פרויקט סיום בקורס מבוא לבינה מלאכותית

3D multi-player connect 4

מגישים:

איתמר שרם, 206762551

שלום בלוי, 319144762

עבד נירוך, 213668700

כאן יהיה תוכן עניינים

מבוא

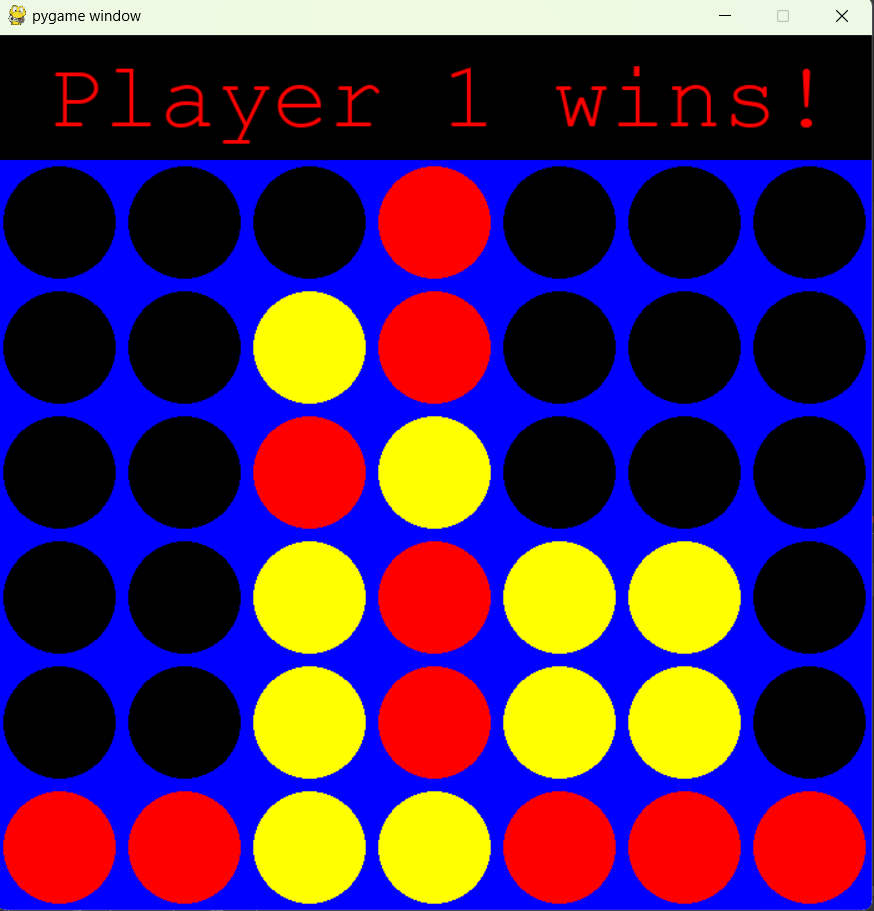
עבודות קודמות

מתודולוגיה

תוצאות

סיכום

ביביליוגרפיה(לא דרשו, אבל כדאי כדי לשים גם קישורים לUI ולעבודות קודמות וכו')

מבוא:

המשחק שלנו מבוסס על המשחק 4 בשורה. המשחק המקורי מתקיים בלוח דו ממדי בגודל קבוע, לשני שחקנים. כל שחקן בתורו משחיל דיסקית בצבע שלו לאחת העמודות בלוח. מטרתו של כל שחקן להגיע ראשון לרצף של 4 דיסקיות בצבע שלו. הרצף יכול להיות בשורה, בעמודה או באלכסון.

בפרויקט שלנו הרחבנו את המשחק, על מנת לאתגר את הסוכנים שלנו ולבחון אותם בסביבות מורכבות יותר. השדרוג חל בכמה אופנים:

* לוח המשחק מוגדר כעת ע"י הרביעייה הבאה: (R,C,D,W), כאשר R הוא מספר השורות, C הוא מספר העמודות, D הוא העומק וW הוא רצף הנצחון
* ניתן להריץ את המשחק עם יותר משני שחקנים, כל אחד מתחרה בשאר השחקנים על מנת לנצח.

