האוניברסיטה הפתוחה
המחלקה למתמטיקה ולמדעי המחשב

למידה עמוקה באמצעות חיזוקים: תאוריה וייישומים עכשוויים

עבודה מסכמת זו הוגשה כחלק מהדרישות לקבלת תואר
"מוסמך למדעים" .M.Sc במדעי המחשב
באוניברסיטה הפתוחה
המחלקה למתמטיקה ולמדעי המחשב

על-ידי
חנן אהרונוף

העבודה הוכנה בהדרכתה של ד"ר מיריי אביגל

מאי 2019

<u>תוכן עניינים</u>

<u>רשימת איורים</u>

תקציר

למידה באמצעות חיזוקים (Reinforcement Learning) היא תת-תחום של למידה חישובית העוסק בהתקשרות של סוכן עם הסביבה ולמידה של מדיניות אופטימלית (Optimal Policy) על ידי ניסוי וטעייה. השילוב של למידה באמצעות חיזוקים עם למידה עמוקה אינו רעיון חדש, אך עם התקדמות המחקר של למידה עמוקה בשנים האחרונות, אנו עדים להתעוררות מחודשת בנושא עם מספר מאמרים מפורסמים אשר גדל בהתמדה משנה לשנה כבר שניים וחצי עשורים ברציפות. הופעתם של אלגוריתמים פורצי דרך כדוגמת DQN ו- AlphaGo היו להשראה למאות הרחבות ויישומים עבור בעיות ממגוון רחב של תחומי החיים. למרות כל ההצלחה הזו, עדיין ישנם עדיין אתגרים רבים העומדים בפני החוקרים והם אלו אשר יקבעו את כיוון המחקר העתידי. בעבודה זו אנו סוקרים את המחקר העכשווי של למידה עמוקה באמצעות חיזוקים, תוך כדי הצגת רקע תיאורטי נדרש, אלגוריתמים ויישומים מעשיים של התאוריה וכלה בדיון באתגרים העומדים בפני החוקרים ועתיד המחקר.

1. מבוא

למידה באמצעות חיזוקים (Reinforcement Learning) היא תת-תחום של למידה חישובית העוסק בהתקשרות של סוכן עם הסביבה ולמידה של מדיניות אופטימלית (Optimal Policy) על ידי ניסוי וטעייה, עבור בעיות החלטה סדרתיות (Sequential Decision Problems) מתוך קשת רחבה של תחומי חיים [1].

למידה עמוקה (Deep Learning), בניגוד ללמידה רדודה, מאופיינת ברשת בעלת מספר שכבות. הרשת מוזנת בצידה האחד במידע גולמי, אשר עובר שינוי כלשהו בכל שכבה, עד הגיעו אל השכבה האחרונה. השינוי שהמידע עובר בכל שכבה מיוצג על ידי אוסף של פרמטרים אשר משתנים בתהליך הלימוד, עד אשר מגיעים לערכי התכנסות. עדכון הפרמטרים מתבצע באמצעות שימוש באלגוריתם התפשטות אחורי (Backpropagation) [2]. למידה עמוקה הביאה לפריצות דרך בתחומים רבים כדוגמת עיבוד תמונה ווידאו, ניתוח קול ושפה, משחקי מחשב, נהיגה אוטונומית וכד׳ [3].

השילוב של למידה באמצעות חיזוקים עם למידה עמוקה אינו רעיון חדש [4], אך עם התקדמות המחקר של למידה עמוקה בשנים האחרונות, אנו עדים להתעוררות מחודשת בנושא [5] עם מספר מאמרים מפורסמים אשר גדל בהתמדה משנה לשנה כבר שניים וחצי עשורים ברציפות [6] . הופעתם של אלגוריתמים פורצי דרך כדוגמת DQN [7], AlphaGo ו- AlphaZero [8], Dueling Networks [9], Deterministic Policy Gradient [10] ו- A3C [11] היו להשראה למאות הרחבות ויישומים עבור בעיות ממגוון רחב של תחומי החיים.

למידה באמצעות חיזוקים דורגה על ידי המכון הטכנולוגי של מסצ׳וסטס (MIT) כאחת מעשר הטכנולוגיות פורצות הדרך של שנת 2017 [13] ולמידה עמוקה דורגה באותה רשימה בשנת 2013 [14]. חוקרים רבים בתחום מחשיבים את השילוב של למידה עמוקה ולמידה באמצעות חיזוקים כצעד הראשון בדרך ליצירתה של מסגרת תאורטית של בינה מלאכותית כללית, כלומר, כזו שיכולה לבצע כל משימה שאדם יכול לבצע [15]. סילבר, התורם העיקרי של AlphaGo [8] קבע את הנוסחה:

בינה מלאכותית כללית = למידה באמצעות חיזוקים + למידה עמוקה [16]

בחלוף שש שנים של מחקר פורה ועתיר בפריצות דרך של למידה עמוקה באמצעות חיזוקים, ולאחר פרסומם של עשרות מחקרים העוסקים ביישומים מעשיים של התאוריה, ישנם עדיין אתגרים רבים העומדים בפני החוקרים. אתגרים אלו הם שיקבעו את כיוון המחקר העתידי.

מטרתה של עבודה זו היא לסקור את המחקר העכשווי של למידה עמוקה באמצעות חיזוקים, תוך כדי הצגת רקע תיאורטי נדרש, אלגוריתמים ויישומים מעשיים של התאוריה וכלה בדיון באתגרים העומדים בפני החוקרים ועתיד המחקר.

העבודה מחולקת לחמישה חלקים באופן הבא; החלק הראשון מהווה סקירה תאורטית של למידה באמצעות חיזוקים ולמידה עמוקה. החלק השני מציג כיצד ניתן לשלב למידה עמוקה עם למידה באמצעות חיזוקים ודן במספר אלגוריתמים עכשוויים העוסקים בכך. החלק השלישי מתאר שלל מחקרים של יישומים

עכשוויים העוסקים בפתרון בעיות במגוון תחומי חיים. הפרק הרביעי מתמקד ביתר שאת בשני יישומים מהפרק השלישי. ולבסוף החלק החמישי עוסק באתגרים העומדים בפני המחקר העכשווי ובעתידו.

2. רקע תיאורטי

למידה באמצעות חיזוקים זוכה בשנים האחרונות להתעוררות והתעניינות מחדש, יש שיגדירו זאת כתקופת הרנסנס של תחום מחקר שזכה למעט התייחסות בשני העשורים האחרונים. ואכן בשנים האחרונות למידה באמצעות חיזוקים הפכה מבוקשת עם הופעתם חדשות לבקרים של רעיונות חדשים והצלחות יישומיות מדהימות. ההתעניינות הגוברת בתחום הגיעה לנקודת מפנה עם הופעתו של אלגוריתם DQN [7] שהראה כיצד ניתן לשלב אלגוריתמים של למידה באמצעות חיזוקים יחד עם רשתות נוירונים עמוקות (במה שקרוי למידה עמוקה) כדי ללמד מחשב לשחק משחקי וידאו ברמה של אדם. אלגוריתם זה סימן את בואם של אלגוריתמים רבים הנשענים על עיקרון השילוב של רשתות נוירונים ולמידה באמצעות חיזוקים המגדירים תחום מחקר חדש הקרוי למידה עמוקה באמצעות חיזוקים.

המחקר פורץ הדרך העכשווי בתחום של למידה עמוקה באמצעות חיזוקים נשען במידה רבה על התאוריה של למידה באמצעות חיזוקים ולמידה עמוקה. בכדי לקיים דיון מעמיק במחקרים העכשוויים של למידה עמוקה באמצעות חיזוקים עלינו להניח את היסודות התיאורטיים הנדרשים. על כן, פרק זה יעסוק בתאוריה של למידה באמצעות חיזוקים (סעיף מספר 1) ולמידה עמוקה (סעיף מספר 2), בעוד שבפרק הבא נדון במחקרים העכשוויים ופרק השלישי נעסוק ביישומם.

2.1 למידה באמצעות חיזוקים

למידה באמצעות חיזוקים (Reinforcement Learning) הינה תחום מחקר של למידה חישובית (Machine Learning), העוסקת בדרך בה סוכן (Agent) (תוכנת מחשב המהווה את היחידה הלומדת ומקבלת ההחלטות) צריך לפעול בסביבה מסוימת בכדי להגדיל תגמול מצטבר (Cumulative Reward) כלשהו. תחת תחום זה אנו בוחנים כיצד הסוכן יכול ללמוד מהצלחה וכישלון, מתגמול ועונש.

למידה באמצעות חיזוקים עוסקת בבעיות מסוג החלטה סדרתית (Sequential Decision Problems), אשר בהם התגמול של הסוכן תלוי בסדרה של החלטות (פעולות) שהוא מבצע בסביבה. הסוכן נע בסביבה וחש אותה, בהתאם לכך הוא מקבל החלטה כיצד לפעול בצעד הבא בכדי להשיג את המטרה. תוצאת פעולה מזכה את הסוכן בתגמול כלשהו (חיובי או שלילי), אשר בתורו משפיע על החלטות עתידיות של הסוכן. כמו כן, סביבות עולם אמיתיות מאופיינות בחוסר וודאות לגבי הצלחה או כישלון של פעולה, ועל הסוכן לקחת זאת גם כן בחשבון [17].

2.1.1 יחידות הבסיס של למידה באמצעות חיזוקים

מלבד הסוכן והסביבה, למידה באמצעות חיזוקים מאופיינת על ידי ארבע יחידות בסיסיות: מדיניות (Policy), תגמול (Reward), פונקציית ערך (Value Function) ומודל (Model) [1].

מדיניות מגדירה את ההתנהגות של סוכן בכל רגע נתון. ליתר דיוק, מדיניות היא מיפוי בין מצבים נתפסים של הסביבה לפעולות שהסוכן יכול לבצע. ניתן להגדיר מדיניות באופן פשוט, קרי טבלת חיפוש, או באופן מורכב יותר כדוגמת פונקציות קירוב.

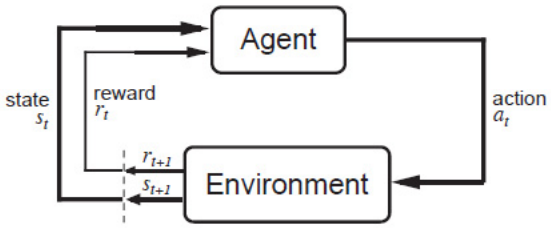
תגמול מגדיר את המטרה של בעיית למידה באמצעות חיזוקים. בכל יחידת זמן, הסוכן מקבל מן הסביבה מספר בודד והוא מהווה את התגמול של הסוכן. מטרת הסוכן היא למקסם את התגמולים שהוא אוסף לאורך זמן. הסוכן לא יכול לשנות את הפונקציה אשר מייצרת את התגמול, אלא רק לבצע פעולות בסביבה אשר בתורם עלולים להשפיע על התהליך שמייצר את התגמול בסביבה.

פונקציית ערך מגדירה תגמול בטווח הרחוק, רוצה לומר, הערך של מצב הוא סך כל התגמולים שהסוכן מצפה לקבל בעתיד, אם יתחיל מאותו מצב. בעוד שתגמול מגדיר את הכדאיות המיידית של מצב, פונקציית הערך מגדירה את הכדאיות שלו בטווח הארוך.

מודל מדמה את הסביבה בה הסוכן פועל. בהינתן מודל לסביבה, ניתן לתכנן כיצד על הסוכן לפעול בכל מצב. שיטות פתרון של למידה באמצעות חיזוקים אשר משתמשים במודלים נקראים מבוססי מודל (Model-Based), זאת בניגוד לשיטות פשוטות של ניסוי וטעייה אשר נקראות מחוסרי מודל (Model-Free). אלגוריתמי למידה באמצעות חיזוקים נעים על המנעד שבין שני סוגי המודלים.

2.1.2 הגדרת בעיית למידה באמצעות חיזוקים

בעיית למידה באמצעות חיזוקים מגדירה מסגרת לבעיית למידה באמצעות התקשרות עם הסביבה בכדי להשיג מטרה מוגדרת כלשהי (למשל, הגעה למקום מסוים או צבירת נקודות). בבעיה מסוג זה אנו מגדירים *סוכן* אשר מהווה את היחידה הלומדת ומקבלת ההחלטות וכן מגדירים *סביבה* אשר מהווה את כל מה שמצוי מחוץ לסוכן. הסוכן נמצא בקשר רציף עם הסביבה, הוא מבצע עליה פעולות וזו מגיבה להן על ידי מתן תגמולים (חיובים או שלילים) [1]. איור 1.1 מדגים את האינטראקציה בין הסוכן לבין הסביבה.



איור 1.1 – אינטראקציה בין סוכן לסביבה בבעיית למידה באמצעות חיזוקים. הסוכן נמצא בקשר רציף עם הסביבה, הוא מבצע עליה פעולות וזו מגיבה להן על ידי מתן תגמולים (חיובים או שלילים) [1]

להלן הגדרה פורמאלית לבעיית למידה באמצעות חיזוקים. תהא P בעיית למידה באמצעות חיזוקים המורכבת משש האברים הבאים:

- S – אוסף מצבים.
- A – אוסף פעולות שהסוכן יכול לבצע.
- $\mathrm{T}(s,a,s')$ – פונקציית מעברים סטוכסטית, כך ש $s,s' \in S, a \in A$.
- $\mathrm{R}(s,a,s')$ – פונקציית תגמולים, כך ש $s,s' \in S, a \in A$.
- מצב התחלה $s_0 \in S$.
- מצב סיום $s_f \in S$ – אופציונלי.

כאשר פונקציית המעברים מתארת את התוצאה של פעולה בכל מצב. היות ואנו עוסקים בסביבות המאופיינות בחוסר וודאות, פונקציית המעברים הינה סטוכסטית, וניתן להגדיר $\mathrm{T}(s,a,s') = P(s'|s,a)$.

2.1.3 תגמול מצטבר (תשואה)

כאמור, מטרתו של הסוכן היא למקסם את התגמולים שהוא אוסף לאורך זמן. בכל יחידת זמן הסוכן מקבל תגמול מהסביבה בהתאם למצב בו הוא נמצא. נגדיר את התגמול המצטבר G_t (או תשואה) באופן הבא:

$$G_t = R_{t+1} + R_{t+2} + \cdots + R_T$$

כאשר R_i הם התגמולים המתקבלים ביחידות הזמן השונות. על כן, על הסוכן למקסם את G_t. גישה זו הגיונית עבור בעיות בהן הסוכן פועל בפרקים (*Episodes*), כלומר, ההילוך של הסוכן בסביבה נעשה בפרקים; התחלה במצב כלשהו, ביצוע מספר פעולות סופי, הגעה למצב סופי וחוזר חלילה. בבעיות בהן ההילוך של הסוכן הוא רציף ואינו מוגבל (לדוגמא מכונית אוטונומית), פונקציית התגמול המצטבר בעייתית, שכן G_t עלול להיות אינסופי.

נגדיר מקדם הפחתה (Discount Rate) γ עם $0 \leq \gamma \leq 1$ כדי לקבוע את הערך הנוכחי של תגמולים עתידיים באופן הבא:

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

כאשר $\gamma < 1$, וסדרת התגמולים חסומה, הטור האינסופי G_t מתכנס למספר סופי. הבחירה של γ קובעת את מידת ההתייחסות של הסוכן לתגמולים עכשוויים לעומת תגמולים עתידיים. ככל ש- γ מתקרב ל 1, הסוכן יתחשב יותר בתגמולים עתידיים וככל שיתקרב ל 0, כך יעדיף תגמולים עכשוויים [1].

2.1.4 התכונה המרקובית ותהליך החלטה מרקובי

יהי מצב S של הסביבה, נאמר כי S הוא בעל תכונה מרקובית (Markov Property) אם ורק אם הוא מכיל בתוכו את כל המידע הרלוונטי שנאסף עד הרגע שבו הסוכן ביקר בו לראשונה. בהינתן שכל המצבים בסביבה הינם בעלי התכונה המרקובית הרי שהתגובה של הסביבה ברגע t+1 תלוי אך ורק במצב והפעולה ברגע t. כלומר, ניתן לנבא את המצב והתגמול הבאים בהינתן המצב והפעולה הנוכחיים [1], ומתקבל:

$$P(s', r|s, a) = \Pr\{S_{t+1} = s', R_{t+1} = r|S_t = s, A_t = a\}$$

לכל 's ו r.

סביבה בעלת התכונה המרקובית מאופיינת באמצעות תהליך החלטה מרקובי (Markov Decision Process) או MDP. אם מרחב המצבים הינו סופי זה נקרא תהליך החלטה מרקובי סופי. התאוריה של למידה באמצעות חיזוקים עוסקת לרוב בתהליכים מסוג זה. עבור MDP סופי ניתן לחשב את התגמול הצפוי לכל זוג מצב-פעולה:

$$r(s, a) = \mathrm{E}[R_{t+1}|S_t = s, A_t = a] = \sum_{r \in \mathcal{R}} r \sum_{s' \in \mathcal{S}} p(s', r|s, a)$$

וכן את ההסתברויות של פונקציית המעברים:

$$p(s'|s, a) = \Pr\{S_{t+1} = s'|S_t = s, A_t = a\} = \sum_{r \in \mathcal{R}} p(s', r|s, a)$$

2.1.5 פונקציות ערך

פתרון לבעיית למידה באמצעות חיזוקים חייב לתאר מה הסוכן צריך לעשות בכל מצב שאליו הוא מגיע, שכן היות והסוכן פועל בחוסר וודאות, פתרון המתאר סדרה קבועה של צעדים עלול להוביל אותו למצב שאיננו מצב הסיום. פתרון מסוג זה קרוי מדיניות ונהוג לסמנו ב- π, כאשר $\pi(s)$ היא הפעולה המומלצת על ידי המדיניות במצב s. עם מדיניות מלאה, כלומר מיפוי בין כל מצב לכל פעולה, הסוכן יכול לפעול בכל מצב ובהתאם לכל תוצאה של פעולה. בכל פעם שמדיניות נתונה מבוצעת החל ממצב ההתחלה, האופי האקראי של הסביבה עלול להוביל לתגמול מצטבר שונה עבור הסוכן. כלומר, איכותה של מדיניות נקבעת על פי תוחלת התועלת של התגמולים המתקבלים מהפעלתה. מדיניות אופטימלית $\pi \star$ מוגדרת להיות המדיניות אשר ממקסמת את תוחלת התגמול המצטבר.

נגדיר את $V^*(s)$ להיות תוחלת התגמול המצטבר כאשר הסוכן מתחיל ממצב s ופועל באופן אופטימלי לאחר מכן. כמו כן, נגדיר $Q^*(s,a)$ להיות תוחלת התגמול המצטבר כאשר הסוכן מתחיל מביצוע פעולה a במצב s ופועל באופן אופטימלי לאחר מכן (מצבים אלו קרואים מצבי-q). מתקיים ש:

$$V^*(s) = \max_{a \in A} Q^*(s,a)$$

$$Q^*(s,a) = \sum_{s' \in S} T(s,a,s')[R(s,a,s') + \gamma V^*(s)]$$

כלומר,

$$V^*(s) = \max_{a \in A} \sum_{s' \in S} T(s,a,s')[R(s,a,s') + \gamma V^*(s)]$$

משוואה זו נקראת משוואת בלמן (Bellman Equation). עבור MDP סופי למשוואה זו יש פתרון יחיד בלתי תלוי במדיניות. משוואה זו היא בעצם מערכת משוואות כמספר המצבים המוגדרים ב MDP. כלומר, בהינתן $p(s', r|s, a)$ של הסביבה ניתן לפתור מערכת זו. כאשר $V^*(s)$ ידוע ניתן לחשב מדיניות אופטימלית באופן הבא:

$$\pi^*(s) = \operatorname*{argmax}_{a \in A} \sum_{s' \in S} T(s,a,s')[R(s,a,s') + \gamma V^*(s)]$$

2.1.6 <u>פתרונות מבוססי תכנון דינאמי</u>

המינוח "מבוססי תכנון דינאמי" מתייחס לאוסף אלגוריתמים אשר יכול לחשב מדיניות אופטימלית בהינתן מודל מושלם של הסביבה, כדוגמת תהליך החלטה מרקובי [1]. כאמור, אם סביבה מיוצגת על ידי N מצבים, הרי שמשוואת בלמן המייצגת אותה היא מערכת בת N משוואות. אלא שמשוואות אלו אינן לינאריות, ועל כן לא ניתן לפתור אותן בשיטות של אלגברה לינארית. אחת הגישות לפתרון מערכת משוואות לא לינאריות היא חיזרור (Iteration) [17]. הסעיפים הבאים מביאים דוגמאות לשני אלגוריתמים הפועלים באופן זה.

2.1.6.1 <u>חיזרור ערכים</u>

נגדיר $V_k(s)$ להיות התועלת האופטימלית של מצב s אם הסוכן עוצר אחרי k צעדים. נוכל להגדיר את $V_{k+1}(s)$ על ידי הצעד הרקורסיבי,

$$V_{k+1}(s) = \max_{a \in A} \sum_{s' \in S} T(s,a,s')[R(s,a,s') + \gamma V_k(s)]$$

מכיוון שמקדם ההפחתה קטן ממש מאחד, כאשר $k \to \infty$,$V_k(s)$ מתכנס. האלגוריתם חיזרור ערכים (Value Iteration) מנצל עובדה זו כדי לספק פתרון לאוסף משוואות אלו. על פי אלגוריתם זה, יש לחשב תחילה את $V_0(s)$, ולהמשיך לחשב $V_k(s)$ על פי הנוסחה הרקורסיבית. האלגוריתם נפסק כאשר $V_k(s)$ מתכנס [17]. משיש בידנו את $V^*(s)$ ניתן לחשב את המדיניות האופטימלית על ידי בחירה לכל מצב של הפעולה שממקסמת את הרווח כפי שהוגדר בסעיף 1.1.5.

2.1.6.2 חיזרור מדיניות

נגדיר $V_k^\pi(s)$ להיות תוחלת התועלת של מצב s כאשר מבוצעת מדיניות π והסוכן עוצר לאחר k מצבים.
נגדיר את $V_{k+1}^\pi(s)$ באופן רקורסיבי:

$$V_{k+1}^\pi(s) = \sum_{s' \in S} T(s, \pi(s), s')[R(s, \pi(s), s') + \gamma V_k^\pi(s')]$$

מאותם טיעונים של אלגוריתם חיזרור ערכים, מתקיים שכאשר $k \to \infty$ אז $V_k^\pi(s)$ מתכנס. על ידי שימוש בערכי $V_k^\pi(s)$ ניתן לשפר את המדיניות π, על ידי כך שנבחר את הפעולה הראשונית הטובה ביותר ולאחר מכן נפעל על פי π:

$$\pi_{i+1}(s) = \underset{a \in A}{arg\max} \sum_{s' \in S} T(s, a, s')[R(s, a, s') + \gamma V^{\pi_i}(s')]$$

על פי האלגוריתם חיזרור מדיניות (Policy Iteration) יש לבחור תחילה מדיניות π כלשהי ולאחר מכן יש לחשב את ערכי $V_k^\pi(s)$ עד התכנסות. מתוך ערכים אלו מחשבים מדיניות משופרת. חוזרים על שני הצעדים האחרונים עד התכנסות אל המדיניות האופטימלית [17].

2.1.6.3 יעילות פתרונות מבוססי תכנון דינאמי

נסמן ב- n ו- k את מספר המצבים בסביבה ומספר הפעולות שניתן לבצע בה בהתאמה. אלגוריתמים מבוססי תכנון דינאמי דורשים מעבר על כל המצבים וכל הפעולות בהם. כלומר, במקרה הגרוע זמן הריצה של אלגוריתמים מסוג זה הינו פולינומי בגודל הקלט. כמו כן, מובטח כי אלגוריתמים מסוג זה מוצאים תמיד את הפתרון האופטימאלי. הפתרון הנאיבי ידרוש מעבר על פני כל המרחב והרי מרחב המדיניות הינו k^n כלומר, פתרונות מבוססי תכנון דינאמי מהירים בסדר גודל מעריכי מהפתרון הנאיבי [1]. אומנם, לעיתים פתרונות מבוססי תכנון לינארי מהירים יותר, אך פתרונות אלו מתמודדים עם מרחב מצבים קטן בהרבה (פי 100 פחות).
אף על פי שיעילותם של אלגוריתמים מסוג זה טוב בהרבה מהפתרון הנאיבי, הרי שבסביבות עם מספר עצום של מצבים ופעולות זמן החישוב הארוך לא מאפשר שימוש סביר בהם. במחקר משנת 1957 העוסק בשליטה אופטימלית במרחבים רבי ממדים, טבע בלמן (Bellman) [18] את הביטוי קללת המימדיות (Curse of Dimensionality) כדי לתאר את ההתפוצצות המעריכית של פתרונות מבוססי תכנון דינאמי כאשר מספר הצירים (מצבים ופעולות) גדל.

2.1.7 פתרונות מונטה קרלו

בניגוד לפתרונות המבוססים על תכנון דינאמי, פתרונות המבוססים על היוריסטיקת מונטה קרלו אינן מניחים ידע מלא של הסביבה. שיטות אלו דורשות רק ניסיון (Experience). ניסיון מוגדר כאוסף של מצבים, פעולות ותגמולים מאינטראקציה אמיתית עם הסביבה [1]. שיטות מונטה קרלו (Monte Carlo) עוסקות בבעיות למידה באמצעות חיזוקים המורכבות מפרקים (Episodes).

2.1.7.1 שיטת מונטה קרלו למציאת מדיניות אופטימלית תוך שיפור מדיניות קיימת

שיטת מונטה קרלו למציאת מדיניות אופטימלית תוך שיפור מדיניות קיימת (On-Policy Monte Carlo Control) דומה מאוד במבנה לאלגוריתם של חזרור מדיניות שהוצג בסעיף הקודם במובן זה ששני האלגוריתמים מבצעים הערכה של המדיניות ושיפור שלה וחוזר חלילה עד להתכנסות.

הערכה של מדיניות נעשית על ידי קירוב ערכי $Q(s,a)$. קירוב זה נעשה על ידי מיצוע ערכי התגמול המתקבלים מביקור של הסוכן במצב s וביצוע פעולה a בכל פרק (כלומר סדרה של פעולות ממצב התחלתי ועד מצב סופי). ישנם שני סוגי מיצוע שניתן לעשות, קרי, מיצוע עבור הערך של הביקור הראשון בכל פרק (First Visit Monte Carlo) או מיצוע של כל הערכים בכל הביקורים בכל הפרקים (Every Visit Monte Carlo). התאוריה מוכיחה כי שתי השיטות מתכנסות אל תוחלת ערכי Q בקצב ריבועי פוחת ככל שדגימות הזוגות s ו a מתקרב לאינסוף ובלבד שהדגימות מכסות את כל מרחב מצב-פעולה של הסביבה. שיפור המדיניות נעשה בדיוק כמו שהוגדר באלגוריתם חזרור מדיניות, כלומר לכל מצב נחשב את הפעולה a שמייצרת את התגמול הגבוה ביותר לפי הקירוב של $Q(s,a)$ שחושב בשלב ההערכה. כדי להבטיח שהדגימות יכסו את מרחב מצב-פעולה יש צורך בהפעלת מנגנון אשר יסטה בהסתברות קטנה מבחירת הפעולה שממקסמת את התגמול בכל מצב. מנגנון $\varepsilon-\text{greedy}$ פועל באופן הבא: המדיניות שנקבעת בשלב השיפור תבצע את הפעולה שממקסמת את התגמול לכל שלב בהסתברות $1-\varepsilon+\frac{\varepsilon}{|\mathcal{A}(s)|}$ וכל פעולה אחרת בהסתברות $\frac{\varepsilon}{|\mathcal{A}(s)|}$ כאשר ε הוא מספר קטן ו- $|\mathcal{A}(s)|$ הוא גודל מרחב מצב-פעולה.

