Általános információk

A diplomaterv szerkezete:

1. Diplomaterv feladatkiírás
2. Címoldal
3. Tartalomjegyzék
4. A diplomatervező nyilatkozata az önálló munkáról és az elektronikus adatok kezeléséről
5. Tartalmi összefoglaló magyarul és angolul
6. Bevezetés: a feladat értelmezése, a tervezés célja, a feladat indokoltsága, a diplomaterv felépítésének rövid összefoglalása
7. A feladatkiírás pontosítása és részletes elemzése
8. Előzmények (irodalomkutatás, hasonló alkotások), az ezekből levonható következtetések
9. A tervezés részletes leírása, a döntési lehetőségek értékelése és a választott megoldások indoklása
10. A megtervezett műszaki alkotás értékelése, kritikai elemzése, továbbfejlesztési lehetőségek
11. Esetleges köszönetnyilvánítások
12. Részletesés pontos irodalomjegyzék
13. Függelék(ek)

Felhasználható a következő oldaltól kezdődő Diplomaterv sablon dokumentum tartalma. Ügyeljen a konzulens nevét és a beadás évét jelölő szövegdobozokra, mert azokra külön ki kell adni a frissítést. A mezők tartalma a sablonban a dokumentum adatlapja alapján automatikusan kerül kitöltésre.

A diplomaterv szabványos méretű A4-es lapokra kerüljön. Az oldalak tükörmargóval készüljenek (mindenhol 2.5cm, baloldalon 1cm-es kötéssel). Az alapértelmezett betűkészlet a 12 pontos Times New Roman, másfeles sorközzel.

Minden oldalon - az első négy szerkezeti elem kivételével - szerepelnie kell az oldalszámnak.

A fejezeteket decimális beosztással kell ellátni. Az ábrákat a megfelelő helyre be kell illeszteni, fejezetenként decimális számmal és kifejező címmel kell ellátni. A fejezeteket decimális aláosztással számozzuk, maximálisan 3 aláosztás mélységben (pl. 2.3.4.1.). Az ábrákat, táblázatokat és képleteket célszerű fejezetenként külön számozni (pl. 2.4. ábra, 4.2 táblázat vagy képletnél (3.2)). A fejezetcímeket igazítsuk balra, a normál szövegnél viszont használjunk sorkiegyenlítést. Az ábrákat, táblázatokat és a hozzájuk tartozó címet igazítsuk középre. A cím a jelölt rész alatt helyezkedjen el.

A képeket lehetőleg rajzoló programmal készítsék el, az egyenleteket egyenlet-szerkesztő segítségével írják le.

Az irodalomjegyzék szövegközi hivatkozása történhet a Harvard-rendszerben (a szerző és az évszám megadásával) vagy sorszámozva. A teljes lista névsor szerinti sorrendben a szöveg végén szerepeljen (sorszámozott irodalmi hivatkozások esetén hivatkozási sorrendben). A szakirodalmi források címeit azonban mindig az eredeti nyelven kell megadni, esetleg zárójelben a fordítással. A listában szereplő valamennyi publikációra hivatkozni kell a szövegben. Minden publikáció a szerzők után a következő adatok szerepelnek: folyóirat cikkeknél a pontos cím, a folyóirat címe, évfolyam, szám, oldalszám tól-ig. A folyóirat címeket csak akkor rövidítsük, ha azok nagyon közismertek vagy nagyon hosszúak. Internet hivatkozások megadásakor fontos, hogy az elérési út előtt megadjuk az oldal tulajdonosát és tartalmát (mivel a link egy idő után akár elérhetetlenné is válhat), valamint az elérés időpontját.

Fontos:

* a szakdolgozat készítő/diplomatervező nyilatkozata (a jelen sablonban szereplő szövegtartalommal) kötelező előírás Karunkon, ennek hiányában a szakdolgozat/diplomaterv nem bírálható és nem védhető!
* mind a dolgozat, mind a melléklet maximálisan 15 MB méretű lehet!

Jó munkát, sikeres szakdolgozat készítést ill. diplomatervezést kívánunk!

