ESIEA

Big data architecture and data processing

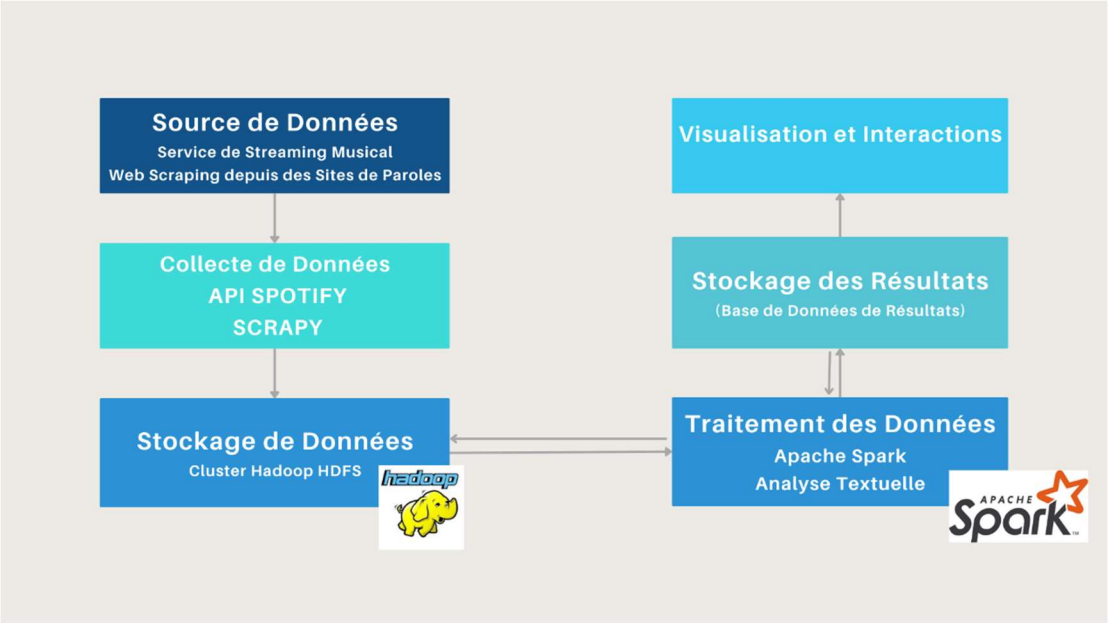
Analyse des genres musicaux basée sur les paroles

CAO Yunjie G1
PATRU Mathieu G2
SPETEIL Guillaume G2

10/01/2024

Table des matières

Les étapes projet



Introduction

Dans le cadre du cours Big data, nous avons réalisé un projet visant à collecter, stocker et traiter des données de paroles de chansons en utilisant diverses technologies, notamment l'API Genius en Java pour le web scraping, le stockage sur un cluster Hadoop, et le traitement ultérieur avec Apache Spark et Docker.

Les objectifs du Projet sont les suivants :

Récupérer les paroles de chanson et prédire le genre et les sentiments derrière celle-ci.

On stocke les paroles de chansons depuis le site Genius à l'aide de l'API Genius en Java.

Stocker les données collectées sous forme de .txt, de noms de chansons et de paroles associées.

Transférer les données vers un cluster Hadoop pour un stockage distribué et évolutif.

Appliquer des opérations de traitement sur le fichier texte stocké en utilisant Apache Spark pour en extraire des informations pertinentes. (Fréquence des mots notamment)

Utilisé un NPL pour analyser nos données et en déduire un sentiment/genre

L'afficher depuis le docker

Web Scraping avec l'API Genius en Java :

Pour la collecte de paroles, notre choix s'est porté sur l'API Genius, intégrée au site du même nom.

Genius, en tant que plateforme web, offre un accès régulièrement actualisé aux paroles de chansons, en en faisant ainsi une ressource privilégiée pour notre projet. Une API dédiée à Genius facilite l'extraction d'informations depuis le site, et dans notre cas, nous avons implémenté son utilisation en Java. La première étape consiste à créer un compte sur Genius pour obtenir un jeton d'accès, essentiel à l'utilisation de l'API.

Cette démarche nous permet d'acquérir des données exhaustives, incluant non seulement le nom des chansons et leurs paroles, mais aussi d'autres informations pertinentes disponibles sur la plateforme Genius. Ces données seront cruciales pour notre analyse et traitement ultérieur dans le cadre de notre projet.

Les données collectées comprennent le nom de la chanson, les paroles et d'autres informations pertinentes disponibles sur le site.

Une fois les données scrappée nous stockons celle-ci dans un fichier texte contentant toutes les lyrics des paroles.

