УДК 004.94

Д. Галайко, Ю. Олійник

ЗАСТОСУВАННЯ СХОВИЩ ДАНИХ ДЛЯ ВИЯВЛЕННЯ ПЛАГІАТУ В ТЕКСТОВИХ ДОКУМЕНТАХ

Анотація. Наразі більшість інформації зберігається в мережі Інтернет. Це прискорило розповсюдження та створення нової інформації. З одного боку це надає можливість легко спілкуватися та отримувати доступ до велетенської бібліотеки інформації людства. З іншого боку, існує ймовірність що отримана інформація буде направлена не на розвиток людства та створення прогресу та здорової конкуренції, а направлена на копіювання та плагіат чужих ідей та праць. Робота присвячена застосуванню сховищ даних для виявлення плагіату в текстових документах, в результаті якої побудовано масштабовану програмну архітектуру. Проведено дослідження ефективності застосованих сховищ даних та виконано їх порівняльний аналіз. Програмне забезпечення реалізоване мовою Python та Go із застосуванням бібліотек spaCy, pymorphy3.

Ключові слова: виявлення плагіату, NLP, потік, Apache Hive, DynamoDB, сховища даних.

Вступ

Виявлення текстових збігів є основною задачею для виявлення запозичень або виявлення пропаганди. Задача масштабованості таких рішень є особливо актуальною у сучасному світі, коли інформація створюється та розповсюджується терабайтами за секунду. Для обробки та аналізу такого обсягу інформації в найкоротші проміжки часу потрібна нова ефективна програмна архітектура.

Метою дослідження є порівняння різних сховищ даних в задачах виявлення плагіату та побудова масштабованої архітектури здатної легко масштабуватися без змін програмного коду.

Матеріали та методи

Для виявлення запозичень існує декілька підходів: використання машинного навчання або алгоритмічний. Для побудови хорошої моделі машинного навчання потрібен великий об’єм розміченого тренувального датасету та доступ до необмеженої кількості обчислювальної потужності. При появі нових даних такі моделі треба донавчати. Конкурентним напрямком є використання алгоритмічного підходу, який є більш точним, результати можна легко протестувати та наповнення сховища відбувається легко та розподілено за рахунок виконання не складної математичної функції. В даній статті сфокусуємося на паралелізації алгоритму виявлення плагіату, оскільки дана тема є не розкритою та має місце для застосування[1].

 ISSN 1560-8956

Щоб ефективно виявляти запозичення на великих обсягах інформації потрібно побудувати систему, яка буде приймати потік даних на вхід для перевірки та віддавати результат перевірки у вихідний потік. Також така система повинна мати можливість отримувати дані про запозичення для динамічного до наповнення бібліотеки документів.

Для виконання дослідження було зібрано датасет студентських бакалаврських та магістерських робіт КПІ розміром понад 20 000 записів[2]. Програмне забезпечення було реалізовано мовою Python[3] та Go[4], для проведення морфологічного використано популярну бібліотеку spacy[5], яка має підтримку української, російської, англійської та багатьох інших мов.

Алгоритм для обробки тексту є розширеною версією алгоритму пошуку плагіату [1] та наведено на рисунку 1. Даний алгоритм було розширено використанням словника синонімів. Даний словник було розроблено з підтримкою української, російської та англійської мови. Розроблена бібліотека була розповсюджена під назвою synonymset[6].

Рисунок 1. Алгоритм виявлення текстових запозичень

Для масштабування рішення необхідна можливість легкої масштабованості сховища даних. Для цього проведено аналіз 3-х найпопулярніших сховищ різної архітектури: PostgreSQL[7], DynamoDB[8], Apache Hive[9]. Кожне сховище має свої переваги та недоліки, для виявлення найкращого було проведена серія експериментів.

