Modeling Conflict De-Escalation in Shakespeare through Hybrid NLP & Symbolic Approaches

Nicholas Treynor*, Kyle Mitchell*, Nick Toothman†, Gina Bloom‡, Colin Milburn§, Michael Neff¶, Joshua McCoy¶
*Dept. of Computer Science, University of California, Davis, Davis, CA, USA
Emails: {ntreynor, kdmitch}@ucdavis.edu
†Dept. of Computer and Electrical Engineering and Computer Science,
California State University, Bakersfield, Bakersfield, CA, USA
Email: ntoothman@csub.edu
‡Dept. of English, University of California, Davis, Davis, CA, USA
Email: gbloom@ucdavis.edu
§Depts. of Science and Technology Studies, English, and Cinema and Digital Media,
University of California, Davis, Davis, CA, USA
Email: cnmilburn@ucdavis.edu
¶Depts. of Computer Science and Cinema and Digital Media,
University of California, Davis, Davis, CA, USA
Emails: {mpneff, jamccoy}@ucdavis.edu

***Abstract*—This demo paper presents an interactive narrative game where players intervene in Shakespearean scenes to de-escalate imminent violence using natural speech. The system employs a hybrid architecture: modern natural language processing (NLP) tools (speech-to-text, large language models, sentence transformers) interpret user utterances for known de-escalation strategies (e.g., active listening, redirection) and gauge their effectiveness, while symbolic systems model character emotional stances, beliefs, and personality traits that affect input interpretation, in order to select fitting responses. The project contributes an investigation of current NLP capabilities for interactive character-driven games and serves a pedagogical purpose, teaching conflict resolution and violence prevention within a broader educational framework.**

I. Introduction

Creating believable, responsive non-player characters (NPCs) remains a central challenge in interactive narrative design, with a significant body of research exploring architectures for agent reasoning, such as those based on Belief, Desire, and Intention (BDI) [1] [2]. Common approaches to character implementation (e.g., dialogue trees) often don't reflect the fluidity of human interaction, so we explore a system where NPCs update their understanding of conversations in real time, filtered through their defined personality and current disposition. While not a full BDI model, our approach similarly models internal mental states to drive behavior.

A combination of lasting *Traits* and temporary *Stances* governs the revision of *Beliefs*, and serves an important design goal: to challenge players to read and adapt to a character's personality and mood. This paper provides a technical overview of our methodology, covering two major components: the Game Environment, which models the character and their internal logic, and the Server Architecture, which processes and interprets spoken input using modern Natural Language Processing (NLP) techniques.

Paper category: Demo Paper.
We acknowledge the support of the National Science Foundation under Grant No. IIS-2232066, as well as UC Davis and the Center for Artificial Intelligence and Computational Futures.

II. Related Works

Our inspiration for this system is multifaceted. Whilst there are many examples of real-time knowledge representation logics that have been used to drive interactive systems in the past [3], the demands of our experience call for a streamlined approach. Consequently, NPCs track core conversational *Beliefs* as discrete, usually boolean states (e.g., `BelievesPlayerSincere`), allowing for rapid updates based on player input.

We also build upon principles from social AI systems like Comme il Faut (CiF) by McCoy et al. [4], which show the value of character-specific *Traits* in shaping an agent's choices. We adapted this concept: rather than *Traits* directly causing a desire for a specific action, as is common in CiF, character *Traits* in our system act as cognitive filters. For instance, a `Paranoid` character is less likely to trust the player, and so is less to adopt the `BelievesPlayerSincere` *Belief*.

To complement these persistent *Traits*, we introduce temporary *Stances* (e.g., `Defensive`, `Neutral`, `Vulnerable`) that reflect an NPC's short-term emotional or conversational posture. These shape which aspects of an interaction—or which of the character's own *Beliefs*—are most important to the character at that moment. This models the psychological observation that individuals in certain affective states may become less receptive to certain communication strategies, as their focus narrows [5].

III. Methodology

A. Game Environment

Our NPC AI, implemented in Unity (C#), centers on a `CharacterState` script that manages each NPC's *Traits* (e.g., `Paranoid`), *Stance* (e.g., `Defensive`), and conversational *Beliefs* (e.g., `BelievesPlayerSincere`). These components jointly mediate how NPCs interpret user input.

Player interactions engage the `CharacterState` script, where core logic updates *Beliefs* by filtering input through the NPC's *Traits* and current *Stance*. For instance, an NPC in a `Defensive` *Stance* might need more sustained or persuasive player input to alter certain *Beliefs* than one in a `Neutral` or `Vulnerable` *Stance*.

