Automatic Learning and Generation of Social Behavior from Collective Human Gameplay

Jeff Orkin
Massachusetts Institute of Technology
Media Laboratory
Cambridge, Massachusetts, USA
jorkin@media.mit.edu

Deb Roy
Massachusetts Institute of Technology
Media Laboratory
Cambridge, Massachusetts, USA
dkroy@media.mit.edu

ABSTRACT
Current approaches to authoring behavior and dialogue for agents that interact with humans in virtual environments are labor intensive, yet often yield less robust results than desired in the face of the incredible variance possible in human input. The growing number of people playing multiplayer games online provides a potentially better alternative to hand-authored content – capturing behavior and dialogue from human-human interactions, and automating agents with this data. This paper documents promising results from the first iteration of a *Collective Artificial Intelligence* system that generates behavior and dialogue in real-time from data captured from over 11,000 players of *The Restaurant Game*. We first describe the game, the collective memory system, and the proposal-critique driven agent architecture, and then demonstrate quantitatively that our system preserves the *texture*, or meaningful local coherence, of human social interaction.

Categories and Subject Descriptors
I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search – *plan execution, formation, and generation.*

General Terms
Algorithms, Measurement, Design, Experimentation.

Keywords
Social Behavior, Natural Language, Dialogue, Learning, Computer Games.

1. INTRODUCTION
Current approaches to authoring behavior and dialogue for agents that interact with humans in virtual environments are labor intensive, yet often yield less robust results than desired in the face of the incredible variance possible in human input. Few games or training simulations allow open-ended natural language dialogue input. The growing number of people playing multiplayer games online provides a potentially better alternative to hand-authored content – capturing behavior and dialogue from human-human interactions, and automating agents with this data. This *Collective Artificial Intelligence* approach empowers agents with shared memories spanning the wide range of language and behavior found in human interactions. *Collective Intelligence* refers to the phenomenon where a large number of ordinary people collectively make better decisions than a small number of experts. Similarly, Collective *Artificial* Intelligence refers to capturing data from a large number of ordinary players in order to produce behavior and dialogue more robust than can be authored by a small number of skilled designers or engineers.

In previous work [10], we have described collecting data from thousands of people online with a multiplayer game, and using this data to learn statistical models of language and behavior in an everyday scenario with an unsupervised system. These models of social norms have been demonstrated to have the ability to estimate the likelihood of observed interactions with accuracy that correlates well with human judgment. This paper describes taking the next step – automating characters with learned behavior and dialogue.

Our study documents promising results from the first iteration of a system that generates behavior and dialogue in real-time from collective memories captured from over 11,000 people playing *The Restaurant Game*. We first describe the game, the collective memory system, and the proposal-critique driven agent architecture, and then demonstrate quantitatively that our system

Figure 1. Screenshot from *The Restaurant Game*

Cite as: Automatic Learning and Generation of Social Behavior from Collective Human Gameplay, Jeff Orkin, Deb Roy, *Proc. of 8th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2009)*, Decker, Sichman, Sierra and Castelfranchi (eds.), May, 10–15, 2009, Budapest, Hungary, pp. 385–392

preserves the *texture*, or meaningful local coherence, of human social interaction, while future work remains to capture the higher-level structure.

Each gameplay session produces a raw log file that can be summarized into a more human-readable form. We provide some sample summarized output from an actual interaction between a human customer and an agent waitress in Figure 2, which we will refer back to as we describe parts of the system.

```
WAITRESS: "table for one?"
CUSTOMER: "yes, just me tonight"
WAITRESS: "Follow me, please"
CUSTOMER SITSON chair3(Chair)
WAITRESS: "Sit here. Would you like to see a menu?"
CUSTOMER: "yes please"
WAITRESS: "ok, one second then"
WAITRESS PICKSUP dyn008(Menu) FROM podium(Podium)
WAITRESS GIVES dyn008(Menu) TO CUSTOMER
CUSTOMER: "thanks"
WAITRESS: "there you go"
CUSTOMER: "can I start with a beer?"
WAITRESS: "certainly, coming right up"
WAITRESS: "can i have a beer please"
dyn028(Beer) APPEARS ON bar(Bar)
WAITRESS PICKSUP dyn028(Beer) FROM bar(Bar)
WAITRESS PUTSDOWN dyn028(Beer) ON table1(Table)
CUSTOMER: "thanks"
```

Figure 2. A human customer and an agent waitress interact.

2. THE RESTAURANT GAME

We designed *The Restaurant Game* as a platform for collecting rich physical and linguistic interaction among humans. Players of *The Restaurant Game* are anonymously paired online to play the roles of a customer and waitress in a 3D virtual restaurant. Players can move around the environment, type open-ended chat text, and manipulate objects with a point-and-click interface. Every object provides the same interaction options: pick up, put down, give, inspect, sit on, eat, and touch. Objects respond to these actions in different ways. Food diminishes bite by bite when eaten, while eating a chair makes a crunch sound, but does not change the shape of the chair. Picking up drinks from the bar allows a player to give the drink to the other player, or put the drink down on a surface. Trying to pick up the bar itself results in playing an audio clip expressing physical exertion.

