რობოტის დევნა

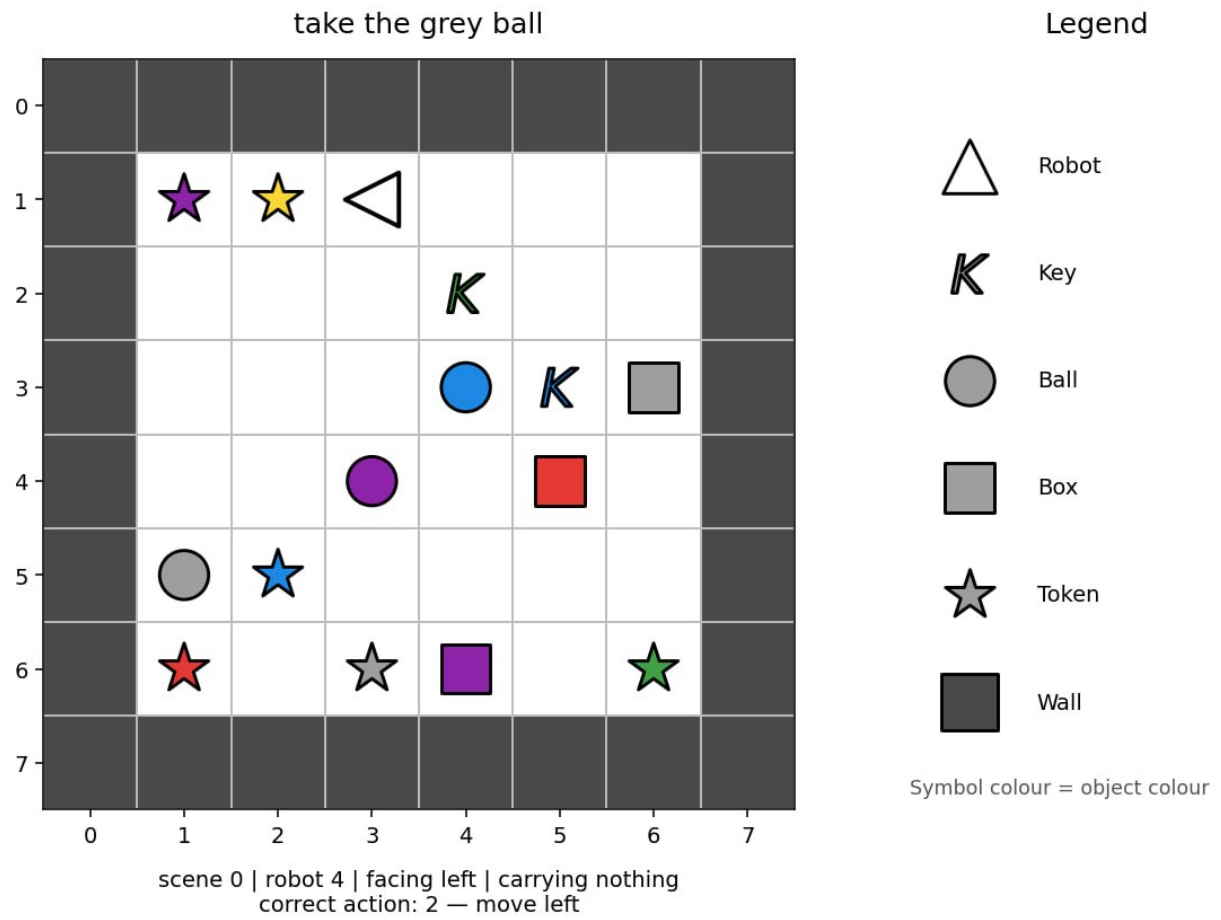
- **დროის ლიმიტი:** 5 წუთი
- **გარემო:** ერთი GPU (≈16 GB VRAM), ინტერნეტის გარეშე
- **ამოხსნის ზომა:** `solution.ipynb` ≤ 1 MB
- **მეხსიერება:** 5 GB

დავალება

სამყაროში სულ ექვსი რობოტია. თითოეული რობოტი ოპერირებს ბადით (grid) წარმოდგენილ პატარა ოთახში. თითოეულ ოთახს აქვს კედლებით გარშემორტყმული 6×6 სათამაშო არე, ასე რომ სრული `image` მასივის ზომა არის 8×8 (სათამაშო არე + კედლები).

თითოეული რობოტი იღებს ინგლისურენოვან ინსტრუქციას, რომელიც აღწერს დავალებას. მომენტალური ფოტო (snapshot) შეიძლება გადაღებულ იქნას ნებისმიერ მომენტში, როდესაც რობოტი ასრულებს მას. თქვენი მიზანია იწინასწარმეტყველოთ რობოტის შემდეგი მოქმედება.

