VINF dokument Tomáš Kalný

Cieľom mojej práce bolo umožniť vyhľadávanie vedeckých publikácií zo stránky IEEE na základe témy článku (topic). Vedecké články obsahujú veľmi často kľúčové slová avšak nie vždy sa týkajú celkovej oblasti o čom práca je, pričom častokrát kľúčové slová nezodpovedajú téme práce. Existuje mnoho vyhľadávačov vedeckých článkov, avšak väčšina vyhľadávaní sa sústredí skôr na vyhľadávanie podľa mena autora, názvu článku alebo kľúčového slova. Mojim cieľom bolo extrahovať z kľúčových slov témy článkov a umožniť vyhľadávanie na základe tejto témy/oblasti. Zameral som sa hlavne na 2 druhy kľúčových slov author keys a IEEE keys, ktoré IEEE poskytuje (skoro) ku každému článku.

Súčasné riešenia

Existuje viacero archívov vedeckých článkov, ktoré umožňujú vyhľadávanie vedeckých článkov akými sú napríklad arxiv, už spomínané IEEE, ACM digital library atď. Tie umožňujú viacero typov vyhľadávaní či už je to podľa názvu, podľa typu publikácie (kniha, článok, časopis), podľa roku, autorov atď. Arxiv napríklad umožňuje vyhľadávať len v určitej oblasti (napr. matematika) ale aj tak treba zadať ešte nejakú dodatočnú informáciu k článku. Nepodarilo sa mi však, nájsť vyhľadávač, ktorý podporuje aj vyhľadávanie na základe topicu, keďže väčšinou takýmito informáciami nedisponujú.

Popis riešenia

Riešenie som implementoval vo viacerých krokoch. Najprv bolo potrebné vyextrahovať samotné vedecké články, konkrétne metadáta o vedeckých článkov s IEEE explore. Celý projekt som implementoval v jazyku Python. Crawler som implementoval pomocou knižnice selenium, ktorá umožňuje interakciu so stránkou a spúšťa pri crawlovaní prehliadač, ktorý vykonáva automatizované kroky. Dôvodom využitia tejto knižnice bolo, že IEEE načítava dokumenty vo vyhľadávačí postupne až po prvotnom načítani stránky cez Javascript a teda samotný čistý get na danú doménu neobsahoval žiadny zoznam článkov.

Po uložení html súborov, bolo potrebné vyextrahovať z nich potrebné dáta o článkoch (názov, autori, abstrakt, kľúčové slová, počet strán atď.) To bolo implementované pomocou regexov, pričom html stránky boli uložené v 1 veľkom textovom súbore, kde 1 riadok bola 1 html stránka. Následne som extrahované informácie zapisoval do samostatného csv súboru.

Neskôr som ešte upravil štruktúru csv súboru a pridal som stĺpec merged_keys, ktorý sa skladal zo spojených autorovych kľúčových slov a IEEE kľúčových slov, pričom bolo zabezpečené aby tieto merged_keys neobsahovali duplikáty. Po prvotnej extrakcií čŕt som implementoval aj jednoduchý index a vyhľadávanie. Konkrétne prvý index a vyhľadávanie bol čisto v pythone a podporuje vyhľadanie kľúčového slova alebo viacerých kľúčových slov, ktoré musia byť v článku (AND) operácia. Taktiež som implementoval index a vyhľadávanie aj v pylucene, ktorý podporoval vyhľadávanie kľúčového slova v článku.

V druhej časti projektu bolo potrebné distribuovane spracovať dump wikipédie na clustri. Na to som použil pyspark, kde si pri spúšťaní programu pošlem na každý node moje extrahované dáta, následne z nich vyberiem zoznam všetkých merge-nutých kľúčových slov. Potom distribuovane čítam wiki dump cez pyspark dataframe, k dataframe si vyrobím nový stĺpec key, ktorý obsahuje kľúč zo zoznamu kľúčov, ktorý matchem buď podľa názvu wiki stránky (title) alebo redirectu stránky. Následne odfiltrujem stránky, ktoré nematchli žiaden z kľúčov. Zo zvyšných stránok sa snažím extrahovať tému pomocou regexov, jednoduchých fráz a kategórií, ktoré sa nachádzajú vždy na konci wiki stránky. Ak nájdem frázu, ktorá indikuje, že moje kľúčové slovo je oblasť/téma, uložím dané slovo ako prvé slovo do listu kategórií danej stránky a vyextrahujem aj kategórie zo spodnej strany stránky. Ak nájdem nejakú frázu, kde očakávam, že kľúčové slovo patrí do nejakej oblasti, extrahujem znova kategórie z konca stránky a vyberiem druhú kategóriu v poradí ako primárnu kategóriu (dám ju na prvú pozíciu). Ak nenájdem žiadnu z fráz, extrahujem kategórie a predpokladám, že prvá v poradí označuje danú tému.