2.1.7.2 שיטת מונטה קרלו למציאת מדיניות אופטימלית באמצעות מדיניות קיימת

בניגוד לשיטת מונטה קרלו למציאת מדיניות אופטימלית תוך שיפור מדיניות קיימת מהסעיף הקודם, בה משתמשים במדיניות אחת להערכה ולשיפור, בשיטת מונטה קרלו למציאת מדיניות אופטימלית באמצעות מדיניות קיימת (Off-Policy Monte Carlo Control) משתמשים בשתי מדיניויות; מדיניות התנהגות (Behavior Policy) אשר ממנה דוגמים ומדיניות מטרה (Target Policy) אשר לה מבצעים הערכה ושיפור בדומה לשיטה הקודמת. היתרון בפיצול זה הוא בכך שמדיניות ההתנהגות אינה חייבת להיות קשורה למדיניות המטרה, יתרה מכך, בעוד מדיניות המטרה יכולה להיות דטרמיניסטית מדיניות ההתנהגות יכולה להיות לא דטרמיניסטית, שכן בסופו של דבר משתמשים במדיניות המטרה בלבד. האלגוריתם ממומש כמו האלגוריתם הקודם, אלא שמשתמשים במדיניות ההתנהגות כדי לייצר דגימות ובמדיניות המטרה להערכה ועדכון.

2.1.8 פתרונות מבוססי הפרש טמפורלי

שיטת הפרש טמפורלי (Temporal Difference) היא שיטה מרכזית בלמידה באמצעות חיזוקים. שיטה זו משלבת אלמנטים משיטות מונטה קרלו ומשיטות תכנון דינאמי. בדומה למונטה קרלו, שיטת זו לומדת מתוך ניסיון (Experience) ללא מודל של הסביבה, ובדומה לתכנון דינאמי שיטה זו מעדכנת את הערכותיה על הערכות שנלמדו כבר וזאת מבלי להמתין עד שהתוצאה הסופית תתקבל (תהליך הנקרא Bootstrapping).

שיטת הפרש טמפורלי הפשוטה ביותר נקראת TD(0) והיא מאופיינת כך:

$$V(S_t) \leftarrow V(S_t) + \propto [R_{t+1} + \gamma V(S_{t+1}) - V(S_t)]$$

כאשר α הינו קצב הלמידה (Learning Rate). עם $\alpha = 1$, הלימוד מתבצע אך ורק על סמך דגימות חדשות, בעוד שעם $\alpha = 0$ אין לימוד כלל מדגימות. בכדי להגיע להתכנסות יש להגדיר את α כפונקציית דעיכה ל – 0 עם הזמן.

שיטות המבוססות על הפרש טמפורלי עדיפות על שיטות תכנון דינאמי משום שהן אינן דורשות ידע על המודל של הסביבה. שיטות אלו עדיפות על פני שיטות מונטה קרלו מכיוון שהן אינן דורשות להמתין עד סוף הפרק בכדי לבצע עדכון של הערכים, ועל כן לרוב ההתכנסות מהירה יותר. כמו כן, ניתן להשתמש בשיטות אלו כדי לפתור בעיות רציפות שאינן מאופיינות בפרקים.

2.1.8.1 <u>שיטת הפרש טמפורלי למציאת מדיניות אופטימלית תוך שיפור מדיניות קיימת</u>

שיטת הפרש טמפורלי למציאת מדיניות אופטימלית תוך שיפור מדיניות קיימת (SARSA - State–action–reward–state–action) דומה במבנה לאלגוריתם חזרור מדיניות ואלגוריתמי מונטה קרלו שהוצגו בסעיפים הקודמים במובן זה שכל אלו מבצעים הערכה של המדיניות ושיפור שלה וחוזר חלילה עד להתכנסות. הערכה של מדיניות נעשית על ידי קירוב ערכי $Q(s,a)$ באמצעות שימוש ב TD(0). להלן כלל העדכון המתאים:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \propto [R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)]$$

העדכון נעשה אחרי כל מעבר מכל מצב למצב S_t. שיפור המדיניות נעשה בדיוק כמו שהוגדר באלגוריתם חזרור מדיניות, כלומר לכל מצב נחשב את הפעולה a שמייצרת את התגמול הגבוה ביותר לפי הקירוב של $Q(s,a)$ שחושב בשלב ההערכה. כמו כן נשתמש במנגנון $\varepsilon - \text{greedy}$ כדי להבטיח כסוי למרחב מצב-פעולה. ההערכה הרווחת כי אלגוריתם זה מתכנס לפתרון האופטימלי ככל שמספר זוגות מצב-פעולה מתקרב לאינסוף [1].

2.1.8.2 <u>הפרש טמפורלי למציאת מדיניות אופטימלית באמצעות מדיניות קיימת</u>

אלגוריתם הפרש טמפורלי למציאת מדיניות אופטימלית באמצעות מדיניות קיימת (Q-Learning) נחשב לפריצת דרך בהתפתחות של למידה באמצעות חיזוקים. אלגוריתם זה שונה מ SARSA במובן זה שהוא מחשב מדיניות אופטימלית באופן בלתי תלוי מהמדיניות שהוא עוקב אחריה, כלומר זהו אלגוריתם מסוג Off-Policy. להלן כלל העדכון המתאים:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \propto \left[R_{t+1} + \gamma \max_a Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)\right]$$

המשך האלגוריתם זהה לזה מהסעיף הקודם.

2.1.9 פתרונות קירוב

עד כה כלל הפתרונות שהוצגו (תכנון דינאמי, מונטה קרלו והפרשי רגעים) משתייכים למשפחת הפתרונות הטבלאיים (Tabular). שיטות השייכות למשפחה זו מוצאות את הפתרון האופטימאלי על ידי מעבר על כל המצבים והפעולות. אי לכך כמות הזיכרון והזמן הנדרשים נמצאים ביחס ישיר לגודל מרחב מצב-פעולה של הבעיה. אולם בעיות רבות מאופיינות במרחבי מצב-פעולה עצומים והשימוש בשיטות אלו אינו ישים עבורן.

פתרונות קירוב (Approximation) מקריבים את האופטימליות של הפתרון בתמורה לפתרון מקורב אך פתיר בזמן ובמשאבים סבירים. בפתרונות אלו אנו מחפשים לקרב פונקציה על ידי הכללה (Generalization) של מספר קטן של דוגמאות מתוך הפונקציה.

בפרק 2 נתמקד בשני אופנים לשילוב פונקציות קירוב באלגוריתמים של למידה באמצעות חיזוקים, קרי קירוב פונקציית הערך וקירוב המדיניות עצמה על ידי שימוש ברשתות נוירונים עמוקות. בחלק זה נביא את הרקע התיאורטי הנדרש באלגוריתמים של קירוב מדיניות.

2.1.9.1 קירוב מדיניות

בקירוב מדיניות מקצים למדיניות אוסף של פרמטרים θ אשר המדיניות נקבעת על פיהם. אוסף זה יכול להיות כל אוסף ובלבד שהמדיניות $\pi(a|s,\theta)$ תחת אוסף זה גזירה, כלומר, $\nabla_\pi \pi(a|s,\theta)$ קיים [1]. בפרט θ יכול להיות אוסף משקולות המחושב על ידי רשת נוירונים עמוקה ואכן כפי שנראה המחקר העכשווי עוסק בקירוב מדיניות באמצעות רשתות נוירונים עמוקות.

ישנם מספר יתרונות בקירוב מדיניות על פני קירוב פונקציית ערך. ראשית, בעיות נבדלות זאת מזאת בסיבוכיות של המדיניות בפונקציות הערך שלהם. בעבור חלקן פונקציות הערך פשוטות יותר ולכן קלות יותר לקירוב, בעוד שלאחרות המדיניות פשוטה יותר. על כן בבעיות בהן המדיניות פשוטה יותר קירוב מדיניות יתבצע מהר יותר ויוליד מדיניות טובה יותר [1]. שנית, באמצעות שימוש בפרמטרים מעל המדיניות ניתן להזריק ידע קודם אל תהליך הלמידה ובכך להאיץ אותו. לבסוף, תחת קירוב מדיניות ניתן לייצר מדיניות סטוכסטית כלומר מדיניות שמקיימת התפלגות על הפעולות. מדיניות סטוכסטית היא לעיתים המדיניות האופטימאלית בבעיות עם ידע חלקי, לדוגמה משחק פוקר.

2.1.9.2 שיטות גראדיינט המדיניות

שיטות המבוססות על גראדיינט המדיניות (Policy Gradient Methods) פועלות בסבבים כאשר בכל סיבוב המטרה היא לשפר את המדיניות ביחס לגראדיינט של מדד ביצועים התלוי בפרמטרים של המדיניות. מדד ביצועים למדיניות הוא מדד השוואתי אשר בעזרתו ניתן להעריך אם מדיניות השתפרה לאחר עדכון של הפרמטרים בהם היא תלויה. נגדיר $J(\theta) = v_{\pi_\theta}(s_0)$ להיות מדד ביצועים כך ש- v_{π_θ} היא פונקציית הערך האמתית של π_θ, המדיניות שנקבעת על ידי θ, ו- s_0 הוא מצב התחלתי ספציפי. היות וברצוננו למקסם את הביצועים של המדיניות, עלינו לנוע בעקבות העלייה של הגראדיינט (Gradient Ascent) ביחס ל J. נגדיר אפוא את כלל העדכון הבא:

$$\theta_{t+1} = \theta_t + \alpha\widehat{\nabla J(\theta_t)}$$

כאשר $\widehat{\nabla J(\theta_t)}$ הוא הערכה סטוכסטית אשר התוחלת שלה מקרבת את הגראדיינט של מדד הביצועים ביחס לפרמטרים θ_t. ביטוי זה מהווה את הסכמה הכללית של כל שיטות גראדיינט המדיניות.

יש לזכור כי כאשר מקרבים פונקציות ישנו אתגר בשינוי הפרמטרים של המדיניות באופן שמבטיח שיפור. הבעיה היא שביצועי המדיניות תלויים בפעולות שנבחרו עם התפלגות המצבים בהם נעשו הבחירות האלו, וכן העובדה ששני אלה מושפעים מהפרמטרים של המדיניות. אמנם בהינתן מצב, ניתן לחשב את ההשפעה של פרמטר של המדיניות על הפעולות בקלות יחסית מתוך הידע על הפרמטרים, אך לא ניתן להעריך את ההשפעה של המדיניות על התפלגות המצבים שכן זוהי פונקציה של הסביבה והיא אינה ידועה [1].

תיאורית גראדיינט המדיניות (Policy Gradient Theory) [19] מספקת ביטוי אנאליטי של גראדיינט ביצועי המדיניות ביחס לפרמטרים אשר לא תלוי בנגזרת של התפלגות המצבים:

$$\nabla J(\theta) = \sum_s \mu(s) \sum_a q_\pi(s,a)\, \nabla_\pi\, \pi(a|s,\theta)$$

כאשר $\nabla J(\theta)$ הוא וקטור של נגזרות חלקיות ביחס לפרמטרים θ והוא למעשה הגראדיינט של מדד הביצועים, π היא המדיניות המתאימה לפרמטרים θ ו-μ היא התפלגות המצבים תחת המדיניות π.

אלגוריתמים המבוססים על תיאורית גראדיינט המדיניות שייכים למשפחת אלגוריתמים הקרויה REINFORCE. דוגמה לאלגוריתם פשוט ממשפחה זו הוא אלגוריתם REINFORCE מונטה קרלו [1]. אלגוריתם זה עוקב אחרי הסכמה הכללית של אלגוריתמים מסוג גראדיינט המדיניות כפי שהוצגה קודם וכן בתוצאה של תיאורית גראדיינט המדיניות בכדי לחשב את כלל העדכון. נשים לב שהצג הימני של הנוסחה האחרונה הוא בעצם סכום מעל מצבים אשר ממושקלים בכמות הפעמים שכל מצב קורה תחת המדיניות π שהיא עצמה ממושקלת על ידי γ צעדים הנדרשים בכדי להגיע לאותם מצבים. על כן, אם נעקוב אחרי המדיניות π, אנו ניתקל במצבים בפרופורציות האלו אשר ניתנים למשקול על ידי γ^t בכדי לשמר את ערך התוחלת, ומתקבל:

$$\nabla J(\theta) = \mathbb{E}_\pi \left[\gamma^t \sum_a q_\pi(s,a) \nabla_\pi\, \pi(a|s,\theta) \right]$$

אם נחליף את a עם מרחב הדגימות A_t מעל המדיניות נקבל:

$$\nabla J(\theta) = \mathbb{E}_\pi \left[\gamma^t G_t \frac{\nabla_\pi\, \pi(A_t|S_t,\theta)}{\pi(A_t|S_t,\theta)} \right]$$

כאשר G_t היא התגמול בזמן t, A_t הפעולה בזמן t ו-S_t המצב בזמן t. משוואה זו מתארת כמות שניתן לדגום בכל נקודת זמן עם תוחלת אשר שווה לגראדיינט. נציב את הביטוי בכלל העדכון שהגדרנו קודם, ונקבל שעל הסכמה הכללית לקיים,

$$\theta_{t+1} = \theta_t + \alpha \gamma^t G_t \frac{\nabla_\pi\, \pi(A_t|S_t,\theta)}{\pi(A_t|S_t,\theta)}$$

כעת בהינתן כלל העדכון מהלך האלגוריתם הינו פשוט ביותר. בהינתן אוסף פרמטרים של מדיניות, מאתחלים את הפרמטרים ומבצעים מספר רב של פעמים (עד אשר אין יותר שינויים במדיניות):

- יצירת פרק (Episode) של מצב-פעולה-תגמול באורך T כאשר פועלים על פי המדיניות הנוכחית
- לכל צעד בפרק, מפעילים את כלל העדכון הנ"ל.

2.1.9.3 שיטות שחקן-מבקר

בניגוד לשיטות המבוססת על גראדיינט המדיניויות אשר לומדות קירוב של המדיניות בלבד, שיטות שחקן-מבקר (Actor-Critic Methods) לומדות הן קירוב של המדיניות והן קירוב של פונקציית הערך. המשל "שחקן" בשיטות מסוג זה מתייחס למדיניות הנלמדת בעוד שה"מבקר" מתייחס לפונקציית הערך הנלמדת [1]. המבקר מעדכן פרמטרים של פונקציית הערך והשחקן מעדכן פרמטרים של המדיניות בכיוון המוצע על ידי המבקר בתהליך הקרוי Bootstrapping. באמצעות שיטה זו ניתן להקטין את השונות ולהאיץ את קצב הלמידה.

2.2 למידה חישובית

למידה חישובית (Machine Learning) היא תחום מחקר המשתייך לבינה מלאכותית אשר התפתח מתוך תחומי המחקר של זיהוי דפוסים (Pattern Recognition) ותאוריית הלימוד החישובי (Computational Learning Theory). למידה חישובית חוקרת את הלימוד ויצירה של אלגוריתמים אשר לומדים ומבצעים תחזיות על מידע [20].

2.2.1 אלגוריתם למידה (מודל)

מיטשל [21] מגדיר אלגוריתם למידה באופן הבא:

"נאמר שתוכנית מחשב לומדת מניסיון E ביחס לקטגוריית משימות T ומדד ביצועים P, אם הביצועים שלה במשימות T כפי שנמדדו על ידי P משתפרים עם ניסיון E"

2.2.1.1 ניסיון

בלמידה חישובית אלגוריתם למידה לומד מתוך דוגמאות (Example). דוגמא היא אוסף של מאפיינים (Features) כמותיים נמדדים של אובייקט או אירוע מסוים שאלגוריתם הלמידה אמור לעבד. בדרך כלל דוגמא מאופיינית על ידי ווקטור $x \in \mathbb{R}^n$ כך שכל x_i הוא מאפיין כמותי [2].

ניסיון (Experience) מוגדר כחשיפה של אלגוריתם למידה אל קבוצת נתונים (Dataset) המכיל דוגמאות. מבחינים בין למידה מונחית ולמידה לא-מונחית.

א. למידה מונחית (Supervised Learning) מתייחסת לבעיות בהן האלגוריתם מוזן עם דוגמאות מתויגות. כלומר המידע מוצג יחד עם התוצאה הרצויה של האלגוריתם. התיוג יכול להיות חלקי, מוגבל או בצורה של משוב כתוצאה מביצוע פעולות בסביבה דינאמית, כדוגמת למידה באמצעות חיזוקים [20].

ב. למידה לא-מונחית (Unsupervised Learning) מתייחסת לבעיות בהן המידע המוזן לאלגוריתם איננו מתויג, ועל האלגוריתם ללמוד את המבנה שלו [2].

2.2.1.2 משימה

למידה חישובית מאפשרת להתמודד עם משימות (Task) שקשות מידי לפתרון באמצעות תוכניות קבועות שנכתבות על ידי בני אדם. תהליך הלמידה לכשעצמו אינו המשימה, אלא האמצעי בו ניתן להשיג את היכולת לבצע את המשימה. ישנם הרבה סוגים של משימות, להלן דוגמא לשתיים מהן [20]:

א. משימות סיווג (Classification) – במשימות מסוג זה על האלגוריתם להכריע לאיזה מבין k קטגוריות משתייך קלט מסוים.
ב. משימות רגרסיה (Regression) – במשימות מסוג זה על האלגוריתם לנבא ערך נומרי עבור קלט מסוים.

2.2.1.3 מדד ביצועים

פונקציית הפסד (Loss Function) מעריכה באיזו מידה אלגוריתם הלמידה הולם את המידע עליו הוא לומד. אם אלגוריתם הלמידה מנבא בצורה שגויה אזי פונקציית ההפסד תקבל ערך גבוה, בעוד שאם הניבוי נכון הרי שהיא קבל ערך נמוך. פונקציית ההפסד מחושבת ביחס למדד ביצועים כלשהו, לדוגמא, במשימות סיווג ניתן להשתמש במדד דיוק (Accuracy) אשר מחשב את אחוז הסיווגים הנכונים ביחס לכל הסיווגים. המדידה נעשית על חלק מאוסף המידע אשר המודל לא התאמן עליו, כלומר, אוסף המידע מחולק לשני חלקים; קבוצת אימון וקבוצת מבחן. שגיאת האימון (Training Error) מוגדרת להיות השגיאה של פונקציית ההפסד על קבוצת האימון. שגיאת המבחן (Test Error) מוגדרת כשגיאה על קבוצת המבחן. מטרת אלגוריתם למידה היא, אם כן, למזער ככל שניתן את שגיאת האימון ואת הפער שבין שגיאת האימון לשגיאת המבחן. נאמר שמודל הוא בהתאמת-חסר (Under-fitting) אם אינו מצליח להשיג שגיאת אימון נמוכה, כמו כן נאמר שמודל הוא בהתאמת-יתר (Over-fitting) אם הפער בין שגיאת האימון לשגיאת הבדיקה גדול [22].

2.3 למידה עמוקה

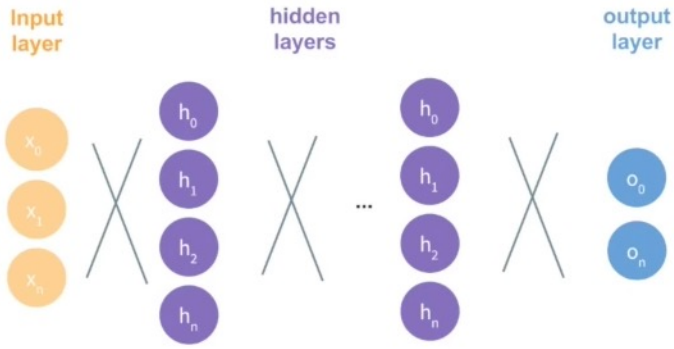
אלגוריתמים רבים של למידה חישובית הם בעלי מבנה רשת רדודה, כלומר הם מורכבים משכבת קלט ומשכבת פלט ואולי שכבה חבויה, כאשר ייתכן והמאפיינים עוברים תהליך המרה ידני (Feature Extraction). אלגוריתמים של למידה עמוקה הם בעלי מבנה רשת עמוקה, כלומר הם מורכבים משכבת קלט, שכבת פלט ושכבות חבויות ביניהם [22].
למידה עמוקה לעיתים היא למידה באמצעות רשת נוירונים מלאכותית עמוקה (Deep Neural Network) וניתן באמצעותה ללמוד ייצוגים של מאפיינים (Features Representations) באופן אוטומטי, וזאת בניגוד לאלגוריתמי למידה קלאסיים בו המאפיינים מחולצים באופן ידני, תוך הכרה עמוקה של עולם הבעיה [2].

2.3.1 רשתות נוירונים עמוקות מסוג התפשטות קדימה

רשתות נוירונים עמוקות מסוג התפשטות קדימה (Deep Feedforward Networks) אשר לעיתים מכונות גם Multi-layer Perceptrons (MLP), נחשבות לאבן היסוד של למידה עמוקה. המטרה של רשת מסוג זה הוא לקרב פונקציה f^*. הרשת מגדירה מיפוי $y = f(x;\theta)$ ולומדת את הערכים של הפרמטרים θ אשר מייצרים את קירוב הפונקציה הטוב ביותר [2].

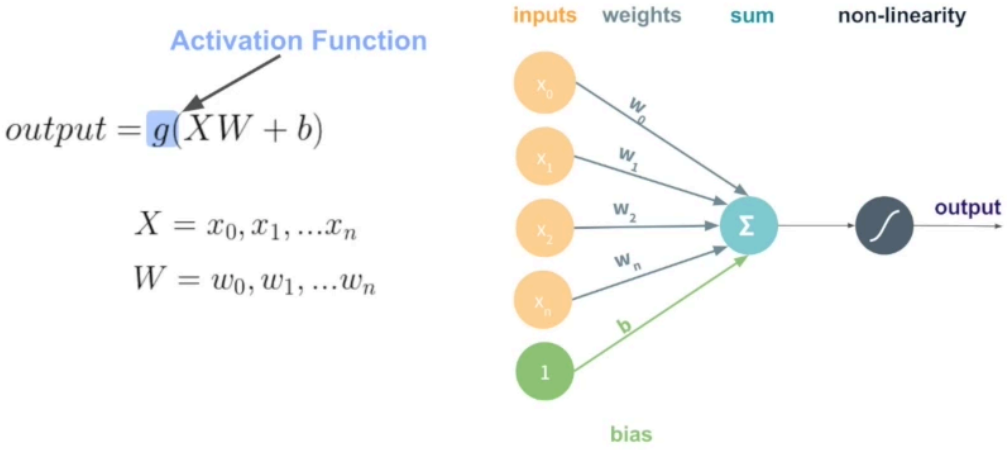
2.3.1.1 מבנה הרשת

ברשתות נוירונים עמוקות מסוג Feedforward המידע זורם משכבת הקלט דרך השכבות החבויות (Hidden Layers) אל שכבת הפלט ללא משוב, כאשר מספר השכבות החבויות קובע את עומק הרשת. איור 1.2 מציג אילוסטרציה של רשת נוירונים עמוקה מסוג Feedforward. ברשת זו ישנה שכבת קלט, שכבת פלט ומספר שכבות חבויות.



איור 1.2 - אילוסטרציה של רשת נוירונים עמוקה עם שכבת קלט, פלט ומספר שכבות חבויות [23]

כל שכבה חבויה בנויה מיחידות חישוב קטנות (Units) אשר מוזנות על ידי הפלט (או חלקו) של השכבה הקודמת. כל יחידה מחשבת מכפלה של הקלט המוזן באוסף משקולות וסכימה שלהם ומסיימת בהפעלת פונקציה לא-לינארית הנקראת פונקציית הפעלה (Activation Function). איור 1.3 מציג דוגמא ליחידת חישוב של שכבה חבויה.



איור 1.3 – דוגמא ליחידת חישוב של שכבת חבויה. ערכי הקלט (Xi) מוכפלים במשקולות (Wi) לאחר מכן מפעילים פונקציה לא לינארית על הסכימה של המכפלות [23]

כאמור פונקציות הפעלה מאופיינות בכך שהן אינן לינאריות ולא בכדי, שכן היות ורוב המידע בעולם הוא לא לינארי, בכדי שנוכל לקרב אותו בצורה טובה עלינו להשתמש בפונקציות שאינן לינאריות. ישנן מספר רב של פונקציות כאלו, המוכרות ביותר הן: Sigmoid, TanH ו ReLU.

2.3.1.2 <u>אימון הרשת (למידה)</u>

בהינתן קלט x לרשת, הרשת מפיקה ערך y* שהוא קירוב לערך $y = f(x)$, במהלך הקרוי התפשטות קדימה (Forward Propagation). בהינתן אוסף קלטים נחשב לכל קלט x את שגיאת הקירוב בין y* ל y ונגדיר את השגיאה הכוללת באופן הבא:

$$J(\theta) = \frac{1}{N}\sum_{i=1}^{N} loss(f(x^i;\theta), y^i))$$

כאשר N הוא מספר הפעמים בהם נחזור על האימון, y^i הוא הערך האמיתי של y באיטרציה i, $f(x^i;\theta)$ הוא הקירוב ל y באיטרציה i ו- $loss(f(x^i;\theta),y^i)$ הוא ערך הפונקציה לחישוב השגיאה בין הערך האמיתי של y לבין הקירוב שלו על ידי הרשת. בעצם, אימון פירושו מציאת θ אשר ממזער ככל הניתן את $J(\theta)$, כאשר θ היא וקטור משקולות של הרשת. ישנם מספר אלגוריתמים למציאת θ שממזער את J, אחד מהם הוא Stochastic Gradient Descent. אלגוריתם זה מתאר תהליך איטרטיבי בו בכל שלב לכל דוגמא באוסף האימון מחשבים את הגראדיינט (Gradient) של J ביחס ל θ ומעדכנים את θ עם הכיוון ההפוך של הגראדיינט כפול קבוע למידה η:

$$\theta := \theta - \eta \frac{\nabla J(\theta)}{\nabla \theta}$$

האלגוריתם מסתיים כאשר הגראדיינט מתכנס ל 0. חישוב הגראדיינט נעשה באמצעות שיטה הקרויה התפשטות אחורה (Back Propagation) והיא בעצם יישום של כלל השרשרת לנגזרות. נשים לב שבכדי לחשב את J במהלך ההתפשטות הקדמית, האלגוריתם עבר על מסלול בגרף של הרשת, בכל שכבה דרך יחידה $u^{(i)}$ כלשהי. נסמן את היחידה האחרונה ב $u^{(n)}$ ואת היחידה הראשונה ב $u^{(j)}$ ונחשב את הגראדיינט על ידי הליכה אחורית על אותו מסלול תוך כדי הפעלת כלל השרשרת לנגזרות, ומתקיים:

$$\frac{\nabla u^{(n)}}{\nabla u^{(j)}} = \sum_i \frac{\nabla u^{(n)}}{\nabla u^{(i)}} \frac{\nabla u^{(i)}}{\nabla u^{(j)}}$$

כלומר, לכל משקולת w ב θ נוכל לחשב את הנגזרת המתאימה לה, ובאמצעות כלל השרשרת נוכל לחשב את $\frac{\nabla J(\theta)}{\nabla \theta}$ [22].

2.3.1.3 <u>מקדם הלמידה η</u>

כלל העדכון של אלגוריתם Stochastic Gradient Descent כרוך בהחסרת הגראדיינט בכיוון ההפוך כפול מקדם למידה קבוע η מ θ. מקדם זה מהווה בעצם את גודל הצעד שהאלגוריתם עושה במרחב בדרך למציאת נקודת המינימום. מחד, ערכו של המקדם η צריך להיות גדול מספיק כדי שהאלגוריתם לא ייתקע במינימום מקומי ומאידך, קטן מספיק כדי שהאלגוריתם לא יחטיא את נקודת המינימום [22].
ישנם אלגוריתמים אשר משנים את ערכו של η באופן דינאמי במה שקרוי Adaptive Learning Rate או יש כאלו שמקצים מספר מקדמים לכל קבוצת פרמטרים מ θ.

2.3.1.4 <u>הסדרה</u>

הסדרה או רגולריזציה (Regularization) מתייחסת לאוסף שיטות אשר מטרתן להקטין את שגיאת המבחן (כפי שהוגדרה בסעיף 1.2.1.3). ישנן שיטות רבות מסוג זה, להלן תיאור של כמה נפוצות [2]:

א. הפלה (Dropout) – מתייחס לשיטה בה בוחרים באופן אקראי יחידות חישוב של הרשת ומסירים אותן ממנה לסיבוב הנוכחי. בסיבוב הבא נבחר יחידות אחרות באותו אופן ונסיר אותן וכך הלאה. באופן הזה נוצר מצב בו הרשת לא מסתמכת באופן בלעדי על קבוצה של יחידות חישוב.