עץ המצבים של המשחק גדל בצורה אקספוננציאלית- כאשר t הוא מספר התורות עד לניצחון. כלומר סוכנים שיממשו אלגוריתמי חיפוש פשוטים ירוצו למשך זמן רב! לפיכך צריך לחשוב על אלגוריתמים חכמים, וזו הסיבה שבחרנו במשחק זה.

לקחנו מימוש למשחק 4 בשורה שמצאנו באינטרנט, על מנת לאפשר לבן אדם לשחק נגד אחד הסוכנים בצורה אינטרקטיבית. שדרגנו משמעותית מימוש זה כדי שיתאים להרחבות שהוספנו למשחק. להוסיף קישור לgithub של המימוש, או רפרנס לנקודה מתאימה בביביליוגרפיה.

מימשנו סוכני AI שונים, לפי רעיונות שונים שלמדנו בכיתה, שמטרתם היא לנצח במשחק את השחקנים שיתמודדו מולם. הסוכנים שמימשנו הם:

1. minmax\_agent
2. alpha\_beta\_agent
3. RL\_agent
4. רשת CNN

נשים לב שהמשחק המקורי הוא משחק סכום אפס- נצחונו של שחקן אחד הוא הפסדו של האחר. לפיכך חשבנו כי האלגוריתם הטבעי ביותר להתחיל איתו הוא אלגוריתם המינמקס, שמותאם לנצח בסוג זה של משחקים. אלגוריתם מינמקס במימוש נאיבי, יורד עד לעומק עץ המצבים, ומהעלים מתחיל לשערך את הציון למצבים. במשחק 4 בשורה, שבו עץ המצבים עמוק מאוד, דבר זה אינו ישים. לכן נצטרך להגביל את עומק החיפוש שלנו ולתת ציון למצב לוח למרות שהמשחק עדיין לא הסתיים. לשם כך השתמשנו בהיוריסטיקה/היוריסטיקות שהגדרנו, אותן נתאר בחלק המתודולוגיה.

סוכן האלפא-בטא הוא הסוכן ההגיוני הבא- הוא פועל כמו סוכן המינמקס מבחינת ההחלטות שבוחר במהלך המשחק, אך מדלג על ענפים מסוימים ובכך מקצר את זמן הריצה.

לסיום, רצינו לבחון את ביצועיו של סוכן reinforcement learning . דרך אפשרית להפעיל את האלגוריתם במקרה שלנו, היא לתת לשחקן למידת החיזוק לשחק מול שחקן מינימקס/רנדומי ודרכו ללמוד איך לשחק את המשחק בצורה הטובה ביותר.

סוכן reinforcement learning – סוכן זה לומד באמצעות משחקים רבים את הפעולות הכדאיות בהינתן מצבי הלוח, כאשר במהלך הלמידה הסוכן מעדכן את הציון על מצב לוח מסוים, באמצעות הרצת סימולציות עד לנצחון/הפסד(monte carlo).

אולי יהיה הסבר קצר על רשת CNN

עבודות קודמות:

בחנו עבודות קודמות(לשים קישורים לרלוונטיים בביביליוגרפיה)

בשתיהן, האלגוריתמים בהם השתמשו הם minmax ו-alpha beta pruning . כפי שאנחנו הסקנו, עץ המצבים במשחק גדול מאוד, ולכן החוקרים(?) בחרו להגביל את עומק החיפוש ולחשב ציון ללוח של משחק שטרם הסתיים. ההבדלים בין האלגוריתמים שהם בחרו הוא בפונקציית האבליואציה שמחשבת ציון ללוח. נציג את ההיוריסטיקות המוצלחות ביותר בכל אחת מהעבודות.

