BigTexts: Framework de pre-procesamiento de datos en minería de texto basado en tecnologías de Big Data

Ing. Wilson Alzate Calderón
Email: walzate@javeriana.edu.co[1] and
Ing. Alexandra Pomares Ph.D.
Email: pomares@javeriana.edu.co[2]

[1,2]Departamento de Ingeniería de Sistemas, Pontificia Universidad Javeriana

***Abstract*—Into the Health Care industry, the Electronic Medical (Health) Records (EMRs or EHRs) management can be classified as a Big Data problem because they have the three fundamental characteristics of that paradigm: Volume, Variety and Velocity. A big inconvenient is that the bigest part of the information (the most relevant) is found into the narrative text fields, having the Text-Mining as the key technology for knowledge discovering in that kind of problems. Within the tasks performed into a Text-Mining project, the more critical in terms of performance (time consuming) are the pre-processing ones, having there a big opportunity for optimizing processes using Big Data technologies in order to help speeding up the response times. In the present article is presented BigTexts: a framework that offers pre-built functions for the execution of pre-processing tasks in Text-Mining based on Big Data technologies.**

En la industria de la salud la gestión de historias clínicas electrónicas se puede clasificar como un problema de Big Data ya que cuenta con las tres características fundamentales: Volumen (gran cantidad de exámenes, notas, tratamientos, diagnósticos etc.), Variedad (datos estructurados y no estructurados) y Velocidad (necesidad de grandes velocidades de procesamiento al realizar ciertos análisis). Un gran inconveniente que se tiene es que la mayor parte de la información se encuentra en los campos de texto narrativo, siendo la minería de texto una tecnología clave para el descubrimiento de conocimiento. Dentro de las tareas que se realizan en los proyectos de minería de texto las más críticas en términos de consumo de tiempo son las de pre-procesamiento y de análisis, por lo que se encuentra allí una gran oportunidad de optimización de procesos utilizando tecnologías de Big Data que permitan agilizar los tiempos de respuesta. En el presente artículo se presenta BigTexts: un framework que ofrece funcionalidades pre-construidas para la ejecución de tareas de pre-procesamiento en minería de texto basado en tecnologías de Big Data.

I. Introducción

En la actualidad, el monto de información almacenada en el mundo se estima alrededor de 1.200 exabytes(10^{18} bytes) [12], los cuales son producidos por múltiples fuentes; por ejemplo, Google procesa más de 24 petabytes(10^{15} bytes) de datos por día y Facebook obtiene más de 10 millones de fotos cada hora [12]. En el mundo de los negocios el 80% de los datos existen en un formato no estructurado [16] y si este monto fuera solamente libros impresos, ellos cubrirían la superficie entera de Estados Unidos en 52 capas de grueso [12]. Se estima que en algún momento entre 2003 y 2004, el monto de datos creados por el mundo digital aceleró exponencialmente, sobrepasando el monto total de datos creado en los 40.000 años previos de la civilización humana [19].

Teniendo en cuenta estas grandes cantidades de datos que se están generando actualmente, surge el paradigma de Big Data, que es un concepto generalmente usado para describir montos vastos de datos diversos, tanto estructurados como no estructurados, a los cuales las organizaciones pueden acceder de manera rápida, para analizarlos usando herramientas innovadoras, que en conjunto, ayudan a determinar con precisión oportunidades de mejora en la gestión y generación de valor [15]. Un problema de Big Data es aquel que cumple con las 3 Vs: Volumen (Escala de los datos), Variedad (Diferentes formas de datos) y Velocidad (Análisis de flujo de datos)[13]. Dado que Big Data no es como tal una tecnología, sino más bien un paradigma que debe ser atacado por medio de un conjunto de soluciones(Tabla I). Se cuenta con diversas aproximaciones que pueden ayudar a abordar las propiedades de un problema de este tipo, como lo son Cloud Computing [18], Analytics [26], MapReduce [28], NoSQL [27], Apache Hive [3], Apache Pig [8] o Hadoop [1].