ב. עצירה מוקדמת (Early Stopping) – מתייחס לשיטה בה עוצרים את אימון הרשת ברגע שהשגיאה של אוסף המבחן יורדת מערך שיא אחרון אליו הגיעה במהלך האימון ובלבד שהוא מעל לסף קבוע. כלומר, לא מאפשרים לרשת להתאמן יותר. בכך מונעים מצבים של התאמת-יתר.

ג. קיזוז משקולות (Weights Regularization) - ברוב המקרים משקולות גדולים מעידים על התאמת-יתר של המודל. בשיטה של קיזוז משקולות מוסיפים את גודל המשקולות לפונקציית ההפסד J, אלגוריתם האימון מחד יחפש את המינימום הנדרש אך מאידך ישמור על משקולות קטנים.

2.3.2 רשות נוירונים עמוקות מחזוריות

רשתות נוירונים עמוקות מחזוריות (Recurrent Neural Networks) הן משפחה של רשתות נוירונים המיועדות לטיפול במידע סדרתי. מידע סדרתי מוגדר כסדרה של ערכים $x_1 \dots x_n$ שיש ביניהם קשר כלשהו, לדוגמא משפט בשפה טבעית בו כל מילה היא ערך אחד בסדרה. בשונה מרשתות מסוג Feedforward רשתות Recurrent רגישות סדר האיברים והקשר בין אברים מרוחקים של המידע הסדרתי. רשתות אלו מאפשרות גם שימוש בפרמטרים משותפים לכל אברי הסדרה [22].

רשתות Recurrent מוסיפות את מימד הזמן מעל רשתות Feedforward על ידי שמירת המצב (State) של יחידת חישוב והתחשבות בו כאשר היחידה מחושבת. בכדי להמחיש את פעולת היחידות ברשתות אלו ניתן לחשוב עליהן כעל אוסף יחידות חישוב רגילות על פני הזמן, כך שבכל יחידת זמן משתמשים ביחידת חישוב מתאימה.

2.3.2.1 אימון הרשת

בדומה לרשתות מסוג Feedforward אימון רשת מסוג Recurrent מבקש למצוא את θ אשר תמזער את J ככל שניתן. אולם, בשונה מרשתות מסוג Feedforward, ברשתות מסוג Recurrent הגראדיינט תלוי גם במימד הזמן, כלומר פונקציית ההפסד מחושבת בכל יחידת זמן בנפרד. נגדיר את $J_t(\theta)$ בתור השגיאה ברגע t ביחס ל θ. מכיוון שההפסד הכולל הוא הסכום של כל ההפסדים:

$$J(\theta) = \sum_t J_t(\theta)$$

הרי שהגראדיינט של ההפסד הכולל הוא סכום של הגראדיינטים השונים, ומתקבל:

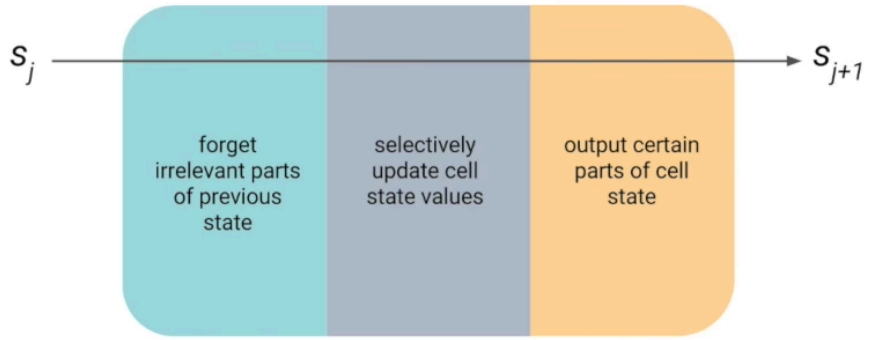
$$\frac{\nabla J(\theta)}{\nabla \theta} = \sum_t \frac{\nabla J_t(\theta)}{\nabla \theta}$$

כמו קודם, גם כאן ניתן להפעיל את כלל השרשרת לנגזרות בכדי לחשב את הגראדיינט, אם כי יש לשים לב שהנגזרות תלויות גם בתוצאות של שלבי האימון הקודמים (כלומר במצבים)של כל יחידת חישוב ולא רק במשקולות, שכן המצבים מקפלים בתוכם את גורם הזמן.

בעיית הגראדיינט הנעלם היא תופעה בה החשיבות של פרט מידע בשלב מוקדם דועכת מהר בשלבים מאוחרים יותר. הבעיה נובעת מהעובדה שחישוב הגראדיינט ביחס למשקולות ולמצבים מצריך הכפלה של מספרים רבים כאשר כמעט כולם קטנים ממש מאחת ומספר המכפילים גדל ככל שבוחנים מידע ישן יותר. תופעה זו מקנה לרשת נטייה להעדיף אירועים מהזמן האחרון לעומת אירועים ישנים יותר [2].

רשתות LSTM מספקות מענה לבעיית הגראדיינט הנעלם, על ידי הוספת לוגיקה בכל יחידת חישוב ברשת. באמצעות הלוגיקה הזו כל יחידת זמן יכולה לסנן מידע לא רלוונטי, לעדכן פרטי מידע ספציפיים לאורך זמן

ולהוציא החוצה רק פרטי מידע רלוונטים. איור 1.4 מציג אילוסטרציה של יחידת חישוב ברשת מסוג LSTM.



איור 1.4 – אילוסטרציה של יחידת חישוב ברשת LSTM. הקלט מעובד באמצעות לוגיקה של היחידה המאפשרת בין השאר להסיר חלקים לא רלוונטיים של הקלט, לבחור אילו פרטי מידע לעדכן במצב של היחידה ולבסוף להחליט אילו פרטי מידע יופנו כפלט [23]

2.3.3 רשתות נוירונים עמוקות קונבולוציוניות

רשתות נוירונים עמוקות קונבולוציוניות (Convolutional Neural Networks) הן רשתות נוירונים אשר מחשבות קונבולוציה (Convolution) על ערכי הקלט שלהן בלפחות אחת מן השכבות שלהן. רשתות מסוג זה הן בעלות יכולת לעבד מידע המאורגן בטופולוגיה של רשת כדוגמת תמונה [2].

2.3.3.1 קונבולוציה

פעולת הקונבולוציה המסומנת בכוכבית מוגדרת עבור פונקציית מקור x ופונקציית גרעין (מסנן) w באופן הבא:

$$s(t) = (x * w)(t) = \int x(a)w(t-a)da$$

ניתן להגדיר פעולת קונבולוציה עבור פונקציות דיסקרטיות באופן הבא:

$$s(t) = (x * w)(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a)$$

כמו כן, ניתן לחשב קונבולוציה על מספר מימדים, לדוגמא קונבולוציה עבור שני מימדים מוגדרת כך:

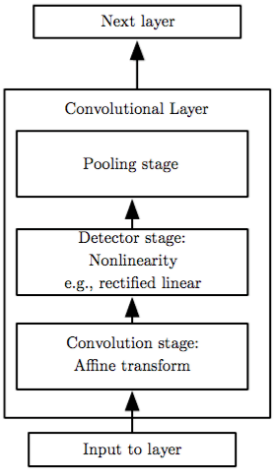
$$s(i,j) = (I * W)(i,j) = \sum_{m}\sum_{n} I(m,n)W(i-m,j-n)$$

2.3.3.2 מבנה הרשת

רשתות נוירונים מסוג Convolutional מכילות לפחות שכבת קונבולוציה אחת. שכבת קונבולוציה מורכבת משלושה שלבים, קרי, קונבולוציה, הפעלה לא לינארית ו Pooling.

בשלב **הקונבולוציה** מחשבים את הקונבולוציה בין הקלט לבין אוסף מסננים, כאשר ערכי המסננים הם פרמטרים השייכים ל θ והרשת משנה את ערכם כחלק מתהליך הלמידה. בשלב **ההפעלה הלא לינארית** מפעילים פונקציית הפעלה לא לינארית כמו ברשתות נוירונים רגילות כדוגמת ReLU. שלב ה **Pooling משמש להקטנת מספר המימדים של הקלט באמצעות** דגימה מקטינה (Down-sampling) כלשהי של

התוצאה מהשכבה הקודמת. ישנן דרכים רבות לבצע זאת, דרך אחת לעשות זאת היא Max Pooling. על פי שיטה זו מחלקים את הקלט לאזורים בגודל קבוע, ומכל אזור בוחרים את הערך המקסימלי. איור 1.5 מציג אילוסטרציה של השכבות השונות ברשת נוירונים קונבולוציונית.



איור 1.5 – אילוסטרציה של השלבים השונים של שכבת הקונבולוציה. הקלט לשכבה עובר טרנספורמציה אפינית, לאחר מכן הוא עובר דרך יחידה לא לינארית ומסיים ב- pooling בכדי להוריד את גודל המימד שלו [2]

3. למידה עמוקה באמצעות חיזוקים

בפרק 1 סקרנו שיטות למידה באמצעות חיזוקים השייכות למשפחת הפתרונות הטבלאיות. שיטות אלו מוצאות את הפתרון האופטימאלי על ידי מעבר על כל המצבים והפעולות. אי לכך כמות הזיכרון והזמן הנדרשים נמצאים ביחס ישיר לגודל מרחב מצב-פעולה של הבעיה, והרי, בעיות רבות מאופיינות במרחבי מצב-פעולה עצומים והשימוש בשיטות אלו אינו ישים. יחד עם זאת, ראינו כיצד ניתן להשתמש ברשתות נוירונים עמוקות כדי לקרב פונקציות עם מרחבים מסדר גודל זה. למעשה, למידה עמוקה באמצעות חיזוקים (Deep Reinforcement Learning) היא יישום של השיטות הקלאסיות של למידה באמצעות חיזוקים יחד עם קירוב פונקציות על ידי למידה עמוקה. באופן כללי, למידה עמוקה באמצעות חיזוקים מבוססת על אימון רשתות נוירונים עמוקות כדי לקרב את **המדיניות** האופטימלית ו/או **פונקציית הערך** האופטימלית.

בטרם נסקור שיטות עכשוויות ליישום רשתות נוירונים עמוקות ביחד עם אלגוריתמים של למידה באמצעות חיזוקים, ראוי להזכיר את אחת ההצלחות הראשונות בתחום- האלגוריתם TD-Gammon [24] מאמצע שנות התשעים. האלגוריתם מתאר רשת נוירונים בשילוב שיטות הפרש טמפורלי (Temporal Difference) אשר הגיעה לרמת משחק של אדם במשחק שש-בש. זהו למעשה האלגוריתם הראשון אשר הצליח לשלב שיטות למידה באמצעות חיזוקים יחד עם רשתות נוירונים עמוקות כדי להשיג תוצאות משחק ברמה של אדם.

3.1 רשתות נוירונים עמוקות כקירוב לפונקציית ערך

כפי שנידון בפרק הקודם, פונקציות ערך הן מושג יסוד בתיאוריה של למידה באמצעות חיזוקים, בפרט בשיטות הפרש טמפורלי ו-Q-Learning. בתת פרק זה נדון בשאלה כיצד ניתן לקרב פונקציות ערך באמצעות רשתות נוירונים עמוקות. נתחיל בדיון על אלגוריתם הראשון שעשה שימוש ברשת נוירונים בכדי לקרב פונקציות ערך בשם Neural Fitted Q-Iteration [25] ונתמקד באלגוריתם פורץ הדרך DQN [7] ובמספר הרחבות שונות שלו Double DQN [26], Prioritized Experience Replay [27], Dueling Architecture [9], Averaged DQN [28] ו Rainbow [29].

3.1.1 קירוב ערכי Q באמצעות רשת נוירונים עמוקה

האלגוריתם אשר עשה שימוש לראשונה ברשת נוירונים כדי לקרב פונקציית ערך הינו Neural Fitted Q-Iteration [25]. האלגוריתם, המכונה בקיצור NFQ, מבוסס על האלגוריתם הקלאסי של Q-Learning אך בשונה ממנו ערכי Q מתקבלים על ידי הפעלה של רשת נוירונים לכל זוג מצב-פעולה נתון. גישה זו מגלמת בתוכה את הרעיון שעל ידי קירוב ערכי Q ניתן לתפוס בצורה מדויקת יותר את מבנה הסביבה. אמנם השימוש ברשת נוירונים פותר את הצורך בתחזוקה של טבלת ערכי Q, אך מציב בעיה אחרת הנוגעת לזמן האימון הנדרש בכדי להביא הרשת לכדי התכנסות. כאמור בהינתן עדכון לרשת, אלגוריתם הפעפוע האחורי (Back-propagation) מעדכן משקולות בכל המרחב, כלומר עדכון מקומי עלול להשפיע על אזורים אחרים שחושבו כבר ובכך "להרוס" למידה שנעשתה עד כה או אף להוביל להתבדרות הרשת [2].

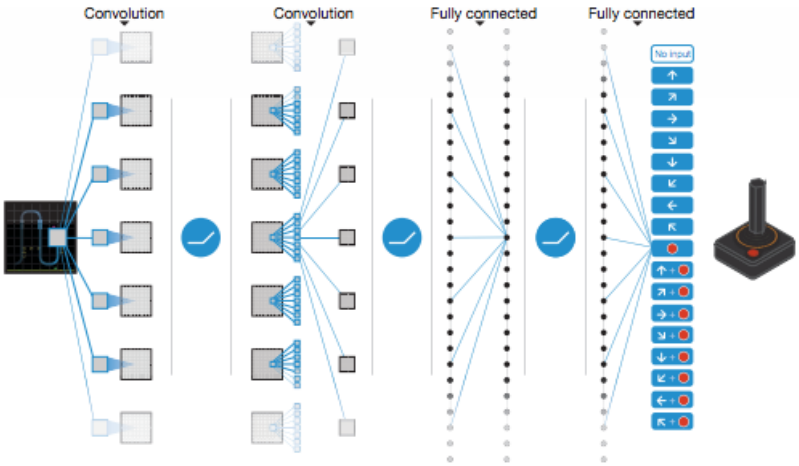
כדי לפתור בעיה זו, NFQ מציע לשלב את הטכניקה של Experience Replay [30] אשר בה חיזרור הערכים מבוצע על כל המעברים שנראו עד כה. שיטה זו מבטיחה שכל עדכון חדש לרשת יבוצע יחד עם עדכונים קודמים ובכך ישמור את מה שנלמד עד כה, ובמקרים רבים יאיץ את ההתכנסות של הרשת [25].

האלגוריתם NFQ הורחב [31] כדי לאמן רשת נוירונים להסיע רכב על מסילה עם קלט של תמונות לא מעובדות ממצלמה שהונחה מול המסלול. זאת בניגוד לאלגוריתם המקורי אשר קיבל כקלט זוגות של מצב-פעולה, מה שדרש הכרות עם מרחב הבעיה הספציפית, על כן, NFQ הוא למעשה האלגוריתם הראשון בתחום אשר פועל באופן זה.

3.1.2 רשתות Q עמוקות

האלגוריתם רשתות Q עמוקות (Deep Q-Network) [7] היווה את פריצת הדרך הגדולה ביותר בתחום של למידה עמוקה באמצעות חזקים בשנים האחרונות. מודל האלגוריתם מזכיר מאוד את המודל שהוצג ב-NFQ [25] במובן זה שהקלט הינו חזותי (לדוגמה: ייצוג באמצעות פיקסלים) ובכך שאין דרישה להכרות עם מרחב הבעיה. עם זאת, DQN שאומן על סביבה מסוימת יכול לפעול על מגוון רחב של סביבות אחרות, וכדוגמה יישמו החוקרים בהצלחה את DQN על קונסולת המשחקים Atari 2600 עם 49 משחקים, מתוכם 43 משחקים שוחקו ברמת אדם, זאת להבדיל מ-NFQ בו הרשת המאומנת התאימה לסביבה ספציפית עליה היא אומנה ולא ניתן להפעיל אותה על סביבות אחרות.

בדומה ל-NFQ החוקרים משתמשים ברשת נוירונים על מנת לקרב ערכי Q. רשת הנוירונים עוצבה בצורה כזאת כך שהשכבה המחוברת באופן מלא (Fully connected layer) האחרונה של הפלטים היא אוסף פעולות בדיד המותר במשחקי אטארי, לדוגמה הפעולות למעלה, למטה אחורה קדימה וכד'. החוקרים הראו שמבנה זה מאפשר לקודד בצורה פשוטה יותר ידע בשכבות קונבולוציה נמוכות (Lower convolutional layers), ואכן DQN מצליח לחלץ מאפיינים בולטים בסצנה, את תנועתם בה ואת האינטראקציות ביניהם [7].



איור 2.1 – אילוסטרציה של רשת נוירונים קונבולוציונית כפי שעיצבו החוקרים של DQN. הרשת כוללת שתי שכבות קונבולוציה ושכבה אחת מחוברת באופן מלא שלאחריה יש שכבת pooling. [7]

החוזקה הגדולה ביותר של DQN היא היכולת להתמודד עם מרחבי קלט עצומים ולייצג את ערכי Q שלהם בצורה קומפקטית באמצעות רשתות נוירונים. לדוגמה קונסולת המשחקים Atari 2600 מורכבת מג'ויסטיק שמייצר 18 פעולות אפשריות ומסך בגודל 160X210 פיקסלים עם עומק צבע 8 ביט RGB. לפיכך מרחב

הקלט ל-DQN הוא בגודל $|\mathcal{S}| \times |\mathcal{A}| = 18 * (2^8)^{3*210*160}$. ברור כי פתרון בעיה עם מרחב קלט בגודל כזה בשימוש עם אלגוריתם Q-Learning קלאסי אינו אפשרי מפאת גודל הזיכרון הדרוש לתחזוקה של טבלת ערכי Q.

בכדי להתמודד עם בעיית התכנסות רשת הנוירונים שהוזכרה קודם, ובדומה ל-NFQ, DQN עושה שימוש בטכניקת Experience Reply [30] כפי שהוסבר בסעיף הקודם. יתרה מזאת, DQN מוסיף טכניקה נוספת להתמודדות עם הבעיה ומשתמש בטכניקת Target Network [7]. באמצעות טכניקה זו מתחזקים שתי רשתות נוירונים: אחת שמתעדכנת כל העת ואחת שמתעדכנת כל מספר צעדים קבוע (להלן הרשת הקבועה). חישוב ערכי Q נעשה מתוך הרשת הקבועה ובכך מקטין את הרגישות לשינויים התכופים מהרשת שמתעדכנת כל העת. איור 2.1 מציג אילוסטרציה של רשת הנוירונים הקונבולוציונית שהחוקרים עיצבו. הרשת כוללת שתי שכבות קונבולוציה ושכבה אחת מחוברת באופן מלא שלאחריה יש שכבת pooling.

לסיכום, תוצאות המחקר הראו ש-DQN מסוגל לפתור בעיות עם מרחב קלט עצום ועם ידע מינימאלי על מרחב הבעיה באופן שעולה בביצועיו על פני כל אלגוריתם למידה עמוקה באמצעות חיזוקים ידוע אחר עד לאותה העת.

3.1.3 <u>רשתות Q עמוקות כפולות</u>

אלגוריתם רשתות Q עמוקות כפולות (Double DQN) [26] הוא הרחבה לאלגוריתם DQN שהוזכר לעיל. מטרת אלגוריתם זה היא להתגבר על בעיית הערכת יתר (Over-estimation) של אלגוריתם Q-Learning הקלאסי אשר קיימת גם באלגוריתם DQN. כפי שהובא בפרק הקודם (ר' סעיף 1.1.5), כלל העדכון של אלגוריתם Q-Learning כולל בתוכו אופרטור מקסימום אשר משמש לבחירת הפעולה ולהערכתה גם יחד. מכאן, סביר שהאלגוריתם יבחר ערכים שהינם יתר על המידה, דבר אשר יוביל להערכות אופטימיות יתר על המידה [32]. יש לציין כי אם הערכות היתר נעשות בצורה אחידה על פני כלל הדגימות אין בעיה של ממש, אך כפי שהחוקרים הראו [26] זה אינו המצב עבור DQN ואכן ישנם משחקים שבהם הביצועים של DQN אינם טובים.

החוקרים [26] מציעים לנתק את הבחירה של הפעולות מהערכה של ערכי Q, וזאת על ידי חישוב שתי פונקציות ערך: אחת מחושבת ביחס לרשת שמתעדכנת כל העת, והשנייה מחושבת ביחס לרשת המטרה (Target network) כפי שהוסבר בסעיף הקודם. השינוי המוצע כולל שינוי זעום באלגוריתם DQN המקורי ואכן ביצועי האלגוריתם עבור אותו אוסף משחקי Atari ש-DQN נבחן עליהם הניבו תוצאות טובות בהרבה.

3.1.4 <u>שימוש מתועדף בניסיון קודם</u>

האלגוריתמים שנסקרו עד כה שילבו בתוכם את הטכניקה של Experience Replay [30] כדי להתמודד עם בעיית ההתכנסות של רשת הנוירונים. כאמור, אלגוריתם Experience Replay שומר את כל המעברים שנעשו בזיכרון מיועד ובכך מאפשר שימוש חוזר באותם ערכים. כתוצאה מטכניקה זו ניתן להשתמש בערכים נדירים כשכיחים מה שתורם להפחתת שגיאת ההערכה [30].

האלגוריתם המקורי בוחר תת קבוצה באופן אחיד מתוך הזיכרון המיועד. אלגוריתם שימוש מתועדף בניסיון קודם (Prioritized Experience Replay) [27] מציע לבחור את תת הקבוצה לפי עדיפויות המחושבות לכל

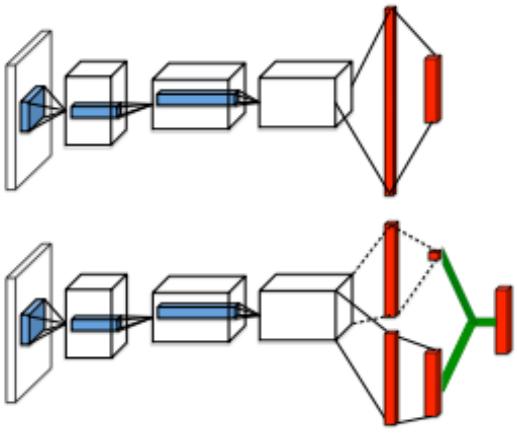
מעבר שנשמר בזיכרון המיועד. החוקרים מציעים אלגוריתם סטוכסטי כדי לבחור את תת הקבוצה. באלגוריתם יבחר בצורה אקראית בין בחירה אחידה לבין העדפה של מעברים בהם השגיאה הטמפורלית הייתה גדולה מאוד. טכניקה זו טובה יותר מאשר בחירה על פי גודל השגיאה הטמפורלית בלבד ממספר סיבות: ראשית, הדבר עלול לגרום לאי שימוש למעברים עם שגיאה נמוכה; שנית, שגיאה טמפורלית רגישה לרעשים חדים. החוקרים יישמו את האלגוריתם המוצע כחלק מ-DQN ו-Double DQN והראו ביצועים טובים יותר על אותו אוסף משחקי Atari שהאלגוריתמים נבחנו על פיהם.

3.1.5 ארכיטקטורות יריבות

מרבית האלגוריתמים בתחום של למידה עמוקה באמצעות חיזוקים עושים שימוש בארכיטקטורות קונבנציונאליות כדוגמת רשתות קונבולוציה, LSTMs וכד'. החוקרים [9] מציעים ארכיטקטורת רשת נוירונים חדשה בשם ארכיטקטורת יריבות (Dueling Architecture) אשר יתרונה על פני הרשתות האחרות הוא בכך שהיא מצליחה להכליל את תהליך הלמידה טוב יותר מארכיטקטורות אחרות.

הרעיון בבסיס ארכיטקטורת Dueling הוא חישוב של שתי פונקציות ערך Value ו-Advantage. פונקציית Value זהה לפונקציית הערך המחושבת בכל האלגוריתמים הקודמים. כלומר, מטרתה להעריך את ההשפעה של פעולה בכל מצב. מאידך, מטרת פונקציית ה-Advantage היא לזהות אילו מצבים הינם בעלי ערך ואילו אינם. שתי פונקציות אלו מסתכמות לפונקציה אחת המשמשת לקירוב ערכי Q. טכניקה זו היא בעלת שני יתרונות מובהקים: האחד הוא התכנסות מהירה של רשת הנוירונים; והשני הוא שניתן להשתמש בארכיטקטורה זו בכל האלגוריתמים הקודמים שנסקרו היות והתוצר הסופי של הרשת זהה לרשתות הקונבנציונאליות.

חישוב שתי הפונקציות נעשה על ידי פיצול השכבה המקושרת באופן מלא (Fully connected layer) האחרונה של DQN לשני זרמים (Streams) של שכבות מקושרות באופן מלא. כל זרם נבנה באופן כזה שהוא יכול לספק הערכות לחישוב שתי הפונקציות. שני הזרמים מחוברים יחד כדי ליצור פלט ערכי Q אחד. איור 2.2 מציג אילוסטרציה של פיצול השכבה המחוברת המלאה האחרונה לשני זרמים בכדי לחשב את שתי פונקציות הערך. החוקרים יישמו את הארכיטקטורה החדשה יחד עם אלגוריתם DQN, Double DQN בשיתוף עם אלגוריתם Prioritized Experience Replay . כל השילובים הנ"ל הראו ביצועים טובים יותר מאשר שימור בארכיטקטורות הקונבנציונאליות, כאשר נבחנו על משחקי Atari.



איור 2.2- אילוסטרציה של פיצול השכבה המחוברת המלאה האחרונה לשני זרמים בכדי לחשב את שתי פונקציות הערך. בחלק העליון של האיור מוצגת הארכיטקטורה של הקלאסית של DQN ובחלק התחתון מוצגת הארכיטקטורה המוצעת על ידי החוקרים [9]

3.1.6 רשתות Q עמוקות ממוצעות

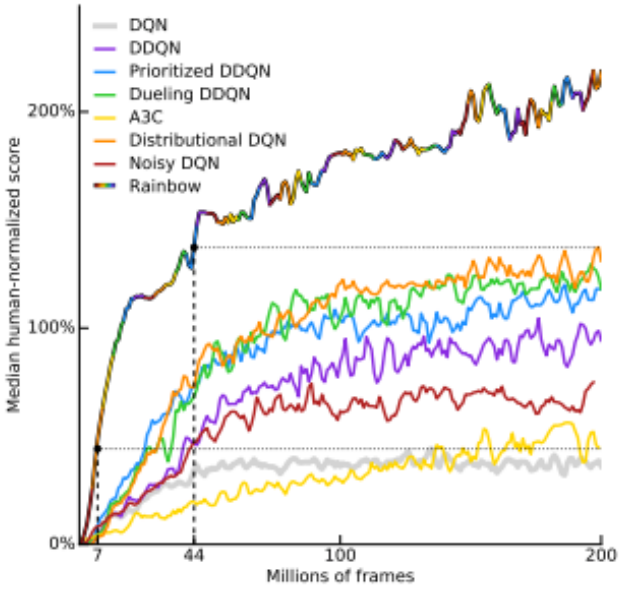
אלגוריתם רשתות Q עמוקות ממוצעות (Averaged DQN) [28] הוא הרחבה נוספת לאלגוריתם DQN. בדומה לאלגוריתם Double DQN גם אלגוריתם Averaged DQN מתייחס לבעיית הערכת היתר של DQN ומציע דרך אחרת להתמודדות עמה, קרי, מיצוע K (קבוע כלשהו) ערכי Q אחרונים שנלמדו. מיצוע זה מפחית את שונות השגיאה אשר מוגדרת כשגיאה בין ערכי Q המקורבים (על ידי רשת הנוירונים) לערכי Q האופטימאליים. הפחתת שונות השגיאה מסייעת ליציבות הרשת ולקבלת תוצאות טובות יותר.

מבחינה חישובית Averaged DQN דורש K מעברים קדמיים דרך ה-Q-Network. מספר עדכוני ההילוכים האחוריים (Back-Propagation update) נשאר זהה לזה של DQN. החוקרים מדווחים שהשינוי הפשוט המוצע משפר את הביצועים של DQN ו-Double DQN כאשר בוחנים אותו מעל משחקי Atari 2600, וניכר כי השיפור משמעותי עוד יותר ככל ש-K גדל.