To date, we have collected log files from 8,430 completed games, played by 11,187 different people. A game takes about 10-15 minutes to play, and is considered complete if two players joined, and at least one player filled out a survey indicating an intentional end to the interaction. An average game consists of 84 physical actions, and 40 utterances with an average length of four words each. Prior to completion, player interactions vary greatly, ranging from games where players dramatize what one would expect to witness in a restaurant, to games where players fill the restaurant with cherry pies. While many players do misbehave, our previous work [10] has demonstrated that when immersed in a familiar environment, enough people do engage in behavior that allows an automatic system to learn valid statistical models of typical behavior and language.

3. AGENT ARCHITECTURE OVERVIEW

We first describe the agent's behavior at a high level, and provide more detail in following sections. The agent is driven to keep the interaction progressing. As the agent observes player actions and utterances, it executes plans in response, where plans consist of sequences of physical actions and/or dialogue utterances. Rather than formulating plans at run-time, agents retrieve predefined plans from collective memory, generated from gameplay logs of previously recorded human-human games. Plans direct the behavior of the agent, and provide expectations about the behavior of the other player. When observations do not match expectations, the agent re-evaluates available plans, and selects the plan that minimizes deviation from the norm. Typicality of behavior is assessed by exploiting preprocessed recurrence patterns acquired by analyzing the human gameplay data. This system enables the agent to pivot from one gameplay log to another, providing contextually appropriate responses as the situation unfolds.

Figure 3 depicts the flow of information through the agent architecture. The agent observes utterances, physical actions, and state changes to the game world through external sensors. Observations from sensory input are stored in an interaction history on a blackboard, shared between the sensory, action selection, and actuation systems. Internal, reflective sensors inspect the blackboard to detect broken expectations in the currently active plan, and may broadcast the need to select a new plan. The action selection system conducts a process of proposal and critique to select a plan. Proposed plans are retrieved from collective memory. A proposal consists of an action or dialogue sequence extracted from a human gameplay log, and an offset index from which to start executing the plan's actions. The index may point to a physical action or utterance, to be executed by the agent or the other player. If the action is to be executed by the other player, it is treated as an expectation that agent waits to observe. Critics scrutinize proposed plans by comparing them to recurrence patterns of human behavior and dialogue learned by

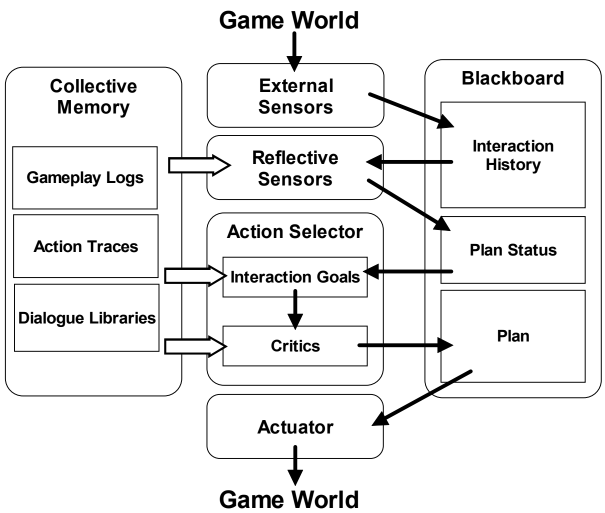

Figure 3. Proposal-critique driven agent architecture.

reflecting on gameplay logs in collective memory. Plans may be rejected for a number of reasons – the next action in the proposed plan is unlikely or physically impossible, or the previous actions in the proposed plan do not match the recent history from the current game. Finally, the actuator executes the next action in the current plan by sending a command to the agent's representation in the game world. A command might instruct an agent to sit on a chair, pick up a steak from the counter, or utter "What are your specials tonight?"

Separating the critique process from the plan retrieval system allows critics to exploit information that collective memory may not be privy to, such as the interaction history, and the agent's believed state of the world. It is conceivable that collective memory could exist on its own server, shared by many agents over the internet. Collective memory might continually incorporate new human gameplay data, and heterogeneous agents could choose to generate different abstractions from this data.

4. RELATED WORK

Von Ahn and Dabbish [15] demonstrated using an online game as a data collection device with the *ESP* game. This game collects one-shot text labels for images by anonymously pairing thousands of players who score points by typing the same word. *The Restaurant Game* yields time-coded logs of 10-15 minutes of interaction, from which we need to mine recurrence patterns from dialogue and action sequences. The incentive to play *The Restaurant Game* is social interaction, and contributing data for a new collaboratively authored game, rather than scoring points. Gorniak, Fleischman, and Roy [5, 7] have collected data from a smaller number of subjects with a role playing game, and hand-annotated human interactions to generate hierarchical models of intention. This paper explores unsupervised generation of behavior and dialogue from thousands of human examples.

