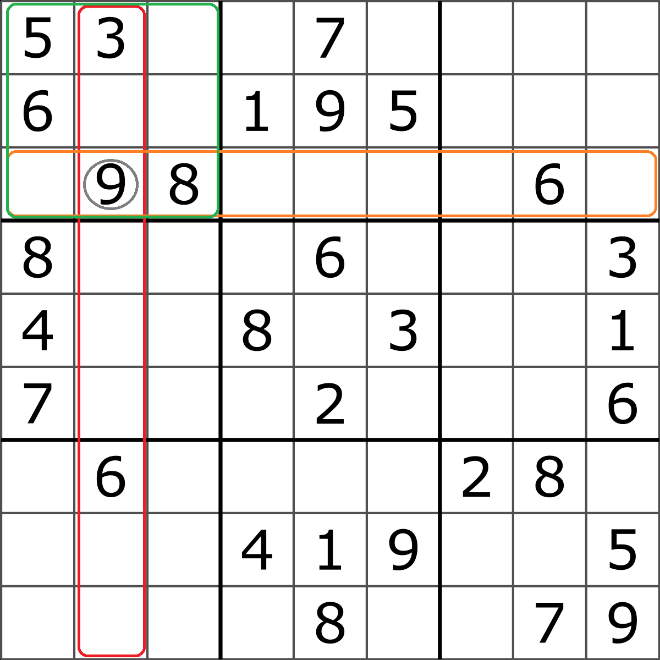
Решавање на Судоку загатки со Deep Q-Learning

Павел Ѓорговски

***Абстракт: Судоку загатки се меѓу најчестите проблеми со задоволување на ограничувања (constraint satisfaction problems) кои се користат во машинско учење за тестирање на проблеми во пребарување преку состојби и поттикнато учење. Но, со тоа што за секоја Судоку загатка може да се најде решение со поконвенционален метод кој бара помалку време, досега има многу малку случаи каде Q-Learning е користено за решавање на истите. Овој труд ќе ги анализира проблемите при користење на Q-Learning при решавање на Судоку, како и загатки слични на Судоку, и дали е воопшто тоа изводливо.***

*1. Вовед*

Судоку загатки претставуваат матрици од 9 редици и 9 колони, во кои можат да се внесе број од 1 до 9, така што тој број нема да се наоѓа повторно во истата редица, колона или пак во истата 3х3 матрица, од кои има 9. Сликовито претставено, 9ката не може да се појави повторно во ни едно од обоените полиња.



Во секоја загатка, има одреден број на однапред предефинирани вредности, и задачата на личноста која ја решава е да го најде решението на загатката. Правилна Судоку загатка има точно 1 решение, но ако има повеќе, секое решение кое ги исполнува горните услови за сите вредности се важи како точно решение. Судокуто се смета за решено кога секое поле е пополнето со вредност од 1 до 9, и условите се исполнети.

Решавање на Судоку е веќе постигнато на повеќе начини, од кој наједноставен и најефикасен е backtracking алгоритмот, односно пробување на некоја вредност на некое поле, се додека не настане конфликт, при што се менува во друга вредност. Иако секоја Судоку загатка може вака да се реши, ова е најекстремниот случај на brute force, и постојат и алгоритми кои според условите дадени од ограничувањата, ја внесуваат вредноста која со најголема „веројатност“ припаѓа во полето.

Во овој труд ќе се прави анализа на Q-Learning алгоритмот, со цел да се провери дали може да се истренира општ агентен модел за да се решаваат Судоку загатки, кој би ги задоволувал сите ограничувања поефикасно од простите алгоритми за пребарување низ состојби.

*2. Сродни истражувања*

Алгоритмот кој е користен во ова истражување е Q-Learning [1], поточно варијатната со длабоко учење [2], кој е алгоритам кој за одредена матрица на состојби и можни акции, доделува вредност за секоја акција во секоја состојба, така што следниот пат кога агентот ќе се најде во некоја состојба, тој ќе може да ја избере акцијата со најголема вредност. За да се овозможи учење на агентот преку пробување на нови акции, воведена е и променлива, која се намалува со тек на време, и овозможува да се избере „случаен“ потег наместо „оптималниот“ потег откриен дотогаш. Веројатноста да се избере случаен потег се намалува со текот на времето.

Иако досега Deep Q-Learning се има покажано како еден од најдобрите алгоритми за поттикнато учење, и се користи и за покомплицирани игри, како Doom [3], Atari [4] и борбени игри за 2 играчи [5], Судоку поседува уникатен проблем при Q-Learning кој е тежок да се надмине.

