VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
PROGRAMŲ SISTEMŲ KATEDRA

Rodiklių duomenų kaupimas, transformavimas ir analizė, naudojant NoSQL duomenų bazę

(Storage, transformation and analysis of indicator data with the help of NoSQL database)

Kursinis darbas

Atliko: 3 kurso studentas
Vytautas Žilinas (parašas)

Darbo vadovas: lekt. Andrius Adamonis (parašas)

Vilnius – 2018

TURINYS

Įvadas

Darbo tikslas: Eksperimento būdu išbandyti rodiklių duomenų kaupimo, transformavimo ir analizės sprendimus, palyginant sprendimą, naudojanti reliacinę duomenų bazę, su sprendimu naudojančiu srautinį duomenų apdorojimą.

Užduotys:

1. Apsibrėžti įmanomus rodiklių duomenų analizės būdus.
2. Sukurti testiniu duomenų generatorių.
3. Išmatuoti reliacinės duomenų bazės sprendimo pralaidumą.
4. Išbandyti skirtingas srautinio duomenų apdorojimo architektūras.
5. Išmatuoti pasirinktos srautinio duomenų apdorojimo architektūros sprendimo pralaidumą.
6. Palyginti sprendimus iš kitų pusiu (setup, interchangabilty, consumption, learning curve)

1. Rodiklių duomenys

1.1. Apibrėžimas

Rodiklių duomenys - tai didelių duomenų tipas, kurį galima transformuoti ir analizuoti ir kuris yra sugrupuotas pagal rodiklius, pavyzdžiui: bazinė menėsio alga, mirusiųjų skaičius pagal mirties priežastis, krituliai per metus. Šie duomenys dažniausiai yra saugomi reliacinėse duomenų bazėse, kur užklausus vartotojui skaičiuojami apibendrinti rodikliai - sumos, vidurkiai ir kita statistika. Lietuvoje pagrindinis rodiklių duomenų bazės pavyzdys yra „Lietuvos statistikos departamento" duomenų bazė, kurios duomenis galima pasiekti `https://osp.stat.gov.lt/statistiniu-rodikliu-analize#/` puslapyje, kuris leidžia ieškoti duomenis pagal vieną arba kelis rodiklius. Didesnis pavyzdys yra „DataBank" `http://databank.worldbank.org` - pasaulinio lygio rodiklių duomenų bazių rinkinys, turintis 69 skirtingas duomenų bazes, pavyzdžiui - „World development indicators", „Gander statistics" ir kitus[Ban18].

1.2. Charakteristikos

Apibrėžime minėjau, kad rodiklių duomenis yra didelių duomenų tipas, todėl galime jiems pritaikyti didelių duomenų charakteristikas ir apsibrėžti, kurios iš jų mums sudaro daugiausiai problemų. Šie iššūkiai apibrėžiami Gartner's Doug Laney pristatytu 3V modeliu[Lan01], kuris veliau buvo papildytas Bernard Marr iki 5V modelio[Mar14]:

- Tūris (angl. Volume). Apibrežia generuojamų duomenų kiekius. Didelių duomenų atveju yra šnekama apie duomenų kiekius, kuriuos yra sudetinga arba neįmanoma saugoti ir analizuoti tradicinėmis duomenų bazių technologijomis. Rodiklių duomenų kiekiai dažniausiai nesudaro problemos saugojant, tačiau didelė problema yra rodiklių duomenų analizė, kadangi tuos pačius duomenis reikia apdoroti pagal neapribotą skaičių skirtingų rodiklių.
- Greitis (angl. Velocity). Apibrežia greitį, kuriuo nauji duomenis yra generuojami. Rodiklių duomenų atveju, tai yra labai svarbu, kadnagi nauji duomenis, kurie gali tikti skirtingiems rodikliams yra generuojami visais laikais.
- Įvairovė (angl. Variety). Apibrežia duomenų tipus. Duomenys gali būti: strukturizuoti, nestrukturizuoti arba dalinai strukturizuoti[ZE+11]. Rodiklių duomenis dažinau yra strukturizuoti, todėl tai nėra aktualus iššūkis.
- Tikrumas (angl. Veracity). Apibrežia duomenų tesingumą ir kokybę. Pavyzdžiui, jeigu analizuotume „Twitter" socialinio tinklo žinučių turinį gautume daug gramatikos klaidų, naujadarų, slengo. Statistinio departamento atveju duomenys visada bus tvarkingi, kadangi tai dažniausiai yra duomenys surinkti iš dokumentų ir apklausų, o ne laisvo įvedimo.
- Vertė (angl. Value). Apibrežia duomenų ekonominę vertę. Rodiklių duomenys yra labai vertingi įstaigoms, nes dažniausiai tos įstaigos užsiema tik rodiklių duomenų kaupimų ir analizė, iš techninės pusės ši charakteristika yra svarbi iš tos pusės, kad duomenų apdorojimo ir kaupimo sprendimai labai stipriai daro įtaką įstaigos, kaupiančios rodiklių duomenis, ekonomikai. Taip pat šių duomenys ir jų analizė turi būti pasiekiama be prastovos laiko.