3.1.7 איחוד אלגוריתמים המבוססים על אלגוריתם רשתות Q עמוקות לכדי אלגוריתם אחד: קשת-בענן

מאז פרסום אלגוריתם DQN חוקרים רבים חיפשו דרכים שונות להרחיב אותו ולשפר את ביצועיו. דוגמאות לכך הם אלגוריתמים Double DQN, Prioritized Experience Replay ו-Dueling Architecture אשר נסקרו בסעיפים הקודמים. כל אלגוריתם מציע שיפור משמעותי כשהוא נבחן למול אלגוריתם DQN המקורי. יתרה מזאת, כל אלגוריתם מתייחס לבעיות שונות באלגוריתם המקורי.

אלגוריתם קשת-בענן (Rainbow) [29] הינו תוצאה של מחקר אמפירי אודות ביצועי אלגוריתם המשלב שש הרחבות שונות שפורסמו לאלגוריתם DQN המקורי. הרחבות אלו כוללות את האלגוריתמים הבאים: DDQN, Prioritized DDQN, Dueling DDQN, A3C, Distributional DQN ו- Noisy DQN. כפי שניתן לראות מאיור 2.3, החוקרים מצאו כי האלגוריתם המשולב משיג תוצאות טובות יותר מכל הרחבה כאשר היא נבחנת בנפרד.



איור 2.3- השוואת הביצועים של האלגוריתם המשולב (Rainbow) כנגד כל אחת מההרחבות DQN בנפרד. הגרף באיור מודד את כמות המסכים (של משחק אטארי) שנדרשו לאימון האלגוריתם ביחס לביצועיו. ברור כי האלגוריתם המשולב משיג תוצאות טובות יותר כמעט מההתחלה ועם כמות פחותה משמעותית של מסכי משחק [29]

3.2 רשתות נוירונים עמוקות כקירוב למדיניות

כפי שהוסבר בפרק הקודם מדיניות הינה מיפוי בין מצב לפעולה, ואופטימיזציה של מדיניות (Policy optimization) הידועה גם בשם חיפוש מדיניות (Policy search) היא התהליך של מציאת המיפוי האופטימאלי. במסגרת האלגוריתמים המבוססים תכנון דינאמי, הצגנו בפרק הקודם את אלגוריתם חזרור המדיניות אשר מוצא מדיניות אופטימאלית על ידי שימוש בפונקציית ערך. בסעיף זה נעסוק בשיטות הלומדות מדיניות התלויה בפרמטרים (Parameterized policy) אשר בוחרות פעולות מבלי להשתמש בפונקציית ערך, קרי שיטות גראדיינט המדיניות (Policy Gradient Methods), וכן בשיטות המשתמשות בפונקציית ערך אך לא לבחירת הפעולה, קרי שיטות שחקן-מבקר (Actor-Critic Methods).

3.2.1 שיטות גראדיינט המדיניות ורשתות נוירונים עמוקות

המחקר בשנים האחרונות עסק בשיפור האלגוריתמים הבסיסיים המבוססים על שיטות גראדיינט המדיניות תוך כדי שימוש ברשתות נוירונים עמוקות כקירוב למדיניות. בחלק זה נדון בשני מחקרים פורצי דרך מהשנים האחרונות:

3.2.1.1 גראדיינט מדיניות דטרמיניסטי

לרוב שיטות המבוססות על גראדיינט המדיניות מייצרות מדיניות סטוכסטית (מדיניות אשר בוחרת פעולה a במצב s באופן סטוכסטי על פי וקטור פרמטרים θ). כפי שראינו השיטה הנפוצה ביותר שיטות מסוג זה היא דגימה של המדיניות הסטוכסטית והתאמה של משתני המדיניות כך שהתגמול המצטבר גדל. גראדיינט מדיניות דטרמיניסטי (Deterministic Policy Gradient) [10] פועל באותו האופן, רק עם מדיניות דטרמיניסטית במקום מדיניות סטוכסטית. החוקרים מראים שגראדיינט מדיניות דטרמיניסטי אכן קיים

ומציעים דרך יעילה לחשבו. שכן, ישנו הבדל מהותי בין השניים; במקרה הסטוכסטי גראדיינט המדיניות מחושב מעל מרחב המצבים והפעולות יחד בעוד שבמקרה הדטרמיניסטי הוא מחושב מעל מרחב המצבים בלבד. כתוצאה מכך חישוב גראדיינט המדיניות הסטוכסטי עלול לדרוש יותר דוגמאות, בפרט כאשר מרחב המצבים הוא בעל מספר עצום של מימדים.

החוקרים [10] מציעים אלגוריתם off-policy מסוג שחקן-מבקר אשר בוחר פעולות על פי מדיניות התנהגות סטוכסטית (על מנת להבטיח חקר הולם של המרחב) בכדי ללמוד מדיניות דטרמיניסטית (ניצול של היעילות בחישוב גראדיינט מדיניות דטרמיניסטי). תוצאות המחקר מראות ששיטה זו טובה יותר משיטות המבוססות על גראדיינט מדיניות סטוכסטי, בפרט במשימות בעלות מימד גדול.

אלגוריתם גראדיינט מדיניות דטרמיניסטי עמוק (Deep Deterministic Policy Gradient) [33] משלב את הטכניקה של אלגוריתם גראדיינט מדיניות דטרמיניסטי יחד עם אלמנטים של אלגוריתם DQN. שילוב זה נותן מענה לבעיות בהן מרחב הפעולות הינו רציף (אלגוריתם DQN הוכח כיעיל מעל מרחבי מצב בדידים קטנים).

החוקרים [33] מציעים אלגוריתם Off Policy מסוג שחקן-מבקר אשר עושה שימוש ברשתות נוירונים עמוקות אשר יכולות ללמוד מדיניות ממימד גדול במרחב מצבים רציף. באמצעות שימוש בטכניקת שחקן-מבקר, האלגוריתם נמנע מאופטימיזציה של הפעולה בכל שלב כדי להשיג מדיניות חמדנית כפי שנעשה ב-Q-Learning (מה שאינו מעשי במרחב מצבים גדול). בכדי להפוך את תהליך הלמידה ליציב, איתן ודומה ל-DQN האלגוריתם עושה שימוש בטכניקות של Experience Replay ו-Target Network. תוצאות המחקר הראו שהאלגוריתם המוצע פותר בעיות עם אופי פיזיקאלי מדרגות קושי שונות ועם פי 20 פחות צעדים מאשר אלגוריתם DQN.

3.2.1.2 <u>מיטוב מדיניות באמצעות אזורים בטוחים</u>

אלגוריתם מיטוב מדיניות באמצעות אזורים בטוחים (Trust Region Policy Optimization) (או TRPO) [34] הוא אלגוריתם פרקטי לאופטימיזציה של מדיניות תוך הבטחה לשיפור מונוטוני בכל שלב. האלגוריתם נשען על שני רעיונות עיקריים: שימוש באלגוריתם מסוג Minorization-Maximization ואלגוריתם מסוג Trust Region.

הרעיון העיקרי באלגוריתמים מסוג Minorization-Maximization הוא אופטימיזציה של פונקציית מטרה תוך הישענות על פונקציית מטרה חלופית. כלומר, במקרה של בעיית מקסימום ראשית מוצאים חסם תחתון מקורב למטרה המקורית (מטרה חלופית) ולאחר מכן ממקסמים את החסם התחתון המקורב בכדי לשפר את המטרה המקורית. ב-TRPO החוקרים הציעו להשתמש בפונקציית מטרה חלופית אשר מהווה חסם תחתון לתוחלת התגמול המצטבר של המדיניות.

באלגוריתמים מסוג Trust Region מגדירים אזור מסביב לאיטראציה הנוכחית (בתהליך האופטימיזציה) כך שמאמינים שהמודל מהווה ייצוג הולם של פונקציית המטרה והצעד לשיפור נעשה בתוך אותו אזור. כלומר, בתהליך האופטימיזציה אחרי שחושב הכיוון של הגראדיינט אורך הצד באותו הכיוון מוגבל על ידי אותו אזור. החוקרים השתמשו בהתפצלות Kullback-Leibler בין המדיניות הישנה לבין המדיניות המעודכנת בכדי לחשב את אזור האמון. השימוש באזורי אמון מבטל את החשש שצעד באורך לא נכון בכיוון הגראדיינט עלול להסיט את הקירוב של המדיניות.

החוקרים מראים שהאלגוריתם מסוגל להתמודד עם חישוב של מדיניות לא לינארית עם עשרות אלפי פרמטרים (כמו רשת נוירונים עמוקה) בצורה יעילה. החוקרים ערכו מספר ניסויים והראו שהאלגוריתם מסוגל ללמוד מדיניות מורכבת עבור פעולות כמו שחייה, קפיצה והליכה וכן משחק אטארי בדומה לניסויים שערכו ב-DQN.

3.2.2 שיטות שחקן-מבקר ורשתות נוירונים עמוקות

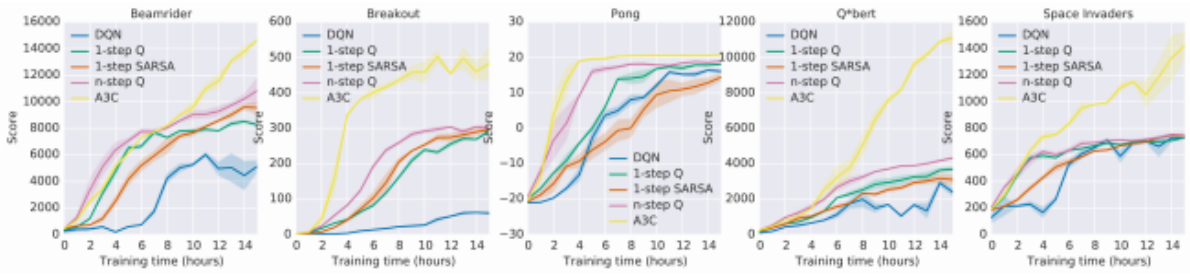
בשנים האחרונות שיטות שחקן-מבקר זוכות להתייחסות גדולה במחקרים בתחום של למידה עמוקה באמצעות חיזוקים, הודות ליכולת שלהם להקטין את השונות ולהאיץ את תהליך הלמידה. להלן שני מחקרים מהשנים האחרונות המציעים אלגוריתמים מסוג זה:

3.2.2.1 שחקן-מבקר אסינכרוני מתקדם

אלגוריתם שחקן-מבקר אסינכרוני מתקדם (Asynchronous Advantage Actor-Critic) (או A3C) [11] הוא אלגוריתם מסוג שחקן-מבקר של למידה באמצעות חיזוקים הפועל בצורה מקבילית א-סינכרונית עם n סוכנים על מספר העתקים של הסביבה. באמצעות מקבילויות זו בכל נקודת זמן הסוכנים חווים מגוון רחב של מצבים שונים ובאמצעות כך ניתן לכסות חלקים גדולים יותר של מרחב המצבים.

יתרה מכך, מעצם הפעלת n סוכנים במקביל, החוקרים הראו שניתן להריצו על מעבד מרובה ליבות סטנדרטי (Multi core CPU) ולא באמצעות יחידת עיבוד גרפית (GPU) כנהוג באלגוריתמים של למידה עמוקה באמצעות חיזוקים אחרים, וכל זה בזמן ריצה הקטן בחצי מהם.

במהלך ריצתו, האלגוריתם A3C מתחזק מדיניות והערכה של פונקציית הערך (על פי הטכניקה של שחקן-מבקר) אשר מתעדכנים על ידי שימוש בתוצאות שחוזרות מ-n הסוכנים שרצים כל מספר צעדים קבוע. החוקרים יישמו את האלגוריתם עבור מבחר רחב של בעיות שליטה מנועית רציפה (כדוגמת הזזת זרוע מכנית בצורה רציפה), בבעיות ניווט במבוכים תלת מימדיים תוך שימוש בקלט חזותי וכן עבור 57 משחקי אטארי (כפי שנעשה ב DQN). איור 2.4 מציג השוואה בין ביצועי A3C לבין אלו של אלגוריתם DQN עבור חמישה משחקי אטארי נבחרים. בכל אחד מהמשחקים שנבחנו האלגוריתם הפיק מדיניות טובה יותר וביצועיו מבחינת זמן אימון היו טובים יותר מכל אלגוריתם מבוסס DQN אחר.



איור 2.4- השוואה בין ביצועיו של אלגוריתם A3C לבין אלו של אלגוריתם DQN עבור חמישה משחקי אטארי. כל הגרפים המוצגים באיור מראים בבירור כי אלגוריתם A3C הוא בעל ביצועים טובים יותר [11]

אלגוריתם A3C הורחב ב [35] כדי ליצור גרסה היברידית המשלבת חישוב באמצעות מעבד מרובה ליבות (Multi-Core CPU) ויחידת עיבוד גרפית (GPU) אשר מכונה Hybrid CPU/GPU A3C (או GA3C).

החוקרים יישמו את A3C באמצעות מעבד מרובה ליבות ובאמצעות יחידת עיבוד גרפית וחקרו כיצד ניתן לכוון את המערכת שמריצה את האלגוריתם כך שהביצועים יגברו. באמצעות הידע שנאסף החוקרים מראים כיצד ניתן לתכנן מערכת אשר מתכוננת באופן אוטומטי בזמן האימון. ואכן, תוצאות המחקר מראות שזמן אימון הרשת עבור משחקי אטארי מהיר יותר מהגרסה המקורית של A3C.

3.2.2.2 שחקן-מבקר עם שימוש חוזר בניסיון

בכל המחקרים והאלגוריתמים שהוצגו עד כה יש צורך לבצע דגימות מעל הסביבה של הבעיה. ככל שהסביבות מורכבות יותר כך אנו נדרשים לבצע יותר דגימות בהן. דגימות אלו נאספות על ידי הדמיות של הסביבות השונות, כלומר עבור כל דגימה אנו נדרשים לבצע שלב יקר של הדמייה. על כן, בכדי להקטין את העלות של ההדמיות עלינו לייצר אלגוריתמים יעילים בכמות הדגימות שהן דורשות. אלגוריתם שחקן-מבקר עם שימוש בניסיון (Actor-Critic with Experience Replay) (או ACER) [36] הוא אלגוריתם מסוג שחקן מבקר אשר מצטיין בהקטנת מספר הדגימות הנדרשות ואשר מתאים לפתרון בעיות עם מרחב מצבים בדיד או רציף.

בדומה לאלגוריתמים אחרים של למידה עמוקה באמצעות חיזוקים, ACER משתמש ברשת נוירונים עמוקה בכדי לקרב את המדיניות ואת פונקציית הערך. האלגוריתם משלב שלוש טכניקות עיקריות:

- Experience Replay – בדומה לשימוש שהוצג ב- [25] ו- [7] גם כאן החוקרים דוגמים מתוך זיכרון מצבים ייעודי בכדי להקטין את השגיאה.
- Stochastic Dueling Network – זוהי הרחבה של הרעיון שתואר ב- [9], אשר מציע ארכיטקטורת רשת נוירונים אשר מחשבת שתי פונקציות ומאחדת את תוצאותיהם. החוקרים מציעים להשתמש בארכיטקטורה זו בכדי לחשב מדיניות סטוכסטית מקורבת ופונקציית ערך מקורבת.
- Trust Region Policy Optimization – על פי העיקרון שהוצג ב- [34], מחשבים אזור אמון, כך שאורך הצעד בכיוון הגראדיינט בכל שלב, נשאר בתוך האזור. בשונה מ- [34] בו הוצע לחשב את אזור האמון באמצעות התפצלות Kullback-Leibler בין המדיניות הישנה לבין המדיניות המעודכנת, ACER מציע לחשב את אזור האמון על ידי המדיניות הממוצעת (ממוצע נע של מדיניות העבר).

החוקרים בחנו את יעילות האלגוריתם על פני 57 משחקי אטארי אשר בכולם מספר הדגימות שנדרש היה קטן משמעותית מזה שנדרש על ידי אלגוריתמים מבוססי DQN אחרים.

4. יישומים עכשוויים של למידה עמוקה באמצעות חיזוקים

בפרק הקודם דנו במגוון אלגוריתמים של המחקר העכשווי בתחום של למידה עמוקה באמצעות חיזוקים. בכל האלגוריתמים שהוצגו, העיקרון המנחה היה שילוב של שיטות למידה באמצעות חיזוקים עם למידה עמוקה במה שקרוי למידה עמוקה באמצעות חיזוקים. שילוב זה מאפשר להתמודד עם בעיות החלטה בעלות מרחבי מצב-פעולה עצומים אשר נחשבו בעבר לקשות לפתרון בזמן סביר. עובדה זו יחד עם המשך המגמה של ירידת מחירי החומרה המשמשת לחישוב רשתות נוירונים מלאכותיות, הביאו לפרץ של מאמרים העוסקים ביישום למידה עמוקה באמצעות חיזוקים בבעיות הנוגעות בכל תחומי החיים. בפרק זה נדון במגוון יישומים עכשוויים של למידה עמוקה באמצעות חיזוקים העוסקים בשבעה תחומים, קרי, פיננסים, רפואה, עיבוד שפות טבעיות, נהיגה אוטונומית, רובוטיקה, משחקים וראייה ממוחשבת.

4.1 פיננסים

למידה באמצעות חיזוקים היא פתרון טבעי לבעיות החלטה כלכליות או פיננסיות ונעשו מספר ניסיונות ליישם זאת אך עם הצלחה מוגבלת [37]. העולם הפיננסי בנוי מביצוע פעולות מסחר באמצעות מגוון כלים פיננסים שונים כגון מניות, אג״ח, נגזרים, אופציות וכד׳. באמצעות מכירה וקניה של נכסים פיננסים, הסוחר שואף להגדיל את הרווח המצטבר שלו או של קבוצת משקיעים אשר הוא מייצג. בסעיף זה נסקור שלושה מחקרים העוסקים ביישום למידה עמוקה באמצעות חיזוקים כדי לייצר סוכן אשר מבצע פעולות מסחר באופן אוטומטי במטרה להגדיל ככל שניתן את הרווחים הכספיים של המשקיעים.

4.1.1 למידה עמוקה באמצעות חיזוקים עבור ייצוג אובייקטים כלכליים ומסחר

אלגוריתם ה DDR [38] הוא למעשה האלגוריתם הידוע הראשון אשר עושה שימוש בלמידה עמוקה ובלמידה באמצעות חיזוקים במסחר פיננסי בזמן אמת. אלגוריתם זה הוא הרחבה של אלגוריתם למידה באמצעות חיזוקים למסחר Recurrent Reinforcement Learning (RRL) [37]. האלגוריתם המקורי עסק במסחר של נכס פיננסי בודד בעל מחיר P_t המתעדכן בכל יחידת זמן t. כמו כן, הפעולות שניתן לעשות הן מכירה (-1), קניה (+1) או שום דבר (0): $\delta_t \in \{-1,0,1\}$. החוקרים הגדירו פונקציית תגמול R_t באופן הבא: $R_t = \delta_t(P_t - P_{t-1}) + c|\delta_t - \delta_{t-1}|$, כאשר c היא עמלת ביצוע פעולת מסחר בודדת. מחירו של נכס פיננסי מושפע מהרבה פרמטרים ולכן גודל מרחב המצבים הוא עצום ופתרונות קלאסיים של למידה באמצעות חיזוקים אינם יכולים להתכנס לפתרון בזמן סביר. על כן, החוקרים החליטו להשתמש בפונקציית קירוב לא-לינארית התלויה בווקטור פרמטרים (שנבחרו בקפידה בהתאם למחקרים שונים על התנהגות מחירים) וב- m המצבים האחרונים. למעשה זהו מבנה של רשת מסוג Recurrent בעומק 1 (ללא שכבות נסתרות).
החוקרים [38] הרחיבו את האלגוריתם המתואר על ידי הוספת מספר שכבות נסתרות לרשת שהוצגה ב [37]. החוקרים בחנו את האלגוריתם בשלושה שווקים פיננסים שונים והראו כי בכל שוק האלגוריתם הצליח לייצר רווח גדול יותר מהאלגוריתם המקורי [37].

4.1.2 סוכן סוחר באמצעות רשתות נוירונים עמוקות מחזוריות

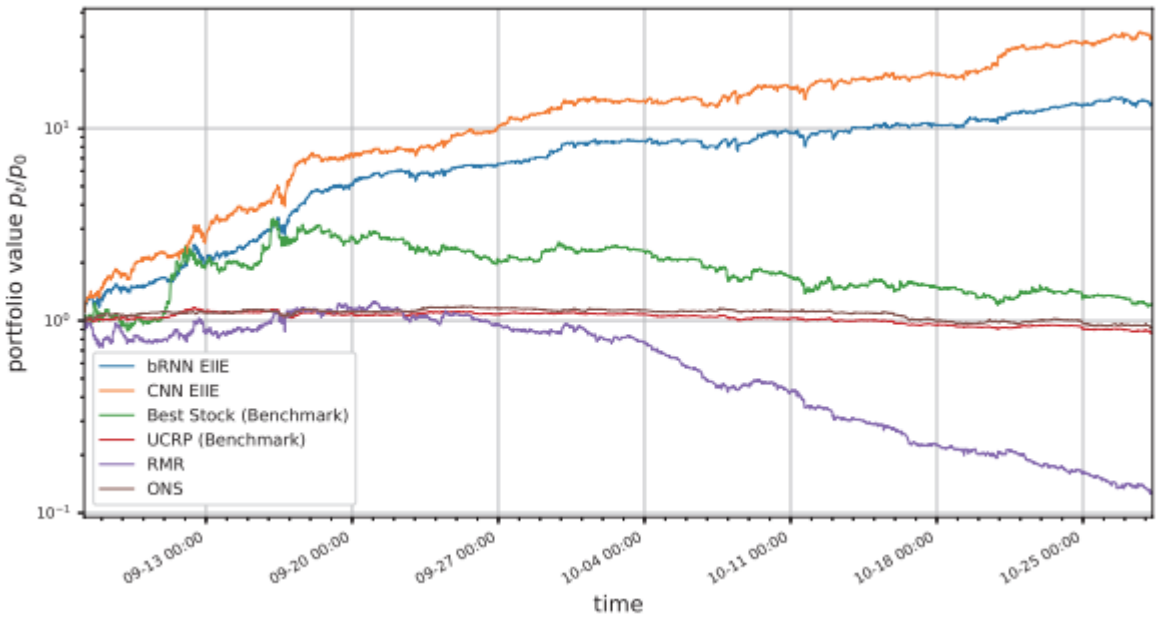
אלגוריתם Agent Inspired Trading Using Recurrent Reinforcement Learning and LSTM [39] הוא דוגמא נוספת לאלגוריתם אשר יישם שיטות למידה עמוקה באמצעות חיזוקים בכדי לייצר מכונת מסחר של נכסים פיננסים בזמן אמת. החוקרים הגדירו פונקציית תגמול המבוססת על מדד שארפ (מדד לביצועים של תיקי השקעות אשר מודד את עודף התשואה ליחידת סיכון על נכסי השקעה, כך ככל שהתשואה גבוהה יותר ביחס לסיכון כך המדד גבוה יותר) באופן הבא:

$$S_T = \frac{E[R_t]}{\sqrt{E\left[{R_t}^2\right] - (E[R_t])^2}}$$

כך ש R_t הוא הרווח של השקעה עבור זמן השקעה t (בדיוק כפי שהוגדר על ידי [37]) ו $E[R_t]$ היא תוחלת הרווח שלה. החוקרים השתמשו ברשת מסוג recurrent עם יחידות LSTM בכדי לקרב את המדיניות האופטימאלית מבלי לחשב פונקציית ערך (משתייך למשפחת שיטות גראדיינט המדיניות). כזכור, רשתות LSTM מסוג recurrent הן בעלות מספר רב של שכבות נסתרות מה שתורם ליכולת שלהן לבטא מערכות קשרים מורכבות, בדומה למערכות הקשרים הקיימות במסחר פיננסי אמיתי. החוקרים אימנו את הרשת עם מידע פיננסי של מדד USDGBP (מבטא את הקשר בין שער דולר אמריקאי ללירה סטרלינג) והשוו את ביצועיה אל ביצועי האלגוריתם שתואר בסעיף הקודם [38] והראו כי הרשת שלהם משיגה תוצאות טובות יותר, קרע הרווח הכולל גבוה יותר.

4.1.3 ניהול תיקי השקעות עם למידה עמוקה באמצעות חיזוקים

ניהול תיקי השקעות הוא תהליך בו מנהל השקעות מבצע כל העת החלטות חלוקה מחדש של כספים על פני מגוון נכסים פיננסים כאשר המטרה היא מקסום הרווחים תוך שמירה על סיכון מגודר. ישנם מספר אלגוריתמים למידה חישובית שמתמודדים עם בעיית ניהול תיק ההשקעות על ידי חיזוי ערכיהם של נכסים פיננסים בהתבסס על ערכים היסטוריים כקלט, כלומר מתייחסים לבעיה כבעיית רגרסיה. מתברר אפוא, כי ביצועיהם של אלגוריתמים מסוג זה אינם טובים שכן חיזוי מגמות מחירים של נכסים פיננסים היא בעיה מורכבת וקשה. קבוצת חוקרים [40] מציעה אלגוריתם המבוסס על אלגוריתם Deterministic Policy Gradient בשילוב עם Memory Replay. קירוב המדיניות נעשה בשלוש דרכים שונות קרי, באמצעות רשת מסוג Convolutional, רשת מסוג recurrent ורשת מסוג recurrent עם יחידות LSTM. כל רשת הוזנה בקלט הכולל את הנכסים שניתן לסחור בהם וערכיהם ההיסטוריים. כמו כן, הרשת הפיקה וקטור הסתברויות הקובע את משקל כל נכס בתיק ההשקעות. החוקרים בחרו לאמן את הרשת עבור מדדים של שוק המטבעות הדיגיטליים (Cryptocurrency). ביצועי האלגוריתם הושוו עם מספר אלגוריתמים קיימים וכן עם מספר אסטרטגיות קלאסיות מעולם הכלכלה. איור 3.1 מציג את ביצועי האלגוריתם המוצע אל מול אסטרטגיות השקע קלאסיות; החוקרים הראו כי באופן מובהק האלגוריתם עבור כל אחת משלושת האפשרויות שלו, הניב רווחים גבוהים יותר.



איור 3.1 – תוצאות השוואת האלגוריתמים המוצעים (רשת קונבולוציה בקו גרף כתום ורשת מסוג recurrent בקו גרף כחול) אל מול שתי אסטרטגיות השקעה קלאסיות מעולם הכלכלה (קווי גרף ירוק ואדום) ושני אלגוריתמים קיימים (קווי גרף סגול ואפור) [40]

4.2 רפואה

ישנן הרבה הזדמנויות ואתגרים ללמידה עמוקה בתחומי הרפואה ובכלל זה ללמידה עמוקה באמצעות חיזוקים [41]. המעבר לניהול רשומות רפואיות באופן דיגיטלי הוליד מאגרי מידע עצומים בגודלם עם מיליוני מקרים של אבחונים וטיפולים שניתנו בעקבתם. כעת, עם התפתחות המחקר של למידה עמוקה יחד עם התפתחות החומרה התומכת בהרצה של אלגוריתמים מסוג זה, החלו מופיעים מגוון רחב של יישומי למידה עמוקה בתחום הרפואה ובעיקר בתחום הרפואה המותאמת אישית (Personalized Medicine). הרפואה המותאמת אישית עוסקת באבחון והתאמת טיפול מתאים בצורה איטרטיבית בעיקר עבור מחלות כרוניות תוך שימוש בכל המידע הרפואי הקיים עבור חולה מסוים. בסעיף זה נביא שלושה מחקרים המדגימים כיצד ניתן לפתור בעיות בתחום הרופאה המותאמת אישית באמצעות למידה עמוקה באמצעות חיזוקים.

