Cognitive Twin: A Cognitive Approach to Personalized Assistants

Sterling Somers, Alessandro Oltramari and Christian Lebiere
Psychology Department, Carnegie Mellon University, Pittsburgh, PA
Bosch Research and Technology Center, Pittsburgh, PA
{sterling@sterlingsomers.com,Alessandro.Oltramari@us.bosch.com, cl@cmu.edu}

Abstract

This paper presents and evaluates an early prototype of a cognitive twin: a digital reflection of the user, intended to make decisions and carry out tasks on the user's behalf. The intention of the cognitive twin is to model the cognitive processes underlying the user's decisions. To that end, we implemented the twin in a computational cognitive architecture. The model specifically uses an Instance-Based Learning approach to modeling human decision making that leverages the architecture's memory mechanisms. The task of this prototype model is to help organize a social gathering. Data is generated in a discrete simulation that we are developing to freely experiment with issues ranging from environment structure to data availability. We implement two versions of the cognitive twin and show that it provides an effective personalized assistant with limited data availability.

Introduction

In the context of the Internet of Things (IoT), a *digital twin* is the virtual replica of a sensor-based connected device and of the processes that are associated with it. A network of digital twins can then be conceived as a virtual space that mirrors the physical properties of IoT entities, and that enables advanced monitoring, predictive analytics, and ubiquitous computing. The benefits of digital twins are evident in different domains, from industrial manufacturing (Kritzinger et al. 2018), where anticipating machine failures through digital simulations can dramatically reduce maintenance costs, to healthcare, where the growing number of interconnected health-monitoring systems (e.g, wearables) is prompting providers to create personalized digital solutions for their patients. Healthcare is a perfect example of an area where IoT virtualization also factors the human in, e.g., by creating digital twins of the human body that can be used for real-time remote-monitoring of physiological functions (Bruynseels, Santoni de Sio, and van den Hoven 2018). Beyond modeling the human as an organism, digital twins can expand to cover social dimensions of human life, which are characterized by our daily interactions with portable electronics, digital services, social networks, etc. Accordingly, in this paper we introduce the notion of *cognitive* (digital) *twin*, to highlight the key role that cognitive mechanisms play in modeling human decision making in the IoT digital space.

The aim of the cognitive twin is to make the decisions that a user would would make and to deal with those aspects of life that can be handled over the internet. Much of our social lives are planned over the internet and a cognitive twin would be able to handle tasks such as the planning of social events: who should attend, when it should take place, what activities should occur, etc. With an increase in the use of IoT devices, we can imagine a cognitive twin that not only knows when to turn on the dishwasher, when to buy the soap, where to

buy the soap, but actually buy the soap, and even turn on the dishwasher.

We propose a hybrid data-driven and knowledge-based approach. Because our context is human decision making, we implement our digital twin in a computational cognitive architecture. Our approach is to take advantage of existing knowledge ontologies, and structured procedural knowledge, but also learn decisions from data. We believe this approach can balance domain generality while maintaining a low implementation and data cost.

The architecture we are working with is a hybrid cognitive architecture, ACT-R (Anderson 2007). The ACT-R architecture, discussed more thoroughly below, consists of both symbolic elements: declarative knowledge and procedural knowledge; as well as sub-symbolic elements that support statistical learning at a human-timescale. The tight integration of these two levels of the architecture enables the combination of knowledge representation and reasoning with statistical machine learning in every aspect of decision making rather than artificially segregating them into one or the other. In this project we use a cognitive modeling approach, Instance-Based Learning (IBL), in which past decisions are used to inform future decisions by directly leveraging the mechanisms of the cognitive architecture.

Dinner Party Planning

We have chosen as our proof-of-concept scenario to implement the planning of a dinner party. We chose this scenario for a number of reasons.

