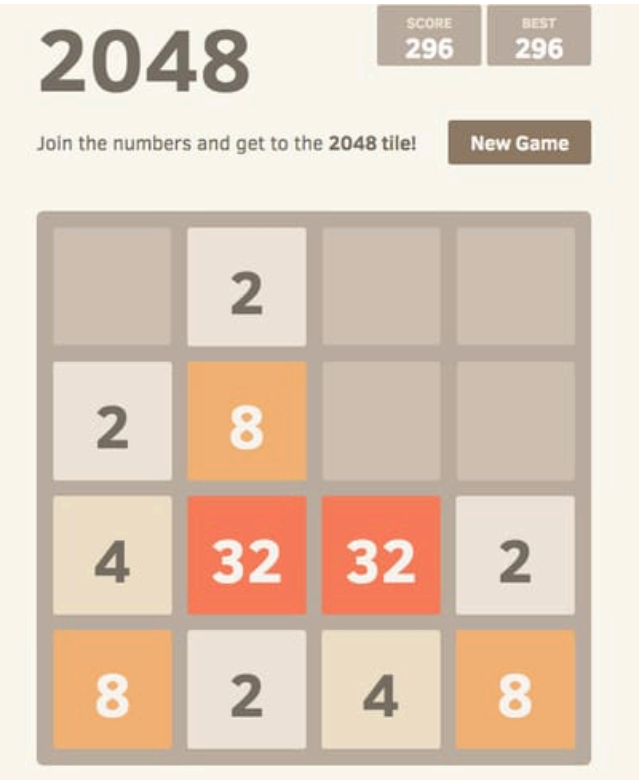
אוניברסיטת
אריאל
בשומרון

# דו"ח למידת חיזוקים- 2048

**שם המרצה:**

ד"ר טדי לזבניק

| שם הסטודנט | תעודת זהות |
|---|---|
| איתמר מלניק | 207312307 |
| תומר סבג | 318814977 |
| איתי ראובן חבשה | 207379447 |
| רותם בראל | 318223872 |



אוניברסיטת
אריאל
בשומרון

## תוכן עניינים:

## 1.מבוא לפרויקט

### 1.1 רקע ומוטיבציה

משחק 2048 מהווה מקרה בוחן עבור מחקר בלמידת חיזוקים, בשל שילוב ייחודי בין מרחב מצבים עצום, לוח בגודל 4×4 עם ערכי אריחים בקידוד לוגריתמי לבין דינמיקה מורכבת הכוללת גם רכיבים דטרמיניסטיים (הזזה ומיזוג של אריחים) וגם רכיב אקראי (הופעת אריח חדש מסוג 2 או 4 במיקום אקראי).
במסגרת פרויקט זה, התמקדנו ביישום גישת **After-State Value Learning**, המבוססת על הערכת מצב הלוח לאחר ביצוע הפעולה אך טרם הופעת האלמנט הסטוכסטי, ובכך מאפשרת הפחתה משמעותית של אי-הוודאות בעדכון ערכי ה-TD.

המטרה המרכזית הייתה להמחיש כי ניתן להגיע לביצועים מרשימים מבחינת ניקוד ושיעור הגעה לאריח 2048 ומעלה באמצעות ייצוג חכם ויעיל של מרחב המצבים, תוך שמירה על פשטות חישובית ויכולת פרשנות גבוהה, וזאת מבלי להסתמך על רשתות עצביות עמוקות.

### 1.2 מטרות ויעדים

**ביצועים התנהגותיים:** ניתוח ממוצע הניקוד, הסתברות הגעה לאריח 2048 ומעלה, תחת תצורות שונות של קצב למידה (lr). מטרת הניתוח היא לבחון את השפעת ההיפר-פרמטרים על יציבות הלמידה והתכנסותה.

**השוואה שיטתית:** כימות תרומתם של דפוסי N-tuple אשר נלמדים בכל סימטריית סיבוב וההשתקפות של הלוח (שמונה אפשרויות עבור כל דפוס), במטרה לזהות אילו מבין התבניות תורמות באופן המשמעותי ביותר להערכת מצבי המשחק ולהצלחת הסוכן בקבלת החלטות.

**פרשנות ונגישות**: הדגמת יתרונות הגישה מבחינת שקיפות ויכולת הבנה, באמצעות ייצוג מפורש של ערכי הלמידה בטבלאות LUTS, בניגוד לגישות המבוססות על רשתות נוירונים (כגון DQN) הנתפסות כ"קופסה שחורה". בנוסף, מוצגת פשטות יחסית בכיול היפר-פרמטרים ובפירוש תוצאות הלמידה.

### 1.3. חדשנות ותרומה

הפרויקט משלב בין שתי שיטות למידה מוכחות המשלימות זו את זו:

1. After-State Learning- למידה על מצבי-ביניים (לאחר פעולת הסוכן ולפני הופעת האלמנט האקראי) מפחיתה את השונות בעדכוני הלמידה ומובילה להתכנסות מהירה ויציבה יותר.

2. N-tuple Symmetric Representation- זיהוי תבניות מקומיות בלוח תוך שימוש בכל הסיבובים וההשתקפויות של הדפוסים מאפשר העברת מידע בין מצבים שקולים, מפחית משמעותית את מספר הפרמטרים הנדרשים (ביחס של פי 8), ומשפר את יכולת ההכללה של הסוכן.

3.ייצוג לוגריתמי של ערכי האריחים- קידוד הערכים באמצעות $\log_2$ מצמצם את טווח המספרים ומפחית את מורכבות מרחב המצבים, מאיץ את חישובי הלמידה ומקל על הסוכן להכליל גם אריחים גבוהים שלא הופיעו בשלבי האימון.

השילוב בין הגישות, יחד עם מימוש יעיל בשפת Python, מאפשר בניית סוכן מהיר, יציב וקל לפרשנות, הנשען על קוד פשוט ונגיש.

## 1.4. היקף העבודה ומשאבים

הפרויקט כלל שלושה רכיבי עבודה עיקריים:

1. **פיתוח תשתית המשחק והסוכן:** יצרנו את מחלקת Game 2048 כממשק המשחק שעליו אימנו ובדקנו את הסוכן, לצד זה פיתחנו סוכן למדת חיזוקים מסוג After-State TD עם מדיניות ε-greedy. הסוכן מבוסס על ייצוג N-tuple תוך שימוש בכל הסימטריות של הדפוסים להרחבת הלמידה.

2. **הרצת ניסויים וכיול פרמטרים:** כל סוכן עבר אימון של 50,000 אפיזודות, עם דעיכה הדרגתית של ε עד לערך מינימלי של 0.001 לאחר האימון, ביצענו ולידציה של ביצועי הסוכן אשר רצה על 1000 אפיזודות שבהן משתמשים במשקולות המעודכנות של הסוכן וללא חקירה, לצורך מדידת ביצועים בצורה עקבית ודטרמיניסטית.

