 

**Balso atpažinimas naudojantis dirbtiniais neuroniniais tinklais**

**(Speaker Recognition using Artificial Neural Networks)**

Nikodemas Žaliauskas

Vilniaus universitetas

Matematikos ir informatikos fakultetas

Gegužės 16 d., 2016

Darbo vadovas: Andrius Grevys

Turinys

[Įvadas 3](#_Toc470712496)

[Neuroninis tinklas 3](#_Toc470712497)

[Kalbėtojų duomenų bazė 4](#_Toc470712498)

[Kalbėtojų reprezentaciniai duomenys 4](#_Toc470712499)

[Parametrai 5](#_Toc470712500)

[Tinklo tikrinimas 5](#_Toc470712501)

[Analizė 5](#_Toc470712502)

[Išvada 5](#_Toc470712503)

[Ateities darbai 5](#_Toc470712504)

[Šaltiniai 5](#_Toc470712505)

# Įvadas

Kalbėtojo atpažinimas naudojatis dirbtiniais neuroniniais tinklais yra smarkiai populiarėjanti technologija, kuri dar nėra labai plačiai naudojama visuomenėje, tačiau programuotojai ir mokslininkai siekia, kad būtų galima tai naudoti kasdieniniame gyvenime. Būtų nuostabu sedėtint namų svetainėje, naudojant vien savo balsą, valdyti namų įrenginius.

Šio darbo tikslas yra atlikti praktinį tyrimą, naudojant jau sukurtą neuroninį tinklą ir naudojant kalbėtojų duomenų bazę, išmokyti tinklą atpažinti kalbėtojų balsus ir išmatuoti, kiek klaidų padaro sistema, kiek atpažįsta kalbėtojų teisingai.

Darbo metu bus aptartas neuroninis tinklas, kalbėtojų duomenų bazė, kokie reprezentaciniai duomenys buvo naudoti ir bus aprašyta detali praktinio darbo analizė.

# Neuroninis tinklas

Prisimenant ankstesnį darbą, trumpai primenu, kad neuronas tai yra matematinė funkcija, kuri sugalvota smegenų neurono modelio pagrindu, kur dirbtinis neuronas turi įvestį, aktyvacijos funkciją ir išvestį. Atitinkamai neuroninis tinklas yra sudarytas iš daugybės tarpusavyje sujungtų dirbtinių neuronų.

Šiam praktiniam darbui pasirinkau „Tensorflow“ atviro kodo programą, kuri yra skirta mašinos mokymosi įvairioms percepsinėms užduotims atlikti. Pasirinkau tokią programą, nes ji yra šiuo metu viena populiariausių ir plačiai naudojamų. Programa yra išleista 2015 m. „Google“ darbuotojų. Taip pat „Tensorflow“ dokumentacija yra pakankamai aiški ir plati, tad galima nesunkiai susirasti visus norimus atsakymus ir problemų sprendimus.

Pasirinkau pora neuroninio tinklo modelių palyginimui. RNN ir LSTM. RNN (angl. Recurrent Neural Networks) – Rekurentiniai neuroniniai tinklai, tai tinklai, kur yra sudaromas orentuotas grafas iš neuronų junginių. LSTM (angl. Long Short-Term Memory) – Ilga trumpalaikė atmintis, tai yra rekurentinių neuroninių tinklų architektūra. Šie du modeliai turi vieną pagrindinį skirtumą. Rekurentiniai neuroniniai tinklai turi bėdą prisiminti praeities įvestis po daugiau atliktų žingsnių skaičiaus. Todėl buvo sukurta LSTM architektūra, kuri gali prisiminti praeities įvestis iš labai senai. Grubiai tariant, jei RNN pamiršta įvestis po daugiau nei 10 žingsnių, tai LSTM prisimena praeities įvestis ir po 1000 žingsnių. Tokie skaičiai verčia manyti, kad išviso neverta naudoti gryno RNN tinklo, bet kaip yra ištikro, pamatysime vėliau.

Taigi, dabar trumpai parodysiu dalį kodo, kaip atrodo tinklo sukūrimas. Tačiau tai nėra pilnas modelio kodas. Čia tik mano manymų vienį svarbiausių aspektų.

self.graph = tf.Graph()

with self.graph.as\_default():

forwardH1 = rnn\_cell.BasicRNNCell(args.num\_hidden)

backwardH1 = rnn\_cell.BasicRNNCell(args.num\_hidden)

fbH1, \_, \_ = rnn(forwardH1, backwardH1, self.inputList, dtype=tf.float32, scope='BDRNN\_H1')

Svarbiausias dalykas, ką būtina padaryti, tai sukurti „Graph“ objektą, kuris saugo tinklą sudarytą iš mazgų, kur kiekvienas mazgas reprezentuoja po vieną operaciją ir sujungti tarpusavį įvestim ir išvestim. Tuomet sukuriame priekinius ir atgalinius perdavimų objektus ir tuomet šiuos objektus perduodame į „rnn“ funkciją, kuri ir sukuria būtent rekurentini neuroninį tinklą.