Týmto spôsobom si vytvorím python slovník v tvare kľúč: zoznam tém. Následne z tohto slovníka vytvorím dataframe s 2 stĺpcami. Tento dataframe si separátne uložím ale tiež ho rovno spojím s mojimi dátami. Tie si tiež načítam do dataframe, vytvorím z nich viacero záznamov pre jeden článok tak, aby jeden záznam mal 1 kľúč (cez funkciu explode), následne spravím join podľa kľúču s dataframe-om, ktorý som vytvoril zo slovníka a zgrupím záznamy podľa názvu, linku, autorov pričom spojím témy do jedného stringu (zgrupovanie je vlastne len aby som spojil záznamy, ktoré sa týkajú toho istého článku, avšak na začiatku som potreboval rozbiť 1 článok na viacero záznamov ak mal viacero kľúčov).

Nakoniec som z týchto spojených dát vytvoril nový index a vyhľadávanie, ktoré teraz zaindexuje hodnoty všetkých stĺpcov (je možné vyhľadávať aj pomocou title aj pomocou topics) a má 2 rôzne funkcionality. Ak zadáme query syntax v tvare key:value a operátory

AND OR s ďalšími key, values, vyhľadávame pomocou pylucene a zobrazíme výsledky. Ak chceme porovnať, frekvencie výskytov vo výsledkoch, je potrebné zadať query v podobe key:value, key2:value2 bez operátorov AND a OR. Taktiež viacslovné spojenia v druhom prípade nedávame do úvodzoviek a key:value páry rozdeľujeme cez čiarky. V takomto prípade, sú výsledky matchnuté a zoradené podľa najčastejšie vyskytujúcej sa hodnoty. Takto napríklad, vieme zistiť, ktoré z nami zadaných tém sú najčastejšie.

Popis dát

Dáta, s ktorými som pracoval po prvotnom vyextrahovaní z HTML súborov, boli vo formáte csv súboru, ktorý mal nasledovnú štruktúru.

link	title	author	content	publisher	year	pages	ieee_keys
link1	title1	author1	content1	publisher1	year1	pages1	key1;key2;key

Riadky boli teda jednotlivé záznamy, pričom kľúče boli oddelené cez ; a uložené ako string hodnota. Po spojení kľúčov zo stĺpcov ieee_keys a author_keys, nám len vznikol ďalší stĺpec, kde boli spojené kľúče bez duplikátov.

Na začiatku sa mi podarilo extrahovať zhruba 74 000 záznamov.

Po spojení s wikipédiou, sa dáta rozšírili o ďalší stĺpec (combined_topics). Tu už zostalo len 53 000 záznamov, nakoľko okolo 21 000 záznamov nemali žiadne kľúče a pri spájaní s dátami z wikipédie boli odstránené.

link	title	author	content	publisher	year	pages	ieee_keys
link1	title1	author1	content1	publisher1	year1	pages1	key1;key2;key

Názov prvotných extrahovaných dát: data.csv

Názov spracovaných dát so spojenými kľúčmi: data_merged_keys.csv

Názov finálnych dát je: Kalny_df_join_finalv2.csv

Spustenie

Projekt sa skladá z 2 hlavných častí

- crawler, extrahovanie informácií a jednoduchý index a search
- zdrojový kód k distribuovanému spracovaniu wikipédie a finálny index a search

Spustenie crawlera

Pre spustenie crawlera je potrebné spustiť súbor crawler_IEEE.py. Po spustení crawlera bude mať priečinok v ktorom sa nachádza zdrojový kód nasledovnú štruktúru.

- visited_pages_history - priečinok s textovými súbormi, kde ukladáme posledné navštívené stránky, aby sa v programe dalo po opätovnom spustení pokračovať tam, kde sme skončili
- data_ieee/concat - v tomto priečinku sa nachádza textový súbor, ktorý obsahuje všetky stiahnuté stránky
- ieee_visited_links.txt - všetky navštívené stránky, aby sme predišli duplicitným záznamom

Následne sa zadajú vstupné parametre a začne sa crawlovanie.