Під час вибору сховищ даних проаналізовано джерела на дану тематику. Так обширний огляд надає книжка по проектуванню високонавантажених систем[10]. Найбільш розповсюдженими є реляційні сховища, тому для повноцінного експерименту їх не можна оминути. Також було обрано представника masterless сховищ DynamoDB, який запозичує багато ознак в Cassandra[11]. Під час побудови даного сховища основний вклад слугувала книжка[12] присвячена різноманітним тонкощам побудови правильної архітектури. Останнє сховище було обрано як дослідження можливостей аналітичних засобів для роботи в даному напрямі. Основні ідеї роботи та аналізу були отримані з джерела[13] по роботі з даним сховищем.

Результати та основний матеріал дослідження

Для порівняння ефективності сховищ даних для виконання перевірки на плагіат було виконано серію тестів на пошук співпадінь з документами та додавання документа в колекцію.

Apache Hive. На рисунках 2-5 показаний процес наповнення сховища Apache Hive з використанням механізму партиціонування та в таблиці 1 показані результати отримані при експерименті. Партиції тільки сповільнили процес наповнення сховища та читання, оскільки для вставки потрібно виконувати розбивку даних, а для читання ця розбивка є малоефективною оскільки потрібен перегляд більшості з партицій та об’єднання їх результатів. Спосіб без партицій навпаки показав кращий результат оскільки для вставки великого об’єму інформації достатньо скопіювати файл у систему HDFS. Під час експерименту було проведено пошук співпадіння хешів конкретного файлу з сховищем даних, наповненим датасетом. Оскільки деякі сховища мають кеш на пошук це унеможливить їхню оптимізацію при проведенні експериментів та допоможе отримати більш точні результати. Останній рядок таблиці відповідає за середнє арифметичне процесу пошуку.

```
hive> LOAD DATA INPATH '/tables_data/word_sequences/fa8c2a18-f052-4475-b3d9-18e8586ee941.csv' INTO TABLE word_sequences;
Loading data to table default.word_sequences
OK
Time taken: 0.203 seconds
```

Рисунок 2. Наповнення сховища датасетом (20 000 документів)

 ISSN 1560-8956

```
hive> INSERT INTO word_sequences_buckets SELECT * FROM word_sequences;
Query ID = halaiko_danylo_20231123172204_d08b82c0-7325-4438-b4e0-ddd284834dd2
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1700753571733_0005)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED     17          17        0        0       0       0
Reducer 2 ...... container     SUCCEEDED    100         100        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1           1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 522.18 s
----------------------------------------------------------------------------------------------
Loading data to table default.word_sequences_buckets
OK
Time taken: 529.044 seconds
```

Рисунок 3. Наповнення сховища датасетом
з використання механізму партицій Apache Hive

```
hive> SELECT * FROM documents
    > WHERE document_uuid IN (
    > SELECT document_uuid FROM word_sequences ws
    > INNER JOIN input_partioned I ON i.word_sequence=ws.word_sequence
    > WHERE i.index=9
    > ) LIMIT 10;
Query ID = halaiko_danylo_20231124114414_9c1235cc-bfb8-4919-a8fa-a738228b162b
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1700824174417_0001)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 3 .......... container     SUCCEEDED      1          1        0        0       0       0
Map 2 .......... container     SUCCEEDED     17         17        0        0       0       0
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 105.76 s
----------------------------------------------------------------------------------------------
OK
23c199c5-2de2-40c3-a98e-e647c09cd8e4    22971-Maliuta_magistr   kpi-works/22971-Maliuta_magistr.pdf     documents-content-generated/22971-Maliuta_magistr.txt
18e658aa-bc76-49c3-b42f-ee9c3590439b    22975-Loboda_magistr    kpi-works/22975-Loboda_magistr.pdf      documents-content-generated/22975-Loboda_magistr.txt
6c34959e-414e-4c15-b3e0-a0705ae4ccde    22979-Brage_magistr     kpi-works/22979-Brage_magistr.pdf       documents-content-generated/22979-Brage_magistr.txt
3fd424cc-6c70-4349-bc91-f8421a49d727    22981-Shevliakova_magistr       kpi-works/22981-Shevliakova_magistr.pdf documents-content-generated/22981-Shevliakova_magistr.txt
acd8573b-5001-4370-be9d-048cc3423id1    22983-Sotnikov_magistr  kpi-works/22983-Sotnikov_magistr.pdf    documents-content-generated/22983-Sotnikov_magistr.txt
7b0854c9-61e7-49e8-84b6-6173b00fda91    22986-Serhieieva_magistr        kpi-works/22986-Serhieieva_magistr.pdf  documents-content-generated/22986-Serhieieva_magistr.txt
e91ca966-6b22-43ba-a30b-5111fbc99612    22987-Tsyganov_magistr  kpi-works/22987-Tsyganov_magistr.docx   documents-content-generated/22987-Tsyganov_magistr.txt
8edbcb82-ec71-437e-b974-7e79dd5a5479    22990-Popravka_magistr  kpi-works/22990-Popravka_magistr.pdf    documents-content-generated/22990-Popravka_magistr.txt
60a2f1e2-62d6-4d51-bc34-8dcdb26fb72c    22999-MaiierIS_magistr  kpi-works/22999-MaiierIS_magistr.pdf    documents-content-generated/22999-MaiierIS_magistr.txt
02d21c25-261a-4a86-98d1-683f331f14a6    23004-Pohrebenko_magistr        kpi-works/23004-Pohrebenko_magistr.pdf  documents-content-generated/23004-Pohrebenko_magistr.txt
Time taken: 106.328 seconds, Fetched: 10 row(s)
```