რობოტები ყოველთვის არ მიჰყვებიან ყველაზე მოკლე გზას. რობოტი 0 შეიძლება განსხვავებულად იქცეოდეს, ვიდრე რობოტი 1, მაგრამ თითოეული რობოტი მიჰყვება თავის საკუთარ თანმიმდევრულ პატერნს. გამოიყენეთ საწვრთნელი მაგალითები, რომლებიც მოიცავს სწორ შემდეგ მოქმედებებს, ამ პატერნების შესასწავლად.



არსებობს სამი ტიპის მისია:

- ობიექტთან **მივლა** (go to), მაგალითად `"approach the red ball"`;
- ობიექტის **აღება** (pick up), მაგალითად `"grab the blue key"`;
- **ერთი ობიექტის მეორის გვერდით დადება** (put one object next to another), მაგალითად `"place the red box beside the green ball"`.

ერთი და იგივე ინსტრუქცია შეიძლება რამდენიმენაირად დაიწეროს. ტესტირების სიმრავლე შეიძლება შეიცავდეს ნაცნობი ფრაზების, ფერებისა და ობიექტის ტიპების ახალ კომბინაციებს. თუმცა, ყოველი სიტყვა, ფრაზის პატერნი, ფერი, ობიექტის ტიპი და მისიის ტიპი, რომელიც გამოიყენება ტესტირების სიმრავლეში, ასევე გვხვდება საწვრთნელ სიმრავლეში.

თითოეულ ნიმუშს აქვს შემდეგი ველები:

ველი	მნიშვნელობა
`robot_id`	6 რობოტიდან რომელი რობოტია ეს (`0–5`)
`image`	ოთახი, `8×8×2` მთელი რიცხვების მასივი, სადაც არხი 0 შეიცავს კატეგორიულ object_idx-ს (მაგ., 1=ცარიელი, 2=კედელი, 10=რობოტი) და არხი 1 შეიცავს კატეგორიულ colour_idx-ს (0–5).
`direction`	მიმართულება, საითკენაც რობოტი ამჟამად არის მიმართული
`mission`	ხილული ბუნებრივ-ენოვანი ინსტრუქცია
`carrying`	`null` ან `[object_idx, colour_idx]` ხელში დაჭერილი ობიექტისთვის

სტრიქონები წარმოადგენს დამოუკიდებელ მომენტალურ ფოტოებს (snapshots) შემთხვევითი თანმიმდევრობით. ისინი არ ქმნიან ეპიზოდებს და ევალუაციის დროს არცერთი წინა დაკვირვება ან მოქმედება არ არის ხელმისაწვდომი.

მოწოდებული `visualize_dataset.ipynb` საშუალებას გაძლევთ დაათვალიეროთ მოდელისთვის სხვადასხვა სიტუაციაში ხელმისაწვდომი დაკვირვებები.

გრიდის კოდირება

`image[row][column] = [object_idx, colour_idx]`. პირველი ინდექსი არის სტრიქონი ზემოდან ქვემოთ, ხოლო მეორე — სვეტი მარცხნიდან მარჯვნივ. მასივი მოიცავს კედლის გარეთა საზღვარს, ასე რომ სანავიგაციო შიდა არე არის `6×6`.

ობიექტის id-ები:

id	ობიექტი
1	ცარიელი უჯრა
2	კედელი
5	გასაღები
6	ბურთი
7	ყუთი
10	რობოტი
11	ტოკენი

ტოკენები შეიძლება გამოჩნდნენ ოთახში, მაგრამ ისინი არასოდეს არიან დასახელებულნი მისიებში.

ფერის id-ებია: `0` წითელი, `1` მწვანე, `2` ლურჯი, `3` იასამნისფერი, `4` ყვითელი და `5` ნაცრისფერი. ფერის არხს არ აქვს მნიშვნელობა ცარიელი უჯრებისთვის და კედლებისთვის.

გამოსახულებას აქვს მხოლოდ ზემოთ აღნიშნული ორი არხი. რობოტის მიმართულება მოწოდებულია ერთხელ, ზედა დონის `direction` ველში; ის არ არის დუბლირებული `image`-ის შიგნით.

მოქმედებები

კოდებისთვის `0–3`, გადაადგილების მოქმედებები იყენებს შემდეგ აბსოლუტურ ასახვას:

მოქმედება	მნიშვნელობა
0	ზემოთ გადაადგილება
1	ქვემოთ გადაადგილება
2	მარცხნივ გადაადგილება
3	მარჯვნივ გადაადგილება
4	აღება
5	დაგდება

`direction` ველი მიუთითებს მიმართულების ამჟამინდელ ორიენტაციას შემდეგის გამოყენებით: 0 = ზემოთ (row - 1), 1 = ქვემოთ (row + 1), 2 = მარცხნივ (col - 1), 3 = მარჯვნივ (col + 1).

