**מבוא לבינה מלאכותית**

**סמסטר חורף תשפ"ב**

**מטלה 5**תאריך הגשה: 28.12.2021 23:55

הנחיות

* שאלות בנושא מטלה זו יש לשאול דרך המודל, בפורום "מטלה 5".
* הוראות להגשת המטלה מופיעים בסוף מסמך זה.
* הקבצים הנדרשים להרצת הקוד הינם:
  + main.py
  + grid.py
  + q\_learning.py
  + uncertain\_Grid.py
  + value\_iteration.py
  + routes\_1.txt
  + routes\_2.txt
* העבודה להגשה ביחידים בלבד אלא אם כן המגישים קיבלו אישור להגשה שאינה ביחידים.
* לפני שניגשים לממש את המטלה מומלץ לעיין רבות בהסברים וכן בקוד הקיים.
* פתרון המטלה שתגישו ייבדק מול שאר ההגשות על ידי תוכנת העתקות.
* **מי שימצא כי העתיק יכשל בקורס וכן יועבר לוועדת משמעת אוניברסיטאית**.
* הפרויקט נכתב וייבדק בשפת התכנות python, גרסה 3.6 או 3.7.
* יש להתקין את החבילות numpy ,pandas tabulate,.
* מסמך זה בנוי באופן הבא: תיאור המטלה, תיאור הבעיה, שאלות המטלה והסבר על המימוש הקיים והסבר על הגשת המטלה.

תיאור המטלה

**מטרות המטלה:**

* מימוש והכרת אלגוריתם value iteration למציאת policy אופטימלי
* מימוש והכרת אלגוריתם Q Learning למציאת policy אופטימלי

הוראות כלליות:

חלקי הקוד שיש עליכם להשלים כתובים בהערות בצורה מפורטת ומסומנים על ידי הערת TODO.

instance של בעיה ייוצג בקוד על ידי מופע של המחלקה grid.py או uncertain\_grid.py.

שימו לב, בגלל שהאלגוריתם של Q Learning הינו אלגוריתם הסתברותי ייתכן מצב שבו ה sanity test לא יעבור בכל הפעמים (בעת ההגשה ישנה בדיקה יתירה כדי לוודא policy תקין לפחות פעם אחת).

תיאור הבעיה

ברכב האוטונומי שלנו התגלתה בעיה במערכת הניווט האוטומטית. בכל תזוזה ממקום למקום במהלך נסיעה קיים סיכוי שמערכת הניווט תחליט להכווין את הרכב דרומה, גם אם זה לא חלק מהמסלול.

במטלה זו, עליכם לעזור להכווין את המכונית למוסך הקרוב, בעזרת תכנון מדיניות (Policy) לבעיה.

מרחב התנועה של הרכב מיוצג ע"י גריד בגודל NxN והעלויות במיקומים השונים מוצגות במטריצה בגודל NxN.

בחלק מהגריד ישנם מיקומים 'פקוקים'. כאשר הרכב נכנס למיקומים אלה, ישנה עלות (Reward שלילי) גבוהה לצאת מהם. מצד שני, קיימים מיקומים שבהם הנסיעה 'חלקה' ולכן הפעולות במיקומים אלה יתוגמלו (Reward חיובי). חוץ ממיקומים אלה, עלות פעולה רגילה הינה 1-.

הנסיעה מסתיימת כאשר מגיעים עם הרכב למוסך לתיקון (במיקום ה instace.end\_state כאשר instance הינו ה class המייצג את הבעיה), על כך נקבל Reward גבוה.

שאלות המטלה ומימוש נדרש

הרצת האלגוריתמים השונים תתבצע בקובץ main.py .

שימו לב, עבור כל אלגוריתם מומלץ לעבור על הקוד המתאים לו גם בחלק הממומש וגם בפונקציות שאינן מצויינות במסמך זה. יתכן וקיימים מספר רמזים בקוד עצמו שעשויים לעזור לכם לפתור בצורה קלה יותר. בנוסף, מעבר על הקוד יעזור לכם להבין את מבני הנתונים השונים בהם אתם צריכים להשתמש (הקוד כולו מתועד).

כמו כן, פסאודו-קוד של כל אלגוריתם שעליכם לממש קיים במצגות.

**אלגוריתם Value Iteration:**

נתונה בעיית MDP. בבעיה זו ידוע ה Grid במלואו, וכמו כן ידועות ההסתברויות לביצוע פעולות שונות בכל מיקום בGrid, וכמו כן ה Rewards עבור כל מיקום ופעולה. עליכם לממש את אלגוריתם Value Iteration עבור הבעיה באופן הבא:

משימות:

1. השלימו את הפונקציה הבאה, המבצעת את bellman update:

**value\_iteration\_update**(instance, all\_states, q\_table, previous\_iteration\_value\_array, current\_iteration\_value\_array):

עליכם להשלים את הפונקציה כך שתעדכן את ה value array בהינתן ה value array הקודם (previous\_iteration\_value\_array).

ה- value array ממפה מצב לתועלת שלו באיטרציה מסוימת.

הפונקציה מחזירה את ה value arrays הישן, החדש, ובנוסף את ה- q table **מעודכן**.

1. השלימו את הפונקציה הבאה, הגוזרת policy מתוך q-table:

**get\_policy**(instance, q\_table):

הפונקציה מחזירה Policy מתאים ל Q Table האחרון אותו עדכנו בצורת Dictionary בדומה לדוגמה בקוד.

