Marco de referencia

Eduard Damiam Londoño Johany Londoño Toro
Jose Alberto Mejia

November 2019

1 Introducción

La minería de texto es una variante de la minería de datos con la cual se genera modelos y se identifica patrones de fuentes de datos cuya información esta en formato narrativo o texto libre [4]. La minería de texto permite una serie de aplicaciones como Búsqueda en texto, relevancia de documentos, entendimiento natural del lenguaje (NLP), entre otras aplicaciones. El objetivo de nuestro proyecto es dado un conjunto de noticias en texto libre, preprocesar los datos de las noticias para posteriormente explorar un método de analítica de texto, que nos ayude a encontrar información relevante en el conjunto de noticias.

2 Tecnologías

Actualmente dos de las tecnologías mas usadas para big-data y analítica son Hadoop y Apache Spark,Hadoop es un proyecto open-source desarrollado para almacenamiento y computación distribuida, de manera confiable y escalable [8],Hadoop permite el procesamiento distribuido de grandes volúmenes de datos en cluster usando modelos de programación simples,esta diseñado para escalar de una sola maquina,hasta a cientos de maquinas [3].

Spark es un sistema de cluster-computing rápido y de propósito general, para el procesamiento de largos volúmenes de datos, fue creado originalmente como una respuesta a las limitaciones de Hadoop MapReduce al leer y guardar resultados en disco, Spark es hasta 100 veces mas rápido que Map reduce, al trabajar sobre la memoria RAM y guardar en CACHE los resultados [1] Apache-Spark es bastante útil para la minería de texto ya ofrece una plataforma escalable para programación distribuida,[2].

Para trabajar en Spark es muy común usar PySpark, una API construida sobre Apache-Spark, PySpark extiende el "runtime" de spark para ejecutar sobre el programas escritos en python, algo muy util ya que python tiene librerías muy poderosas como pandas, numpy, scipy, etc. esto es posible ya que la JVM de spark se comunica con el interpretador de python a través de Py4J, el interpretador de python crea un objeto Spark-Context en la JVM y el Spark-Context orquesta la computación como lo haria con cualquier otro proceso de Spark. [5].

3 Técnicas de minado de texto

Un método para minar texto es el agrupamiento de texto, para agrupamiento de datos existen varios algoritmos, uno de los mas populares es K-means,una adaptación de este algoritmo para texto es TF-IDF, el peso TF-IDF es usado para saber que tan importante es una palabra por documento en una colección de documentos usando medidas estadísticas (la frecuencia de la palabra, y la frecuencia inversa del documento), usando este peso es posible agrupar documentos con palabras que aparezcan en pequeños grupos de documentos[7], este metodo es simple de implementar en pyspark.

Por otra parte otra manera de minar texto es a través de Detección de tópicos y Clasificación de textos, un método para esto es LDA (Latent Dirichlet Allocation) un modelo que infiere temas de una colección de datos no estructurados, dado un conjunto de documentos LDA intenta lograr lo siguiente[6] :

- identificar un conjunto de temas
- asociar conjuntos de palabras con un tema
- definir una mezcla de los temas para cada documento

References

[1] Amer Al-Badarneh. Join algorithms under apache spark: Revisited. In *Proceedings of the 2019 5th International Conference on Computer and Technology Applications*, ICCTA 2019, pages 56–62, New York, NY, USA, 2019. ACM.

[2] dzaratsian. Spark text analytics - uncovering data-driven topics. 2019.

[3] http://hadoop.apache.org. Apache hadoop.

[4] Pontificia Universidad Javeriana. Analítica de texto.

[5] Do Le Quoc, Franz Gregor, Jatinder Singh, and Christof Fetzer. Sgx-pyspark: Secure distributed data analytics. In *The World Wide Web Conference*, WWW '19, pages 3564–3563, New York, NY, USA, 2019. ACM.

[6] Girish Maskeri, Santonu Sarkar, and Kenneth Heafield. Mining business topics in source code using latent dirichlet allocation. In *Proceedings of the 1st India Software Engineering Conference*, ISEC '08, pages 113–120, New York, NY, USA, 2008. ACM.

[7] reza andriyunanto. Find most relevance text data using pyspark with tf-idf. 2018.

[8] Feng Wang, Jie Qiu, Jie Yang, Bo Dong, Xinhui Li, and Ying Li. Hadoop high availability through metadata replication. In *Proceedings of the First International Workshop on Cloud Data Management*, CloudDB '09, pages 37–44, New York, NY, USA, 2009. ACM.