`CharacterState` also governs how *Traits* influence *Stance* transitions, how *Stances* shape the weighting of input, and how *Belief* changes are conditioned on previous *Beliefs*. This ensures *Belief* modifications consistently reflect the NPC's personality, *Stance*, and dialogue.

Updated *Beliefs* and *Stances* within `CharacterState` then determine subsequent actions, with `CharacterState` acting as Unity's central processing hub, dispatching responses to other components to determine appropriate dialogue lines, animation states, or gesture triggers. Thus, it translates player interactions into cognitive shifts manifested as observable NPC reactions. An image of the interaction between the player and NPC is visible in Fig. 1.

B. Server Architecture

The server-side system works to identify conflict de-escalation strategies in user speech, and scores their prose's effectiveness at this task. While earlier systems like Façade [6] required custom parsers to interpret user input, advances in NLP and the rise of LLMs (e.g., GPT-3/4 [7]) have made building systems to parse natural language much simpler.

To enable natural interaction, player speech is first captured and processed server-side. Audio is transcribed using OpenAI's Whisper model, then passed to GPT-4o with structured prompts to classify the player's utterances. The model is tasked with identifying one of five possible de-escalation strategies, and one of seven "dramatic strategies" (lines of argument or inquiry likely to be particularly resonant to the specific character we are modeling) present within the user's response, provides justification for its classifications, and scores the argument's effectiveness on a 0–1.0 scale. Results are returned to the server as a JSON object for further processing.

To handle occasional hallucinations or formatting issues from LLMs, we use sentence transformers (SBERT [11]) to compare responses to a set of predefined options via cosine similarity. This allows us to match the LLM's output to the closest intended meaning and assign the correct tag, even if the format is off. These tags then inform NPC emotion and response selection in the game layer, as shown in Fig. 2.

IV. Conclusion & Future Work

Feedback on the demo from students praised voice input as a natural alternative to typing or dialogue trees, though speech-to-text accuracy was a concern. Macbeth's persona felt authentic, but the character lacked plot memory and voiced responses. Future plans include adding plot-aware dialogue, voicing Macbeth, and improving transcription. We look forward to bringing the game to wider audiences to solicit feedback as we iterate on the current system.

Fig. 1. The game's UI layout. At the top, a bar displays Macbeth's current emotional state. Transcription of the user's last spoken words are highlighted at the bottom, while Macbeth's response is present in white text above.

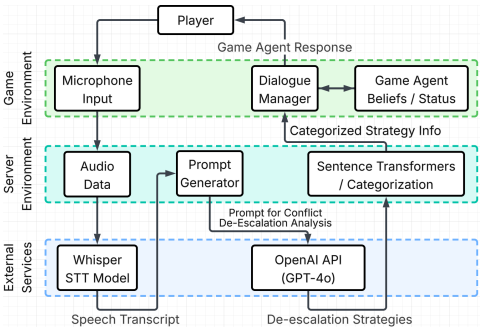

Fig. 2. A system diagram, showing the main game loop. The user uses natural language to interact with the game, processed through text transcription, LLM prompting, and categorization of used conflict de-escalation strategies, before use in the system's personality and *Beliefs* module to select an appropriate response to the player.

References

[1] J. McCarthy, "Programs with common sense," in *Proceedings of the Teddington Conference on the Mechanization of Thought Processes*. London: Her Majesty's Stationery Office, 1959, pp. 75–91.

[2] M. E. Bratman, *Intention, Plans, and Practical Reason*. Cambridge, MA: Harvard University Press, 1987.

[3] I. Horswill, "Postmortem: MKULTRA, an experimental AI-Based game," in *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, AIIDE 2018, November 13-17, 2018, Edmonton, Alberta, Canada*, vol. 14, no. 1. AAAI Press, 2018, pp. 45–51.

[4] J. McCoy, M. Treanor, B. Samuel, and N. Wardrip-Fruin, "Comme il faut: A system for authoring playable social models," in *Proceedings of the Seventh AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, AIIDE 2011, October 10-14, 2011, Stanford, California, USA*. AAAI Press, 2011, pp. 158–163.

[5] C. Argyris, *Overcoming Organizational Defenses: Facilitating Organizational Learning*. Boston, MA: Allyn and Bacon, 1990.

[6] M. Mateas and A. Stern, "Façade: An experiment in building a fully-realized interactive drama," in *Game developers conference*, vol. 2. Citeseer, 2003, pp. 4–8.

[7] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.