Pagal apibrėžtas charakteristikas matome, kad pagrindiniai rodiklių duomenų iššūkiai yra tūris, greitis ir vertė. Todėl mūsų bandomas sprendimas turi galėti greitai susidoriti su dideliu kiekių labai skirtingų duomenų ir taip pat turi būti įmanoma šį sprendimą paleisti į realią aplinką nepertraukiant įstaigos veiklą.

2. Rodyklių duomenų talpinimo paradigmos

Jungtys tarp neuronų yra pateiktos skaitine išraiška ir vadinamos svoriu. Kuo didesnis šis svoris tuo didesnę įtaką turi vienas neuronas kitam. Vienam neuronui yra pateikiama visų prieš jį buvusių neuronų informacija ir jungčių svoriai. Kiekvieno neurono informacija yra sudauginama su jo svoriu ir visi šie duomenys yra sudedami tarpusavyje. Taip iš vektoriaus gaunamas vienas rezultatas ir jei šis rezultatas tinka aktyvavimo funkcijai, jis yra perduodamas tolimesniems neuronams. Tokio tipo veikimo dizainas yra vadinamas „feedforward" tinklu.

Tačiau jungčių svoriai nėra pastovūs. Kai dirbtinis neuroninis tinklas mokosi, galutinis rezultatas yra lyginamas su tikėtinu teisingu rezultatu, jei šie rezultatai skiriasi, svoriai yra keičiami atitinkamai, tai vadinama „backpropagation". Tokiu būdu yra gerinamas rezultatas ir mažinamas skirtumas tarp tikėtino ir gauto atsakymų.

2.1. Reliacinis duomenu talpinimas

Aktyvavimo funkcijų yra įvairių, kadangi sprendžiant tam tikrą problemą yra geriau naudoti vienas funkcijas, o kitas problemas - kitas funkcijas. Pagrinde yra dviejų tipų aktyvavimo funkcijos - tiesinės ir netiesinės. Tiesinės nėra tokios populiarios, kadangi jos neleidžia įvesčiai būti lanksčiai. Nors tiesinė funkcija labai dažnai naudojama išeities sluoksnyje. Netiesinės funkcijos dažniausiai naudojamos vidiniuose sluoksniuose. Šiuo metu labiausiai naudojama yra ReLU, kadangi naudojant šią funkciją mokymo rezultatai nuolatos gerėja, tačiau ReLU funkcijos spraustumas nesuteikia efektyvumo tinklui [XWC+15].

2.2. Srautinis duomenu apdorojimas įdėjimo metu

Kai dirbtinis neuroninis tinklas mokosi, jo gaunami rezultatai gali labai skirtis nuo tikėtinų rezultatų. Todėl nuostolio funkcija apskaičiuoja kaip stipriai skiriasi gautas rezultatas nuo tikėtino. Kuo didesnis nuostolis tuo toliau nuo teisingo atsakymo yra dirbtinis neuroninis tinklas [Dav15]. Paprasčiausia ir dažniausiai naudojama nuostolio funkcija yra vidutinio kvadrato klaida. Ši funkcija apskaičiuoja kvadratinį skirtumą tarp tikėtino ir gauto rezultatų. Tačiau šios funkcijos vienas iš didesnių trūkumų - neproporcingas išskyrimas didelių rezultatų. Kadangi funkcija didėja kvadratiniai, o ne tiesiniai, kai gaunamas rezultatas tolsta nuo tikėtino rezultato.