Рисунок 4. Замір часу пошуку плагіату без використання партицій

```
hive> SELECT * FROM documents
    > WHERE document_uuid IN (
    > SELECT document_uuid FROM word_sequences_buckets ws
    > INNER JOIN input_partioned I ON i.word_sequence=ws.word_sequence
    > WHERE i.index=0
    > ) LIMIT 10;
Query ID = halaiko_danylo_20231124120539_9dc5ca5c-866c-4efb-a042-41a7aa95f3c4
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1700824174417_0001)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
Map 4 .......... container     SUCCEEDED      1          1        0        0       0       0
Map 3 .......... container     SUCCEEDED    200        200        0        0       0       0
Reducer 2 ...... container     SUCCEEDED     66         66        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 04/04  [==========================>>] 100%  ELAPSED TIME: 142.07 s
----------------------------------------------------------------------------------------------
OK
03ad9279-a3ff-49f3-9616-50bce87aad89    42782-Semchenko_bakalavr        kpi-works/42782-Semchenko_bakalavr.docx documents-content-generated/42782-Semchenko_bakalavr.txt
07fd8195-f241-4db9-b348-f469d4f481e8    45268-Cholovskyi_bakalavr       kpi-works/45268-Cholovskyi_bakalavr.pdf documents-content-generated/45268-Cholovskyi_bakalavr.txt
08747cc1-86a5-4d36-a21d-2bc71eb40a7a    38906-Lampiha_bakalavr  kpi-works/38906-Lampiha_bakalavr.pdf    documents-content-generated/38906-Lampiha_bakalavr.txt
0b5b8c74-c795-49e8-8c8b-1ff41b35dbac    43205-Bahdasarian_bakalavr      kpi-works/43205-Bahdasarian_bakalavr.pdf        documents-content-generated/43205-Bahdasarian_bakalavr.txt
0c9f4e04-6f8c-4259-bbab-50341465la44    42767-Ivanov_bakalavr   kpi-works/42767-Ivanov_bakalavr.pdf     documents-content-generated/42767-Ivanov_bakalavr.txt
0d24eeee-b630-4a32-9496-fc73ff8495d4    28558-Sofienko_bakalavr kpi-works/28558-Sofienko_bakalavr.pdf   documents-content-generated/28558-Sofienko_bakalavr.txt
0dc61d30-c941-4a50-ae53-fc3322c7b4eb    23213-TkachenkoMH_magistr       kpi-works/23213-TkachenkoMH_magistr.pdf documents-content-generated/23213-TkachenkoMH_magistr.txt
0f4b2b29-13ae-4e85-b65e-f8504093a525    41879-Berenko_bakalavr  kpi-works/41879-Berenko_bakalavr.pdf    documents-content-generated/41879-Berenko_bakalavr.txt
111e3bee-8e1b-49f0-8bde-2f4b9d6adf49    27433-Ovdiienko_magistr kpi-works/27433-Ovdiienko_magistr.pdf   documents-content-generated/27433-Ovdiienko_magistr.txt
133e580a-b0a2-4925-ba7f-0aabe1a31c2f    43724-Shavarska_bakalavr        kpi-works/43724-Shavarska_bakalavr.pdf  documents-content-generated/43724-Shavarska_bakalavr.txt
Time taken: 142.564 seconds, Fetched: 10 row(s)
```

