 MESTERSÉGES INTELLIGENCIA

Mesterséges Intelligencia Snake tanulására

Vass Kornél

CCD1VK

(GKNB\_INTM002)

2023/2024/2

Tartalom

[Használati útmutató 3](#_Toc160701652)

[Probléma bemutatása 4](#_Toc160701653)

[Használt könyvtárak, csomagok 5](#_Toc160701654)

[Anaconda 5](#_Toc160701655)

[Pygame 5](#_Toc160701656)

[Pytorch 5](#_Toc160701657)

[Matplotlib 5](#_Toc160701658)

[Ipython 5](#_Toc160701659)

[Játék bemutatása 6](#_Toc160701660)

[Az alapok 6](#_Toc160701661)

[Inicializálás 6](#_Toc160701662)

[play\_step függvény szerkezete 7](#_Toc160701663)

[Mesterséges intelligencia bemutatása 8](#_Toc160701664)

[Tesztek 9](#_Toc160701665)

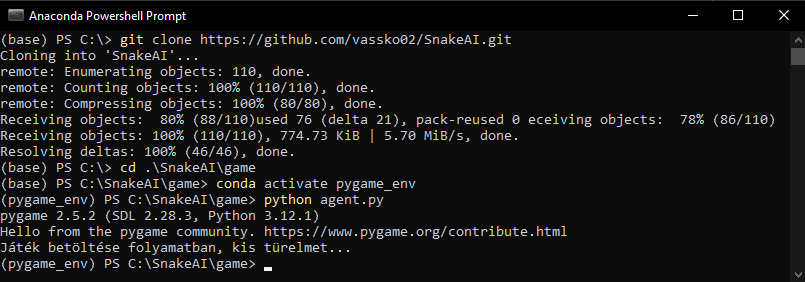
[Források 10](#_Toc160701666)

# Használati útmutató

A GitHub repository, amelyben dolgoztam: <https://github.com/vassko02/SnakeAI.git>

1. Telepítsük az Anaconda-t, az alábbi linkről: <https://www.anaconda.com/download>
2. Indítsuk el az Anaconda-t és egy „Anaconda Powershell Prompt” -ot
3. Klónozzuk a GitHub repository-t a „git clone <https://github.com/vassko02/SnakeAI.git>” parancs segítségével
4. Lépjünk be a GitHub-ról letöltött fájlba a „cd SnakeAI/game
5. Hozzuk létre egy pygame környezetet: „conda create -n pygame\_env python=3.12”
6. Adjuk ki a következő parancsot: „conda activate pygame\_env”
7. Telepítsük a szükséges csomagokat:

* pip install pygame
* pip install torch torchvision
* pip install matplotlib ipython

1. Indítsuk el az AI-t a „python agent.py” utasítással
2. (Miután elindult a játék, az ablakok mozgatása, áthelyezése összeomlást okozhat, erre próbáltam megoldást keresni, de sajnos nem találtam)

1. ábra Minta a kiadott parancsokról (5. és 7. lépés nélkül)

# Probléma bemutatása

Az ember gyakran azt gondolná, hogy egy egyszerű játék, mint a Snake, nem okozhat különösebb problémákat, de mégis, ha nem sikerül elérni egy magasabb pontszámot, könnyen felmerülhet a frusztráció. A kérdés, hogyan lehet konzisztensen magas pontszámot elérni egy ilyen játékban, elég bonyolult lehet. Az egyik lehetséges megoldás egy mesterséges intelligencia fejlesztése, amely képes folyamatosan fejlődni a játékban, és így egyre magasabb pontszámokat ér el.

Amint az AI megfelelően ki lett fejlesztve és tanítva, felmerülnek további kérdések is. Például, mennyi időbe telik az AI-nak az első élelem megtalálása, és mennyi idő alatt éri el a száz pontot? Hány próbálkozás után lehet biztos abban, hogy az elért pontszám nem fog ötven alá esni? Ezek a válaszok nagyon sok tényezőtől függenek. Lényeges például az, hogy milyen paraméterekkel és algoritmusokkal rendelkezik az AI, illetve mennyi időt és erőforrást szánunk a fejlesztésre és tanításra. Ezekre a kérdésekre a [tesztek című fejezetben](#_Tesztek) visszatérünk.