2.3. Duomenų indeksavimas įdėjimo metų NoSQL duomenų bazės pagalba

Optimizavimo funkcijos naudojamos vidinių tinklo parametrų atnaujinimui, kad sumažinti gaunamų rezultatų netikslumą. Visos optimizavimo funkcijos gali būti suskirtytos į du tipus - nuolatinio mokymosi greičio ir prisitaikančio mokymosi.

Nuolatinio mokymosi greičio funkcijos turi hiperparametrą - mokymosi greitį. Jis privalo būti nustatytas, tačiau pasirinkti tinkamą mokymosi greitį gali būti sudėtinga - pasirinkus per mažą vidiniai parametrai gali labai lėtai konverguoti, o pasirinkus per didelį parametrams gali trukdyti konverguoti ir priversti nuostolio funkciją svyruoti apie minimumą arba diverguoti. Šio tipo

funkcijos turi panašų hiperparametrą - momentą - kuris didina mokymosi greitį, kai jis artėja prie minimumo.

Vienos iš pagrindinių problemų nuolationio mokymosi greičio funkicijų, kad jos privalo turėti nustatytus hiperparametrus iš anksto ir jie labai stipriai priklauso nuo modelio ir sprendžiamos problemos. Dar vienas trūkumas, kad toks pats mokymosi greitis yra pritaikomas visiems vidinių parametrų atnaujinimams.

Prisitaikančio mokymosi funkcijos turi atskirus kiekvieno parametro mokymosi greičio metodus, kurie teikia euristikos metodą, nereikalaujant brangaus darbo rankiniu būdu nustatant hiperparametrus mokymosi greičiui. Tačiau šios funkcijos generalizuoja blogiau negu nuolatinio mokymosi greičio funkcijos, nors ir mokymosi metu pasirodo geriau [WRS+17].

3. Konvoliucinis neuroninis tinklas

Konvoliuciniai neuroniniai tinklai yra labai panašūs į paprastus dirbtinius neuroninius tinklus (daugiau informacijos skyriuje „Dirbtinis neuroninis tinklas"). Tačiau pagrindinis skirtumas tarp šių tinklų yra, kad konvoliucinio įeities sluoksnis priima tik tai paveiksliukus, kurie jei padaryti su standartine skaitmenine kamera, turi tris komponentus - raudoną, žalią ir mėlyną. Šiuos komponentus galima įsivaizduoti kaip tris 2D matricas sudėtas viena ant kitos. Kiekvienos matricos i-osios eilutės ir j-ojo stulpelio elementas atitinka nuotraukos pikselį, kurio reikšmė yra intervale nuo 0 iki 255. Kadangi naudojamos informacijos tipas yra specifinis, tai labai sumažina tinklo parametrų kiekį ir tinklą padaro efektyvesnį.

Objektų atpažinimas paveiksliukuose yra sudėtingas dėl šių iššukių:

- Segmentavimas - paveiksliukai gali atvaizduoti įvairias scenas, kuriose gali būti pavaizduota daug objektų, kurie vienas kita gali dalinai uždengti.
- Šviesa - pikselių intensyvumas gali būti paveiktas šviesos šaltinio ar pačio objekto.
- Deformacija - objektai gali būti deformuoti įvairiais būdais, pavyzdžiui, kiekvieno žmogaus ranka parašyti skaičiai skiriasi.
- Galimybės - objektų klasės dažnai nustatomos pagal tai kaip patys objektai yra naudojami, pavyzdžiui, kėdės yra objektai sukurti sėdėti, tačiau jos gali turėti įvairų dizainą.
- Žvilgsnio taškas - keičiant vietą iš kurios yra žiūrima gali keistis objekto forma, informacija šokinėja per įeities sluoksnio dimensiją (t.y. pikselius).

3.1. Konvoliucija

Konvoliucija yra matematinė operacija, kuri apibūdina taisyklę, kuri parodo kaip reikia sujungti du informacijos rinkinius [PG17]. Paveiksliukų analizėje, statinė ir pagrindinė funckija yra įeities paveiksliukas, kuris yra analizuojamas, o antroji, judanti funkcija, žinoma kaip filtras, nes ji išskiria paveiksliuko ypatybę. Abi funkcijos yra susietos daugyba (2 pav.).

