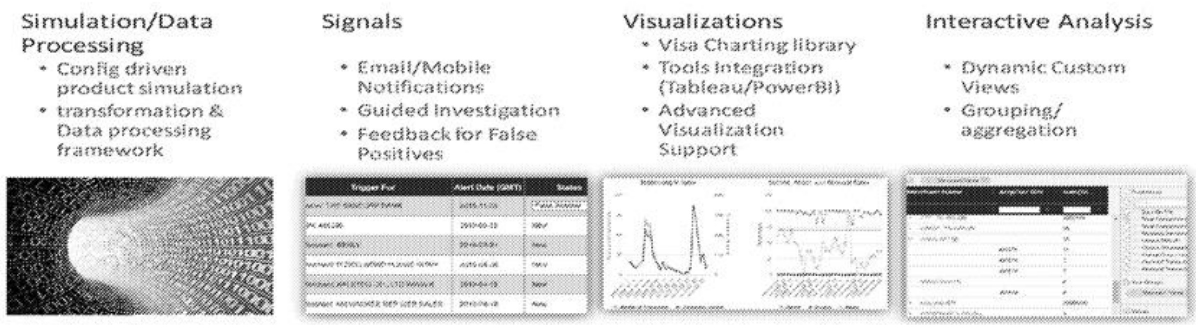


Fig. 1c

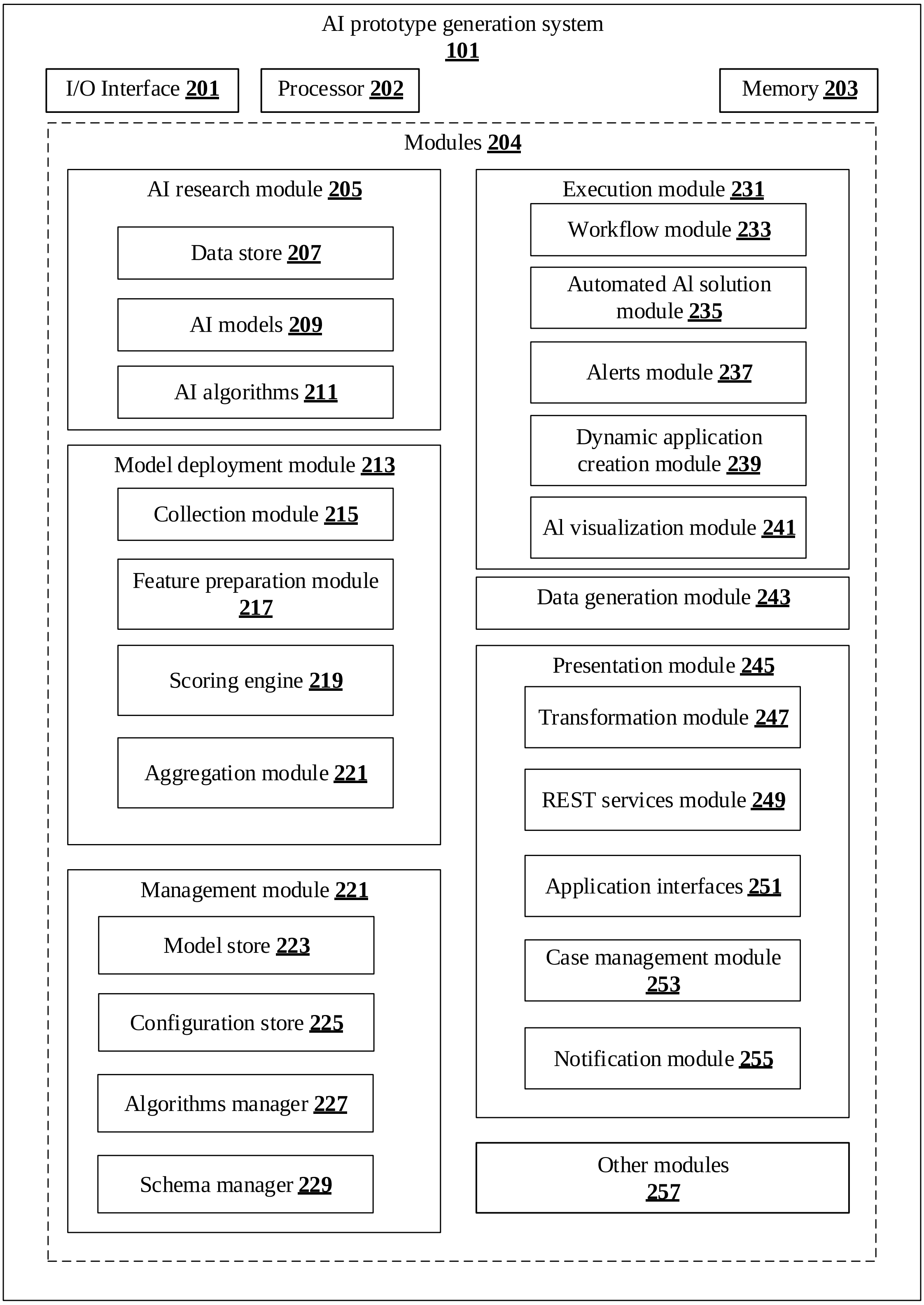

AI prototype generation system
101
I/O Interface 201
Processor 202
Memory 203
Modules 204
AI research module 205
Data store 207
AI models 209
AI algorithms 211
Model deployment module 213
Collection module 215
Feature preparation module 217
Scoring engine 219
Aggregation module 221
Management module 221
Model store 223
Configuration store 225
Algorithms manager 227
Schema manager 229
Execution module 231
Workflow module 233
Automated AI solution module 235
Alerts module 237
Dynamic application creation module 239
AI visualization module 241
Data generation module 243
Presentation module 245
Transformation module 247
REST services module 249
Application interfaces 251
Case management module 253
Notification module 255
Other modules 257