A válaszok keresése és a mesterséges intelligencia továbbfejlesztése során rengeteg kihívással és lehetőséggel találhatjuk magunkat szemben. Fontos, hogy nyitottak maradjunk az új megközelítések és technikák felé, és folyamatosan tanuljunk a fejlesztési folyamat során felmerülő kihívásokból. Ez lehetővé teszi, hogy egyre hatékonyabb és kifinomultabb AI-t hozzunk létre, amely képes megoldani az előtte álló problémákat és elérni a kitűzött célokat a Snake játékban.

# Használt könyvtárak, csomagok

## Anaconda

Az Anaconda egy olyan python alapú program, amelyet gépi tanulásra és adat elemzésre specifikáltak. Több előre telepített csomagot is tartalmaz, például: [matplotlib](#_Matplotlib), [ipython](#_Ipython), vagy pandas. Ez a legnagyobb fő előnye a programnak, amellett, hogy van egy saját csomagkezelője a „Conda”, aminek a segítségével további csomagokhoz tudunk hozzáférni. Mindezek mellett az Anaconda egy biztos környezetet biztosít a programok futtatására, így biztosak lehetünk, hogy a programunk minden eszközön ugyanúgy fut, ahol telepítve van a program.[1]

## Pygame

A Pygame több platformfüggetlen Python modul összessége, amelyet videójátékok készítésére használnak. Tartalmaz számítógépes grafika és hangkönyvtárakat, amelyeket a Python programozási nyelvvel használhatunk. A Pygame hivatalosan Pete Shinners által íródott, hogy leváltsa a PySDL-t. A Pygame alkalmas kliensoldali alkalmazások létrehozására, amelyek potenciálisan önálló végrehajtható fájlba csomagolhatók.[2]

## Pytorch

PyTorch egy nyílt forráskódú gépi tanulási (ML) keretrendszer, amely a Python programozási nyelvre és a Torch könyvtárra épül. A Torch egy nyílt forráskódú ML könyvtár, amelyet mély neurális hálózatok létrehozására használnak, és a Lua szkriptelési nyelven íródott. Ez az egyik preferált platform a mély tanulási kutatásokhoz. A keretrendszer azért lett kifejlesztve, hogy felgyorsítsa a kutatási prototípusok és a bevezetés közötti folyamatot.[3]

## Matplotlib

Matplotlib egy adatvizualizációs könyvtár Pythonban, amelyet 2D plotokhoz használnak. Matplotlib egy többplatformos adatvizualizációs könyvtár, amelyet NumPy tömbökön alapul, és tervezték a szélesebb körű SciPy veremmel való együttműködésre. John Hunter vezette be 2002-ben. Az egyik legnagyobb előnye a vizualizációnak, hogy lehetővé teszi számunkra a hatalmas adatmennyiség vizuális hozzáférését könnyen fogyasztható ábrák formájában. A Matplotlib többféle diagramot is kirajzolhat, mint például vonaldiagram, oszlopdiagram, szórásdiagram, hisztogram.[4]

## Ipython

Az IPython egy olyan fejlett eszköz, amely számos előnyt kínál a hagyományos Python-hoz képest. Kiemelkedő tulajdonságai közé tartozik egy erőteljes interaktív Python shell, amely lehetővé teszi a gyors és hatékony kódírást és végrehajtást. A beépített objektum-introspekció funkció lehetővé teszi az objektumok tulajdonságainak ellenőrzését futásidőben, ami rendkívül hasznos a fejlesztés során. Az IPython továbbá támogatja a szintaxiskiemelést, tárolja az interakciók történetét, és lehetővé teszi a tabulációval történő kulcsszó, változó és függvénynevek kiegészítését. A varázslatparancs rendszer segítségével könnyen irányítható a Python környezet, és végrehajthatók operációs rendszer feladatok. Az IPython továbbá beágyazható más Python programokba, és hozzáférést biztosít a Python hibakeresőhöz, amely segít a hibák azonosításában és javításában a fejlesztés során.[5]

# Játék bemutatása

Maga a játék a GitHubon a [game.py](game/game.py) fájlban található. Ez tartalmazza magát a játékot, ez nem azt jelenti, hogy ha ezt a fájlt futtatnánk akkor manuális irányítással működne a játék, mivel az AI miatt módosításokat kellett végezni a fájlban, például az irányítás részét teljesen át kellett alakítani mert nyilvánvalóan nem billentyűzetről fogja kapni a bemenetet a program.

