Probabilistic Inference in the Bust the Ghost Game: A Bayesian Approach

Laaziri Omar / Choukri Laila

April 2, 2024

Contents

1 Introduction

1.1 Overview of the Game

"Bust the Ghost" is an interactive game designed to challenge players' deductive reasoning and probabilistic inference skills. Set within a 9×12 grid, the game revolves around locating a ghost hidden in one of the cells. Players interact with the game by selecting cells to gather sensor readings, which indicate the proximity of the ghost through a color-coded system: red for direct contact, orange for near proximity, yellow for moderate distance, and green for distant or no proximity. The game incorporates a scoring system where each action depletes the player's points, adding a strategic layer to the gameplay. The ultimate goal is to "bust" the ghost by accurately determining its location within a limited number of attempts, using the sensor readings to inform decisions.

1.2 Objective

The primary objective of this report is to delve into the probabilistic models and Bayesian inference techniques underpinning the "Bust the Ghost" game. By dissecting the game's mechanics, this report aims to elucidate how probabilistic reasoning facilitates the estimation of the ghost's location, transforming raw sensor data into actionable intelligence. Through this analysis, we seek to demonstrate the practical application of probabilistic inference in game design, providing insights into its role in enhancing player engagement and decision-making processes.

1.3 Probabilistic Inference and its Importance

Probabilistic inference is a cornerstone of uncertainty modeling in artificial intelligence, enabling systems to make predictions, decisions, and updates based on incomplete or noisy information. In the context of "Bust the Ghost," probabilistic inference allows for the dynamic updating of the ghost's location probability distribution based on sensor readings. This approach not only simulates a realistic sensing environment but also introduces a nuanced challenge for players, requiring them to navigate and interpret probabilistic cues. The game thus serves as an exemplary case study for examining the application and impact of Bayesian inference in interactive systems, showcasing how abstract mathematical concepts can be embodied in engaging and educational gameplay.

2 Theoretical Background

This section provides the foundational concepts of probability theory and Bayesian inference necessary for understanding the probabilistic inferencing mechanisms employed in the "Bust the Ghost" game. Additionally, we discuss the Chebyshev distance metric used for spatial analysis within the game grid.

2.1 Bayesian Inference

Bayesian inference is a method of statistical inference in which Bayes' theorem is used to update the probability for a hypothesis as more evidence or information becomes available.

2.1.1 Bayes' Theorem

Bayes' theorem describes the probability of an event, based on prior knowledge of conditions that might be related to the event. Mathematically, it is expressed as:

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} \tag{1}$$

where A and B are events and $P(B) \neq 0$.

- $P(A|B)$ is a conditional probability: the likelihood of event A occurring given that B is true.
- $P(B|A)$ is also a conditional probability: the likelihood of event B occurring given that A is true.
- $P(A)$ and $P(B)$ are the probabilities of observing A and B independently of each other.

2.1.2 Application in Probabilistic Models

In the context of the game, Bayesian inference is utilized to update the probability distribution of the ghost's location based on sensor readings. If G represents the ghost's location and S the sensor reading, then Bayes' theorem helps us update our beliefs about G after observing S:

$$P(G|S) = \frac{P(S|G) \cdot P(G)}{P(S)} \tag{2}$$

$P(G|S)$ is the posterior probability of the ghost being at location G given the sensor reading S. $P(S|G)$ is the likelihood of observing S given the ghost's location G. $P(G)$ is the prior probability of the ghost being at location G, and $P(S)$ is the probability of observing S under all possible scenarios.

2.2 Probability Distributions

In Bayesian analysis, probability distributions are essential for defining and updating our beliefs about the world.

2.2.1 Uniform Distribution

Initially, the location of the ghost is unknown and is equally likely to be in any cell of the grid. This scenario is modeled using a uniform distribution:

$$P(G = g) = \frac{1}{N} \tag{3}$$

where N is the total number of cells in the grid.

2.2.2 Conditional Probability

Conditional probability distributions, such as $P(S|G)$, are central to updating our beliefs based on new evidence. These distributions depend on the distance between the sensor reading location and the ghost's actual location, influencing the sensor's color output.

