PICA: Proactive Intelligent Conversational Agent for Interactive Narratives

Jessica Falk
ETH Zurich

Steven Poulakos
Disney Research

Mubbasir Kapadia
Rutgers University

Robert W. Sumner
Disney Research

ABSTRACT

A narrative relies on the imperfect knowledge of the user to create interactions between the characters that are ultimately used as a plot device to drive the narrative. This motivates our exploration of ways to encode this information, provides means for a user to both query and influence the knowledge, and guides the user based on a model of their experience. We developed PICA: a proactive intelligent conversational agent for interactive narratives that can guide users through such experiences. The underlying knowledge base is designed using a sub-symbolic architecture, which encodes belief models for multiple users and autonomous agents in addition to the actual story knowledge. We also developed a discourse module using Behavior Trees to intuitively design the proactive and reactive capabilities of PICA. We compare our approach to neural networks and symbolic knowledge bases and demonstrate its functionality.

KEYWORDS

Conversational Agents; Interactive Narratives; Knowledge Maps; Hybrid Knowledge Representation

ACM Reference Format:
Jessica Falk, Steven Poulakos, Mubbasir Kapadia, and Robert W. Sumner. 2018. PICA: Proactive Intelligent Conversational Agent for Interactive Narratives. In *IVA '18: International Conference on Intelligent Virtual Agents (IVA '18), November 5–8, 2018, Sydney, NSW, Australia.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3267851.3267892

1 INTRODUCTION

The knowledge a user or an agent has about a narrative is often imperfect. Tuetenberg and Porteous [35] argue that imperfect knowledge is one of the key components of a narrative. A narrative relies on imperfect knowledge to stimulate the interactions between the different characters given their beliefs, desires and intentions. This is ultimately used as a plot device to drive the narrative, for example by creating plot twists. However, it is also important to have access to the perfect knowledge of a narrative, in this paper called story knowledge, to provide the best possible guidance for the user. By encoding this information, we are able to reason about the user and create more compelling narratives using a user's belief model to provide better guidance based on their beliefs and preferences.

IVA '18, November 5–8, 2018, Sydney, NSW, Australia

ACM ISBN 978-1-4503-6013-5/18/11…$15.00
https://doi.org/10.1145/3267851.3267892

We present PICA: a proactive intelligent conversational agent for interactive narratives. PICA is able to converse using natural language, react to user input and proactively initiate conversations about the narrative. PICA encodes the past, present and future of both story knowledge and belief models and can be used together with story-authoring systems to guide the user through interactive narratives by acting as an extradiegetic story guide. We are targeting narratives where the user is watching a story unfold. Using PICA as an embodied virtual character, we can increase the immersion by having a participant of the narrative be explicitly aware of the user. By reacting to the user's engagement with the story, PICA is able to initiate conversations with the user to inform them about any parts of the story they might have missed and is able to guide them to the most interesting path through the story. For instance, Figure 1 shows an example of the user not being very engaged with the story, when Horton and Victor were talking. Thus, PICA stores a value close to 0 for its belief of the user knowing about Celia. A value of -1 indicates that PICA thinks that the user is not aware of it, while 1 would represent the user definitely knowing about it. Since this is a crucial part of the story, PICA initiates a conversation in which it informs the user about this fact, thereby increasing its belief of the user's knowledge according to the engagement.

This paper makes the following main technical contributions: We developed a sub-symbolic architecture, called the Temporal Knowledge Map (TKM), to encode story knowledge. This architecture is inspired by neural-symbolic systems [7, 37], where a symbolic knowledge base provides the foundation to initialize and design the resulting neural network to model uncertainty and story graphs [10, 11] to model time. This guarantees a modular structure at the event level, which cannot be guaranteed by using a neural network for knowledge representation, and thereby it offers a convenient way to selectively update and reason about every node in the TKM. Compared to symbolic knowledge bases, the TKM is able to modify parts of its story graph to model the progression of the story. This is enabled by adding an inference system to the TKM that handles these modifications. We also developed a discourse module using Behavior Trees to author a modular representation of the proactive and reactive capabilities of PICA.