Fig. 2a

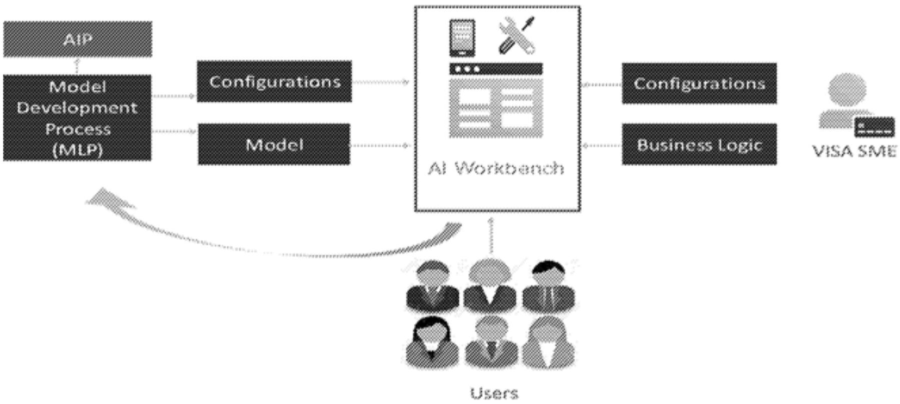
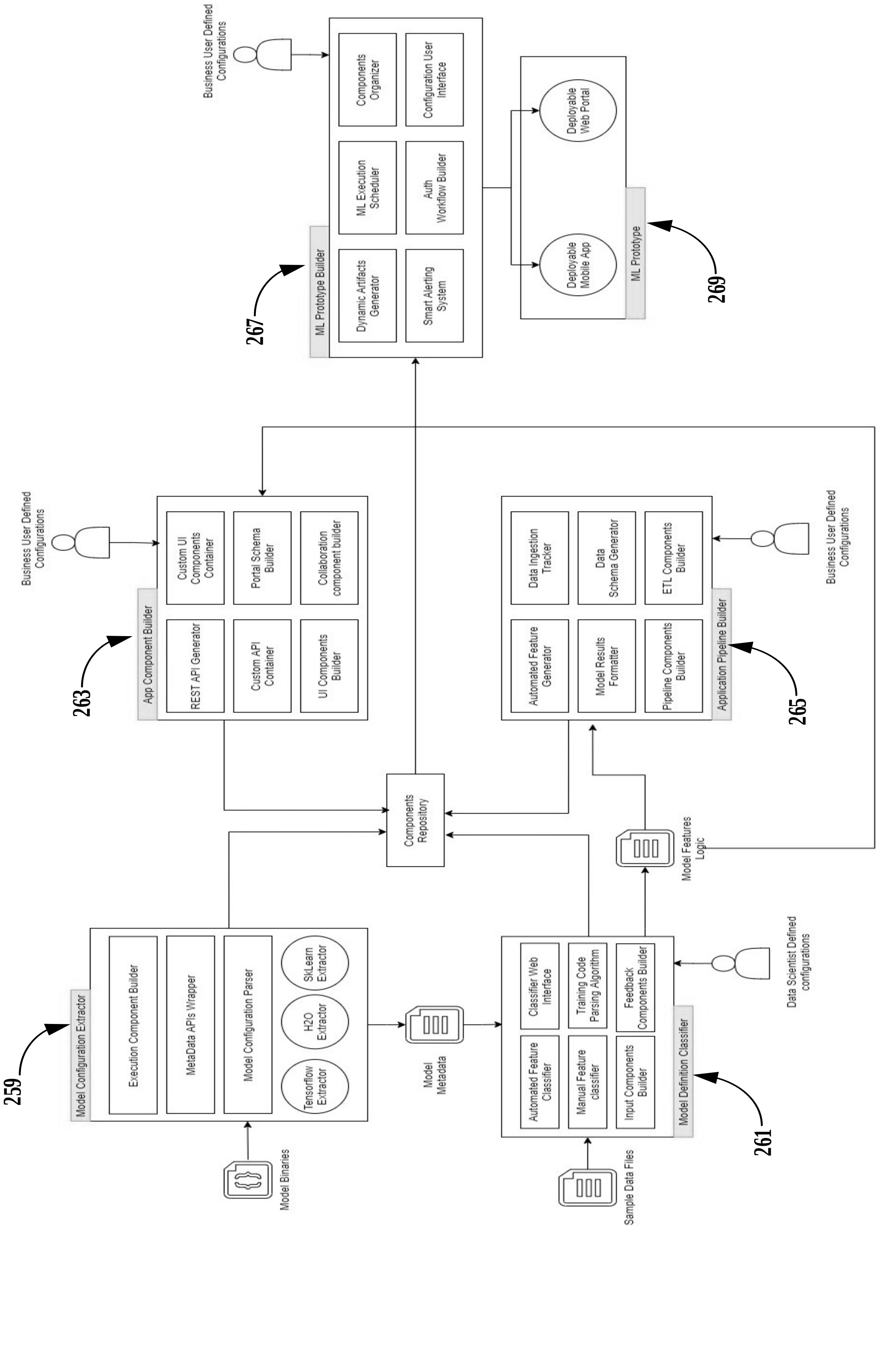

AIP
Model Development Process (MLP)
Configurations
Model
AI Workbench
Configurations
Business Logic
VISA SME
Users

Fig. 2b



259
Model Configuration Extractor
Execution Component Builder
MetaData APIs Wrapper
Model Configuration Parser
Tensorflow Extractor
H2O Extractor
SkLearn Extractor
Model Binaries
Model Metadata
261
Model Definition Classifier
Automated Feature Classifier
Manual Feature classifier
Input Components Builder
Classifier Web Interface
Training Code Parsing Algorithm
Feedback Components Builder
Sample Data Files
Data Scientist Defined configurations
Model Features Logic
Components Repository
263
App Component Builder
REST API Generator
Custom API Container
UI Components Builder
Custom UI Components Container
Portal Schema Builder
Collaboration component builder
Business User Defined Configurations
265
Application Pipeline Builder
Automated Feature Generator
Model Results Formatter
Pipeline Components Builder
Data Ingestion Tracker
Data Schema Generator
ETL Components Builder
Business User Defined Configurations
267
ML Prototype Builder
Dynamic Artifacts Generator
Smart Alerting System
ML Execution Scheduler
Auth Workflow Builder
Components Organizer
Configuration User Interface
Business User Defined Configurations
269
ML Prototype
Deployable Mobile App
Deployable Web Portal