2.3 Distance Metrics

The game employs a distance metric to determine the sensor's output color based on the ghost's proximity.

2.3.1 Chebyshev Distance

The Chebyshev distance between two points p and q, with coordinates (p_x, p_y) and (q_x, q_y) respectively, is defined as:

$$d_{\text{Chebyshev}}(p, q) = \max(|p_x - q_x|, |p_y - q_y|) \tag{4}$$

This metric is particularly relevant for grid-based games like "Bust the Ghost", as it measures the minimum number of moves required to go from one cell to another when moving horizontally, vertically, or diagonally.

2.3.2 Relevance to the Game

In "Bust the Ghost", the Chebyshev distance informs the sensor model about how far a clicked cell is from the ghost's actual location, thereby determining the color of the sensor reading based on predefined conditional probability distributions.

3 Probabilistic Inferencing and Implementation

This section combines the methodology and implementation of probabilistic inferencing in the "Bust the Ghost" game, focusing on Bayesian inferencing, the distribution used, and updates. We'll explore the initial setup, the role of distance metrics, conditional probability distributions, and how Bayesian inference updates the belief state based on sensor readings.

3.1 Game Setup and Initial Probabilities

The game grid is initialized with dimensions 9×12, representing the possible locations (x, y) where the ghost could be placed. Initially, the probability distribution over the grid is uniform, indicating equal likelihood of the ghost being in any cell.

The Ghost is initially placed at a random location with respect to the grid height and width:

```python
GRID_HEIGHT = 9
GRID_WIDTH = 12

def PlaceGhost():
    return np.random.choice(GRID_HEIGHT), np.random.choice(GRID_WIDTH)
```

3.1.1 Initial Probability Distribution

Each cell in the grid is assigned an initial probability of $\frac{1}{108}$, reflecting a uniform distribution. This is represented in the code as follows:

```python
def ComputeInitialPriorProbabilities():
    global probabilities
    probabilities = np.full((GRID_HEIGHT, GRID_WIDTH), 1.0 / (
   GRID_WIDTH * GRID_HEIGHT))
```

3.2 Distance Metric and Sensor Model

3.2.1 Chebyshev Distance

The distance between any two points (x_1, y_1) and (x_2, y_2) on the grid is calculated using the Chebyshev distance, defined as $\max(|x_2 - x_1|, |y_2 - y_1|)$. This metric is chosen because it effectively captures the gameplay's requirement that the distance considers the greatest single axis difference between points, which aligns with the movement possibilities of the ghost and sensor readings.

```python
def distance(loc1, loc2):
    return max(abs(loc1[0] - loc2[0]), abs(loc1[1] - loc2[1]))
```

3.2.2 Sensor Model and Conditional Probabilities

The sensor model defines the probability of observing a specific color given the distance from the ghost. These conditional probabilities are crucial for Bayesian updating. The following table outlines the conditional probability distribution used:

Distance	Red	Orange	Yellow	Green
0	0.9	0.1	0.0	0.0
1-2	0.1	0.8	0.1	0.0
3-4	0.0	0.1	0.8	0.1
5+	0.0	0.0	0.1	0.9

Table 1: Conditional probabilities of sensor colors given distance

The ‘DistanceSense‘ function utilizes these probabilities to sample a sensor reading based on the actual distance from the ghost:

```python
def conditional_prob_color_given_distance(dist):
    # Conditional probabilities mapping
    if dist == 0: return {'red': 0.9, 'orange': 0.1, 'yellow': 0, '
   green': 0}
    elif dist in [1, 2]: return {'red': 0.1, 'orange': 0.8, 'yellow':
   0.1, 'green': 0}
    elif dist in [3, 4]: return {'red': 0, 'orange': 0.1, 'yellow':
   0.8, 'green': 0.1}
    else: return {'red': 0, 'orange': 0, 'yellow': 0.1, 'green': 0.9}
```

3.3 Bayesian Updating of Probabilities

Upon receiving a sensor reading, the game updates the probabilities across the grid using Bayesian inference. This process is essential for refining the estimate of the ghost's location based on new information obtained through gameplay. The posterior probability $P(G = (x, y)|\text{Color})$ is recalculated for each grid cell, incorporating the likelihood of observing the reported sensor color if the ghost were in that cell.