Como en diversos sectores como el automotriz, el transporte, el industrial, el de servicios y el comercial, en el cuidado de la salud, se están generando grandes cantidades de datos rutinarios, los cuales pueden ser minados e incluso combinados con tweets y blogs. Es un gran reto procesar, analizar y conservar dicha masa de datos. Darle sentido a esa gran cantidad de información ofrece oportunidades para el mejor tratamiento de una enfermedad, abordar temas de salud pública o para el funcionamiento eficiente de los proveedores de servicios de salud [10]. El análisis de datos contenidos en las historias clínicas electrónicas usando técnicas computacionales es un problema de Big Data [16] debido al gran volumen de información contenida, a la variedad entre datos estructurados y no estructurados, así como también, la velocidad con la que se requieren ciertos análisis. Las técnicas de minería de texto son las más indicadas para el análisis del contenido narrativo de las historias clínicas electrónicas. Algunas de las tareas relacionadas a la minería de texto son el pre-

Tecnología	Sub-tecnología	Velocidad	Variedad	Volumen
NoSQL			X	X
Cloud computing	Infraestructure as a Service (IaaS)	X		X
	Software as a Service (SaaS)	X		X
	Platform as a Service (PaaS)	X		X
	Analytics as a Service (AaaS)	X	X	X
Analytics	Exploratory Data Analysis (EDA)		X	X
	Confirmatory Data Analysis (CDA)		X	X
	Qualitative Data Analysis (QDA)	X	X	X
MapReduce		X		X
Apache Hive		X		X
Apache Pig		X		X

Table I
CLASIFICACIÓN DE TECNOLOGÍAS ASOCIADAS A BIG DATA

procesamiento de colecciones de documentos (categorización de texto, extracción de información, extracción de términos), el almacenamiento de representaciones intermedias, técnicas para analizar dichas representaciones intermedias (como análisis de distribución, clustering, análisis de tendencias y reglas de asociación), así como la visualización de los resultados [6]. Las operaciones de pre-procesamiento se centran en la identificación y extracción de características representativas para documentos en lenguaje natural y son responsables de la transformación de datos no estructurados almacenados en colecciones de documentos a un formato intermedio explícitamente estructurado [6]. Siendo tanto las tareas de pre-procesamiento como las del núcleo, las dos áreas más críticas para cualquier sistema de minería [6] por lo que se encuentra allí una gran oportunidad de optimización de procesos utilizando tecnologías de Big Data que permitan agilizar los tiempos de respuesta de los sistemas de minería.

Teniendo en cuenta la problemática antes descrita se llega a la pregunta de ¿Cómo generar un framework que ofrezca funcionalidades pre-construidas para ejecutar tareas de pre-procesamiento en minería de texto basado en tecnologías de Big Data, en donde se logre adicionar un nivel de abstracción a la complejidad que implica tener que enlazar las diferentes tecnologías disponibles y así poder mejorar los tiempos de procesamiento en sistemas de minería de texto?, para lo cual en este proyecto se propone generar un modelo de aplicación de Big Data que soporte la fase de pre-procesamiento en los proyectos de minería de texto a través de un framework, el cual será validado en un caso de estudio basado en análisis de historias clínicas electrónicas, específicamente en el análisis de notas de ingreso, notas de enfermería y plan de tratamiento.

En las secciones subsecuentes se presentará inicialmente una descripción de los trabajos relacionados(Sección II), inicialmente a Big Data aplicado a la salud(Sección II-A) y posteriormente a Big Data para la gestión de datos textuales(Sección II-B). A continuación se presenta el framework BigTexts(Sección III), especificando sus características(Sección III-A) y arquitectura(Sección III-B). Posteriormente se expone la validación del framework en el contexto de aplicación(Sección IV), especificando el escenario(Sección IV-A) y los resultados obtenidos(Sección IV-B). Finalmente se presentan las conclusiones y los trabajos futuros(Sección V).

