SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

SPARK INTEGRÁCIA S ĎALŠÍMI TECHNOLOGIAMI (TENSORFLOW, PYTORCH)

SEMINÁRNA PRÁCA

2022 Filip Poljak Škobla, Patrik Šebeš

Obsah

Zoznam obrázkov a tabuliek

Zoznam skratiek

AI	Artificial intelligence
API	Application Programming Interface
BSD	Berkeley Software Distribution
CPU	Central Processing Unit
GPU	Graphics Processing Unit
ML	Machine Learning
RAM	Random Access Memory
REST	Representational State Transfer
SQL	Structured Query Language
TPU	Tensor Processing Unit

Úvod

Údaje sú neodmysliteľnou súčasťou nášho života. Stretávame sa s nimi každodenne, už od narodenia, aj keď si to niekedy neuvedomujeme. Vďaka tomu sa v dnešnej dobe dostávajú do popredia mnohé odvetvia a pojmy, ktoré v minulosti neexistovali. Jednými z týchto pojmov sú: big data a umelá inteligencia. My si preto, v tejto semestrálnej práci povieme, ako vieme pracovať s tými pojmami a konkrétne sa zameriame na Apache Spark, ktorý sa zaoberá spracovaním dát a ukážeme si ako vieme interagovať túto technológiu s knižnicami TensorFlow a PyTorch, ktoré sa zaoberajú umelou inteligenciou.

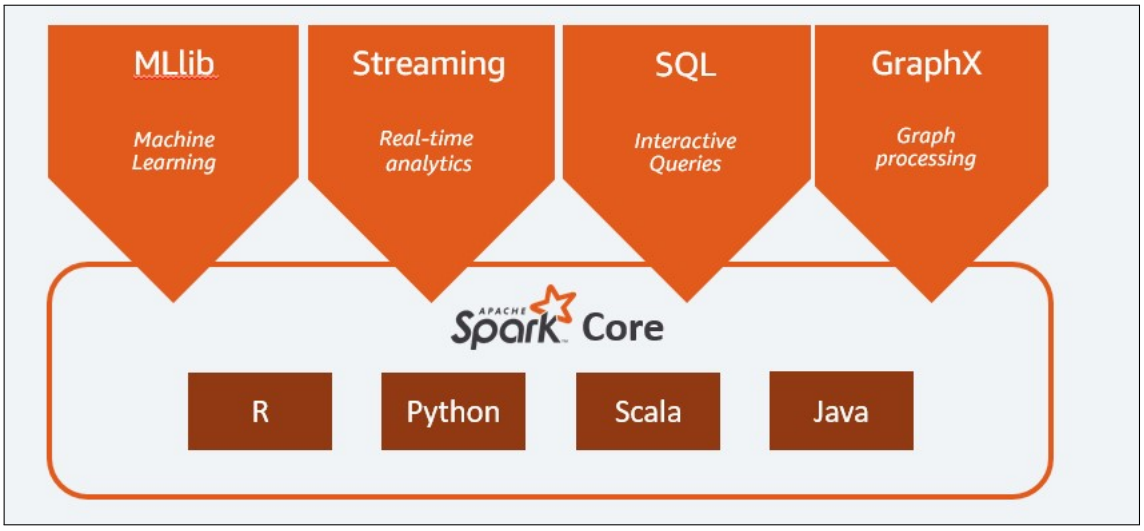
1 Apache Spark

Framework Apache Spark je open-source, distribuovaný systém spracovania používajucí sa na veľké objemy dát napísaný v programovacom jazyku Scala. Využíva pamäť s priamym prístupom a poskytuje optimalizované a rýchle vykonávanie dotazov. Patrí k najpopulárnejším frameworkom na distribuované spracovanie veľkých dát, pretože podporuje opätovné použitie kódu v rámci viacerých pracovných zaťažení a taktiež poskytuje vývojárske API vo veľa programovacích jazykoch ako: Java, Scala, Python a R.

Pôvodným autorom tohto frameworku bol Matei Zaharia, ktorý ho prvýkrát vydal pod licenciou BSD. Ale v roku 2013 bol projekt venovaný Apache Software Foundation a jeho licencia sa zmenila na Apache 2.0 a odvtedy ho táto inštitúcia udržiava.