1. השלימו את הפונקציה הבאה, המממשת את לולאת value iteration:

**value\_iteration**(instance):

עליכם להשלים את תנאי העצירה של value iteration בלולאה, במקום המתאים (כתוב בהערות בקוד).

פונקציות שימושיות עבור המימוש:

לטובת המימוש כדאי להשתמש בפונקציות הבאות של הclass בשם Grid (בקובץ grid.py):

Grid\_object.**get\_actions**(state):

הפונקציה מחזירה את אוסף הפעולות האפשריות עבור state (מיקום x,y), עבור כל פעולה היא מחזירה את ה outcome-ים השונים האפשריים ומה ההסתברות המתאימה להם. הפעולות האפשריות הן ‘north’, ‘south’, ‘east’, ‘west’.

לדוגמה:

example = {'north': {(0,0):0.9, (2,0):0.1}, 'south':{(2,0):1.0}}

במצב זה ניתן לזוז למעלה או למטה.

והשימוש המתאים:

example[‘north’] == {(0,0): 0.9, (2,0): 0.1}

אם זזים למעלה מגיעים לנקודה (0,() בהסתברות 0.9 ולנקודה (2,0) בהסתברות 0.1.

example[‘north’][(0,0)] == 0.9

Grid\_object.**get\_reward**(state, action):

הפונקציה מחזירה עבור state (מיקום x,y) ופעולה (action, כלומר: צפון, דרום, מזרח או מערב) את הreward המתאים לה.

לדוגמא:

reward = get\_reward((0,0), ‘north’)

**אלגוריתם Q Learning:**

נתונה בעיית MDP. הפעם, בבעיה לא ידוע ה Grid במלואו, אלא ידועים רק המיקום ההתחלתי והסופי. כמו כן לא ידועות ההסתברויות לביצוע פעולות שונות בGrid, וה Rewards השונים. עליכם לממש את אלגוריתם Q Learning עבור הבעיה באופן הבא:

משימות:

1. השלימו את הפונקציה הבאה:

**update\_q\_table**(q\_table, cur\_state, new\_state, action, reward, learning\_rate):

הפונקציה מקבלת state נוכחי וחדש, פעולה אותו היא ביצעה, את ה reward עבור הפעולה ו learning rate (שישאר קבוע לאורך הריצה). הפונקציה צריכה לעדכן את ערך ה q של ה state הנוכחי ולהחזיר את ה Q table החדש.

1. השלימו את הפונקציה הבאה:

**choose\_action\_to\_execute**(q\_table, current\_state)

הפונקציה ממומשת חלקית (עם אתחול הערך המתאים להסתברות ל exploration, כאשר הוא יישאר קבוע לאורך הריצה) ומקבלת את ה Q table והstate הנוכחיים. הפונקציה מחזירה איזו פעולה לבצע (action, כלומר: צפון, דרום, מזרח או מערב).

1. השלימו את הפונקציה הבאה:

**get\_policy**(instance, q\_table):

הפונקציה מחזירה Policy מתאים ל Q Table האחרון אותו עדכנו בצורת Dictionary בדומה לדוגמה בקוד (אמור להיות זהה עבור value iteration).

1. השלימו את הפונקציה הבאה:

**q\_learning**(instance, learning\_rate=0.01, num\_episodes=10000):

פונקציה זו מקבלת instance של הבעיה ומחזירה את ה policy האופטימלי. עליכם להשלים את התנאי שבו מצב אינו מופיע עדיין ב q\_table. רמז: ניתן להשתמש בפונקציות קיימות.

פונקציות שימושיות עבור המימוש:

**max\_q\_value\_for\_state**(q\_table, state)

הפונקציה מחזירה את הערך המקסימלי מבין הפעולות האפשריות ב state הנתון לפי ה q table. רמז: ניתן להשתמש בזה גם עבור ה state החדש.

**add\_new\_state\_to\_q\_table**(instance, q\_table, state)

הפונקציה מוסיפה state חדש ל q\_table הנוכחי ומחזירה את ה q\_table המעודכן.

הגשת המטלה

* יש להגיש **שני קבצים בלבד** לאתר המודל בדומה להגשת המטלה הקודמת. הקובצים הנדרשים:
  + q\_learning.py
  + value\_iteration.py
* ניתן לראות את הפידבק להרצה (קומפילציה, מספר טסטים שעברו, שגיאות זמן ריצה וכו'...). ניתן להעלות קובץ נוסף ולהריץ אם היו טעויות שאתם מעוניינים לתקן ולהשלים.

שימו לב, הבדיקה האוטומטית תיקח מעל דקה. כמו כן, קיים סיכוי קטן מאוד שגם השלמתם כהלכה את החלק של Q Learning ה policy שיתקבל יהיה שגוי. לכן, ניתן להריץ יותר מפעם אחת את אותו הקובץ שהעלתם.

לאחר סיום ההרצה יתקבלו התוצאות. ישנם 2 טסטים גדולים הבודקים גרסאות שונות של הבעיה, אחד עבור value iteration ואחד עבור Q learning. אם יוגש קובץ עם שגיאת קומפליציה יש עליכם להעלות קובץ חדש בכדי לקבל ציון לאחר התיקונים. הקבצים המקוריים שקיבלתם עם המטלה הינם ללא שגיאות ריצה.

בהצלחה 😊