A Development and Teaching Framework for Codenames

Robert Morain
Computer Science Department
Brigham Young University
Provo, UT 84602
rmorain2@byu.edu

Brad Spendlove
Department of Computer Science
Randolph College
Lynchburg, VA 2450
bspendlove@randolphcollege.edu

Dan Ventura
Department of Computer Science
Brigham Young University
Provo, UT 84602
ventura@cs.byu.edu

Abstract

A significant challenge associated with developing creative agents is the necessity of reconciling tractable evaluation metrics with the subjective nature of creativity. Traditional approaches often rely on bespoke methodologies that require significant abstraction and domain-specific engineering, creating barriers to rapid iteration. Language games like Codenames offer an alternative paradigm by providing a structured environment with inherent evaluation mechanisms, enabling clearer feedback loops for agent development. This work introduces an open-source resource designed to streamline creative agent development for the game Codenames. The resource consists of a web application for real-time game simulation and evaluation and a modular Python client that supports both spymaster and guesser agent roles. Initially presented at an *ICCC24* workshop, the system was refined through a code jam session with computational creativity researchers before being deployed in a study set in a graduate-level computational creativity class. Validation with 25 students demonstrates the effectiveness of the resource as both a development tool and an educational instrument. Quantitative and qualitative analysis of a user survey shows that the resource facilitates meaningful iterative improvements, exploration of approaches to clue generation and semantic reasoning, and application and understanding of fundamental CC concepts.

Introduction

Methods for the external evaluation of creative computational (CC) systems have long been the subject of debate, and thinking on the subject continues to evolve (Ritchie 2007; Pease and Colton 2011b; Jordanous 2012; Bown 2014; Lamb, Brown, and Clarke 2015; Ventura 2016; Carnovalini et al. 2021; Peeperkorn, Brown, and Jordanous 2023). In a significant sense, this question of how to evaluate CC systems is foundational in that it plays a large role in specifying the goals of the field and what it means to accomplish them. Historically, many evaluation methods have aimed to measure inherently subjective creative attributes, such as quality, novelty, typicality, and surprise, whose interpretations and relative importance vary across individuals and domains. Given that accurate evaluation is critical for advancement and maturation of the field, the limitations imposed by the approximation of subjective features represent a major challenge. To address this issue, Spendlove and Ventura proposed that competitive language games could serve as creative tasks with well-defined goals (2022). The win/loss outcome of the game eliminates the need to approximate subjective measures by providing an objective proxy measure of creativity. The well-defined goals of the game do not reduce the need for creativity on the part of the system. Rather, the constraints of the task provide an opportunity to highlight creative gameplay.

In particular, the authors explore the design and evaluation of CC systems with the game Codenames (Chvátil 2015), a word-based guessing game. In the game, two teams of players are tasked with identifying which of 25 cards, drawn at random from a large deck, belongs to their team. One member of each team serves as the spymaster. They are given a secret key that shows which words belong to each team. The teams then take turns in which the spymaster gives a one-word clue, accompanied by a number, that attempts to communicate a subset of the team's words to their teammates. The clue word must relate to the meanings of the word cards, and the clue number represents how many cards are intended to be related to the clue. Their teammates then guess one card at a time. If they correctly guess a word that belongs to their team, they may continue guessing, otherwise the turn is over. Wrong guesses may reveal opponents' words or even instantly lose the game. The challenge for the spymaster lies in the puzzle of coming up with a clue that strongly relates to their team's cards without unintentionally relating to any others.

Creative domains are characterized by their extremely large combinatorial spaces. Playing the spymaster role constitutes a creative task given the vast number of possible solutions (semantic graphlets that indicate positive and negative relationships between candidate clue words and the target words to be guessed) and the complexity of language inherent in the gameplay. Designing an agent to play the game is challenging, as it requires both a deep understanding of language and potentially some kind of theory-of-mind-like model of the way teammates understand and think about language. The agent must employ creativity to efficiently select clues from a large solution space for each possible game state in a (typically) limited amount of time. These requirements position Codenames as a challenging creative

task that can serve as an ideal mechanism for the evaluation of creativity through gameplay (Spendlove and Ventura 2023).

To kickstart the research community's interest in games as creative tasks, an introductory workshop was held as part of *ICCC24*. In this workshop, participants were introduced to the necessary theory and mechanics of gameplay before being invited to modify a provided agent or create a new one. A web application with a Python client developed for the workshop helped participants build and evaluate their agents. This enabled participants to examine the behavior of an agent, come up with new ideas for how to improve it, and efficiently build a working prototype of their solution. The workshop outcomes provided further evidence of the usefulness of Codenames as a platform for CC research.

