Requirements Engineering for Actors-with-Learning: Encompassing the Two Kinds of Modeling for Full Cognitive Cycle RE

Eric Yu

University of Toronto, Toronto, ON, Canada

Abstract

Models are used extensively in software engineering – to facilitate requirements analysis, architectural design, and so on. Models are also central to recent approaches to AI, being the output of computationally intensive machine learning algorithms drawing on large datasets. In this short paper, we first suggest that developing machine learning applications, like other software initiatives, can benefit from familiar requirements engineering (RE) techniques such as goal- and agent-oriented RE, so that the resulting systems will address the needs and wants of end-users and other stakeholders. Then, we suggest that the two kinds of modeling are fundamentally complementary in that they pertain to the two sides of a cognitive cycle. We propose that an RE framework needs to address both sides of the cognitive cycle for all relevant actors, in order to be able to respond to the complex socio-technological realities of a data-rich world.

Keywords [1]

Requirements modeling, AI, iStar, agent-oriented, goal-oriented, knowledge level, cognitive agent, learning

1. Introduction

Models play a crucial role in requirements engineering (RE). They offer a simplified view of reality, facilitating specialized and focused lines of inquiry and analysis. Models are also central to recent approaches to AI, being the output of computationally intensive machine learning algorithms drawing on large datasets. Today, machine learning (ML) has become pervasive in software systems and already has far reaching impacts on everyday life. Having effective RE techniques that ensure technology systems meet humans needs and wants is therefore of crucial importance to system designers as well as to technology users and society at large.

In this paper, we first argue that ML models, despite being computationally generated, require considerable human conceptual effort and reasoning to achieve desired results, and can therefore benefit from familiar RE techniques based on conceptual modeling such as goal-oriented approaches, with some modifications and extensions.

We then note that, although the two kinds of models and modeling appear to be at odds with each other, one focusing on empirical data while the other relies on domain experts and human knowledge, the two are ultimately complementary. Drawing on the notion of a cognitive cycle from the study of human cognition [1], we suggest that each kind of modeling addresses one side of the cognitive cycle – perception-to-knowledge on one side, and knowledge-to-action on the other. When used together, the two kinds of models complete the cognitive cycle for each other.

In: F.B. Aydemir, C. Gralha, S. Abualhaija, T. Breaux, M. Daneva, N. Ernst, A. Ferrari, X. Franch, S. Ghanavati, E. Groen, R. Guizzardi, J. Guo, A. Herrmann, J. Horkoff, P. Mennig, E. Paja, A. Perini, N. Seyff, A. Susi, A. Vogelsang (eds.): Joint Proceedings of REFSQ-2021 Workshops, OpenRE, Posters and Tools Track, and Doctoral Symposium, Essen, Germany, 12-04-2021

EMAIL: eric.yu@utoronto.ca

We argue that a framework that conceptually encompasses both kinds of modeling can offer the more comprehensive requirements analysis that is needed for today's complex sociotechnical realities. We envision an agent-oriented requirements modeling framework based on an abstract "knowledge-level" cognitive agent covering the full cognitive cycle, so that each side of the cycle can potentially be implemented at the physical level with a mix of human and machine intelligence and enactment.

2. Machine Learning Models Are Models Too

In building software systems and applications, we are now routinely encountering two kinds of models. Models have been core to the practice of software engineering for a long time. It has been said that the critical skill of the software professional and computer scientist is the ability to work with abstractions [2]. Models are used to conceive of systems to be built and to shape a new reality. They serve as blueprints for analysis and communication with project sponsors and among project team members. These models are handcrafted, relying on the knowledge of domain experts, business stakeholders, and the creativity of system architects, strategist, and designers.

Models of a different type are equally pervasive, in the empirical sciences. Models in this context are derived from data collected from the world. Models aim to accurately capture the essence of reality as it exists, so as to be able to build theories and understand phenomena, and to predict the future.

In the past decade or so, this second kind of modeling has received a big boost from computational advances - a convergence of advances in machine learning algorithms, raw computing power, and network and mobile technologies. As modern life has gone digital, the massive amounts of data that is the digital footprint of business transactions, social interactions, and physical activities are being tapped to produce models that in turn reshape our lives.

Although ML-generated models aim to faithfully reflect what is in the "raw data", they are nevertheless artifacts, produced through the deliberate actions and decisions of data scientists, ML engineers, and project sponsors. While the specific activities in an ML project may be substantially different from those in a conventional software development project [3], and involving more experimentation and iteration, the models must ultimately serve some purpose and meet the needs of project sponsors and stakeholders. Goal-oriented RE techniques (employing modeling of the first kind) that have been developed to support conventional systems development could potentially be adapted to support the development of ML systems (which produces models of the second kind).