Gorin, Riccardi, and Wright [6] have shown that it is possible to learn utterance recurrence patterns given a corpus comparable in size to ours. Their system analyzed spoken dialogue captured from 10,000 customer support calls, and automatically acquired salient phrases found to have high mutual information with 14 call-types. These phrases were used to classify calls for a routing system where humans respond to the open-ended prompt "How may I help you?" While language acquisition from *The Restaurant Game* is simplified by directly capturing typed text rather than speech, recognizing salient phrases is complicated by the fact that unique dialogues occur at hundreds of different branch points in the interaction, and consist of several turns taken by two distinct roles.

Ravichandran and Hovy [13] automatically trained a question answering system by employing a suffix tree [8] to find recurring phrases in thousands of web pages containing the question and answer. Our approach to learning recurring phrases in dialogue iterates over substrings in chat text from 5,000 games, and inserts each string into a suffix tree. We prune the tree and build a dictionary of recurring phrases, used to abstract all lines of dialogue into signatures consisting of sets of dictionary indices.

Our modular agent architecture resembles other agent architectures designed for synthetic characters and planning systems in commercial games [3, 11]. However, rather than generating plans at run-time, our system selects and critiques previously existing plans, abstracted from gameplay logs captured from human-human interactions. Singh and Minsky [14] have proposed an architecture where critics detect when an agent's current approach fails to solve a problem, and suggest alternative ways to think about it. Our critics proactively scrutinize available plans to find one that conforms to social norms and the believed state of the world. Barto, Sutton, and Anderson [1] have also proposed the Actor-Critic method in the domain of reinforcement learning.

An obvious approach to authoring agent behavior would be to adopt an established formalism such as BDI or HTN [2, 9]. While these formalisms provide high-level structure, many questions remain regarding how to build models of behavior that are robust in the face of unpredictable open-ended human interaction and dialogue. In contrast, this paper explores learning the surface-level behavior directly from human-human interactions. Ultimately, a robust agent requires models of both the high-level structure and surface-level behavior.

5. COLLECTIVE MEMORY

The collective memory system stores the library of human gameplay logs, and engages in reflective processes to generate abstractions representing recurrence patterns among logs. These abstractions are generated automatically, and are useful in critiquing proposed plans, and in assessing the likelihood of a currently active plan. The collective memory system also stores the predefined plans themselves, in the form of either *action traces* or *dialogue instances*, defined in sections 5.1 and 5.2. In practice, mining recurrence patterns is performed as a preprocessing step before the agent begins interaction, but the potential exists to mine patterns continuously over the course of the agent's life. Recurrence patterns are stored for both physical action sequences, and dialogue utterances, as detailed below.

5.1 Action Lexicon, Traces, and Trigrams

The *action lexicon* is generated by compiling a list of every unique action observed in 5,000 games. Actions are stored in a STRIPS-like representation [4] with parameters, preconditions, and effects, described in detail in previous work [10]. Each unique action is role-dependent and context-sensitive; for instance a waitress picking up pie from the counter is different from a customer picking up pie from the counter, or a waitress picking up pie from a table while sitting on a chair. We have observed 11,206 unique actions out of over 100,000 possible actions.

An *action trace* is a condensed representation of a gameplay log, consisting of a time-coded sequence of action lexicon indices. Between any pair of action indices, there may optionally be a reference to a dialogue instance that contains utterances observed between these actions. These action traces serve as one form of predefined plans that the agent may select from at runtime through the proposal-critique process described in section 7.

The *trigram database* stores action sequences of length three that have been observed in more than 1% of the action traces. We store a separate list of trigrams for each social role, with 312 trigrams for the customer and 443 for the waitress. Members of the trigrams are indices into the action lexicon. For example, a trigram might be <38, 59, 61> representing a customer putting a menu down on a table, eating food, then drinking a beverage. Along with each trigram, we store the associated observation likelihood.

5.2 Dialogue Instances, Libraries, Signatures

A *dialogue instance* is defined as a sequence of one or more utterances occurring between two physical actions. For example, in Figure 2 the three utterance exchange that occurs between the customer sitting down and the waitress picking up a menu composes one dialogue instance. Dialogue instances serve as the second type of predefined plan that the agent may select from at runtime through the proposal-critique process.

The physical action that precedes a dialogue gives it context. Dialogue instances are clustered by context prior to mining recurrent surface text patterns, to maximize utterance overlap. For instance, we cluster all dialogues observed immediately after a customer sits down. We find recurring phrases by following a procedure similar to that described by Ravichandran and Hovy [13], iterating over each three to five word substring within each utterance, and inserting each substring into a suffix tree [8]. Our suffix tree is simplified by restricting each branch and leaf node to reference only one word. Utterances are enclosed in `#START#` and `#END#` markers, so even a one word utterance becomes long enough to insert into the tree. We build a separate tree for each social role. Each node in the tree keeps track of observed frequency. Leaf nodes observed five times or less are pruned.