Extrahovanie informácií

Extrakciu informácií z html súborov spustíme pomocou ieee_information_extractor.py. Pred extrahovaním informácií je potrebné spustit crawlera, ktorý vytvorí aj priečinky potrebné na správne fungovanie programu.

Spojenie kľúčov

Pred indexáciou a vyhľadávaním je potrebné spustiť súbor join_keys.py, ktorý spojí IEEE_keys a author_keys bez duplikátov a vytvorí nový stĺpec merged_keys.

Prvotné indexovanie a vyhľadávanie

Na prvotné indexovanie existujú 2 implementácie.

Simple_index.py

Prvú som implementoval v čistom pythone a podporuje vyhľadávanie podľa key:value pričom podporuje aj viacero hodnôt, ktoré sa reťazia cez and. Program si na začiatku vypýta cestu k súboru. Tento index zaindexuje všetky 3 stĺpce ieee_keys, author_keys ako aj merged_keys.

Syntax vyhľadávania je key:value pričom viacero hodnôt je oddelených čiarkov. Ak chceme obmedziť počet zaindexovaných záznamov, stačí zmeniť hodnotu premennej MAX v kóde. Ak chceme zaindexovať všetky záznamy nastavíme MAX na -1, to platí pri všetkých indexeroch, v mojom projekte.

Pylucene_indexer_keys.py

Druhý index a vyhľadávanie má rovnakú funkcionalitu ako prvotný index v čistom pythone avšak je implementovaný pomocou knižnice pylucene. Preto je potrebné spustiť ho v dockeri, kde je nainštalovaný pylucene.

Spojenie dát s wikipédiou

Zdrojový kód ku spájaniu dát sa nachádza v súbore wiki_parser.py na spustenie je potrebné mať nainštalovaný spark a hadoop teda ja som testoval kód lokálne cez docker image so sparkom a hadoopom a následne po otestovaní sa spájanie realizovalo na mesos clustri. Pred spustením je potrebné do hadoop súborového systému vložiť súbor data_merged_keys.csv s mojimi zatiaľ extrahovanými dátami.

Následne kód spustíme cez spark-submit s parametrami

```
spark-submit --master mesos://147.213.75.180:5050 \
--packages com.databricks:spark-xml_2.12:0.17.0 \
--deploy-mode client
--conf spark.executor.memory=24G
 --conf spark.executor.uri=hdfs://147.213.75.180:8020/user/hadmin/spark-3.4.1-bin-hadoop3.tgz
 --files data_merged_keys.csv
wiki_parser.py merged_keys data_merged_keys.csv hdfs://147.213.75.180:8020/user/hadmin/enwiki-latest-pages-articles.xml
```

V podstate sa mení len master parameter, kde je potrebné zadať IP adresu, kde beží cluster, prípadne mesos nahradiť spark ak beží čistý spark. Posledný riadok je python súbor, ktorý spúšťame spolu s argumentami, mení sa v podstate len posledný argument (cesta k dumpu wikipédie), prípadne predposledný argument (názov súboru, kt. čítame).

Výsledok sa uloží do hadoopu do priečinkov Kalny_df_join a Kalny_exported_key_topics.

Finálny index

Finálny index je implementovaný pomocou pylucene v súbore pylucene_general_index_search.py. Po spustení je potrebné len zadať argumenty, ktoré si pýta program, ako je cesta k súboru, kde sa nachádzajú dáta, ktoré máme zaindexovať a názov priečinku do ktorého sa index uloži (používam index, ktorý sa ukladá na disk).

Vyhodnotenie

Podarilo sa nám extrahovať zhruba stotinu celkového obsahu IEEE archívu, čo značne ovplyvnilo možnosti vyhodnocovania nášho vyhľadávania. Zvoliť ako referenčný model IEEE nie je vhodné, vzhľadom na to, že pri väčšine dopytov IEEE dáva do popredia záznamy, ktorými nedisponujeme. Pri spájaní s wikipédiou sa nám nepodarilo nájsť ku všetkým záznamom oblasť témy, ktorej sa článok venuje podľa keywords. To môže byť spôsobené, tým, že kľúčové slovo sa buď nenachádzalo v dumpe alebo stránka neobsahovala žiadne kategórie. Konkrétne sme nenašli žiadne informácie k 18134 záznamom z 25039. Čiže sme našli len zhruba 30% kľúčov.