## Az alapok

Az alábbi képen a game.py fájl első pár sora látható, ahol a csomagok kerültek beimportálásra és a konstansok kerültek definiálásra. Az irányokat egy enum-ban tároltam, amit akkor használnak, ha egy változónak egy meghatározott értékkészlete legyen, mint most esetünkben az irány. Ezután létrehoztam a Point nevű „namedtuple”-t aminek a segítségével majd a későbbiekben a megjelenített ablak adott pontjaira tudok hivatkozni. Majd definiáltam a kígyó sebességét és a blokkok méretét 20 pixelre, röviden ez annyit takar, hogy felbontottam az ablakot 20\*20-as négyzetekre. Végül pedig színeket definiáltam, ezeket a konstansokat használtam a kígyó, a keret és a háttér színezéséhez. Ezen adatok rögzítése után pedig a játékhoz használt függvényeket hoztam létre.

## Inicializálás

Először inicializálni kell a programot. Ez esetünkben annyit takar, hogy létrehozzuk az ablakot, ahol a játék futni fog, majd az ablak közepére tesszük a kígyót, úgy, hogy jobbra induljon el, majd lehelyezünk egy ételt. Magát a kígyót egy Point-okból álló listában tároltam, aminek a nulladik eleme a kígyó feje utána pedig a folyamatosan növekvő test. Ehhez egy következő függvényt hívunk meg amely randomizál, egy képet, hogy melyik gyümölcs kerüljön az ablakra és két számot, amelyek x és y koordinátaként működnek. Fontos, hogy ezek mindig olyan számok lesznek, amelyek oszthatóak a 2. ábrán látható BLOCK\_SIZE-zal. Ha a random pont a kígyóban van akkor új pontot csinál a függvény és azt helyezi le.

2. ábra Kódrészlet a game.py fájlból

Inicializálás után kezdődik a játék, ami egy több lépcsős függvényből áll, amely a kígyó haláláig ismétlődik, majd, ha ez bekövetkezik akkor az előző bekezdésben bemutatott függvény segítségével visszaállítjuk a játékot a kezdeti helyzetbe.

## play\_step függvény szerkezete

* **Bemenet kezelése**

Mivel a programot nem manuálisan vezéreljük hanem AI irányítja ezért ebben a lépésben csak az ablak bezárását kell kezelni, azaz, ha az ablakot bezárjuk akkor a teljes programot le kell állítani.

* **Mozgás**

A mozgás levezényléséhez először azt kell megvizsgálnunk, hogy van-e irányváltozás az előző frame óta. Ha van akkor a Directions enum segítségével megváltoztatjuk az irányt. Végül pedig a blokk méretével eltoljuk a kígyót a megfelelő irányba, úgy, hogy a fejét beszúrjuk a lista elejére.

* **Játék végének vizsgálata**

Egy adott körnek akkor van vége, ha a kígyó meghal, ez akkor következik be, ha beleütközik a falba vagy saját magába. Emellett a játéknak akkor is vége van, ha a kígyó huzamosabb ideig nem ér el semmit. Ha ezek valamelyike bekövetkezik akkor a függvény negatív 10-et ad vissza jutalomként, és igaznak állítja be a game\_over változót.

* **Új étel elhelyezése**

Ennél a résznél azt kell megvizsgálni, hogy a kígyó összeszedte-e a lehelyezett ételt, ha igen akkor a jutalmat 10-re állítjuk és növeljük a pontszámot eggyel, ezzel a kígyó hossza is nő. Ha nem vesz fel ételt az adott frame-ben a kígyó akkor az utolsó blokkját eltávolítjuk. Erre azért van szükség mert a „Mozgás” résznél beszúrtuk a fejet a lista elejére, de azért, hogy ne nőjön a kígyó hossza mikor nem szed össze gyümölcsöt el kell távolítanunk az utolsó blokkot a listából.