FELADATKIÍRÁS

A feladatkiírást a **tanszék saját előírása szerint** vagy a tanszéki adminisztrációban lehet átvenni, és a tanszéki pecséttel ellátott, a tanszékvezető által aláírt lapot kell belefűzni a leadott munkába, vagy a tanszékvezető által elektronikusan jóváhagyott feladatkiírást kell a Diplomaterv Portálról letölteni és a leadott munkába belefűzni (ezen oldal HELYETT, ez az oldal csak útmutatás). Az elektronikusan feltöltött dolgozatban már nem kell megismételni a feladatkiírást.



Budapesti Műszaki és Gazdaságtudományi Egyetem

Villamosmérnöki és Informatikai Kar

Fehér Gergő

Versenyautó pályaívének tervezése mesterséges intelligencia módszerekkel

Konzulens

BUDAPEST, 2017

Tartalomjegyzék

[Összefoglaló 3](#_Toc332798843)

[Abstract 4](#_Toc332798844)

[1 Bevezetés 4](#_Toc332798845)

[1.1 Formázási tudnivalók 4](#_Toc332798846)

[1.1.1 Címsorok 4](#_Toc332798847)

[1.1.2 Képek 4](#_Toc332798848)

[1.1.3 Kódrészletek 4](#_Toc332798849)

[1.1.4 Irodalomjegyzék 4](#_Toc332798850)

[2 Utolsó simítások 4](#_Toc332798851)

[Irodalomjegyzék 5](#_Toc332798852)

[Függelék 5](#_Toc332798853)

Hallgatói nyilatkozat

Alulírott **Fehér Gergő**, szigorló hallgató kijelentem, hogy ezt a szakdolgozatot meg nem engedett segítség nélkül, saját magam készítettem, csak a megadott forrásokat (szakirodalom, eszközök stb.) használtam fel. Minden olyan részt, melyet szó szerint, vagy azonos értelemben, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

Hozzájárulok, hogy a jelen munkám alapadatait (szerző(k), cím, angol és magyar nyelvű tartalmi kivonat, készítés éve, konzulens(ek) neve) a BME VIK nyilvánosan hozzáférhető elektronikus formában, a munka teljes szövegét pedig az egyetem belső hálózatán keresztül (vagy hitelesített felhasználók számára) közzétegye. Kijelentem, hogy a benyújtott munka és annak elektronikus verziója megegyezik. Dékáni engedéllyel titkosított diplomatervek esetén a dolgozat szövege csak 3 év eltelte után válik hozzáférhetővé.

Kelt: Budapest, 2017. 11. 04.

...…………………………………………….

Fehér Gergő

Összefoglaló

Az utóbbi években a számítási kapacitás növekedésével egyre nagyobb teret nyernek gépi tanuló algoritmusok és az ún. mély tanulás, amivel a kutatók a műszaki problémák egyre több területén tudnak eddig még soha nem látott sikereket elérni. Ezek az algoritmusok ma már igen összetett problémákat is meg tudnak oldani emberi vagy sok esetben akár annál jobb szinten is, mint például a képfelismerés vagy akár olyan komplex játékokkal játszani, mint a Go.

Jelen szakdolgozat témája, ezen algoritmusoknak az irányítástechnikában való lehetséges alkalmazása, azon belül egy versenyautó modell irányításnak elméleti megvalósítása. Ennek megvalósításához a mély neurális hálózatokat, illetve megerősítéses tanuláson alapuló algoritmusokat használtam fel.

A szakdolgozat során először egy egyszerűbb példán összehasonlítottam Matlabban és Pythonban az algoritmus implementáláshoz szükséges eszköztárakat. Ezután implementálásra került a feladat megvalósításához használt keretrendszer ahol grafikus felületen tudjuk beállítani a környezet és az autó paramétereit, illetve nyomon tudjuk követi az algoritmus tanulásának előre haladását is.

Abstract

Ide jön a ½-1 oldalas angol nyelvű összefoglaló, amelynek szövege a Diplomaterv Portálra külön is feltöltésre kerül.