Apache Spark framework sa skladá z:

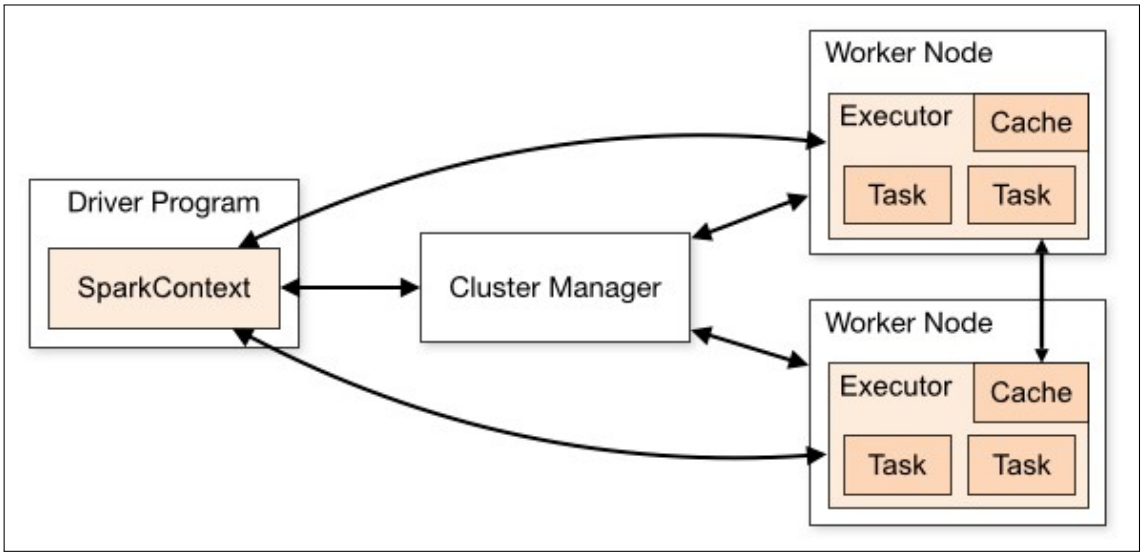
- Spark Core - tvorí základ platformy.
- Spark SQL - pre interaktívne dotazy.
- Spark Streaming - pre analýzu v reálnom čase.
- Spark MLlib - využíva sa na strojové učenie.
- Spark GraphX - využíva sa na spracovanie grafov [1].



Obr. 1: Zloženie Apache Spark frameworku

1.1 Fungovanie Apache Spark

Na základnej úrovni sa aplikácia Apache Spark skladá z dvoch hlavných komponentov: ovládača, ktorý konvertuje kód používateľa na viacero úloh, ktoré možno distribuovať medzi pracovné uzly a spúšťačov, ktorí bežia na týchto uzloch a vykonávajú priradené úlohy. Na sprostredkovanie medzi nimi je potrebná určitá forma manažéra klastra [2].



Obr. 2: Fungovanie Apache Spark frameworku

1.2 Výhody Apache Spark

V tejto sekcií si povieme, aké ma tento framework výhody narozdiel od iných frameworkov, ktoré sa zaoberajú spracovávaním dát.

1. **Rýchlosť**
   Najväčšou výhodou tohto frameworku je rýchlosť, kvôli čomu je u odborníkoch na big data aj veľmi populárny. Oproti konkurencií - Hadoop je až 100x rýchlejší v spracovaní dát vo veľkom meradle, pretože využíva pamäť s priamym prístupom(RAM). Taktiež dokáže spracovávať niekoľko petabajtov klastrových dát na viac ako 8000 uzloch naraz.

2. **Jednoduchosť použitia**
   Obsahuje ľahko použiteľné rozhrania API na prácu s veľkými súbormi dát a ponúka viac ako 80 operátorov na vysokej úrovni, ktorí uľahčujú vytváranie parelelných aplikácií.

3. **Pokročilá analytika**
   Podpora strojového učenia, grafových algoritmov, streaming dát a SQL dotazov.

4. **Viacjazyková podpora**
   Podpora mnoho programovacích jazykov pre písanie kódu: Python, Scala, Java, R.

5. **Open-source komunita**
   Podpora iných vývojárov podielať sa na zlepšení tohto frameworku neustále rastie.