Perhaps one of the most obvious reasons for using a party planning scenario is that it is such a strong use-case for a cognitive twin. Party planning requires back and forth negotiation between the potential attendees with respect to scheduling and also requires information about the social dynamics of possible attendees (avoid having people that do not like each other at a party), food preferences or diet restrictions, etc. Some of that information is sensitive, such as personal likes and dislikes. These are the the sort of tedious details that not only require time of the host but time of the potential guests. Planning a party requires a great deal of structured knowledge and systematic process in order to make decisions about the various facets that need to be planned. Those facets interact as well (for instance the guest list might determine when the party can take place and vice versa) resulting in a potential complex process with multiple cycles of replanning.

If the cognitive twin is going to plan a party for a user, it has to know who that user would want to attend. From a practical perspective, the twin likely has to *learn* about that user's friendships, professional relationships, etc. One could imagine a twin that works by asking about attendees but, then, such a twin has less utility because it requires significant effort on the part of the user.

From a practical perspective, there are a number of tasks we would expect the digital twin to learn and a number of tasks that are more practically represented as knowledge. Between these two extremes of pure data-driven learning, and pure knowledge acquisition and authoring, there are cases where we would expect a balanced hybrid approach to be the most practical.

Knowledge-based Party planning is a complex *process*. There are a number of sub-tasks that are required and the ordering of the sub-tasks is important. The knowledge of how to plan a party, however, would be difficult to learn from data because the steps are not necessarily performed in any fixed sequence, the patterns of occurrence are abstract, and requires tying together different facets of information that are rarely explicitly exhibited in behavior (therefore, would be difficult to collect data for). Instead those processes are typically communicated at least in part explicitly as instructions, in a process known as Interactive Task Learning (Laird et al. 2017).

Data-Driven There are plenty of aspects to the party planning task that can be purely data-driven. For example, preferences in diets (e.g. vegetarian) are exhibited in behavior and, therefore, leaves a trace that can plausibly be learned from. Other regularities like schedules, party sizes, and social preferences, could all be plausibly learned from available data.

Hybrid We believe that the party-planning scenario can benefit from a balanced, hybrid approach. There are some aspects that can be and should be data-driven (e.g. who your friends are), while other aspects can benefit from both structure and data while others can largely be represented as structured knowledge (e.g. the process

of planning itself, diet categories, etc). An integrative framework that can combine both process aspects is therefore required. Previous attempts have been made at integrating cognitive architectures and knowledge ontologies (Oltramari and Lebiere 2013).

ACT-R

ACT-R is a hybrid (symbolic, sub-symbolic) computational cognitive architecture. The components of the architecture represent a unified, neurally inspired theory of cognition (Anderson 2007). The architecture is comprised of a set of modules, each of which are constrained by the biological processes and limitations of the brain. While the architecture does include modules for vision, audition, and motor activity, the core of the system is comprised of a central production system and a declarative memory.

The central production system represents procedural knowledge (skills). Each production 'fires' in response to particular patterns or 'chunks' of information held in buffers that constitute the cognitive architecture's working memory. Skill acquisition, response competition, and multitasking capacity are among the cognitive capacities modeled in the production system.

Declarative memory is a long-term knowledge storage and retrieval system. Declarative knowledge is represented as chunks, that are propositional compositions of other chunks or more basic values. Each chunk in declarative memory has an activation that modulates its retrieval. The history-based activation component is called base-level activation, which reflects the chunk's recency (when it was encoded or recited) and frequency (number of times it has been recited) to capture regularities of human behavior such as the power laws of practice and forgetting. Furthermore a chunk's activation is affected by spreading activation, such that related chunks (e.g., context information) can boost the activation of chunks in memory, depending on their degree of co-occurrence. Chunks can be retrieved through a pattern-matching process in which a sub-pattern is requested and a complete pattern is (potentially) returned. Importantly, memory retrieval in ACT-R is not perfect. Chunks with a low activation can fail to be retrieved (forgetting) while a partial-matching process can generalize to similar but not perfectly matching chunks depending on how close their similarity is to the requested pattern. Finally, chunk activation includes a stochastic component that makes memory retrieval a probabilistic process.

