LET'S PLAY!

W O R D L E

A DAILY WORD GAME

A Reinforcement learning Approach

Poornima Devi Krishnasamy Karthikeyan

HOW TO PLAY

Guess the Wordle in 6 tries.

1 Each guess must be a valid 5-letter word.

2 Color of Titles will change to show how close your guess was to the word

Examples

W O R D Y

W is in the word and in the correct spot.

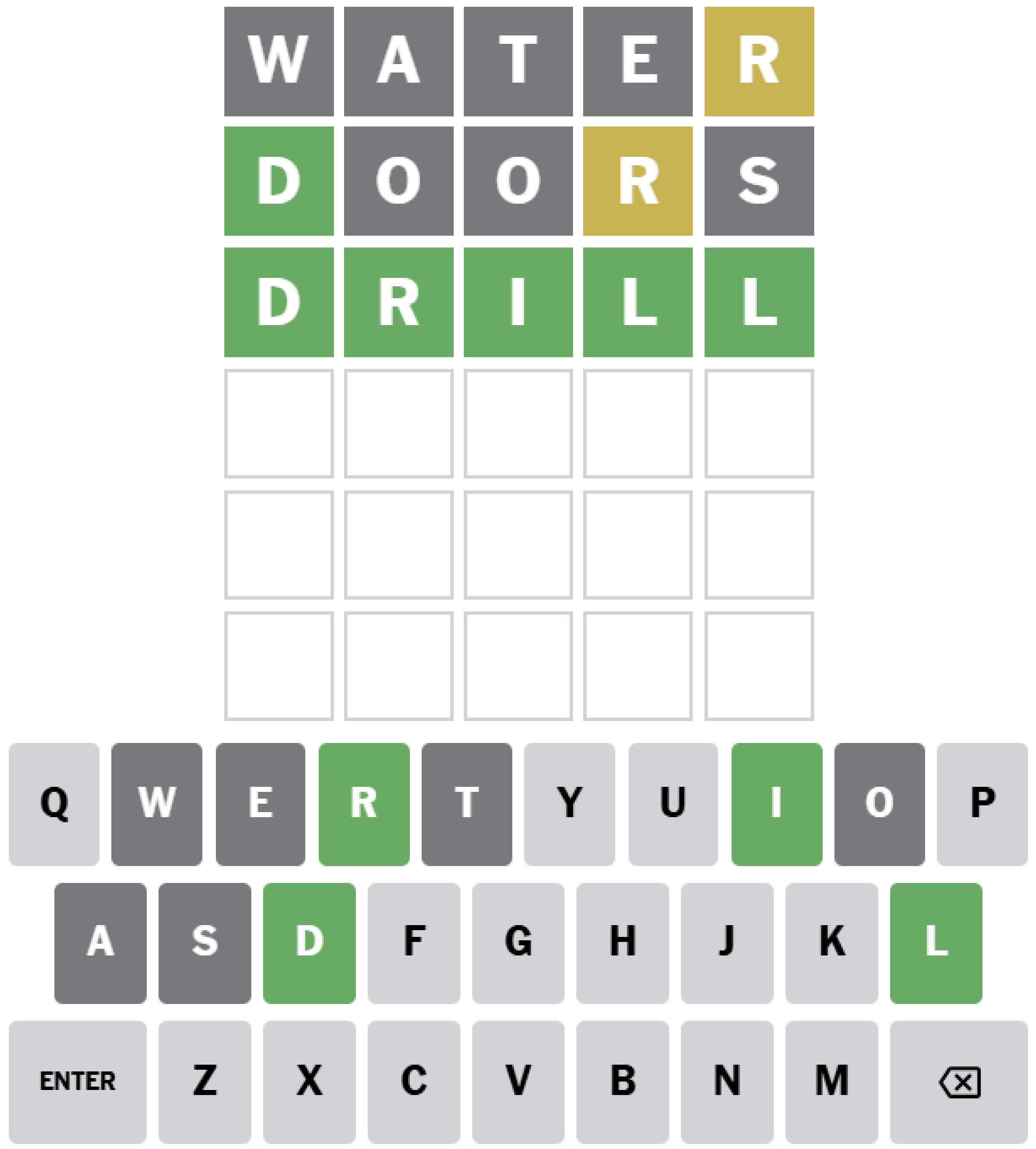
L I G H T

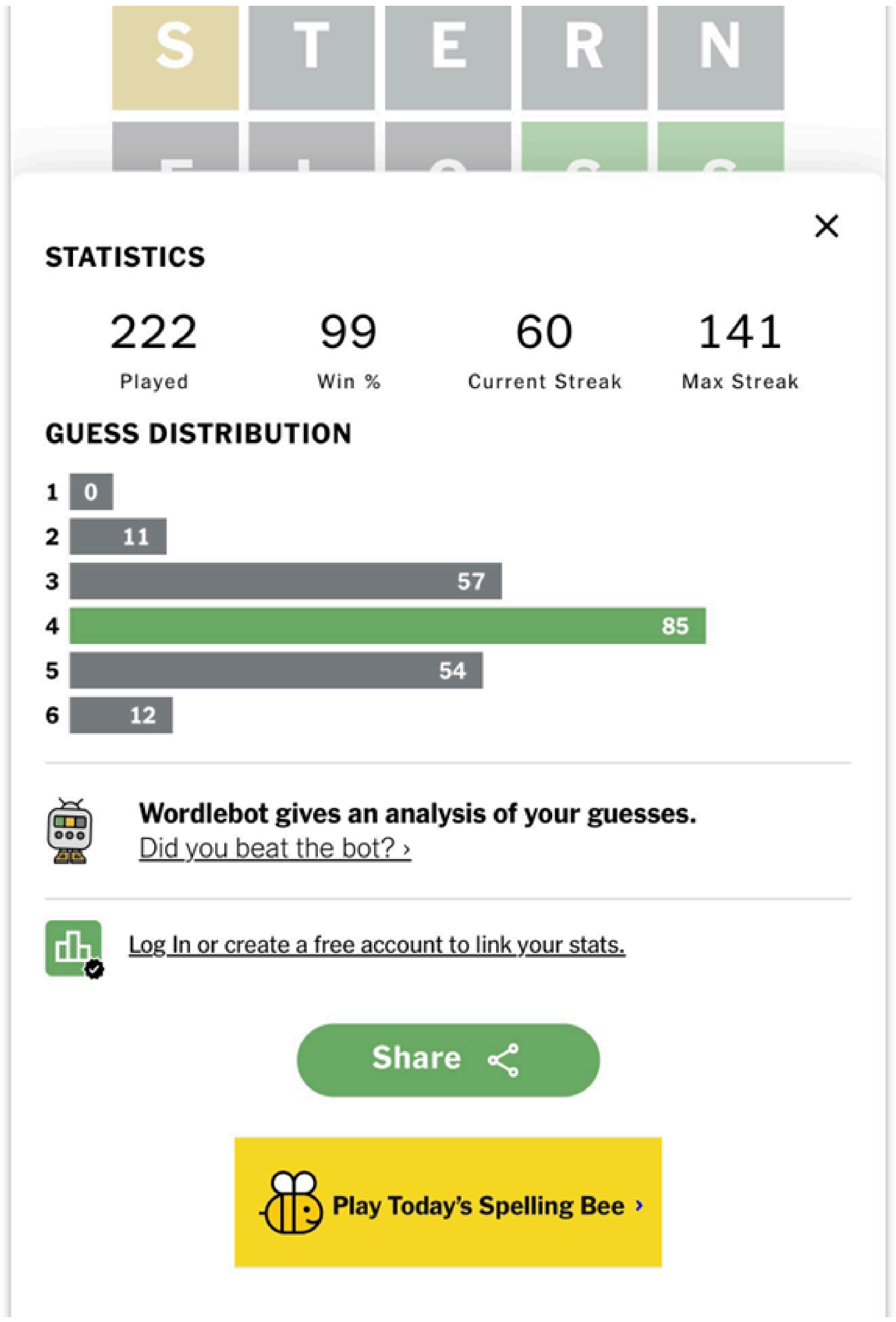
I is in the word but in the wrong spot.

R O G U E

U is not in the word in any spot.

GAME PLAY





STATISTICS

222	99	60	141
Played	Win %	Current Streak	Max Streak

GUESS DISTRIBUTION

1	0
2	11
3	57
4	85
5	54
6	12

Wordlebot gives an analysis of your guesses.
Did you beat the bot? ›

Log In or create a free account to link your stats.

Share

Play Today's Spelling Bee ›

CHALLENGE

GOAL → Guess 5-letter word in 6 attempts

Receive color-coded feedback:

- 🟩 Green: Correct letter, correct position
- 🟨 Yellow: Correct letter, wrong position
- ⬜ Gray: Letter not in word

OBJECTIVE → Maximize win rate • Minimize average guesses

VOCABULARY LIST

ACTUAL TEXT LIST

12972 WORDS

```
1 cigar
2 rebut
3 sissy
```

....

```
12969 zygal
12970 zygon
12971 zymes
12972 zymic
```

SOLUTION TEXT LIST

2315 WORDS

```
1 aback
2 abase
3 abate
```

....

```
2312 youth
2313 zebra
2314 zesty
2315 zonal
```

TARGET

MOTIVATION TO USE RL

- Sequential problem: each guess affects future states.
- Built-in reward signal (colour feedback).
- Adapts easily to rule changes (word length, attempts).
- Avoids brute-force search , fixed rule based system or trying to define what the best guess is through entropy, information gain, likelihood, etc.

EVVIRONMENT

- Game state representation
- Feedback mechanism
- Reward function
- Implementation

GAME STATE REPRESENTATION

State representation: 725-dimensional vector capturing the full game state, including:

- Encoded feedback from previous guesses
- Letter constraints (present/absent/position)
- Remaining valid word structure

Action representation: Selecting a word from the dictionary, with masking to prevent repeated guesses.

FEEDBACK MECHANISM

After each guess, the environment returns a 5-element feedback vector:

- 2 → Green (correct letter, correct position)
- 1 → Yellow (correct letter, wrong position)
- 0 → Gray (letter not in the target word)

This is the state information. Agent uses this to update the state representation.

REWARD FUNCTION

1.Solve the word (Correct guess):

- Reward = 10 points
- Bonus = 2 points for each remaining turn

Encourages the agent to solve the word as quickly as possible.