1.3 Nevýhody Apache Spark

V predchádzajúcej sekcií sme si uviedli výhody tohto frameworku, ale taktiež pre vývojárov a ľudí, ktorí sa zaoberajú dátami a chceli by použiť tento framework - je dôležité vedieť aj jeho nevýhody, na ktoré treba myslieť dopredu.

1. **Neposkytuje proces automatickej optimalizácie**
   Pri písaní kódu musí vývojár optimalizovať kód manuálne, pretože tento framework neposkytuje automatickú optimalizáciu kódu.

2. **Systém správy súborov**
   Apache Spark neprichádza s vlastným systémom správy súborov. Závisí to od niektorých iných platforiem, ako je Hadoop alebo iných cloudových platforiem.

3. **Malý počet algoritmov**
   V prípade Apache Spark Machine Learning Spark MLlib je prítomných menej algoritmov. Zaostáva z hľadiska množstva dostupných algoritmov.

4. **Problém s malými súbormi**
   Pri používaní Apache Spark spolu s Hadoop sa pomerne často vyskytujú problémy so spracovaním malých súborov.

5. **Kritériá okien**
   Dáta v Apache Spark sú rozdelené do malých dávok vopred definovaného časového intervalu. Nepodporuje tak kritéria okien založených na záznamoch.

6. **Nevhodnosť pre prostredie pre viacerých používateľov** Apache Spark nedokáže zvládnúť súbežnosť viacerých používateľov [3].

2 TensorFlow

TensorFlow je bezplatná softvérová knižnica s otvoreným zdrojovým kódom pre strojové učenie a umelú inteligenciu. Ide o symbolickú matematickú knižnicu, ktorá využíva tok údajov a diferencovateľné programovanie na vykonávanie rôznych úloh zameraných na trénovanie a odvodzovanie hlbokých neurónových sietí. Umožňuje vývojárom vytvárať aplikácie strojového učenia pomocou rôznych nástrojov, knižníc a zdrojov komunity.

TensorFlow bol vyvinutý tímom Google Brain pre interné použitie Google vo výskume a výrobe v roku 2015. TensorFlow Google využíva na strojové učenie vo všetkých svojich produktoch na zlepšenie vyhľadávača, prekladu, popisovania obrázkov alebo odporúčaní. Táto knižnica bola vytvorená tak, aby fungovala na viacerých CPU alebo GPU a dokonca aj na mobilných operačných systémoch, a má niekoľko obalov v niekoľkých jazykoch, ako je Python, C++ alebo Java.

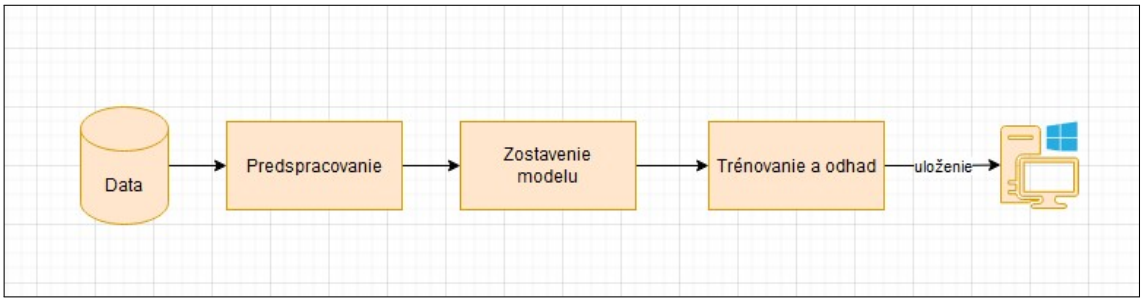
2.1 Fungovanie a architektúra TensorFlow

Umožňuje vytvárať grafy a štruktúry toku údajov, pričom sa definuje, ako sa údaje budú pohybovať v grafe, a to tak, že vstupy sa berú ako viacrozmerné pole nazývané Tensor. Taktiež umožňuje zostaviť vývojový diagram operácií, ktoré je možné vykonať na týchto vstupoch, ktoré idú na jednom konci a prichádzajú na druhý koniec ako výstup.

Architektúra TensorFlow

Je založená z troch hlavných častí:

- Predspracovanie údajov
- Zostavenie modelu
- Trénovanie a odhad modelu



Obr. 3: Architektúra TensorFlow

V TensorFlow všetky výpočty zahŕňajú tenzory. Tenzor je vektor alebo matica n-dimenzií, ktorá predstavuje všetky typy údajov. Všetky hodnoty v tenzore obsahujú rovnaký dátový typ so známym (alebo čiastočne známym) tvarom. Tvar údajov je rozmernosť matice alebo poľa. Tenzor môže pochádzať zo vstupných údajov alebo z výsledku výpočtu.