To však netvorilo až taký veľký problém vzhľadom na to, že jeden článok má viacero kľúčov. Konkrétne z 53 141 záznamov, ktoré nám zostali po spojení s wikipédiou iba 7 970 nemalo žiadnu tému čo je zhruba 14%.

Taktiež z pôvodných 74 tisíc záznamov nám zostalo po spojení s wikipédiou 53 tisíc, čo predstavuje stratu zhruba 29%.

Čo sa týka najčastejších oblastí, výskumu v extrahovaných článkoch, okrem spomínaných 5292 nenájdených oblastí, boli najčastejšie oblasti:

- applied mathematics 2098 záznamov
- computer architecture 1687 záznamov
- voltage 1647 záznamov
- personality traits 1641 záznamov
- training 1295 záznamov
- cognition 1179 záznamov
- topology 1109
- covariance and correlation 995

Ako referenčný model som si nevedel zobrať IEEE, z ktorého som pôvodne crawloval stránky, nakoľko nemám všetky dáta, ktoré má IEEE a teda logicky nedáva zmysel porovnávať výsledky vyhľadávania, s IEEE keď IEEE vráti záznamy, ktoré nemám. Preto som si vždy pred vyhľadávaním vytvoril očakávaný výstup, ktorý som porovnával s výstupom môjho searchera. Čo sa týka precision a recall, bolo pomerne náročné napasovať tieto výpočty do môjho zadania, väčšinou mi totiž program správne matchol všetko alebo nematchol nič, preto som sa snažil nájsť len nejaké špeciálne prípady. Precision a recall teda počítam iba v 2 testoch.

Testy na precision a recall

Query: author:”Wei Xu”

Ako môžeme vidieť náš vyhľadávač matchol Wei Xu aj v prípade, kedy sa nejednalo o jedného autora ale wei xu bolo rozbité medzi 2 autormi. Tieto výsledky teda nie sú správne. Očakávaný výstup články od autora Wei Xu, ktoré máme k dispozícií (z 21). Toto je spôsobené tým, že pri indexácií nerozbíjam stĺpec autorov ale rovno zaindexujem celý text (viacero autorov naraz).