During the workshop, participants demonstrated the ability to successfully and rapidly design, implement, and iteratively improve a creative agent to play Codenames using the resources provided. This inspired the idea to evaluate the resource's effectiveness in CC education. To do this, a study was developed for a graduate-level introductory CC class. In the study, students were assigned to develop a creative agent using an improved version of the Codenames resource developed for the workshop. Students then engaged in an iterative development process, during which they had the opportunity to evaluate their agents in a tournament, identify areas for improvement, and redesign their agents. Finally, students participated in a second tournament to evaluate their improved agents. Throughout the experience, students completed two surveys designed to gain insight into the effectiveness of the resource for the development process and to gauge student's understanding of fundamental CC concepts.

Motivated by the positive experiences of the workshop participants and graduate student study participants, this paper addresses the need for expanded access to game-based creative evaluation metrics by publishing this open-source resource for building and evaluating agents that play Codenames. We anticipate that this resource will serve the research community in many ways, and we put forward these two use cases as examples of its utility. We have found that the game's engaging nature draws people in, sparks ideas, and motivates them to design agents to play it. This framework leverages those unique qualities to make the development process both enjoyable and effective in research and pedagogical settings.

This paper contains a report on the workshop, a description of the Codenames framework and web application, the results of the study, and discussion of future use cases and applications. By providing a practical tool for game-based creativity research, this work aims to expand access to well-defined evaluation metrics and encourage further exploration of competitive language games as a testbed for CC systems. All of the code used for this paper is available on Github.[1]

[1] `https://github.com/rmorain/codenames-ai-client` and `https://github.com/rmorain/codenames-workshop`.

Workshop Report

The purposes of the workshop were to introduce our work to the broader research community, engage in discussion of the creativity of gameplay and game-playing agents, and host a Codenames agent code jam. To support these activities, we developed a programming and UI framework for playing Codenames with human and AI players. The framework includes a default spymaster agent capable of generating clues for any Codenames game state without additional setup. The game UI is a user-friendly web interface that makes it easy to create and play Codenames games. These tools facilitated explanation, discussion, and experimentation at all levels of the workshop.

At the beginning of the workshop, we played a sample game of Codenames with the participants via the web application. The web application automates game rules such as turn passing and scoring, which helped the participants quickly understand the spymaster and guesser tasks. After playing a single game of Codenames, the participants already had ideas for how to develop AI agents to play the game. We discussed how they, as human players, approached clue-giving and guessing, and how those strategies could inform the design of an automated agent.

A large portion of the workshop was dedicated to a code jam, in which participants developed their own Codenames agents. Participants leveraged our Codenames framework and UI to quickly prototype working agents that could play against humans and one another. This allowed them to actually implement and test the ideas they shared in the discussion and see how they performed in the game. Their agents' designs included strategic selection of which subset of cards to relate to a clue, querying large language models to generate clues, and clustering the different colored cards using k-means. Participants reported that this exercise was an enjoyable and interesting way to interact with a creative task that was new to them and pointed to the framework as enabling this positive experience. The framework allowed them to focus on the core task of developing an agent that can create clues, without spending unnecessary time writing boilerplate or interface code.

In our concluding discussion, participants shared their takeaways from the workshop. We discussed whether playing Codenames is a creative task, the potential benefits of studying games as creative tasks, and how Codenames' focus on language could cross over to other language-based creative domains. We were pleased by the positive reaction to the workshop and encouraged that our Codenames framework facilitated rapid prototyping and experimentation.

Resource Description

The code base for the Codenames development framework includes a game server, a web client with a graphical user interface (UI), and Python clients for the spymaster and guesser roles. Both the web UI and the Python clients connect to the same game server, allowing for gameplay with any combination of human players and AI agents in the various game roles. The Python clients each wrap an agent program that makes the gameplay decisions, marking a clear

