בינה מלאכותית מבוזרת ומתקדמת

ראינו:
1. תיאור בעיה דטרמיניסטית באמצעות גרף, וחיפוש תכנית - שהיא מסלול בגרף.
2. תיאור אותה בעיה בסביבות סטוכסטיות, וקבלת reward שהמטרה היא למקסם אותו (במקום "סתם" למזער את מספר הצעדים ליעד). מחפשים policy ולא תכנית.

- לפעמים לא יודעים מה ההתפלגויות, וצריך לבצע Reinforment Learning.

עכשיו נדבר על מצבים בהם יש יותר מסוכן אחד - Multi-agent / Distributed AI. אלו בעיות שנמצאות כיום בחזית המחקר.

נשים לב: צריך תמיד לזכור שכל השיטות שאנו מדברים עליהן הן חלק מהפתרון - לא חלק מהבעיה. הן דרך שלנו למדל את המציאות, לא משהו שמחייב את המציאות.

תזכורת: חישוב מבוזר.

היחס בין מספר המחשבים למספר האנשים שמשתמשים בהם הולך ועולה. אם פעם היה מחשב למדינה, ואז לארגון, ואז למשק בית... היום לרוב האנשים יש מספר מחשבים (מחשב אישי, טלפון, וכו׳)

למרות ש: אנחנו קצת נסוגים אחורה, כי יש שירותי ענן - מחשב אחד מאוד חזק שמשרת הרבה אנשים.

ביזור מאפשר יותר כוח חישוב (יותר מחשבים), יותר רובסטיות (אם מחשב אחד נפל, יש אחרים) וגישה יותר מהירה לדברים מקומיים למחשב אחד.

בAI, נבדיל בין:

- חישוב מקביל - אותם מחשבים עובדים על אותה משימה.
- מחשוב מבוזר - מחשבים שונים (או תהליכים שונים באותו מחשב) עובדים על משימות שונות - אבל הכל למען אותה מטרה.

למשל: תתי־סוכנים של LLMים. לכל סוכן יש מטרות שונות, יכולות שונות, וגישה למידע שונה. הם יכולים לשתף פעולה - או יכולים שלא. כל מערכת היא סוכן, ומותר להם להגיד "לא" אחד לשני.

ביזוריות מציבה גם אתגרים:

- האם הסוכנים משתפים פעולה, או מתחרים אחד בשני? או אולי מצב ביניים - collaborating - משתפים פעולה רק כשזה מתאים להם.
- מה האחריות של כל סוכן מול הסוכנים האחרים שהוא עובד מולם? למשל - אם המשתמש ביטל פעולה, האם הסוכן אמור לבקש מהסוכנים האחרים לבטל כדי לחסוך טוקנים?
- פרטיות - איך הסוכנים מעבירים מידע פרטי אחד לשני? איזה מידע מותר להם להעביר?

- אלתרואיזם - באיזה מצבים סוכן אמור לבצע משהו שטוב לכולם למרות שזה לא מקדם את המטרה הספציפית שלו.

בכל נושא של AI שנגענו בו בקורס, אפשר להניח שמישהו כבר בדק את הגישה המבוזרת שלו.

נחילים ו-Multi-Agent RL

איך בינה יחידה הופכת להיות חלק מרבים? איך סוכנים הופכים להיות חברתיים?
ההבדל בין "צוות" ל"נחיל" הוא לא מספר הפריטים בקבוצה, אלא יכולת התקשורת.

- ככל שעולים בגדלים של הקבוצות, ככה יכולות התקשורת מגיעות לגבול - וככה יותר קשה לקיים עבודת צוות.

  בנחילים אי אפשר לבצע תיאום באמצעות תקשורת גלובלית, ולכן כל פרט צריך לבצע את המשימה שלו בהסתמך על תקשורת לוקאלית.

  (מבנה היררכי הוא דרך להתגבר על בעיית התקשורת. אבל זה לא מבנה שקיים בטבע. נחילים הם שטוחים - אין היררכי ולכן צריך פתרונות אחרים)

- בצוותים לכולם יש מטרה אחת. בנחילים לכל אחד יש מטרה משלו.
- בצוותים כל סוכן אחראי להצלחה של כולם. בנחילים כל סוכן אחראי לעצמו. בנחילים, סוכן בודד לא תמיד יכול לדעת אם וויתור על החלק שלו יעזור לקולקטיב.