After pruning, each tree is flattened into a *phrase library*, consisting of an indexed list of salient phrases. The libraries are generated by adding each root-to-leaf sequence as one phrase, plus each root-to-branch sequence for branch nodes whose frequency exceeds the sum of its children's frequencies by five. In other words, if "Hello" has two children, and we observe "Hello" at least five times more often than the sum of "Hello there" and "Hello sir," we preserve "Hello" as its own phrase. As we add phrases to the library, we cluster them if they share the same number of words, and only differ by one non-marker word. For example, we might cluster "Welcome to *the* restaurant" with "Welcome to *our* restaurant," and assign indices 76.1 and 76.2 respectively. Note that this criterion does allow incorrect clustering, such as "Would you like a *beer*?" and "Would you like a *table*?" Preserving the sub-indices enables the utterance retrieval system to find the best match when discrepancies exist.

We generate one *dialogue library* for each physical action that precedes a dialogue in one or more games. The library contains all dialogue instances observed following the action. Each line of dialogue is tagged with an abstract *signature* composed of the unordered set of phrase library indices for phrases found within the utterance. Phrases referenced in the signature may overlap. The signature for "Hi, how are you today?" will include indices for both hi_how_are_you, and how_are_you_today.

6. SENSORY SYSTEM

The agent has external sensors to detect player actions and utterances, and internal reflective sensors which recognize when plan re-selection is necessary. External sensors receive data from the game world about utterances, actions, and resulting state changes. Observations about these external events are stored on the blackboard in an interaction history, recording all that has transpired since the beginning of the gameplay session.

Internal sensors reflect on recent history updates, and broadcast the need to select a new plan when expectations are broken. We have implemented the following internal sensors.

- **SensorBrokenExpectation** – the observed physical action does not match the next action in the current plan.
- **SensorExpiredExpectation** – too much time has passed while the agent was waiting for some action to occur.
- **SensorFailedActionExecution** – the agent's action resulted in unexpected state changes, or no change at all.
- **SensorInterruption** – a physical action was observed when an utterance was expected, or vice-versa.
- **SensorUnlikelyNextAction** – the action that the agents plans to execute next completes a trigram that does not exist in the collective memory trigram database, and is therefore considered highly unlikely.

7. PLAN PROPOSAL AND CRITIQUE

If the sensors determine that recent observations are aligned with the agent's expectations, and the next action in the plan conforms to typical behavior, the agent can simply continue following the current plan (where a plan is represented as an action trace or dialogue instance, as described in section 5). However, if expectations have been broken, execution has failed, or the current plan is going to lead the agent to misbehave, the agent needs to select a better plan to continue the interaction. Plan selection occurs through a proposal-critique process, where the agent's highest priority interaction goal leads to the retrieval of one or more candidate plan proposals that are validated by a number of critics based on different elimination criteria. One plan is arbitrarily selected to activate from the pool of proposals approved by all critics.

7.1 Interaction Goals

The agent has a set of prioritized interaction goals motivated by the constant drive to keep the interaction progressing. These goals are responsible for retrieving candidate plans from collective memory, which are submitted to critics for approval. A candidate plan consists of an action trace or dialogue instance, along with an associated starting offset within the trace or dialogue. Following is a list of interaction goals, sorted from highest to lowest priority.

- **GoalRespondToUtterance** – Respond to an utterance directed at the agent by speaking or taking a physical action.
- **GoalWaitForInteraction** – Wait to see what transpires next after the human player speaks to someone other than the agent (e.g. the chef or bartender).
- **GoalForceDialogueConclusion** – Execute a physical action to force a dialogue to conclusion, when no response can be found.
- **GoalInitiatePhysicalAction** – Initiate physical interaction after the agent has concluded dialogue with an utterance.
- **GoalInitiateDialogue** – Initiate dialogue with another player.
- **GoalRespondToPhysicalAction** – Execute a physical action in response to the last observed physical action.
- **GoalBeginInteraction** – When all else fails, start over by initiating dialogue contextually appropriate, given the last observed physical action.

Once the highest priority, currently relevant goal is selected, the agent retrieves candidate plans to satisfy the goal, relying on critics to scrutinize the validity of the plans. If the critics do not approve any of the candidates, the selected goal cannot be satisfied, and the proposal process repeats for the next highest priority relevant goal.

The goals that lead the agent to initiate physical or dialogue interaction (GoalInitiatePhysicalAction and GoalInitiateDialogue) are unique, in that these goals each only propose a single candidate plan. These interaction initiation goals become relevant when the currently active plan instructs the agent to initiate a dialogue or physical interaction. For example, in Figure 2 after the waitress says "ok, one second then," GoalInitiatePhysicalAction leads her to execute the action that occurred immediately after this dialogue instance in the original human game – picking up a menu from the podium. Similarly, if the agent was executing a plan of physical actions, and encountered a reference to a dialogue instance, GoalInitiateDialogue will lead the agent to utter the first line of dialogue in the specified dialogue instance (or wait for the other player to speak, in the case of an expectation).

