Published in final edited form as:

Med Image Comput Comput Assist Interv. 2018 September ; 11073: 197–205.

doi:10.1007/978-3-030-00937-3_23.

A Natural Language Interface for Dissemination of Reproducible Biomedical Data Science

Rogers Jeffrey Leo John, Jignesh M. Patel, Andrew L. Alexander, Vikas Singh, Nagesh Adluru

University of Wisconsin-Madison, Madison, USA

Abstract

Computational tools in the form of software packages are burgeoning in the field of medical imaging and biomedical research. These tools enable biomedical researchers to analyze a variety of data using modern machine learning and statistical analysis techniques. While these publicly available software packages are a great step towards a multiplicative increase in the biomedical research productivity, there are still many open issues related to validation and reproducibility of the results. A key gap is that while scientists can validate domain insights that are implicit in the analysis, the analysis itself is coded in a programming language and that domain scientist may not be a programmer. Thus, there is no/limited direct validation of the program that carries out the desired analysis. We propose a novel solution, building upon recent successes in natural language understanding, to address this problem. Our platform allows researchers to perform, share, reproduce and interpret the analysis pipelines and results via natural language. While this approach still requires users to have a conceptual understanding of the techniques, it removes the burden of programming syntax and thus lowers the barriers to advanced and reproducible neuroimaging and biomedical research.

Keywords

Natural language user interface; Systems; Reproducibility; Provenance tracking; Neuro/medical image analysis; Surgical data science

1 Introduction

Large amounts of complex data are available in neuroimaging and biomedical imaging domains. Advances in machine learning and data science have set the stage for a new generation of analytics that will support improved decision-making by leveraging insights from data [1,2]. However, obtaining insights from data is often non-trivial. The researcher, who sifts through the data, must develop and adopt sophisticated computational pipelines to arrive at meaningful insights. These pipelines involve several complex stages such as data cleaning, data merging, data exploration, machine learning model estimation, statistical testing, and visualization, in addition to sophisticated image processing [1,3,4]. This

necessarily complex nature of the modern medical imaging methods challenges adoptability, reproducibility and ultimately their translational impact.

Furthermore, each stage in the analysis may involve using various software packages, and require that the researchers be proficient in using these packages (e.g., SQL to filter, slice and dice the data and R for data analysis). These diverse requirements for constructing data science pipelines place a significant cognitive overhead on the researcher, and increase the barriers for entry and reproducibility. We note that many researchers publish code repositories, but it is well-recognized that such an approach does not fully address the core issue of sharing and reproducing analysis pipelines [5]. This current situation holds back progress in the field as testing of new hypotheses and models can take much longer. Recent emerging research suggests that conversational interfaces can reduce some of the barriers in reproducibility and adaptability of advanced computational methods [6,7]. In this paper, we present a conversational interface that allows dissemination and use of advanced neuroimaging and general biomedical data analysis pipelines such as those in [1,3,4] with excellent reproducibility and provenance tracking. We believe that such an interface is a key step towards democratizing biomedical data analysis.

Provenance Tracking and Reproducibility.

One of the key aspects that distinguishes our system from GUI-based pipeline tools is the ability to easily construct shareable and reproducible pipelines. Also, as described later in Sect. 3, the system records all the natural language conversations, and these records/logs not only serve as documentation of the thought processes of the researcher but also provide a rich source for learning and improving the analyses themselves.

Our system has a replay mechanism through which entire pipelines can be re-created from the conversation logs. Researchers can also create variants of their pipelines by modifying the conversation logs and feeding it through the replay mechanism. For example, in our Scenario-2, Daisy – a researcher in the surgery department of a hospital – receives additional data on surgeries. She wants to retrain her model on the new data with modifications to the hyper-parameters. Retraining the model requires Daisy to reproduce all the steps that she took previously to prepare the data for training. In addition to that, Daisy wants to add more visualizations. She can recreate the complete pipeline with the necessary modifications by editing the conversation logs of the original pipeline (in Sect. 2) and replaying the conversations. As shown in Fig. 1, Daisy recreates the complete pipeline by asking the system to replay the conversation log. Sharing a pipeline created in our system now is as simple as sharing the conversation that was used to create the pipeline.