הרעיון שצריך להתרגל אליו זה שאי אפשר לבנות סוכן שהוא פרט ופשוט להגיד לו "אתה עכשיו חלק מנחיל". צריך לבנות בתוכו דברים שיאפשרו לו להיות חלק ממאמץ חברתי.
אחד מהדרכים לעשות את זה זה Reinforcement Learning.
עבור סוכן בודד בד"כ יש לולאה, שבה הוא מקבל מידע מהסביבה, בוחר פעולה, ופועל.
במקרה של סוכן חברתי, איך הוא יכול לבחור פעולה אישית בלי לדעת מה ההשפעה שלה על הקולקטיב?

למשל: כמה מידע הסוכן צריך לשלוח? מידע יותר מפורט יכול לעזור, אבל גם ליצור עומס על הרשת שיעשה בעיות לסוכנים האחרים שגם צריכים לשלוח מידע.

מודל Rational Swarming

רובוט בודד:

1. עובד על המשימה שלו $\mathcal{P}$ (programmed task)
2. מנסה להימנע מהתנגשות
3. בוחר פעולה כדי למנוע התנגשות - $\mathcal{C}$
4. חוזר ל1

המחקר התחיל במודל שבו שכל סוכן הוא Reinforcement Learner בפני עצמו, ואז הוא לומד איזה פעולה לבחור כדי לסיים את זה יותר מהר.

יש כאן סוג של MAB - בכל פעם שיש התנגשות, צריך לבחור פעולה שתצמצם את הנזק (אבדן זמן) מההתנגשות[1]. אבל בגלל שכל סוכן לומד בעצמו, ובגלל שבסיבוב הבא הסוכנים שוב יצטרכו לקבל החלטה, יש כאן repeating game, ולכן עושים הרחבה של MDP לריבוי סוכנים:

Cooperative Markov Game

$\langle N, S, A, .D, R\rangle$ כאשר:

N	מספר הסוכנים
S	מצבים
A	סט של joint actions
D	íéáöî ïéá øáòîì úåøáúñä
R	íéáöîì äòâäì ìåîâú

הקאצ' זה שסוכן לא יודעים את הS של כולם (רק את החלק שלי), לא יודע את הR של כולם (רק את החלק שלי) וכמובן יכול לבחור רק את הפעולה של עצמו.

התועלת הכללית היא:

$$\mathcal{U}_K(\pi) = \sum_{k=1}^{K} \sum_{s_{t_k} \in S} \left[R\left(s_{t_{k-1}}, \pi\left(s_{t_{k-1}}\right), s_{t_k}\right) \cdot D\left(s_{t_{k-1}}, \pi\left(s_{t_{k-1}}\right), s_{t_k}\right) \right]$$

אבל זה לא ידוע, ואי אפשר למדוד לפי הרובוטים. לכן נקבע תגמול שכל סוכן יכול לחשב לעצמו:

$$R_k^i \coloneqq \left[\beta^i \mathcal{P}_{k-1}^i - \alpha^i \mathcal{C}_{k-1}^i\right]$$

נשים לב:

- צריך לזכור שהreward לא מגיע מהסביבה. הreward הוא פענוח של הסוכן כדי שתהליך הלמידה יעבוד.
- תגמול הוא גם לא פוקנציה של האלגוריתם. reward טוב יעזור לכל אלגוריתם, וreward גרוע יפגע בכל אלגוריתם.

[1] להשתמש בreward הרגיל של ביצוע המשימה לא עבד טוב, כי הreward הזה לא היה מיידי ולכן היה קשה לשייך אותו לפעולה ספציפית שהסוכן בחר.

איך מוודעים שמה שהרובוטים לומדים טוב לכולם? יכול להיות הרי שמיקסום של $\mathcal{P}^i\left(a^i\right)$ יקטין את $\mathcal{P}^j\left(a^j\right)$.

הרעיון: לגרום לכל סוכן להעריך כמה הוא פוגע באחרים, ולהפוך את זה לחלק מהתגמול שלו.