3. **עיבוד תוצאות וייצוג חזותי:** בוצעה הפקת גרפים וטבלאות לצורך ניתוח שלבי האימון והוולידציה.
בשלב האימון הוצגו ממוצע הניקוד, שיעור ההגעה לאריח 2048, שגיאת TD לאורך זמן, והתפלגות האריח המרבי ב־1,000 אפיזודות אחרונות לכל אחד מהסוכנים.
בשלב הוולידציה הוצגה התפלגות האריח המרבי שהושג על ידי כל אחד מארבעת הסוכנים שנבחנו.
בנוסף, פותח ממשק חזותי אינטראקטיבי באמצעות Pygame לצורך הדגמת פעולת הסוכן המאומן בזמן אמת.
כחלק מהערכת הסוכן הנבחר, הופקו גם גרפים לניתוח המשקולות שנלמדו לאורך האימון.

**חלוקת תפקידים** בין חברי הצוות התבצעה באופן גלובלי כך שכל אחד לקח אחריות על אחד מהשלבים: תשתית הקוד, ייצוג התכונות והאלגוריתם, הרצת הניסויים ועיבוד התוצאות, והכתיבה והתיעוד בדו״ח.

**משאבים טכניים**: Python- סביבת עבודה, Numba JIT- להאיץ את הקוד, Pickle -לשמירה, Matplotlib -לשרטוט, Pygame -לדמו בזמן אמת.

## 1.5. מבנה הדו״ח

הדו"ח בנוי במבנה מדורג הכולל את כל שלבי הפרויקט. תחילה נציג את ניסוח הבעיה במונחי למידת חיזוקים (MDP), ולאחר מכן נפרט את המתודולוגיה שנבחרה – החל מעקרונות הלמידה ועד להיבטי המימוש הטכני. בהמשך נציג את התוצאות הכמותיות והשוואתיות שהתקבלו בניסויים, בליווי גרפים וניתוחים סטטיסטיים. לאחר מכן נבחן את הממצאים במסגרת דיון ביקורתי, שבו נעמוד על יתרונות הגישה, מגבלותיה, והמלצות להמשך פיתוח עתידי. לבסוף, נסכם את התרומות המרכזיות של הפרויקט ואת התובנות העיקריות שעלו ממנו.

## 2. ניסוח הבעיה במונחי למידת חיזוקים (MDP)

### 2.1 הגדרת מרחב המצבים (State Space)

משחק 2048 מתבצע על לוח בגודל 4×4, כאשר כל תא בלוח יכול להכיל אריח שערכו חזקה של 2 (כגון 2, 4, 8, ..., 2048, 4096, 8192) או להיות ריק. באופן תיאורטי, מרחב המצבים הגולמי כולל את כל הסידורים האפשריים של 16 תאים עם ערכים אלה – מרחב עצום בגודלו, ולכן נדרש ייצוג מקוצר או חכם יותר של המצבים האפשריים.

על מנת להתמודד עם מורכבות זו, נעשה שימוש **בקידוד לוגריתמי** של ערכי האריחים. בקידוד זה:

- ערך ריק → 0
- ערך 2 → log2(2) = 1
- ערך 4 → log2(4) = 2
- ערך 8 → log2(8) = 3
- וכן הלאה, עד לערך 8192 שמיוצג כ־13 (log2(8192))

באמצעות קידוד זה, כל תא בלוח מיוצג באמצעות מספר שלם בטווח 0–13. ייצוג זה מצמצם משמעותית את מורכבות מרחב המצבים, מאפשר שימוש בטבלאות חיפוש (LUTS), תורם ליעילות חישובית, ומאפשר לו לקבל החלטות טובות גם במצבים חדשים שלא הופיעו במהלך האימון.

### 2.2 הגדרת מרחב הפעולות (Action Space)

מרחב הפעולות במשחק 2048 כולל ארבע פעולות דיסקרטיות, המסומנות כך **A = {0,1,2,3}**, כאשר

- 0 – הזזה למעלה (Up)
- 1 – הזזה ימינה (Right)
- 2 – הזזה למטה (Down)
- 3 – הזזה שמאלה (Left)

יש להבחין כי לא כל פעולה תקפה בכל מצב. פעולה נחשבת תקפה רק אם היא משנה את מצב הלוח בפועל, כלומר אם היא גורמת להזזת אריחים או למיזוגם. במקרים שבהם הפעולה אינה משפיעה על מצב הלוח (למשל ניסיון להזיז לכיוון חסום), היא תיחשב כבלתי חוקית ותסונן במהלך הרצת הסוכן.

### 2.3 פונקציית התגמול (Reward Function)

פונקציית התגמול במשחק מוגדרת כ־ r(s, a) = Σ Merge Values, כלומר סכום ערכי האריחים שנוצרו כתוצאה ממיזוג במהלך ביצוע הפעולה. עבור כל פעולה שמביאה למיזוג, הסוכן מקבל תגמול השווה לסכום הערכים של האריחים החדשים שנוצרו. לדוגמה, מיזוג של שני אריחים בערך 16 יניב תגמול של 32. אם לא התרחש מיזוג, התגמול הוא 0, וכך גם כאשר המשחק מסתיים ואין פעולות חוקיות. מבנה תגמול זה נועד לעודד את הסוכן לבצע מיזוגים משמעותיים, להעדיף יצירת אריחים גדולים, ולמקסם את הניקוד המצטבר לאורך זמן.

## 2.4 דינמיקת הסביבה (Environment Dynamics)

הסביבה במשחק 2048 משלבת שני רכיבים עיקריים: דטרמיניסטי וסטוכסטי, הפועלים בזה אחר זה בכל מהלך.

**הרכיב הדטרמיניסטי:**
לאחר בחירת פעולה על ידי הסוכן (הזזה לאחד מארבעת הכיוונים), מתבצע שלב דטרמיניסטי שבו כל האריחים נעים לכיוון שנבחר, ומתמזגים לפי חוקי המשחק. כאשר שני אריחים זהים נצמדים זה לזה במהלך ההזזה, הם מתמזגים ליצירת אריח חדש שערכו כפול. כל אריח יכול להתמזג רק פעם אחת במהלך אותו תור.

**הרכיב הסטוכסטי:**
לאחר סיום ההזזה והמיזוגים, נוסף אריח חדש באחד מהמיקומים הריקים על הלוח, באופן אקראי. ב- 90% מהמקרים יתווסף אריח בערך 2, וב־10% מהמקרים יתווסף אריח בערך 4. שלב זה הוא מקרי, ומכניס שונות לתהליך הלמידה.

**מצב סיום:**
המשחק מסתיים כאשר הלוח מלא ואין אף פעולה חוקית שניתן לבצע כלומר, לא קיימת הזזה שמובילה לשינוי במצב הלוח.

## 2.5 After-State Representation

במסגרת עבודה זו נעשה שימוש בייצוג מבוסס **מצבי-ביניים (After-States)**. בניגוד ללמידה רגילה המתבצעת על המצב לפני ביצוע הפעולה, בלמידת After-State מתבצע עדכון על המצב שנוצר מיד לאחר ביצוע הפעולה, אך **לפני** הופעת האריח הרנדומלי.