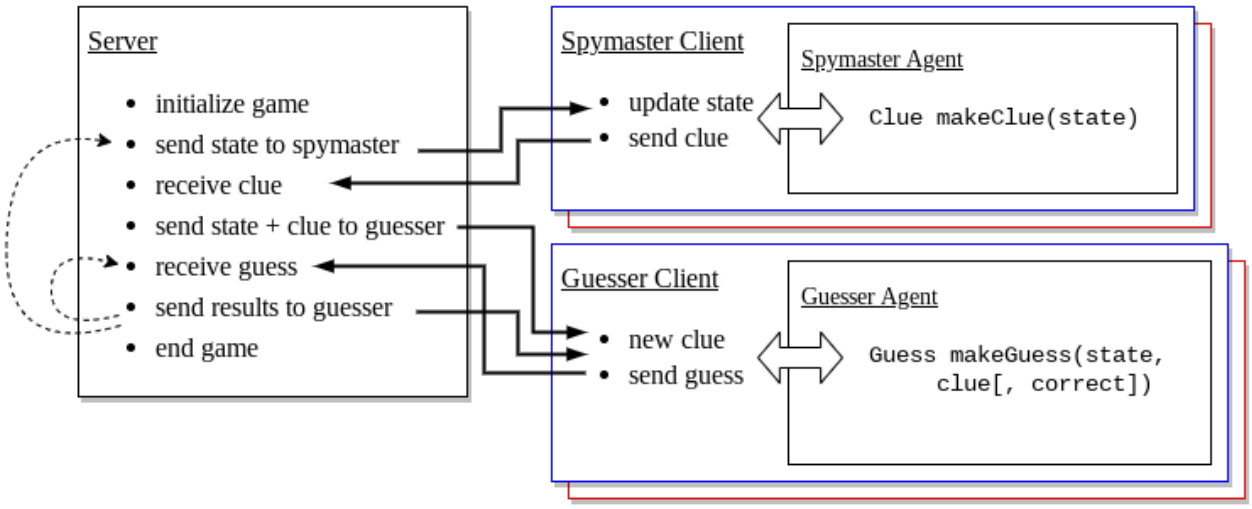

Figure 1: The server/client/agent architecture of the workshop codebase, including information flow between the components. Note that there are two spymasters and guessers, one pair each for the red and blue teams.

division between the creative agent code and the server interface code.

The primary use cases for this code base are playing Codenames as a human player and modifying the agent code to implement new creative play agents. Neither of these requires knowledge of or modification to the client/server code. We host a public instance of this server here: `https://mind.cs.byu.edu/codenames`, where anyone can connect and play via browser or client.

The backend development framework is divided into five components: the server that runs games, a spymaster agent, a spymaster client, a guesser agent, and a guesser client. The two types of clients facilitate communication between the agent and the server. See Figure 1 for a diagram of the complete architecture, including information flow between components. These components are described in detail in this section, and some gameplay details that were left out of the description of Codenames above are included here for completeness.

The server tracks the state of the game in progress, communicates that state to the clients, listens for their moves, validates those moves, and updates the internal game state. This state can be retrieved by either the web UI or the client code. This section focuses on the latter. Codenames is not a complex game to represent computationally. At the beginning of a new game, the board of 25 word cards is drawn from the deck of 400 (the list of which is available online). A coin is flipped to decide which team goes first, red or blue, then the first team is randomly assigned 9 cards and the other 8. The remaining cards are neutral, except for one assassin card chosen at random. This completes the starting board state, and the game is ready to play.

Each new game is assigned a unique four-letter code that clients use to join that specific game session on the server. Four clients must connect to the server: the two spymasters and the two guessers. When playing the game, each team may have any number of guessers, but from an information perspective they act as one unit when choosing words to guess, so one client suffices.

The server sends the game state to all clients, including the secret key to the spymasters, and waits for the current team's spymaster to return a clue consisting of one word and a number. The server then sends the clue to the guesser client. The guesser can guess one word card at a time, after which the server updates the game state by revealing the hidden identity of that word card (red, blue, neutral, or assassin). If the guess is correct (revealed to belong to their team), the guesser may guess again, up to a maximum of one plus the clue number. If the guess is revealed to be the assassin, the game immediately ends, and the guessing team loses the game. Otherwise, if the guess is incorrect or the team is out of guesses, play passes to the other team and the process repeats.

Thus, the server requires the following functions:

- Initialize game, update state
- Transmit game state to current spymaster
- Receive clue from spymaster, update state
- Transmit game state and clue to guesser
- Receive guess from guesser, update state
- Transmit result of guess to guesser
- End game, update state

The clients handle communication between an agent and the server, so the clients' required functions mirror the server's. The spymaster client receives the game state from the server and passes it to the spymaster agent, which returns a clue word and number. The client transmits that to the server and waits until it hears from the server again to get the next clue.

The guesser client has two similar information flows. In the first, it receives the clue and current game state from the server, passes them to the guesser agent, gets a guess back, and transmits it to the server. In the second, it receives the result of the previous guess from the server and passes it to the agent, sending another guess back if necessary.

The spymaster client requires the following functions:

- Receive updated game state
- Submit clue

And the guesser client requires these functions:

- Receive game state and new clue
- Submit guess

The agents' functions follow from the description of their clients. The spymaster agent has a `makeClue` method that accepts a game state and returns a clue. The guesser agent has a `makeGuess` method that takes a game state and clue and returns a guess. The basic game state data consists of the list of word cards and a list of the revealed identity of each word card (if any). Additionally, the spymaster receives the secret key showing all the cards' colors and the guesser receives the current clue and number of guesses remaining. The clients are very thin wrappers around the agents but are engineered this way to allow for simple modification to or replacement of the underlying agent. This separates the client/server communication logic from the agent so that its developers can focus only on the relevant creativity task.

To run the provided automated clients, the user only needs to provide the game code and team color. The client connects to the game server, then repeatedly polls the current game state, calling upon its agent module to generate a game action when required.