V TensorFlow sa všetky operácie vykonávajú vo vnútri grafu. Graf je množina výpočtov, ktoré prebiehajú postupne. Každá operácia sa nazýva operačný uzol a sú navzájom prepojené. Graf znázorňuje operácie a spojenia medzi uzlami. Nezobrazuje však hodnoty. Okraj uzlov je tenzor, je to spôsob, ako naplniť operáciu údajmi [4].

2.2 Výhody a nevýhody TensorFlow

V tejto sekcií si rozoberieme, aké má táto knižnica výhody a nevýhody.

Výhody

1. **Škálovateľnosť**
   Tensorflow nie je obmedzený na jedno konkrétne zariadenie. Na mobilnom zariadení funguje rovnako efektívne ako na akomkoľvek inom komplexnom stroji. Knižnica je definovaná tak, že jej nasadenie nie je obmedzené na žiadne konkrétne zariadenie.

2. **Open-source knižnica**
   Je k dispozícii bezplatne každému, kto s tým chce pracovať. Táto funkcia umožňuje každému používateľovi použiť tento modul kedykoľvek a kdekoľvek je to potrebné.

3. **Grafy**
   Tensorflow má lepší výkon na vizualizáciu údajov ako ktorákoľvek iná dostupná knižnica. To uľahčuje prácu v neurónových sieťach.

4. **Paralelizmus**
   TensorFlow na svoje fungovanie využíva GPU a CPU systémy. Užívateľ môže voľne používať akúkoľvek architektúru podľa požiadaviek. Ak to nie je výslovne uvedené, systém používa GPU. Tento proces do určitej miery znižuje využitie pamäte. Vďaka tejto kapacite je Tensorflow vnímaný ako knižnica hardvérovej akcelerácie.

5. **Architektonická podpora**
   Architektúra TensorFlow využíva TPU, vďaka čomu je výpočet rýchlejší ako CPU a GPU. Modely, ktoré sú postavené na TPU, sa dajú ľahko nasadiť v cloude a fungujú rýchlejšie v porovnaní s ostatnými dvoma.

Nevýhody

1. **Rýchlosť**
   Je porovnateľne pomalší a menej použiteľný v porovnaní s jeho konkurenčnými frameworkami.

2. **GPU podpora**
   Tensorflow má iba podporu NVIDIA pre GPU a podporu programovacieho jazyka Python pre programovanie GPU.

3. **Nekonzisteľnosť**
   Tensorflow obsahuje homonymá ako názvy svojho obsahu, čo používateľovi sťažuje zapamätanie a používanie. Jedno meno sa používa na rôzne účely a tu začína zmätok.

4. **Dependency**
   Aj keď TensorFlow zmenšuje veľkosť programu a robí ho užívateľsky prívetivým, pridáva mu ďalšiu vrstvu zložitosti. Každý kód potrebuje na svoje spustenie nejakú platformu, ktorá zvyšuje závislosť.

5. **Architektonické obmedzenie**
   Architektúra TPU Tensorflow umožňuje iba vykonávanie modelov a neumožňuje ich trénovanie.

6. **Slabá podpora systému Windows**
   Okrem všetkých výhod, ktoré má Tensorflow, má veľmi obmedzený súbor funkcií pre používateľov systému Windows. Pre používateľov Linuxu to tak nie je, existuje široká škála funkcií, pokiaľ ide o nich [5].

3 PyTorch

PyTorch je open source framework strojového učenia (ML) založený na programovacom jazyku Python a knižnici Torch. Torch je open source knižnica ML používaná na vytváranie hlbokých neurónových sietí a je napísaná v skriptovacom jazyku Lua. Je to jedna z preferovaných platforiem pre výskum hlbokého učenia. Rámec je vytvorený tak, aby urýchlil proces medzi prototypovaním výskumu a nasadením.

Rámec PyTorch podporuje viac ako 200 rôznych matematických operácií. Popularita PyTorch stále rastie, pretože zjednodušuje vytváranie modelov umelých neurónových sietí. PyTorch používajú hlavne dátoví vedci na výskum a aplikácie umelej inteligencie (AI).