ההבחנה המרכזית היא בין:

- **מצב רגיל (s):** מצב הלוח לפני בחירת פעולה
- **After-State:** מצב הלוח לאחר ביצוע הפעולה אך לפני הופעת האריח החדש

לגישה זו מספר יתרונות:
ראשית, מכיוון שה־After-State הוא תוצאה דטרמיניסטית של פעולה נתונה, הוא מפחית את רמת השונות (variance) בתהליך הלמידה. שנית, עדכוני ה-TD הופכים ליציבים ואמינים יותר, כיוון שהם אינם מושפעים מהגורם הסטוכסטי של הופעת האריח החדש. לבסוף, הייצוג הזה מקטין את מספר המצבים שדורשים עדכון, ולכן תורם גם ליעילות חישובית גבוהה יותר.

## 3. מתודולוגיה ומימוש

### 3.1 דפוסים בסיסיים (Base Patterns)

לצורך ייצוג תכונות הלוח, הוגדרו שמונה דפוסים בסיסיים המתפרסים על אזורים שונים בלוח המשחק. באמצעות סיבובים והשתקפויות, כל דפוס משמש כמסגרת ללמידה על מבנים מקומיים בלוח בצורה כללית, יעילה חישובית וחוסכת בזיכרון.
הדפוסים נבחרו בקפידה מתוך מטרה לאפשר כיסוי מרחבי מלא של הלוח, תוך שמירה על איזון בין פשטות יישומית לבין עומק למידה. הם כוללים מבנים אופקיים, מלבניים וריבועיים, אשר ביחד מצליחים ללכוד מגוון רחב של אינטראקציות בין אריחים – החל מרצפים ליניאריים ועד לאשכולות מורכבים.
צורת L אינו מכסה אזור ייחודי בלוח ואינו תורם ישירות להשלמת כיסוי מלא, אך הוא נשמר כחלק מהייצוג מתוך מטרה ללכוד אינטראקציות מרחביות לא סימטריות ולהעשיר את הגיוון במבני הלמידה.

דפוסי שורות (Horizontal Patterns):

- Row 1: התאים (0,0), (0,1), (0,2), (0,3) – מייצגים את השורה העליונה
- Row 2: התאים (1,0), (1,1), (1,2), (1,3) – מייצגים את השורה השנייה

דפוסי 2×3:

- Top 3×2: התאים (0,0), (0,1), (0,2), (1,0), (1,1), (1,2) – מייצגים מלבן עליון
- Bottom 3×2: התאים (1,1), (1,2), (1,3), (2,1), (2,2), (2,3) – מייצגים מלבן תחתון

דפוסי 2×2:

- TL 2×2: התאים (0,0), (0,1), (1,0), (1,1) – פינה שמאלית עליונה
- TM 2×2: התאים (0,1), (0,2), (1,1), (1,2) – מרכז עליון
- BM 2×2: התאים (1,1), (1,2), (2,1), (2,2) – מרכז תחתון

דפוס בצורת L:

- L-shape: התאים (0,0), (0,1), (0,2), (1,2), (2,2) – יוצרים תבנית בצורת האות L

### 3.2 ייצוג סימטרי

במטרה לשפר את יכולת ההכללה של הסוכן ולצמצם את מספר הפרמטרים הנדרשים לאחסון וללמידה, כל דפוס בסיסי הורחב לשמונה וריאציות סימטריות. ההרחבה כוללת ארבעה סיבובים של הדפוס המקורי בזוויות של 0, 90, 180 ו־270 מעלות, וכן ארבע השתקפויות אופקיות, שכל אחת מהן משולבת עם אותם סיבובים. כתוצאה מכך, כל תבנית נלמדת במקביל בכל אחת מהגרסאות הסימטריות שלה על גבי הלוח. שיטה זו מאפשרת לסוכן לשתף מידע בין מצבים שקולים מבחינה מבנית, תורמת לצמצום משמעותי של מספר הפרמטרים (ביחס של פי שמונה), ומאפשרת תהליך למידה מהיר ויעיל יותר, תוך הכללה רחבה יותר של דפוסים מוכרים במרחב המצבים.

## 3.3 טבלאות LUTS

כל אחד מהדפוסים הבסיסיים מיוצג באמצעות טבלת חיפוש (LUT), המכילה ערך נלמד עבור כל קומבינציה אפשרית של ערכי התאים בדפוס. גודל הטבלה נקבע לפי מספר התאים בדפוס בחזקת 14, שכן כל תא יכול לקבל אחת מ־14 רמות (0-13) בהתאם לקידוד הלוגריתמי של ערכי האריחים. כל הערכים בטבלאות מאותחלים לערך קבוע של 10, אתחול זה נבחר מאחר שהתחלה בערך אפס הייתה מביאה לכך ששגיאת ה-TD תישאר אפס מה שהיה מונע מהסוכן ללמוד. הערך 10 נועד להגדיר את השגיאה ההתחלתית בצורה מבוקרת שתעודד חקירה מוגברת בשלבי האימון הראשונים.

העדכונים בטבלאות מתבצעים לפי כלל עדכון TD סטנדרטי, בהתאם לשגיאת התחזית (TD error) ולפי קצב הלמידה שנבחר. עבור כל דפוס, נבנית טבלה נפרדת בהתאם לגודלו:

- דפוסי שורות (4 תאים × 2 שורות): $2 \times 14^4 = 38{,}416 \times 2 = 76{,}832$
- דפוסי 2×3 (6 תאים × 2): $2 \times 14^6 = 7{,}529{,}536 \times 2 = 15{,}059{,}072$
- דפוסי 2×2 (4 תאים × 3): $3 \times 14^4 = 38{,}416 \times 3 = 115{,}248$
- דפוס L
  5 תאים×1: $1 \times 14^5 = 537{,}824 \times 1 = 537{,}824$

בסך הכול נוצרים כ־15.79 מיליון פרמטרים נלמדים, הערוכים בטבלאות דיסקרטיות ומופרדות עבור כל דפוס.

## 3.4 אלגוריתם After-State TD Learning

להלן תיאור פסאודו-קוד של אלגוריתם הלמידה שבו השתמשנו. האלגוריתם מבוסס על עדכוני TD-Temporal Difference המבוצעים על מצבי-ביניים (after-states), תוך שימוש במדיניות ε-greedy ובערכת תכונות מסוג N-tuple עם עדכונים ליניאריים.

```
Algorithm: After-State TD Learning
Input: Learning rate α, discount γ, ε-greedy policy
Initialize: Feature network with random/neutral weights

For each episode:
    s ← reset_environment()

    While not terminal(s):
        valid_actions ← get_valid_moves(s)

        // ε-greedy action selection
        if random() < ε:
            a ← random_choice(valid_actions)
        else:
            a ← argmax_{a'} [R(s,a') + V(after_state(s,a'))]
```

```
        // Execute action and get after-state
        s_after, R ← execute_action(s, a)
        current_value ← V(s_after)

        // Environment adds random tile
        s' ← add_random_tile(s_after)

        // Compute TD target
        if terminal(s'):
            target ← 0
        else:
            next_actions ← get_valid_moves(s')
            target ← γ × max_{a'} V(after_state(s',a'))

        // TD update
        td_error ← target - current_value
        update_features(s_after, α × td_error)

        s ← s'

    decay_epsilon()
```

<u>הסבר לוגי של שלב עדכון הערכים באלגוריתם After-State TD</u>

לאחר שהסוכן בחר פעולה וביצע אותה (כלומר הזיז אריחים בלוח), הוא מקבל את מצב הלוח החדש שנוצר בעקבות הפעולה, מצב זה מכונה **after-state** ומסומן s_after. עבור מצב זה מחושב ערך משוער על ידי פונקציית הערך V (s_after) , אשר מייצגת את התחזית של הסוכן לגבי איכות מצב זה.