II. TRABAJOS RELACIONADOS

A. BigData en Salud

En la tabla II se presenta un resúmen de los principales artículos relacionados a la aplicación de las tecnologías relacionadas al paradigma de Big Data en el dominio de la salud, encontrando que en su mayoría los artículos son propuestas teóricas de usos deseables de las tecnologías de Big Data en dicho dominio. De igual manera son pocas las aproximaciones prácticas y palpables del uso de este paradigma en productos o prototipos que realmente hagan realidad los beneficios esperados. También se pudo evidenciar que uno de los factores recurrentes en los artículos, y por lo cual se habla de la necesidad del uso del paradigma Big Data en salud, es la alta cantidad de datos que se requiere analizar para que el personal médico y administrativo pueda tomar las mejores decisiones, entendiendo que una de las mayores fuentes de información es el texto narrativo consignado en las historias clínicas electrónicas. Por lo anterior vale la pena revisar los trabajos realizados con el paradigma en cuanto a la gestión de datos textuales (Sección II-B).

B. BigData para gestión de datos textuales

En cuanto al análisis de tecnologías para datos textules, se revisarán a continuación un par de casos de estudio, uno para análisis de datos no estructurados (Sección II-B1) y otro relacionado al procesamiento a gran escala en la nube (Sección II-B2)

1) Big Data para análisis de datos no estructurados: Se propone la utilización de Big Data unido a minería de texto para trabajar con contenido narrativo en forma de tweets públicos, usando tecnologías como HBase, Java, REST y Hadoop [5]. Dado que es un proyecto en progreso, solamente se ha completado la primera fase, en donde se toman los tweets y los almacenan en la base de datos HBase, pero no se realiza minería de texto aún.

2) GATECloud.net: una plataforma de código abierto para el procesamiento de texto a gran escala en la nube: Se mencionan varias estrategias para mejorar los tiempos de respuesta en procesamiento de lenguaje natural, entre las cuales se tenían dos opciones: el uso de MapReduce y el manejo de tecnologías de Cloud Computing, siendo para ellos mejor la segunda, bajo un modelo PaaS (Platform as a Service), ya que no es necesario reescribir los algoritmos de procesamiento

Artículo	Tecnología	Problemática	Tipo	Contenido
Uso de Big Data y Cloud Computing para BI operacional en Salud [18]	Cloud Computing	Superar la brecha digital para los Sistemas de Información Médica en los países en desarrollo	Práctico	Texto
Big Data en cuidado de la salud personalizado [4]	Minería de datos	Perfil de riesgo de enfermedades personalizado	Teórico-Práctico	Texto
Big Data en Ecocardiografía Funcional [19]	Cloud computing, robots, Inteligencia Artificial	Poder contar con más variables en la ecocardiografía funcional	Teórico	Imágenes
Adquisición, compresión, encripción y almacenamiento de Big Data en Salud [2]	Procesamiento de archivos	Compresión de datos obtenidos en registros electrofisiológicos	Práctico	Imágenes
Big Data en dispositivos médicos [14]	Sensores	Confiabilidad y disponibilidad de dispositivos médicos	Teórico	Texto
Desarrollo de un ecosistema de cuidado de la salud usando IPHIs [10]	Ontologías	Ecosistema de interoperabilidad en salud para sistemas heterogéneos	Teórico	Texto
Big Data para evaluar el impacto de programas médicos [7]	Analytics	Destinar y retener pacientes reales	Teórico	Texto

Table II
COMPARACIÓN DE ARTÍCULOS DE BIG DATA EN SALUD

de texto que se van a ejecutar. El nombre de la solución es GATECloud.net, que es una plataforma Web en la que los científicos pueden realizar sus experimentos de minería de texto, cuyos servicios son cobrados mediante el modelo pago por uso y donde no se proponen librerías (APIs) para el empleo de la herramienta desde otros sistemas [20].

Como se evidencia, aunque existen aproximaciones que tratan de unir el paradigma de Big Data a la minería de texto, aún no son lo suficientemente maduras o no ofrecen APIs por medio de las cuales sistemas existentes puedan ejecutar tareas (como por ejemplo de pre-procesamiento) en grandes cantidades de texto, lo que brinda una gran oportunidad que se intenta atacar mediante BigTexts (Sección III).

III. BIGTEXTS

BigTexts es un framework para pre-procesamiento de datos en minería de texto basado en tecnologías de Big Data, el cual ofrece funcionalidades pre-construidas que pueden ser usadas mediante la ejecución de una aplicación independiente (con una interfaz gráfica de usuario - GUI) o como librería (API) de una aplicación existente.