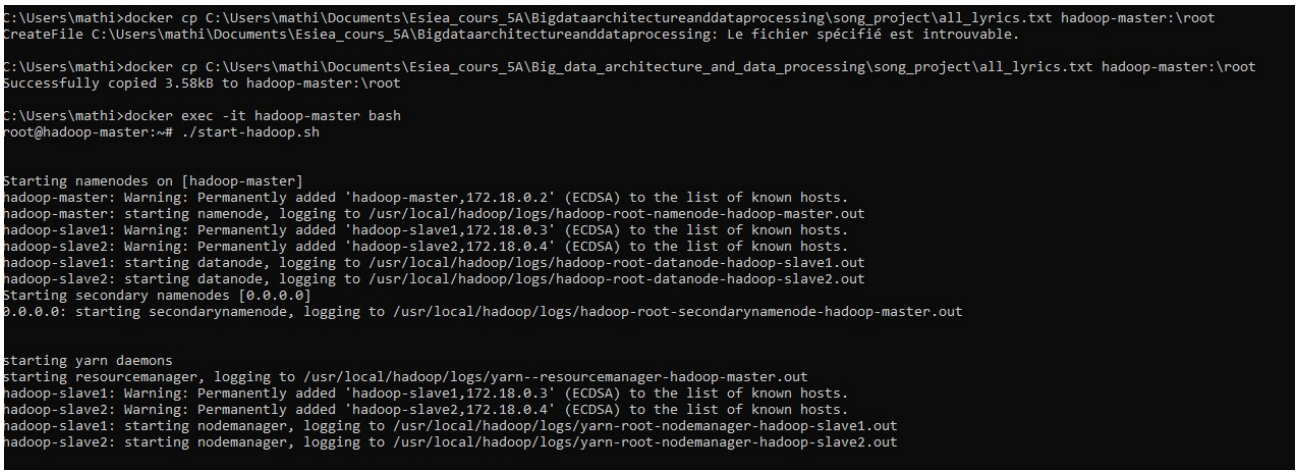
Ce fichier pourra ensuite être traité.

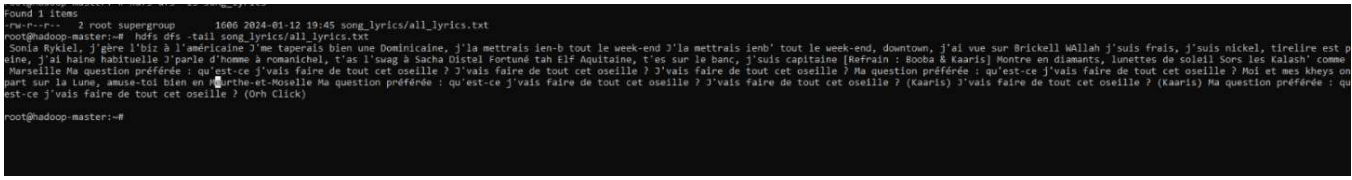
```
Lyrics: [Paroles de "Kalash" ft. Kaaris] [Intro : Booba] Sors les Kalash' comme à
Lyrics have been appended to the file: src/main/resources/input/all_lyrics.txt

Process finished with exit code 0
```

Stockage sur Cluster Hadoop :

Hadoop permet de traiter de grandes quantités de données en les distribuant sur un cluster de machines, permettant ainsi un traitement parallèle efficace. Donc nous allons d'abord upload le fichier des paroles en txt qu'on a récupéré dans l'étape précédente afin qu'il soit disponible sur le cluster. Cela nous permettra de l'utiliser à nouveau ce document de parole lorsque du traitement des données avec Spark ensuite.





On retrouve bien notre fichier texte dans le cluster, donc ensuite lorsque de l'exécution de notre code, il va prendre en entrée ce fichier. Voici la commande utilisée :

hadoop jar song_projet.jar api.LyricsScraper song_lyrics

Traitement avec Apache Spark :

Afin de traiter les paroles de nos chansons nous allons utiliser SPARK ainsi que le logiciel Stanford NLP (Natural Language Processing). Il regroupe les classes et les modèles nécessaires pour effectuer diverses tâches de traitement du langage naturel, comme l'analyse syntaxique, la détection de sentiment, etc.