In all other cases, the agent retrieves a list of candidate action traces or dialogue instances to deliberate between. We retrieve dialogue instances and action traces in different ways.

7.1.1 Dialogue Instance Candidate Retrieval

In Figure 2, the human customer says "can I start with a beer?" soon after receiving a menu from the agent. The agent observes this utterance and assigns it a signature representing the recurring phrases found within. The signature consists of phrase indices associated with the dialogue library that corresponds to the last observed physical action – giving the customer a menu. The library assigns the signature <65.1, 34.1>, which translates to <start_with_a, with_a_beer_#END#>. The agent finds candidate responses by searching for the dialogue instances in the same library that contain utterances whose signatures tie for the best match with the input. Candidates are scored based on how many indices match between a pair of signatures, with one point added for each matching cluster index, and an additional point for each matching cluster sub-index.

In our example, the library does not contain any dialogue instances with a perfect match for the words "can I start with a beer?", but several dialogue instances exist that begin with "*may* I start with." Table 1 lists the signatures and respective scores that lead the agent to select dialogue instances containing "may I start with a beer?" as the best candidates. These words do not match exactly, but the signature is a perfect match.

Table 1. Signatures and scores for utterances.

Utterance	Signature	Score
"may I start with a beer"	<65.1, 34.1>	4
"may I start with a salad"	<65.1, 34.3>	3
"may I start with some soup"	<65.1, 38.1>	2

After selecting the best matching dialogue instances, these candidates are further pruned to the group that best matches the dialogue history since the last physical action. Candidates are scored by counting the number of utterances prior to the reference utterance (the utterance that matches the input) whose signatures include phrase clusters found in the current dialogue history. Sub-indices are ignored during history scoring.

7.1.2 Action Trace Retrieval

Retrieving action trace candidates is a simpler process, described through an example. In Figure 2, the agent picks up a beer from the bar, puts it on his table, and expects the customer to initiate a dialogue, which he does by saying "thanks." If the customer broke the agent's expectations by saying nothing, and instead picked up the beer and took a sip, the agent would need to search for plans that align with observations, following a two step process. First, the agent loads the dialogue library that corresponds to the action that preceded the last observed dialogue – giving the customer a menu. Next, the agent searches for all dialogue instances in this library that conclude with the action observed after the last dialogue – a beverage appearing in the bar. All of the action traces that refer to these dialogue instances are retrieved as candidates. Critics are responsible for filtering out the action traces whose histories since the last dialogue do not match recent observations.

7.2 Critics

The agent employs critics, responsible for rejecting proposals that will lead to behavior that is unconventional or impossible. The proposal-critique system iterates over each proposal, and gives each critic the opportunity to reject it. Below are the rejection criteria for the critics we have implemented.

- **CriticUnlikelyNextAction** – The next physical action in a plan is unlikely given the previous two physical actions in the interaction history for the same role.
- **CriticUnmetPrecondition** -- The next physical action in a plan is impossible given the agent's belief about the current state of the world. For example, the customer cannot drink a beverage if no beverage exists.
- **CriticHistoryMisMatch** – The physical actions observed since the last dialogue do not match the physical actions that precede the next action in the proposed plan.
- **CriticRequiredRole** – The player who will act or speak next does not match a requirement set by the goal (and communicated via the blackboard). For example, after action execution fails, or an expectation expires, the agent should immediately try to say or do something else to move the interaction forward. In these cases, we should not approve plans that set up expectations for the human to act or speak.

8. EXECUTION IN THE GAME WORLD

Thus far we have been describing the high-level reasoning that takes place in the mind of the agent. In this section, we describe how the agent actually carries out plans in the virtual environment. The agent executes physical actions and utterances by sending commands to its embodied, animated, physical instantiation in the 3D world. Commands are specified at a high-level, such as:

```
action(waitress, pickup, food, table);
```

The in-game character is responsible for resolving object instances and navigating as necessary. While this does require some hand coding of character behavior, navigation in virtual worlds is a well studied problem, and it is simple to encode heuristics to select contextual objects near the interaction partner. The aspects of interaction that are more difficult to encode by hand are the dynamics of social interaction and dialogue, which our high-level reasoning system guides, by leveraging collective memory.

This separation between high-level social reasoning and low-level instantiation is beneficial for a number of reasons. Separating the high-level reasoning from the low-level implementation details allows the learned model of social interaction to transfer to virtual worlds beyond the game where these behaviors were learned. Models of behavior and dialogue learned in *The Restaurant Game* might control avatars in *Second Life*. Perhaps these models could even guide the behavior of physical robots in the real world. In addition, by abstracting away the details at the high-level, we avoid over-fitting the behavior to a particular restaurant configuration.