A. Qué hace

Como se muestra en la figura 1, en la situación actual se cuenta con un conjunto de documentos, a los cuales le es aplicada una serie de tareas de pre-procesamiento de manera secuencial, para obtener una colección de documentos pre-procesados. La situación deseada, apoyada por BigTexts, toma los documentos iniciales, los particiona en el cluster Hadoop mediante HDFS, y usando MapReduce, ejecuta las tareas de pre-procesamiento en paralelo, llegando finalmente a la colección de documentos pre-procesados.

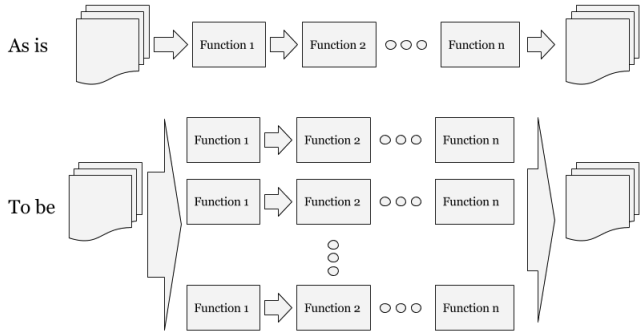

Figure 1. Situación actual vs situación deseada

BigTexts ofrece las siguientes funcionalidades de pre-procesamiento en minería de texto, las cuales fueron construidas usando mayoritariamente la librería Stanford CoreNLP[22], la cual provee las funcionalidades de Tokenization, Part Of Speech, Stemming, Named Entity Recognition, Correference Recognition, etc. Adicional a ello se usa la librería de Weka[25] para el Lovins Stemmer y libstemmer[17] para la implementación del Snowball Stemmer. Dichas funcionalidades son encapsuladas en UDFs de Apache Pig:

- Identificación de correferencias[1] en inglés
- Lematización[2] usando el algoritmo Lovins Stemmer para inglés
- Named Entity Recognition[3] en inglés
- Named Entity Recognition en español

[1]Es la tarea de encontrar todas las expresiones que se refieren a la misma entidad en un texto [21]

[2]Reducir una palabra a su lema o raíz[9]

[3]Etiquetar palabras en un texto como verbos, adjetivos, sustantivos, etc. [24]

- Part Of Speech[4] en inglés
- Part Of Speech en español
- Named Entity Recognition basado en expresiones regulares
- Lematización usando el algoritmo Snowball Stemmer para inglés
- Lematización usando el algoritmo Snowball Stemmer para español
- Partición de textos por frases
- Tokenización
- Paso a mayúsculas de un texto

La adición de una nueva tarea de pre-procesamiento ha sido diseñada para ser un proceso sencillo y flexible. Para ello simplemente se crea una nueva clase en la cual se implementa la funcionalidad y se adiciona al catálogo de tareas de pre-procesamiento[5].

Los siguientes casos de uso describen las funcionalidades de BigTexts:

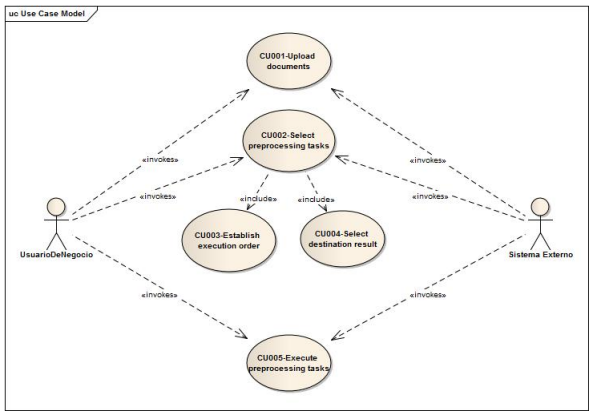

Figure 2. Vista de Escenarios

1) CU001-Upload documents: Describe el proceso de realizar el cargue de documentos para su posterior procesamiento
2) CU002-Select preprocessing tasks: Describe la selección de las tareas de pre-procesamiento que se desean realizar en el sistema. Las tareas que se podrán seleccionar son Part Of Speech, Splitter, Tokenizer, Entity Recognition, búsqueda basado en expresiones regulares(NE Transducer) e identificación de coreferencias
3) CU003-Establish execution order: Se establece el orden en que se deben ejecutar las tareas seleccionadas y configuradas en el CU002.
4) CU004-Select destination result: El sistema elige la forma en que quiere que le sea entregado el resultado, teniendo como opciones: una ubicación en un directorio FTP, en el sistema de archivos local, o en un directorio HDFS.
5) CU005-Execute preprocessing tasks: El sistema ejecuta la lista de tareas de pre-procesamiento sobre los documentos cargados.

B. Cómo lo hace

BigTexts es una aplicación que está desarrollada en lenguaje Java, que en términos generales, como se muestra en la Figura 3, está compuesto por dos artefactos, un cliente (Cliente BigTexts) y un servidor (BigTexts).

El cliente ofrece una interfaz gráfica por medio de la cual el usuario puede seleccionar los archivos a procesar, las tareas de pre-procesamiento a ejecutar (con sus respectivas parametrizaciones) y la forma de entrega[6]. Dicha aplicación cliente también puede ser usada por una aplicación externa como librería. El cliente se encarga de enviarle los archivos al servidor mediante un FTP y los llamados para las ejecuciones se realizan de manera asíncrona, enviando mensajes en formato XML a un servidor de colas.

El servidor BigTexts se conecta al servidor de colas (ActiveMQ) y obtiene el mensaje encolado, transformándolo de XML (usando la librería JAXB[7]) a objetos. Posteriormente el mensaje convertido en objetos es ejecutado en el clúster Hadoop[8] mediante la construcción de un script de Apache Pig[9] que usa UDFs[10] con las tareas de pre-procesamiento. El clúster Hadoop está compuesto por una máquina principal (el master), la cual se encarga de gestionar tanto el almacenamiento como el processamiento en paralelo, y una serie de máquinas secundarias (slaves) que son las que almacenan los bloques de datos y los procesan de manera paralela. Finalmente se entrega un conjunto de documentos pre-procesados en el HDFS o en el directorio FTP, según lo haya seleccionado el usuario.

Se usa además una base de datos PostgreSQL[11] para el almacenamiento de la siguiente información de auditoría relacionada a las ejecuciones:

- Fecha y hora de inicio de procesamiento
- Fecha y hora final de procesamiento

[4]Etiquetar palabras en un texto como personas, nombres de compañías, géneros, paises, ciudades, etc.[23]

[5]Un archivo XML con todas las tareas de preprocesamiento y los valores de parametrización por defecto.

[6]La forma en la que el cliente requiere que se le entreguen los documentos resultado del proceso, que puede ser en un directorio del HDFS o en el servidor FTP.

[7]Java Architecture for XML Binding

[8]*Framework* que permite el procesamiento distribuido de conjuntos grandes de datos a través de clusters de computadores usando modelos de programación simples. Está diseñado para ir escalando de un solo servidor a miles de máquinas, cada una ofreciendo capacidad de cálculo y almacenamiento. En lugar de confiar en el Hardware para entregar alta disponibilidad, la librería por sí misma está diseñada para detectar y manejar fallos en la capa de aplicación, entregando así un servicio altamente disponible por encima de un clúster de computadores, cada uno de los cuales puede ser propenso a fallos [1].

[9]Motor para la ejecución de flujos de datos en paralelo sobre Hadoop y un lenguaje llamado Pig Latin para la expresión de dichos flujos de datos. Dicho lenguaje incluye operadores para muchas de las operaciones de datos tradicionales (join, sort, filter, etc.), así como también ofrece la posibilidad de que los usuarios puedan desarrollar sus propias funciones para lectura, procesamiento y escritura de datos (User Defined Functions - UDFs). Dado que pig corre sobre Hadoop, hace uso directo de HDFS (Hadoop Distributed File System - sistema de ficheros distribuido que almacena los archivos a través de todos los nodos en un cluster Hadoop) Y MapReduce (El sistema de procesamiento de Hadoop) [8]

[10]User Defined Functions, La forma en que Apache Pig permite la creación de funciones personalizadas en Java.