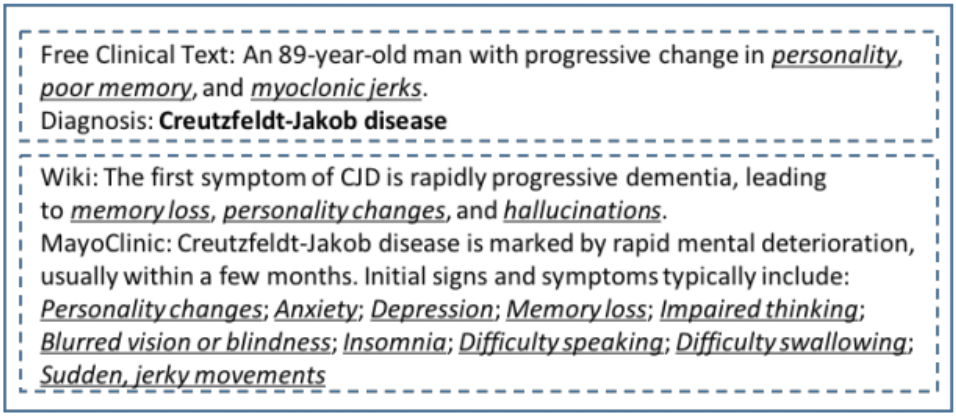
4.2.1 טיפולים רפואיים דינאמיים עם למידה עמוקה באמצעות חיזוקים

טיפולים רפואיים מורכבים לעיתים מסדרה של החלטות התערבויות (כדג' ניתוח כירוגי) אשר משתנות בהתאם למצבו הקליני של החולה בכל רגע נתון של הטיפול. טיפולים מסוג זה קרויים Dynamic Treatment Regimes או בקיצור DTR והמטרה העיקרית של הצוות הרפואי היא להגיע ל DTR האופטימלי, כלומר זה אשר ממקסם את התוצאה הקלינית בטווח הארוך [42]. בעיית מציאת הטיפול האופטימלי באמצעות למידה באמצעות חיזוקים נבחנה בעבר באמצעות שיטות הקרויות Sequential Multiple Assignment Randomized Trial. שיטות אלו היו מוגבלות על ידי מרחב החלטות הומוגני ומרחב פעולות מדרגה נמוכה, בעוד שמידע רפואי אלקטרוני מאופיין במרחב החלטות הטרוגני ובמרחב פעולות גדול. על כן, בכדי לעשות שימוש בשיטות אלו היה צורך לבצע תהליך של פישוט (Simplification) המידע הרפואי על ידי מומחים בתחום – מה שלא הוביל תמיד למציאת הטיפול האופטימלי. קבוצת החוקרים [42] יישמה את אלגוריתם DQN בכדי להתמודד עם בעיית מציאת הטיפול האופטימלי מבלי

לבצע פישוט של המידע הרפואי. על מנת להדגים את מידת האפקטיביות של יישום האלגוריתם, החוקרים מימשו אותו עבור טיפולים לחולי לוקמיה ומחלת GVHD מתוך מאגר המידע CIBMTR. בכל שלב, האלגוריתם בוחר N פעולות הטובות ביותר לטיפול עבור חולה נתון. החוקרים הצליחו להראות שרמת הדיוק של האלגוריתם מגיעה עבור N=10 ל- 90%. כמו כן, החוקרים השוו את ביצועי האלגוריתם אל מול מדיניות של בחירה אקראית של פעולות והראו כי האלגוריתם המוצע מייצר תוצאות טובות בהרבה.

4.2.2 אבחון קליני עם למידה עמוקה באמצעות חיזוקים

אבחון קליני היא משימה מאתגרת במיוחד שכן בהינתן מקרה קליני הכולל היסטוריה רפואית, סימפטומים ומצב נוכחי של חולה, על המאבחן לבצע בדיקות ופרוצדורות רפואיות, להסיק אבחון מדויק ולרשום את הטיפול הטוב ביותר שניתן בהתבסס על הניסיון והידע שלו. כמות המידע שהמאבחן נדרש לעבד ומורכבותו הגדולה הופכות את משימת האבחון למשימה קשה. מטרת המחקר [43] היא לפתח שיטה אשר מחלצת מתוך המידע הנתון למאבחן את המושגים הקליניים אשר מתארים באופן המיטבי את מצב החולה על ידי שימוש בראיות (Evidences) אשר נאספות ממקורות חיצוניים. איור 3.2 מציג דוגמא לקלט אשר חולצו ממנו מושגיים קליניים באמצעות שימוש במקור החיצוני ויקיפדיה.



איור 3.2 – דוגמא לחילוץ מושגים קליניים ממקורות חיצונים באמצעות המקור החיצוני ויקיפדיה (מלבן תחתון) מתוך תיק רפואי (מלבן עליון) [43]

האלגוריתם המוצע במאמר [43] הוא למעשה יישום של אלגוריתם DQN. החוקרים הגדירו מצב כווקטור עם ערכים רציפים המורכב מאוסף של מושגים קליניים שחולצו מתוך המידע הנתון למאבחן וכן אוסף מושגים קליניים אשר חולצו ממקורות מידע חיצוניים. הווקטור המייצג את המצב הוא למעשה ווקטור המתאר את הדמיון בין שני אוספי המושגים הקליניים. החוקרים הגדירו גם אוסף פעולות הכולל קבלת כל המושגים החדשים, דחייתם, קבלת מושג אחד או עצירה. פונקציית התגמול מחושבת על ידי שילוב מדידת האבחון הסופי וכן מידת דיוק חילוץ המושגים הקליניים. לצורך אימון הרשת החוקרים השתמשו ברשומות הרפואיות של מאגרי המידע TREC CDS ו HumanDX וכן במקורות המידע החיצוניים ויקיפדיה ו- MayoClinic. תוצאות המחקר מראות שהשילוב של למידה עמוקה באמצעות חיזוקים הצליח לייצר אבחון טובים יותר מאשר מערכות אשר אינן מבוססות על למידה באמצעות חיזוקים.

4.2.3 טיפול באלח דם עם למידה באמצעות חיזוקים

אלח דם (Sepsis) הוא מצב רופאי מסוכן המהווה גורם מוביל בתמותת חולים ואשר עלות הטיפול בו היא גבוהה מאוד. מלבד השימוש באנטיביוטיקה, הטיפול באלחי דם כולל שימוש בעירויים תוך וריידיים (IV)

ותרופות להורדת לחץ הדם (Vasopressors). אסטרטגיות השימוש השונות בטיפולים אלו הראו שינויים קיצוניים בתמותת חולים, מה שהופך את החלטות הטיפול לקריטיות. החוקרים [44] מציעים גישה חדשה למציאת אסטרטגיות טיפול באלחי דם המבוססת על שימוש ברשומות מידע רפואיות ובפרט במאגר המידע MIMIC-3. כמו במחקרים הקודמים, גם במחקר זה החוקרים יישמו את אלגוריתם DQN, אך בשונה מהם החוקרים יישמו גם את ההרחבות שלו, קרי Double DQN ו Dueling. מרחב הפעולות הוגדר להיות מטריצה בגודל 5x5 כאשר ציר אחד מתייחס לעירוי תוך ורידי והציר השני מתייחס לתרופות להורדת לחץ דם. החוקרים השתמשו במדד SOFA המתאר כישלון אברים (אשר נובעים כתוצאה מאלח הדם) כפונקציית התגמול. תוצאות המחקר אכן גילו אסטרטגיות טיפול אשר יכולות לשפר את איכות הטיפולים הניתנים במצבים של אלח דם.

4.3 עיבוד שפות טבעיות

אלגוריתמים רבים בתחום עיבוד שפות טבעיות (NLP) נשענים באופן ניכר על למידה חישובית, בין אם למידה רדודה (Shallow Learning) או למידה עמוקה (Deep Learning). על כן, יישום של למידה עמוקה באמצעות חיזוקים הוא טבעי. ואכן בשנים האחרונות החלו להופיע מחקרים המיישמים שיטות של למידה עמוקה באמצעות חיזוקים בכדי לפתור בעיות מעולם עיבוד השפות הטבעיות. בחלק זה נדון בשלושה מחקרים העוסקים בתתי התחומים של עיבוד שפות טבעיות, קרי, מערכות יוצרות דו-שיח, הבנת שפה ופישוט משפטים.

4.3.1 יצירת דו-שיח עם למידה עמוקה באמצעות חיזוקים

מערכות דו-שיח (Dialogue Systems) או מערכות יוצרות דו-שיח (Dialogue Generation Systems) הן מערכות אשר מיועדות לתקשר עם משתמשיהן באמצעות שפה טבעית (Natural Language) באמצעות טקסט, דיבור או שילוב של השניים. ניתן לחלק את המערכות לשני סוגים. הסוג הראשון הוא דו-שיח מכוון משימה אשר מיועד לקיים דו-שיח קצר עבור משימה ייעודית. דוגמאות מובהקות לסוג זה של מערכות דו-שיח הן סירי (Siri) של חברת אפל ואלקסה (Alexa) של חברת אמאזון אשר משמשות כעוזרות אישיות המסוגלות לספק הכוונות, להפעיל ולכבות מוצרי אלקטרוניקה וכד׳. הסוג השני הוא רובוט-שיחוח (ChatBot) אשר מיועד לקיים דו-שיח מתמשך עם המשתמש באופן המחקה התנהלות של שיחוח (Chat) טקסטואלי. דוגמאות מובהקות לכך הן הנציגים הווירטואליים באתרי אינטרנט המספקים שירות לקוחות ללקוחות האתרים בהם הם מוטמעים [45]. חוקרים הצליחו ליישם בהצלחה מערכות מסוג רובוט-שיחוח באמצעות רשתות נוירונים עמוקות במודל שזכה לשם SEQ2SEQ. למרות ההצלחה של מודל זה, עדיין רמת השיח הנוצרת על ידו אינה מחקה באופן מספק שיחה עם בן אדם. כך למשל, המודל נוטה לייצר תגובות כלליות בסגנון ״אינני יודע״ הנוטות לסיים שיחה. כמו כן, בעיה נוספת של המודל היא הנטייה להיכנס למעגל אינסופי של אותן התגובות. קבוצת חוקרים [46] מציעה כעת ליישם שיטות של למידה עמוקה באמצעות חיזוקים יחד עם SEQ2SEQ כדי להתגבר על הבעיות שהוזכרו לעיל. במודל שהחוקרים מציעים ישנם שני סוכנים, q ו-p אשר מתקשרים ביניהם בתורות, כך שבכל תור סוכן מייצר משפט והסוכן האחר מגיב לו עם משפט מתאים משלו. המשפטים אשר נוצרים הם בעצם הפעולות של הסוכנים בסביבה (מעל מרחב אינסופי של משפטים). המודל מבוסס על שיטת מדיניות הגראדיינט, כלומר קירוב של המדיניות על ידי שימוש ברשת נוירונים עמוקה. פונקציית התגמול היא הרכבה של מספר מקדמים, קרי, מידת *הקלילות* של התשובה כלומר כמה קל למצוא תשובה הולמת למשפט הנתון, כמות המידע החדש שהתשובה מכילה וקוהרנטיות סמנטית. החוקרים אימנו את המודל על מאגר המידע של OpenSubtitles ומדדו את ביצועי המודל ביחס

לאורך דו-שיח, מגוון המשפטים שבו ובאמצעות הערכה של בני אדם. בכל המדדים הללו המודל הצליח לייצר תוצאות טובות בהרבה ממודל SEQ2SEQ. איור 3.3 מביא מספר דוגמאות להשוואה בין תשובות לשאלות שסופקו על ידי האלגוריתם המוצע ואלגוריתם מסוג SEQ2SEQ.

Input Message	Mutual Information Model	RL
How old are you?	I'm 16.	I'm 16, why are you asking?
What's your full name?	I have no idea.	What's yours?
I don't want to go home tonight.	Really?	Why?
Do you have any feelings for me?	I don't know what you are talking about.	I don't want to hurt your feelings.
How much time do you have here?	Not long enough. Sorry, sir.	Ten seconds.
Shall we get started?	Of course!	Yes. We've got a lot of work to do here.
Do you play football?	No, i don't	Yes. I love football!
We'd have to talk to him.	I mean, he's a good guy	About what ?
How come you never say it?	Because I don't want to hurt you.	I don't think it's a good idea to say it.

איור 3.3 – השוואה בין תשובות לשאלות שסופקו על ידי האלגוריתם המוצע ואלגוריתם מסוג SEQ2SEQ [46]

4.3.2 הבנת שפה של משחקים מבוססי טקסט עם למידה עמוקה באמצעות חיזוקים

בטרם הופיעו משחקי האסטרטגיה עם גרפיקה ממחושבת הענף נשלט על ידי משחקים מבוססי טקסט, כלומר, הסצנה הייתה מתוארת לשחקן באמצעות תיאורים טקסטואליים ובהתאם לתיאורים אלו השחקן היה צריך להגיב באמצעות שפה טבעית. כך לדוגמא, המשחק היה מתאר סיטואציה בה השחקן עומד באמצע שביל, משמאלו יש בור ומימינו יש מסלול פנוי, כדי להתקדם על השחקן לפקוד בשפה טבעית את המילים *צעד ימינה*. היות והסצנה המתוארת אינה ניתנת לצפייה באופן ישיר, על השחקן להבין את משמעות הטקסט כדי לפעול, מהלך אשר הופך פתרון משחקים אלו באמצעות בינה מלאכותית למאתגר [47]. ניסיונות עבר לפתרון בעיות אלו כללו ייצוג של המצבים באמצעות שק מילים (Bag of Words) מתוך התיאורים של המצבים. ייצוג זה אינו מבחין בסדר בו המילים ניתנות ומהדקויות בהם המשפטים והפסקאות מורכבים. קבוצת חוקרים [47] מציעה כעת לפתור בעיות מסוג זה באמצעות יישום טכניקות של למידה עמוקה באמצעות חיזוקים. החוקרים בחרו לקרב את פונקציית Q באמצעות רשת נוירונים עמוקה המורכבת משני חלקים, קרי, החלק הראשון ממיר תיאורים טקסטואליים לווקטור ייצוג של המצבים. חלק זה מומש באמצעות יחידות LSTM בכדי לתפוס תיאורים על פני הזמן. החלק השני מנקד את הפעולות בהינתן ווקטור הייצוג של המצבים אשר חושב בחלק הראשון. החוקרים אימנו את המודל עבור שני משחקי אסטרטגיה והשוו את הביצועים בין שלושה מודלים, קרי, המודל המוצע, מודל בו שחקן המבצע פעולות באופן אקראי ומודל בו שחקן המבוסס על שיטת שק המילים. החוקרים הצליחו להראות שהמודל המוצע טוב יותר בהרבה משני המודלים האחרים.

4.3.3 פישוט משפטים עם למידה עמוקה באמצעות חיזוקים

המטרה העיקרית של תהליך פישוט משפטים היא להפוך משפטים לקלים לקריאה ולהבנה תוך שמירה על המידע המקורי ומשמעותו. ישנם שימושים נוספים לתהליך זה, כך לדוגמא ניתן להשתמש בפישוט משפטים כשלב מקדים באלגוריתמים אחרים של עיבוד שפות טבעיות כמו מערכות דו-שיח. פישוט משפטים יכול אפילו לשמש אנשים עם כישורי שפה נמוכים כמו אנשים עם מוגבלויות. תהליך פישוט כולל בתוכו *החלפה* של מילים נדירות במילים או ביטויים שכיחים, המרת מבנים סינטקטיים מורכבים לפשוטים יותר ומחיקת אלמנטים אשר לא תורמים למידע המקורי [48]. לאחרונה קבוצת חוקרים [48] מציעה ליישם שיטות של למידה עמוקה באמצעות חיזוקים בכדי לפתור את בעיית פישוט המשפטים. המודל המוצע חוקר את מרחב הפישוטים האפשריים תוך כדי למידה כיצד ניתן למקסם פונקציית תגמול אשר מעודדת תוצרים מופשטים

תחת מגבלות ברורים. החוקרים מאפיינים את הבעיה באופן הבא: תחילה הסוכן קורא את המשפט לפישוט ולאחר מכן בכל שלב, מבצע פעולה אחת (החלפה, מחיקה, העתקה וכדי) בהתאם למדיניות, עד אשר לא ניתן לבצע עוד פעולות על המשפט. המשפט הסופי הוא המשפט המפושט. החוקרים בחרו לקרב את המדיניות באמצעות שיטת מדיניות הגראדיינט. פונקציית התגמול היא הרכבה של שלושה גורמים, קרי, רמת הפישוט באמצעות שימוש באלגוריתם SARI לניקוד, רלוונטיות ורהיטות. החוקרים אימנו את המודל על שלושה מאגרי מידע: WikiSmall אשר מיישר ערכים מויקיפדיה הרגיל ומויקיפדיה המופשט, WikiLarge בדומה למאגר הראשון אך עם יותר ערכים ו- Newsela המכיל למעלה מאלף ידיעות חדשותיות אשר כל אחת נכתבה ארבע פעמים בידי עורכים מקצועיים עבור ילדים. החוקרים השוו את ביצועי המודל המוצע אל מול מספר מודלים ידועים, בכל אחת מן השוואות ביצועי במודל המוצע היו טובים יותר.

4.4 נהיגה אוטונומית

נהיגה אוטונומית מתייחסת ליכולת של רכבים לנהוג את עצמם מנקודה לנקודה ללא הכוונה של אדם. בשנים האחרונות תחום זה צובר פופולאריות רבה בקרב יצרניות רכב, חברות הייטק ואנשי אקדמיה וניתן לראות היום מכוניות אוטונומיות נעות במספר ערים ברחבי העולם. הנהיגה האוטונומית איננה מתבטאת במכונה או המצאה בודדת, אלא במכלול שלהם המהווה מעין מערכת אקולוגית (Ecosystem) בין מערכות רכב פנימיות, בין רכבים שונים ובין הסביבה. בחלק זה נסקור שלושה מחקרים העוסקים בשילוב למידה עמוקה באמצעות חיזוקים בכדי ליישם מערכות נהיגה אוטונומית.

4.4.1 מערכת בלימה אוטונומית לרכב עם למידה עמוקה באמצעות חיזוקים

תפקידה של מערכת הבלימה ברכב הוא להקטין את המהירות של הרכב עד לעצירה מוחלטת שלו עבור מגוון של סיטואציות כדוגמת רמזור אדום, העלאה והורדה של נוסעים וכדי. קבוצת חוקרים [49] בחרה לתכנן מערכת בלימה אוטונומית לסיטואציות של נהיגה במרחב אורבני בו רכבים נפגשים עם הולכי רגל אשר חוצים את הכביש בזמן אקראי. החוקרים אפיינו את הבעיה כבעיית למידה באמצעות חיזוקים והשתמשו באלגוריתם DQN בכדי לפתור אותה. תחילה אפיינו החוקרים את אוסף הפעולות האפשריות כהאצה גדולה, האצה בינונית, האצה קטנה או לא לעשות דבר. אחר כך הם הגדירו פונקציית תגמול מתוך ההכרה שיש לאזן בין שני מצבים, קרי, הימנעות מוחלטת מהתנגשות ויציאה מסיטואציה מסוכנת מהר ככל שאפשר, שכן באין איזון אנו עלולים לקבל מערכת שמרנית או חסרת אחריות. החוקרים בחנו את האלגוריתם באמצעות שימוש בסימולטור המדמה מערכת פיזיקלית אמיתית. הניסוי הראה כי עבור מצבים בהם יש יותר משניה וחצי לפני פגיעה הרכב עצר בכל הפעמים, במצבים בהם יש פחות משניה וחצי היו מעט פגיעות – שלא היה ניתן להימנע מהם בגלל מהירות הרכב.

4.4.2 מערכת שמירה על אי-סטייה מנתיב נסיעה עם למידה עמוקה באמצעות חיזוקים

רכיב חשוב בנהיגה אוטונומית הוא שמירה על נתיב הנסיעה (Lane Keeping) כפי שהוא מסומן על הכביש. קבוצת חוקרים [50] יישמה מודל נהיגה אוטונומית לרכב אשר שומרת על הרכב בתוך תוואי של נתיב מוגדר באמצעות טכניקות של למידה עמוקה באמצעות חיזוקים. הקלט למערכת הוא הרכב של מצב הרכב (מהירות, האצה, מיקום, משקל וכדי) ומצב הסביבה (אובייקטים בסיבה, מימדיהם, כיוונם, גבולות הנתיב וכדי). החוקרים בחרו לממש שני אלגוריתמים של למידה עמוקה באמצעות חיזוקים, קרי, DQN עבור מרחב פעולות בדיד ושחקן-מבקר עבור מרחב פעולות רציף. בכדי לבדוק את ביצועי המודלים המוצעים, החוקרים השתמשו בסימולטור אשר מדמה את מצב הרכב והסביבה. שני האלגוריתמים התכנסו לתוצאה, אך בעוד

שאלגוריתם DQN התכנס מהר יותר, אלגוריתם שחקן-מבקר הניב תוצאה חלקה יותר מבחינת איכות הנסיעה.

4.4.3 מערכת ניהול צמתים אוטונומית עם למידה עמוקה באמצעות חיזוקים

מערכות ניהול צמתים אוטונומיים (Autonomous Intersection Management) מהוות נקודת שליטה מרכזית עבור רכבים המעוניינים לחצות צומת באופן זה שהמערכת מקצה מרווח זמן (Timeslot) לכל רכב בכדי שיוכל לחצות בבטחה (מבלי להתנגש ברכבים אחרים) את הצומת. בפרט, ההחלטות על מהירות והאצה של הרכב מוטלת על כל רכב באופן בלתי תלוי ברכבים האחרים, על כן מערכות מסוג זה מתאימות יותר לצמתים מרכזיים בהם המהירות של הרכבים השונים תואמים פחות או יותר. בסיטואציות בהם יש תנועה דלה של רכבים (כבישים מקומיים לדוגמא) אנו זקוקים למערכת יעילה יותר. שני חוקרים [51] פיתחו מערכת ניהול צמתים אוטונומיים המתאימה לסיטואציות כאלו בשילוב עם שיטות למידה עמוקה באמצעות חיזוקים. המערכת מנהלת אזורים גאוגרפים באופן בלתי תלוי. כל הרכבים הנמצאים באזור גאוגרפי נתון מתקשרים עם המערכת ומעבירים לה את נתוני המהירות שלהם, משקל הרכב וכד׳. היות ורכבים יכולים להיכנס ולצאת מהאזורים המנוהלים, המערכת מחשבת בכל רגע נתון את המהירות והאצה הנדרשים לכל רכב. החוקרים החליטו לנסות לפתור את הבעיה באמצעות אלגוריתם TRPO (סעיף 2.2.1.2) עם שימוש בפונקציית תגמול אשר מעניקה ניקוד חיובי כאשר כל המכניות מגיעות ליעדן בזמן כולל סביר וניקוד שלילי בכל מצב אם ישנה התנגשות בין מכוניות. החוקרים בנו מודל מסוג תכנון דינאמי לפתרון הבעיה, והשתמשו בסימולטור בכדי להשוות בין הביצועים של מודל TRPO ומודל התכנון הדינאמי. תוצאות הניסוי הראו מחד, שאזורים קטנים יחסית הפתרון הדינאמי יעיל יותר מבחינת הזמן אך זמן החישוב שלו הוא עצום לעומת זמן החישוב של מודל המוצע. כמו כן, עבור אזורים גדולים יחסית מודל התכנון הדינאמי אינו מסיים את החישוב כלל בעוד שהמודל המוצע מסיים בזמן סביר ומייצר תוצאה.

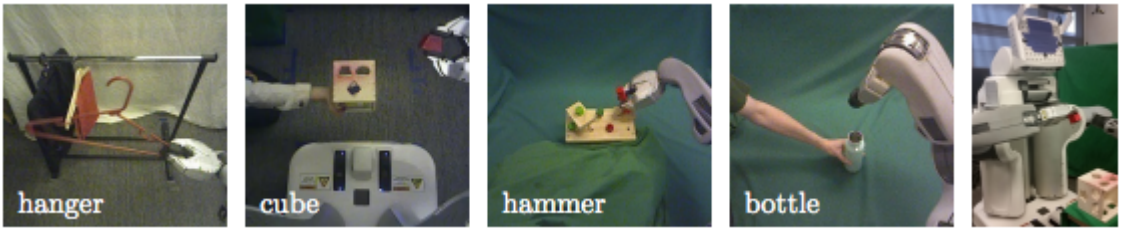
4.5 רובוטיקה

רובוטים הם סוכנים אשר מבצעים משימות על ידי תמרון העולם הפיזי. אחת המטרות החשובות ביותר של רובוטיקה היא בניית רובוטים המסוגלים לבצע פעילויות אנושיות בצורה חלקה וטבעית. דרך מבטיחה אחת להשיג זאת היא על ידי יצירת רובוטים היכולים ללמוד כישורים חדשים בעצמם, בדומה לבני אדם. אך, רכישת כישורים מוטוריים חדשים אינה משימה קלה ומערבת סוגים שונים של למידה. למידה באמצעות חיזוקים בהקשר של רובוטיקה שונה באופן ניכר מרוב בעיות למידה באמצעות חיזוקים המוכרות. בעיות ברובוטיקה מיוצגות לעיתים עם מימדיות גבוה, פעולות ומצבים רציפים. לעיתים ברובוטיקה לא מציאותי להניח שהמצב האמיתי הוא נצפה לחלוטין וחסר רעשים. מערכת הלמידה לא תמיד תוכל לדעת באיזה מצב היא נמצאת וייתכן אפילו שמצבים שונים ייראו דומים מאוד. כידוע, למידה באמצעות חיזוקים מבוססת על לימוד מתוך ניסיון, אך השגת ניסיון במערכת פיזית אמיתית היא מאתגרת, יקרה ולפעמים אף קשה לשחזור. מסיבות אלו היישום של למידה עמוקה באמצעות חיזוקים עבור רובוטים הפך פופולארי מאוד בעשור האחרון בקרב קבוצות מחקר וחברות טכנולוגיה. בחלק זה נתמקד בשני מחקרים פורצי דרך בתחום הרובוטיקה.

4.5.1 חיפוש מדיניות מונחה

מערכת רובוטית מורכבת בדרך כלל משתי יחידות, קרי, מערכת החישה המורכבת ממספר חיישנים, מצלמות וכד׳ ומערכת השליטה המבטאת את היכולות פעולה של הרובוט בעולם הפיזי כגון מנוע, מפרק

מכני וכד׳. אחד האתגרים הגדולים ביותר בתכנון מערכות רובוטיות הוא יצירת התוכנה אשר מנהלת את שתי היחידות האלו. קבוצת חוקרים [52] בדקה את האפשרות להשתמש בטכניקות של חיפוש מדיניות (Policy Search) בכדי לאמן את שתי המערכות יחד כיחידה אחת. האלגוריתם שהם יצרו נקרא Guided Policy Search והוא בעצם ממיר את אלגוריתם חיפוש המדיניות הקלאסי לאלגוריתם למידה מפוקחת (או מודרכת) (Supervised Learning) על ידי בניית קבוצת האימון (Training Data) בתהליך איטרטיבי ויעיל של מיטוב מסלול (Trajectory Optimization). תהליך מיטוב מסלול הוא תהליך מציאת מסלול אשר ממקסם מדד ביצועים כלשהו תחת אוסף מגבלות ידוע. החוקרים בחרו לייצג את המדיניות כרשת נוירונים עמוקה קונבולוציונית (Convolutional Neural Network) בעלת שבע שכבות חבויות ועם קלט ממימד 92,000. כמו כן, החוקרים מימשו את האלגוריתם באופן שבו כל אחת ממערכות הרובוט, קרי, חישה ושליטה מאומנות בנפרד. לצורך ההשוואה בין שני האלגוריתם החוקרים בחנו את ביצועי כל אחד מהאלגוריתמים בארבע משימות: תליית קולב על מתלה, הכנסת קובייה לחור בגודל המתאים, דפיקת מסמר וחליצת פקק של בקבוק. איור 3.4 מציג תמונה לכל משימה בה נבדק האלגוריתם. תוצאות המחקר הראו כי האלגוריתם אשר מאמן את שתי המערכות יחד הצליח במשימות השונות באחוזים גבוהים יותר, לעיתים פער של חמישים אחוזים, מאשר האלגוריתם השני אשר מאמן את המערכות בנפרד.