3.1 Fungovanie PyTorch

PyTorch je vo svojej podstate pythonic, čo znamená, že sleduje štýl kódovania, ktorý využíva jedinečné funkcie Pythonu na písanie čitateľného kódu. Python je obľúbený aj pre použitie dynamických výpočtových grafov. Umožňuje vývojárom, vedcom a debuggerom neurónových sietí spúšťať a testovať časť kódu v reálnom čase namiesto čakania na napísanie celého programu.
PyTorch poskytuje nasledujúce kľúčové funkcie a komponenty:

Funkcie

1. **TorchScript**
   Toto je produkčné prostredie PyTorch, ktoré používateľom umožňuje bezproblémový prechod medzi režimami. TorchScript optimalizuje funkčnosť, rýchlosť, jednoduchosť použitia a flexibilitu.

2. **Výpočet dynamického grafu**
   Táto funkcia umožňuje používateľom meniť správanie siete za chodu, namiesto čakania na vykonanie celého kódu.

3. **Automatická diferenciácia**
   Táto technika sa používa na vytváranie a trénovanie neurónových sietí. Numericky vypočítava deriváciu funkcie spätným prechodom v neurónových sieťach.

4. **Podpora Pythonu**
   Pretože PyTorch je založený na Pythone, možno ho použiť s populárnymi knižnicami a balíkmi, ako sú NumPy, SciPy, Numba a Cynthon [6].

Komponenty

1. **Výpočet tenzora**

   Podobne ako pole NumPy – open source knižnica Pythonu, ktorá pridáva podporu pre veľké, viacrozmerné polia – tenzory sú generické n-rozmerné polia používané na ľubovoľné numerické výpočty a urýchľujú ich jednotky grafického spracovania. Tieto multidimenzionálne štruktúry možno prevádzkovať a manipulovať s aplikačnými programovými rozhraniami (API).

2. **Premenné**

   Premenná je uzavretá mimo tenzora, aby udržala gradient. Predstavuje uzol vo výpočtovom grafe.

3. **Parameter**

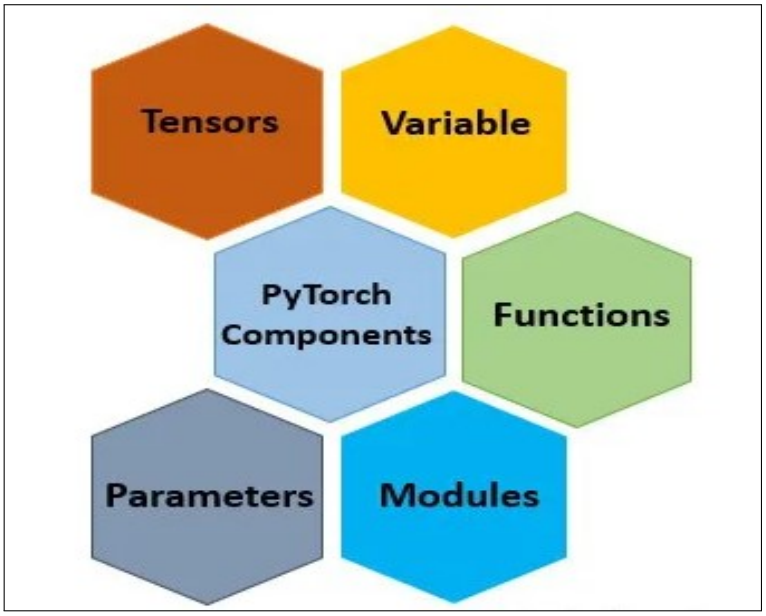
   Parametre sú zabalené okolo premennej. Používajú sa, keď je potrebné použiť parameter ako tenzor, čo pri použití premennej nie je možné.

4. **Modul**

   Moduly predstavujú neurónové siete a sú stavebnými kameňmi stavových výpočtov. Modul môže obsahovať ďalšie moduly a parametre.

5. **Funkcie**

   Toto sú vzťahy medzi dvoma premennými. Funkcie nemajú pamäť na uloženie akéhokoľvek stavu alebo vyrovnávacej pamäte a nemajú žiadnu vlastnú pamäť [7].