Tačiau konvoliuciniai tinklai turi daug filtrų, kurie pereina per vieną paveiksliuką, kiekvienas išskirdamas skirtingą paveiksliuko ypatybę. Pirmuose sluoksniuose šiuos filtrus galima įsivaizduoti kaip horizontalių, vertikalių ar įstrižų linijų filtrus, kurie sukuria paveikslėlio kraštų planą. Tinklas paima visus filtrus, gabaliukus paveiksliukų ypatybių vietų, ir juos sudeda į planą, kuris parodo ypatybės vietą. Mokydamasis skirtingų proporcijų ypatybių, tinklas leidžia lengvai kurti greitą ypatybių atpažinimą.

3.2. Konvoliucinio neuroninio tinklo sluoksniai

Konvoliuciniai neuroniniai tinklai tai yra sluoksnių rinkinys, kuris turi įeities, vidinius ir išeities sluoksnius. Tačiau priklausomai kokio tipo konvoliucinis neuroninis tinklas vidiniai sluoksniai gali skirtis. Konvoliuciniai neuroniniai tinklai turi tris pagrindinius sluoksnių tipus, kurie sudaro vidinį sluoksnį. Šie tipai yra konvoliucinis, sujungimo ir pilno sujungimo sluoksniai.

3.2.1. Konvoliucinis sluoksnis

Šis sluoksnis yra pagrindinis konvoliucinio neuroninio tinklo sluoksnis, kuris atlieka daugiausia skaičiavimų, nustato visas paveiksliuko ypatybes. Kadangi, įeities informacija (paveiksliukas) yra didelės dimencijos neefektyvu visų neuronų sujungti vienus su kitais. Todėl neuronai yra sujungiami su lokaliu informacijos kiekiu, kuris yra lygus filtro dydžiui ir vadinamas erdviniu mastu [Li15].

Neuronų kiekis po konvoliucijos (ypatybių plano dydis) yra nustatomas trimis parametrais:

- Gylis - atitinka filtrų skaičių.
- Žingsnis - pikselių kiekis, kuris parodo per kiek reikia slinkti filtro matrica per įeities informacijos matricą.
- Nulių pamušalas - įeities informacijos matrcios kraštus užpilduti nuliais.

3.2.2. Sujungimo sluoksnis

Periodiškai sujungimo sluoksnis yra įterpiamas tarp konvoliucinių. Pagrindinis sluoksnio tikslas yra laipsniškai mažinti erdvinį filtruojamo paveiskliuko mąstą. Šis veikslas yra atliekams tam, kad sumažinti parametrų ir skaičiavimų kiekį bei kontroliuoti perjungimą. Sujungimo sluoksnis veikia nepriklausomai nuo kiekvieno gabalėlio gylio ir keičia jo dydį erdviškai, naudodamas MAX operaciją. Dažniausiai šis sluoksnis yra naudojamas su 2x2 dydžio filtru - kas antras po konvoliucijos gauto gabaliuko kiekvienas gylio sluoksnis yra mažinamas per pusę ties ilgiu ir pločiu, taip yra atsikratoma 75 procentų aktyvacijų. Po šios operacijos gabaliuko gylis nepasikeičia.

Dažniausiai yra naudojamos dvi šio sluoksnio variacijos. Pirmasis yra vadinamas persidengiantis sujungimas, kur filtro dydis yra lygus 3 ir žingsnis yra lygus 2. O kitas dažniau naudojamas turi filtro dydį lygų 2 ir žingsnį taip pat 2. Sujungimo sluoksniai su labai dideliais parametrais yra labai desktruktyvūs.

3.2.3. Pilno sujungimo sluoksnis

Konvoliucinio ir sujungimo sluoksnių išeitys yra aukšto lygio ypatybės, kurios yra gautos iš įeities paveiksliuko. Pilno sujungimo sluoksnis yra sujungtas su visais neuronais iš sluoksnio buvusio prieš jį. Šio sluoksnio tikslas yra panaudojant tas ypatybes, kurios yra gautos iš prieš tai buvusių sluoksnių, nustatyti kokioms klasėms priklauso įeities paveiksliukas pagal mokymo informacijos rinkinį, kai nueroninio tinklo problema yra klasifikacija [Kar16]. Jei šiam sluoksniui yra naudojama minkštojo maksimumo funkcija tuomet sudėtis visų gautų galimybių turi būti lygi 1. Minkštojo maksimumo funkcija priima vektorių įvertinimų ir jį suspaudžia į vektorių, kuriame yra klasių tikimybių įvertinimai intervale nuo 0 iki 1, kur tikimybė arčiausiai vieneto reiškia, kad labiausiai užtikrintas dėl tos klasės.