[11]Se realiza la conexión a la base de datos mediante JPA (Java Persistence API)

- Fecha y hora de carga de archivos
- Nombre y peso de archivos pre-procesados
- Las tareas de pre-procesamiento parametrizadas y
- El número de máquinas secundarias disponibles

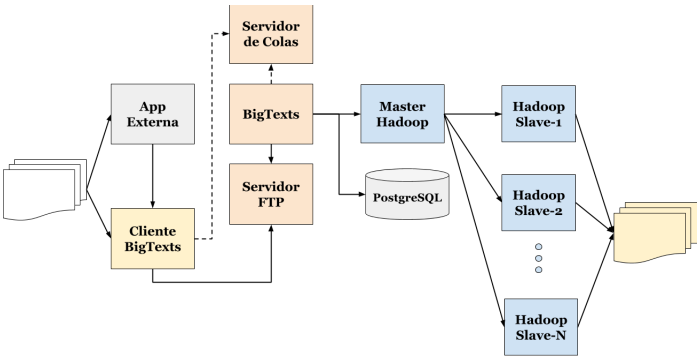

Figure 3. BigTexts

IV. Validación

A continuación se presenta tanto el escenario en el cual se realizó la validación del sistema (Sección IV-A), como también los resultados de la misma (Sección IV-B).

A. Escenario de validación

Las máquinas provistas por el programa de maestría para realizar la configuración del clúster Hadoop (Cuadro III), no son equipos con especificaciones altas (tipo servidor) sino más bien equipos de uso común. Se contó con una máquina principal (master) y cuatro máquinas secundarias (slaves). La aplicación cliente se instaló en la máquina master, así como el servidor BigTexts. Todas los computadores se instalaron con el sistema operativo Ubuntu 14.0.LTS. Las máquinas se conectaron en una red de área local (LAN) estableciendo direcciones IP estáticas. Se contó con 4 máquinas de marca HP y una Lenovo. La máquina master, que fué en donde se instaló el servidor de colas (ActiveMQ), el servidor FTP y las dos aplicaciones de BigTexts (el cliente y el servidor). El computador master contó con el doble de memoria RAM que el resto de equipos. Todas las máquinas contaban con procesadores Intel, 4 de ellas con Core 2 Duo y una con Core 2.

Se tomaron 4 archivos con registros de historias clínicas electrónicas del Hospital Universitario San Ignacio con los siguientes tamaños:

- 1.945.886 bytes (1.945 Mb)
- 1.423.803 bytes (1.423 Mb)
- 437.643 bytes (437.643 Kb) y
- 761 bytes

Realizando 5 iteraciones para cada archivo en cada una de las siguientes tareas de pre-procesamiento:

- RegexNamedRecognition: Dada una lista de tokens con una etiqueta para cada una de ellas (medicamentos), el sistema identifica usando expresiones regulares si una palabra se encuentra en dicha lista y le establece la marca de MEDICAMENTO.
- Tokenizer: Partición de archivos en tokens.
- Tokenizer + POS-Tagger: Se realizan dos tareas de pre-procesamiento: Partición del archivo en tokens y luego (Part of Speech) identificando si cada uno de ellos es un verbo, un adjetivo, un adverbio, etc.
- Tokenizer + SnowballStemmer: Igualmente se realiza el particionado del archivo en tokens y luego se realiza la identificación de raices (Stemming) de cada uno de ellos.

B. Resultado de validación

A continuación se muestra el análisis de los datos obtenidos, usando figuras de líneas para identificar tendencias. Para la correcta visualización en la gráfica se dividió el tamaño del archivo por 10.000 y así tener dichos datos en la misma escala del tiempo.

Encontrando por ejemplo para la tarea de Regex Entity Recognition (Figura 4) que con el archivo de 1.945.886 bytes que para una máquina el tiempo promedio para las iteraciones fue de 109,13 segundos, para tres máquinas de 92.78 segundos y para cinco de 82.25 segundos. Para el archivo de 1.423.803 bytes los tiempos fueron de 104.07, 98.33, 80.45 segundos. Así mismo, el archivo de 437.643 bytes tuvo tiempos de 70.54, 65.13, 58.71 y el de 761 bytes tiempos de 61.57, 54 y 51 segundos para una, tres y cinco máquinas activas en el clúster. Se muestra entonces una tendencia por archivo a la baja en tiempos al aumentar el número de máquinas disponibles en el cluster.