לאחר מכן, הסביבה מוסיפה אריח אקראי בלוח, ונוצר מצב חדש s', שהוא זה שממשיך את האפיזודה בפועל. כעת, הסוכן מחשב את TD Target, כלומר את ערך המטרה שאותו הוא היה מצפה לראות אם התחזית הייתה מדויקת.
אם s' הוא מצב סופי (כלומר אין תזוזות חוקיות) – ערך המטרה מוגדר כ־0.
אם לא, הסוכן מחשב את המקסימום של הערכים שהוא יכול לקבל מה־after-states של כל פעולה חוקית מ־s'. כלומר, הוא בונה תחזית מושכלת של מה שצפוי לקרות בהמשך, דרך:

```
target = γ × max_a' V(after_state(s', a'))
```

לאחר חישוב זה, הסוכן מבצע את עדכון הלמידה בפועל:
הוא מחשב את טעות התחזית (TD Error) כפער בין הערך שחושב ב־s_after לבין ערך המטרה:

```
TD_error = target - V(s_after)
```

ולבסוף, הוא מעדכן את פונקציית הערך כך שתתקרב למטרה המחושבת:

```
V(s_after) ← V(s_after) + α × TD_error
```

העדכון נעשה על המשקולות (features) המייצגות את המצב s_after, בעזרת קצב הלמידה α, אשר שולט על גודל הצעד שהסוכן לוקח לעבר ערך התחזית החדש.

במהלך האפיזודות, ערך ε קטן באופן הדרגתי, כך שהסוכן מבצע פחות חקירה אקראית (random moves) ויותר ניצול של המדיניות שלמד – דבר שמוביל להתכנסות יציבה ומהירה.

גישה זו מאפשרת עדכון מדויק יותר, מאחר והיא מפרידה בין פעולת הסוכן לבין הרעש האקראי שמתווסף מהסביבה, וכך תורמת לא רק ליעילות הלמידה אלא גם ליכולת להבין ולנתח את מה שהסוכן למד בפועל.

### 3.5 עדכון המשקולות

במהלך תהליך הלמידה, כל עדכון של ערך ה־TD משפיע על כל אחת מהטבלאות (LUT) הרלוונטיות לדפוסים שנמצאו במצב הביניים (after-state). מאחר שכל דפוס נלמד בכל אחת משמונה הסימטריות שלו, מתבצע עדכון ממוצע לכל סימטריה, כך שסכום השינויים יתחשב באופן שווה בכלל הווריאציות. להלן מימוש פונקציית העדכון:

```
def update_features(after_state, td_error, alpha):
    update_value = (alpha * td_error) / 8.0  # חלקים ב-8 סימטריות

    for pattern, lut in patterns:
        for symmetry in range(8):
            feature_values = extract_pattern(after_state, pattern, symmetry)
            index = pattern_to_index(feature_values)
            lut[index] += update_value
```

## 3.6 היפר-פרמטרים וכיולים

### 3.6.1 פרמטרי למידה

במהלך הניסוי נבחנו ארבעה ערכים שונים לקצב הלמידה α והם: 0.0, 0.01, 0.005, 0.0025, מקדם ההנחה (γ) הוגדר כ־1.0, כדי לתת משקל מלא לתגמולים עתידיים ולמקסם את הניקוד לאורך כל המשחק. הבחירה בפעולות נעשתה לפי מדיניות ε-greedy, כאשר ערך ההתחלה של ε היה 0.5, והוא דעך בהדרגה פי 0.995 לאחר כל אפיזודה עד שהגיע לערך מינימלי של 0.001. כך נשמר איזון בין חקירה לניצול, עם מעבר הדרגתי למהלכים מחושבים ככל שהאימון מתקדם.

### 3.6.2 הגדרת אימון

כל אחד מהסוכנים אומן במשך 50,000 אפיזודות, כאשר משך האימון נע בין חמש לשבע שעות, בהתאם לקצב הלמידה (α). במהלך האימון נמדדו ביצועים לאורך זמן לפי ארבעה ערכי α והם: 0.0025, 0.005, 0.01 ו־0.05.

המדדים שנבחנו:

- ניקוד ממוצע לאורך האימון
- שגיאת TD
- שיעור הצלחה בהגעה לאריח 2048
- התפלגות האריח המרבי (1000 אפיזודות אחרונות)

## 3.7 מימוש טכני

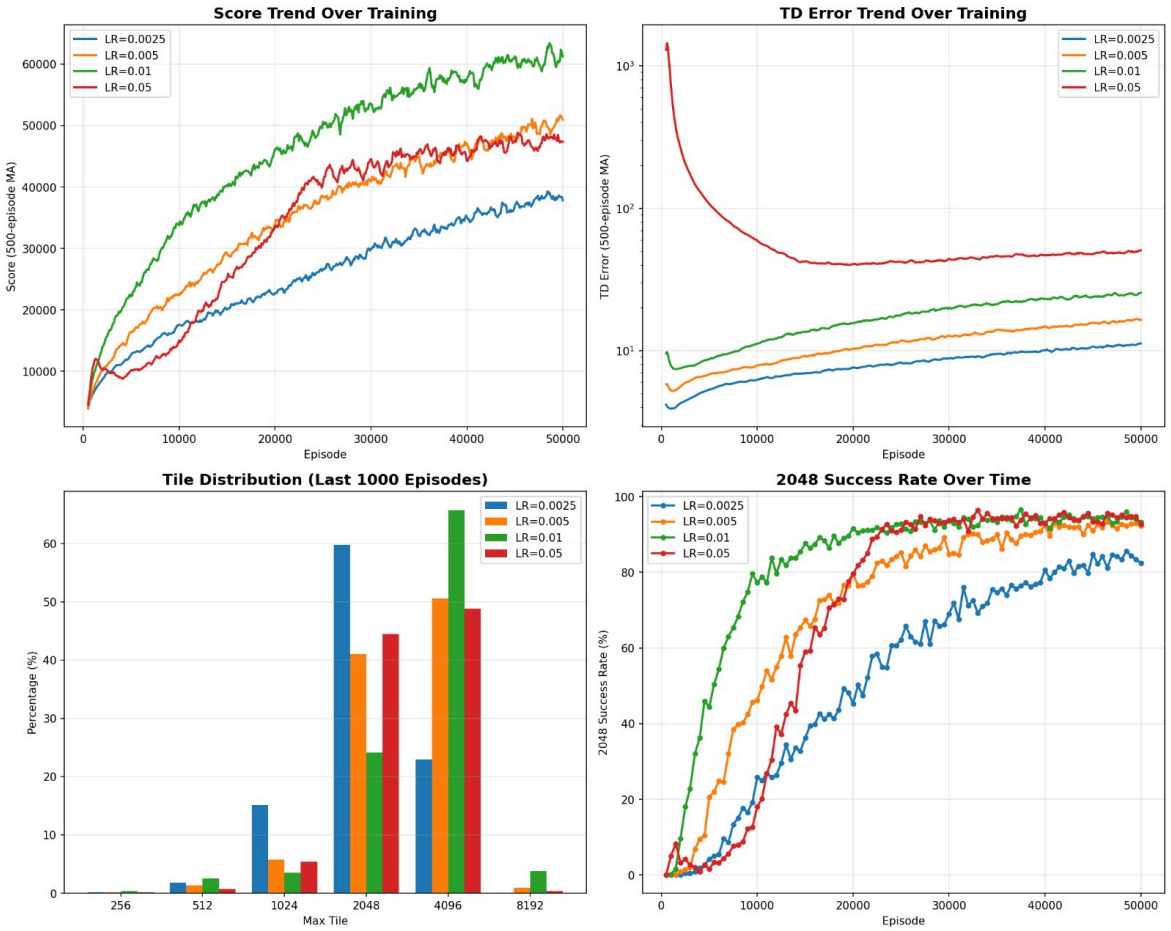
- Python: שפת פיתוח עיקרית
- NumPy: חישובים נומריים וייצוג לוחות
- Numba: קומפילציה JIT לקוד קריטי
- Matplotlib: ויזואליזציה ותרשימים
- Pandas: עיבוד נתונים וטבלאות
- Pygame: ממשק גרפי להדמיית הסוכן המאומן
- Pickle: שמירה וטעינה של מודלים מאומנים