Related Work and Core Contributions.

There is a large body of research aimed at simplifying data access for non-programmers through the use of natural language [8,9]. Recent advances in natural language understanding has seen the emergence of bot frameworks such as Microsoft LUIS [10], Watson Conversation [11] and Amazon Lex [12] that are used for general purpose and simpler tasks such as ordering pizza or navigating maps. Our core technical innovation is to disseminate biomedical data science pipelines using a finite state machine (FSM)-based

Natural Language (NL) interface that allows the researcher to compose complex, domain-specific image processing and data analysis tasks using dialogues that can be translated in appropriate analysis action(s). We note that our interface is complementary and targets a significantly broader set of researchers than the alternative methods to disseminate neuroimaging pipelines, such as Nipype, Dipy, C-PAC, PyMVPA, DLTk or NiftyNet. With these existing approaches, a typical researcher is still left with the time-consuming steps of learning how to code using the various tools (each with its own pros and cons) and gluing together tasks performed using these tools into a workflow, all the while taking on the challenge of making decisions to navigate the search space of possible pipelines. Our natural language interface is a layer over programming language interfaces, which makes building workflows easier and provides a general architecture that can amplify the translational impact of advanced computational methodologies and software tools such as the ones listed above.

2 Archetypal Analysis Scenarios

Our system is implemented as an intelligent chatbot agent that lets users assemble complex data analysis pipelines through conversations. While the precise interpretation of general natural language continues to be challenging, *controlled* natural language (CNL) [13] methods are starting to become practical as natural interfaces in complex decision-making domains [14]. This observation is the crucial insight and foundation for our system. In addition, data science pipeline components can often be abstracted into “templates of code”. These two features enable us to develop a system that uses CNL to create and share reproducible biomedical data science pipelines. We demonstrate our system using two archetypal examples, one in neuroimaging and the other in surgical data science.

Scenario-1: A Neuroimaging Data Science Pipeline.

Imagine that a neuroscientist, Sally, is interested in observing the effects of age on one of the cognitive measures. She is an expert in neuroscience. While conversant in neuroimaging methods, she is not an expert in that area. She has conducted a longitudinal study and collected various cognitive features and MRI data at several time points. She is interested in performing mixed-effects analysis using both the imaging and cognitive data. One of her goals is to visualize the effects of a cognitive measure on longitudinal change in a brain region. This task is conceptually simple. However, to perform this analysis, Sally needs to carry out significant amount of longitudinal image processing, derive appropriate deformation representations and estimate the mixed effects models [4]. Our system can abstract all such processing, with provenance tracking should she want to dive into the actual steps, and avail the longitudinal model parameters for her to explore. In addition to image processing, she also needs to combine/join imaging and the cognitive information. Our system offers simple-to-use data join features with interactions such as “combine the longitudinal imaging features with the cognitive measures”. Sally can explore (Fig. 2) the data to find various measures of interest. For example, to pick measures that are correlated with age, Sally can visualize scatter plots of various cognitive measures against age (Fig. 2b). She can then estimate a statistical model to see the effects of the measure on a specific region of the brain. Once Sally estimates the model she can explore the various parameters

of the statistical model (Fig. 3). Internally, the system loads data from a database or file into a Pandas [15] dataframe, visualizes data using Plotly [16] and uses scikit-learn [17] for machine learning. For neuroimaging, the statistical model is built in Matlab whereas the visualizations are built using R. The system seamlessly orchestrates all these tools and libraries without requiring any such knowledge on the part of the user.

Scenario 2: A Surgical Data Science Pipeline.

Daisy is a researcher in the department of surgery at a hospital. Daisy has years of experience in surgical procedures and fundamentals. She is interested in identifying patterns from accumulated data on existing surgical case durations and developing models to predict the duration of a new surgery. Such a model would enable the operating room (OR) planners in making efficient OR schedules, decrease the cost and improve patient care, while maintaining the current OR utilization rate. While she has conceptual understanding of the importance of such models she is less familiar with which features to use and what model to build. A sample of Daisy's interactions with our system to analyze the surgical dataset is presented in Figs. 4 and 5.