Obr. 4: PyTorch komponenty

3.2 Porovnanie s TensorFlow

PyTorch a TensorFlow sa často porovnávajú. Oba tieto frameworky slúžia na strojové učenie. Pričom TensorFlow od spoločnosti Google bol vyvinutý skorej, tak má aj väčšiu komunitu vývojárov a viac dokumentácie.

1. **Dynamické verzus statické**
   Hoci PyTorch aj TensorFlow pracujú na tenzoroch, primárny rozdiel medzi PyTorch a Tensorflow je v tom, že zatiaľ čo PyTorch používa dynamické výpočtové grafy, TensorFlow používa statické výpočtové grafy. S vydaním TensorFlow 2.0 došlo k veľkému posunu smerom k horlivému vykonávaniu a odklonu od výpočtu statického grafu. Dychtivé vykonávanie v TensorFlow 2.0 vyhodnocuje operácie okamžite, bez vytvárania grafov.

2. **Dátový paralelizmus**
   PyTorch používa asynchrónne vykonávanie Pythonu na implementáciu dátového paralelizmu, ale s TensorFlow to tak nie je. TensorFlowe sa musí manuálne nakonfigurovať každá operácia na dátový paralelizmus.

3. **Podpora vizualizácie**
   TensorFlow má veľmi dobrú vizualizačnú knižnicu s názvom TensorBoard. Táto podpora vizualizácie pomáha vývojárom pekne sledovať tréningový proces modelu. PyTorch mal pôvodne vizualizačnú knižnicu s názvom Visdom, ale odvtedy poskytuje plnú podporu aj pre TensorBoard. Používatelia PyTorch môžu využiť TensorBoard na zaznamenávanie modelov a metrík PyTorch v rámci používateľského rozhrania TensorBoard. Pre modely a tenzory PyTorch sú podporované skaláre, obrázky, histogramy, grafy a vložené vizualizácie.

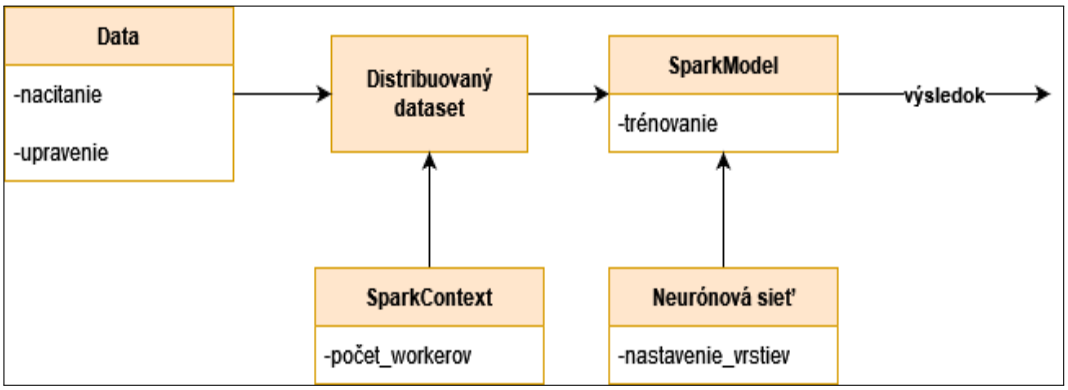
4. **Nasadenie modelu**
   TensorFlow má skvelú podporu pre nasadenie modelov pomocou rámca nazývaného TensorFlow service. Je to rámec, ktorý používa REST Client API na použitie modelu na predikciu po nasadení. Na druhej strane PyTorch neposkytuje rámec, ako je poskytovanie modelov na nasadenie modelov na web pomocou klienta REST.

4 Apache Spark a TensorFlow

Na ukážku integrácie Apache Spark a TensorFlow sme si pripravili jednoduchý príklad skriptu distribuovaného trénovania jednoduchej neúronovej siete.