Рисунок 5. Замір часу пошуку плагіату з використанням партиціями

Таблиця 1.

Результати експерименту використання Apache Hive

Номер файлу	Пошук плагіату без використанням партицій, с	Пошуку плагіату з використанням партицій, с
1	104,1	142,564
2	105,811	142,155
3	93,536	119,468
4	87,873	118,471
5	106,229	143,812
6	104,905	123,96
7	105,108	130,389
8	107,155	144,616
9	104,726	152,749
10	105,76	144,897
Середній час, с	102,5203	136,3081

DynamoDB. Під час побудови моделі використовувався Query-First Design[12]. Наповнення сховища показано на рисунку 6. Час запису та читання документів в DynamoDB показано в таблиці 2.

Таблиця 2.

Час запису файлу в сховище та пошук плагіату в DynamoDB

Номер файлу	1	2	3	4	5	6	7	8	9	10	Середній час, с
Пошук плагіату, с	24	31	31	43	35	32	51	21	33	24	32,52
Запис документу до сховища, с	3,3	3	3,2	4	4,3	3,6	5,2	3,6	3	3,4	3,64

 ISSN 1560-8956

e	Format	Status	Import job start time	End time
ıment_info_by_uuid	DynamoDB JSON	Completed	Nov 24, 2023, 16:58:28 (UTC+02:00)	Nov 24, 2023, 17:02:39 (UTC+02:00)
ıment_uuid_by_word_sequence	DynamoDB JSON	Completed	Nov 24, 2023, 16:57:47 (UTC+02:00)	Nov 24, 2023, 17:51:17 (UTC+02:00)

Рисунок 6. Наповнення сховища DynamoDB роботами (20 000 документів)

Час наповнення сховища склав 54 хвилини. Пошук плагіату з документом займає в середньому 33 секунди, а вставка документа в систему складає близько чотирьох секунд.

PostgreSQL. Наповнення сховища показано на рисунку 7. Час читання документів в DynamoDB показано в таблиці 3. Час запису не вдалося заміряти, оскільки при побудованому індексі вставка елементу займає занадто багато часу. Тому вставка великих даних в режимі реального часу не є можливою. Як видно з рисунку 7 наповнення сховища зайняло 8 годин. Побудова індекса на повному сховище зайняла 10 хвилин.

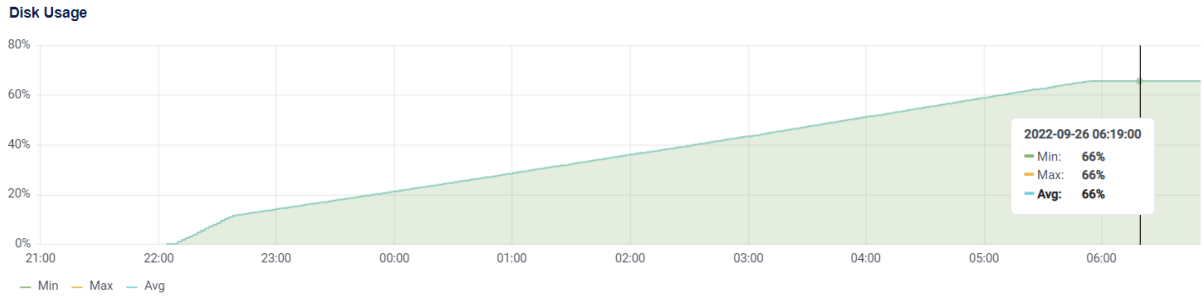

Рисунок 7. Наповнення сховища PostgreSQL роботами (20 000 документів)

Таблиця 3.