We demonstrate PICA in the context of an existing interactive narrative planning system, to show its ability to interface with experience managers [28, 29].

2 RELATED WORK

Encoding of Knowledge and Narratives: Min et al. [24] propose using LSTMs to infer the user's goals during a game. Thue et al. [36] propose PASSAGE, a system to combine the modelling of the user with the story generation to provide stories that are tailored for the user. Ramirez and Bulitko [27] improved PASSAGE further to dynamically adapt stories depending on the user's style of play.

Figure 1: An example conversation with PICA using a Symbolic Knowledge Base and a Temporal Knowledge Map. It shows the updates of two events stored inside the knowledge base and their confidence vector at different points of the story. The first value is the story knowledge and the second one stores the user belief.

Approaches on modeling story representations are described by Elson [10, 11] and Eger, Beret and Young [9]. Chekol et al. [4] encodes time and uncertainty in a knowledge base representing a general storage of facts.

Conversational Agents: Eliza [38] and ALICE [14] are both chatbots using rules and pattern matching. Deep recurrent neural networks [18, 31, 33] provide a powerful way of creating general conversational agents. Reinforcement Learning [12] enables the agent to learn during the conversation. However, all of the machine learning approaches require a large amount of training data which can easily result in a knowledge acquisition bottleneck. There have been approaches in using conversational agents [3] in stories and games [25] as well as in educational environments [30, 34]. Yu and Riedl [39] propose an invisible story guide guiding the user.

Neural-Symbolic Systems: KBANN [37] and C-IL2P [5–7] provide simple and intuitive algorithms to translate a symbolic knowledge base into a neural network. To refine their network, they both still need training data. This obscures the initial facts and rules that were used to design the neural network. There have also been discussions [15] concerning the advantages and challenges that neural-symbolic systems and sub-symbolic systems [22, 32] still face. A symbolic knowledge base [2] is still the system used in many applications as it provides an intuitive way of understanding and reasoning about the knowledge. The issue remains how to acquire knowledge and modify the knowledge base [19].

Story graphs and symbolic knowledge bases provide a way to encode narratives. We want to encode uncertainty to model belief which is made possible by using neural networks. We propose to combine these approaches to create PICA.

3 OVERVIEW

PICA is designed to be a guide in interactive narratives. Users can interact with PICA to learn more about the story and influence it. The flow of a conversation between PICA and the user is depicted

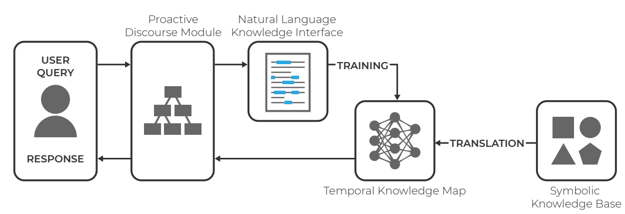

Figure 2: PICA constructs its knowledge base using a symbolic representation of the narrative. The knowledge base can be queried and modified at any time.

in Figure 2. Any user input will be passed through the Proactive Discourse Module directly to the Natural Language Knowledge Interface, where the user intention and important parts of the user input will be extracted before propagating them to the TKM. The following steps need to be executed for setting up the TKM:

(1) PICA receives information about participants, actions, states and the story by receiving a symbolic knowledge base encoding this information.
(2) It constructs the overall graph given this information using the rules defined in Section 5.1 to define the edges.

During the execution of the story, the TKM can be modified at any time. This can be triggered by either the user interacting with PICA or the story-authoring tool informing PICA about the progression of the story. To modify the TKM, PICA isolates the corresponding event in the graph and trains only its direct weights with gradient descent using exactly one data point. Using a neural network approach for knowledge representation, one would need to train the full network. However, in our approach the TKM represents a story graph that stores a neural network for each event. This modular representation allows to encode and train the knowledge and the user's belief for each event individually. In the next sections, we explain the architecture of PICA in more detail.