Fig. 2c

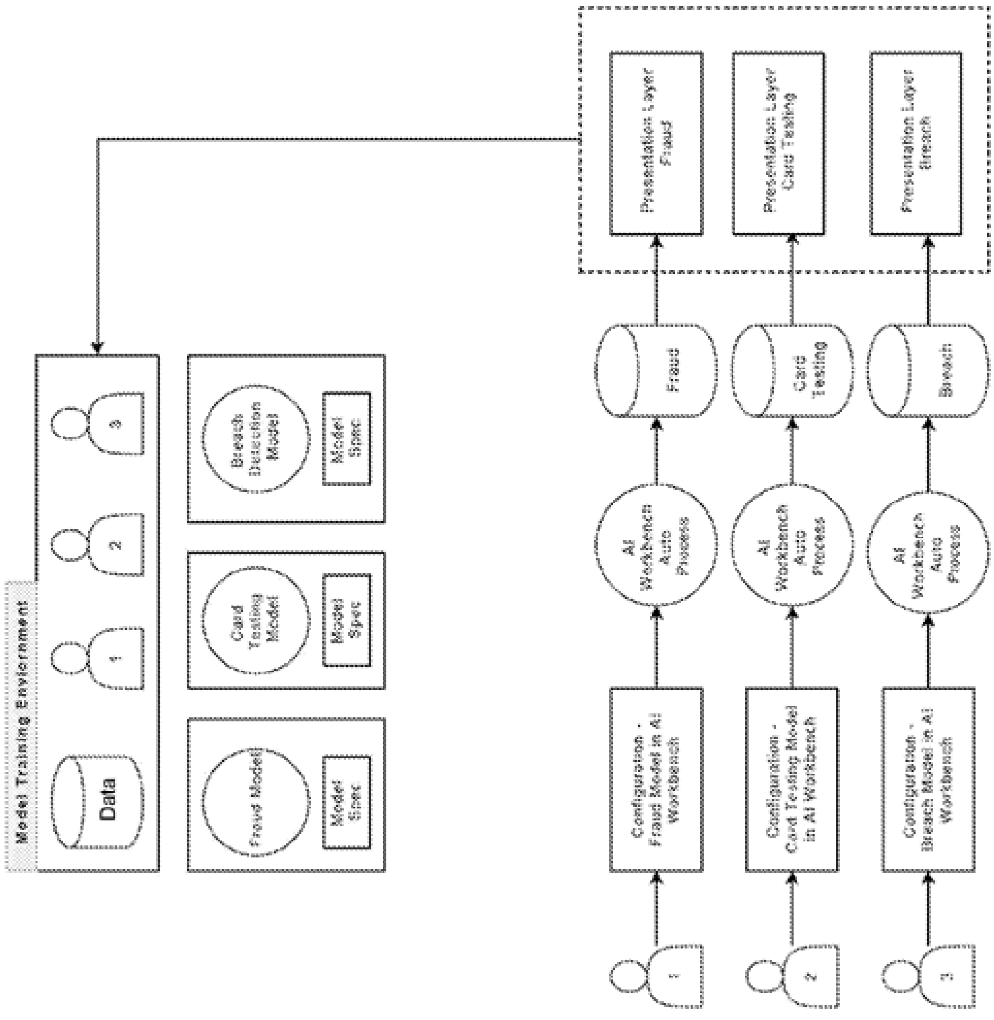

Model Training Enviornment
Data
1
2
3
Fraud Model
Model Spec
Card Testing Model
Model Spec
Breach Detection Model
Model Spec
Configuration - Fraud Model in AI Workbench
Configuration - Card Testing Model in AI Workbench
Configuration - Breach Model in AI Workbench
AI Workbench Auto Process
AI Workbench Auto Process
AI Workbench Auto Process
Fraud
Card Testing
Breach
Presentation Layer Fraud
Presentation Layer Card Testing
Presentation Layer Breach