# Bevezetés

Manapság talán az egyik legfelkapottabb kutatási területnek a mesterséges intelligencia kutatása számít. Bár az alap ötletek és algoritmusok már régóta léteznek, sokáig hiányzott a tudósok mögül a számítási kapacitás, így a kezdeti lendület hamar alább hagyott. Viszont a modern több magos CPU-k és a GPU-k számítási kapcíásának köszönhetően a kétezres évek elején új lökést kapott a terület. Bár amit ma mesterséges inteligenciának hívunk, az még nagyon messze áll attól amit egy laikus maga eléképzel, viszont bizonyos területekre, problémákra fokuszáltan a modern algoritmusok már sokszor túlszárnyalják az ember teljesítményét is. Legyen az képeken objektumok esetleg emberek felismerése vagy olyan bonyolult játékokkal való játék mint a go. Minden esetre a kifejlesztett algoritmusokat nagyon sok féle probléma területen lehet alkalmazni. A dolgozatomban én egy irányítástechnikai probléma alternatív megoldását tűztem ki célul az később részletezett módszerekkel. A feladatom egy versenyautó modell irányítása a gépi tanulás egy válfajának a megerősítéses tanulás (Reinforcement learning) segítségével olyan formában, hogy az az általa elképzelt szuboptimális íven haladjon végig a pályán, anélkül, hogy közben a pálya szélének vagy az esetlegesen a pályán található akadályoknak ütközzön, mindezt minél rövidebb idő alatt. Természetesen erre léteznek más gépi tanulást nem alkalmazó algoritmusok is, viszont a feladatot egy kicsit megváltoztatva ezek már részletesebb tervezést és nagyon komplex algoritmusokat eredményeznek.

A dolgozatom három fő részből áll. Az első részben a későbbi feladathoz használt algoritmusok elméleti háttere kerül bemutatásra, kitérve a később implementált algoritmusokra is. A második részben egy egyszerűbb gyakorlati példán kerülnek bemutatásra az alkalmazott módszerek, a megerősítéses tanulás és a mély tanulás (Deep learning). Ezután ismertetem a konkrét feladat megvalósítását. A szimulációk és a tanítás futtatásához implementált keretrendszert, a környezet és az autó fizikai modelljét, az alkalmazott algoritmusokat és az azokkal elért eredményeket.

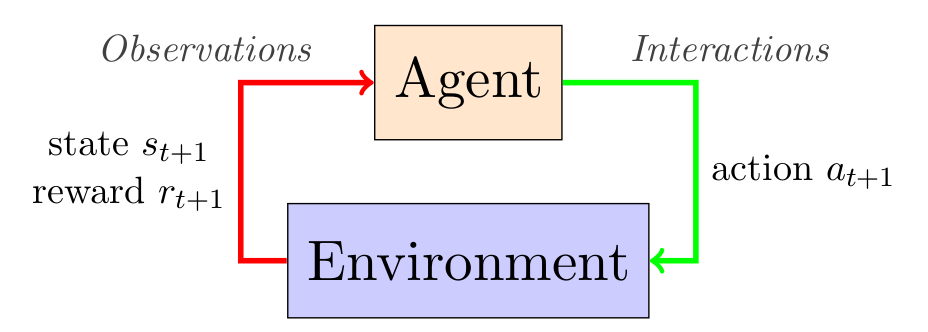
# Elméleti háttér

## Megerősítéses tanulás

A megerősítéses tanulás a gépi tanulásnak egy olyan válfaja, melyet a viselkedés pszichológia ihletett. Arra a problémára keresi a megoldást, hogy egy ágens az adott környezetben milyen döntéseket hozzon annak érdekében, hogy az azért kapott kumulált jutalmat maximalizálja. A szituációt az ún markov döntési folyamattal (MDP,Markovian Decision Process) modellezik.

### Markov döntési folyamat

A vizsgált folyamatok modellezhetőek az ún MDP-vel. Ez egy döntéshozatalok modellezésére létrehozott matematikai modell, melynek a determinisztikus változata négy fő elemből áll. Az adott környezetben lévő állapotunk S, az S állapotban hozott döntésünk A, az ezért kapott jutalom R és ennek hatására az új állapot.



. Markov döntési folyamat

Az ágensnek van egy π(s) stratégiája (policy) amelyet követve minden s állapothoz hozzárendel egy a döntést (action). Az a döntést meghozva kap a környezettől egy r jutalmat és az s’ következő állapotba jut. A jutalmat előre nem ismeri, ezeket a döntéseket meghozva térképezi fel. Ebben a kontextusban minden állapothoz rendelhetünk egy ún V(s) állapot érték függvényt (state value), ami azt mondja meg, hogy az adott s állapotból az π(s) stratégiát követve mennyi a várható diskontált kummulált jutalom amit el fog érni.