4 NATURAL LANGUAGE KNOWLEDGE INTERFACE

The Natural Language Knowledge Interface extracts important information from the user input. User intent is classified and parsed data associated with the input is forwarded to the TKM. Extracting the user intent and parsing the user input are done in parallel.
Intent Classification: Given the user input x, we classify x to retrieve the most-likely intention $i \in I$, where I is the set of user intentions. Knowing the intention enables us to improve PICA's responses. To classify x, we use the Rasa NLU library [1]. We defined seven user intentions:

- **Query:** The user queries an action, state or participant.
- **Belief:** The user agrees or disagrees how certain events in the story played out.
- **Location:** The user can ask where someone or something is and the agent is able to guide the user to its location.
- **Narration:** The user may not observe all actions and can ask the agent to narrate the currently executed events.
- **Guidance:** The user can either learn about the currently available user interactions, how to follow a certain character path or how to follow the best possible path in the story as provided by the story-authoring system.
- **Repetition:** The user can ask PICA to repeat its responses.
- **Narration of Story Information:** The user can ask whether they missed something during an event. PICA will check the knowledge base and tell the user about any important story information that was conveyed during that event.

Syntax Parser: The Natural Language Knowledge Interface provides an intuitive interface to the user to communicate with PICA. Using Google's SyntaxNet [17] to extract dependencies from the user input x, we are able to retrieve the subject, verb (or adjective) and object and pass them to the TKM.

5 KNOWLEDGE REPRESENTATION USING TEMPORAL KNOWLEDGE MAPS

Our goal is to create a conversational agent acting as a guide in interactive narratives. To model the knowledge base for such an agent, we propose our Temporal Knowledge Map (TKM), which combines a story graph with a neural network. This enables the modeling of time and uncertainty in a single knowledge base. The most important parts of our TKM are:
Affordance: An affordance $a \in A$ is an action that is physically possible for an object or a person as described by Gibson [16]. We define the affordee as the object or person that offers its affordances, while the afforder is the person interacting with the affordee.
State: A state $s \in S$ describes an attribute and has an afforder associated with it.

An affordance or state change is stored inside the TKM using Knowledge Nodes. A Knowledge Node consists of a triple (i, n, f), where $i \in \mathbb{N}_{\geqslant 0}$ is the global index, n is the neuron inside our TKM and $f \in F$ is the affordance or state. When querying a node, we

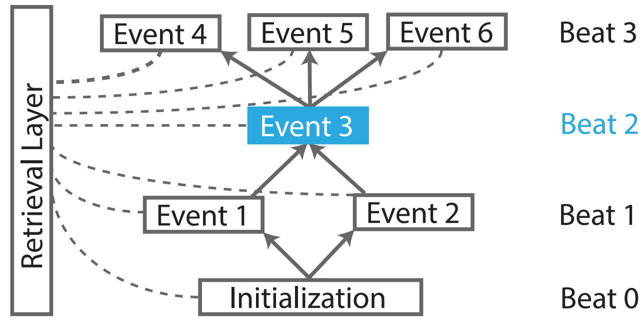

Figure 3: A visualization of our Temporal Knowledge Map.

receive a vector of confidence values. Each confidence value may describe the actual narrative, the user's belief or an agent's belief.
Events: An event $e \in E$ consists of a list of states and affordances that can be executed in parallel or sequentially. Figure 4 visualizes how an event is represented in the TKM. Each event has its own network containing knowledge nodes and defining the causal dependencies between them. Section 5.1 defines the logical inference necessary to construct the event network.
Story Information: A story information $t \in T$ describes information that is conveyed during the story and is vital for understanding it. Each t is connected to an event e during which the information got conveyed. Story information may be used to provide constraints guiding the narrative progression [26].
User Interaction: User interactions $u \in U, U \subseteq E$ are a special type of event that the user can trigger by e.g. clicking on an object. The supplementary material describes how PICA identifies which user interactions are currently available.
Beats: Each beat $b \in B$ describes one time step in a story. It contains a list of all events $[e_1, ..., e_n]$ that will be executed in parallel. It holds that for any two events in the same beat, they cannot have the same characters involved in them.
Arcs: Each arc $c \in C$ stores information about its beats $[b_1, ..., b_n]$, the active user interactions $[u_{c1}, ..., u_{cm}]$ and the story information $[t_1, ..., t_l]$ constraint associated with the arc. In a linear story, there would be only one arc containing the narrative. An interactive narrative consists of multiple arcs that may emerge at any time and act as branching points throughout the story.
Deviations: The user may interact with the story at any time. This can result in a deviation $d \in D, d = [b_1, ..., b_t]$ that needs to be executed before continuing with the story.