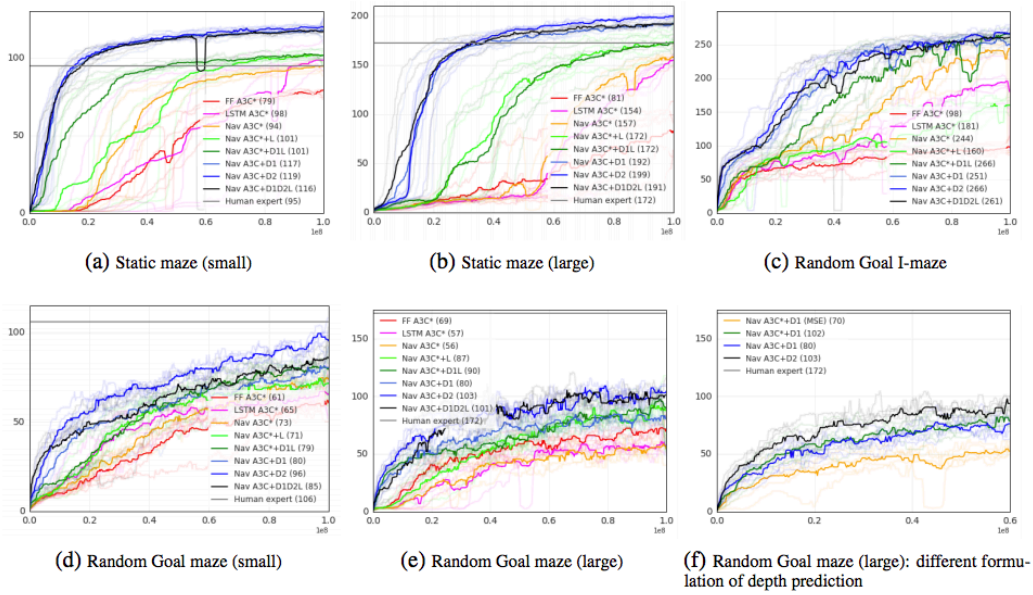


איור 3.4 – הרובוט (ימין) וארבע המשימות שהאלגוריתם נבחן עליהם (שמאל) [52]

4.5.2 ללמוד לנווט ביעילות בסביבות מורכבות

היכולת לנווט ביעילות בסביבה נתונה הוא בסיסי בכדי לייצר התנהגות אינטליגנטית לרובוט. השיטות המקובלות היום בכדי להשיג זאת מתבססות על הסקת מיקום ומיפוי של הסביבה. קבוצת חוקרים מחברת DeepMind [53] מציעה לפתח אלגוריתם ניווט על ידי יישום מודל למידה עמוקה באמצעות חיזוקים, קרי, A3C (שחקן-מבקר). ישנם מספר אתגרים בפיתוח אלגוריתם ניווט באמצעות למידה באמצעות חיזוקים. ראשית, התגמולים בד״כ נדירים על פני הסביבה בה הסוכן מסתובב, שכן לעיתים ישנה רק נקודת מטרה אחת ורק כאשר הרובוט מגיע אליה ניתן תגמול. שנית, סביבות מכילות לעיתים אלמנטים דינאמיים והרובוט הלומד חייב לנהל זיכרון לטווח ארוך עבור מציאת המטרה וזיכרון לטווח קצר בכדי ללמוד תנועה ומיקום של אלמנטים דינאמיים. בכדי להתמודד עם האתגרים האלו, החוקרים מציעים לחשב את פונקציית התגמול בסיוע של שתי טכניקות. טכניקה אחת היא למידת מפת עומקים ממימד נמוך על ידי חיזוי האפנות (Modality) של ערוץ צבע אחד מתוך ערוצי הצבע האחרים. מטרת טכניקה זו היא ללמוד את הגיאומטריה התלת ממדית של הסביבה ובכך למנוע היתקלות במכשולים. טכניקה נוספת להתמודדות עם האתגרים שצוינו היא באמצעות חיזוי בכל רגע נתון האם הרובוט ביקר במיקום הנוכחי. החוקרים יישמו מספר אלגוריתמים אשר כולם מתבססים על אלגוריתם A3C. השוני בכל מימוש בא לידי ביטוי במבנה רשת הנוירונים: רשת קונבולוציה, רשת מסוג Recurrent עם יחידת LSTM, רשת מסוג Recurrent עם שתי יחידות LSTM ו רשת מסוג Recurrent עם שתי יחידות LSTM יחד עם חיזוי עומקים ומיקום (כפי שהוסבר קודם). החוקרים אימנו את המודלים באמצעות סימולטור במגוון מבוכים תלת ממדיים עם מטרות/תגמולים נייחים ואקראיים כאשר הקלט הוא תמונת RGB. איור 3.5 מציג השוואה בין ביצועי כל

האלגוריתמים שמומשו במסגרת המחקר בכל אחד מהמבוכים שנבדקו. תוצאות המחקר הראו כי השימוש בטכניקות השונות זירזו את התכנסות האלגוריתם והצליחו לייצר ניקוד גבוה יותר.



איור 3.5 – השוואה בין ביצועי כל האלגוריתמים שמומשו בכל אחד מהמבוכים שנבחנו [53]

4.6 משחקים

משחקים הם פלטפורמות טובות מאוד לבחינה של אלגוריתמים של בינה מלאכותית ובפרט של למידה באמצעות חיזוקים. דוגמא טובה לכך הוא אלגוריתם DQN שנחשב לפריצת דרך בתחום של למידה עמוקה באמצעות חיזוקים וכל הנגזרות שלו (שנסקרו בפרק 2) אומנו ונבדקו מעל פלטפורמת משחקי הוידאו אטארי 2600. באופן כללי ניתן לסווג משחקים לשתי קבוצות, קרי, משחקים עם ידע מלא כדוגמת שח-מט וכן משחקים עם ידע חלקי כדוגמת פוקר. בחלק זה נדון בשני מחקרים פורצי דרך אשר משלבים טכניקות של למידה עמוקה באמצעות חיזוקים בכדי לפתור משחקים עם ידע מלא וחלקי.

4.6.1 אלפא-גו

אלגוריתם אלפא-גו (AlphaGo) [8] הוא אלגוריתם אשר לומד לשחק את המשחק GO. משחק GO הוא משחק לוח אסטרטגי לשני שחקנים והוא משתייך לקבוצת המשחקים עם ידע מלא, שכן כל שחקן רואה את כל הלוח. האלגוריתם נחשב לאלגוריתם הראשון אשר השיג ביצועים טובים יותר משל אדם, ויכולות אלו הודגמו בניצחון אלוף העולם במשחק GO בשנת 2015. האלגוריתם כלל שתי רשתות נוירונים עמוקות אשר אומנו באמצעות למידה מונחית (Supervised Learning) של מומחים אנושיים וכן באמצעות למידה באמצעות חיזוקים. הפעולות שנבחרו בכל שלב של המשחק נבחרו באמצעות שימוש בעצי חיפוש, בפרט עצי חיפוש מסוג מונטה קרלו (Monte Carlo Tree Search) מעל התוצאות של רשת הנוירונים. אף על פי שאלגוריתם זכה להצלחה גדולה יש לו מספר חסרונות אשר מקשים על הכללתו עבור משחקים אחרים. ראשית, האלגוריתם זקוק למומחים אנושיים בכדי ללמוד אשר לעיתים אינו אמין או פשוט בלתי ניתן להשגה. שנית, האלגוריתם היה מורכב מאוד וכלל שתי רשתות נוירונים ואלגוריתם חיפוש עצים מורכב מה שתרם לזמן אימון ארוך מאוד של 60 ימים. בכדי להתמודד עם חסרונות אלו, ובכדי להותיר אפשרות של הכללת האלגוריתם לטובת פתרון משחקים נוספים חוקרי AlphaGo פיתחו את אלגוריתם AlphaGo Zero. אלגוריתם זה הוא למעשה יישום מלא של למידה עמוקה באמצעות חיזוקים ולכן איננו דורש ידע אנושי

כלשהו על המשחק (ומכאן שמו). האלגוריתם כולל רשת נוירונים עמוקה בודדת אשר מקבלת כקלט ייצוג של הלוח ומיקומי הכלים עליו וכן עץ חיפוש מסוג מונטה קרלו אשר בוחר פעולות מתוך הפעולות שהרשת מחשבת. החוקרים אימנו את הרשת במשך שלושה ימים והשוו את ביצועיה ביחס לאלגוריתם המקורי. תוצאות ההשוואה הראו באופן חד משמעי כי האלגוריתם החדש הוא בעל ביצועים טובים וכן מהירות ההתכנסות שלו קטנה פי עשרים. יתרה מכך, החוקרים בחנו את שני האלגוריתמים כאשר הם משחקים אחד מול השני והראו ש- AlphaGo Zero מנצח בכל משחק.

4.6.2 למידת אסטרטגיות למציאת קירוב לנקודת שווי משקל נאש עבור משחקים עם ידע חלקי

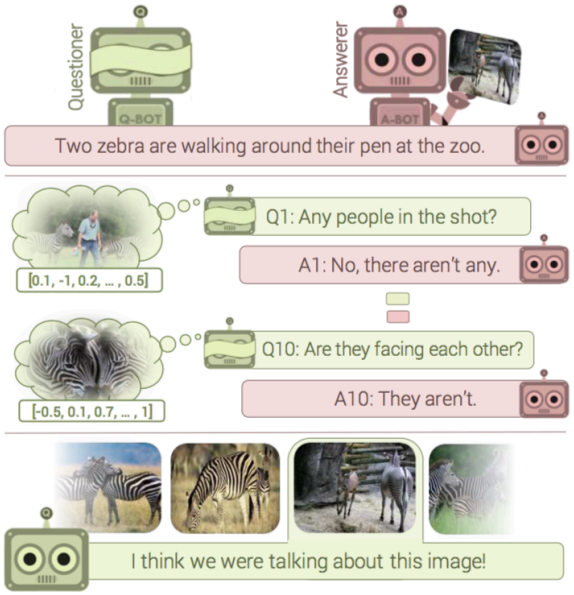
אחת הסיבות לעסוק במציאת פתרונות יעילים למשחקים היא בכדי לפתח אלגוריתמים אשר יוכלו להתמודד עם בעיות עולם מורכבות כדוגמת ביטחון, אבטחת רשתות, מסחר פיננסי, מערכות בקרה ושליטה וכד׳. מרבית בעיות העולם האמיתי כרוכות בקבלת החלטות עם ידע חלקי. תורת המשחקים מגדירה פתרון אופטימלי לבעיות מסוג זה כנקודת שוויי משקל נאש (Nash Equilibrium), כלומר אסטרטגיה בה אין לשום שחקן מוטיבציה (רווח או תועלת) לסטות ממנה [54]. אף על פי שמרבית האלגוריתמים של למידה חישובית מצליחים למצוא פתרון כמעט אופטימלי למשחקים עם ידע מלא, הם אינם מצליחים להתכנס לפתרונות עבור משחקים עם ידע חלקי. כמו כן מרבית הגישות של תורת המשחקים למציאת נקודת שיווי משקל נאש הן קשות להכללה. אחת השיטות למציאת נקודת שיווי משקל נאש היא באמצעות Fictitious Self-Play. בשיטה זו השחקנים בוחרים את התגובה הטובה ביותר בהתאם להתנהגות הממוצעת של היריב שלהם. קבוצת חוקרים [54] יישמה שיטות של למידה עמוקה באמצעות חיזוקים בכדי לפתח את האלגוריתם Neural Fictitious Self-Play ללמידת אסטרטגיות שהן קירוב לנקודת שיווי משקל נאש של משחקים עם ידע חלקי. שחקן (או סוכן) של האלגוריתם מורכב משתי רשתות נוירונים עמוקות. הרשת הראשונה מאומנת על ידי למידה באמצעות חיזוקים המבוססת על ניסיון המשחק של השחקן מול שחקנים אחרים. הרשת השנייה מאומנת באמצעות למידה בהשגחה המבוססת על ניסיון של השחקן עם ההתנהגות שלו עצמו. השחקן של האלגוריתם מגיב באמצעות ערבוב בין שתי הרשתות. החוקרים בחנו את האלגוריתם המוצע באמצעות שני משחקי פוקר המיועדים לשני שחקנים, קרי, Leduc Poker ו- Limit Texas Hold'em. עבור Leduc Poker האלגוריתם המוצע התקרב לנקודת שיווי משקל נאש בעוד שאלגוריתמים למידה באמצעות חיזוקים אחרים לא הצליחו לכלל להתכנס. עבור Limit Texas Hold'em האלגוריתם למד אסטרטגיה אשר התקרבה לביצועים של האלגוריתם הטוב ביותר אשר הונדס בצורה ידנית שלא ניתנת להכללה. יתרה מכך, החוקרים יישמו את אלגוריתם DQN עבור משחקי הפוקר שתוארו קודם והראו כי ביצועיו אל מול אלגוריתם ה- Neural Fictitious Self-Play נחותים בהרבה.

4.7 ראייה חישובית

ראייה חישובית (Computer Vision) היא תחום מחקר במדעי המחשב העוסק בשאלה כיצד מחשב יכול *להבין* משמעות של תמונה או קטע וידאו. הבנה בהקשר זה מתייחסת ליכולת של האלגוריתם לחלץ ולנתח בצורה אוטומטית פרטי מידע שימושים מן התמונה ולהמיר אותם לייצוג סימבולי כלשהו. בשנים האחרונות החלו חוקרים ליישם שיטות של למידה עמוקה וכן למידה עמוקה באמצעות חיזוקים בכדי לפתור בעיות מתחום הראייה החישובית. שילובים מסוג זה הולידו אלגוריתמים פורצי דרך עם ביצועים ברמת אדם אשר עולים בכמה מונים על אלגוריתמים קלאסיים של ראייה חישובית. בחלק זה נתאר אלגוריתם ראייה חישובית אשר מיישם שיטות של למידה עמוקה באמצעות חיזוקים.

4.7.1 מערכות דו-שיח ויזואליות

מערכות דו-שיח ויזואליות הן מערכות דו-שיח אשר מתנהלות בהקשר לתמונה, כלומר, במערכות אלו הסוכנים מתקשרים ביניהם בהתאם להבנה שלהם את תוכן תמונה. האלגוריתמים הקיימים אשר עוסקים בתחום זה משתמשים בטכניקות של למידה מפוקחת (מודרכת), מה שיצר מוגבלות ביכולות ללמוד ולפתח שיחות אנושיות אמתיות ביחס לתמונה. קבוצת חוקרים [55] פיתחה אלגוריתם למידה עמוקה באמצעות חיזוקים אשר מאמן שני סוכנים בכדי שיוכלו לנהל ביניהם דו-שיח שיתופי במטרה להבין תוכן של תמונה. החוקרים הגדירו את הבעיה בצורת של משחק בין שני סוכנים, קרי, סוכן-שואל וסוכן-עונה. בתחילת כל משחק הסוכן-שואל מקבל תיאור של תמונה (שלא ראה) בן משפט אחד בלבד והסוכן-עונה מקבל את התמונה עצמה. הסוכן-שואל מתקשר בשפה טבעית עם הסוכן-עונה לגבי מה מופיע בתמונה, כאשר המטרה בסופו של דבר שהסוכן-שואל יזהה את התמונה המדוברת מתוך אוסף תמונות בהתאם להבנתו את תוכן התמונה. איור 3.6 מתאר דוגמא לדו-שיח בין הסוכן השואל לבין הסוכן העונה ביחס לתמונה המתארת שתי זברות שהולכות ליד הכלוב שלהן בגן החיות.



איור 3.6 – דוגמא לדו-שיח בין סוכן-שואל וסוכן עונה ביחס לתמונה המתארת שתי זברות שהולכות ליד הכלוב שלהן בגן חיות [55]

החוקרים הגדירו את מרחב הפעולות שיש לסוכנים כרצף מעל מרחב מילים נתון באנגלית. מרחב המצבים מוגדר לכל סוכן בנפרד היות וכל סוכן חשוף למידע שונה. האלגוריתם לומד לקרב באמצעות שתי רשתות נוירונים עמוקות שתי מדינויות- אחת לכל סוכן, כאשר לסוכן-שואל יש רשת נוספת מייצרת תמונה אחרי כל סיבוב בהתאם למה שנלמד בו. הרשתות אומנו באמצעות שיטת גראדיינט המדיניות. התגמול משותף לשני הסוכנים ומוגדר להיות בכל שלב בשינוי במרחק בין הייצוג האמיתי של התמונה לבין החיזוי שלה. החוקרים אימנו את האלגוריתם מעל מאגר התמונות VisDial ובחנו את ביצועיו אל מול אלגוריתם קיים של מערכות דו-שיח ויזואלי המבוסס על למידה בהשגחה. תוצאות המחקר מראות שהאלגוריתם המוצע מזהה טוב יותר תמונות ומייצר דו-שיח אינפורמטיבי יותר.

5. יישומים עכשוויים של למידה עמוקה באמצעות חיזוקים – מבט מעמיק

בפרק הקודם סקרנו שבעה עשר מחקרים עכשוויים העוסקים ביישום עקרונות של למידה עמוקה באמצעות חיזוקים במטרה לפתור בעיות ממגוון תחומי חיים. סקירה רב-תחומית זו היא אולי העדות המשמעותית ביותר לפוטנציאל האדיר של למידה עמוקה באמצעות חיזוקים בדרכה להפוך למסגרת תאורטית של בינה מלאכותית לפתרון בעיות כלליות. בפרק זה נתמקד ביתר שאת בשני מחקרים שהוצגו בפרק הקודם קרי, אלפא-גו אפס (AlphaGo Zero) [8] מעולם המשחקים ומערכות דו-שיח ויזואליות (Visual Dialog Systems)55]] מעולם הראייה החישובית.

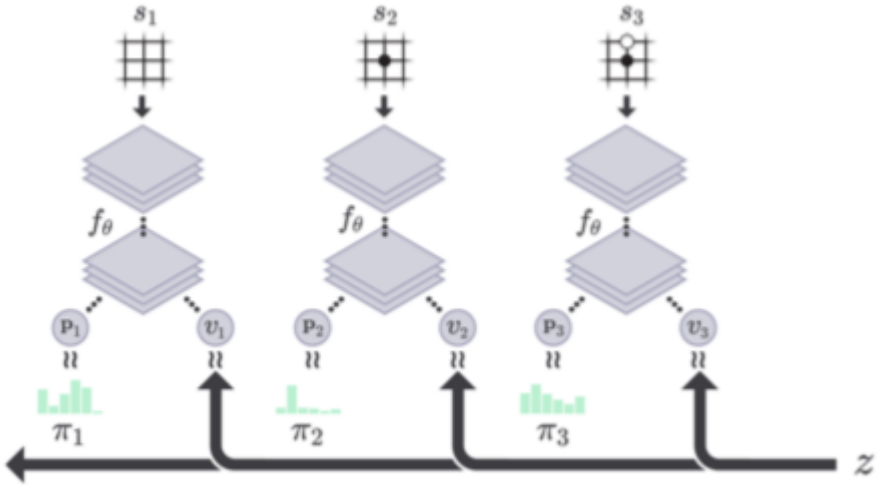
5.1 אלפא-גו אפס

אלגוריתם אלפא-גו המקורי [56] או בשמו המלא אלפא-גו פאן (AlphaGo Fan) הדהים את העולם כאשר בשנת 2015 הביס את אלוף אירופה במשחק גו (Go) Fan Hui והיה לתוכנת המחשב הראשונה שעושה זאת. אלגוריתם אלפא-גו הוא אלגוריתם למידה עמוקה באמצעות חיזוקים אשר מבוסס על שימוש בשני רשתות נוירונים, קרי, רשת לקירוב המדיניות (Policy Network) אשר מחשבת הסתברויות למהלכי משחק אפשריים ורשת לקירוב ערכים (Value Network). הרשת לקירוב המדיניות אומנה תחילה באמצעות למידה מונחית בכדי לנבא בדיוק רב מהלכים של מומחי משחק אנושיים, ולאחר מכן "חודדה" על יד שימוש באלגוריתם מדיניות הגראדיינט (Policy Gradient). הרשת לקירוב ערכים אומנה בכדי לנבא את המנצח של משחקים אשר שוחקו באמצעות רשת המדיניות כנגד עצמה. לאחר שהרשתות אומנו הן שולבו יחד עם עץ חיפוש מונטה קרלו (Monte Carlo Tree Search) כך שרשת המדיניות שימשה ככלי לצמצום החיפוש למהלכים עם הסתברות גבוהה ורשת הערך בכדי להעריך מיקומים בעץ עצמו [56]. בשנת 2016 פורסם אלגוריתם אלפא-גו לי (AlphaGo Lee) אשר משתמש בגישה דומה לאלפא-גו פאן ואשר הצליח לנצח את Lee Sedol שניצח 18 תחרויות בינלאומיות [8].

אף על פי שאלגוריתם אלפא-גו זכה להצלחה גדולה יש לו מספר חסרונות אשר מקשים על הכללתו עבור משחקים אחרים. ראשית, האלגוריתם זקוק למומחים אנושיים בכדי ללמוד דבר אשר לעיתים אינו אמין או פשוט בלתי ניתן להשגה. שנית, האלגוריתם היה מורכב מאוד וכלל שתי רשתות נוירונים ואלגוריתם חיפוש עצים מורכב מה שתרם לזמן אימון ארוך מאוד של 60 ימים. בכדי להתמודד עם חסרונות אלו, ובכדי להותיר אפשרות של הכללת האלגוריתם לטובת פתרון משחקים נוספים חוקרי אלפא-גו פיתחו את אלגוריתם אלפא-גו אפס (AlphaGo Zero) [8]. אלגוריתם זה הוא למעשה יישום מלא של למידה עמוקה באמצעות חיזוקים ולכן איננו דורש ידע אנושי כלשהו על המשחק (ומכאן שמו). האלגוריתם כולל רשת נוירונים עמוקה בודדת אשר מקבלת כקלט ייצוג של הלוח ומיקומי הכלים עליו וכן עץ חיפוש מסוג מונטה קרלו אשר בוחר פעולות מתוך הפעולות שהרשת מחשבת. החוקרים אימנו את הרשת במשך שלושה ימים והשוו את ביצועיה ביחס לאלגוריתם המקורי. תוצאות ההשוואה הראו באופן חד משמעי כי האלגוריתם החדש הוא בעל ביצועים טובים וכן מהירות ההתכנסות שלו קטנה פי עשרים. יתרה מכך, החוקרים בחנו את שני האלגוריתמים כאשר הם משחקים אחד מול השני והראו שאלפא-גו אפס מנצח בכל משחק.

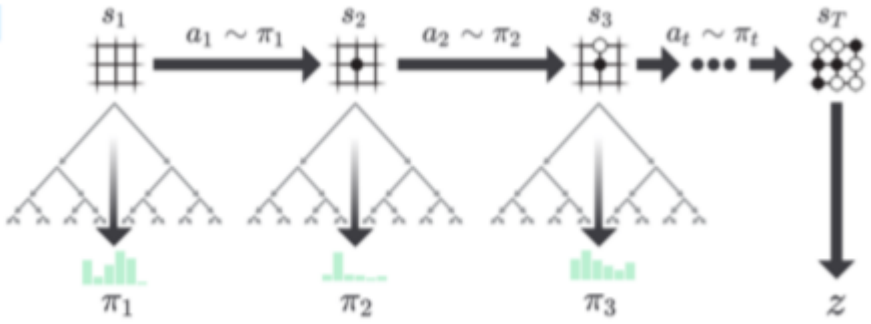
אלפא-גו אפס משתמש ברשת נוירונים עמוקה בודדת f_θ עם הפרמטרים θ. הרשת מקבלת כקלט את מצב הלוח כפי שהוא לאורך שמונת המהלכים הקודמים, s, ומחשבת וקטור (p, v) כך ש- p מייצג את ההסתברות לבחירת מהלך עבור מצב s ו- v הוא סקאלר המעריך את הסיכוי של השחקן הנוכחי לנצח את המשחק ממצב s. רשת זו היא למעשה מיזוג של שתי הרשתות (רשת המדיניות ורשת הערך) של אלפא-גו פאן ואלפא-דו לי אל תוך רשת אחת. כאמור, הקלט לרשת כולל את מצב לוח המשחק בשמונת המצבים האחרונים עבור כל

שחקן וכמו כן את צבע האבן של השחקן הנוכחי, והיות ולוח משחק גו הוא בגודל של 19 x 19 משבצות, הרי שהקלט לרשת הוא מטריצה בגודל 19 x 19 x 17. ישנה חשיבות גדולה מאוד בכך שהקלט יכיל את שמונת המצבים האחרונים של כל שחקן שכן לא ניתן במשחק גו להבין את התמונה הכללית מתוך מבט במיקום האבנים הלוח ברגע זמן בודד (Not Fully Observable Environment). הקלט לרשת עובר דרך סדרה של 19 שכבות קונבולוציה. בכל שכבה מופעל פילטר בגודל 3 x 3 עם אורך צעד של אחד (Stride) ולאחר מכן דרך יחידת חישוב לא לינארית כמקובל ברשתות מסוג זה. בתום סדרת השכבות הקונבולציוניות המידע מתפצל לשני ראשים בכדי לספק את שני ערכי המוצא p ו- v. ראש אחד מופנה לעבר שני פילטרים קונבולציונים מגודל 1 x 1 עם אורך צעד אחד ולאחר מכן יחידה לא לינארית וראש אחר מופנה לעבר פילטר קונבולוציה בודד מגודל 1 x 1 עם אורך צעד אחד ולאחר מכן יחידה לא לינארית. איור 4.1 מדגים את פעולת אימון הרשת, קרי, עבור מצב s_t מתקבל קלט המכיל את שמונת המהלכים של כל שחקן, הקלט מוזן אל תוך סדרה של שכבות קונבולוציוניות ובסופם מתפצל לשני ראשנים המייצרים את הערכים p ו-v כפי שהוסבר.



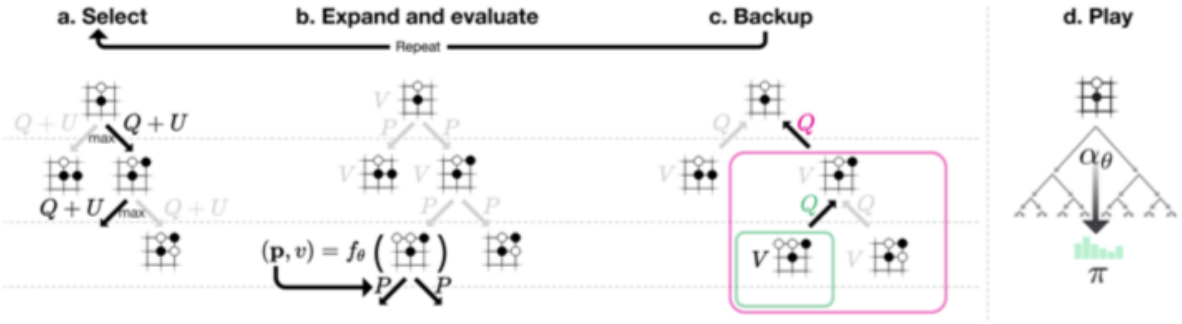
איור 4.1 – תהליך אימון רשת הנוירונים של אלפא-גו אפס. עבור מצב s_t מתקבל קלט המכיל את שמונת המהלכים של כל שחקן, הקלט מוזן אל תוך סדרה של שכבות קונבולוציוניות ובסופם מתפצל לשני ראשנים המייצרים את הערכים p ו- v, כאשר p מייצג את ההסתברות לבחירת מהלך עבור מצב s_t ו- v הוא סקאלר המעריך את הסיכוי של השחקן הנוכחי לנצח את המשחק ממצב s_t [8].

רשת הנוירונים של אלפא-גו אפס מאומנת על ידי טכניקה הקרויה *משחק-עצמי* יחד עם למידה באמצעות חיזוקים. על פי טכניקה זו סוכן למידה באמצעות חיזוקים משחק את שני השחקנים במשחק בשימוש אותה פונקציית ערך – כלומר הסוכן משחק נגד עצמו [1]. איור 4.2 מתאר כיצד אלפא-גו אפס מיישם משחק עצמי עבור מצב s, מפעילים עץ חיפוש מונטה קרלו (Monte-Carlo tree search) אשר מונחה על ידי רשת הנוירונים f_θ. הפלט של עץ החיפוש הוא אוסף הסתברויות לכל מהלך משחק אפשרי. החוקרים הראו כי הסתברויות אלו לעיתים קרובות מייצרות מהלכים טובים יותר מאלו אשר מחושבים על ידי רשת הנוירונים בלבד. הרעיון המרכזי של אלפא-גו אפס הוא בעצם להשתמש בטכניקה זו יחד עם משחק עצמי כחלק מתוך שיטת חיזרור המדיניות (Policy Iteration).



איור 4.2 – משחק עצמי עם למידה באמצעות חיזוקים כי שהוא ממומש באלפא-גו אפס. האלגוריתם משחק משחק כנגד עצמו. בכל שלב של המשחק s_t, מפעילים עץ חיפוש מונטה קרלו בשימוש יחד עם רשת הנוירונים $s1, \ldots, s_T$ העדכנית ביותר. מהלך המשחק הבא נבחר על פי ההסתברויות המופקות על ידי העץ. המצב הסופי של המשחק s_T מקבל ניקוד על פי חוקי משחק גו ומחושב המנצח z [8].