2. Informative letters:

- Green letter (correct position) = +0.8
- Yellow letter (correct letter, wrong position)= +0.3

Rewards guesses that give useful information.

3.Unhelpful guesses:

- All gray letters (no info) = -0.5

Discourages wasting guesses on uninformative words.

4. Fail to solve (turns exhausted):

- Reward = -5

Penalizes failure to solve the word.

EXPLORATION

Adaptive Temperature:

- Starts high (~1.5) → explore many words, discover strategies
- Gradually decreases to low (~0.3) → exploit best known words

```python
def masked_softmax(logits, mask, episode, total_episodes):
    adaptive_temp = max(0.3, 1.5 - (episode / total_episodes) * 1.2)
    logits = logits / max(adaptive_temp, 1e-6)
    logits = logits.masked_fill(mask == 0, -1e9)
    return F.softmax(logits, dim=-1), adaptive_temp
```

Behavior:

- High Temp: Tries diverse words, avoids local optima
- Low Temp: Focuses on high-probability words, maximizes success

ALGORITHM CHOSEN

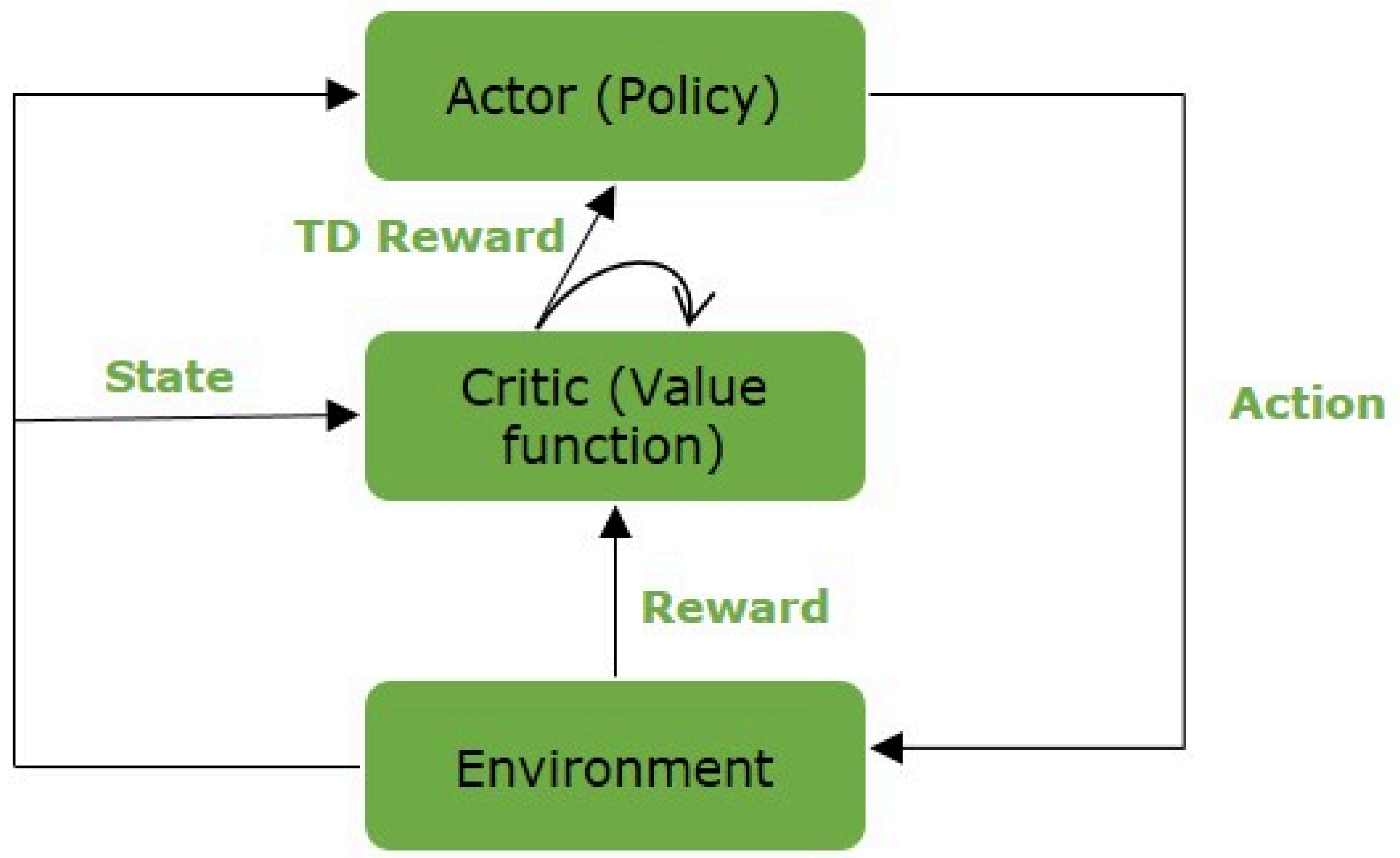
Environment formulation reveals that state space and action space are large. Not feasible to apply tabular methods. Hence, the following algorithm was implemented:

- **Advantage Actor Critic** →A2C handles large state and action spaces efficiently by learning both the policy and value, enabling smart and stable Wordle strategies.