Finally, the code base also includes a web UI client for human gameplay that connects to the server using the same interface as the automated clients. This client presents a UI that mimics the layout of the physical game and through which users can input the required game actions. The game state is represented by a grid of word cards that are either uncovered, showing the word, or covered with the appropriate color after they're guessed. One client is used for both the spymaster and guesser role by either prompting the spymaster for a clue word and number or allowing the guesser to click on a word card to guess it when prompted. This simple interface is low overhead and allows human players to play with the AI agents, bringing more richness and interactivity to the application. The web client also provides an observer mode which simply displays the game state without allowing for any game action input. This is useful, for example, for displaying the game to spectators while others play the game.

The entire code base is public, allowing future users to reuse or modify any portion necessary. The server can be run locally, or games can be run on our public server. At the same time, the code base is designed to require only minimal modification to run a custom gameplay agent. The programmer must only replace the appropriate agent module within the client. They will then be able to play their agent against humans or other agents without modifying the client, server, or web UI code.

Example agents

To complete the code base, we include two basic gameplay agents, a spymaster and a guesser. These agents allow the clients to be run out-of-the-box without writing any code, and they serve as a reference implementation for the simple client-agent interface. Furthermore, the agents were designed to be simple to understand even for relatively inexperienced programmers, providing a jumping-off point for their own modification and improvement.

Both example agents are powered by word embeddings, which are vector representations of words that capture their semantic meaning (Mikolov et al. 2013). These high-dimensional vectors enable computers to understand and process natural language to facilitate many tasks such as sentiment analysis, machine translation, and text classification. There exist straightforward analogues between vector operations and the cognitive task of comparing the relationships between words, which makes them a natural fit for Codenames agent development.

The cosine similarity between two word embedding vectors corresponds to the semantic similarity of the words those vectors represent (Orkphol and Yang 2019). By comparing the similarity of a clue word candidate to the word cards, the spymaster agent determines whether that clue will effectively relate to those cards. This information is used to inform the choice of a clue that is similar to the team's words and dissimilar from the others. Conversely, the guesser agent calculates the similarity between a given clue and the word cards and guesses the one(s) that most closely relate.

These simple agents are complete agents in that they can play the game, and they use a reasonable natural language understanding approach to do so. They perform reasonably well in-game, but may often generate unclear clues. Of course, this is by no means the only way to design these agents, which is our motivation for publishing this code base. These starter agents will serve the needs of users with widely differing programming skills and AI or NLP backgrounds.

CC Pedagogy Study

Computational creativity in education is beginning to gain traction, both as an approach to teaching creativity and computational thinking (Kakavas 2019; Fragapane and Standl 2021; Yee-King, Fiorucci, and d'Inverno 2024) and as a first-class pedagogical subject itself. In particular, Ackerman et al. discuss the importance of CC education in the broader context of AI and provide guidance for instructors on how to approach CC pedagogy. Further, they suggest the benefit of students engaging in introductory assignments to reinforce the material and develop their own ability to develop creative systems (2017). Ventura provides an introductory guide for how students (as well as researchers) might approach building such CC systems by identifying their essential components (2017). This Codenames framework advances these pedagogical objectives by providing

students with a structured environment for developing creative agents while avoiding the complexity of subjective evaluation methods.

To validate the effectiveness of this Codenames framework in an educational setting, we conducted a study with students in a graduate-level CC course, examining the feasibility and efficacy of the resource as a tool for teaching fundamental CC concepts and providing structure for creative agent development. In what follows, we outline the study's objectives; describe how the resource was incorporated into the course curriculum; analyze outcomes based on two student surveys; and discuss key lessons learned from the experience.

Study goals

The study aims to assess the pedagogical value of the Codenames resource for

1. helping students develop their own (creative) agents
2. helping students learn and apply CC concepts

The resource is designed to help students develop their own agents by providing clear goals and tools to achieve success. It establishes a well-defined creative task with a clear evaluation metric inherent in the game. Students can easily evaluate the effectiveness of their spymaster agent by directly playing with it in the guesser role through the web UI or by running a provided test that shows the spymaster's response to a random game state. For this study, the effectiveness of the Codenames resource is evaluated through its capacity to facilitate a) students' development of their initial creative agents and b) their identification of ways to iteratively improve those agents.

The resource has been developed in a modular way that allows students to consider and implement different aspects of a creative system, providing a natural introduction to fundamental computational creativity concepts that might be covered in a course on the subject. Success is measured by students' ability to analyze their work through the theoretical frameworks presented in course readings.

Study methodology

The study was conducted during the first two weeks of a graduate-level CC course of 25 students that met twice per week (the entire study spanned a total of five class periods of 75 minutes each). The format of the study was as follows:

Period 1: Introduction of framework and codebase and discussion of study outline and goals. Students were introduced to the game Codenames and the basic features of the Python client, the example spymaster agent, and the web UI and were instructed on how to modify the provided base spymaster agent or create a new one.

Period 2: Class discussion of CC concepts from the following papers: (Wiggins 2006; Ritchie 2007; Ventura 2017; Colton, Charnley, and Pease 2011; Pease and Colton 2011a) and how they might be related to the task of building a creative/competitive spymaster agent.