9. EVALUATION

We performed a quantitative evaluation of our system by generating data from 100 gameplay sessions, where both roles (customer and waitress) were played by autonomous agents, and compared the output to gameplay sessions between two human players. While our system is designed to interact with humans, our evaluation was intended to measure performance when interacting with a compliant partner over the course of many games. Online players cannot be guaranteed to play compliantly, and a small deviation in agent behavior could throw an otherwise compliant player off course completely. Pitting two agents against each other levels the playing field, and allows us to focus on compliant behavior for this first iteration of the system.

There was no centralized control over agent interactions. Communication between agents occurs through the exact same channels as between a human and an agent, and agents were not permitted to draw data from the same recorded human gameplay session. We ensured that the two agents were not syncing up to follow the same recorded human game by broadcasting a reservation message each time an agent accessed data in collective memory, where the message identified the gameplay log file where the data originated. Agents were not allowed to access data associated with a game reserved by the other agent.

We evaluated gameplay occurring in the first 20 physical actions of gameplay, where an unlimited number of dialogue utterances may occur between any pair of physical actions. On average, each evaluation game contains 24 utterances with a mean length of three words, and a maximum length of 14 words. Each game draws from an average of 24 different recorded human games, with a median of 28 games. In total, the 100 evaluation games drew data from 2,409 different recorded human games.

9.1 Evaluation Metric

We employed the BLEU score [12] from the field of machine translation as our evaluation metric. A parallel can be drawn between evaluating translation, and evaluating two agents dramatizing a restaurant scenario. In the task of machine translation, the machine is asked to translate a sentence from one natural language to another (e.g. from French to English). When physical actions are represented as action lexicon indices, we can represent a gameplay session as one long sentence composed of a mixture of action indices and words, all produced from the same input – vague instructions to play the roles of customers and waitresses. In a sense, the agents are translating their interpretation of the restaurant scenario. Translation can be evaluated by comparing the machine's candidate translation against some number of reference translations provided by humans for the same input. There are many correct ways to translate a sentence, and the machine's translation may be some combination of those provided by humans. In our case, every game plays out differently, but we expect significant portions of games to consist of behavior and dialogue that we have observed previously in human games.

The BLEU score is a modified measure of precision between a candidate translation and a corpus of reference sentences. The simple unigram precision of a candidate sentence is the ratio between the number of words in a sentence that can be found in at least one reference sentence, to the total number of words in a sentence. Note that precision can be measured for larger n-grams by iterating over the words of the sentence and treating the next n words as one unit. In Table 2, the simple unigram precision of the candidate sentence is 7/7 = 1.0. The BLEU score modifies the precision by limiting the total count of a word by the maximum number of times that word appears in any single reference. In our example, the word "the" appears at most twice in any reference, so the modified precision is 2/7.

Table 2. Poor machine translation with high precision.

Candidate	the	the	the	the	the	the	the
Reference 1	the	cat	is	on	the	mat	
Reference 2	there	is	a	cat	on	the	mat

We evaluated our agent interactions by using the BLEU score to compute the modified trigram precision of an agent-agent game, as compared to a corpus of human-human reference games. From our original pool of 8,430 completed games, we randomly selected 5,000 training games and 3,000 test games, with no overlap between training and test games. The training games formed the basis of the agents' collective memories, while the test games provided references for computing the BLEU score. Driven by the training corpus, we generated 100 agent-agent games for evaluation.

9.2 Experimental Results

Figure 4 illustrates a comparison between the histograms of sorted BLEU scores computed for 100 agent, human, and randomly generated games, all compared with the same corpus of 3,000 reference human games. The mean BLEU scores of human and agent behavior are very close, 0.55 and 0.6 respectively, with the agent's mean slightly higher. Variance of human scores is greater,