ADVANTAGE ACTOR CRITIC

Actor → Selects actions based on the policy to **maximize rewards**, continually refining it to adapt to the environment.
Critic → Evaluates actor's actions, providing feedback to guide towards higher returns and improve learning.
The **advantage function**, A(s,a), measures the advantage of taking action a in state s over the expected value of the state under the current policy.

$$A(s,a) = Q(s,a) - V(s)$$

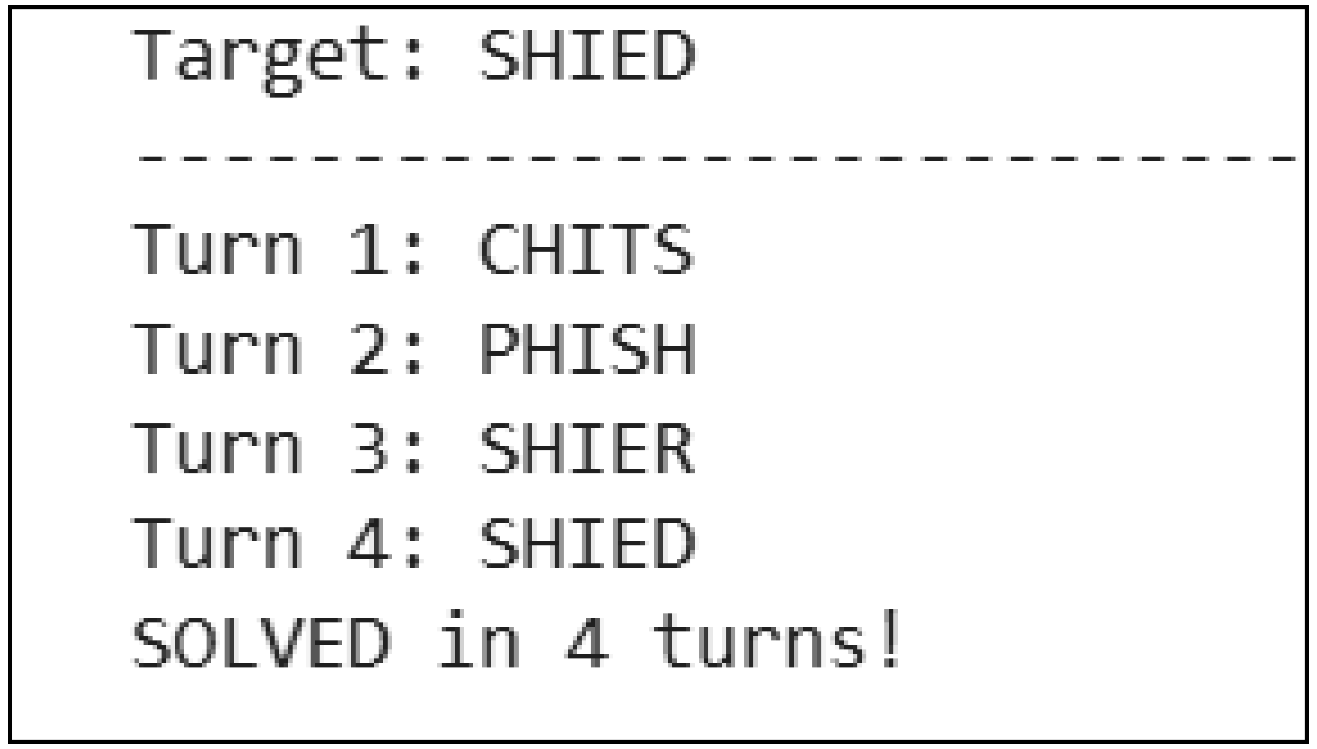


RESULTS

When training the model for 5000 epochs, it took approximately 1 hour.