At each stage, Daisy issues commands in natural language. Daisy begins by creating exploratory visualizations to explore the data and gain intuition into the relevant feature representations that can be used in the model. Daisy proceeds to carve out training and validation datasets and builds a regression model. The system also *proactively* reports metrics such as cross-validation accuracy after training (Fig. 4b) that may help Daisy take the next set of actions. The system also interactively guides her towards constructing a pipeline by providing hints and recommending further actions to the user. As seen in Fig. 4b, the system currently uses simple heuristics built into its knowledge base to recommend a gradient boosting regression model for the task.

Deep Learning (DL) via Dialogue.

Daisy is excited about the recent developments of DL and its impact on biomedical science. However, due to lack of background in programming, Daisy does not have a comfortable place to begin exploring such models for her work.

Her journey into DL-based analysis can begin with a simple conversation with our system such as “show me what you can do with deep learning”. Figure 6 shows Daisy creating a simple DL pipeline to predict surgical case duration. The system employs a simple but intuitive vocabulary to build deep networks. Internally the system uses Keras [18] to construct DL pipelines. The DL capabilities of the system are currently limited to the deployment of deep networks on a single machine. Future extensions will include support to build complex DL pipelines and the capabilities to deploy and monitor DL pipelines in the cloud. We note that such capabilities augment (not replace) already publicly available tools such as Matlab, DLTK and NiftyNet for applying deep learning in neuroimaging and other biomedical imaging domains.

3 The Core System Architecture

Next, we describe the various components of our conversational interface and the underlying system that powers the interface.

Client-Server Design.

The conversational nature of the system naturally lends itself to using a Jupyter [19] notebook style of interactive computing, where a Programming Language (PL)-specific kernel controls code executions triggered by the client. The chat server parses the messages, extracting semantic information about the task to be performed, disambiguating whenever required, and finally generating the executable code. The chat server triggers code generation from the code templates. A dynamic repository of code templates is maintained, along with a mapping from specific task to the corresponding code template. These templates are specific to the underlying libraries and can be automatically learned by employing techniques from PL research.

Control Flow Architecture.

The control flow in the system is shown in Fig. 7. The user chats with the **conversational agent** in a Controlled Natural Language (CNL). The conversation agent is responsible for steering the conversation towards a full task specification. The chat client sends the natural language conversations to the chat server. If the chat server determines that it needs more information to complete the task, it prompts the user until it has a complete task specification. The system then signals the **task code generator** to identify the template that best matches the specification. The chat server also consults the knowledge base if necessary, to guide the user during data analysis.

Controlled Natural Language and Storyboard.

While commonly used natural language is expressive and capable of representing arbitrary concepts, it poses a challenge for automatic semantic inference. CNLs offer a balance between the expressiveness and ease of semantic inference by restricting the vocabulary and grammar. The conversations between the user and our system are guided through a “storybook”. The storyboard describes the dialogue between the user and the system, and the actions that the system must take in response. It is essentially a finite state machine (FSM) implemented in Python. The FSM framework is crucial to the system to unambiguously extract information from conversations and map the extracted information to executable code. The FSM transitions allow the system to drive the conversation towards a complete task specification. Finally, the task templates allow the system to generate and execute the code, decoupling underlying libraries from the conversational agent.

4 Conclusion

We present a framework for performing biomedical data analysis using a natural language (NL) interface. Our system, the first of its kind for the medical imaging community, provides a novel framework for combining domain-specific NL and domain-specific computational methods to orchestrate complex analysis tasks using easy conversations. We believe this

framework will significantly lower the burden of provenance tracking and barriers to complicated setups of software packages and programming syntax required in advanced statistical and machine learning based analysis methods. The ultimate outcome would be increased productivity, rigor, reproducibility and the translational impact of advanced analysis workflows in neuro and biomedical imaging domains. In the future, we plan to perform rigorous user studies to study the benefits of using such a conversational approach with appropriate IRB approvals and randomized user selection.

Acknowledgments.

This work was supported in part by the NIH grants CPCP U54-AI117924, BRAIN Initiative R01-EB022883 and Waisman IDRC U54-HD090256.