3. Proposal #1: Requirements Modeling for Machine Learning (RE4ML)

The recently developed GR4ML framework [4] follows a goal-oriented conceptual modeling based approach to provide systematic guidance on developing ML applications. High-level strategic business goals are progressively refined until the need for decisions are encountered. In order for decision makers to benefit from data assets to make informed decisions, they need to be prompted to ask relevant questions. These are formulated as "Question-Goals". Based on the type of Question-Goals, candidate classes of ML algorithms producing desired "Insights" in the form of models are suggested using pre-compiled catalogs. The framework also includes conceptual modeling support for selecting among candidate algorithms, as well as for determining data preparation workflow.

As in other types of system development, goal-oriented requirements modeling can facilitate making the connection between domain problem understanding and technology solution, clarifying and justifying the step by step progression from problem to solution, all the while avoiding building solutions that solve the wrong problem. Some initiatives may start top-down from business drivers, while others may emphasize bottom-up leveraging of existing assets, such as readily available data assets and reusable components. Ultimately the top-down and bottom-up elements must align for a successful project. These are supported by connections across the three submodels in GR4ML – the Business View, the Analytics/ML Design View, and the Data Prep View [16].

The goal-oriented approach of GR4ML can potentially be extended to an agent-oriented approach along the lines of i* [5], so as to reflect the differing interests of various parties and strategic

dependencies among them. The modeling concepts introduced in GR4ML specifically to address ML needs suggests corresponding dependency types as extensions to i*. One actor might pose a Question-Goal to another actor and depend on the latter for Insight (an ML-model), which the first actor can use to perform predictions. An alternate relationship might be that the second actor, instead of providing the model, offers a prediction instead. This suggests a different type of dependency. In the first case, the depender actor avoids exposing its query data to the model provider, and can further test the model to determine its reliability. In the second case, the dependee actor need not disclose the model to the depender, expecting the latter to trust the inference results. These distinctions can help the various parties decide what relationships to adopt.

By analyzing the network of dependencies among actors, one could determine whether the actor is using appropriate data sources and other inputs, which may come with biases reflecting the strategic interests of actors along the data supply chain. As in i*, quality (softgoal) dependencies among actors can be identified and evaluated.

Catalogues of design techniques for achieving common recurring softgoals can be compiled to assist data scientists and system designers, as done for non-functional requirements (NFR's) in conventional software engineering [6]. For ML and AI applications, NFR's such as fairness, accuracy, trust, and ethical principles are of particular concern [7, 8]

The actor abstraction in i* hides the details of internal operations of the actor. How an actor is able to produce an ML-model can be revealed incrementally by treating that actor as composed of multiple collaborating actors each responsible for some part of the ML process, e.g., data sourcing, cleaning, feature engineering, algorithm selection, training, testing, tuning, etc. These responsibilities could be treated as roles which may be performed by the same or different individuals (i* agents) [9].

4. The Two Sides of the Cognitive Cycle

In treating an ML model as an artifact to be produced in a development project, as we did in the preceding section, we have not given special consideration to the complementary nature of the two type of models. Both types are models in that they are representations of reality, but each is arrived at and used differently. ML modeling generates an abstraction of the world based on empirical data, whereas conventional requirements and software engineering models are abstractions used by human professionals mainly to produce software that act on the world when instantiated. From the viewpoint of creating an intelligent agent or an effective information system, it is clearly equally important that its models of the world be grounded in empirical data, and that its vision of the desired world as expressed in models can be realized through action.

By analogy to human cognition [1], the two kinds of models and modeling correspond to the two sides of a cognitive cycle – data-driven ML corresponds to the perception-to-model side, whereas model-driven software development corresponds to the model-to-action side. A cognitive agent is incomplete if either side is missing. In humans, the cognitive cycle is repeated frequently (estimated cycle time of around 300 milliseconds [10]) to be able to sense changes in the environment and to act accordingly.

Conventional software constructed based on handcrafted models with knowledge elicited from domain experts (e.g., through interviews, surveys, etc.) would typically have a long lag time on the perception-to-model side. Today, with ML, many software systems are approaching near real-time cognitive cycles in sensing the environment and responding to changes, by adopting, for example, self-adaptive software architectures [11] or data-driven requirements engineering techniques [12].

RE frameworks today need to encompass the full cognitive cycle, at least at a conceptual level (regardless of the extent to which ML models or conceptual models are directly used in the automated systems), so that requirements for both sides of the cycle will be identified, analyzed, and used to guide development of a coherent system comparable to an intelligent cognitive agent.

We briefly envision such a full cognitive cycle RE framework in the next section.