לכל סוכן יש את התועלת הלוקלית שלו. אפשר מתוך לזה לחשב "כמה קיבלנו כולנו ביחד כשאני מעורב (R_+), לעומת כמה היינו מקבלים אם לא הייתי מעורב (R_-)".

באופן נאיבי:

$$\phi\left(R^i\left(a^i\right)\right) = R_+ - R_- = \sum_{j\in N} R^j - \sum_{j\in N, j\neq i} R^j = R^i\left(a^i\right)$$

אבל זה **<u>לא נכון</u>**! כי המעורבות של הסוכן משפיעה גם על כמה האחרים יכולים לתרום.

בשביל לשערך את $\widehat{R_+}$ אפשר פשוט לשערך כמה כל אחד מהסוכנים האחרים תורם, בהנחה שהוא פועל כמוני. בשביל לשערך את $\widehat{R_-}$ צריך להניח שכל ההתנגשויות בינינו לא קרו ולכן הסוכנים האחרים קיבלו R^* כי הם עבדו תמיד ב$\mathcal{P}$ ולא ב$\mathcal{C}$ - כלומר $\widehat{R_-} = (N-1)\,\widehat{R^*}$. ואז:

$$\phi\left(R^i\left(a^i\right)\right) = \cdots = R^i\left(a^i\right) - (N-1)\left[\widehat{R_+^*} - \widehat{R_+^i}\right]$$

אבל איך רובוט יודע את N? הוא יכול להשתמש בחיישנים שלו, אבל אז יש לו רק מידע על מספר הסוכנים בסביבה הלוקאלית שלו. זה עובד, בגלל שממילא הסוכן מתנגש רק עם הסביבה המקומית שלו[2].

הפתרון הוא MAB, אבל הreward הוא זמן ולכן זה משנה קצת את החישובים.

חשוב לזכור: בד"כ בRL לא צריך אלגוריתם אחר - צריך פשוט reward function אחר.

[2] זהו הסבר אינטואיטיבי, לא מתמטי.

מקליד: עידן אריה
מרצה: פרופ׳ גל קמינקא
תאריך: 2026-06-11

בינה מלאכותית
89-5570-06

נחילים ברפואה

רובוטים מולקולרים הם רובוטים בגדלים של ווירוסים. אפשר להתייחס לבעיות של ביולוגיה מולקולרית בתור בעיות של מדעי המחשב (להרכיב מהם שערים לוגיים) - מאוד מורכב לבני אדם לתכנן, אבל optimizerים וplannerים יכולים לעשות את זה.

נדבר על בעיית Personalized Medication Planning:

1. איזה תרופות לתת?
2. כמה לתת מכל תרופה?
3. מתי לתת כל מנה?

זוהי בעיית temporal planning.

ישנו תחום ברפואה - pharmacokinetics (PK) - שמתעסק באיך תרופה מתפשטת בגוף. תרופה שהיא לא targeted מגיעה לכל מקום ומשפיעה על איברים שהיא לא עוזרת בהם - ולפעמים אפילו עושה נזק.

הערה: על זה מורכב תחום אחר - pharmacodynamica - שחוקר מה התרופה עושה לאיבר אליו היא מגיעה.

איך עושים על זה planning?

מצבים: לכל צירוף של איבר ותרופה יש ערך מספרי, שמתאר את ריכוז התרופה באיבר. מצב הגוף מתואר לפי אוסף כל האיברים, אוסף כל התרופות, והערכים.

פעולות: פעולה זה הזרקה של תרופה.

- לתרופה יש התנהגות לאורך זמן - בניגוד למה שראינו עד כה, בעולם האמיתי לפעולות יש זמן, ובזמן הזה היא יכולה לעבור דרך הרבה stateים. ההזרקה עצמה רק מתחילה את התהליך.
- יכולות להיות כמה הזרקות במקביל, או שאולי אחרי קצת זמן נזריק עוד מאותה תרופה.

מטרות: ריכוז תרופה ואפקט ברקמה מסויימת.

אילוצים: רוצים להימנע מאפקטים שליליים של התרופות, ולכן צריך להגדיר מצבים מסויימים כלא־חוקיים.