For this project we make use of the Instance-Based Learning methodology (IBL), which leverages declarative memory by making decisions based on retrieval from memory of previous decisions rather than through heuristics or production rules (Gonzalez, Lerch, and Lebiere 2003).

Data / Simulation

In order to collect data for a cognitive twin we have chosen to implement a discrete simulation. The simulation is currently under development but produces sufficient data to test the cognitive twin. Our aim, currently, with the simulation is a vehicle for generating data in liue of human data. The simulation represents human agents going about their lives, while personal data is being collected and stored. The agents in the simulation are distinct from cognitive agents. In a sense, the simulated agents represent 'real' human beings (in liue of human data) and the cognitive agents represent their cognitive twins. Long-term, we aim to expand on the simulation, making the 'lives' of the agents more veridical, the properties of the network to reflect human social networks, etcetera. Also, as we develop the simulation further, we hope to make the data collected about users reflect more closely the quality of data potentially collectable from real humans. In the following we describe the simulation in its current form, including aspects we are not currently using in development and evaluation of the cognitive twin.

The simulation creates a population of n agents that carry out both scheduled (e.g. work) and unscheduled events (socializing, eating) over a period of days. For this paper we simulate 720 days of simulation. As our simulation is generated stochastically, for this analysis we set a random seed, and all analysis is performed on a single, generated world. Relevant information about the activities including the time, the event type, and specific details about the event are recorded in data files.

The simulation is controlled by a number of parameters. To generate a 'world', a user specifies a population of n size. The simulation then generates families, where the member-size is controlled by a weighted distribution parameter. Although family

dynamics is not captured in the simulation at this early stage of development, we have designed the simulation to accommodate more complex interactions in the future.

Since our main proof-of-concept case is the party planning scenario, we have developed the simulation such that activities include social activities, to track positive and negative social interactions; food preparation, to track diet restrictions; and daily activities, to track an individual's schedule.

Schedule The simulation ticks by the hour and, currently, events have a 1-hour resolution. Implementation-wise, there is no known restrictions on resolution. Days of the week are soft-coded into the simulation (by dividing hours of simulation time) and are used to design the schedule. Each individual is assigned a static work and sleep schedule, reflecting real-world schedules (9-5 Monday to Friday). Eating can occur nested within another event (i.e. eating at work), and the start and end times are chosen from a weighted distribution of pre-selected hours. For example, breakfast can occur between 6 am and 9 am with different weights for different hours (free parameter).

At generation, agents are assigned scheduled events. These events are mandatory and are called 'sports', 'clubs', and 'hobbies'. The intention is to create regularly-occurring events that should be considered when scheduling a social event. The events are selected from a weighted random list, with the possibility of repeats reduced for each by re-weighting the events. For example, the likelihood of sports being selected is reduced if sports is selected once. There is also a placeholder 'None' event. Events are selected, scheduled, and re-weighted; the process repeats until an event fails to be scheduled on randomly selected days or until the None event is selected. It is, therefore, possible (though low probability) that some people will have no activities for which they are committed.

All times, event types, and event details are recorded in an individual's log-file as well in a global log-file.

Social Interactions When a 'world' is generated, social links between every pair of individual agents in the simulation are generated randomly from a truncated normal distribution, the variance of which is a free parameter. Values for the social links are real numbers ranging from -1 to 1. The social links are both incoming and outgoing and, as a result, can be asymmetric (i.e., one agent may favor another, but may not be favored in return). We consider a negative connection between person A and person B to represent tension or negative feelings of A towards B. Large negative values represent strong dislike. Positive values represent a positive relation and we define values greater than 0.75 as friendship. When the simulation is running, the social links are used when setting up social interactions in the simulation. The social links between agents change over time as they interact with one another (see Figures 4 and 5). Roughly, friends become more likely to interact and enemies less likely, as described below.