Figure 3 depicts the overall TKM with Figure 4 representing the internal implementation of each event node in the TKM. The narrative is structured as a graph $G = (L, K)$. $L = (e \in E | \exists d \in D : e \in d \vee e \in c, \text{where executed}(c))$ that defines the story log describing which arcs and deviations have already occurred. The first beat serves as the initialization beat. Each state will be set to its initial value. $K = \langle e_1, e_2 \rangle$ describes the edges between two events and it holds that $e_1, e_2 \in L \wedge e_1 \in b_i \wedge e_2 \in b_{i+1}$.

Each layer of our TKM represents one beat in the narrative. PICA is aware of the active beat and can reason about the past, present and future. The future is only encoded for the currently executed arc. As PICA is guiding an interactive narrative, it is possible at

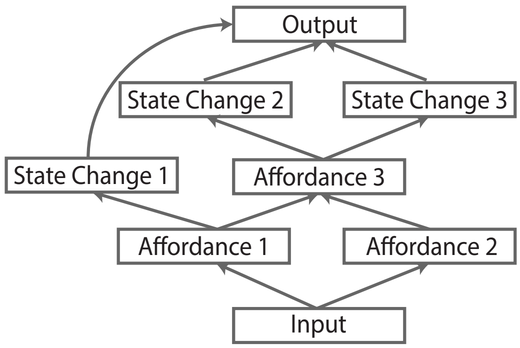

Figure 4: Each event in our TKM has its own network to store affordances and state changes.

any point that the currently executed arc will change making the overall future of the story difficult to predict. The retrieval layer shown in Figure 3 provides a look-up table to query nodes at certain beats by accessing references to the neurons.

5.1 Setting up the Temporal Knowledge Map

Figure 4 provides an overview of an event structure. Each event contains affordances and state changes that are causally dependent on each other.

Implication: The implication represents the connection between the affordances and the state changes. Each affordance might only be executed after certain state changes were triggered.

Negation: Not all states might be true from the beginning and some may be set to false during story execution. We define the negation function of a neuron n as

$$val(n) = \begin{cases} 1 & \text{if the affordance got executed.} \\ -1 & \text{if the affordance will not be executed.} \\ 1 & \text{if the state will be true.} \\ -1 & \text{if the state will be false.} \end{cases}$$

Logical Conjunction: Each affordance or state can have multiple dependencies. If it can only be executed after multiple other affordances or state changes have been executed, we concatenate them using a soft $\wedge$-operation. The weight function for the resulting connections is $w_{ab} = \frac{val(n_a)}{deg^-(n_b)}$.
The activation function is $a_b = \sum_{i \in deg^-(n_b)} w_{ib} \cdot a_i$.
Meaning that we only activate n_b, if all input nodes are active.

Logical Disjunction: If only one of multiple affordances or state changes need to be executed to start the next one, we use the $\vee$-operation. Each dependency is able to activate the outgoing neuron. This is achieved by setting all weights to $w_{ab} = val(n_a)$.
The activation function is $a_b = \max_{i \in deg^-(n_b)} w_{ib} \cdot a_i$ and therefore we only activate n_b, if at least one input node is active.

Universal Quantification: Each $\forall$-rule is stored in a separate list. Before adding a new neuron, we first check whether any stored $\forall$-rule is applicable. If a $\forall$-rule is applicable and we haven't stored this information before, we add a new instance of this rule to L.

Existential Quantification: This rule represents one type of user query. Instead of querying the confidence value of a neuron, the user can ask for all neurons that are related to some event.

5.2 Querying the Temporal Knowledge Map

To match the parsed user input to a Knowledge Node, we need to compare it to the information stored inside the Knowledge Node. By searching for an intersection between their synonym groups using WordNet [13] and utilizing Word2Vec [23] to retrieve a similarity score, we are able to reliably match the user input to the correct node and either return its confidence value for the actual story knowledge (or the relevant participants) or PICA will tell the user that it couldn't find any information about it.

