

Budapesti Műszaki és Gazdaságtudományi Egyetem

Villamosmérnöki és Informatikai Kar

Szilágyi Krisztián Gergely

Autonóm jármű fejlesztése

Buggy projekt

Konzulens

Szemenyei Márton

BUDAPEST, 2020

Tartalomjegyzék

[Összefoglaló 3](#_Toc41478180)

[1 Irodalomkutatás 4](#_Toc41478181)

[1.1 Reinforcment Learning 4](#_Toc41478182)

[1.2 PyBullet 4](#_Toc41478183)

[2 Felhasznált technológia 5](#_Toc41478184)

[2.1 Colaboratory 5](#_Toc41478185)

[2.2 PyTorch 6](#_Toc41478186)

[3 Architektúra 7](#_Toc41478187)

[3.1 Felépítése 7](#_Toc41478188)

[3.1.1 Environments 8](#_Toc41478189)

[3.1.2 Multi-head attention 8](#_Toc41478190)

[3.1.3 A2C 8](#_Toc41478191)

[3.1.4 RAdam 8](#_Toc41478192)

[3.2 Tesztelés, eredmények 9](#_Toc41478193)

[Irodalomjegyzék 10](#_Toc41478194)

Összefoglaló

A feladatom egy az Irányítástechnika és Informatika Tanszéken fejlesztett autonóm BB-8 droid beszéd felismerésének és hangvezérlésének megalkotása az Önálló laboratórium tárgy keretein belül. A feladat komplexitása miatt a második pontra, vagyis a parancsok teljesítésére, valamint a megalkotott rendszer beágyazására nem jutott idő ebben a félévben. A beszámolómban időrendi sorrendben mutatom be a feladatom elvégzését, a végén leírom a távlati célokat is, amelyeket a következő félévben kell teljesíteni.

A robot a szenzorjai és feldolgozó egységének köszönhetően rengeteg funkcióval rendelkezik, egy több fős csapat dolgozik ezeknek a funkcióknak az implementálásán. Az én feladatom a hang felismerés és hangvezérlés. Ehhez segítségünkre van négy mikrofon a robot fejegységében, valamint egy NVIDIA Titan X, ami a nagy teljesítményének köszönhetően gyorsabb fejlesztést tesz lehetővé, hiszen a neurális hálók paramétereinek beállítása (a tanítás) rendkívül számításigényes éppen ezért időigényes folyamat. Egy komolyabb struktúra betanítása akár napokban mérhető. Nálunk inkább percekben vagy órákban még a komoly teljesítmény ellenére is.

Alapvetően a feladat egy speech-to-text (STT) működés létrehozása neurális háló segítségével. Felügyelt (supervised) tanítást alkalmazva kéne létrehoznunk egy osztályozót, amely a bemenetére kapott hangból, vagyis amit feldolgoznak a mikrofonok, képez egy karakterláncot és eldönti, hogy ez érvényes utasítás-e és ha igen, melyik utasításnak felel ez meg. Ennek a további feldolgozása, végrehajtása már a csapat más tagjainak a feladata. Mivel nulláról létrehozni egy mély neurális hálót (DNN) túl nagy feladat, ahhoz, hogy egy félév alatt erre sor kerüljön, ezért úgynevezett transfer learning-et fogunk alkalmazni. Ennek lényege, hogy egy nagy adatbázissal előre betanított, kész hálót használunk fel és finom hangoljuk a saját adatainkkal.

# Irodalomkutatás

A projektet megvalósítása egy hosszútávú cél, így ebben a félévben a projekt szkópja főleg az irodalomkutatás és a tervezés volt, nem a megvalósítás és tesztelés. Így a hónapokat leginkább tanulással töltöttem, megismerkedtem a megerősítéses tanulásban használatos fogalmakkal és ezek alapján a cél volt összerakni egy kezdeti architektúrát, melyet a későbbiekben továbbfejlesztve megvalósíthatunk egy teljesen autonóm módon működő jármű szoftverét.