Link/id dokumentu	Autori	Očakávaný dokument	Očakávaný autori
https://ieeexplore.ieee.org/document/5497218	Wei Xu;Chunming Zhao	https://ieeexplore.ieee.org/document/5497218	Wei Xu;Chunming Zhao
https://ieeexplore.ieee.org/document/5497218	Wei Xu;Tong Zhang	https://ieeexplore.ieee.org/document/5497218	'Wei Xu;Tong Zhang'
https://ieeexplore.ieee.org/document/10269747	Maoxin Zhang;Yi Liu;Wei Xu	https://ieeexplore.ieee.org/document/10269747	Maoxin Zhang;Yi Liu;Wei Xu
https://ieeexplore.ieee.org/document/4806136	'Wei Xu;Tong Zhang;Yiran Chen'	https://ieeexplore.ieee.org/document/4806136	'Wei Xu;Tong Zhang;Yiran Chen'
https://ieeexplore.ieee.org/document/10257968	Haowen Zhang;Jianliang Lu;Wei Xu	https://ieeexplore.ieee.org/document/10257968	Haowen Zhang;Jianliang Lu;Wei Xu
https://ieeexplore.ieee.org/document/5575420	Yongchang Zhang;Jianguo Zhu;Wei Xu;Youguang Guo	https://ieeexplore.ieee.org/document/5575420	Yongchang Zhang;Jianguo Zhu;Wei Xu;Youguang Guo
https://ieeexplore.ieee.org/document/10256826	Bowen Tian;Long Rao;Wei Xu;Wenqing Cheng	https://ieeexplore.ieee.org/document/10256826	Bowen Tian;Long Rao;Wei Xu;Wenqing Cheng
https://ieeexplore.ieee.org/document/4360116	Wei Xu;David L. Mathine;Jennifer K. Barton	https://ieeexplore.ieee.org/document/4360116	Wei Xu;David L. Mathine;Jennifer K. Barton
https://ieeexplore.ieee.org/document/6165309	Shuiwang Ji;Wei Xu;Ming Yang;Kai Yu	https://ieeexplore.ieee.org/document/6165309	Shuiwang Ji;Wei Xu;Ming Yang;Kai Yu
https://ieeexplore.ieee.org/document/6212462	Qi Wu;Fei Sun;Wei Xu;Tong Zhang'	https://ieeexplore.ieee.org/document/6212462	Qi Wu;Fei Sun;Wei Xu;Tong Zhang'
https://ieeexplore.ieee.org/document/6180167	Shan Chu;Peng Wei;Xu Zhong;Xin Wang;Yu Zhou'	https://ieeexplore.ieee.org/document/5660072	Wei Xu;Saurabh Sinha;Tawab Dastagir;Hao Wu;Bertan Bakkaloglu;Donald S. Gardner;Yu Cao;Hongbin Yu
https://ieeexplore.ieee.org/document/10110359	Xiang Li;Meihui Jiang;Wei Xu;Thomas Wu;Dongdong Zhang'	https://ieeexplore.ieee.org/document/10110359	Xiang Li;Meihui Jiang;Wei Xu;Thomas Wu;Dongdong Zhang'
https://ieeexplore.ieee.org/document/5352236	'Wei Xu;Hongbin Sun;Xiaobin Wang;Yiran Chen;Tong Zhang	https://ieeexplore.ieee.org/document/5352236	'Wei Xu;Hongbin Sun;Xiaobin Wang;Yiran Chen;Tong Zhang
https://ieeexplore.ieee.org/document/5492208	'Yi Lei;Zhengming Zhao;Shuping Wang;David G. Dorrell;Wei Xu'	https://ieeexplore.ieee.org/document/5492208	'Yi Lei;Zhengming Zhao;Shuping Wang;David G. Dorrell;Wei Xu'
https://ieeexplore.ieee.org/document/5200366	Huazhong Ning;Wei Xu;Yue Zhou;Yihong Gong;Thomas S. Huang'	https://ieeexplore.ieee.org/document/5200366	Huazhong Ning;Wei Xu;Yue Zhou;Yihong Gong;Thomas S. Huang'
https://ieeexplore.ieee.org/document/10272281	Min Zhou;Wei Xu;Xuan Liu;Zixuan Zhang;Hairong Dong;Ding Wen	https://ieeexplore.ieee.org/document/10272281	Min Zhou;Wei Xu;Xuan Liu;Zixuan Zhang;Hairong Dong;Ding Wen
https://ieeexplore.ieee.org/document/5617262	Wen-Chen Chu;Yi-Ping Chen;Zheng-Wei Xu;Wei-Jen Lee	https://ieeexplore.ieee.org/document/10269747	Maoxin Zhang;Yi Liu;Wei Xu
https://ieeexplore.ieee.org/document/6198907	Yangyang Pan;Yiran Li;Hongbin Sun;Wei Xu;Nanning Zheng;Tong Zhang	https://ieeexplore.ieee.org/document/6198907	Yangyang Pan;Yiran Li;Hongbin Sun;Wei Xu;Nanning Zheng;Tong Zhang
https://ieeexplore.ieee.org/document/10260446	Hong-Wei Xu;Run-Zhi Tan;Qun-Long Chen;Yu Zhou;Wei Qin	https://ieeexplore.ieee.org/document/1578730	O. Borzenko;Wei Xu;M. Obsniuk;A. Chopra;P. Jasiobedzki;M. Jenkin;Y. Lesperance
https://ieeexplore.ieee.org/document/10257975	Yufeng Li;Lei Jiang;Peng Xu;Wei Xu;Yufei Liu;Boyang Xing;Tong Yan	https://ieeexplore.ieee.org/document/10257975	Yufeng Li;Lei Jiang;Peng Xu;Wei Xu;Yufei Liu;Boyang Xing;Tong Yan
https://ieeexplore.ieee.org/document/1578730	'O. Borzenko;Wei Xu;M. Obsniuk;A. Chopra;P. Jasiobedzki;M. Jenkin;Y. Lesperance'	https://ieeexplore.ieee.org/document/1578730	'Yuying Zou;Haotian Li;Yunfan Ren;Wei Xu;Yihang Li;Yixi Cai;Shenji Zhou;Fu Zhang'

V tomto teste sme nesprávne matchli 3 výsledky. Kontrovali sme prvých 20 výsledkov.

Precision = 18/21 = 0,85

Recall = 18/21 = 0,85

Query: combined_topics:voltage, combined_topics:voltage regulation

V takomto vyhľadávaní sa zobrazujú štatistiky jednotlivých topicov, ako na prvý pohľad vidíme,

topic voltage by mal byť matchnutý aj v prípade voltage regulation, keďže je všeobecnejší a voltage regulation je jeho podmnožina. Avšak náš vyhľadávač túto skutočnosť nezaznamená a v prípade, že narazíme na topic voltage regulation, zaradí ho len pod voltage regulation.

```
Topic:voltage, count:1647
Topic:voltage regulation, count:76
```

Pričom teda po správnosti by voltage mal mať počet výskytov 1723.