Spark va nous permettre de retrouver le nombre de fois qu'un mot est utilisé. Notamment grâce à la fonction "AnalyseParoleChanson". Pour cela nous utilisons un JavaRDD, notre RDD (Resilient Distributed Dataset) est utilisé pour charger un fichier texte dans Spark de manière distribuée. Le RDD représente ici les lignes du fichier texte, puis il est transformé pour analyser les paroles d'une chanson en extrayant les mots et en comptant leur fréquence. Enfin, les résultats sont affichés, montrant la fréquence des mots dans les paroles de la chanson.

```
oseille: 16
tout: 22
antihess: 2
question: 8
moi: 2
point: 2
cocktail: 2
lopel: 2
dans: 4
menton: 2
molotov: 2
bouteille: 2
song: 2
orh: 2
weekend: 4
```

Par la suite nous avons essayé d'extraire des sentiments en fonctions des paroles des musiques avec le NPL

Dans notre code on utilise donc la bibliothèque Stanford NLP pour créer un pipeline d'annotation qui tokenize, sépare en phrases, analyse la structure syntaxique (parse), et détermine le sentiment du texte en utilisant des modèles pré-entraînés. Cela permet d'extraire des informations sémantiques et d'évaluer l'orientation émotionnelle des paroles de chansons.

Nous n'avons bien évidemment pas créé un modèle nous même, le NPL propose des modèles dans toutes les langues (le français est disponible)

Ici par exemple nous avons essayé notre modèle sur une chanson RAP. Le sentiment retourné était neutre.

```
24/01/12 22:27:56 INFO sentiment.SentimentModel: Loading sentiment model edu/stanford/nlp/models/sentiment/sentiment.ser.gz ... done [0.2 sec].
Sentiment: Neutral

Process finished with exit code 0
```

Résultats

Nous avons réussi à utiliser de l'API Genius en Java pour extraire de manière systématique les données de paroles de chansons du site Genius. Les données collectées comprennent le nom de la chanson, les paroles, des informations sur l'artiste.

Ensuite nous avons réussi à transfert le fichier texte vers le cluster Hadoop, tirant parti de la capacité de stockage distribuée de Hadoop.

Les données collectées comprennent le nom de la chanson, les paroles, des informations sur l'artiste, et d'autres métadonnées pertinentes. Les résultats du projet sont les données scrapées sur Genius et stockées de manière distribuée sur un cluster Hadoop, contenant les paroles de chansons extraites du site Genius. De plus, des opérations de traitement ont été appliquées avec succès à l'aide d'Apache Spark, permettant d'extraire des informations pertinentes à partir des données collectées. On peut avec notre code présent dans le docker, prédire l'émotion générale d'une musique à partir uniquement des mots la composant

Difficultés rencontrées

Dans le cadre de notre projet, nous avons rencontré plusieurs difficultés, notamment lors de l'assimilation des technologies clés telles que Hadoop et Spark. Malgré les travaux pratiques qui ont été mis en place pour faciliter notre apprentissage, l'acquisition de compétences approfondies dans ces outils s'est avérée être un défi majeur.

La complexité inhérente à Hadoop et Spark, en raison de leurs fonctionnalités avancées et de leur architecture distribuée, a nécessité un investissement significatif en temps et en effort de la part de l'équipe. Les concepts fondamentaux liés au traitement des données massives et à la parallélisation des tâches ont demandé une compréhension approfondie, et même avec les travaux pratiques, certaines subtilités ont pu échapper à notre compréhension immédiate.

Par ailleurs, la gestion des versions compatibles a également posé des défis. Les technologies comme Hadoop et Spark sont constamment mises à jour, et il a été parfois difficile de maintenir la cohérence entre les différentes versions des outils et les dépendances associées. La compatibilité entre ces versions a nécessité une vigilance constante pour éviter des problèmes potentiels liés aux incompatibilités.

Malgré ces difficultés, nous avons adopté une approche proactive en sollicitant de l'aide auprès de ressources en ligne, en consultant la documentation. Ces expériences nous ont permis d'acquérir des compétences précieuses, mais il est important de noter que la courbe d'apprentissage a été plus prononcée que prévu initialement.

Perspectives d'amélioration

Pour renforcer notre projet, nous envisagerions diverses améliorations afin d'optimiser la récupération de chansons, d'intégrer des réponses plus nuancées et de proposer un affichage plus interactif. En ce qui concerne la récupération des chansons, l'idée serait d'explorer de nouvelles sources de données musicales pour élargir notre base. En parallèle, nous pourrions envisager de mettre en œuvre des algorithmes de recherche plus sophistiqués pour garantir une pertinence accrue dans les résultats.

Si nous voulions offrir des réponses plus nuancées, nous pourrions explorer l'intégration de modèles d'analyse musicale avancés. Ces modèles évalueraient les chansons selon plusieurs critères tels que le genre, l'émotion, la tonalité, etc. Cette approche permettrait de fournir des résultats avec des pourcentages de ressemblance, offrant ainsi une évaluation plus détaillée et personnalisée.