עץ החיפוש מונטה קרלו משתמש ברשת הנוירונים f_θ בכדי להנחות את החיפוש. כל ענף של העץ (s, a) כולל שלושה מרכיבים: ההסתברות $P(s,a)$, מונה ביקורים $N(s,a)$ וערך $Q(s,a)$. איור 4.3 מדגים את פעולת החיפוש על העץ: כל חיפוש בעץ מתחיל מהשורש ובצורה איטרטיבית בוחרים את המהלך אשר ממקסם את הערך $Q(s,a) + \frac{P(s,a)}{1+N(s,a)}$ עד אשר מגיעים לעלה. המצב המאוחסן בעלה מופעל על הרשת בכדי לחשב את וקטור הפלט (p, v) ומעדכנים את ערכי ה- Q לאורך המסלול בעץ. הפלט של העץ משמש בכדי לבצע את מהלך המשחק הבא.



איור 4.3 – חיפוש בעץ מונטה קרלו באלפא-גו אפס. החיפוש מבוצע במספר שלבים: בשלב הראשון מחפשים את העלה המתאים על ידי הילוך על העץ ובחירת הצלעות אשר ממקסמות את הערך $Q(s,a) + \frac{P(s,a)}{1+N(s,a)}$. בשלב השני מפעילים את רשת הנוירונים על מצב בלוח במצוי בעלה הנבחר ומחשבים וקטור (p, v). בשלב השלישי מעדכנים את ערכי Q בהתאם לאורך המסלול הנבחר בעץ [8].

כאמור, רשת הנוירונים מאומנת על ידי משחק עצמי של למידה באמצעות חיזוקים אשר משתמשת בעץ חיפוש מונטה קרלו בכדי לשחק כל מהלך. בתחילה הרשת מאותחלת עם משקולות אקראיים. בכל סיבוב משוחק משחק של משחק-עצמי כפי שהוסבר קודם. המשחק מסתיים במקרים בו שחקן אחד ניצח או שאורך המשחק עובר חסם עליון קבוע. כל משחק מקבל ציון 1- או 1. כל מהלכי המשחק כולל המנצח נשמרים במסד נתונים אשר בתורו משמש בכדי לאמן את הרשת לפני המשחק הבא.

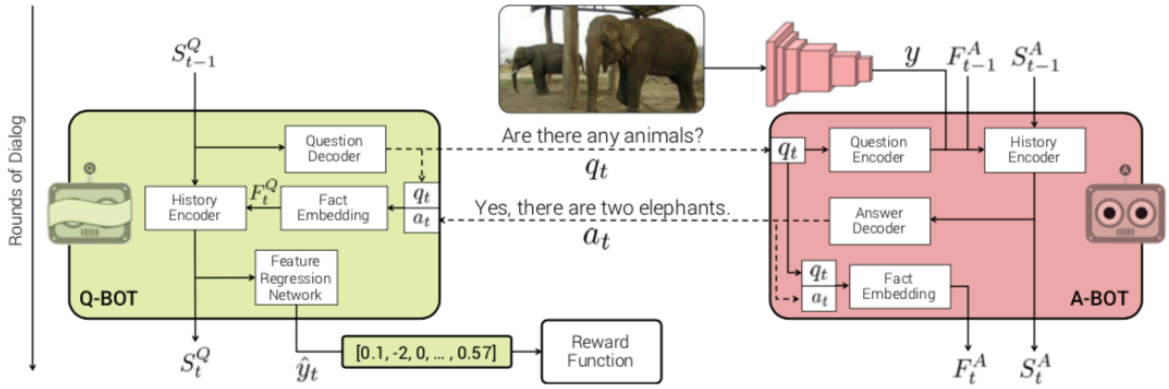
החוקרים הריצו את האלגוריתם המתואר למשך כשלושה ימים. במהלך האימון שוחקו 4.9 מיליון משחקים עצמיים. אלפא-גו אפס עקף את ביצועיו של אלפא-גו לי תוך 36 שעות בלבד (יש לזכור שאלפא-גו לי אומן במשך מספר חודשים!). אלפא-גו אפס הורץ על מכונה בודדת עם 4 יחידות עיבוד טנסוריות (Tensor Processing Units) לעומת אלפא-גו לי אשר הורץ בצורה מבוזרת על מספר רב של מכונות ואשר דרש 48 יחידות עיבוד טנסוריות. אלפא-גו אפס הביס את אלפא-גו לי 100 – 0.

5.2 מערכות דו-שיח ויזואליות

מערכות דו-שיח ויזואליות (Visual Dialog Systems) הן מערכות דו-שיח אשר מתנהלות בהקשר לתמונה, כלומר, במערכות אלו הסוכנים מתקשרים ביניהם בהתאם להבנה שלהם את תוכן תמונה. האלגוריתמים הקיימים אשר עוסקים בתחום זה משתמשים בטכניקות של למידה בהשגחה מה שיצר מוגבלות ביכולות ללמוד ולפתח שיחות אנושיות אמתיות ביחס לתמונה. קבוצת חוקרים [55] פיתחה אלגוריתם למידה עמוקה באמצעות חיזוקים אשר מאמן שני סוכנים בכדי שיוכלו לנהל ביניהם דו-שיח שיתופי במטרה להבין תוכן של תמונה. החוקרים הגדירו את הבעיה בצורת של משחק בין שני סוכנים, קרי, סוכן-שואל וסוכן-עונה. בתחילת כל משחק הסוכן-שואל מקבל תיאור מילולי של תמונה (שלא ראה) בן משפט אחד בלבד והסוכן-עונה מקבל את התמונה עצמה. הסוכן-שואל מתקשר בשפה טבעית עם הסוכן-עונה לגבי תוכן התמונה, כאשר המטרה בסופו של דבר שהסוכן-שואל יזהה את התמונה המדוברת מתוך אוסף תמונות בהתאם להבנתו את תוכן התמונה. תיאור הבעיה באופן שכזה מרמז שעל הסוכנים ללמוד מודלים המבוססים על שיתוף פעולה ביניהם. בכל שלב של הדיאלוג בין הסוכנים, הסוכן השואל מאזין לתשובה המתקבלת מהסוכן העונה ומעדכן את המודל שלו בכדי לבצע ניבוי של ייצוג התמונה שהוא לא ראה. על פי איכות הניבוי הסביבה מעניקה תגמול חיובי או שלילי לסוכן השואל, מכאן שהמטרה של הסוכן השואל ושל הסוכן העונה היא לתקשר ביניהם באופן כזה כך שהתגמול המצטבר יהיה מירבי.

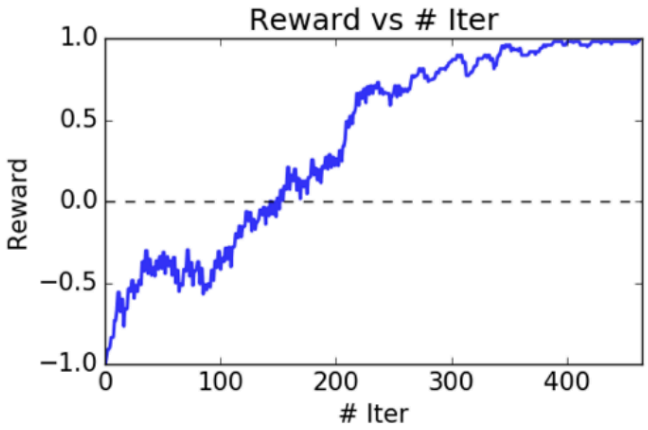
החוקרים מציגים אלגוריתם פתרון לבעיה המבוסס על למידה עמוקה באמצעות חיזוקים. אף על פי שתיאור הבעיה כולל שני סוכנים נפרדים, היות והמשחק הוא משחק שיתוף פעולה מלא ניתן למדל את שני הסוכנים כסוכן אחד אשר מכיל שני יחידות – אחת לסוכן השואל ואחת לסוכן העונה. מרחב הפעולות משותף לשני הסוכנים והוא מורכב מכל הרצפים שניתנים להרכבה תחת מילון נתון V. מרחב המצבים שונה עבור כל סוכן; בעוד שהסוכן העונה חשוף לתמונה I, לתיאור המילולי שלה c ודיאלוג המתנהל בין הסוכנים $q_1, a_1, \dots, q_n, a_n$, הסוכן העונה חשוף רק לתיאור המילולי ודיאלוג. על כן, מרחב המצבים של הסוכן השואל הינו $[\mathrm{c}, q_1, a_1, \dots, q_n, a_n]$ ומרחב המצבים של הסוכן העונה הינו $[I, \mathrm{c}, q_1, a_1, \dots, q_n, a_n]$. לכל סוכן ישנה מדינות נפרדת אשר ממודלת על ידי רשת נוירונים עמוקה עם אוסף פרמטרים שונה, θ_Q עבור המדיניות של הסוכן השואל ו- θ_{A} עבור המדיניות של הסוכן העונה. בנוסף, עבור הסוכן השואל ישנה רשת נוספת עם פרמטרים θ_f עבור ייצוג התמונה מתוך התשובות שהסוכן העונה מספק. איור 4.4 מתאר את המבנה הכללי של הרשתות עבור המדיניות של הסוכן השואל ועבור המדיניות של הסוכן העונה. הרשת של המדיניות של הסוכן השואל (מתוארת במסגרת הירוקה) מורכבת מארבעה חלקים; *מקודד עובדות (Fact Encoder)* – יחידת LSTM אשר תפקידה לעבד זוגות של שאלה-תשובה שהסוכן שואל מקבל. *מקודד מצבים (State Encoder)* – יחידת LSTM אשר מקבלת עובדה מקודדת (מהיחידה הקודמת) בכל יחידת זמן ומקודדת אותה יחד עם עובדות ישנות (מהסיבובים הקודמים). *מפענח שאלות (Question Decoder)* – יחידת LSTM אשר מקבלת את התוצר של היחידה הקודמת ומייצאת שאלה חדשה עבור הסיבוב הבא

מתוך מאגר מילים. *רשת רגרסיה למאפיינים (Feature Regression Network)* – זוהי שכבה מקושרת מלאה אשר מייצרת ייצוג לתמונה מתוך ניבוי המתבצע מעל המצב הנוכחי. לסוכן העונה יש מבנה דומה לזה של הסוכן השואל (מתואר במסגרת האדומה); *מפענח שאלות (Question Encoder)* – יחידת LSTM אשר מקבלת שאלה מהסוכן השואל ומקודדת אותה. *מקודד עובדות (Fact Encoder)* – בדומה לסוכן השואל, הסוכן העונה מקודד גם הוא באמצעות יחידת LSTM זוגות של שאלה-תשובה. *מקודד מצבים (State Encoder)* – יחידת LSTM אשר מקודדת בכל שלב את השאלה המקודדת, את מאפייני התמונה וכן את הקידוד מהסיבוב הקודם. *מקודד תשובות (Answer Decoder)* – יחידת LSTM אשר מקבלת את המצב המקודד מהיחידה הקודמת ומייצרת תשובה מתאימה מתוך מאגר מילים.



איור 4.4 – רשתות מדיניות עבור הסוכן השואל (מסגרת ירוקה) והסוכן העונה (מסגרת אדומה). בכל שלב של הדיאלוג בין הסוכנים הרובוט השואל מייצר שאלה באמצעות יחידת מפענח השאלות אשר מותנה על ידי המצב המקודד. הסוכן העונה מקודד את השאלה, מעדכן את המצב המקודד שלו ומייצר תשובה. שני הסוכנים מקודדים את הזוג שאלה-תשובה שנוצר לעובדה. הסוכן השואל מעדכן את המצב המקודד שלו עם העובדה החדשה, מייצר ייצוג של התמונה באמצעות ניבוי הנשען על המצב החדש ומקבל תגמול מן הסביבה [55].

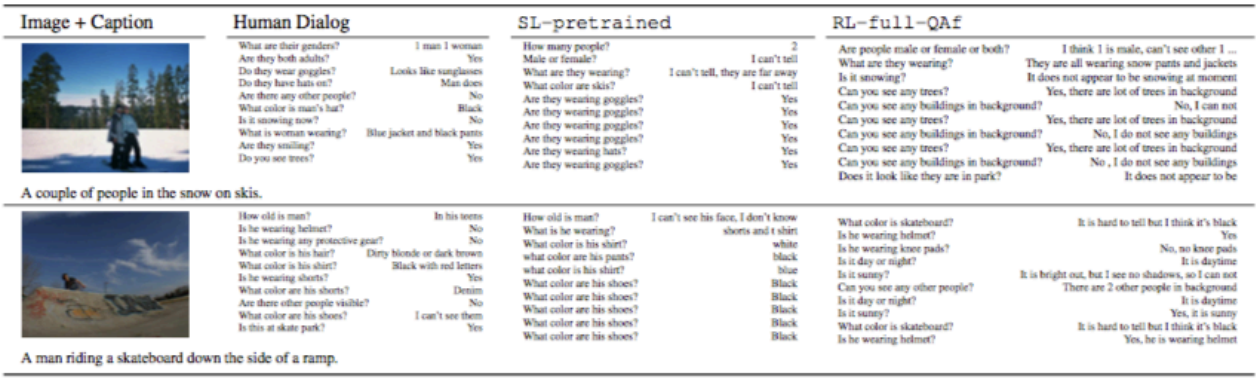
בכדי לאמן את הסוכנים האלו, החוקרים השתמשו באלגוריתם REINFORCE (גראדיינט המדיניות) בכדי לעדכן את הפרמטרים של המדיניות $(\theta_Q, \theta_A, \theta_f)$ בתגובה לתגמולים שניתנים על ידי הסביבה. לצורך בדיקת האלגוריתם החוקרים בחנו תחילה את האלגוריתם עבור סביבה סינטטית. בסביבה זו, התמונות האפשריות מורכבות משלושה מאפיינים, קרי, צורה צבע וסגנון ולכל מאפיין יש 4 ערכים אפשריים כך שבסך הכל ישנם 64 תמונות אפשריות. הסוכן העונה מקבל גישה ישירה לייצוג של התמונה והסוכן השואל צריך לנבא שני מאפיינים מתוך השלושה של התמונה בסדר נתון. איור 4.5 מתאר את היחס בין מספר ההרצות של האלגוריתם לבין התגמול המתקבל כאשר האלגוריתם נוסה בסביבה הסינטטית. ניתן לראות שהסוכנים למדו בקלות את המדיניות האופטימלית.



איור 4.5 – תיאור היחס בין מספר ההרצות של האלגוריתם לבין התגמול המתקבל כאשר האלגוריתם רץ מעל סביבה סינטטית [55].

לאחר שהחוקרים השתכנעו כי האלגוריתם אכן מצליח לייצר דיאלוג נכון הם פנו לבחינה שלו אל מול מאגר תמונות אמיתי. לצורך הניסוי החוקרים השתמשו במאגר התמונות והדיאלוגים VisDial אשר מכיל דיאלוגים אנושיים (עשרה זוגות של שאלות ותשובות לגבי תוכן תמונה) עבור 68 אלף תמונות, כלומר 680 אלף זוגות של שאלות ותשובות. החוקרים בחרו לאמן את הסוכנים בשני שלבים: אימון מקדים מושגח (Supervised Pretraining) ולימוד על פי תוכנית (Curriculum Learning). בשלב הראשון מאמנים את שני הסוכנים באופן מושגח (Supervised) על חלק ממאגר הזוגות של VisDial. הסוכן השואל מאומן לייצר שאלות המשך מתוך המאגר והסוכן העונה מאומן לייצר תגובות מתאימות מתוך המאגר. שלב זה מבטיח שהסוכנים יכולים לזהות בכלליות אובייקטים וסצנות ולדבר באנגלית. בשלב השני מבצעים מעבר הדרגתי לעבר למידה באמצעות חיזוקים על פי תוכנית לימודים, קרי, עבור K סיבובים ראשונים של הדיאלוג מאמנים באופן מושגח וב- K – 10 הסיבובים הנותרים משתמשים בגראדיינט המדיניות. השימוש בתוכנית לימודים מבטיח שהמדיניות של הסוכנים לא יסטו במקרה שנוצרה שאלה או תשובה לא נכונה. באופן כללי, החוקרים מריצים את השלב הראשון עבור 15 דיאלוגים, ולאחר מכן מקטינים את K באחד בכל דיאלוג, עד אשר בדיאלוג ה- 25 משתמשים שלימדה עמוקה באמצעות חיזוקים בלבד.

בכדי לבחון את היעילות של האלגוריתם החוקרים השתמשו בחלק מהמאגר VisDial שלא נראה על ידי הסוכנים במהלך האימון (Test-set). עבור כל תמונה שלא נראתה מריצים את הסוכנים 10 סיבובי דיאלוג. בתום הדיאלוג מחשבים את המרחק של התמונה שהסוכן השואל מייצר ביחס לכל התמונות במאגר. תוצאות הניסוי מראות שהשימוש בלמידה באמצעות חיזוקים משפרת את זיהוי התמונות (לעומת שימוש רק בטכניקות של למידה מונחית). כמו כן, החוקרים מראים שהשימוש בלמידה עמוקה באמצעות חיזוקים מייצרת דיאלוג יותר אינפורמטיבי. איור 4.6 מציג השוואה בין מספר דוגמאות נבחרות של דיאלוגים- אנושי, למידה בהשגחה ולמידה באמצעות חיזוקים. ניתן לראות בבירור כי הדיאלוגים שנוצרים באמצעות האלגוריתם של למידה באמצעות חיזוקים מגוונים יותר, נוטים פחות לחזור על עצמם ומתארים את התמונה בבירור יותר.



איור 4.6 – מספר דוגמאות נבחרות של דיאלוג בין סוכן שואל וסוכן עונה עבור אלגוריתמים של למידה בהשגחה ולמידה באמצעות חיזוקים. הדיאלוגים שנוצרים באמצעות אלגוריתם למידה באמצעות חיזוקים מגוונים יותר, נוטים פחות לחזור על עצמם ומתארים את התמונה בבירור יותר [55].

6. אתגרים ומבט לעתיד

למידה עמוקה באמצעות חיזוקים כורכת את הכוח האדיר של רשתות נוירונים עמוקות יחד עם העיקרון הבסיסי של למידה מתוך ניסוי וטעייה. שילוב זה הוכח כיעיל וככזה אשר יכול לפתור מגוון רחב של בעיות בכל תחומי החיים. אכן, למידה עמוקה באמצעות חיזוקים יכולה להיחשב כצעד הראשון בדרך ליצירתה של מסגרת תאורטית של בינה מלאכותית כללית, כלומר, כזו שיכולה לבצע כל משימה שאדם יכול לבצע [15]. עם זאת, ישנם מספר אתגרים אשר דורשים מענה בכדי לממש את מלוא הפוטנציאל של פרדיגמה זו. בפרק זה נדון באתגרים העומדים בפני למידה עמוקה באמצעות חיזוקים ועתיד המחקר שלה.

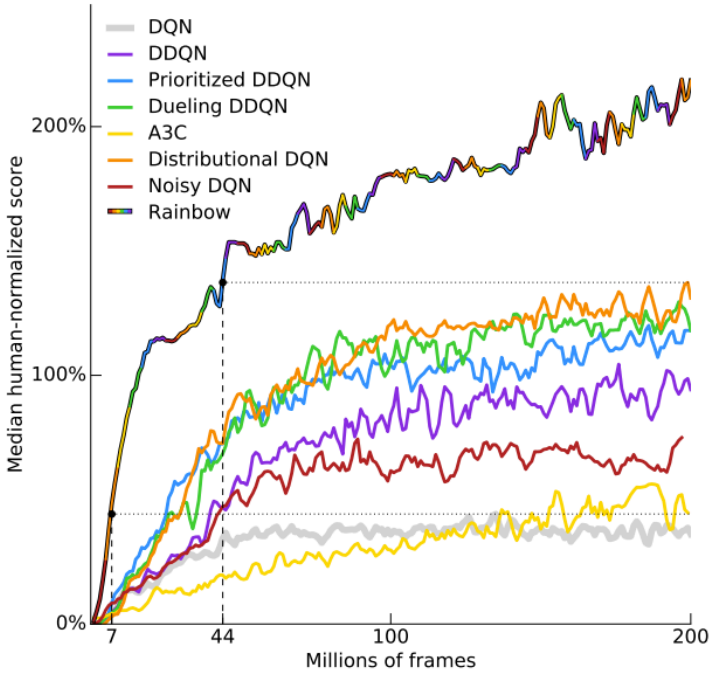
6.1 אתגרים

מטרת העל של למידה באמצעות חיזוקים או למידה עמוקה באמצעות חיזוקים היא לתת מענה למגוון רחב של בעיות בצורה יעילה. המחקר בתחום נמשך לאורך כארבע וחצי עשורים ובמהלכו התגלו לא מעט אתגרים הנוצרים מיישום העיקרון של למידה מתוך ניסוי וטעייה. בחלק זה נדון בחמישה אתגרים מרכזיים העומדים היום בפני החוקרים של למידה עמוקה באמצעות חיזוקים.

6.1.1 יעילות הדגימה

יעילות הדגימה מתייחסת ליכולת של אלגוריתם להתכנס לפתרון איכותי תוך כדי דגימה קטנה כלל האפשר של הסביבה. כך, אף על פי שחלה התקדמות גדולה במחקר של אלגוריתמים מבוססי למידה עמוקה באמצעות חיזוקים, עדיין אלגוריתמים אלו איטיים והם זקוקים לכמה עשרות מיליוני דגימות בכדי להתכנס לפתרון איכותי. בעיית יעילות הדגימה מחריפה כאשר מדובר בסביבות בהן דגימות הן דבר יקר או קשה להשגה.

בפרק 2 הצגנו מחקר [29] בשם Rainbow משנת 2017 אשר עשה השוואה יסודית בין ביצועי אלגוריתם DQN וכל ביצועי האלגוריתמים שנגזרו ממנו עד אותו הרגע, קרי: Prioritized DDQN ,Double-DQN, Dueling DDQN A3C, Distributional DQN ו- Noisy DQN. כמו כן, החוקרים יצרו אלגוריתם חדש (Rainbow) אשר משלב בתוכו את העקרונות של כל אחד מהאלגוריתם שנגזרו מ- DQN. ההשוואה נעשתה מעל פלטפורמת משחקי אטארי 2600. החוקרים בחנו את יעילות הדגימה של כל אחד מן האלגוריתמים על ידי הפעלת כל אלגוריתם על כל אחד מ- 57 משחקי אטארי שבהשוואה, כאשר הציון של כל אלגוריתם נורמל (Normalized) כך שציון של שחקן אנושי הוא 100%.



איור 5.1 – מתאר את היחס בין מספר הדגימות שביצע כל אלגוריתם לבין ביצועי שחקן אנושי חציוני על פני 57 משחקי אטארי [29].

מאיור 5.1 ניתן לראות בברור כי האלגוריתם הטוב ביותר מבחינת יעילות הדגימה הוא Rainbow, אשר נזקק ל- 18 מיליון דגימות כדי להגיע לרמת משחק של אדם. אף על פי ש- 18 מיליון דגימות הם תוצאה טובה בהשוואה לביצועיהם של האלגוריתמים האחרים (אשר נעים בין 70 מיליון ליותר מ- 200 מיליון), עדיין מדובר בכ- 83 שעות משחק אדם (עם 60 דגימות בשנייה כקצב ריענון המסך) שהם הרבה זמן בשביל בני אדם אשר לומדים לשחק טוב משחקי אטארי אחרי מספר דקות.

בעיית יעילות הדגימה באה לידי ביטוי גם בפלטפורמות נוספות, כך שני מחקרים אחרים שנסקרו בפרק 2: Trust Region Policy Optimization [34] ו- Deep Deterministic Policy Gradient [33], הראו שנדרשים כמה עשרות מיליוני צעדים בפלטפורמת MuJoCo בכדי להתכנס לפתרון איכותי. מחקר נוסף של קבוצת החוקרים DeepMind משנת 2017 [57] הראה שבאמצעות שימוש ב- 64 מעבדים ניתן לאמן סוכן פארקור (Parkour) ב 100 שעות בפלטפורמת MuJoCo. זו כמובן תוצאה טובה, אך עדיין נמצאת בכמה סדרי גודל מעל הרמה הפרקטית של יעילות דגימה אצל בני אדם.

6.1.2 <u>פונקציית התגמול</u>

בבסיסה של למידה עמוקה באמצעות חיזוקים עומדת פונקציית התגמול אשר מעניקה תגמול חיובי כאשר הסוכן מבצע פעולה או סדרה של פעולות אשר מובילות לתוצאה רצויה או תגמול שלילי כאשר הם אינם. מכאן, בכדי שלמידה באמצעות חיזוקים תעשה את הדבר הנכון, פונקציית התגמול חייבת לשקף במדויק את התוצאה הרצויה. בסביבות עם ידע מלא כדוגמת פלטפורמת משחקי אטארי התכנון של פונקציית התגמול היא תהליך קל, שכן כל הפרטים ידועים והמטרה ברורה. אך, בסביבות בהם קיים ידע חלקי הדבר הופך קשה יותר ומתבטא בין השאר במה שמכונה תגמול דליל (Sparse Reward). תגמול דליל או תגמול אופק-ארוך (Long Horizon Reward) הוא מצב בו הסוכן מקבל תגמול חיובי או שלילי רק כאשר הוא מגיע

למצב סופי, לדוגמא, רובוט אשר צריך להרים חפץ משולחן צריך לבצע סדרה של פעולות עד אשר הוא יאחז בחפץ, אבל התגמול יתקבל רק ברגע האחיזה. התנהלות זו הופכת את הלמידה לארוכה ולעיתים לכזאת שאינה מתכנסת. פתרון אפשרי לבעיה זו היא עיצוב תגמולים (Reward Shaping) אשר מעניק תגמולים גם כאשר המדיניות עדיין לא מצאה פתרון לבעיה. פתרון זה אומנם פותר את הבעיה והופך את הלימוד לקל יותר אבל מוביל לתופעה אחרת בשם Reward Hacking [58]. היות ולמידה באמצעות חיזוקים מבוססת על מקסום התגמולים לאורך זמן, תכנון לקוי או מצבים לא צפויים בסביבה עלולים להוביל את הסוכן להעדיף את תגמולי הביניים על פני התגמול הסופי ובכך לא ללמוד מדיניות נכונה.

6.1.3 <u>הימלטות מנקודות קיצון מקומיות</u>

תופעת ה- Reward Hacking שהוזכרה בסעיף הקודם היא מקרה פרטי של בעיה רחבה יותר בלמידה באמצעות חיזוקים, קרי, בעיית ההימלטות מנקודות קיצון מקומיות (Escaping Local Optima). בעיה זו היא תולדה של איזון לא נכון בין חקר הסביבה לבין ניצול הידע שנצבר (Exploration vs. Exploitation). מצד אחד אם המדיניות הנוכחית מבצעת חקר יתר של הסביבה אז הסוכן עלול לקבל מידע לא שמיש (לדוגמא סביבות גדולות מאוד והסוכן חוקר אזורים שאינם קשורים ללב הבעיה) ובעצם לא ללמוד שום דבר חדש לגבי הבעיה, ומהצד השני שימוש יתר במידע שנצבר עלול להוביל את הסוכן לביצועים לא אופטימליים, קרי, היקלעות לנקודות קיצון מקומיות. כבר מימי ראשית המחקר של למידה באמצעות חיזוקים הוצעו מספר פתרונות אפשריים לבעיה זו [1] אך אף אחד מהם לא עובד בצורה עקבית בכל הסביבות. האתגר שעומד בפני המחקר העכשווי הוא לנסות למצוא שיטה לאזן בין חקירת הסיבה לבין ניצול ידע נצבר באופן כזה שיהיה ניתן ליישם אותה בכל סביבה.