Ahogy a való életben az ágensünk számra is kevésbé értékes a jövőbeni jutalom, így azok értékét a számításkor egy γ értékkel diszkontáljuk. Definiálhatjuk továbbá az ún. Q(s,a) döntés érték függvényt (action value), ami azt adja meg,hogy az s álapotból a döntés után a π(s) stratégiát követve mennyi a várható diskontált kummulált jutalom. Elméletben ha ismerjük ezt a Q(s,a) függvényt minden állapotra és döntésre, akkor a kummulált jutalom maximalizáláshoz egyszerűen minden állapotban követjük mindig azt a döntést ahol a Q(s,a) függvény maximális. A probléma viszont, hogy a valóságban ezt szinte sohasem ismerjük, viszont szerencsére az egyes lépések után a kapott jutalom és a következő állapot megfigyelése után a Bellman egyenlet segítségével

### Algoritmusok kategorizálása

Action value based

Policy based

Actor Critic

Q learning

Reinforce

## Neurális hálók

## Deep reinfocement learning

### DDPG

### DDQN

# Megerősítéses tanulás a gyakorlatban

## Kő-papír-olló megerősítéses tanulással

## Kő papír-olló neurális hálóval

## Értékelés

# Versenyautó irányításának megvalósítása

## Keretrendszer

## Modell

## Implementált algoritmusok

## Eredmények értékelése

## Lehetséges tovább fejlesztések

A következő fejezet pár példán keresztül bemutatja a diplomatervekben és szakdolgozatokban szokásosan előkerülő formázások megvalósítását.

## Formázási tudnivalók

A dokumentum folyószövegéhez használjuk a **Normál** (angol Word esetén Normal) stílust.

### Címsorok

A fejezetcímek esetén a **Címsor 1-4** (Heading 1-4) stílusokat használjuk.

### Képek

A képhez használjuk a **Kép** stílust.

Képaláírást a képen jobb gombbal kattintva a Képaláírás beszúrása… opcióval adhatunk hozzá, így az automatikusan **Képaláírás** (Caption) stílusú lesz.



.. ábra: Példa képaláírásra

### Kódrészletek

Kódrészletek beillesztése esetén használjuk a **Kód** stílust.

using System;

namespace MyApp

{

class Program

{

static void Main( string[] args )

{

Console.WriteLine( "Szia Világ!" );

}

}

}

### Irodalomjegyzék

Az Irodalomjegyzékben szereplő hivatkozásokat **Irodalomjegyzék sor** stílussal formázzuk, a címüket pedig **Irodalomjegyzék forrás** stílussal emeljük ki.

A szövegbe a hivatkozásokat a Kereszthivatkozás beszúrása (Insert cross-reference) funkcióval helyezzük el (példa egy így beszúrt hivatkozásra: [1]), így azok automatikusan frissülnek a hivatkozások átrendezésekor.

# Utolsó simítások

Miután elkészültünk a dokumentációval, ne felejtsük el a következő lépéseket:

* Kereszthivatkozások frissítése: miután kijelöltük a teljes szöveget (Ctrl+A), nyomjuk meg az F9 billentyűt, és a Word frissíti az összes kereszthivatkozást. Ilyenkor ellenőrizzük, hogy nem jelent-e meg valahol a "Hiba! A könyvjelző nem létezik." szöveg.
* Dokumentum tulajdonságok megadása: a dokumentumhoz tartozó meta adatok kitöltése (szerző, cím, kulcsszavak stb.). Erre való a Dokumentum tulajdonságai panel, mely a Fájl / Információ / Tulajdonságok / Dokumentumpanel megjelenítése úton érhető el.
* Kinézet ellenőrzése PDF-ben: a legjobb teszt a végén, ha PDF-et készítünk a dokumentumból, és azt leellenőrizzük.

Irodalomjegyzék

1. Levendovszky, J., Jereb, L., Elek, Zs., Vesztergombi, Gy.: Adaptive statistical algorithms in network reliability analysis, Performance Evaluation - Elsevier, Vol. 48, 2002, pp. 225-236
2. National Istruments: LabVIEW grafikus fejlesztői környezet leírása, <http://www.ni.com/> (2010. nov.)
3. Fowler, M.: UML Distilled, 3rd edition, ISBN 0-321-19368-7, Addison-Wesley, 2004
4. Wikipedia: Evaluation strategy, <http://en.wikipedia.org/wiki/Evaluation_strategy> (revision 18:11, 31 July 2012)

Függelék