3.3. Architektūros

Konvoliuciniai neuroniniai tinklai turi keletą skirtingų architektūrų, kurios yra naudojamos atinkmai pagal sprendžiamą problemą. 1 lentelėje pateikta infromaciją apie įvairias architektūras.

Pavadinimas	Metai	Parametrų kiekis	Veikimas	ILSVRC vieta
LeNet	1998	60 000	Geriausiai atpažysta ranka paraštytus skaičius. Susideda iš sluoksnių - kelių pasikartojančių konvoliucijos ir sujungimo bei pasibaigia dviem pilno sujungimo sluoksniais.	-
AlexNet	2012	60 000 000	Veikimu panašus į LeNet, tačiau turi daug daugiau parametrų ir filtrų bei sudėtus konvoliucinius sluoksnius.	pirma
GoogLeNet/Inception	2014	4 000 000	Vidiniai sluoksniai sudėti paraleliai, naudojami Inception moduliai. Vienas modulis savyje turi 1x1, 3x3 ir 5x5 dydžių konvoliucijos filtrų bei vidurkio sudėjimo sluoksnius.	pirma
VGGNet	2014	138 000 000	Panašus veikimas į AlexNet, tačiau daug gilesnis. Naudojamų filtrų dydis yra 3x3 ir jie yra sudėti vienas po kito.	antra
ResNet	2015	25 000 000	Turi labai daug sluoksnių, sudėtų vienas po kito, kurie turi liekamąjį bloką, kuris įeities informaciją perduoda tolimesniam sluoksniui ją pridėdamas ir taip sumažina konvoliucijos ir aktyvavimo funkcijų kiekį.	pirma

1 lentelė. Konvoliucinių neuroninių tinklų architektūros

3.4. Modelio reguliavimas

Pilnas konvoliucinio neuroninio tinklo apmokymas gali užtrukti labai ilgą laiką ir išnaudoti daug resursų. Todėl yra kai kurios įstaigos arba žmonės, kurie apmoko savo tinklą ir jo svorius bei reikšmes, vadinamą modeliu, pateikia visuomenei, tačiau šis modelis yra nepritaikytas individuoliai žmogaus užduočiai. Modelį reikia reguliuoti - iš naujo apmokyti paskutinius sluoksnius su

individuolios užduoties parametrais.

Daugelis konvoliucinių neuroninių tinklų apmokytų su natūraliais paveiksliukais turi fenomeną. Pirmuosiuose sluoksniuose jie išmoksta ypatybių panašių į Gaboro filtą (tiesinis filtras naudojamas tekstūroms analizuoti) ir spalvų dėmes. Šios pirmojo sluoksnio ypatybės nepriklauso nuo duomenų rinkinio, bet yra bendros ir tinkamos daugeliui duomenų rinkinių ir užduočių [YCB+14]. Dėl šio fenomeno galima naudoti modelius neapmokytus su specifiniu duomenų rinkiniu, bet jį minimaliai modifiktą, kitoms užduotims spręsti, kas leidžia sutaupyti resursų bei turėti mažesnį duomenų rinkinį.

4. Technologijos

Naudojamų technologijų išsirinkimas yra pradinis žingsnis siekiant įvykdyti išsikeltas užduotis. Šiame skyriuje pateiktos populiariausios šių laikų technologijos bei trumpai papasakota apie jas.

4.1. ImageNet

ImageNet yra projektas sugalvotas profesorės Li Fei-Fei 2009 metais. Projekto tikslas buvo sukurti didelę sukategorizuotų paveiksliukų ir jų etikečių duomenų bazę, kuri butų skirta vizualinio objekto atpažinimo programinės įrangos tyrimams. Ši duomenų bazė yra suorganizuota pagal WorldNet hierarchija - anglų kalbos žodžiai yra grupuojami į sinonimų rinkinius, kurie turi apibūdinimus ir naudojimo pavyzdžius bei saugo ryšių kiekį tarp sinonimų arba jų narių. ImageNet turi daugiau nei 100 000 sinonimų rinkinių, kur didžioji dalis yra daiktavardžiai (80 000+).