## 4. תוצאות ניסיוניות

### 4.1 ביצועי אימון

להלן תוצאות ההשוואה- ממצאים עיקריים:

| Agent | Learning Rate | Episodes | Avg Score | Success Rate (2048+) | 256 | 512 | 1024 | 2048 | 4096 | 8192 |
|---|---|---|---|---|---|---|---|---|---|---|
| Agent 1 | 0.0025 | 50000 | 38086 | 82.9% | 0.2% | 1.8% | 15.1% | 59.8% | 23.0% | 0.1% |
| Agent 2 | 0.0050 | 50000 | 50849 | 92.6% | 0.2% | 1.4% | 5.8% | 41.1% | 50.6% | 0.9% |
| Agent 3 | 0.0100 | 50000 | 60888 | 93.6% | 0.4% | 2.5% | 3.5% | 24.1% | 65.7% | 3.8% |
| Agent 4 | 0.0500 | 50000 | 47895 | 93.7% | 0.2% | 0.7% | 5.4% | 44.5% | 48.8% | 0.4% |




ניתוח כלל המדדים הגרפיים מצביע על כך שקצב הלמידה α = 0.01 סיפק את התוצאות האופטימליות ביותר לאורך האימון, תוך איזון בין מהירות למידה, יציבות ויכולת הכללה.

- **ניקוד ממוצע לאורך האפיזודות- Score Trend**
  α = 0.01 הוביל לניקוד הממוצע הגבוה ביותר (מעל 60,000), עם מגמת עלייה יציבה לאורך כל שלבי האימון. α נמוך (0.0025) הפגין שיפור איטי, בעוד ש־α גבוה מדי (0.05) האיץ את העלייה הראשונית אך התייצב מוקדם יותר ובערכים נמוכים יותר.

- **שגיאת TD Error Trend:**
  בכל הסוכנים נצפתה ירידה חדה בשגיאת התחזית בשלבים הראשונים של האימון, בהמשך

האימון הסוכנים שמרו על מגמה יציבה של עליה, מה שכנראה מעיד שהם עדיין בלמידה ולא הגיעו להתכנסות.

- **שיעור הצלחה בהגעה לאריח Success Rate 2048:**
שיעור ההצלחה עלה באופן משמעותי לאחר כ־10,000–15,000 אפיזודות, והגיע ליותר מ־90% עבור α = 0.01 ו־0.05. עם זאת, רק α = 0.01 שמר על עקביות גבוהה לאורך זמן, בעוד ש־0.05 סבל מתנודתיות.

- **התפלגות האריח המרבי Tile Distribution:**
הטבלה מציגה את התפלגות האריח המרבי שהושג על ידי כל אחד מהסוכנים במהלך 1000 אפיזודות האחרונות של האימון.

| Agent | 256 | 512 | 2048 | 4096 | 8192 |
|---|---|---|---|---|---|
| Agent 1 | 0.2% | 1.8% | 15.1% | 23.0% | 0.1% |
| Agent 2 | 0.2% | 1.4% | 5.8% | 50.6% | 0.9% |
| Agent 3 | 0.4% | 2.5% | 3.5% | 65.7% | 3.8% |
| Agent 4 | 0.2% | 0.7% | 5.4% | 48.8% | 0.4% |

הטבלה מציגה את שיעור האפיזודות שבהן הושג כל ערך מרבי בלוח. מבין הסוכנים, Agent 3 הציג את הביצועים הגבוהים ביותר, עם שיעור של 65.7% הגעה לאריח 4096 ו־3.8% הגעה לאריח 8192 — נתונים שמעידים על יכולת למידה עמוקה והתכנסות יציבה.

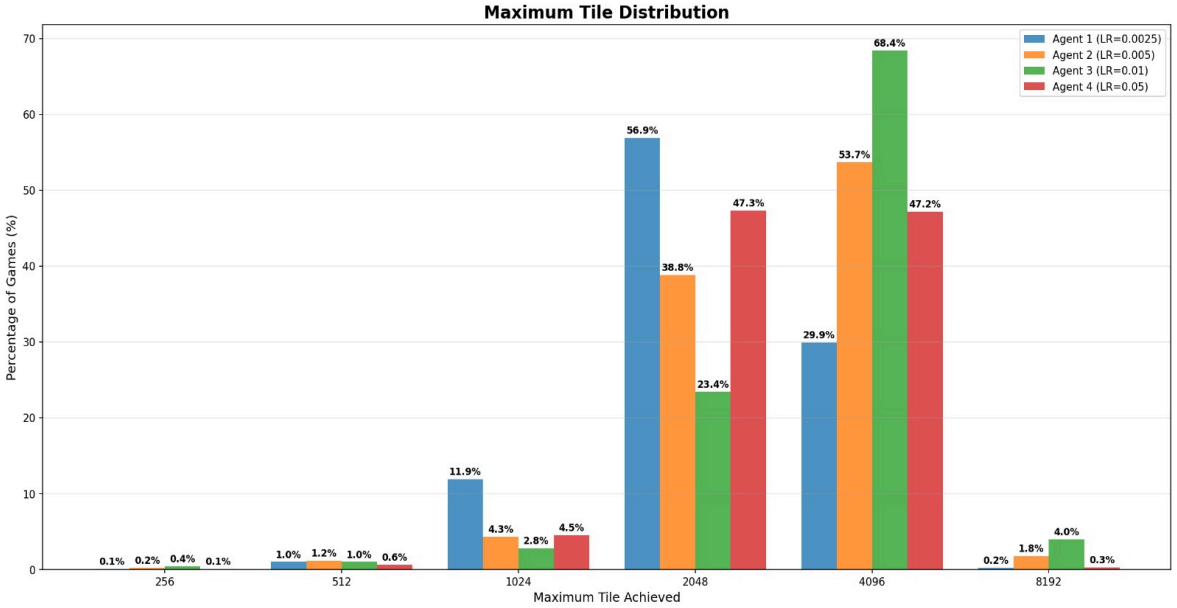
בסך הכול, הגרפים משקפים כי תהליך הלמידה טרם הגיע למיצוי מלא, שכן הן הניקוד והן שגיאת התחזית ממשיכים לעלות. נתון זה מעיד על כך שלסוכנים יש פוטנציאל להמשך התקדמות, אך גם מחזק את החשיבות שבהגדרת קצב למידה מדויק לצורך התכנסות מיטבית ויעילות גבוהה.