5.3 Modifying the Temporal Knowledge Map

The TKM can be modified either by the user or by the story-authoring system connected to PICA. Depending on the intention, we either want to modify a belief model or the actual story knowledge. This is stored for each Knowledge Node in a confidence vector y, where the first value contains the story knowledge and all the other values map to one of the belief models. The story knowledge is modified whenever

- an event ends, we receive information of whether it was interrupted. If it was, we set its story knowledge to -1.
- the currently executed arc changes, any future beats of that arc will not be executed. We invalidate them by setting the story knowledge of their events to -1.

The user belief model is modified whenever

- the user states their belief about an affordance or state.
- the end of a beat is reached. The story-authoring tool sends additional information about the engagement of the user with regards to the currently executed events to PICA.

To train the TKM and adjust its weights w, we are using gradient descent on only the mentioned events, affordances or state changes. Therefore, we isolate the corresponding Knowledge Nodes and extract their neurons as well as their direct weights.

5.4 Handling Deviations and Arc Changes

The user is able to interact with the story, which can lead to deviations and arc changes. We need to handle the following deviations:

Arc Change. When an arc change occurs, the current arc is stopped and all future beats are invalidated. The future arc is retrieved and all of its beats are added to L by connecting them to the events from the current beat.

User Interaction Deviation. Every interaction that the user triggers leads to a sequence of events that will be executed and a plan is computed by the story authoring system to proceed with the story. These events and the resulting plan can consist of several beats and we denote the number of beats as l. To handle this deviation d, we take the following steps and denote the current beat as b_t:

(1) All future beats are invalidated and stored temporarily.
(2) The beats of d will be connected to b_t.
(3) The stored beats are added to beat b_{t+l}, if the plan results in coming back to the current story arc. Otherwise, an arc change will be initiated after adding d.

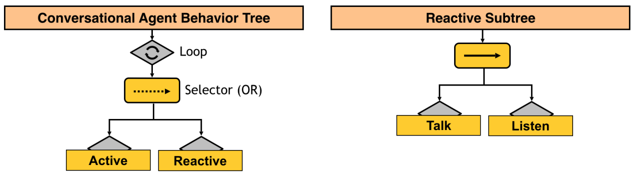

Figure 5: An overview of the Proactive Discourse Module.

6 PROACTIVE DISCOURSE MODULE

The Proactive Discourse Module is defined by a Behavior Tree (BT) [8] and provides the direct interface to the user. BTs provide a modular and intuitive formalism of modeling an agent's behavior and have been successfully used in games like Halo 2 [20]. The task of the Proactive Discourse Module is to provide a simple and modular interface for the author to extend the agent with predefined responses. Figure 5 provides an overview of how we separated the different types of interactions that the agent can execute.

Active Agent Behavior: The active part of the BT provides PICA with the ability to initiate conversations. Using the user belief model, PICA can identify whether the user is about to miss or already has missed an important event. We want to inform the user about missed information or guide the user to experience this event. An example of this is shown in Figure 1. PICA can also make the user aware of possible user interactions.

Reactive Agent Behavior: The reactive BT is separated into two parts to define the tasks PICA has to be capable of to correctly respond to the user input. The Listen subtree simply takes the user input and passes it to the Natural Language Knowledge Interface. The Talk subtree handles PICA's response. Each supported intention has its own subtree that contains the check for the intention, which was extracted from the user input as explained in Section 4.1, and the action needed to be executed.

7 EVALUATION

To evaluate our system, we conducted a performance evaluation to demonstrate the real-time aspect of PICA, which can be found in the supplementary material. We compare the TKM with alternative approaches and show an example of PICA guiding the user through an interactive experience in the supplementary video.

7.1 Knowledge Base Comparison

As our knowledge base combines a symbolic representation with a neural network, we compare our system with these two approaches to evaluate whether our approach overcomes their limitations.