Social interactions (parties) are created during run-time at the start of each simulation day. Social interactions, on any given day, are potentially centered around randomly selected individuals. The parameter s is set as a proportion of the total population and we used a value of 25 to ensure that we generate sufficient data for testing. This value was qualitatively assessed to maintain steady world statistics and, as described below, we evaluate the model with different ranges of history. Once centers (hosts) are selected, attendees are selected to attend.

Attendees to each social interaction are selected from the population such that their likelihood of being selected is weighted by the outgoing social link from the 'center' to the potential attendees. The outgoing links between the center and the attendees are transformed to produce a probability distribution, and non-uniform random samples are selected. A free parameter which is set to control the minimum social interaction connection required to attend is set to a minimum value of -0.1, so that people with slight dislike for each other might attend. A small bias is also added to the weights of connections above the friend threshold (how we separate friends and close friends). The bias is set to 0.3 and the friend threshold is set to 0.75. We use the values to help control the patterns of interactions. These values were set qualitatively to produce steady world statistics.

Not all centers will result in a social interaction. Once attendees are picked, the social interaction then has to be scheduled. Scheduling may fail due

to firm commitments by attendees. The minimum size of social interactions are selected randomly from a truncated normal distribution between 4 and 10. The shape of that distribution is also a free parameter. If an insufficient number of people are able to attend, the social interaction is canceled.

Furthermore, social interactions are resolved in a queue, which is ordered by the original selection of centers. Attendees that are in the queue for multiple events (lots of friends) may become busy, potentially becoming unavailable for events further down the queue.

During a social interaction, all agents in attendance receive an interaction score with the other guests. The interaction scores are selected from a truncated normal distribution between -1 and 1, and the shape of the distribution is a free parameter. The total score that an interaction receives is a linear combination of: the random score (-1 to 1), the link from person A to person B, and the link from person B to person A. For this paper, those factors are all weighed the same as we have yet to analyze the impact of these values on the social dynamics across time. Values above 1 or below -1 are truncated. As a result, it is possible for the quality of relationships to change: friendship links between agents may go from positive to negative after interactions and, likewise, negative links can become positive after interaction. Negative links becoming positive are most likely to occur in social interactions not centered by the agent with outgoing negative links. Either the attendees show up at a mutual friends party or there is an imbalance in their outgoing and incoming links. Although crude, this is meant to capture the influence of past interactions. The time of interaction and the scores between attendees is recorded in the log-file for the individual as well as a global log-file.

To get a sense of what the socialization looks like, we have plotted some basic statistics of the interactions over 720 days of simulation.

Figure 1 shows the percent of people who have a zero or greater (i.e., positive) outgoing social link between them (blue), with the shaded area (gray) showing one standard deviation. In orange is illustrated the percent of people who are above the 'close-friend' threshold, with the orange shaded area showing one standard deviation. Figure 2 shows the percent of people at a party who are close friends with the party center (or host) (blue) with the gray area showing one standard deviation.

As can be seen from those two figures, the relationships in the simulation are fairly stable. These can be modified using the free parameters but we have chosen to keep the relationships rather stable, as we assume this reflects the nature of relationships in the real world.

Figure 1: Social Connections

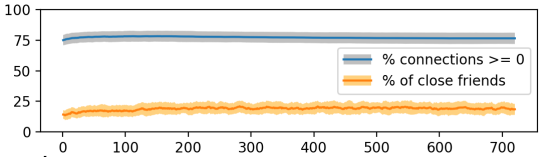


Figure 2: Friendship With Party Centre

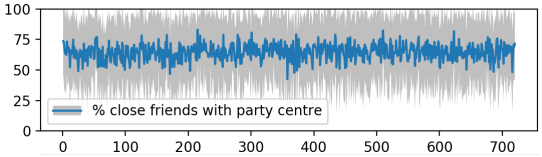
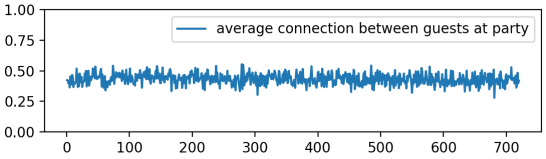


Figure 3 represents the average connection between all guests in attendance at the social interactions. Note that the simulation does not try to maximize all the social links of a social interaction, as attendees are chosen stochastically, weighted by the outgoing social link from the party center.