Period 3: First in-class tournament, enabling students an initial evaluation of their agents' performance through direct competition with their peers' implementations. In each tournament match, students competed in one-on-one games where they played as guessers via the web interface while their agents served as spymasters connected via the Python client running on their personal computers. The tournament consisted of an initial group stage followed by a knock-out round, featuring the winners from each group, with two semi-finals and one finals match. The group stage consisted of 3 groups of 6 (student/agent) teams and one group of 7 teams and featured round-robin play, so that every team faced every other team in a match. As a result, each student/agent team played (at least) 5 matches; four teams played (at least) 6 matches; and two teams played (at least) 7 matches. Group matches were run concurrently, while knock-out matches were run consecutively and broadcast so the entire class could watch and participate. The students completed a post-tournament survey in which they described their implementation approaches, reflected on their agents' performance, and made plans for improving their agents. Students were assigned to continue working on their agents in anticipation of a second tournament.

Period 4: Class discussion of general "questions for CC researchers", another way to orient the students to important ideas in the field.

Period 5: Second in-class tournament, identical format to the first one. A second follow-up survey asked students to report the extent and effectiveness of the modifications they made to their agents and reflect on their experience with the Codenames project.

Survey Questions

Below are the questions included in the two post-tournament surveys. The first survey focused on the students' agent design and performance. The second survey followed up on the modifications the students made and their resulting changes in performance and asked students to reflect on their experiences. Questions marked with a dagger (†) were answered with a 5-point Likert scale, all others were free-response.

Post-Tournament Survey 1

- How far did your agent advance in the tournament?
- What technique(s) does your agent use to determine relationships between words (e.g., word embeddings, knowledge graphs, co-occurrence counting)?
- How does your agent measure or score the strength of word relationships?
- How does your agent identify and evaluate potential clue words?
- What criteria does your agent use to balance between covering multiple target words versus avoiding opponent and assassin words?
- How does your agent determine the optimal number of words to target with each clue?

- What factors do you believe contributed most to your agent's performance in the tournament?
- What specific concepts or approaches from our course readings have influenced or could improve your agent's design? Please reference at least one reading in your response.
- How will you improve your agent for the next tournament?
- How satisfied are you with your overall learning experience with this project?†
- Do you have any suggestions for how your learning experience could be improved?

Post-Tournament Survey 2

- How far did your agent advance in the tournament?
- Rate the extent of changes made to your agent since the first round:†
- What specific modifications did you make to your agent? For each modification you made, please: 1) Describe the original implementation, 2) Explain what you changed, 3) Share why you made this change, 4) Indicate if the change was successful
- Which of these modifications had the most significant impact on your agent's performance?
- How effective were the insights from the first round in improving your agent? (Scale: Very ineffective to Very effective)†
- Describe the most significant lesson from the first round that influenced your agent's improvement
- What do you think about Codenames as a creative task from the spymaster's perspective? How do the game's constraints facilitate or inhibit creative play?
- To what extent does your agent demonstrate the following aspect of creativity: Novelty†
- To what extent does your agent demonstrate the following aspect of creativity: Value†
- To what extent does your agent demonstrate the following aspect of creativity: Intentionality†
- Do you think your agent is creative? Why or why not?
- How would you rate the client's (the provided code) effectiveness in supporting agent development?†
- How would you rate the web application's (the UI) effectiveness in supporting agent development?†
- What suggestions do you have for improving the client or the web application?

Study results and analysis

Initial survey responses revealed two primary approaches to word association: embedding-based methods (utilizing word2vec, BERT, GloVe, or spaCy) and large language models. Agent strategies generally focused on basic similarity metrics between clue words and targets while avoiding opponent words and the assassin card. While students demonstrated general comprehension of course readings, many struggled to meaningfully connect theoretical concepts to their agent implementations. When asked how they would improve their agents, students mentioned adding a history to the agent to avoid repeating words, using LLMs, multiple LLMs, or reasoning LLMs. One student who used an LLM in the first round reported that strong word association skills were not sufficient without enhanced strategy: "I had a superior or competitive method with exception that it did not protect me from the assassin." Another student suggested that because many students wanted to replace the example agent with an LLM-based one, they would have "loved to learn some prompt engineering techniques or techniques to help LLMs be more creative."

The winner of the first tournament developed an agent that provided clues that reveal the coordinates of their own team cards. While this type of gameplay violates Codename's official rules,[2] the experience provided the opportunity for students to engage in debate over how an agent's performance in the game relates to the creativity of the agent. In this case, the cheater agent communicates with the guesser using a cleverly designed code, in which the first letter of the word corresponds to the column and the vowels indicate the rows that contain team words. Students agreed that while the agent itself did not possess any creative ability, the student who developed the agent and its bespoke code exhibited a high level of creativity (cf. the discussion on game rules/design and their intended effect on creativity in (Spendlove and Ventura 2023)). Students also noted that, of course, the cheating spymaster agent is only useful for a guesser that knows its special code, while their general approaches to making agents that use semantic relationships between words as the game rules intend may generalize well to many other guessing partners.