Neural Network. A neural network provides the possibility to identify complex dependencies and predict the correct outcome fairly accurately given a suitable set of training data and the correct architecture for building the neural network. A traditional neural network is trained by providing values for the input and the output layer and then use back propagation to update the weights accordingly. When training a neural network, there is a risk of losing parts of the story as the neural network may decide during training to store other information and dependencies. In our case, we want to be able to exactly map each neuron to an affordance or state change so that we are able to reason about it. Neural networks do not provide such a modular structure. Our TKM is able to update the underlying network training any node without having to retrain the whole network.

Symbolic Knowledge Base. A symbolic knowledge base provides the possibility to clearly define and understand the structure of the story. These advantages align with our use case and provide the possibility to clearly identify the different steps and dependencies inside the story. However, to provide the best-possible guidance, we need to store the user's belief model inside our knowledge base to identify which parts of the story the user has witnessed and understood. The TKM provides an inference system that takes as input observations about the user and retrieves the correct node and its direct weights that need to be updated to facilitates this. By using gradient descent to update them, we are able to incorporate uncertainty and make the overall transition between states more smooth. Figure 1 illustrates this by showing an example of using a symbolic knowledge base and the TKM. Both are able to keep track of the progression of the story. However, the symbolic knowledge base does not have an inference model and therefore it becomes much more difficult to model the user's belief correctly as can be seen in beat 3 and 4, where PICA is unsure of whether the user was paying attention to the story and therefore initiates a conversation with them to ensure they didn't miss anything important.

7.2 Demonstration of PICA

Interactive narratives allow the user to influence the story. Depending on the interactions executed, the chosen story arcs may change. This allows for a highly interactive and immersive experience. We integrate PICA into the interactive extension of the story-authoring tool CANVAS [21] that allows the author to easily create such interactive narratives.

To ensure that we do not lose the modularity of PICA, we only loosely couple the two systems. We created an interface that allows any story-authoring system to communicate with PICA without direct integration. For this purpose, we defined system commands to make use of the existing features of PICA. These commands allow PICA to be aware of the current story progression, whether a user interaction has happened and whether the story has deviated.

The supplementary video demonstrates PICA's capabilities in such a story-authoring tool. The first two interactions between PICA and the user showed the proactive capabilities that PICA offers. Due to storing a model of the story knowledge and the user's belief model, PICA is able to inform the user about missed information and nudge them about important events. The other interactions are examples of how PICA can respond to a variety of different user inputs by taking into account the derived user intention and whether the user is talking about the past, present or future. The last interaction demonstrates the user's ability to influence the story by asking PICA for guidance. Depending on whether they scare Horton when he is about to drink the suspicious potion, the user will either help to save Celia or Horton.

8 LIMITATIONS & FUTURE WORK

PICA still faces several limitations. PICA is only able to reliably parse simple sentences. As soon as the user uses a more complex sentence structure, PICA might misunderstand the user. Providing a more robust syntax parser would allow PICA to parse more complex sentence structures. We currently encode seven user intentions. As we want to encode more intentions, it becomes more difficult to train them efficiently. Implementing a more advanced intent classification would allow PICA to identify more intentions.

The TKM can encode multiple belief models at the same time, but we only explored having one user model. More complex stories could test how PICA can be used to not only act as a guide, but also as the underlying AI for NPCs. A future user study could evaluate how useful PICA's guidance is to the user and how well the user's belief model stored in PICA matches the actual user beliefs.

9 CONCLUSION

We have presented PICA: a proactive intelligent conversational agent for interactive narratives. Our work is motivated by the value of representing imperfect story knowledge in interactive narratives. PICA can be used in story-authoring systems to guide the user through the experience. By storing the user's belief model, PICA is able to initiate conversation with the user whenever necessary and the user is able to interact with PICA to learn more about the story and update PICA's belief model about the user.

PICA's architecture consists of multiple components. The Natural Language Knowledge Interface preprocesses the input received from the user and prepares it for the TKM. The TKM will extract the corresponding nodes, before sending the results back to the Proactive Discourse Module to create a response for the user.

We have compared our approach with two alternatives using neural networks and symbolic knowledge bases. In both cases, we are able to identify the advantages and limitations. We have demonstrated that combining both approaches enables us to use their advantages and overcome their limitations. Our TKM is able to perform in real-time and can encode both time and uncertainty.