6.1.4 <u>הכללה</u>

כפי שהוסבר בפרק הראשון, ביצועיו של אלגוריתם למידה מתאפיינים בין השאר ביכולת שלו להכליל (Generalize) את המודל הנלמד עם מצבים שלא נלמדו. אלגוריתם DQN יכול לפתור מספר רב של משחקי אטארי ברמה של שחקן אנושי, אך הוא עושה זאת על ידי למידת מטרה אחת, כלומר לשחק טוב במשחק אחד. המודל הסופי שנלמד על ידי אלגוריתם DQN עבור משחק ספציפי לא מכליל את עצמו טוב עבור משחקי אטארי אחרים בגלל שהוא לא התאמן עבורם. ניתן אמנם לכונן את המודל הנלמד כך שיתאים למשחק אחר, אבל אין שום הבטחה שהוא אכן יעבוד בצורה טובה. יתרה מכך, אין שום אלגוריתם ידוע של למידה עמוקה באמצעות חיזוקים אשר מאפשר למידה איכותית של אוסף משימות מגוון. שני חוקרים [59] בחנו את יכולת ההכללה של אלגוריתמים של למידה עמוקה באמצעות חיזוקים עבור משחקי קומבינטוריקה לשני שחקנים שיש להם פתרון אנליטי למשחק אופטימלי. באחד הניסויים שהחוקרים ערכו הם קיבעו את ההתנהגות של שחקן אחד (על פי האסטרטגיה האופטימלית) ואימנו מודל למידה עמוקה באמצעות חיזוקים עבור השחקן השני. בצורה זו, ההתנהגות של השחקן הראשון בעצם מהווה חלק מן הסביבה הכללית. ואכן, החוקרים הראו שהשחקן השני מגיע לרמת ביצועים גבוהה מאוד. אך כאשר החוקרים בדקו את ביצועי המודל כאשר השחקן הראשון השתמש באסטרטגיה לא אופטימלית, הם נכחו לגלות רמת ביצועים נמוכה מאוד, כלומר המודל לא הכליל את עצמו אל מול משחק עם שחקן לא אופטימלי. קבוצת חוקרים נוספת [60] הגיעה למסקנה דומה כאשר חקרה הכללה של מודל הלומד לשחק בטורניר לייזר זוגי. הסוכנים אומנו יחד חמש פעמים כך שבכל פעם נעשה שימוש בערך אקראי. החוקרים אכן ראו שהמודל נלמד היטב והשחקנים משחקים אחד כנגד השני ברמה גבוהה. בשלב השני, החוקרים בחנו משחקים של

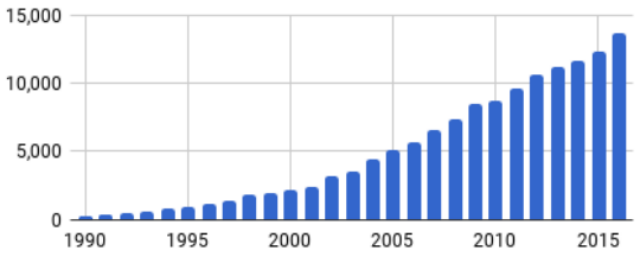
שחקנים אשר נלקחו מסיבובי אימון שונים, כלומר כל שחקן נלקח ממודל עם ערך אקראי אחר. החוקרים הראו שבכל המשחקים השחקנים שיחקו ברמה נמוכה, כלומר, המודלים השונים לא הכלילו את עצמם אפילו כאשר כל מה שהשתנה הוא ערך אקראי בודד.

6.1.5 יכולת שחזור פתרונות

בכדי לשמר קצב פיתוח מהיר במחקר של למידה עמוקה באמצעות חיזוקים, חשוב שיהיה ניתן לשחזר ולהשוות בקלות תוצאות של מחקרים קודמים. אך, יכולת השחזור של פתרונות קיימים הוא לעיתים רחוקות תהליך פשוט והספרות המחקרית מדווחת על מגוון רחב של תוצאות מעל אותם האלגוריתמים. יכולת שיחזור יכולה להיות מושפעת מגורמים חיצוניים כדוגמת Hyper-Parameters ומגורמים פנימיים כמו השפעות של ערכים אקראיים (בדרך כלל המשקולות של רשתות נוירונים עמוקות מאותחלות באופן אקראי) [6].

6.2 מבט לעתיד

בעשור האחרון המחקר בתחום של למידה באמצעות חיזוקים הפך לפופולארי מאוד בעיקר הודות להצלחות של שילוב טכניקות של למידה עמוקה. איור 5.2 מתאר את מספר המאמרים בתחום של למידה באמצעות חיזוקים לפי שנת פרסום. מן הגרף ניתן לראות כי מספר המאמרים שמתפרסמים בנושא הולך וגדל בצורה עקבית על פני שניים וחצי העשורים האחרונים [6]. יש להניח אם כן, כי בשנים האחרונות המגמה תמשך והמחקר בתחום יזכה לפריצות דרך נוספות. בחלק זה נדון במספר כיווני מחקר אפשריים במטרה לענות על האתגרים שתיארנו בחלק הקודם.



איור 5.2 – גרף המתאר את מספר המאמרים בתחום של למידה באמצעות חיזוקים לפי שנות הפרסום [6].

6.2.1 התפתחות החומרה

אחת הסיבות לצבירת הפופולאריות של המחקר בתחום של למידה עמוקה באמצעות חיזוקים בשנים האחרונות קשורה להתפתחות המואצת של החומרה באותה התקופה. באופן כללי, ככל שהיכולות של החומרה הולכות וגדלות כך ניתן לפתור בעיות גדולות יותר וכן להתמודד עם בעיות בהם סיבוכיות החישוב גדולה. אין בכך לטעון כי הגדלת יכולות החומרה תפתורנה את כל הבעיות שעומדות בפני המחקר העכשווי, אך היא יכולה לסייע מאוד בקידומו. שכן, ככל שניתן להריץ אלגוריתמים בצורה מהירה יותר, כך בעיות כמו יעילות הדגימה חשובות פחות.

6.2.2 למידה באמצעות חיזוקים ככיוון עדין

בשנתיים האחרונות החלו להתפרסם מחקרים בהם החוקרים השתמשו בלמידה עמוקה באמצעות חיזוקים ככלי לכיוון עדין עבור אלגוריתם אחר. הרעיון הזה יעיל מכיוון שמתחילים עם אלגוריתם מהיר יותר אך חלש יותר מבחינת ביצועים ואז משתמשים בידע שנצבר ומפעילים מעליו אלגוריתם של למידה עמוקה

באמצעות חיזוקים ובכך מאיצים את החלק הראשוני של הלמידה. כך למשל קבוצת החוקרים של AlphaGo [8] השתמשה תחילה בלמידה מונחית ואחר כך בלמידה עמוקה באמצעות חיזוקים כדי לפתור את המשחק Go. הצלחה זו חזרה על עצמה במחקר אחר [61] בו החוקרים פיתחו את אלגוריתם Sequence Tutor במטרה לייצר שיטה כללית המשפרת את המבנה והאיכות של סדרות הנוצרות על ידי רשת נוירונים עמוקה מסוג Recurrent.

6.2.3 למידת פונקציית התגמול

הרעיון המרכזי של למידה חישובית הוא שניתן להשתמש במידע בכדי ללמוד דברים באופן טוב יותר מאשר בני אדם. כפי שנטען בחלק הקודם תכנון ועיצוב של פונקציית תגמול הוא דבר קשה ולכן לא מן הנמנע שלמידה של פונקציית התגמול עצמה באמצעות שיטות של למידה חישובית יניב תוצאות טובות יותר מאשר פונקציות קבועות המהונדסות על ידי בני אדם. מספר מחקרים בנושא פורסמו כבר לפני עשרים שנה, אך לאחרונה התחום זוכה לעדנה מחודשת עם הופעתם של מחקרים [62] [63] [64] אשר משלבים טכניקות של למידה עמוקה בכדי ללמוד פונקציות תגמול.

6.2.4 שימוש בסביבות מסובכות

המחקר של קבוצת החוקרים DeepMind משנת 2017 [57] הראה שבאמצעות שימוש ב- 64 מעבדים ניתן לאמן סוכן פארקור (Parkour) ב 100 שעות בפלטפורמת MuJoCo. כמו כן, החוקרים הראו שאם מגדירים את המשימה בצורה מורכבת על ידי הוספת מספר משימות בתצורות שונות תהליך הלמידה הופך קל יותר, שכן המדיניות לא יכולה להילמד בהתאמת-יתר למשימה מסוימת מבלי לאבד יעילות במשימות האחרות. כלומר, עצם הוספת משימות מגוונות מאפשרת הכללה של המודל והופכת את תהליך הלמידה למהיר יותר.

6.2.5 הוספת מנגנונים ללמידה נכונה

כאמור, בעיית התגמול הדליל הופכת את תהליך הלימוד לקשה היות ויש מעט מאוד מידע שניתן ללמוד ממנו. ישנם מספר כיוונים חדשים במחקר העכשווי אשר מנסים להתמודד עם הבעיה. כך למשל, קבוצת החוקרים OpenAI [65] מציעה טכניקה חדשה בשם Hindsight Experience Replay אשר מאפשרת לימוד יעיל בדגימות של תגמולים דלילים. החוקרים מראים שניתן לשלב את הטכניקה בכל אלגוריתם של למידה באמצעות חיזוקים. קבוצת חוקרים אחרת [66] מציעה טכניקה אשר מאפשרת לסוכן הלומד להיעזר במידע נוסף שנמצא בסביבה ולייצר פונקציות תגמול נלוות באמצעותו. החוקרים מדווחים על שיפור משמעותי בביצועי אלגוריתם DQN.

6.2.6 למידה מועברת

למידה מועברת (Transfer Learning) היא תחום מחקר של למידה חישובית אשר מתמקד באחסון מידע שנאגר מפתרון בעיה אחת ובהעברה שלו לצורך פתרון בעיה אחרת אך מאותו תחום. עד לפני כשנה הדבר לא היה אפשרי עבור רשתות נוירונים עמוקות שכן גראדיינטים ממשימות שונות עלולות להפריע אחד לשני בצורה שלילית ולהפוך את תהליך הלמידה ללא יציב ואף למנוע התכנסות [67]. עם זאת, קבוצת חוקרים [67] מציעה שיטה בה ניתן לשתף מדיניות בין המודלים השונים אשר מכילה בתוכה את החלקים המשותפים

בלבד. קבוצת חוקרים [68] פיתחה שיטה בה ניתן לגרום לרשת ללמוד מספר משימות שונות בצורה סדרתית על ידי האטה של הלמידה על משקולות של משימות שנלמדו כבר.

סיכום

בסקירה זו, בחנו את יחידות הבסיס של למידה באמצעות חיזוקים, קרי, פונקציות ערך, מדיניות ותגמול. סקרנו מגוון של אלגוריתמים קלאסיים כמו תכנון דינאמי, מונטה קרלו והפרש טמפורלי. בחנו מושגים ואלגוריתמים מהתחום של למידה חישובית ולמידה עמוקה, בפרט, הצגנו את העקרונות של רשתות נוירונים עמוקות ו Backpropagation. דנו באלגוריתמים עכשוויים לשילוב למידה באמצעות חיזוקים ולמידה עמוקה הן בדרכים של קירוב פונקציות ערך והן בדרכים של קירוב מדיניות. סקרנו מגוון רחב של יישומים מעשיים של למידה עמוקה באמצעות חיזוקים לפתרון בעיות ממגוון תחומי חיים, קרי, פיננסים, רפואה, עיבוד שפות טבעיות, נהיגה אוטונומית, רובוטיקה, משחקים וראייה ממוחשבת. לבסוף, הצגנו אוסף של אתגרים העומדים בפני החוקרים וראינו כיצד הם ישפיעו על עתיד המחקר של למידה עמוקה באמצעות חיזוקים.

למידה עמוקה באמצעות חיזוקים זוכה בשנים האחרונות להתעניינות מחודשת הודות להצלחות הרבות של למידה עמוקה יחד עם התפתחות מואצת של רכיבי חומרה. זה כבר שניים וחצי עשורים שכמות המאמרים האקדמיים בתחום נמצאת במגמת עלייה מתמדת וההערכה של החוקרים היא שהמגמה תמשך בשנים הקרובות.

למידה עמוקה באמצעות חיזוקים על ידי קירוב פונקציות ערך היא מרכזית במחקר העכשווי. אלגוריתם DQN היווה למעשה את פריצת הדרך הגדולה בתחום שלאחריה החלו להתפרסם אלגוריתמים רבים המהווים הרחבה ל- DQN, קרי, Double DQN, Prioritized Experience Reply, Dueling Networks, Average DQN ו- Rainbow וכן מגוון רחב של יישומים מעשיים. למידה עמוקה באמצעות חיזוקים על ידי קירוב מדיניות זוכה גם היא להצלחה רבה עם פרסומם של אלגוריתמים פורצי דרך, קרי, Deterministic Policy Gradient, Trust Region Policy Optimization ו- A3C.

למידה באמצעות חיזוקים דורגה על ידי המכון הטכנולוגי של מסצ'וסטס (MIT) כאחת מעשר הטכנולוגיות פורצות הדרך של שנת 2017 ולמידה עמוקה דורגה באותה רשימה בשנת 2013. חוקרים רבים בתחום מחשיבים את השילוב של למידה עמוקה ולמידה באמצעות חיזוקים כצעד הראשון בדרך ליצירתה של מסגרת תאורטית של בינה מלאכותית כללית, כלומר, כזו שיכולה לבצע כל משימה שאדם יכול לבצע. סילבר [8], התורם העיקרי של AlphaGo קבע את הנוסחה:

בינה מלאכותית כללית = למידה באמצעות חיזוקים + למידה עמוקה

חמשת האתגרים המרכזיים כיום בעומדים בפני החוקרים, קרי, יעילות הדגימה, תכנון פונקציית תגמול, הימלטות מנקודות קיצון מקומיות, הכללה ויכולות שחזור פתרונות מהווים את החסם העיקרי לשימוש בלמידה באמצעות חיזוקים ככלי ליצירת בינה מלאכותית כללית. אתגרים אלו הם שהניעו חוקרים לבדוק את גבולות היכולת של למידה באמצעות חיזוקים בבעיות שונות, חלקם נפתרו באופן חסר תקדים וחלקם טרם נפתרו.

המחקר העכשווי הוליד מספר לא מבוטל של רעיונות שניתן להניח כי הם יהוו את הבסיס להמשך המחקר העתידי, קרי, שימוש בלמידה עמוקה באמצעות חיזוקים ככלי כיוון עדין, למידת פונקציית התגמול והימנעות מתכנון ידני שלה, שימוש במשימות מורכבות, הוספת מנגנוני עזר ללמידה ופיתוח פרדיגמות למידה חדשות כמו למידה מועברת.

לסיכום, מטרת המחקר של למידה עמוקה באמצעות חיזוקים היא יצירתה של מערכת בינה מלאכותית כללית אשר יכולה לתקשר וללמוד מן העולם שמסביבה. ההתקשרות עם הסביבה היא היתרון והחיסרון של למידה באמצעות חיזוקים. על אף שישנם אתגרים רבים הנובעים מחקר של סביבות מסובכות בעולם שמשתנה כל העת, למידה באמצעות חיזוקים מאפשרת לנו לבחור באיזה אופן אנו רוצים לחקור את העולם. בעצם, למידה באמצעות חיזוקים מאפשרת לסוכן לבצע ניסויים על הסביבה בכדי להבין אותה טוב יותר וללמוד קשרים מסובכים בה. ההתקדמות המדהימה של השנים האחרונות הודות לשילוב של עקרונות הלמידה העמוקה יחד עם למידה באמצעות חיזוקים מקדמים אותנו צעד גדול בכיוון של יצירת מערכות בינה מלאכותיות כלליות אשר לומדות ופועלות בדרכים שהן יותר דומות לבני אדם.

רשימת מקורות

[1] Sutton, R. S. and Barto, A. G. (2016). Reinforcement Learning 2'nd edition. MIT Press, Boston, MA.

[2] Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT press.

[3] LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. "Deep learning." Nature 521.7553 (2015): 436-444.

[4] Schmidhuber, Jürgen. "Deep learning in neural networks: An overview." Neural networks 61 (2015): 85-117.

[5] Krakovsky, M. (2016). Reinforcement renaissance. Communications of the ACM, 59(8): 12–14.

[6] Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., & Meger, D. (2017). Deep reinforcement learning that matters. arXiv preprint arXiv: 1709.06560.

[7] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Petersen, S. (2015). Human-level control through deep reinforcement learning. Nature, 518(7540), 529. Chicago

[8] Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., ... & Chen, Y. (2017). Mastering the game of go without human knowledge. Nature, 550(7676), 354.

[9] Wang, Z., Schaul, T., Hessel, M., Van Hasselt, H., Lanctot, M., & De Freitas, N. (2015). Dueling network architectures for deep reinforcement learning. arXiv preprint arXiv: 1511.06581. Chicago

[10] Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., & Riedmiller, M. (2014, June). Deterministic policy gradient algorithms. In ICML.

[11] Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., ... & Kavukcuoglu, K. (2016, June). Asynchronous methods for deep reinforcement learning. In International Conference on Machine Learning (pp. 1928-1937).

[13] Knight, W. (2017). Reinforcement learning. MIT technology review 2017. https://www.technologyreview.com/s/603501/10-breakthrough-technologies-2017-reinforcement-learning/

[14] Hof, R. D. (2013). Deep Learning. MIT technology review 2013. https://www.technologyreview.com/s/513696/deep-learning/

[15] Garnelo, M., Arulkumaran, K., & Shanahan, M. (2016). Towards deep symbolic reinforcement learning. arXiv preprint arXiv: 1609.05518.

[16] Silver, D. (2016). Deep reinforcement learning, a tutorial at ICML 2016. http://icml.cc/2016/tutorials/deep_rl_tutorial.pdf.

[17] Russell, S., Norvig, P., & Intelligence, A. (1995). A modern approach. Artificial Intelligence. Prentice-Hall, Egnlewood Cliffs, 25, 27.

[18] Bellman, R. (1957). Dynamic Programming, Princeton, NJ: Princeton Univ.

[19] Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. In Reinforcement Learning (pp. 5-32). Springer, Boston, MA.

[20] Mohri, M., Rostamizadeh, A., & Talwalkar, A. (2012). Foundations of machine learning. MIT press.

[21] Mitchell, T. M. (1997). Machine learning. 1997. Burr Ridge, IL: McGraw Hill, 45(37), 870-877.

[22] Bengio, Y., Goodfellow, I. J., & Courville, A. (2015). Deep learning. Nature, 521, 436-444.

[23] MIT 6.S191: Introduction to Deep Learning. http://introtodeeplearning.com/

[24] Tesauro, G. (1995). Temporal difference learning and TD-Gammon. Communications of the ACM, 38(3), 58-68.

[25] Riedmiller, M. (2005, October). Neural fitted Q iteration–first experiences with a data efficient neural reinforcement learning method. In European Conference on Machine Learning (pp. 317-328). Springer, Berlin, Heidelberg.

[26] Van Hasselt, H., Guez, A., & Silver, D. (2016, February). Deep Reinforcement Learning with Double Q-Learning. In AAAI (Vol. 16, pp. 2094-2100).

[27] Schaul, T., Quan, J., Antonoglou, I., & Silver, D. (2015). Prioritized experience replay. arXiv preprint arXiv: 1511.05952. Chicago

[28] Anschel, O., Baram, N., & Shimkin, N. (2016). Averaged-dqn: Variance reduction and stabilization for deep reinforcement learning. arXiv preprint arXiv: 1611.01929.

[29] Hessel, M., Modayil, J., Van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., ... & Silver, D. (2017). Rainbow: Combining Improvements in Deep Reinforcement Learning. arXiv preprint arXiv: 1710.02298.

[30] Lin, L. J. (1992). Self-improving reactive agents based on reinforcement learning, planning and teaching. Machine learning, 8(3-4), 293-321.

[31] Lange, S., Riedmiller, M., & Voigtlander, A. (2012, June). Autonomous reinforcement learning on raw visual input data in a real world application. In Neural Networks (IJCNN), The 2012 International Joint Conference on (pp. 1-8). IEEE. Chicago

[32] Hasselt, H. V. (2010). Double Q-learning. In Advances in Neural Information Processing Systems (pp. 2613-2621). Chicago

[33] Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., ... & Wierstra, D. (2015). Continuous control with deep reinforcement learning. arXiv preprint arXiv: 1509.02971.

[34] Schulman, J., Levine, S., Abbeel, P., Jordan, M., & Moritz, P. (2015, June). Trust region policy optimization. In International Conference on Machine Learning (pp. 1889-1897).

[35] Babaeizadeh, M., Frosio, I., Tyree, S., Clemons, J., & Kautz, J. (2016). Reinforcement learning through asynchronous advantage actor-critic on a gpu.

[36] Wang, Z., Bapst, V., Heess, N., Mnih, V., Munos, R., Kavukcuoglu, K., & de Freitas, N. (2016). Sample efficient actor-critic with experience replay. arXiv preprint arXiv: 1611.01224.

[37] Moody, J., & Saffell, M. (2001). Learning to trade via direct reinforcement. IEEE transactions on neural Networks, 12(4), 875-889.

[38] Deng, Y., Bao, F., Kong, Y., Ren, Z., & Dai, Q. (2017). Deep direct reinforcement learning for financial signal representation and trading. IEEE transactions on neural networks and learning systems, 28(3), 653-664.

[39] Lu, D. W. (2017). Agent Inspired Trading Using Recurrent Reinforcement Learning and LSTM Neural Networks. arXiv preprint arXiv: 1707.07338.

[40] Jiang, Z., Xu, D., & Liang, J. (2017). A Deep Reinforcement Learning Framework for the Financial Portfolio Management Problem. arXiv preprint arXiv: 1706.10059.

[41] Miotto, R., Wang, F., Wang, S., Jiang, X., & Dudley, J. T. (2017). Deep learning for healthcare: review, opportunities and challenges. Briefings in bioinformatics.

[42] Liu, Y., Logan, B., Liu, N., Xu, Z., Tang, J., & Wang, Y. (2017, August). Deep Reinforcement Learning for Dynamic Treatment Regimes on Medical Registry Data. In Healthcare Informatics (ICHI), 2017 IEEE International Conference on (pp. 380-385). IEEE.

[43] Ling, Y., Hasan, S. A., Datla, V., Qadir, A., Lee, K., Liu, J., & Farri, O. (2017, November). Diagnostic inferencing via improving clinical concept extraction with deep reinforcement learning: A preliminary study. In Machine Learning for Healthcare Conference (pp. 271-285).

[44] Raghu, A., Komorowski, M., Ahmed, I., Celi, L., Szolovits, P., & Ghassemi, M. (2017). Deep Reinforcement Learning for Sepsis Treatment. arXiv preprint arXiv: 1711.09602.

[45] Jurafsky & Martin (2009), Speech and language processing, 2'nd edition. Pearson International Edition, ISBN 978-0-13-504196-3, Chapter 24.

[46] Li, J., Monroe, W., Ritter, A., Galley, M., Gao, J., & Jurafsky, D. (2016). Deep reinforcement learning for dialogue generation. arXiv preprint arXiv: 1606.01541.

[47] Narasimhan, K., Kulkarni, T., & Barzilay, R. (2015). Language understanding for text-based games using deep reinforcement learning. arXiv preprint arXiv: 1506.08941.

[48] Zhang, X., & Lapata, M. (2017). Sentence simplification with deep reinforcement learning. arXiv preprint arXiv: 1703.10931.

[49] Chae, H., Kang, C. M., Kim, B., Kim, J., Chung, C. C., & Choi, J. W. (2017). Autonomous Braking System via Deep Reinforcement Learning. arXiv preprint arXiv: 1702.02302.

[50] Sallab, A. E., Abdou, M., Perot, E., & Yogamani, S. (2016). End-to-End Deep Reinforcement Learning for Lane Keeping Assist. arXiv preprint arXiv: 1612.04340.

[51] Mirzaei, H., & Givargis, T. (2017). Fine-grained acceleration control for autonomous intersection management using deep reinforcement learning. arXiv preprint arXiv: 1705.10432.

[52] Levine, S., Finn, C., Darrell, T., & Abbeel, P. (2016). End-to-end training of deep visuomotor policies. The Journal of Machine Learning Research, 17(1), 1334-1373.

[53] Mirowski, P., Pascanu, R., Viola, F., Soyer, H., Ballard, A. J., Banino, A., ... & Kumaran, D. (2016). Learning to navigate in complex environments. arXiv preprint arXiv: 1611.03673.

[54] Heinrich, J., & Silver, D. (2016). Deep reinforcement learning from self-play in imperfect-information games. arXiv preprint arXiv: 1603.01121.

[55] Das, A., Kottur, S., Moura, J. M., Lee, S., & Batra, D. (2017). Learning cooperative visual dialog agents with deep reinforcement learning. arXiv preprint arXiv: 1703.06585.

[56] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., ... & Dieleman, S. (2016). Mastering the game of Go with deep neural networks and tree search. nature, 529(7587), 484.

[57] Heess, N., Sriram, S., Lemmon, J., Merel, J., Wayne, G., Tassa, Y., ... & Silver, D. (2017). Emergence of locomotion behaviours in rich environments. arXiv preprint arXiv: 1707.02286.

[58] Clark, J., Amodei, D. (2016). Faulty reward functions in the wild. https://blog.openai.com/faulty-reward-functions/.

[59] Raghu, M., Irpan, A., Andreas, J., Kleinberg, R., Le, Q. V., & Kleinberg, J. (2017). Can Deep Reinforcement Learning Solve Erdos-Selfridge-Spencer Games?. arXiv preprint arXiv: 1711.02301.

[60] Lanctot, M., Zambaldi, V., Gruslys, A., Lazaridou, A., Perolat, J., Silver, D., & Graepel, T. (2017). A unified game-theoretic approach to multiagent reinforcement learning. In Advances in Neural Information Processing Systems (pp. 4193-4206).

[61] Jaques, N., Gu, S., Bahdanau, D., Hernández-Lobato, J. M., Turner, R. E., & Eck, D. (2016). Sequence tutor: Conservative fine-tuning of sequence generation models with kl-control. arXiv preprint arXiv: 1611.02796.

[62] Finn, C., Levine, S., & Abbeel, P. (2016, June). Guided cost learning: Deep inverse optimal control via policy optimization. In International Conference on Machine Learning (pp. 49-58).

[63] Christiano, P. F., Leike, J., Brown, T., Martic, M., Legg, S., & Amodei, D. (2017). Deep reinforcement learning from human preferences. In Advances in Neural Information Processing Systems (pp. 4302-4310).

[64] Sermanet, P., Lynch, C., Hsu, J., & Levine, S. (2017). Time-contrastive networks: Self-supervised learning from multi-view observation. arXiv preprint arXiv: 1704.06888.

[65] Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., ... & Zaremba, W. (2017). Hindsight experience replay. In Advances in Neural Information Processing Systems (pp. 5048-5058).

[66] Jaderberg, M., Mnih, V., Czarnecki, W. M., Schaul, T., Leibo, J. Z., Silver, D., & Kavukcuoglu, K. (2016). Reinforcement learning with unsupervised auxiliary tasks. arXiv preprint arXiv: 1611.05397.

[67] Teh, Y., Bapst, V., Czarnecki, W. M., Quan, J., Kirkpatrick, J., Hadsell, R., ... & Pascanu, R. (2017). Distral: Robust multitask reinforcement learning. In Advances in Neural Information Processing Systems (pp. 4499-4509).

[68] Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., ... & Hassabis, D. (2017). Overcoming catastrophic forgetting in neural networks. Proceedings of the National Academy of Sciences, 114(13), 3521-3526.

Abstract

Reinforcement learning is a sub field of machine learning that deals with the interaction of an *agent* and its *environment* while learning an *optimal policy* by trial and error. The integration of reinforcement learning and deep learning has a long history, but with recent achievements in deep learning we have been witnessing a renaissance of reinforcement learning. The number of scientific publications keeps growing steadily year to year for the past two and a half decades. Breakthrough algorithms such as DQN and AlphaGo have been an inspiration for hundreds of extensions and practical real-life applications that soon came after. Still, there are many challenges facing researches today which will determine the future of the field research. In this paper, we survey the current research of deep reinforcement learning, while discussing the necessary theoretical background, algorithms, real-life applications, challenges and future research.

Table of Contents

The Open University of Israel

Department of Mathematics and Computer Science

Deep Reinforcement Learning: Theory, Applications and a look to the Future

Final Paper submitted as partial fulfillment of the requirements

towards an M.Sc. degree in Computer Science

The Open University of Israel

Department of Mathematics and Computer Science

By

Hanan Aharonof

Prepared under the supervision of Dr. Mireille Avigal

May 2019