გადაადგილების მოქმედება ჯერ აბრუნებს რობოტს ამ აბსოლუტური მიმართულებით და შემდეგ ცდილობს მის გადაადგილებას ერთი უჯრით. კედელმა ან ობიექტმა შეიძლება დაბლოკოს გადაადგილება, მაგრამ მიმართულება მაინც იცვლება. `pick up` და `drop` მოქმედებს ექსკლუზიურად მომიჯნავე სამიზნე უჯრაზე, რომელიც განსაზღვრულია მიმართულებით (მაგ., თუ direction=0, ის მოქმედებს (row - 1, col)-ზე).

მონაცემთა სიმრავლე

თქვენ იღებთ ორ საქაღალდეს:

საქაღალდე	სტრიქონები	`labels.json`?	გამოიყენეთ იგი
`dataset/train/`	60,000	მოცემულია	თქვენი მოდელის გასაწვრთნელად
`dataset/test_public/`	3,600	მოცემულია დეველოპმენტის ასლში	თქვენი ფაიფლაინის გასაშვებად და თვითშესაფასებლად

თითოეული საქაღალდე შეიცავს `observations.json`-ს, ზემოთ აღწერილი ნიმუშების JSON სიას. `labels.json` არის მოქმედებების (`0–5`) შესაბამისი JSON სია.

საწვრთნელი სიმრავლე შეიცავს ზუსტად 10,000 სტრიქონს თითოეულ რობოტზე და 20,000 სტრიქონს დავალების თითოეული ოჯახიდან. საჯარო ტესტი შეიცავს 600 სტრიქონს თითოეულ რობოტზე. შეფუთეთ `image` `numpy.asarray(...)`-ით, თუ გჭირდებათ მასივი.

შეფასებისას, `dataset/test_public/` გამჭვირვალედ ჩაენაცვლება იმავე ფორმატის 3,600 დაფარული დაკვირვების სიმრავლეს, მაგრამ `labels.json`-ის გარეშე. საჯარო ლიდერბორდი იყენებს `test_leaderboard_a`-ს; საბოლოო რეიტინგი იყენებს `test_leaderboard_b`-ს. ნოუთბუქი, რომელიც უპირობოდ კითხულობს ტესტის ლეიბლებს, ჩაიშლება. წაიკითხეთ ლეიბლები მხოლოდ `dataset/train/`-დან.

გამოსავალი მონაცემები

ჩაწერეთ `predictions.json` ნოუთბუქის სამუშაო დირექტორიაში. ეს უნდა იყოს JSON სია, რომელიც შეიცავს თითო მთელრიცხოვან მოქმედებას (`0–5`) `dataset/test_public/observations.json`-ის თითოეული სტრიქონისთვის, იმავე თანმიმდევრობით. ჰიპოთეტური ტესტირების სიმრავლისთვის, რომელიც შეიცავს ექვს ნიმუშს, ვალიდური გამოსავალი იქნებოდა:

```
[0, 3, 2, 2, 5, 4]
```

არარსებული ან არავალიდური JSON ფაილი, პროგნოზების არასწორი რაოდენობა, არამთელი მნიშვნელობა ან `{0,1,2,3,4,5}` დიაპაზონს გარეთ არსებული მოქმედება უარყოფილი იქნება ქულის გარეშე.

შეფასება

შეფასება არის **საშუალო სიზუსტე პერ-რობოტის მიხედვით** `0–100` სკალაზე. სიზუსტე ჯერ გამოითვლება დამოუკიდებლად თითოეული რობოტისთვის, ხოლო შემდეგ საშუალოვდება ექვსივე რობოტზე. ამრიგად, თითოეულ რობოტს აქვს თანაბარი წონა.

როგორ წარვადგინოთ

1. გახსენით `solution.ipynb` და გააშვით ყველა უჯრა.
2. დაადასტურეთ, რომ ის წერს `predictions.json`-ს 3,600 პროგნოზით საჯარო ტესტის სიმრავლისთვის.
3. გააუმჯობესეთ მოდელი, თუ გსურთ; მოწოდებული ბეიზლაინი მხოლოდ გვიჩვენებს მოთხოვნილ შეყვანის და გამოყვანის ფორმატს.
4. JupyterLab-ის Git ჩანართში, მოამზადეთ (stage) და დააკომიტეთ (commit) `solution.ipynb`, შემდეგ გააკეთეთ push.
5. დაბრუნდით კონკურსის (Contest) გვერდზე და დააჭირეთ **Submit**.

წარადგინეთ ზუსტად ერთი ფაილი სახელად `solution.ipynb`.