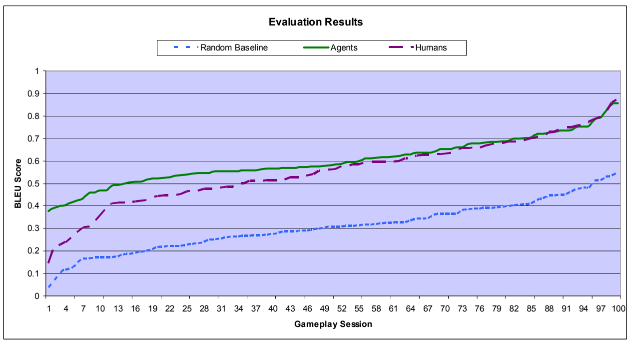

Figure 4. Comparing BLEU score histograms for 100 games.

```
CUSTOMER: "hi"
WAITRESS: "hello sir"
CUSTOMER: "can i get a table ?"
WAITRESS: "sure, take a seat."
CUSTOMER: "thanks"
CUSTOMER SITSON chair6(Chair)
WAITRESS: "i recommend the a chef's tasting menu"
CUSTOMER: "yes please"
WAITRESS: "Great I'll be right with you."
WAITRESS PICKSUP dyn029(Menu) FROM podium(Podium)
CUSTOMER STANDSUP FROM chair6(Chair)
WAITRESS GIVES dyn029(Menu) TO CUSTOMER
CUSTOMER SITSON chair6(Chair)
CUSTOMER LOOKSAT dyn029(Menu)
CUSTOMER: "can i get a cobb salad and a glass ...
CUSTOMER PUTSDOWN dyn029(Menu) ON table3(Table)
WAITRESS: "lobster please"
dyn078(Lobster) APPEARS ON counter3(Counter)
WAITRESS: "water please"
dyn086(Water) APPEARS ON bar(Bar)
WAITRESS PICKSUP dyn078(Lobster) FROM counter3(Cou
WAITRESS PUTSDOWN dyn078(Lobster) ON table3(Table)
WAITRESS PICKSUP dyn086(Water) FROM bar(Bar)
CUSTOMER EATS dyn078(Lobster)
WAITRESS PUTSDOWN dyn086(Water) ON table3(Table)
CUSTOMER EATS dyn078(Lobster_Bite1)
CUSTOMER EATS dyn078(Lobster_Bite2)
CUSTOMER EATS dyn086(Water)
WAITRESS: "will your guest be joining you soon?"
WAITRESS PICKSUP dyn078(Plate)
WAITRESS PUTSDOWN dyn078(Plate) ON counter4(Counte
CUSTOMER: "i'm not sure.i'll go ahead and order...
WAITRESS: "Ill get your bill"
CUSTOMER: "would u like to drink with me"
WAITRESS TOUCHES cash_register(Register)
dyn216(Bill) APPEARS ON podium(Podium)
WAITRESS PICKSUP dyn216(Bill) FROM podium(Podium)
WAITRESS GIVES dyn216(Bill) TO CUSTOMER
WAITRESS LOOKSAT dyn216(Bill)
WAITRESS: "please pay"
CUSTOMER: "nicee!"
WAITRESS: "ok"
CUSTOMER: "one for me and one for you"
WAITRESS: "water please"
dyn253(Water) APPEARS ON bar(Bar)
WAITRESS: "beer"
dyn260(Beer) APPEARS ON bar(Bar)
WAITRESS: "water"
CUSTOMER: "did you know they dont have whiskey ...
dyn268(Water) APPEARS ON bar(Bar)
```

Figure 5. Agent interaction with highest BLEU score.

as humans are given greater freedom of expression than the critics allow the agents. Standard deviation of human BLEU scores is 0.15, compared to 0.1 for agent scores. Human games score both higher and lower than any agent games.

Both human and agents consistently score significantly better than the random baseline. Random baseline games are constructed by stitching together fragments of randomly selected games, until each game contains at least 20 physical actions, and an unlimited number of utterances between actions. We a repeated process of randomly selecting one of the 5,000 training games, picking a random starting offset into the game, and copying the next three interactions. Each interaction could be a physical action index, or a complete utterance.

9.3 Discussion and Future Work

It appears as if our agents are behaving as well or better than human players, but the BLEU score does not give the complete picture. Figure 5 details the interaction in the agent game with the highest BLEU score, 0.85. In this gameplay session, we can see what the BLEU score captures, and what it cannot recognize. Simultaneously, we see the strengths and weaknesses of the implemented system, which parallels those of the BLEU score.

On the surface, the agents are saying and doing what we would expect from players of these roles. This is significant given the enormous variety of language and interaction possible in this simulation, as illustrated by the poor performance of our random baseline. Focusing on any fragment of the interaction, we see apparently coherent dialogue and physical behavior. When we look at the big picture, however, a human observer can recognize a number of issues -- customers normally order an entrée only once, the waitress brings a menu after the customer agreed to the tasting menu and later brings lobster after the customer orders salad, the waitress brings the bill after the customer request to order food (again), and the waitress enters a loop of ordering many drinks after giving the customer the bill.

The BLEU score validates that our system has preserved the *texture* of human social interaction, in terms of meaningful low-level local coherence, while the metric's limited three symbol perspective is incapable of detecting issues in the higher-level intentional structure of the gameplay session. Our current implementation has no notion of goals, aside from continuing the interaction, and has no representation of variables to bind to language. In other words, the BLEU score gives a fair evaluation of our current implementation, because it is as unaware as our system is that "salad" and "lobster" refer to different objects, and can be used in directives to satisfy a customer's desire.

```
WAITRESS: "how are you"
CUSTOMER: "Pretty good, thank you"
WAITRESS: "fine"
CUSTOMER: "how about yourself"
WAITRESS: "i am from brazil"
```

Figure 6. Example of non-sequitur in agent dialogue.