נסמן ב-N את אורך הרצף המוגדר עבור נצחון במשחק(ברירת המחדל היא 4)

פונקציה א:

במשחק 4 בשורה עם לוח דיפולטיבי(6 שורות על 7 עמודות) ישנן 69 פוזיציות בהן אפשר להגיע לנצחון(24 אופקיות, 21 אנכיות ו24 אלכסוניות). אם ברצף באורך N יש לשני השחקנים דיסקיות, נתעלם מרצף זה(שכן אף אחד מהם לא יכול לנצח שם). אחרת, לכל רצף בלוח ניתן ערך לפי כמות הדיסקיות שיש בו לשחקן מסוים. נסכום את כל הערכים הללו עבור השחקן הראשי, ונחסר מהם את סכום הערכים של היריב.

בהיוריסטיקה זו יש כמה חסרונות:

* הערכים שנבחרו מותאמים עבור לוח בגודל הדיפולטיבי בלבד, וקשה להכליל אותה למשחק עם לוח בגודל שונה, עם רצף נצחון שונה.
* ההיוריסטיקה יכולה לתת ערך זהה ללוח בו יש לנו הרבה רצפים קצרים, לעומת מעט רצפים ארוכים. אם נחשוב על הרלקסציה שבה ככל שיש לשחקן יותר דיסקיות ברצף, הוא יותר קרוב לנצחון, היוריסטיקה זו מתעלמת מעיקרון זה.

פונקציה ב(נקראה IBEF2): נספור רצפים לא חסומים(שלא מכילים דיסקיות של היריב).ונחזיר , עבור k מספר הדיסקיות ברצף זה. הפונקציה תסכום את הערכים הללו ותחסיר מערך זה את סכום הערכים של היריב.

חסרונות של היוריסטיקה זו:

בדומה להיוריסטיקה הקודמת, יכולה לתת ערך זהה ללוח בו יש לנו הרבה רצפים קצרים, לעומת מעט רצפים ארוכים.

מתודולוגיה:

* לדבר על הטאפל שני, ועל האיבר השני בו שמהווה שיפור לעומת IBEF2
* לדבר על כך שאם אנחנו נתקלים בנצחון, ישר מחזירים אינסוף ולא נותנים ציון מספרי, בגלל העניין עם העומקים
* לדבר על העניין שעיקרון הפעולה שלנו דומה להפעלת קונבולוציה, אבל הוא יעיל יותר כי לא צריך לחשב את הקונבולוציות על כל הלוח כל פעם מחדש.
* להוסיף שהוכח שהשחקן הראשון בארבע בשורה תמיד מנצח, ולכן כשהרצנו את האלגוריתמים תמיד הרצנו שפעם אחת הם מתחילים, וברצף משחקים אחר הם לא מתחילים.

הרחבת אלגוריתם minmax עבור משחק של יותר משני שחקנים:

במקרה זה, השחקן הראשי יהיה שחקן המקסימום, ומבחינתו כל שאר השחקנים יהיו שחקני המינימום. בשערוך הערך עבור כל קודקוד בעץ ששחקן המקסימום מגדיר(עד עומק מסוים), שחקן המקסימום יבחר את הערך המקסימלי מבין ערכי כל ילדיו. כל אחד משחקני המינימום יבחר את המינימום מבין ערכי כל ילדיו.

נשים לב כי בגישה זו, מטרתו של כל שחקן מינימום היא לוודא ששחקן המקסימום לא מנצח, גם אם זה גורר ששחקן מינימום אחר ינצח- מכיוון שההיוריסטיקות מחושבות במינוס על כלל היריבים, אזי מטרת כל אחד משחקני המינימום היא שכל השחקנים היריבים(כולל הם עצמם) יקבלו ניקוד גבוה, וכך יורידו מערכו של שחקן המקסימום. בעיני שחקן המקסימום, כלל יריביו עשו יד אחת נגדו.