The Bayesian update formula is applied as follows:

$$P(G = (x, y)|\text{Color}) = \frac{P(\text{Color}|G = (x, y)) \cdot P(G = (x, y))}{P(\text{Color})}$$

Where:

- $P(\text{Color}|G = (x, y))$ is the likelihood of observing the specified color, given the ghost's location. This likelihood is derived from the sensor model, which considers the distance between the sensor's location and each grid cell.
- $P(G = (x, y))$ represents the prior probability of the ghost occupying cell (x, y), reflecting our belief about the ghost's location before obtaining the sensor reading.
- $P(\text{Color})$ acts as a normalizing constant, ensuring that the updated probabilities across all cells sum to 1. Although not explicitly calculated, normalization is achieved by dividing the numerator by the sum of all numerators across the grid.

The code implements this Bayesian updating process within the `update_probabilities` function. For each cell in the grid, it calculates the product of the likelihood (based on the sensed color and the cell's distance from the sensor location) and the cell's prior probability. These products are then normalized to ensure they sum to 1, yielding the updated posterior probabilities:

```python
def update_probabilities(sensor_color, xclk, yclk):
    global probabilities
    new_probabilities = np.zeros_like(probabilities)

    for i in range(GRID_HEIGHT):
        for j in range(GRID_WIDTH):
            dist = distance((i, j), (xclk, yclk))
            prob_color_given_dist =
   conditional_prob_color_given_distance(dist)[sensor_color]
            new_probabilities[i][j] = prob_color_given_dist *
   probabilities[i][j]

    probabilities = new_probabilities / np.sum(new_probabilities)
```

The sensed color, crucial for determining $P(\text{Color}|G = (x, y))$, is obtained using the `DistanceSense` function. This function calculates the distance between a clicked cell and the ghost's location, then uses this distance to sample a color based on predefined conditional probabilities:

```python
def DistanceSense(xclk, yclk, gx, gy):
    dist = distance((xclk, yclk), (gx, gy))
    color_probabilities = conditional_prob_color_given_distance(dist)

    colors, probs = zip(*color_probabilities.items())
    color = np.random.choice(colors, p=probs)

    return color
```

This methodology not only updates the probability distribution across the grid in response to sensor inputs but also introduces variability in sensor readings, reflecting real-world uncertainties and enhancing the game's complexity and engagement.

4 Gameplay and User Interface

The user interface (UI) of the "Bust the Ghost" game, implemented with python library PyQt5 consists of several key components designed to provide players with essential information and controls for gameplay. These components are integral to the overall experience and facilitate interaction with the game mechanics.

4.1 Grid Display

The game presents players with a 9x12 grid, representing the playing field where the ghost may be hiding. Each cell in the grid corresponds to a potential location for the ghost.

4.2 Bust and Peep Buttons

Two buttons are prominently featured in the UI: the "Bust" button and the "Peep" button. The "Bust" button allows players to make a guess and attempt to bust the ghost if they believe it is hiding in a specific cell. The "Peep" button enables players to peek at the probability distribution across the grid, revealing the likelihood of the ghost's presence in each cell.