Trained Agent

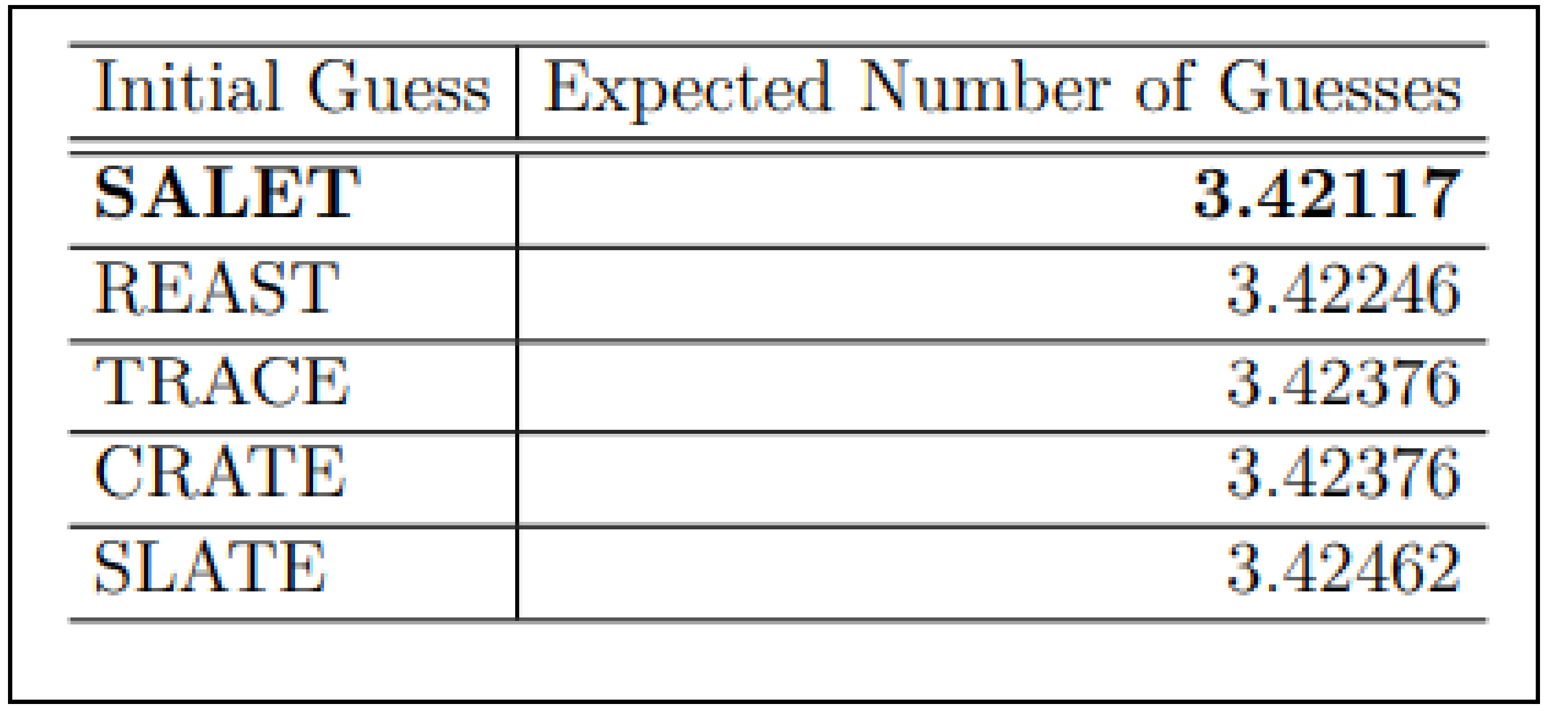
- 93% success rate
- 4.30 average guesses per win



```
Target: SHIED
------------------------------
Turn 1: CHITS
Turn 2: PHISH
Turn 3: SHIER
Turn 4: SHIED
SOLVED in 4 turns!
```

RESULTS : BEST STARTING GUESS

HOW DO YOU PICK A WORD THAT WILL GET TO THE SOLUTION WITH IN 6 GUESSES?