Figure 3: Average Connection Between Party Guests



To have a sense of the social networks over time, Figures 4 and 5 represent the steadiness of people that have positive connections with a target over time. It shows the percent of change of people in the positive social network per day (blue), as well as the change since time zero. Figure 5 shows the same stats (blue vs orange) but where the members of the social network are above the friend threshold (0.75).

Figure 4: Positive Social Network Over Time

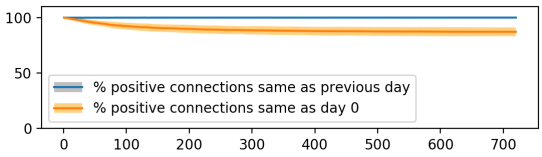
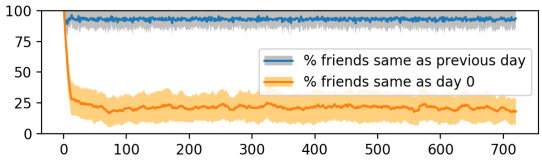


Figure 5: Friend Social Circles Over Time



Diet and Food

Individual agents in the simulation are either omnivores, vegetarians, or vegans, and consume food according to their diets. The ontology of diets is modeled from ConceptNet (Liu and Singh 2004) and the foods in the simulation are, therefore, structurally consistent with that knowledge. Building the structure into the simulation is important for what we are trying to achieve, as we expect that a learning system should recover that structure. We propose that a system like the cognitive twin can take advantage of existing knowledge graphs like ConceptNet to take advantage of known structure but should also be able to learn that structure, or any combination thereof. In this work, we are not not learning diet restrictions but intend to do so for future work.

Cognitive Twin Model

At a high-level, the cognitive twin is intended as an automated decision maker. It uses your data to make the kinds of decisions you would make. In the party-planning scenario, your cognitive twin is intended to do some of the leg work of party planning for you. The cognitive twins, described below, use the personal data from the simulation agents to plan their next party for them (given an existing history of social interactions).

The cognitive twin was developed in ACT-UP (Reitter and Lebiere 2010), a toolkit implementation of ACT-R developed to make it easier to integrate with simulations, and ultimately to develop and distribute on portable platforms with limited footprint and computing power, as well as real-time performance and network-size scaling requirements. The key equations leveraged were those for declarative memory, as it reflects tradeoffs in recency vs frequency in generalizing patterns of user activity, as well as generalization in continuous spaces such as high-dimensional vectors used in distributional semantics.

In the current model, batch activity data is loaded into memory. We will discuss later the type of protocol that could be used to request information from peer cognitive twins, and their impact on the algorithm used by the main cognitive twin planning the activity.

Finding the most compatible set of guests relies on ratings of past joint social activity. Those are represented as chunks in memory consisting of the user who provided the rating, the agent with whom that user interacted, and the rating of the interaction. Those ratings are not necessarily symmetrical, so each pairwise interaction might result in two separate chunks with distinct values. Those memory chunks have associated a base-level activation that reflect the recency (and frequency if the same rating between the same agents has been provided multiple times) of the rating. Each chunk activation can therefore be interpreted as the relative importance of that rating among competing ones, and factored accordingly by memory retrieval processes. Specifically, the blending retrieval process (Lebiere 1999) is used to produce a consensus estimate of social rating between two agents by retrieving and aggregating individual ratings weighted according to their activation. These social compatibility ratings are used in a greedy algorithm that starts with the central user organizing the gathering. The rating of all potential guests are evaluated against the current set of invitees (starting with the host) and the guest with the highest rating is selected. The process is repeated until the guest list is full. It is possible that this process does not yield an optimal outcome because of the sequential selection process could lead to a local optimum. An optional second phase (not yet implemented) would consist of computing the social rating of each guest with respect to all other guests, then swapping out the guest with the lowest rating if another guest with a higher rating with the re-