The second survey revealed substantial evolution in agent design, with students implementing more advanced approaches, including the use of reasoning LLMs, larger LLMs, proxy agent testing, enhanced embedding metrics, iterative clue refinement, and prompt engineering. The students succeeded in using the provided code as a jumping-off point for building better agents. As one student reported in the survey "I feel proud I got it working and it did an okay job, better than word2vec, which was a success."

Finding connections between class readings and the agent development assignment resulted in significant improvement in at least one case. In the first survey, one question asks students to identify specific concepts from class readings that could help improve their agent's design. One student connected the aesthetic evaluation described in the FACE model (Pease and Colton 2011b) to a potential internal proxy evaluation that evaluates clues before submitting them to the guesser. After implementing this idea, the student advanced to the final round of the second tournament. This is just one example of how creative agent development

[2]Specifically, it violates the rule that states "Your clue must be about the meaning of the words. You can't use your clue to talk about the letters in a word or its position on the table." (Czech Games Edition 2015)

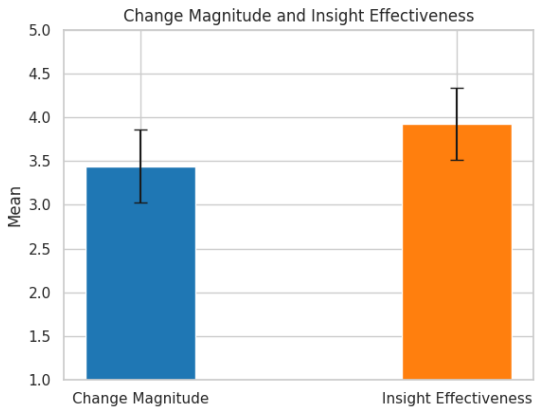

Figure 2: Mean responses from the second survey of students asked, "Rate the extent of changes made to your agent since the first round" (scale: minor tweaks to complete redesign) and "How effective were the insights from the first round in improving your agent?" (scale: very ineffective to very effective).

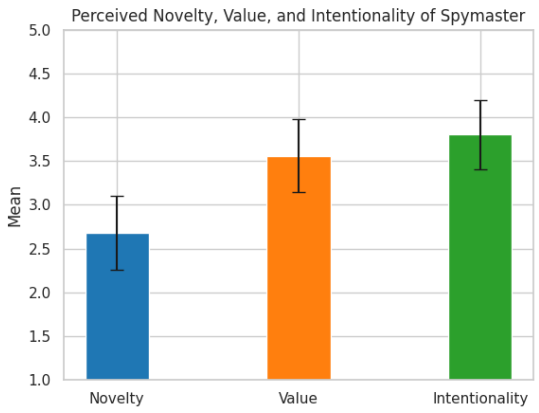

Figure 3: Responses when students were asked, "To what extent does your agent demonstrate the following aspects of creativity?" in the second survey.

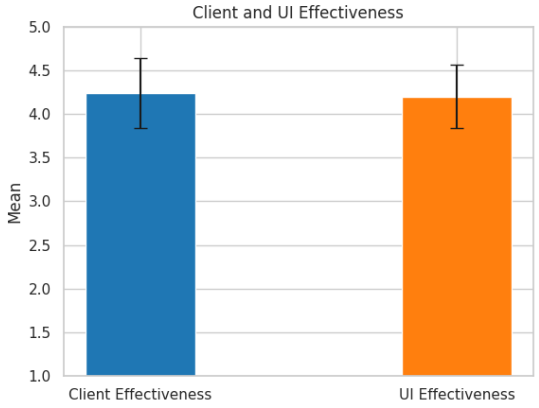

Figure 4: Client and UI effectiveness (scale: not effective to very effective).

facilitated by the Codenames resource can lead to effective learning outcomes for students.

Students generally recognized the spymaster role as inherently creative due to its solution space that, while constrained by the game rules, is still very large. However, some argued that creativity emerges from the approach rather than the task itself, noting that deterministic solutions like lookup tables, while potentially effective, would not demonstrate creative behavior.

Survey responses indicated substantial agent iteration between tournaments, with students reporting a mean modification magnitude of 3.44 on a scale from 1 (minor tweaks) to 5 (complete redesign) (see Fig. 2). Students found the first tournament experience particularly valuable for informing improvements, rating its effectiveness at 3.92 out of 5.

When evaluating their agents along three dimensions of computational creativity (Ventura 2017), students reported moderate to high scores for quality (3.56) and intentionality (3.80), but notably lower scores for novelty (2.68) (see Fig. 3). While students generally considered their agents creative, their justifications focused primarily on their own creative process in agent development rather than demonstrating understanding of formal CC frameworks and definitions.