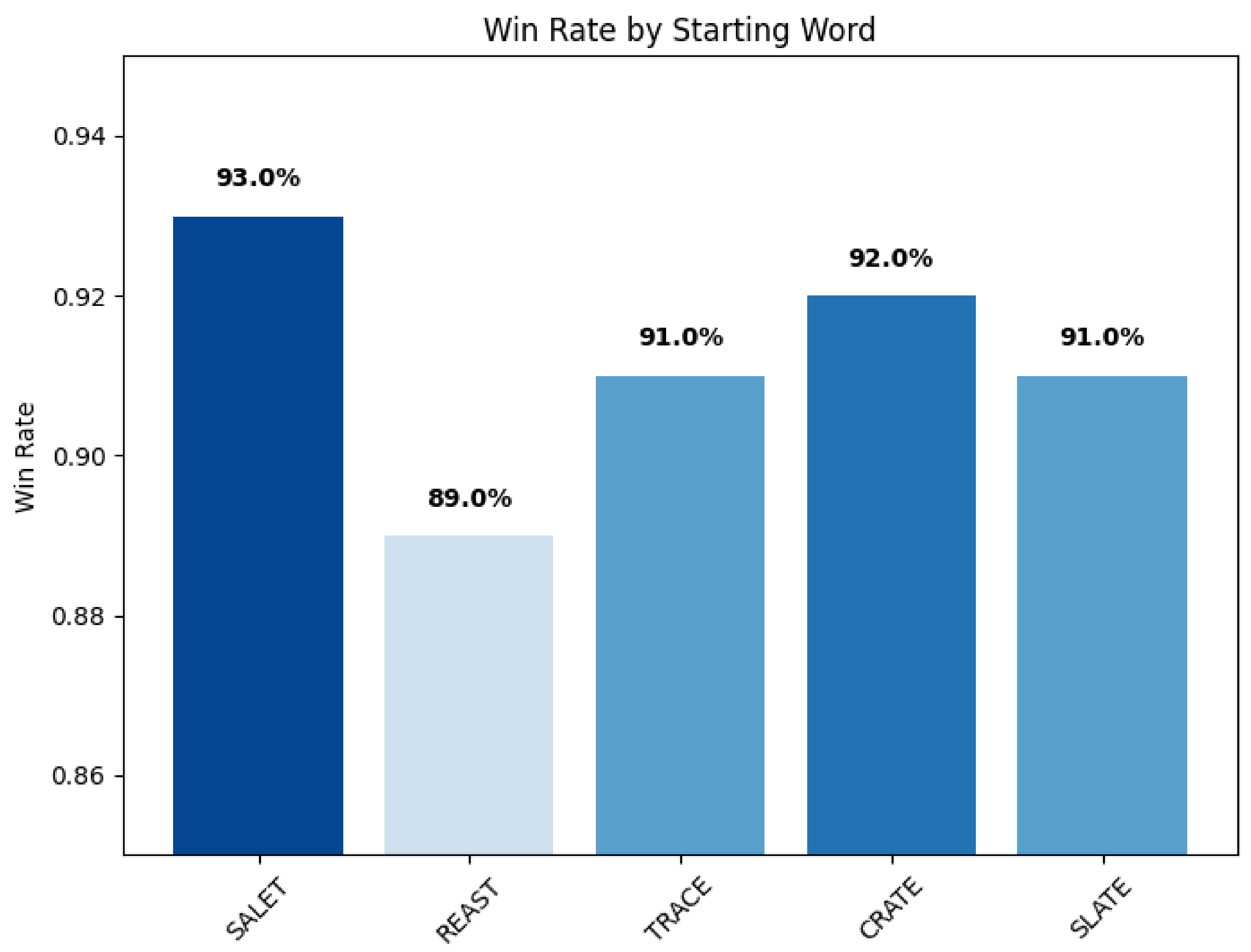


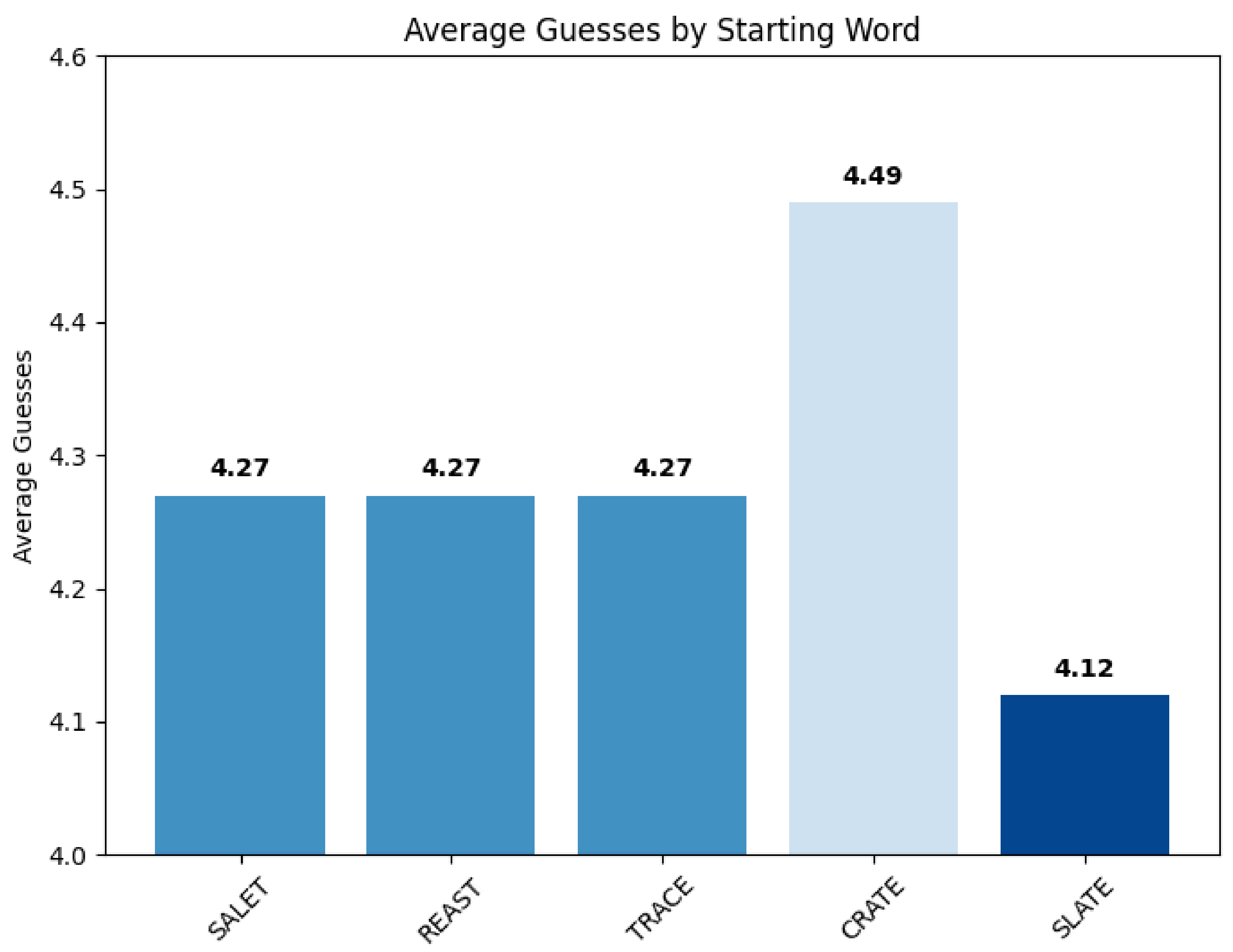
Initial Guess	Expected Number of Guesses
SALET	**3.42117**
REAST	3.42246
TRACE	3.42376
CRATE	3.42376
SLATE	3.42462

Table 1 : listing the five initial guesses in wordle yielding the lowest expected number of guesses to win. If an optimal policy is followed