In other gameplay sessions we find dialogue sequences where transitions between utterance pairs are reasonable, but as a whole yields a non-sequitur. The last utterance pair in Figure 6 does not make sense when the previous history is considered. While the dialogue retrieval system does its best to match previous history, it has no understanding of the semantics of the interaction. Utterances are currently abstracted into collections of low-level surface text patterns, void of any representation of intent or content.

We have identified a number of issues with the current implementation that point the way forward for future work. Agents need a representation of the intentional structure of the scenario, describing the partially ordered sequence of events in terms of high-level goals, which can be accomplished by dividing goals into sub-goals, primitive actions, and utterances. Our utterance representation needs to be augmented with the intent and content of the words in order for utterances to be useful in planning to satisfy goals, and we need some means of associating words with concepts. In other words, agents need to understand that "Can I get a cobb salad" is intended as a directive referring to salad, where salad describes the dish of leafy greens that agents interact with in the world. If our collective memory system was augmented with additional representations of structure and semantics, we could leverage these abstractions in the existing agent architecture by implementing new critics responsible for higher-level cohesion. These representations may be generated through a combination of automated and semi-automated approaches. Semi-automated processes include a human in the loop to annotate some portion of the data, which can then train systems to annotate the rest.

10. CONCLUSION

There are numerous opportunities for virtual or robotic agents that can assist, entertain, or educated us in our homes and workplaces. All of these agents require humans to author their physical and communicative behaviors, and today this requires specialized skills. In this paper, we have demonstrated that it is possible to automatically extract the texture of human interaction given a corpus of thousands of human gameplay logs. Despite the apparent freedom given to players of *The Restaurant Game*, we find that humans tend to naturally restrict their language and behavior when presented with a familiar environment, providing valuable data that can be mined to direct agents. The first iteration of our agent automatically learns to imitate the texture of human dialogue and interaction in a restaurant. This work provides a promising first step towards realizing Collective AI driven agents that can interact and converse with humans without requiring programming or specialists to hand-craft behavior and dialogue. While we acknowledge that we cannot build robust agents based purely on models of surface-level behavior, this texture of interaction is an important piece that is difficult to capture with hand-crafted models of behavior. Future work remains to provide agents with representations of intentional structure and semantics of language. As a step in this direction we are currently developing a speech act classifier that will allow us to model dialogue at goal-driven, intentional level.

11. REFERENCES

[1] Barto, A. G., Sutton, R. S., & Anderson, C. W. 1983. Neuronlike elements that can solve difficult learning control problems. IEEE Trans. on Systems, Man, and Cybernetics, 13, 835 – 846.

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

[3] Burke, R., Isla, D., Downie, M., Ivanov, Y., and Blumberg, B. 2001. CreatureSmarts: The art and architecture of a virtual brain. Proceedings of the Game Developers Conference, 147-166. International Game Developers Association.

[4] Fikes, R.E. and Nilsson, N.J. 1971. STRIPS: A new approach to the application of theorem proving to problem solving. Artificial Intelligence, 2(3-4), 189-208.

[5] Fleischman, M. and Roy, D. 2005. Intentional context in situated language learning. Ninth Conference on Computational Natural Language Learning.

[6] Gorin, A., Riccardi, G., Wright, J. 1997. How may I help you? Speech Communication, Volume 23. Elsevier Science.

[7] Gorniak, P. and Roy, D. 2007. Situated language understanding as filtering perceived affordances. Cognitive Science, 31(2), 197-231.

[8] Gusfield, D. 1997. Algorithms on Strings, Trees and Sequences: Computer Science and Computational Biology. Chapter 6: Linear time construction of suffix trees, 94–121. Cambridge University Press.

[9] Nau, D., Cao, Y., Lotem, A., & Mu˜noz-Avila, H. 1999. SHOP: Simple hierarchical ordered planner. In Proceedings of IJCAI'99.

[10] Orkin, J. and Roy, D. 2007. The Restaurant Game: Learning social behavior and language from thousands of players online. Journal of Game Development, 3(1), 39-60.

[11] Orkin, J. 2005. Agent architecture considerations for real-time planning in games. Proceedings of Artificial Intelligence & Interactive Digital Entertainment. AAAI Press.

[12] Papineni, K., Roukos, S., Ward, T., and Zhu, W. J. 2002. BLEU: A method for automatic evaluation of machine translation. ACL-2002: 40th Annual meeting of the Association for Computational Linguistics. 311-318.

[13] Ravichandran, D. and Hovy, E. 2002. Learning surface text patterns for a question answering system. In Proceedings of the 40th ACL Conference.

[14] Singh, P. and Minsky, M. 2003. An architecture for combining ways to think. Proceedings of the International Conference on Knowledge Intensive Multi-Agent Systems.

[15] von Ahn, L., and Dabbish, L. 2004. Labeling images with a computer game. Proceedings of the SIGCHI conference on Human factors in computing systems.