Both the client framework and web interface proved effective development tools, receiving mean ratings of 4.24 and 4.20 respectively on a 5-point scale (see Fig. 4). Overall satisfaction with the learning experience was high (4.44 out of 5).

The second tournament demonstrated increased competitive balance, with closer games and more diverse winners compared to the first round, suggesting successful agent refinement across the cohort.

In their feedback about the Codenames framework itself, students expressed that they enjoyed using the tools and the learning experience the project provided. Their feedback also highlighted areas where the code base could be improved, such as simplifying the design of the example, displaying a history of game actions in the web client UI, and the addition of a developer mode to the web client that would allow one client to play all roles simultaneously in a test game. We will incorporate their feedback into continuing development and improvement of the public code repository.

Discussion & Future Work

Our objective in presenting this resource is to improve the accessibility of creative gameplaying agents in the CC research community, because accessibility impacts adoption and advancement. We emphasize that this extends to CC pedagogy and presents a powerful tool for introducing stu-

dents to our research community and supporting them as they develop their own creative systems.

One of the major use cases for this resource is in supporting and expanding CC research. The success of the resource at the workshop demonstrates its value in this regard. The framework provides researchers with a structured environment for evaluating word association methods and developing optimal game strategies through objective performance metrics. Creative gameplay represents a fertile avenue for future CC research, and we intend that this resource will be an entry point for the same. It enables automated gameplay between agents to gauge their relative capabilities, even if those agents are engineered differently. Future workshops or demonstrations could include competitions between various researchers' agents or against benchmark systems. We hope that usage of this resource will spur investigation into other creative games to further explore these types of creative domains.

The second major use case for this resource is for students in a CC class or for anyone looking for a low-barrier-to-entry into CC. Games provide inherent evaluation mechanisms through their rule structures, eliminating the need for subjective quality metrics that often complicate traditional CC tasks. Games are also generally well-defined tasks when compared to traditional CC tasks. This added structure is ideal for beginners and is well-suited to creating tools to aid the development process. Our results show that this Codenames resource was successful in its goal to help students more effectively develop their first creative agent. However, more work needs to be done to improve the connection between fundamental CC concepts and the experience of the students developing and evaluating their agents. Overall, both the development and competitive aspects of the learning experience were well received.

CC pedagogy is the primary example we have given for how to apply our Codenames framework, specifically in guiding students in creating their first CC systems. As such, it is appropriate to directly discuss how our Codenames agents include the elements of a CC system outlined in (Ventura 2017), namely *domain*, *knowledge base*, *aesthetic*, *representation*, *generation*, *evaluation*, *conceptualization*, and *translation*.

A subset of these elements are built into the task of playing Codenames. The domain is fixed, which notably includes a clear evaluation criteria—or aesthetic, to use Ventura's term—which many other creative domains lack. The phenotypic representation is also fixed by the game rules to a clue word and number. This representation lends itself to direct translation and trivial phenotypic evaluation; the example agent does not reason about those explicitly. It is notable that half of the considerations for building a CC system are obviated by the task of playing a creative game with well-defined rules!

The remaining elements comprise the design of the example spymaster agent. That agent is powered by word2vec embeddings, which were trained on a knowledge base of text corpora. The embeddings themselves represent the conceptualization of that knowledge. The genotypic representations used by the agent are the way it represents (e.g., vector embeddings) and uses them to manipulate subsets of word cards and candidate clues. The agent considers many such genotypes and internally evaluates them to select the best clue as its generated output. Thus, the included agent serves as a complete example of a CC system for this domain and will serve as a valuable learning tool for students.

Playing Codenames effectively requires understanding semantics and the relationships between words. Those skills are also relevant to a wide range of language-based creative tasks. Therefore, developing and refining an agent's semantic understanding in the context of Codenames could translate into improved performance at other creative tasks or subtasks. Future work could also expand this framework to incorporate other language games with similarly structured evaluation mechanisms, such as Decrypto, Wavelength, and the *New York Times*' Connections. Developing accessible tools and frameworks for a larger set of creative gameplay tasks provides a natural vector for accelerating progress in computational creativity research.

Conclusion

We have presented a public framework for building Codenames agents and playing games against both computational and human players. These tools are simple to use and modify, providing a solid jumping-off point for future computational creativity research. Additionally, these tools are well-suited for use in CC pedagogy. Students can interact with a pre-built CC agent, play games with and against it directly, and modify or replace its functionality without having to write client-server or game simulation code.

We have demonstrated the efficacy of this framework via a 25-student study in which graduate students in a CC class wrote their own agents to play in a Codenames tournament. Quantitative and qualitative data gathered during this study indicate that it was a fun and engaging learning activity. This helped introduce students to CC design principles in a focused and entertaining manner, paving the way for them to design their own CC agents in other domains.