References

1. Kim HJ, Adluru N, et al.: Multivariate GLMS on Riemannian manifolds with applications to statistical analysis of DWI. In: CVPR, pp. 2705–2712 (2014)
2. Maier-Hein L, Vedula SS, et al.: Surgical data science for next-generation interventions. Nat. Biomed. Eng 1(9), 691 (2017) [PubMed: 31015666]
3. Kim HJ, Adluru N, Bendlin BB, Johnson SC, Vemuri BC, Singh V: Canonical correlation analysis on Riemannian manifolds and its applications In: Fleet D, Pajdla T, Schiele B, Tuytelaars T (eds.) ECCV 2014. LNCS, vol. 8690, pp. 251–267. Springer, Cham (2014). 10.1007/978-3-319-10605-217
4. Kim HJ, Adluru N, et al.: Riemannian nonlinear mixed effects models: analyzing longitudinal deformations in neuroimaging. In: CVPR (2017)
5. Halchenko YO, Hanke M: Open is not enough. An integrated, community-driven computing platform for neuroscience. Front. Neuroinform 6, 22 (2012) [PubMed: 23055966]
6. John RJL, Potti N, Patel JM: Ava: from data to insights through conversations. In: CIDR (2017)
7. John RJL, Adluru N, et al.: Image analysis through conversations: reducing barriers and improving provenance tracking in AD research. AAIC 13(7), P1484–P1485 (2017)
8. Androutsopoulos I, Ritchie GD, Thanisch P: Natural language interfaces to databases - an introduction. Nat. Lang. Eng 1(1), 29–81 (1995)
9. Weizenbaum J: Eliza - a computer program for the study of natural language communication between man and machine. CACM 9(1), 36–45 (1966)
10. Williams J, Kamal E, et al.: Fast and easy language understanding for dialog systems with Microsoft LUIS. In: SIGDIAL, pp. 159–161 (2015)
11. IBM: Watson conversation (2016)
12. Amazon: Lex (2016). <https://aws.amazon.com/lex>
13. Kuhn T: A survey and classification of controlled natural languages. Comput. Linguist 40(1), 121–170 (2014)
14. Troegner D: Grammar for NL recognition: adaptation to air traffic phraseology. Technical report, Institute of Flight Control, German Aerospace Center (2011)
15. McKinney W: Pandas: a foundational python lib for data analysis and statistics (2010)
16. Plotly: Collaborative data science (2015)
17. Pedregosa F, Varoquaux G, et al.: Scikit-learn: ML in Python. JMLR 12, 2825–2830 (2011)
18. Chollet F, et al.: Keras (2015). <https://keras.io>
19. Kluyver T, Ragan-Kelley B, Pérez F, et al.: Jupyter notebooks - a publishing format for reproducible computational workflows (2016)

Fig. 1.

Replay of an analysis. A sample of the conversational log is shown in the callout (modifications are highlighted). Researchers can also replay by modifying some parameters selected in the original analysis. This capability can enable researchers to test the robustness of the pipeline and its dependence on the choice of parameters.

Fig. 2.

Exploratory data visualization by the neuroscientist in Sect. 2 to choose the variables of interest for performing advanced deformation analysis of longitudinal MRI data [4].

Fig. 3.

Visualization of relationship between a cognitive score and brain deformations generated using advanced longitudinal analysis of neuroimaging data using mixed effects models on manifolds [4]. The conversation used is shown on the right panel.

Fig. 4. Sample interactions of Daisy.

She can iteratively explore the model space and the feature space until she finds the right combination of model and features that give her the best results. She can then save, export and share the model with other researchers which can enhance the translational impact and reproducibility of her work.

Fig. 5.

Exploratory visualization of features (e.g. age, surgical duration and history) using bubble charts. The conversation by Daisy is shown in the call-out.

Fig. 6.
Sample conversation to design and train a deep learning model.

Fig. 7. Control flow in the system.

The chat client sends the natural language conversations typed by the user to a web server. The web server forwards the conversation to a Natural Language Understanding (NLU) unit which is also a part of the chat server. The argument identifier extracts the template parameters from the task specification, and the template instantiator completes the chosen template with the user specified parameter values.