Taip pat šis projektas kiekvienais metais daro konkursą vadinamą „ImageNet Large Scale Visual Recognition Challenge" (trumpinys ILSVRC). Konkurso užduotis yra išmokinti modelį, kuris galėtų įeities paveiksliuką teisingai klasifikuoti į 1000 skirtingų objektų klasių, kurios atitinka realius daiktus, gyvūnus ir t.t. Modeliai yra apmokomi su apie 1.2 milijonų paveiksliukų ir dar 50 000 paveiksliukų yra naudojami validacijai mokymo metu bei 100 000 paveiksliukų yra panaudojami galutiniam modelio testavimui. Šis konkursas yra paveiksliukų klasifikacijos algoritmų etalonas.

4.2. Keras

Keras yra aukšto lygio programų sąsaja skirta neuroniniams tinklams. Sąsaja parašyta su „Python" programavimo kalba ir vidinėje pusėje galinti veikti su „TensorFlow" ir kitomis bibliotekomis. Keras buvo sukurtas tikintis suteikti greitą ekspermentavimą, kad sugalvojus idėją pasiekti rezultato būtų galima su kiek įmanoma mažiau uždėlsimo.

Ši sąsaja savyje turi visus pagrindinius neuroninio tinklo kūrimo blokus, pavyzdžiui, sluoskniai, aktyvavimo ir optimizavimo funkcijos. Taip pat Keras suteikia modelius, kurie yra apmokyti naudojant ImageNet duomenų bazę. Šiuos modelius galima reguliuoti, pridėti papildomų sluoksnių, pasirinkti esamus sluoksnius bei juos iš naujo apmokyti.

4.3. TensorFlow

TensorFlow yra atviros programinės įrangos biblioteka skirta aukšto našumo skaitinimas skaičiavimams. Jo lanksti architektūra leidžia lengvai diegti skaičiavimus įvairiose platformose - procesoriuose, grafikos procesoriuose. Sukurtas „Google" dirbtinio intelekto skyriaus, tad yra labai palaikomas automatinis ir gilusis mokymasis, tačiau dėl bibliotekos ir skaičiavimų lankstumo yra naudojamas įvairiose mokslinėse srityse.

5. Ekspermentas

Šio ekspermento tikslas yra išanalizuoti mokymosi tikslumą su skirtingų gylių neuroninais tinklais, kai mokymui yra naudojamas mažas paveiksėlių rinkinys. Taigi, pirmas šio ekspermento žingsinis - išsirinkti konvoliucinį neuroninį tinklą. Poskyriuje „Architektūros" yra trumpai apibūdinti pagrindiniai konvolicinių neuroninių tinklų tipai. Iš jų buvo išsirinktas VGG, kadangi jo veikimas ir sluoksnių išsidėstymas yra tinkamiausi išsikeltam tikslui įgyvendiniti. Buvo nuspręsta daryti paprastą, binarinę paveikslėlių klasifikaciją - nustatymas ar katė, ar šuo pavaizduotas paveikslėlyje.

5.1. Duomenų rinkinys

Šiais laikais konvoliuciniai neuroniniai tinklai pranoksta prieš tai buvusias naujausias technologijas naudotas paveiksliukų klasifikacijai. Viena iš pagrindinių priežasčių yra dideli ir gerai aprašyti duomenų rinkiniai, kaip kad ImageNet duomenų bazė. Geriausiai tinklas pasirodo, kai yra apmokomas su bendrinėm ypatybėm ir dar efektyviau pasirodo kai yra sureguliuoti su specifiniu duomenų rinkiniu [**http://adas.cvc.uab.es/task-cv2016/papers/0002.pdf**]. Optimalus duomenų kiekis, kad gerai sureguliuoti modelį yra apie 10 000 paveiksliukų.

Tačiau realybėje visų klasių objektų paveiksliukų nėra be galo daug, kadangi reikia skirti labai daug laiko ir žmogiškųjų resursų paveiksliukų žymėjimui bei turėti tiek daug paveiksliukų. Todėl tenka tinklus apmokyti su limituotais duomenų rinkiniais. Pagal šią realaus pasaulio problemą buvo įvykdyta viena iš ekspermento tikslo sąlygų - mažas duomenų rinkinys. Buvo surastas ir naudotas duomenų rinkinys, kuris turi 10 000 paveikslėlių - 8 000 mokymosi tikslui ir 2 000 validacijos.