## 4.2 תוצאות ולידציה

### 4.2.1 ביצועים על סט הוולידציה (1000 אפיזודות)

לאחר תום שלב האימון בוצעה וולידציה של 1,000 אפיזודות, שבהן הסוכן פעל במדיניות חמדנית טהורה (pure greedy), ללא חקירה ועדכון המשקולות. מטרת שלב זה הייתה למדוד את ביצועי הסוכנים בתנאים יציבים, ולבחון עד כמה המודל הכללי מצליח לשחזר אסטרטגיות.

| Agent | Learning Rate | Avg Score | Games | 256 | 512 | 1024 | 2048 | 4096 | 8192 |
|---|---|---|---|---|---|---|---|---|---|
| Agent 1 | 0.0025 | 41081 | 1000 | 0.1% | 1.0% | 11.9% | 56.9% | 29.9% | 0.2% |
| Agent 2 | 0.0050 | 53691 | 1000 | 0.2% | 1.2% | 4.3% | 38.8% | 53.7% | 1.8% |
| Agent 3 | 0.0100 | 62704 | 1000 | 0.4% | 1.0% | 2.8% | 23.4% | 68.4% | 4.0% |
| Agent 4 | 0.0500 | 47569 | 1000 | 0.1% | 0.6% | 4.5% | 47.3% | 47.2% | 0.3% |

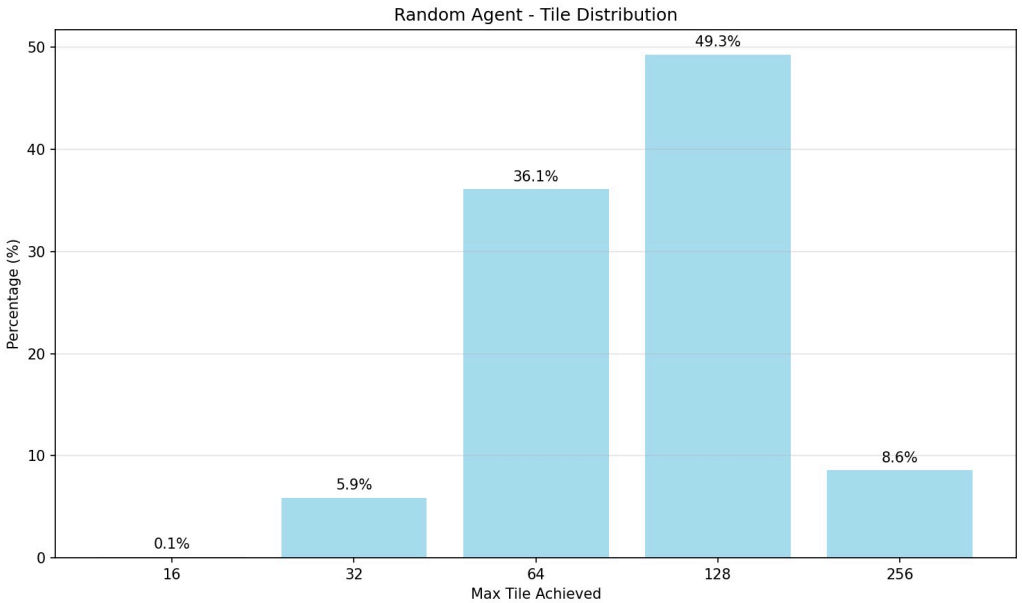



ניתן לראות על סט הוולידציה,כי Agent 3 ממשיך להציג את הביצועים המרשימים ביותר, עם ממוצע ניקוד של 62,704 ושיעור של 68.4% הגעה לאריח 4096. בנוסף, הוא הסוכן שהגיע ל־8192 באחוזים הגבוהים ביותר מה שמעיד על יציבות ויכולת הכללה גם לאחר שלב האימון.

### 4.2.2 השוואה לסוכן אקראי (Baseline)

לצורך קביעת קו בסיס, נבחנו ביצועיו של סוכן אקראי הפועל ללא כל רכיב למידה כלומר- בוחר באקראי פעולה חוקית בכל תור. השוואה זו מחדדת את התרומה של הלמידה המונחת: בעוד הסוכן האקראי מציג דפוסי משחק מוגבלים וסטגנציה בביצועים, הסוכן המאומן מפגין שיפור משמעותי עקבי, הממחיש את היתרון הגלום בגישת After-State בשילוב ייצוג N-tuple סימטרי.
השוואה זו מדגישה את הפער בין פעולה אקראית לבין תהליך למידה מכוון המבוסס על הבנת מבנה המשחק.

| Agent | Avg Score | Games | 16 | 32 | 64 | 128 | 256 |
|---|---|---|---|---|---|---|---|
| Random | 1127 | 1000 | 0.1% | 5.9% | 36.1% | 49.3% | 8.6% |



הגרפים מצביעים על כך שסוכן אקראי מציג התקדמות מוגבלת מאוד במשחק, כאשר כמעט כל האפיזודות מסתיימות באריחים נמוכים (64–128) וממוצע הניקוד נשאר נמוך (1,127).נתונים אלו מדגישים את היעדר היכולת של סוכן ללא למידה להבין את מבנה המשחק, ומהווים קו בסיס ברור ליתרון של סוכנים מבוססי למידה.