4.1 O softvére

Ako bolo vyššie spomenuté v našom skripte sa zaoberáme distribuovaným trénovaním neurónovej siete. Konkrétne sme si pri našej neurónovej sieti zvolili jeden z najznámejších datasetov a to MNIST, ktorý sa dá priamo importovať do projektu. Tento dataset sa skladá z obrázkov čísiel od 0 až po 9 a obrázky sú v rozložení 28x28 pixelov. Tento dataset sa musí pred trénovaním najskôr upraviť, čo sa rieši vo funkcií prepareData(), následne tento dataset musíme pretransformovať na odolný distribuovaný dataset, v čom nám pomáha Spark a konkrétne funkcia sparkContext(), v ktorej je zadefinovaný počet workerov (na koľko podmodelov sa to bude deliť - v našom prípade je to na počet dostupných logických jadier kvôli setMaster=local[*]). Tento distribuovaný dataset ďalej vstupuje do Spark modelu, do ktorého vstupuje aj model neurónovej siete. Nakoniec sa tento model neurónovej siete založenej na TensorFlow(Keras) v Spark modeli natrénuje a tak môžeme vyhodnotiť úspešnosť trénovania daného datasetu.



Obr. 5: Diagram programu TF a Spark

4.2 Použitie

- Distribuované trénovanie modelu - v prípade veľkého množstva údajov
- Ladenie hyperparametrov - pre nájdenie najlepšej sady hyperparametrov, čo vedie k skráteniu trénovania a v niektorých prípadoch aj k zníženiu chybovosti

5 Apache Spark a PyTorch

Pre integráciu Apache Spark a PyTorch existuje viacero knižníc. Na ukážku integrácie Apache Spark a PyTorch sme si pripravili príklad s využitím knižnice SparkTorch.

5.1 SparkTorch

SparkTorch je open source knižnica od jedného autora, ktorá integruje Apache Spark a PyTorch. Knižnica ponúka jednoduché API pre vytváranie distribuovaných neurónových sietí. Nevýhodou tejto knižnice je najmä zlá dokumentácia a podpora pre Windows, knižnica je cielená najmä na Linux. Výhodou je jednoduché API. V príklade sa spracuje dataset MNIST, kde sa trénuje viacero inštancií jedného modelu a nakoniec sa vyberie ten s najlepšími parametrami.

Záver

TensorFlow a PyTorch sú rozdielne frameworky, avšak obidva tieto frameworky sa využívajú na rovnaký účel - strojové učenie a umelú inteligenciu. Hlavným rozdielom je, že TensorFlow používa statické výpočtové grafy a PyTorch používa dynamické výpočtové grafy, pričom pri vývoji musí vývojár mať túto skutočnosť na pamäti a aj to, že TensorFlow má lepšiu dokumentáciu pretože, bol vyvinutý skôr.

Pokiaľ sa jedná o integráciu Apache Spark s obidvomi frameworkami, tak sa takáto integrácia používa najmä pri veľkých objemoch dát, pretože Apache Spark zabezpečuje rozdistribuovanie dát medzi jednotlivých workerov, čím dokáže urýchliť proces trénovania. Preto pri trénovaní neurónových sietí s veľkými dátami je vhodné použiť distribuované trénovanie v kombinácií Apache Spark s TensorFlow alebo PyTorch.

Zoznam použitej literatúry

1. AMAZON, AWS. *Introduction to Apache Spark* [Online]. 2020. Dostupné tiež z: https://aws.amazon.com/big-data/what-is-spark/. [cit. 2.12.2022].
2. INFOWORLD. *What is Apache Spark? The big data platform that crushed Hadoop* [Online]. 2020. Dostupné tiež z: https://www.infoworld.com/article/3236869/what-is-apache-spark-the-big-data-platform-that-crushed-hadoop.html. [cit. 2.12.2022].
3. KNOWLEDGEHUT. *Apache Spark Pros and Cons* [Online]. 2019. Dostupné tiež z: https://www.knowledgehut.com/blog/big-data/apache-spark-advantages-disadvantages. [cit. 2.12.2022].
4. GURU. *What is TensorFlow? How it Works? Introduction Architecture* [Online]. 2022. Dostupné tiež z: https://www.guru99.com/what-is-tensorflow.html. [cit. 2.12.2022].
5. GURU. *Advantages and Disadvantages of TensorFlow* [Online]. 2022. Dostupné tiež z: https://www.geeksforgeeks.org/advantages-and-disadvantages-of-tensorflow/. [cit. 2.12.2022].
6. TECHTARGET. *What is PyTorch?* [Online]. 2022. Dostupné tiež z: https://www.techtarget.com/searchenterpriseai/definition/PyTorch. [cit. 2.12.2022].
7. EDUCBA. *What is PyTorch?* [Online]. 2022. Dostupné tiež z: https://www.educba.com/what-is-pytorch/. [cit. 2.12.2022].