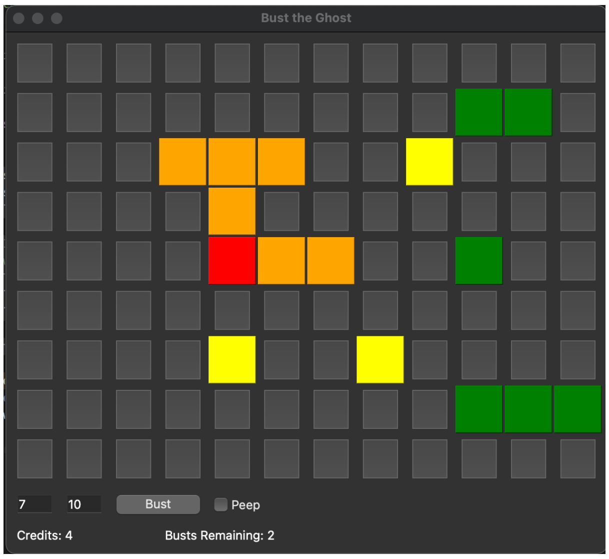
4.3 Peep Toggle Button

Adjacent to the "Peep" button, a toggle switch labeled "Peep" allows players to activate or deactivate the probability display feature. When toggled on, the grid displays the probability values on each cell, providing insight into the likelihood of the ghost's whereabouts. This feature enhances strategic decision-making and adds depth to the gameplay experience.

4.4 Remaining Attempts and Score

Throughout the game, players are informed of two critical metrics: the remaining "bust" attempts and their current score. The UI prominently displays the number of remaining attempts, indicating how many more times players can use the "Bust" button to make guesses. Additionally, the current score is continuously updated and displayed, reflecting the player's progress and performance in the game.

4.5 UI Screenshot

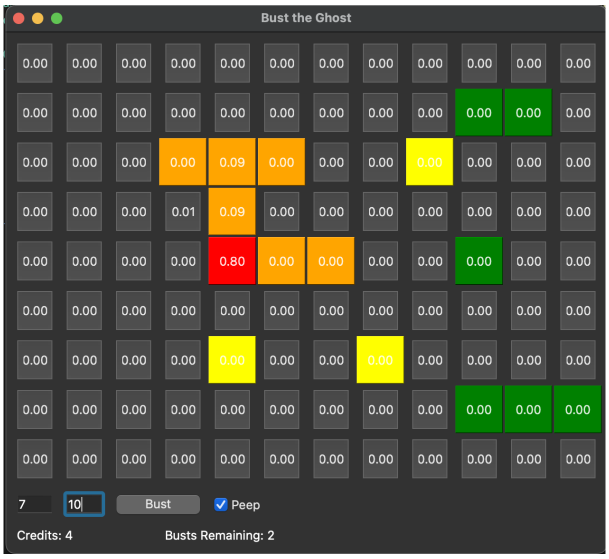


Bust the Ghost

7 10 Bust Peep

Credits: 4 Busts Remaining: 2

Figure 1: Gameplay UI



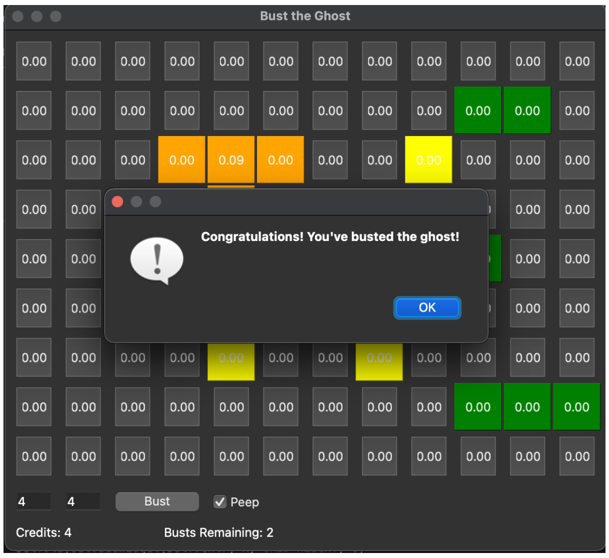
Bust the Ghost

0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00	0.09	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.01	0.09	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00	0.80	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

7 10 Bust ✓ Peep

Credits: 4 Busts Remaining: 2

Figure 2: Gameplay UI with peep button activated (displaying probabilities)



Congratulations! You've busted the ghost!

Credits: 4

Busts Remaining: 2

Figure 3: Busting the ghost

5 Conclusion

The "Bust the Ghost" game leverages probabilistic inferencing through Bayesian updates to dynamically adjust the belief state based on sensor readings. This approach not only adds depth to the gameplay but also introduces players to the principles of probabilistic reasoning and decision-making under uncertainty.

Laaziri Omar	Choukri Laila
Implementation + Report Writing	Implementation + Report Writing

Table 2: Team Contribution