maining guests could be found. This process would be repeated until no such guest would be found.

The time scheduling process operates in a similar manner. All considered time slots are generated by a process that combines user input with knowledge constraints (e.g., social events are not scheduled in the middle of the night, or during regular work hours). Past and scheduled activities of each potential guest are represented in memory, each as a chunk including identity of the agent, day, time and nature of activity. For each potential time slot, the availability of each guest is generated through a blending process that generalizes from past and future activities to infer regularities such as typical activities for a particular day/time slot without needing to represent them explicitly, such as by accessing a calendar. The time slot with the highest expected availability is selected. An optional process (not yet implemented) would trigger another guest selection process if enough guests cannot attend at a given time slot.

Results

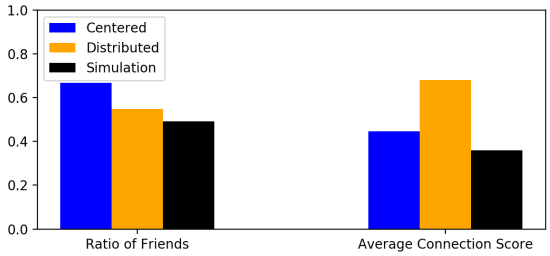
The model was evaluated against a simulation of 100 different agents run for two years of simulated time. At the end of that time, the model consists of 100 distinct cognitive twins, each loaded with the history of its specific agent. That results in about 16,000 chunks in the memory for each twin, about 150 per week, corresponding to a mix of social events and interaction values.

The model is run to generate a party for each of the 100 agents. Two versions of the model were run. Recall, this simulation is a set random seed of possible worlds. The distributed version, as described above, factors in the social interactions with all current scheduled guests when adding a new guest. The centered version, however, only selects guests based upon their social interactions with the host, as is the case for the simulation. The parameters of the model are set to provide fairly accurate performance, including a mismatch penalty parameter (scaling the partial matching process) of 5.0 and an activation noise parameter (scaling the stochastic activation factor) of 0.1 as well as the usual time-based decay parameter (weighting recency against frequency) of 0.5.

This evaluation focuses on the guest selection process rather than the time scheduling. The models are evaluated against the simulation baseline according to two measures, displayed in Figure 6. The first measure is the average percentage of each party that is made of close friends to the host. The centered version of the cognitive twin (69.75%) outperforms the distributed version of the cognitive twin (56.75%) because it is more narrowly focused on optimizing compatibility with the host social connections rather than mutual guest connections. They both outperform the simulation algorithm (49.25%) because of the built-in stochasticity of that algorithm.

The second measure is the average connection strength between each pair of guests (including the host). By that measure, the distributed version of the cognitive twin (0.680) outperforms the centered version of the twin (0.464) because it factors the mutual guests connections instead of strictly those from the host. Both outperform the simulation algorithm (0.358) because of its combination of narrow focus on the host and stochasticity.

Figure 6: Ratio Of Friends and Average Score at Social Event

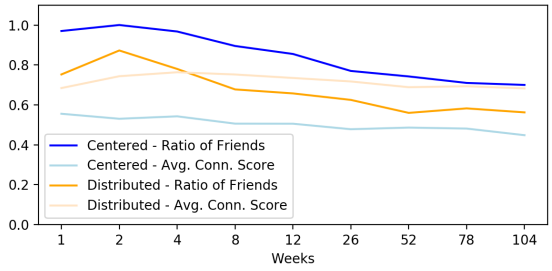


The data requirements of the model were also evaluated. For that purpose, the model was run in the same settings as above, but with a more limited history of social interactions. The length of history was varied from 1 week to the full 2 years. Results for both measures describe above are plotted in Figure 7.