5. Proposal #2: Requirements Modeling for Actors-with-Learning (RE4AL)

As automated systems are ultimately situated within a human social setting, requirements modeling must cover the overall sociotechnical system [17, 8]. The i* agent-oriented modeling framework was designed so that actors can be modelled at an abstract level without prejudging their internal composition in terms of human or machine elements or some combination [13]. As ML becomes prevalent in software systems, we nevertheless do not want to prejudge at the early requirements stage what kinds of learning (the perception-to-model side) will be done by machine or by humans.

We envision an agent-oriented requirements modeling framework based on an abstract "knowledge-level" cognitive agent [14] covering the full cognitive cycle, so that each side of the cycle can potentially be a mix of human and machine intelligence and enactment. Early requirements analysis can thus probe requirements about model accuracy, potential bias, concept drift without prior commitment to implementation technology.

i* was inspired by classical AI and multi-agent systems where actors are modelled as having ability to achieve goals for each other [14, 15]. To accommodate the newly powerful machine-learning variety of AI, new constructs such as the new types of dependencies suggested in Section 3 could be added so that i* actors will cover the entire cognitive cycle. The actors in the extended i* will be actors with learning ability, though not necessarily machine learning.

The abstract actors will be mapped to physical agents for a more implementation-oriented analysis taking into account the strengths and limitations of various classes of machines and humans with respect to their learning and other abilities.

6. Acknowledgements

We acknowledge the support of the Natural Sciences and Engineering Research Council of Canada (NSERC). Constructive comments from anonymous reviewers and from S. Nalchigar, Z. Babar, and A. Lapouchnian are gratefully acknowledged.

7. References

[1] J. E. Laird, C. Lebiere, & P. S. Rosenbloom (2017). A standard model of the mind: Toward a common computational framework across artificial intelligence, cognitive science, neuroscience, and robotics. AI Magazine, 38.4 (2017): 13-26.

[2] J. Kramer. Is abstraction the key to computing? Communications of the ACM, 50.4 (2017): 36-42.

[3] S. Amershi et al. Software engineering for machine learning: A case study. In IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP) IEEE 2019, pp. 291-300.

[4] S. Nalchigar, E. Yu, & K. Keshavjee. Modeling machine learning requirements from three perspectives: a case report from the healthcare domain. Requirements Engineering (2021), 1-18. doi: 10.1007/s00766-020-00343-z

[5] E. Yu, P. Giorgini, N. A. Maiden, & J. Mylopoulos. Social Modeling for Requirements Engineering: MIT Press, 2011.

[6] L. Chung, B.A Nixon, E. Yu, & J. Mylopoulos. Non-functional requirements in software engineering. Springer Science & Business Media, 2000.

[7] J. Horkoff. Non-functional requirements for machine learning: Challenges and new directions. In IEEE 27th International Requirements Engineering Conference (RE) IEEE. 2019. pp. 386-391.

[8] A. D. Selbst, D. Boyd, S. A. Friedler, S. Venkatasubramanian, & J. Vertesi. Fairness and abstraction in sociotechnical systems. In Proceedings of the conference on fairness, accountability, and transparency, 2019, pp. 59-68.

[9] R. Sothilingam & E. Yu. Modeling Agents, Roles, and Positions in Machine Learning Project Organizations. In Proceedings of 13th Int. iStar workshop. CEUR workshop proceedings Vol 2641. 2020. pp. 61-66.

[10] T. Madl, B. J. Baars, & S. Franklin. The timing of the cognitive cycle. PloS one, 6(4), 2011, e14803.

[11] K. Angelopoulos, A. V. Papadopoulos, V. E. S. Souza, & J. Mylopoulos. Engineering self-adaptive software systems: From requirements to model predictive control. ACM Transactions on Autonomous and Adaptive Systems (TAAS), 13(1), (2018): 1-27.
[12] X. Franch, N. Seyff, M. Oriol, S. Fricker, I. Groher, M. Vierhauser, & M. Wimmer. Towards Integrating Data-Driven Requirements Engineering into the Software Development Process: A Vision Paper. In International Working Conference on Requirements Engineering: Foundation for Software Quality, Springer, Cham. 2020, pp. 135-142.
[13] E. S. Yu. Towards modelling and reasoning support for early-phase requirements engineering. In Proceedings of 3rd IEEE International Symposium on Requirements Engineering, IEEE. 1997, pp. 226-235.
[14] A. Newell. The knowledge level. Artificial intelligence, 18(1), (1982):87-127.
[15] E. Yu. Agent orientation as a modelling paradigm. Wirtschaftsinformatik, 43(2), (2001):123-132.
[16] S. Nalchigar & Yu, E. Designing business analytics solutions. Business & Information Systems Engineering, 62(1), (2020): 61-75.
[17] L. Liu & E. Yu. Designing information systems in social context: a goal and scenario modelling approach. Information systems, 29(2), (2004): 187-203.