Fig. 3

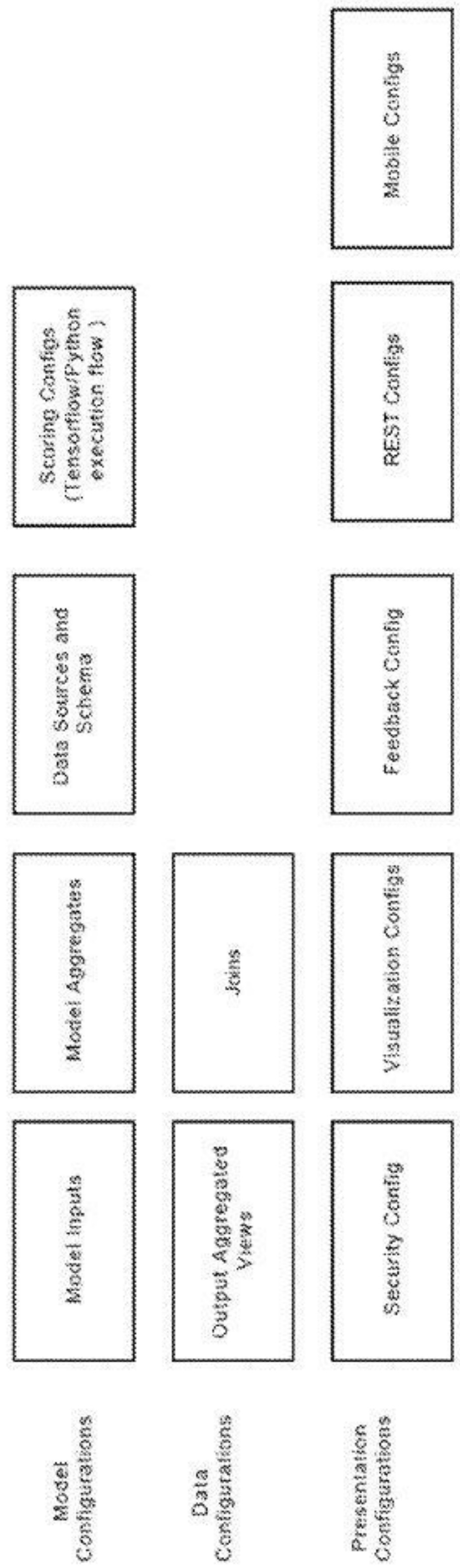

Model Configurations
Model Inputs
Model Aggregates
Data Sources and Schema
Scoring Configs (Tensorflow/Python execution flow)
Data Configurations
Output Aggregated Views
Joins
Presentation Configurations
Security Config
Visualization Configs
Feedback Config
REST Configs
Mobile Configs