* **UI frissítése és a sebesség növelése**

A UI frissítése minden frame-ban lefut. Letisztítja az ablakot, majd újra rajzol rá mindent, beleértve a változásokat. Először feltölt zölddel az egészet, majd a keretet helyezi el. Ezután a kígyó és az étel kerül kirajzolásra. Végül pedig a pontszám felirata. Ha ez sikeresen frissült utána pedig a sebességet növeltem, az alap sebesség 35, ez mindig az adott pontszámmal van növelve.

* **Érték visszaadás**

A ciklus végén több értéket is visszaad a függvény, először a jutalmat, majd azt, hogy vége van-e a játéknak, majd az elért pontszámot. Ezt az agent kapja meg ami ezektől az információktól függően folytatja a játékot.

# Mesterséges intelligencia bemutatása

# Tesztek

A [probléma bemutatása fejezetben](#_Probléma_bemutatása) felsoroltam pár „mérföldkövet” az AI életében. Háromszor harminc percen keresztül futtattam a mesterséges intelligenciát, ezen és további statisztikák felderítése érdekében. A használt ablakméret 640\*480, amiből minden oldalon 20 pixel a keret. Nem akartam nagyobb ablakot készíteni mert akkor az elején jóval tovább tartana míg megtalálja a kígyó az ételt, és ez nagyban lelassítaná a tanulási folyamatot

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Vizsgált adat | 1. mérés | 2. mérés | 3. mérés | Átlag |
| Hány kör kellett az első étel összeszedéséhez? | 5 | 2 | 6 | 4.33 |
| Hány kör kellett 20 étel összeszedéséhez? | 75 | 84 | 73 | 77.33 |
| Hányadik kör volt az utolsó 0 végeredménnyel? | 72 | 68 | 67 | 69 |
| Hány kör futott le 30 perc alatt? | 253 | 252 | 254 | 253 |
| Mi lett a rekord 30 perc után? | 60 | 62 | 77 | 66.33 |
| Mennyi lett az átlagos eredmény 30 után? | 22.04 | 20.98 | 21.59 | 64.61 |

A három teszt figyelmen követése után azt vettem észre, hogy az AI viszonylag gyorsan alkalmazkodik a falakhoz, és egy adott ponttól kezdve csak az vet véget a játéknak mikor magába ütközik. Úgy gondolom, hogy további futtatás után ezt is megtanulná kiküszöbölni az AI. A diagrammot vizuálisan szakaszokra tudtam osztani, az első része az volt amikor még falba ütközött, ez egy ponton abba maradt, és onnantól kezdve kilőtt a teljesítmény. Véleményem szerint további futtatásnál ismét kilőne ez, ha eljut a következő probléma megoldásához. A tesztek futtatása óta lecsökkentettem a gyorsulás sebességét az egyszerűbb követhetőség érdekében.

# Források

[1] „Anaconda vs Python in Software: Know the Key Differences | Simplilearn”, Simplilearn.com. Elérés: 2024. március 4. [Online]. Elérhető: https://www.simplilearn.com/anaconda-vs-python-article

[2] „Pygame Tutorial - javatpoint”, www.javatpoint.com. Elérés: 2024. március 4. [Online]. Elérhető: https://www.javatpoint.com/pygame

[3] „What is PyTorch?”, Enterprise AI. Elérés: 2024. március 4. [Online]. Elérhető: https://www.techtarget.com/searchenterpriseai/definition/PyTorch

[4] „Introduction to Matplotlib”, GeeksforGeeks. Elérés: 2024. március 4. [Online]. Elérhető: https://www.geeksforgeeks.org/python-introduction-matplotlib/

[5] „IPython - Introduction”. Elérés: 2024. március 4. [Online]. Elérhető: https://www.tutorialspoint.com/jupyter/ipython\_introduction.htm