En ce qui concerne l'interface utilisateur, nous pourrions envisager de concevoir une expérience plus interactive. Cela impliquerait le développement d'une interface utilisateur permettant aux utilisateurs de filtrer et de trier les résultats en fonction de leurs préférences. L'intégration de fonctionnalités de prévisualisation audio directement dans l'interface permettrait également aux utilisateurs d'écouter des extraits de chansons avant de faire leur choix.

La personnalisation constituerait également un axe d'amélioration majeur. Nous pourrions envisager de mettre en place des fonctionnalités de personnalisation, telles que la création de profils d'utilisateurs, afin d'adapter les recommandations en fonction des préférences individuelles. En parallèle, un système de feedback utilisateur pourrait être implémenté pour ajuster les recommandations en temps réel, basé sur les retours des utilisateurs.

Enfin, pour garantir des performances optimales, nous pourrions chercher à améliorer la vitesse de traitement et de recherche. Cela pourrait impliquer l'exploration de technologies de traitement parallèle ou distribué pour accélérer le traitement des requêtes. En somme, ces améliorations viseraient à offrir une expérience utilisateur plus riche et personnalisée, tout en élargissant la portée et la qualité des résultats fournis.

Des perspectives d'amélioration pour ce projet pourraient inclure l'exploration de techniques avancées de traitement naturel du langage (NLP) pour analyser le contenu des paroles de chansons, ainsi que l'intégration de visualisations pour une meilleure compréhension des tendances ou des thèmes émergents dans les données collectées.

Ce projet constitue une étape importante dans l'utilisation des technologies Big Data pour extraire des informations significatives à partir de données non structurées, ouvrant ainsi la voie à des applications plus avancées dans le domaine de l'analyse de contenu musical.

Conclusion

Ce projet démontre la faisabilité et l'efficacité de l'utilisation de technologies Big Data telles que Hadoop et Apache Spark stocker et traiter des données de manière distribuée. Hadoop offre un écosystème riche avec de nombreux outils complémentaires, et Spark facilite l'optimisation des opérations complexes grâce à son API de transformation distribuée, ce qui simplifie le traitement de grandes quantités de données. Additionnement Spark peut utiliser Hadoop Distributed File System (HDFS) pour le stockage, ce qui signifie qu'il peut être intégré dans les environnements Hadoop existants. La combinaison de ces outils offre une approche robuste pour gérer des volumes importants de données non structurées, comme les paroles de chansons.

ANNEXE

Lien git:

https://github.com/ycao75/projetData

Les graphes menant à une classification par genre musical (en anglais):

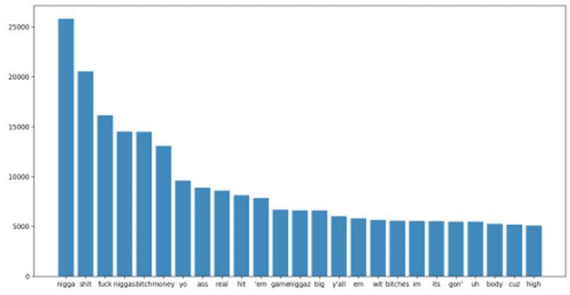


Figure 10: This plots the words and their respective counts in the hip hop genre that are not part of the top 300 words for pop or rock. While there were 97 total unique top words for hip hop, we chose to plot only 25 so that their words would be large enough on the bar graph.

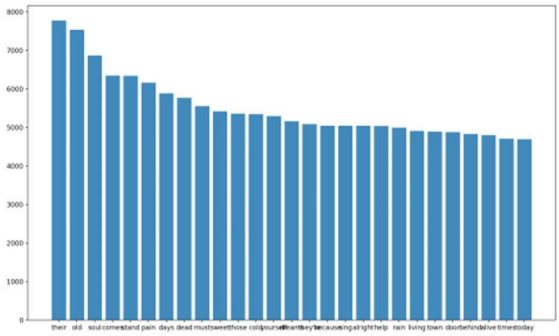


Figure 12: This plots the words and their respective counts in the rock genre that are not part of the top 300 words for pop or hip hop.

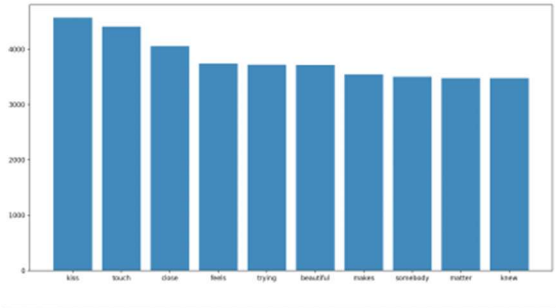


Figure 11: This plots the words and their respective counts in the pop genre that are not part of the top 300 words for hip hop or rock.

Source: Music Genre Classification using Song Lyrics Stanford CS224N Custom Project