Reference : https://auction-upload-files.s3.amazonaws.com/Wordle_Paper_Final.pdf

RESULTS : AGENT WITH BEST STARTING GUESS





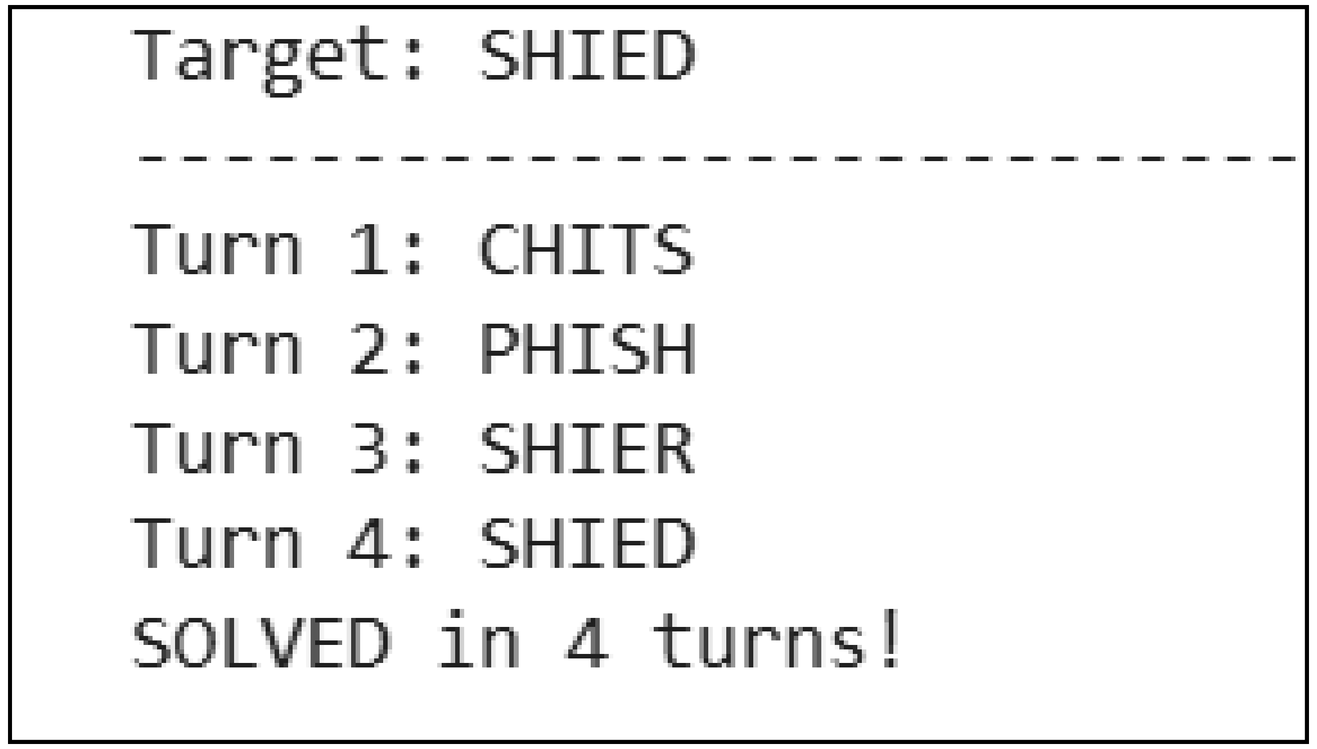
SALET is the best starting word with 93 % win rate and 4.12 average guess

RESULTS

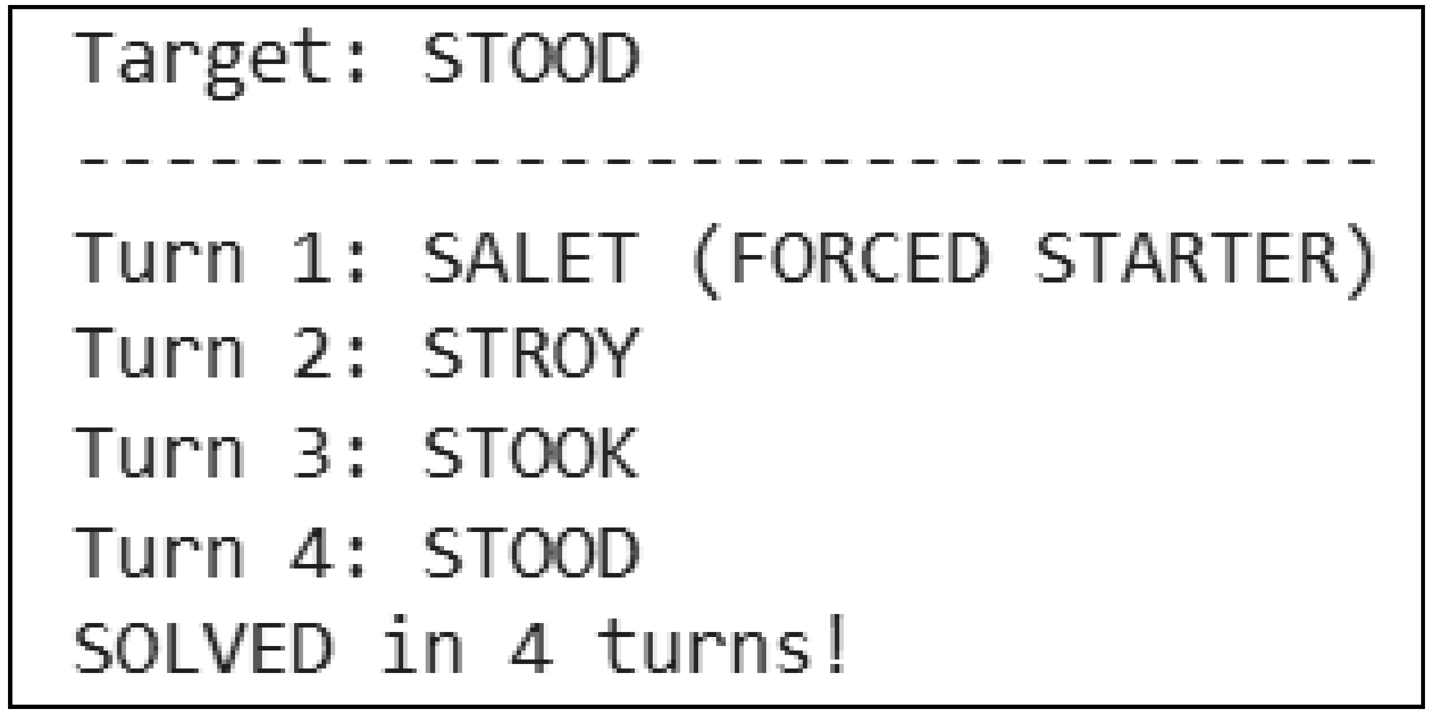
When training the model for 5000 epochs, it took approximately 1 hour.