ACKNOWLEDGMENTS

This work was (partly) funded through the EC H2020 Framework Programme project FLAME, Grant Agreement No. 731677. Kapadia is funded in part by NSF IIS-1703883, NSF S&AS-1723869, and DARPA SocialSim-W911NF-17-C-0098.

REFERENCES

[1] 2016. Rasa NLU. https://github.com/RasaHQ/rasa_nlu.

[2] C. Baral and G. De Giacomo. 2015. Knowledge Representation and Reasoning: What's Hot. In *Proc. of the 29th AAAI Conf. on Artificial Intelligence.* 4316–4317.

[3] Justine Cassell. 2001. Embodied conversational agents: representation and intelligence in user interfaces. *AI magazine* 22, 4 (2001), 67.

[4] Melisachew Wudage Chekol, Giuseppe Pirrò, Joerg Schoenfisch, and Heiner Stuckenschmidt. 2017. Marrying Uncertainty and Time in Knowledge Graphs.. In *Proc. of the 31st AAAI Conf. on Artificial Intelligence.* 88–94.

[5] A. d'Avila Garcez, K. Broda, and D.M. Gabbay. 2002. *Neural-Symbolic Learning Systems: Foundations and Applications.* Springer.

[6] A.S. d'Avila Garcez, K. Broda, and D. M. Gabbay. 2001. Symbolic knowledge extraction from trained neural networks: A sound approach. In *Artificial Intelligence, 125.* 155–207.

[7] A.S. d'Avila Garcez and G. Zaverucha. 1999. The connectionist inductive learning and logic programming system. In *Applied Intelligence Journal, 11(1).* 59–77.

[8] R Geoff Dromey. 2003. From requirements to design: Formalizing the key steps. In *Software Engineering and Formal Methods, 2003. Proceedings. First International Conference on.* IEEE, 2–11.

[9] Markus Eger, Camille Barot, and R Michael Young. 2015. Impulse: a formal characterization of story. In *OASIcs-OpenAccess Series in Informatics*, Vol. 45. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

[10] David K. Elson. 2012. Detecting story analogies from annotations of time, action and agency. In *Proceedings of the LREC 2012 Workshop on Computational Models of Narrative, Istanbul, Turkey.*

[11] David K. Elson. 2012. DramaBank: Annotating Agency in Narrative Discourse. In *Proc. of the 8th Intl. Conf. on Language Resources and Evaluation.* 2813–2819.

[12] Iulian Vlad Serban et al. 2017. A Deep Reinforcement Learning Chatbot. *CoRR* abs/1709.02349 (2017). arXiv:1709.02349 http://arxiv.org/abs/1709.02349

[13] Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database.* Bradford Books.

[14] A.L.I.C.E AI Foundation. 2002. ALICE. http://www.alicebot.org/

[15] A d'Avila Garcez, Tarek R Besold, Luc De Raedt, Peter Földiak, Pascal Hitzler, Thomas Icard, Kai-Uwe Kühnberger, Luis C Lamb, Risto Miikkulainen, and Daniel L Silver. 2015. Neural-symbolic learning and reasoning: Contributions and challenges. In *AAAI Spring Symposium - Knowledge Representation and Reasoning: Integrating Symbolic and Neural Approaches.*

[16] James J. Gibson. 1977. The Theory of Affordances. In *Perceiving, Acting, and Knowing: Toward an ecological psychology.* R.Shaw & J. Bransford (Eds.), 67–82.

[17] Google. 2016. Google Syntaxnet: Neural models of syntax. Retrieved from github.com/tensorflow/models/tree/master/syntaxnet.

[18] Sepp Hochreiter and Juergen Schmidhuber. 1997. Long short-term memory. In *Neural Computation.* 9(8):1735–1780.

[19] L. B. Holder, Z. Markov, and I. Russell. 2006. Advances in Knowledge Acquisition and Representation. In *J. Artif. Intell. Tools, vol. 15, issue 6.* 867–874.