Во трудот на Анав Мехта [6], тој го користи Q-Learning за повеќе игри за задоволување на услови, поточно 15-Puzzle, Minesweeper, 2048 и Судоку. Иако за другите игри, полесните варијанти стигнувале и до 100% рата на победа, а потешките варијанти се симнуваат до околу 20%, „лесните“ загатки на судоку постигнуваат 7% рата на победа, додека „средно тешките“ постигнуваат 2.1%, а тешките 1.2%. За разлика од 2048, каде некогаш среќата влијае врз победата, Судоку секогаш може да се реши и не зависи од среќа.

*3. Опис на системот, методите, експериментот*

За решавање на проблемот беше првично користен основен Q-Learning алгоритам за решавање на Судоку. Судоку загатките се превземаат од csv документ во формат на „стринг“ со 81 цифра меѓу 0 и 9, каде броевите меѓу 1 и 9 се почетните вредности на загатката, додека полињата со 0 се празни и треба соодветно да се пополнат. На секоја 9та цифра почнува нов ред, односно првиот ред се првите 9 цифри, вториот ред се од 10тата до 18тата цифра итн.

Првично, околината беше претставена како секоја можна состојба на полето и секоја акција можна на даденото поле. Ова е невозможен пристап, затоа што просторот на состојби е неверојатно голем:

9 редици и 9 колони, односно 81 поле, со по 9 можни вредности, во секоја можна комбинација, со секој можен број на вредности кои недостасуваат, односно 10^81 (за да се вклучени и сите полиња кои недостасуваат. Ова секако не е вистинската вредност, затоа што ги игнорира правилата на играта. Вистинскиот број на можни различни Судоку „решенија“, односно крајни вредности, е 6,670,903,752,021,072,936,960. Оваа бројка не вклучува вредности кои фалат, туку само крајни состојби на играта. Гледано вака, просторот на акции е истотака многу голем: Внеси вредност од 1 до 9 во полето (х,у), каде х е вредноста на колоната, а у е вредноста на редицата. Ова води до 81\*9 можни акции, кои иако се многу помалку од 6.67 \* 10^21 состојбите, значително ја зголемува димензионалноста на Q-Learning матрицата. Ова е невозможно за користење поради повеќе причини. Прво, големината на матрицата не овозможува нејзино користење во активна меморија. Второ, за да се пополни Q табелата, потребно е да агентот помине низ секоја состојба барем еднаш, а препорачливо е да ги помине сите состојби голем број пати. За ваков случај, ако ја земаме целата Судоку загатка како состојба, ќе има точен преодреден број на „добри“ акции, и таа состојба нема да е користена за учење повторно. Ова значи дека пристапот ќе мора да се смени.

На агентот ќе му дадеме можност да наместо ја гледа целата Судоку табела, тој е присутен во единечно поле на таа табела. Просторот на акции ќе го смениме во „додади вредност на полето во кое се наоѓаш или подвижи се“, односно [1, 2, 3, 4, 5, 6, 7, 8, 9, горе, доле, лево, десно]. Сега, просторот на акции има големина од 13.

Просторот на состојби истотака ќе го ограничиме, така што единствените податоци што ќе се гледани се за полето во кое агентот се наоѓа, односно вредностите во истиот ред, колона и 3х3 матрица. Затоа што местоположбата на вредностите ја сметаме за неважна, состојбите ќе се различниот број на комбинации кои се содржани на даденото поле, односно комбинација од броевите од 1 до 9. Во примерот, во полето на заокружената 9ка би биле содржани вредностите 3, 6 и 8.

Покрај тоа, ограниченоста на просторот на состојби и акции претставува масовен проблем за решавање на судоку загатка. Поради недостигот на информации, како и немање соодветна реалистична казна за потег, многу е тешко да се постави Судоку како проблем за решавање со DQL.

За решавање на проблемот користена беше невронска мрежа од Keras на Python,

*4. Проблемот при експериментот, и проблемот со Q-Learning за Судоку*

Поради природата на Q-Learning, како и DQL, тие се одлични алати за решавање на проблем со отворен простор на состојби и без лабели за состојби, освен завршена и незавршена. Агент во отворена околина може да биде најдобро оптимизиран со RL алгоритми поради нивната можност да:

1. Се „движат“ во различни состојби кои имаат заеднички одлики.