5.2. Programos veikimas

Skyrije „Technologijos" yra išvardintos visos technologijos, kurios buvo naudotos šio ekspermento programai parašyti.

Pirmiausiai reikia paruošti kompiuterį darbui - įrašyti „Python" programavimo įrankius, paruošti „Anaconda" komandinę eilutę, „NVIDIA CUDA" įrankius, Keras ir TensorFlow. Tačiau kompiuteris privalo turėti tinkamą procesorių ar grafinį procesroiaus bloką - kompiuteris, kurį naudojau turėjo XXXX. Po šių pasiruošiamųjų žingsnių galima pradėti rašyti programą.

- CNN issirinkimas - Duomenu radimas CNN reguliavimo programos parasymas bei programos pritaikymas modifikavimui Skirtingu optimizavimo funkciju pritaikymas ir rezultatai

dataset: 8000 train 2000 validate

epoch - 40 trainbatch - 200 valbatch - 50

ft: layers = 23 trainable = 8 total param - 40 406 849 trainable - 32 771 585 netrainable - 7 635 264

add lay: layer = 34 trainable = 19 total param - 51 952 449 trainable - 44 317 185 netrainable - 7 635 264

Rezultaitai ir išvados

Darbo rezultatai:

- Išnagrinėti rodiklinių duomenų analizės budai pagal jų privalumus ir trūkumus.
- Sukurta reliacinė duomenų bazė su testiniais duomenimis, kurių reliacinis apdorojimo užtruks daugiau nei 10 sekundžių.
- Atlikus duomenų bazių apžvalgą pasirinkta ”ElasticSearch” NoSQL duomenų bazė su greita duomenų indeksavimo architektūra savo kuriamam sprendimui.
- Sukurtas srautinio apdorojimo sprendimo prototipas naudojant ”ElasticSearch” duomenų bazę, ”Kafka” Messaging queue ir su ”.NET Web API” karkasą konkrečios užduoties sprendimui.
- Palygintas savo sukurtas prototipas ir reliacinė ”MSSQL” duomenų bazės sprendimas pagal greitaveiką - lyginant per kiek laiko gaunami apdoroti duomenis.
- Palygintas esamas srautinio apdorojimo sprendimas ”Heron” ir savo sukurtas prototipas pagal modalumą, konfigūravimo sudėtingumą, greitaveiką.
- Visi sprendimai palyginti pralaidumo testavimu.

Darbo išvados:

-
-

Literatūra

[Ban18] The World Bank. List of databank databases. http://databank.worldbank.org/data/databases/, 2018.

[Dav15] Cameron Davidson-Pilon. *Bayesian Methods for Hackers*. Addison-Wesley Professional, 2015.

[YCB+14] Jason Yosinski, Jeff Clune, Yoshua Bengio ir Hod Lipson. How transferable are features in deep neural networks? *CoRR*, abs/1411.1792, 2014. arXiv: 1411.1792. URL: http://arxiv.org/abs/1411.1792.

[Kar16] Ujjwal Karn. An intuitive explanation of convolutional neural networks. https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/, 2016.

[Lan01] Doug Laney. 3d data management: controlling data volume, velocity and variety. *META Group Research Note*, 6(70), 2001.

[Li15] Fei-Fei Li. Convolutional neural networks. https://cs231n.github.io/convolutional-networks/, 2015.

[Mar14] Bernard Marr. Big data: the 5 vs everyone must know. *LinkedIn Pulse*, 6, 2014.

[PG17] Josh Patterson ir Adam Gibson. *Deep Learning*. O'Reilly Media, Inc., 2017.

[WRS+17] A. C. Wilson, R. Roelofs, M. Stern, N. Srebro ir B. Recht. The Marginal Value of Adaptive Gradient Methods in Machine Learning. *ArXiv e-prints*, 2017. eprint: 1705.08292 (stat.ML).

[XWC+15] Bing Xu, Naiyan Wang, Tianqi Chen ir Mu Li. Empirical evaluation of rectified activations in convolutional network. *CoRR*, abs/1505.00853, 2015. arXiv: 1505.00853. URL: http://arxiv.org/abs/1505.00853.

[ZE+11] Paul Zikopoulos, Chris Eaton ir k.t. *Understanding big data: Analytics for enterprise class hadoop and streaming data*. McGraw-Hill Osborne Media, 2011.