Час та пошук плагіату в PostgreSQL

Номер файлу	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Середній час**
Пошук плагіату, с	35	66	88	11	100	83	108	86	78	62	71,78

Порівняння результатів для різних сховищ даних. За результатами тестування сховищ даних можна зробити такі висновки:

– Apache Hive має перевагу по запису великого обсягу інформації перед DynamoDB та PostgreSQL;

– читання даних з Apache Hive є часозатратним процесом;

– пошук в DynamoDB в два рази менший ніж аналогічна операція в PostgreSQL;

– DynamoDB має швидку вставку порівняно з PostgreSQL.

Враховуючи те, що найкращу ефективність показала DynamoDB вона є найкращим кандидатом для розробки масштабованої системи виявлення запозичень тексту в потоковому режимі.

Масштабована архітектура системи виявлення плагіату у текстових документах наведена на рисунку 8.

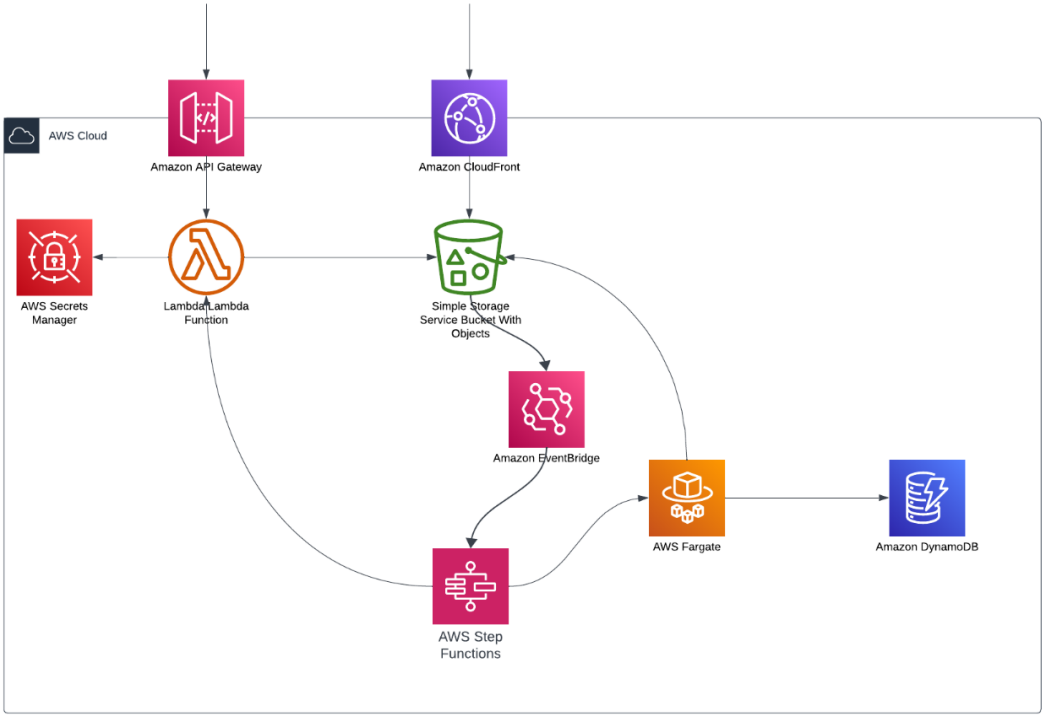

Рисунок 8. Архітектура масштабованого застосунку для виявлення текстових запозичень

Показана архітектура є без серверною. Проте вона може бути легко адаптована під серверну архітектуру. Кожен з компонентів легко масштабується та може виконувати операції паралельно. Головним компонентом є сховище даних: розподілене, масштабоване, що підтримує можливість легкого додавання обчислювальних вузлів. Під час побудови були використані підходи до проектування систем[14].