The Codenames framework presented herein will also benefit the CC research community, as demonstrated by the results of a code jam workshop at *ICCC24*. It allows for rapid prototyping of new CC agents, simplifies the normally complex task of creative evaluation, and could serve to compare CC agent performance via direct competition. We foresee many benefits to expanded research in creative gameplay, which we hope our framework will help facilitate.

References

[Ackerman et al. 2017] Ackerman, M.; Goel, A.; Johnson, C. G.; Jordanous, A.; León, C.; Pérez y Pérez, R.; Toivonen, H.; and Ventura, D. 2017. Teaching computational creativity. In *Proceedings of the 8th International Conference on Computational Creativity*, 9–16. Association for Computational Creativity.

[Bown 2014] Bown, O. 2014. Empirically grounding the evaluation of creative systems: Incorporating interaction design. In *Proceedings of the 5th International Conference on Computational Creativity*, 112–119.

[Carnovalini et al. 2021] Carnovalini, F.; Harley, N.; Homer, S. T.; Roda, A.; and Wiggins, G. A. 2021. Meta-evaluating quantitative internal evaluation: A practical approach for developers. In *Proceedings of the 12th International Conference on Computational Creativity*, 213–217. Association for Computational Creativity.

[Chvátil 2015] Chvátil, V. 2015. Codenames.

[Colton, Charnley, and Pease 2011] Colton, S.; Charnley, J.; and Pease, A. 2011. Computational creativity theory: The face and idea descriptive models. In *Proceedings of the 2nd International Conference on Computational Creativity*, 90–95.

[Czech Games Edition 2015] Czech Games Edition. 2015. Codenames rules. https://czechgames.com/files/rules/codenames-rules-en.pdf. Official rulebook for the Codenames board game.

[Fragapane and Standl 2021] Fragapane, V., and Standl, B. 2021. Work in progress: Creative coding and computer science education – from approach to concept. In *Proceedings of the IEEE Global Engineering Education Conference*, 1233–1236.

[Jordanous 2012] Jordanous, A. 2012. A standardised procedure for evaluating creative systems: Computational creativity evaluation based on what it is to be creative. *Cognitive Computation* 4:246–279.

[Kakavas 2019] Kakavas, P. 2019. Computational thinking and creativity in K-6 education. *Formamente* 14(2):93–108.

[Lamb, Brown, and Clarke 2015] Lamb, C.; Brown, D. G.; and Clarke, C. 2015. Human competence in creativity evaluation. In *Proceedings of the 6th International Conference on Computational Creativity*, 102–109. Brigham Young University.

[Mikolov et al. 2013] Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, volume 26, 3111–3119.

[Orkphol and Yang 2019] Orkphol, K., and Yang, W. 2019. Word sense disambiguation using cosine similarity collaborates with Word2vec and WordNet. *Future Internet* 11(5):114.

[Pease and Colton 2011a] Pease, A., and Colton, S. 2011a. Computational creativity theory: Inspirations behind the FACE and the IDEA models. In *Proceedings of the 2nd International Conference on Computational Creativity*, 72–77.

[Pease and Colton 2011b] Pease, A., and Colton, S. 2011b. On impact and evaluation in computational creativity: A discussion of the Turing test and an alternative proposal. In *Proceedings of the Artificial Intelligence and the Simulation of Behaviour Symposium on Computing and Philosophy*, 15–22.

[Peeperkorn, Brown, and Jordanous 2023] Peeperkorn, M.; Brown, D.; and Jordanous, A. 2023. On characterizations of large language models and creativity evaluation. In *Proceedings of the 14th International Conference on Computational Creativity*, 143–147. Association for Computational Creativity.

[Ritchie 2007] Ritchie, G. 2007. Some empirical criteria for attributing creativity to a computer program. *Minds and Machines* 17:76–99.

[Spendlove and Ventura 2022] Spendlove, B., and Ventura, D. 2022. Competitive language games as creative tasks with well-defined goals. In *Proceedings of the International Conference on Computational Creativity*, 291–299. Association for Computational Creativity.

[Spendlove and Ventura 2023] Spendlove, B., and Ventura, D. 2023. A constraint-centric accounting of some aspects of creativity. In *Proceedings of the 14th International Conference on Computational Creativity*, 332–336. Association for Computational Creativity.

[Ventura 2016] Ventura, D. 2016. Mere generation: Essential barometer or dated concept. In *Proceedings of the 7th International Conference on Computational Creativity*, 17–24. Sony CSL.

[Ventura 2017] Ventura, D. 2017. How to build a CC system. In *Proceedings of the 8th International Conference on Computational Creativity*, 253–260. Association for Computational Creativity.

[Wiggins 2006] Wiggins, G. 2006. A preliminary framework for description, analysis and comparison of creative systems. *Knowlege-Based Systems* 19:449–458.

[Yee-King, Fiorucci, and d'Inverno 2024] Yee-King, M.; Fiorucci, A.; and d'Inverno, M. 2024. Designing an AI-creativity music course. In *Proceedings of The International Conference on AI and Musical Creativity*.