Norint sukurti ilgos trumpalaikės atminties modelį, šiame kodo fragmente reikėtų pakeisti *BasicRNNCell* į *BasicLSTMCell.*

# Kalbėtojų duomenų bazė

Norint apmokyti neuroninį tinklą yra reikalinga nemaža duomenų bazė. Dėl laiko ir resursų stokos, pasirinkau mažesnę duomenų bazę. Pats „Tensorflow“ turi kalbėtojų duomenų bazę skirtą kalbos atpažinimui. Duomenų bazės dydis yra apie 21GB.

|  |  |  |
| --- | --- | --- |
|  | Apmokymas | Testavimas |
| Iš viso | 774 | 149 |
| Vyrai | 437 | 78 |
| Moterys | 337 | 71 |

Šioje duomenų bazėje yra iš viso 923 failai, kurie yra „.sph“ formato. Šis formatas yra sukurtas naudojantis „NIST SPHERE“ programa. Kadangi „.sph“ formato failai yra dažniausiai skirti kalbos, o ne kalbėtojo atpažinimui, visą duomenų bazę konvertavau į „.wav“ formatą, kuris yra patogesnis formatas norint išgauti kalbėtojo reprezentaciją. Kiekvienas „.wav“ failas yra garso įrašas, kur kiekvienas įrašas yra skirtingos trukmės. Įrašų trukmės svyruoja tarp 10 – 30 minučių.

# Kalbėtojų reprezentaciniai duomenys

Neuroniniam tinklui netinka gryni garso įrašai, todėl juos reikia konvertuoti į kažkokias kalbėtojo reprezentacijas.

Kiekvieną garso įrašą konvertuoju į kelių kadrų funkcinius vektorius, kur funkcinis vektorius yra n – dimensinis vektorius su skaitinėmis reikšmėmis, kurios atvaizduoja tam tikrą objektą. Šiuo atveju objektas yra kalbėtojas. Taigi, norint gauti funkcinius vektorius, pirmiausia kiekvieną garso įrašą skaidome po 20 ms naudodami Hamingo lango funkciją be persidengimo. Tuomet, paskaičiuojame 12 – ą Melo dažnių cepstrinių koeficientų, kurie yra papildyti energijos kintamaisiais kiekvienam kadrui. Turėdami 13 ilgio vektorių, mes paskaičiuojame delta ir delta-delta koeficientus, kur iš viso yra po 39 koeficientus kiekvienam kadrui. Taigi, kiekvienas garso įrašas yra padalintas į kelis kadrus naudojant Hamingo lango funckiją ir kiekvienas kadras yra paverstas į funkcijos vektorių, kurio ilgis yra 39. Ir finale gauname matricą, kurios dydis yra garsoĮrašoIlgis \* 39.

# Parametrai

Čia yra aprašyti vieni pagrindinių tinklo parametrų, kurie skirti tinklo apmokymui ir jų paaiškinimai.

1. Mokymosi lygis (angl. Learning rate) – tai parametras, kuris nurodo, kaip greitai tinklas mokysis. Kuo didesnis mokymosi lygis, tuo greičiau tinklas išmoks, bet bus daugiau klaidų atpažinime. Tad savo tinkle nustačiau 0,001 mokymosi lygį. Mokosi ilgai, bet po to atpažįsta geriau.
2. Epocha (angl. Epoch) – tai parametras, kuris nurodo, kiek bus kartu padaryta priekinių ir atgalinių perdavimų su visais duomenimis. Viena epocha turi iteracijų.

|  |  |
| --- | --- |
|  | 1 |
|  | 2 |

– iteracijų skaičius per vieną epochą, – duomenų kiekis, – siuntos dydis, – sąlyginė funkcija.

1. Siuntos dydis (angl. Batch size) – tai parametras, kuris nurodo, kiek bus paduodama kalbėtojų reprezentacijų per vieną epochos iteraciją. Per sekančią epochos iteraciją, bus paimtas sekantis duomenų kiekis, kol bus panaudoti visi duomenys.
2. Ypatybės skaičius – tai parametras, kuris nurodo, kokia bus ypatybės dimensija.
3. Paslėptų neuronų skaičius, tai parametras, kuris parodo kiek bus neuronų paslėptame sluoksnyje.

# Tinklo tikrinimas

Tinklo tikrinimas, tai dalis, kur aš testuoju apmokytą modelį. Tikrinimui aš pasiemiau iš duomenų bazės duomenis, kurie skirti testavimui. Su šiais duomenimis aš praiteruoju per visą modelį ir tikrinu kiek buvo teisingų ar klaidingų atpažinimų.

# Analizė

# Išvada

# Ateities darbai

# Šaltiniai