2. Се најдат во повеќе „различни“ победнички состојби.

Како пример, познатата дигитална загатка 2048[] е игра која ги содржи двете одлики. Агентот во таа игра има многу различни состојби во кои може да се најде, слично како кај Судоку. Но, поради ограничената можност за потези (лизгање на сите коцки горе, доле, лево или десно), како и немање „една точна вредност“ за потег (во Судоку, секоја коцка има само една точна вредност која се состои. Во првиот пример на полето на 9 може да стои само 9), овозможува да Q-Learning алгоритмот извади математички максимални проценти на победа.

Како друг пример, посличен на Судоку, ја имаме дигиталната игра Minesweeper[]. Таа е истотака тешка за Q-Learning поради неверојатниот број на состојби во кои може да биде, како и фактот дека просторот на акции е голем колку самата матрица во играта. Победничката состојба е една, што значи дека е несоодветно за RL алгоритми. Но, покрај тоа, DQL сепак вади подобри резултати за Minesweeper отколку за Судоку, затоа што иако се неверојатно големи просторот на состојби и акции, сепак е полесно да тие се претстават во Q табела.

*5. Резултати од експериментот*

При изведба на DQL мрежата на 500 Судоку загатки за тренирање, со играње на една околина (загатка) по максимум 2000 потега, и максимум 20 реитерации (обновување на состојбата на околината), дојде до крајно успешно решавање на само 30 од тренинг загатките. Тоа е 6% рата на победа. Крајните загатки беа споредени со „веќе решените“ варијанти на соодветната загатка за еднаквост, но решените загатки никако не му помагаа на алгоритмот освен за конечниот done flag.

Принципот на решавање на моделот може да е виден подолу:

Почетна состојба на околината:

[['0' '0' '4' '3' '0' '0' '2' '0' '9']

['0' '0' '5' '0' '0' '9' '0' '0' '1']

['0' '7' '0' '0' '6' '0' '0' '4' '3']

['0' '0' '6' '0' '0' '2' '0' '8' '7']

['1' '9' '0' '0' '0' '7' '4' '0' '0']

['0' '5' '0' '0' '8' '3' '0' '0' '0']

['6' '0' '0' '0' '0' '0' '1' '0' '5']

['0' '0' '3' '5' '0' '8' '6' '9' '0']

['0' '4' '2' '9' '1' '0' '3' '0' '0']]

Почетна состојба на Q-Table-от:

[[0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.]

[0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.]

[0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.]

[0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.]

[0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.]

[0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.]

[0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.]

[0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.]

[0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.]

[0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.]

[0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.]

[0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.]

[0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.]

[0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.]

[0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.]

[0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.]

[0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.]

[0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.]

[0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.]

[0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.]

[0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.]]

Како што може да се примети, Q табелата е преголема за соодветната околина, што значи дека за воопшто да почне да се исполнува, таа бара преголем број на игри.

Вака личи 1500тата епизода на првата итерација:

[['0' '0' '4' '3' '0' '0' '2' '0' '9']

['3' '0' '5' '0' '0' '9' '0' '0' '1']

['3' '7' '0' '0' '6' '0' '0' '4' '3']

['3' '0' '6' '0' '0' '2' '0' '8' '7']

['1' '9' '0' '0' '0' '7' '4' '0' '0']

['0' '5' '0' '0' '8' '3' '0' '0' '0']

['6' '0' '0' '0' '0' '0' '1' '0' '5']

['0' '0' '3' '5' '0' '8' '6' '9' '0']

['0' '4' '2' '9' '1' '0' '3' '0' '0']]

Како што може да се запримети, според табелата, почнува да се исполнува целата табела со вредноста 3 на различни и несоодветни позиции. Тие понатаму се корегираат, но ова е знак за несоодветноста на алгоритмот за дадениот проблем. 95% од Q табелата е сеуште само 0.

*6. Конечен заклучок*

Q-Learning е моќен алгоритам за решавање на проблеми на слободен агент во слични околини со повеќе можни „победнички“, или поволни позиции. Но, за проблеми со затворен и многу менлив простор на состојби и акции, како и со само една победена позиција, Q-Learning е меѓу најнесоодветните алати за решавање на Судоку.

*7. Извори:*

[1] Q-learning - Christopher J. C. H. Watkins & Peter Dayan

[2] Deep Reinforcement Learning with Double Q-Learning. van Hasselt, H., Guez, A., & Silver, D.

[3] Deep Q-learning using redundant outputs in visual doom