ההיוריסטיקה עבור האלגוריתם(גם היא מניחה שיכולים להיות יותר משני שחקנים):

ההיוריסטיקה פועלת באופן הבא:

בהינתן לוח מסוים, ההיוריסטיקה תבחן את כל הרצפים השונים באורך של רצף הניצחון הדרוש. ההיוריסטיקה מחזירה עבור לוח זוג ערכים:

* הרצף הארוך ביותר שהושג- max\_streak
* מספר הרצפים באורך זה- num\_of\_max\_streaks .

עבור כל שחקן:

* אם רצף מסוים לא מכיל כלל דיסקיות שלו, או מכיל דיסקיות שלו אך גם של היריב, הוא לא יילקח בחשבון
* אם הושג נצחון, נחזיר אינסוף עבור לוח זה באופן מיידי, ומספר הרצפים באורך זה.
* אחרת, נחזיר , ומספר הרצפים באורך זה

לסיום, ההיוריסטיקה תחזיר את הערכים הבאים:

* max\_streak(main\_player)-average\_max\_streak(opponents)
* num\_of\_max\_streaks(main\_player)-sum(opponents\*\_num\_of\_streaks) ,כלומר מחסירים מהשחקן הראשי את כמות הרצפים שלו, רק את כמויות רצפים שהצליחו להגיע לאותו רצף.

הסבר:

* ראשית, בהשוואה בין ציונים של שני לוחות שונים, הלוח עם ה-max\_streak הגדול יותר ייבחר קודם. רק אם לשניהם ערך זהה, נכריע לפי num\_of\_max\_streaks
* הסיבה להחזרת אינסוף אם יש נצחון מיידי, היא בגלל העניין הבא: כאשר האלגוריתם מורץ עם עומק גדול מ2, יכול לקרות מצב שיינתן אותו ציון ללוח שבו אנו מגיעים לרצף נצחון בתור הקרוב ואז עושים פעולה כלשהי בתור הבא, או להפך. אך אם היינו יכולים לנצח בתור הקרוב, אין סיבה שלא נעשה זאת כעת!
* רצינו לתת חשיבות קודם כל לרצף הארוך ביותר שהשגנו. אם שאר השחקנים בלוח זה השיגו רצפים מקסימליים קטנים יותר, זהו מדד מבחינתנו ללוח שכדאי לבחור בו, לפיכך כאשר עושים נותנים חשיבות גדולה יותר לרצפים ארוכים יותר, ובנוסף מקטינים את הסיכוי שהלוח לא ייבחר בגלל יכולותיהם של השחקנים האחרים.

ביביליוגרפיה:

* <https://www.cs.huji.ac.il/course/2021/ai/projects/old/4InRow_2.pdf>
* <https://www.cs.huji.ac.il/course/2021/ai/projects/old/4InRow_1.pdf>
* <https://en.wikipedia.org/wiki/Connect_Four>
* <https://www.fierz.ch/strategy1.htm>

דברים להוסיף ולערוך:

1. לעבד- צריך לשנות את ההיוריסטיקה IBEF2 כך שהיא באמת תהיה דומה לה.
2. מעבר לרשת CNN: צריך ליצור דאטה(נריץ מונטה קרלו מספר רב של פעמים ממצבי לוח, על מנת לקבל עבורם ערך. זה יהיה הלייבל של הלוח, שרשת הנוירונים תיתקל בה במהלך האימון.   
   נאמן רשת CNN שתקבל את הדאטה סט הנ"ל, ותפיק עבור כל לוח כזה ציון. סוכן הCNN בהנתן מצב לוח כלשהו, ישחק את כל המהלכים האפשריים עבורו, ויבחר במהלך שנותן לוח עם ציון מקסימלי.
3. לחשוב האם ניתן לחשוב על פונקציית היוריסטיקה אחרת, שבה באלגוריתם minmax עם כמה משתתפים, שחקן המקסימום לא יחשוב שכולם נגדו, אלא כולם נגד כולם.