[20] D Isla. 2005. GDC 2005 Proceeding: Handling Complexity in the Halo 2 AI. Retrieved August 31, 2017 from https://www.gamasutra.com/view/feature/130663/gdc_2005_proceeding_handling_.php

[21] Mubbasir Kapadia, Seth Frey, Alexander Shoulson, Robert W. Sumner, and Markus Gross. 2016. CANVAS: Computer-assisted Narrative Animation Synthesis. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '16).* Eurographics Association, Goslar Germany, Germany, 199–209.

[22] F Kurfess. 2002. Integrating symbol-oriented and subsymbolic reasoning methods into hybrid systems. In *From Synapses to Rules-Discovering Symbolic Rules from Neural Processed Data.* 275–292.

[23] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *ICLR Workshop.*

[24] W. Min, B. Mott, J. Rowe, B. Liu, and J. Lester. 2016. Player Goal Recognition in Open-World Digital Games with Long Short-Term Memory Networks.. In *Proc. of the Twenty-Fifth Intl. Joint Conference on Artificial Intelligence.* AAAI Press.

[25] T.W. Morris. 2002. Conversational Agents for Game-Like Virtual Environments. In *AAAI Spring Symposium on A. I. and Interactive Entertainment.* 82–86.

[26] Julie Porteous, Marc Cavazza, and Fred Charles. 2010. Applying Planning to Interactive Storytelling: Narrative Control Using State Constraints. *ACM Trans. Intell. Syst. Technol.* 1, 2 (Dec. 2010), 10:1–10:21.

[27] Alejandro Jose Ramirez and Vadim Bulitko. 2012. Telling Interactive Player-specific Stories and Planning for It: ASD+ PaSSAGE= PAST. In *Proc. of the Eighth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment.*

[28] Mark O. Riedl and Vadim Bulitko. 2013. Interactive Narrative: An Intelligent Systems Approach. *AI Magazine* 34, 1 (2013), 67–77.

[29] Mark O Riedl, Andrew Stern, Don Dini, and Jason Alderman. 2008. Dynamic experience management in virtual worlds for entertainment, education, and training. *International Transactions on Systems Science and Applications, Special Issue on Agent Based Systems for Human Learning* 4, 2 (2008), 23–42.

[30] S. Robinson, D. Traum, M. Ittycheriah, and J. Henderer. 2008. What would you ask a conversational agent? Observations of human-agent dialogues in a museum setting. In *The 6th Intl. Conf. on Language Resources and Evaluation.*

[31] Iulian V Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2015. Building End-To-End Dialogue Systems Using Generative Hierarchical Neural Network Models. In *Proc. 30th AAAI Conf. on Artificial Intelligence.*

[32] D. Silver. 2013. On Common Ground: Neural-Symbolic Integration and Lifelong Machine Learning. In *9th Workshop on Neural- Symbolic Learning and Reasoning.*

[33] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to Sequence Learning with Neural Networks. In *Proc. of the 27th Intl. Conf. on Neural Information Processing Systems - Volume 2 (NIPS'14).* MIT Press, Cambridge, 3104–3112.

[34] P. Tarau and E. Figa. 2004. Knowledge-Based Conversational Agents and Virtual Storytelling. In *ACM Symposium on Applied Computing.* 39–44.

[35] Jonathan Teutenberg and Julie Porteous. 2015. Incorporating Global and Local Knowledge in Intentional Narrative Planning. In *Proc. of the 2015 Intl. Conf. on Autonomous Agents and Multiagent Systems.* Richland, SC, 1539–1546.

[36] David Thue, Vadim Bulitko, Marcia Spetch, and Eric Wasylishen. 2007. Interactive Storytelling: A Player Modelling Approach. In *Proc. of the Third AAAI Conf. on Artificial Intelligence and Interactive Digital Entertainment.* 43–48.

[37] Geoffrey G. Towell and Jude W. Shavlik. 1994. Knowledge-Based Artificial Neural Networks. In *Artificial Intelligence, vol. 70, no. 1,2.* 119–165.

[38] Weizenbaum. 1966. Eliza. http://www.med-ai.com/models/eliza.html

[39] Hong Yu and Mark O Riedl. 2014. Personalized interactive narratives via sequential recommendation of plot points. *IEEE Transactions on Computational Intelligence and AI in Games* 6, 2 (2014), 174–187.