Precision je v tomto prípade 1, pretože z vrátených sú všetky výsledky relevantné.

Avšak **recall** je 1647/1723 = 0,95

Jednotkové testy

Dopyt:

merged_keys:”data science” AND publisher:IEEE AND content:python

Očakávaný výsledok:
Link: https://ieeexplore.ieee.org/document/10261586
Merged keys ['', 'education', 'big data', 'software', 'training', 'applied statistics', 'python', 'project-based teaching system', 'computer science', 'problem-solving', 'experiment course', 'data science']
Topics: ['Education ', 'Training ', 'Not found']
Authors: ['Xiaohua Zhang;Wenlong Li;GuiXin Wang']

Výsledok programu: program našiel požadovaný dokument

Dopyt:

Write your query combined_topics:topology AND title:"analysis swarm”

Očakávaný výsledok:

Title: Experimental Analysis of Bound Handling Techniques in Particle Swarm Optimization
Link: https://ieeexplore.ieee.org/document/6163405
Merged keys ['', 'optimization', 'electronic mail', 'vectors', 'particle swarm optimization (pso)', 'particle swarm optimization', 'mirrors', 'topology', 'constrained optimization', 'benchmark testing']
Topics: ['Mathematical optimization', 'Topology ', 'Not found']
Authors: ['Sabine Helwig;Juergen Branke;Sanaz Mostaghim']

Výsledok programu: program nenašiel žiadny dokument.

Príčinou je, že v title hľadá program presne slovné spojenie analysis swarm, pričom v dokumente sú medzi týmito slovami ďalšie slová. Ak by sme danú frázu rozbili na title:analysis AND title:swarm daný dokument by sme našli.

Dopyt: combined_topics:wire, combined_topics:data security, combined_topics:quality control

Očakávaný výsledok: všetky články obsahujúce jednu z týchto tém. Tému wire by mali mať 4 články, rovnako aj tému data security, čo sa týka quality control tá by mala byť v 3 článkoch. Na konci budú aj tieto štatistiky.

Výsledok programu: program správne našiel všetky články, avšak pre topicy s rovnakou početnosťou ich zoradil abecedne (teda namiesto článkov s témou wire, idú ako prvé články s témou data science).

Dopyt :combined_topics:"Machine learning" AND combined_topics:privacy AND title:federated

Očakávaný výsledok:

FIDES: A Proposal for Federated Accountability in the Compute Continuum

AdaDpFed: A Differentially Private Federated Learning Algorithm With Adaptive Noise on Non-IID Data

Data-Similarity Based Integrated Federated and Centralized Learning for 6G Communications

Federated Learning Security and Privacy-Preserving Algorithm and Experiments Research Under Internet of Things Critical Infrastructure

LDS-FL: Loss Differential Strategy based Federated Learning for Privacy Preserving

Federated Learning for Beginners: Types, Simulation Environments, and Open Challenges

Privacy Preservation in Federated Learning: its Attacks and Defenses

Výsledok programu:

FIDES: A Proposal for Federated Accountability in the Compute Continuum

Privacy Preservation in Federated Learning: its Attacks and Defenses

LDS-FL: Loss Differential Strategy based Federated Learning for Privacy Preserving

Federated Learning for Beginners: Types, Simulation Environments, and Open Challenges

Data-Similarity Based Integrated Federated and Centralized Learning for 6G Communications

Federated Learning Security and Privacy-Preserving Algorithm and Experiments Research Under Internet of Things Critical Infrastructure

AdaDpFed: A Differentially Private Federated Learning Algorithm With Adaptive Noise on Non-IID Data

Vidíme, že čo sa týka obsahu program našiel všetky relevantné dokumenty, jediný rozdiel je v poradí. Očakávané poradie bolo poradie, v ktorom sa nachádzajú dokumenty v csv súbore, avšak pylucene trochu zmenil poradie.

Celkovo by sme vedeli tieto unit testy vyhodnotit pomocou precision a recall. Čo sa týka TP - 1 + 7 + 11 = 19

FP - 0, FN - 1, TN - 0

Precision = TP/TP+FP = 19/19 = 1

Recall = TP/TP+FN = 19/20 = 0,95