For both model versions and both measures, performance actually increases with decreasing history length. This results because changing social connections means that historical data becomes increasingly inaccurate over time. Cognitive mechanisms such as blending are usually able to generalize quite well with limited data. The optimum history length is between 2 and 4 weeks depending

upon the measure and model version.

Figure 7: Ratio of Friends and Average Connection by Weeks of Data



Discussion

Parameters of the model were selected to be compatible with standard values used to match human performance in ACT-R models (see models and publications on the ACT-R web site at http://act-r.psy.cmu.edu) and have not been optimized for this specific simulation environment. For instance, a larger time decay parameter would allow combining the accuracy of a longer history without the penalty of overly relying on obsolete information.

Another method for evaluating the model is model tracing (Corbett and Anderson 1995). That approach, developed to parameterize general models to match individual behavior traces, would consist of running the cognitive twin alongside the simulation, making predictions at each step that could be compared to simulation events, as well as gradually learning an increasingly long history of its users.

Improved generalization over limited data could also be obtained by relying on similarities between agents rather than treating agents as distinct symbolic entities. For instance, similarities between agents could be set to reflect the degree of overlap in their activities or social connections.

The structure of the various versions of the cognitive twin also has implications for communication protocols between cognitive twins. For instance, the centered version of the model does not rely on any social information from other cognitive twins, while the distributed version does require information from invited guests regarding future potential guests. This leads to a trade-off between overall quality of the social gathering and privacy of the guests. Developing an infrastructure that can optimize those trade-offs while safeguarding user data is an essential goal of the cognitive twin approach.

References

Anderson, J. R. 2007. *How Can The Human Mind Occur In The Physical Universe?* New York, NY: Oxford University Press.

Bruynseels, K.; Santoni de Sio, F.; and van den Hoven, J. 2018. Digital twins in health care: ethical implications of an emerging engineering paradigm. *Frontiers in genetics* 9:31.

Corbett, A. T., and Anderson, J. R. 1995. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modelling and User-Adapted Interaction* 4(4):253–278.

Gonzalez, C.; Lerch, J. F.; and Lebiere, C. 2003. Instance-based learning in dynamic decision making. *Cognitive Science* 27(4):591–635.

Kritzinger, W.; Karner, M.; Traar, G.; Henjes, J.; and Sihn, W. 2018. Digital twin in manufacturing: A categorical literature review and classification. *IFAC-PapersOnLine* 51(11):1016–1022.

Laird, J. E.; Gluck, K.; Anderson, J.; Forbus, K. D.; Jenkins, O. C.; Lebiere, C.; Salvucci, D.; Scheutz, M.; Thomaz, A.; Trafton, G.; Wray, R. E.; Mohan, S.; and Kirk, J. R. 2017. Interactive task learning. *IEEE Intelligent Systems* 32(4):6–21.

Lebiere, C. 1999. The dynamics of cognition: An act-r model of cognitive arithmetic. *Kognitionswissenschaft* 8:5–19.

Liu, H., and Singh, P. 2004. Conceptnet—a practical commonsense reasoning tool-kit. *BT technology journal* 22(4):211–226.

Oltramari, A., and Lebiere, C. 2013. *Knowledge in Action: Integrating Cognitive Architectures and Ontologies*. Berlin, Heidelberg: Springer Berlin Heidelberg. 135–154.

Reitter, D., and Lebiere, C. 2010. Accountable modeling in act-up, a scalable, rapid-prototyping act-r implementation. In *Proceedings of the 10th International Conference on Cognitive Modeling, ICCM 2010*, 199–204.