## 4.3 ניתוח משקולות ותכונות

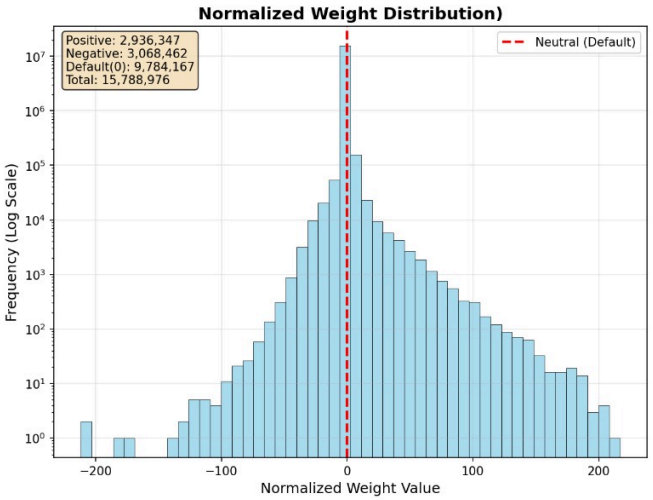
### 4.3.1 כיסוי למידה

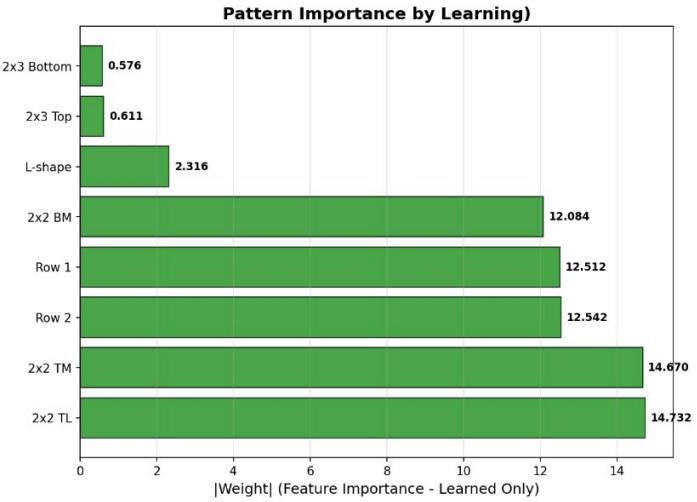
לשם הבנת אופן הלמידה של הסוכן, בוצע ניתוח סטטיסטי וגרפי של משקולות הרשת (LUTS) לאחר סיום האימון. כלל הערכים עברו נרמול על ידי חיסור ערך ברירת המחדל (10), כך שערכים שלא עברו עדכון הפכו לאפס. גישה זו אפשרה להפריד באופן ברור בין משקולות שנלמדו לבין משקולות ניטרליות, ולהבין את מבנה הלמידה בפועל.

מהנתונים עולה כי 38% מהמשקולות (6,004,809 מתוך 15,788,976) עודכנו במהלך האימון, בעוד ש־62% נותרו בערך ברירת המחדל.
התפלגות הערכים הנלמדים (גרף שמאלי) מציגה מבנה הדומה לפעמון, סימטרי סביב אפס, תוך שימוש בסקאלה לוגריתמית. התפלגות זו תואמת את הציפיות מהתנהגות עדכוני TD ואת הסימטריה המובנית בדפוסי ה־N-tuple.

בהיבט הפונקציונלי, בוצע ניתוח חשיבות של כל אחד משמונת הדפוסים בלמידה (גרף ימני). מדד החשיבות חושב על בסיס הממוצע המוחלט של ערכי המשקולות שנלמדו עבור כל דפוס (|weight|).
התוצאות מצביעות בבירור על כך שדפוסי 2x2 (בייחוד TL ו־TM) הם התורמים ביותר להערכת מצבים – עם ערכי חשיבות של 14.7 ו־14.6 בהתאמה. לעומתם, דפוסים מורכבים יותר כגון 2x3 Top/Bottom תרמו מעט מאוד ללמידה בפועל (פחות מ־0.7), למרות שהכילו מספר גדול של משקולות – מה שמרמז על חפיפה מידעית גבוהה או חוסר יעילות מבנית.

הפער בין מספר המשקולות הפעילות לחשיבות הממוצעת בפועל מחדד את התרומה של הדפוסים הקטנים בדגש על דפוסי 2x2 – כבסיס חזק ללמידת ייצוגים משמעותיים בלוח המשחק. הממצאים תומכים ביעילות הארכיטקטורה שנבחרה, ומחזקים את ההצדקה לשימוש ב־N-tuple סימטריים במשחקים כמו 2048.



### 4.3.2 חשיבות דפוסים

הטבלה הבאה מדרגת את שמונת הדפוסים שנבחנו, לפי חשיבותם היחסית:

| דפוס | דפוס | תיאור | דרגה |
|---|---|---|---|
| TL 2x2 | 14.372 | פינה שמאלית עליונה | 1 |
| TM 2x2 | 14.670 | מרכז עליון | 2 |
| Row 2 | 12.542 | שורה שנייה | 3 |
| Row 1 | 12.512 | שורה ראשונה | 4 |
| BM 2x2 | 12.084 | מרכז תחתון | 5 |
| L-shape | 2.316 | דפוס L | 6 |
| Top 3x2 | 0.611 | עליון 3x2 | 7 |
| Bottom 3x2 | 0.576 | תחתון 3x2 | 8 |

## 5. דיון וניתוח

### 5.1 הערכת הגישה

#### 5.1.1 יתרונות After-State Learning

הבחירה להשתמש בגישת After-State Learning סיפקה יתרונות בולטים בפרויקט, בעיקר מבחינת היציבות והיעילות של תהליך הלמידה. אחד הקשיים המרכזיים במשחק 2048 נובע מהאלמנט האקראי של הופעת אריח חדש לאחר כל פעולה. גישה זו עוקפת את הקושי על ידי כך שהיא מחשבת את ערך המצב לאחר ביצוע הפעולה, אך לפני הופעת אריח חדש. בכך היא מפחיתה את מרכיב אי-הוודאות בלמידה ומובילה לעדכוני TD מדויקים, עקביים ופחות רגישים לרעש מקרי. בנוסף לכך, מכיוון שהתוצאה של פעולה ניתנת לחישוב דטרמיניסטי מראש, ניתן לבצע את הלמידה על סמך תוצאה ברורה אחת, ובכך לייעל את תהליך האימון ולחסוך משאבים חישוביים.

#### 5.1.2 יתרונות N-tuple Representation

שיטת N-tuple אפשרה לייצג את כלל הלוח באמצעות אוסף דפוסים מקומיים חוזרים (כגון 2×2), אשר סובבו והשתקפו ל-8 וריאציות סימטריות , כך שנוצר כיסוי מרחבי מלא ויעיל של כל מצבי הלוח האפשריים. מהלך זה הפחית את מספר הפרמטרים הדרושים, שיפר את יכולת ההכללה, ואפשר למידה אפקטיבית גם ממצבים שקולים. יתרונה המרכזי של השיטה הוא השקיפות: כל פעולה של הסוכן נובעת ממשקלי תבניות ברורות וניתנות לפרשנות, בניגוד למודלים עמוקים הפועלים כ"קופסה שחורה".