Під час виконання запиту по API на AWS API Gateway відбувається створення посилання для завантаження файлів на AWS S3. Після завантаження файлу спрацьовує подія в сервісі AWS EventBridg, яка запускає оркестратор AWS StepFunction. AWS Step Functions створює контейнер в AWS Fargate для перевірки файлу на плагіат. Після отримання сигналу про звернення AWS Fargate передає дані до Lambda Function, яка в свою чергу передає посилання на результат обробки

 ISSN 1560-8956

користувачеві. Результат може бути переглянутий за унікальним посиланням, яке веде на CDN. Для безпечної комунікації всі секрети зберігаються в AWS Secrets Manager. Головною особливістю даного підходу є можливість економії ресурсів під час простою. Обчислювальні ресурси створюються тільки при потребі і після завершення вивільнюються.

Система може бути легко масштабована без створення проблем з синхронізацію, скільки кожен з компонентів є легко масштабованим.

Висновки

В роботі проведено дослідження ефективності використання різних сховищ даних в задачі пошуку плагіату в текстових документах. Проведено експерименти з додаванням нових документів до колекції та пошук плагіату. За результатами порівняльного аналізу результатів експериментів було обрано DynamoDB як найшвидше сховище для задачі виявлення текстових запозичень. У той же час Apache Hive має швидку вставку даних та можливість розпаралелювання виконання обробки запитів, що є доволі корисним інструментом при накопичені великого архіву документів і виокремлення необхідної порції даних з нього для подальшої обробки. PostgreSQL найкраще використовувати, коли потрібно застосовувати транзакції, що для даного проекту не є критичним.

Представлено архітектуру програмного забезпечення для виявлення текстових запозичень з можливістю легкого масштабування та розширення без втручання в програмний код.

Отримані в ході дослідження результати можуть бути використані для розробки систем автоматичного виявлення плагіату або пропаганди в мережі Інтернет. Крім того дана система може використовуватись для виявлення першоджерела походження інформації.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Yurii Oliinyk, Danylo Halaiko, Iryna Mukha, Kateryna Lishchuk, Oleksandr Ocheretianyi. Plagiarism Detecting Hash-Based Parallel Method Proceedings of the 7th International Conference, COLINS-2023. Kharkiv, Ukraine April, 2023, 20-21. Volume IV. p.131-143 [Електронний ресурс] – Режим доступу до ресурсу: https://colins.in.ua/wp-content/uploads/2023/10/StudentPoster_Section_new2023.pdf

2. Сайт магістерських та бакалаврських робіт [Електронний ресурс] – Режим доступу до ресурсу: https://ela.kpi.ua

3. Документація мови програмування Python. [Електронний ресурс] – Режим доступу до ресурсу: https://docs.python.org/3/

4. Документація мови програмування Go. [Електронний ресурс] – Режим доступу до ресурсу: https://go.dev/doc/effective_go

5. Документація бібліотеки spaCy. [Електронний ресурс] – Режим доступу до ресурсу: https://spacy.io/usage

6. Бібліотека synonymset [Електронний ресурс] – Режим доступу до ресурсу: https://pypi.org/project/synonymset/

7. Документація PostgreSQL [Електронний ресурс] – Режим доступу до ресурсу: https://www.postgresql.org/docs/

8. Документація DynamoDB [Електронний ресурс] – Режим доступу до ресурсу: https://docs.aws.amazon.com/dynamodb/

9. Документація Apache Hive [Електронний ресурс] – Режим доступу до ресурсу: https://cwiki.apache.org/confluence/display/Hive/LanguageManual

10.Martin Kleppmann, Designing Data-Intensive Applications – 2017

11.Документація Cassandra [Електронний ресурс] – Режим доступу до ресурсу: https://cassandra.apache.org/doc/latest/

12.Jeff Carpenter and Eben Hewitt Cassandra the definitive guide 3rd edition – 2020.

13.Nitin Kumar, Big Data Using Hadoop and Hive – 2021

14.Alex Xu, System Design Interview: An Insider’s Guide -2020