## Reinforcment Learning

## PyBullet

# Felhasznált technológia

## Colaboratory

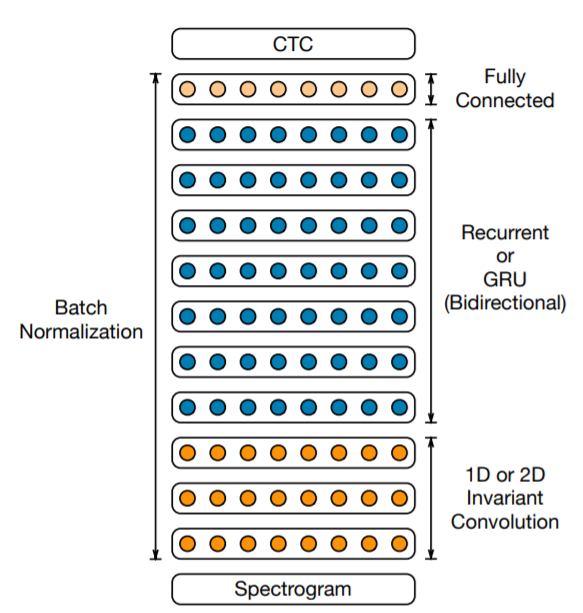
## PyTorch

# Architektúra

A Deep Speech 2 egy speech-to-text működést megvalósító neurális háló, ezt a hálót fogjuk transfer learning-re használni – mivel a mi célunk is egy ilyen működést megvalósítani –, azaz egy hatalmas adatbázissal be lett előre tanítva, mi pedig finom hangoljuk a saját kisebb adatbázisunkkal.

## Felépítése

Ez a háló maximum 11 rétegből állhat beállításoktól függően, lásd **2.1 ábra**, a mi beállításunkban a háló bemenetén van kettő konvolúciós réteg, öt Recurrent Neural Network (RNN) réteg, amelyek lehetnek egyszerű RNN rétegek, de Gated Recurrent Unit (GRU) és Long Short-Term Memory (LSTM) is. A végén az osztályozáshoz található egy teljesen összecsatolt (Fully-Connected) réteg. A háló bemenete egy audio fájl, mi esetünkben WAV, amelyre alkalmazhatunk 1D vagy 2D konvolúciót, első esetben az időtartomány a dimenziónk, második esetben ezt kiegészítjük a frekvenciatartománnyal is, FFT segítségével. Mi a 2D konvolúciót alkalmazzuk. Egy speech-to-text rendszernél figyelni kell a hosszú távú függőségekre, hiszen hangfelismerésnél érdemes „megjegyezni”, hogy mi volt az előző hang részlet, amiből képes lesz összerakni a teljes hangot, abból pedig kitalálni, hogy ez melyik grafémát jelenti. Ehhez a hálónak egy memória elemre van szüksége, ezt oldják meg RNN rétegek, azon belül is az LSTM és a GRU legjobb speciálisan erre a feladatra. Ezekről részletesebben lesz később szó. A hálónk kimenetei grafémák, amely egy fonémikus nyelvnél, mint például az általunk használt angolnál ez az abc betűit jelenti. A háló egy osztályozó, 29 osztályunk van, vagyis az utolsó rétegünk 29 neuront tartalmaz. Ez úgy adódik, hogy az angol abc 26 betűt tartalmaz, ezeken felül használjuk még az aposztrófot (’), szóközt és az alulvonást (\_). A veszteségfüggvény, amit gyakran alkalmaznak hangfelismerésnél a Connectionist Temporal Classification (CTC), mi is ezt használjuk, mivel speciálisan olyan területre lett kifejlesztve, amihez szükség van a hosszú távú függőségek kezelésére. Később részletezzük a működését.



.. ábra

### Environments

### Multi-head attention

### A2C

A hálónk kimenete 29 neuron, viszont minden idő-lépésben osztályoz, azért, hogy ne vesszen el az elő időpont béli becslés, a CTC egy mátrixot kreál.

### RAdam

## Tesztelés, eredmények

|  |  |  |  |
| --- | --- | --- | --- |
| RNN típus/Dekóder | | WER | CER |
| LSTM | Greedy | 8.021 | 3.314 |
| Beam Search | 8.150 | 3.353 |
| GRU | Greedy | 8.668 | 4.444 |
| Beam Search | 8.279 | 4.211 |

2.1 táblázat: AN4 teszthalmazának eredményei, WER/CER eredményei százalékban értendők

Ezután következett a saját adatainkkal való tesztelés. Egy darab címkét csináltam, valamint ehhez hat WAV fájlt. Célok a jövőben

A feladat még korént sincs kész.

Irodalomjegyzék