## 5.2 מגבלות וחסרונות

### 5.2.1 מגבלות הגישה

למרות היתרונות המשמעותיים של השיטה, קיימות מספר מגבלות מהותיות. מגבלה ראשונה היא התלות בדפוסים שנבחרו מראש. הבחירה בדפוסים משפיעה באופן ישיר על יכולת ההכללה של המודל, ואם לא נבחרו דפוסים הכוללים אזורים קריטיים בלוח, הסוכן עלול לא ללמוד התנהגויות אופטימליות גם אם נתקל במצבים אלו במהלך האימון. מגבלה שנייה היא הסקלביליות- ככל שגדל מספר התאים בדפוס, כך עולה באופן מעריכי מספר הקומבינציות האפשריות, מה שמוביל לגידול דרמטי בגודל טבלאות ה-LUTS ובדרישות הזיכרון.

### 5.2.2 השוואה לשיטות מתקדמות

בהשוואה לגישות מתקדמות כמו DQN או MCTS, שיכולות ללמוד ייצוגים מורכבים ולבצע תכנון לטווח ארוך, השיטה שלנו פשוטה יותר אך מאפשרת להבין את תהליך קבלת ההחלטות בצורה ברורה. בעוד ש-DQN דורשת משאבים חישוביים גבוהים כמו שימוש ב-GPU ופועלת כ"קופסה שחורה", N-tuple מספקת שקיפות מלאה וניתנת ליישום גם על מחשבים סטנדרטיים אם כי היא פחות גמישה בהכללה לסביבות חדשות.

## 5.3 השפעת היפר-פרמטרים

### קצב למידה

- LR = 0.0025: למידה איטית מאוד, הסוכן אינו ממשש את מלוא הפוטנציאל.
- LR = 0.005: קצב מאוזן, מאפשר למידה יציבה וביצועים טובים.
- LR = 0.01: הקצב האופטימלי - מוביל ללמידה מהירה, יציבה ויעילה.
- LR = 0.05: למידה מהירה אך תנודתית מאוד, עם ירידה ביציבות הביצועים.

**מסקנה**: כיול קצב הלמידה הוא קריטי. קצב נמוך מדי גורם ללמידה איטית, בעוד שקצב גבוה מדי עלול לפגוע ביציבות.

## 6. מסקנות והמלצות להמשך

### 6.1 תרומות מרכזיות בפרויקט

הפרויקט מדגים תרומה משמעותית הן ברמה המתודולוגית והן ברמה הפרקטית. מהבחינה המתודולוגית, השילוב בין After-State Learning ל־N-tuple Representation יצר בסיס יציב ללמידה מדויקת, מהירה ויעילה, תוך הפחתת שונות בעדכונים והתכנסות גם בקצבי למידה גבוהים. ייצוג ה־N-tuple אפשר הכללה מרחבית של מצבים דומים (כגון סיבוב, השתקפות והזזה) תוך שמירה על פשטות חישובית. שילוב זה מדגים כיצד ניתן להגיע לביצועים גבוהים מבלי להסתמך על מודלים כבדים או משאבים מורכבים. מהבחינה הפרקטית, פותח סוכן קל לשימוש, שניתן להריץ על כל מחשב סטנדרטי. תכונות אלה הופכות את המערכת לכלי אידיאלי למחקר, הוראה ויישומים תעשייתיים שבהם נדרשת הבנה מעמיקה של תהליך הלמידה. בנוסף, המבנה המודולרי מאפשר התאמה והרחבה בקלות לסביבות נוספות או לשילוב רכיבי תכנון מתקדמים.

## 6.2 המלצות לפיתוח עתידי

המשך פיתוח המערכת יכול להתרכז בהרחבת הדפוסים, למשל לתצורות אלכסוניות או מורכבות יותר (3×3, 4×2), תוך שילוב מנגנוני סינון ואופטימיזציה שיזהו דפוסים משמעותיים ויעילים.
כמו כן, ניתן לשלב מנגנון לכוונון דינמי של משקלי התבניות בהתאם למצב הלוח ולשלב המשחק, כדי לשפר את הדיוק וההכללה מבלי להכביד על החישוב.
מן ההיבט המחקרי, מומלץ לבחון שילוב של After-State Learning עם רכיבי תכנון כמו Monte Carlo Tree Search או עם רשתות נוירונים - תוך שמירה על עקרונות השקיפות והפרשנות. בנוסף, כדאי לחקור את השפעת הסימטריות על קצב ההתכנסות, לאפיין תבניות אופטימליות באופן פורמלי, ולבדוק את הרחבת הגישה לבעיות מרחביות/זמניות מעבר ללוחות משחק.

## 6.3 לקחים נלמדים

מן הניתוח עולה כי קיימים שלושה לקחים עיקריים לעיצוב אלגוריתמי יעיל בלמידת חיזוקים בסביבה כגון 2048:

ראשית, שימוש ב-After-State Representation מפחית באופן משמעותי את השונות בתהליך הלמידה. מאחר וה-after-state מייצג מצב דטרמיניסטי (לפני הופעת אריח חדש), הוא מאפשר עדכונים מדויקים יותר ופחות מושפעים מהמרכיב הסטוכסטי של הסביבה.

שנית, ייצוג סימטרי של דפוסים מוכח כמאיץ למידה דרמטי. על ידי הרחבת כל תבנית לשמונה וריאציות סימטריות, ניתן לשפר את ההכללה בין מצבים ולצמצם את מספר הפרמטרים – מה שמוביל להתכנסות מהירה יותר של המודל.

ולבסוף, ניכר כי קצב הלמידה (learning rate) הוא פרמטר קריטי להצלחת האלגוריתם. קצב נמוך מדי מוביל ללמידה איטית ומוגבלת, ואילו קצב גבוה מדי עלול לפגוע ביציבות המודל. כיול מדויק נדרש להשגת ביצועים אופטימליים לאורך זמן.

## 6.4 סיכום

הפרויקט מציג שילוב אפקטיבי בין After-State Learning ל־N-tuple Representation, אשר יחד יצרו סוכן יעיל, שקוף ותחרותי למשחק 2048. הגישה איפשרה למידה מדויקת תוך הפחתת מורכבות חישובית והתמודדות מוצלחת עם אקראיות המשחק. הסוכן השיג תוצאות מרשימות וניתנות לפרשנות, בניגוד למודלים עמוקים הפועלים כ"קופסה שחורה".

מעבר לביצועים, תרומת הפרויקט טמונה במסגרת פשוטה אך עוצמתית ללמידת חיזוקים, המדגימה כיצד תכנון חכם, וכיול מושכל יכולים להוביל לפתרון יעיל, נגיש ובר-הרחבה. מדובר בבסיס מוצק למחקר עתידי וליישומים מעשיים בתחומים רחבים.

## 7. השראה מתודולגית

פרויקט זה קיבל השראה על ידי המאמר:

Lucas, S. M., & Togelius, J. (2019). *Temporal difference learning of N-tuple networks for the game 2048*. IEEE Conference on Games (CoG).

מאמר זה שימש כבסיס מתודולוגי לעבודתנו, היווה השראה לשימוש בייצוג מסוג N-tuple, לניצול סימטריות לצורכי הכללה.
במהלך הפרויקט יישמנו את עקרונות המאמר בהתאמה לפיתוח עצמאי, תוך שמירה על עקרונות הגישה ויישומה במשחק 2048.