Fig. 4

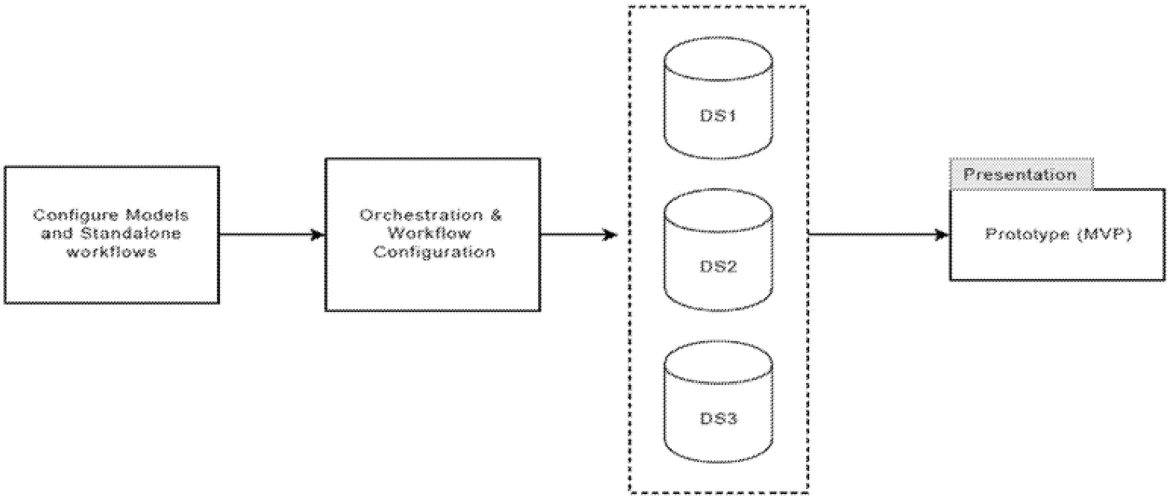

Configure Models and Standalone workflows
Orchestration & Workflow Configuration
DS1
DS2
DS3
Presentation
Prototype (MVP)

Fig. 5

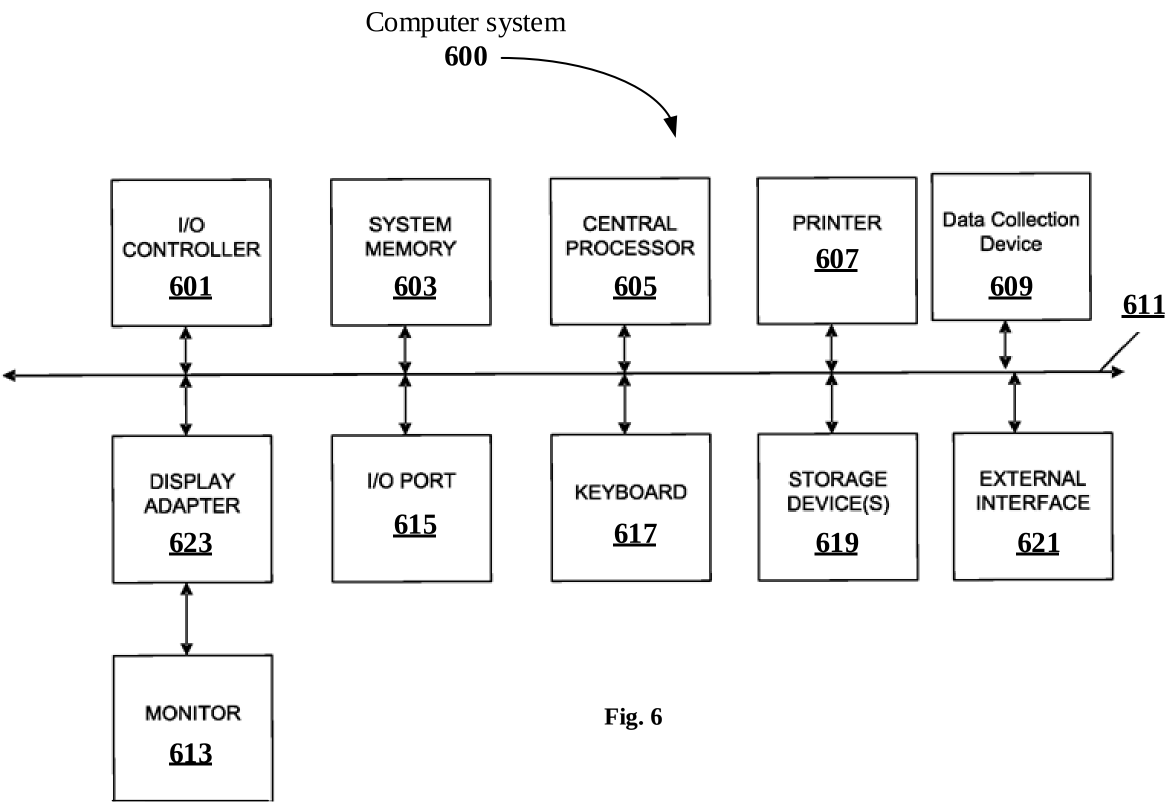

Computer system
600
I/O CONTROLLER
601
SYSTEM MEMORY
603
CENTRAL PROCESSOR
605
PRINTER
607
Data Collection Device
609
611
DISPLAY ADAPTER
623
I/O PORT
615
KEYBOARD
617
STORAGE DEVICE(S)
619
EXTERNAL INTERFACE
621
MONITOR
613

Fig. 6