Trained Agent → CHITS
- 93% success rate
- 4.30 average guesses per win

Agent with best starter → SALET
- 93% success rate (100 games)
- 4.12 average guesses per win

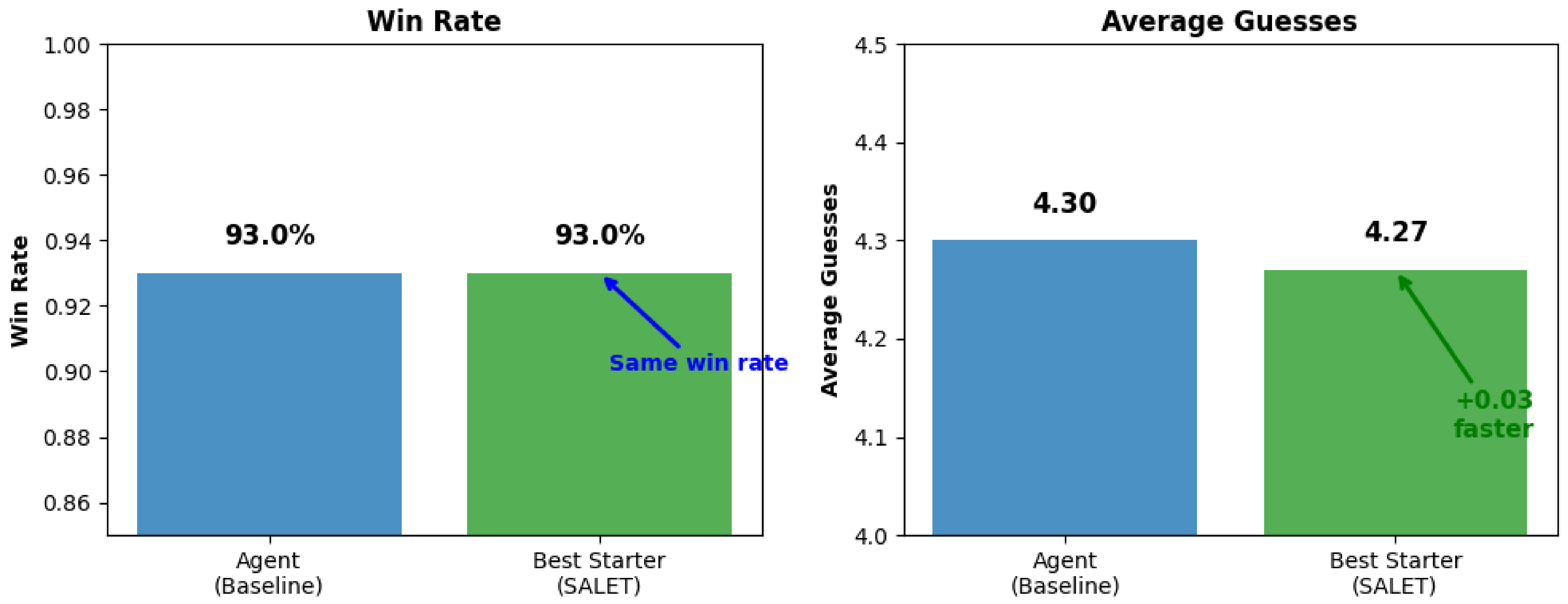


```
Target: SHIED
------------------------------
Turn 1: CHITS
Turn 2: PHISH
Turn 3: SHIER
Turn 4: SHIED
SOLVED in 4 turns!
```



```
Target: STOOD
------------------------------
Turn 1: SALET (FORCED STARTER)
Turn 2: STROY
Turn 3: STOOK
Turn 4: STOOD
SOLVED in 4 turns!
```

RESULTS



comparison of Baseline agent and agent with best starting word

THANK YOU