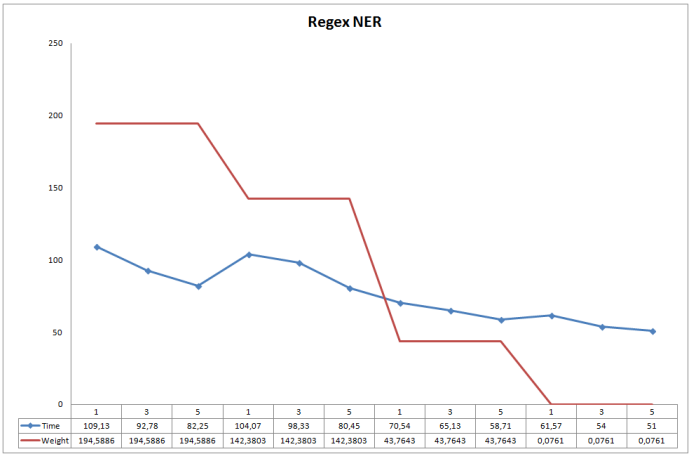

Figure 4. Validación - RegexNER

Para la tarea de Tokenización (Figura 5) para el archivo de 1.945.886 bytes se tuvieron tiempos de 50,22, 46.63, 36 segundos. Con el archivo de 1.423.803 los promedios de tiempos fueron de 56.25, 41.33 y 33.6 segundos. De la misma manera, para los archivos de 437.643 y 761 bytes se tuvo una mejora de tiempos de ejecución al aumentar el número de máquinas disponibles en el clúster.

Nombre	Hostname	IP Interna	Marca	Memoria	Disco	Procesador
bigtexts-1	master	192.168.0.101	HP	3,8 GiB	155,3 GB	Intel Core 2 Duo CPU E7200 2.53GHz x 2
bigtexts-2	slave-2	192.168.0.102	HP	1.9 GiB	155.3 GB	Intel Core 2 Duo CPU E7200 2.53GHz x 2
bigtexts-3	slave-3	192.168.0.103	Lenovo	1.9 GiB	76.5 GB	Intel Core 2 CPU 4400 2.00GHz x 2
bigtexts-4	slave-4	192.168.0.104	HP	1.9 GiB	155.3 GB	Intel Core 2 Duo CPU E4600 2.40GHz x 2
bigtexts-5	slave-5	192.168.0.105	HP	1.9 GiB	155.3 GB	Intel Core 2 Duo CPU E7200 2.53GHz x 2

Table III
MÁQUINAS DEL CLÚSTER

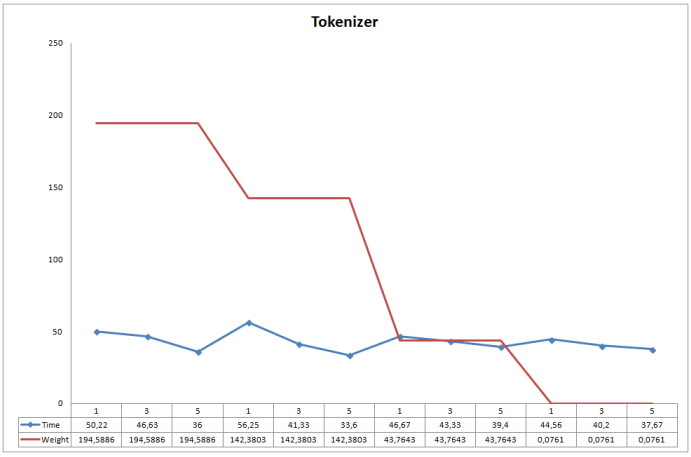

Figure 5. Validación - Tokenizer

En cuanto a la tarea de Tokenización sumada a la de Part of Speech (Figura 6) para los archivos de 1.945.886 bytes (181, 113.75 y 85.67 segundos), 1.423.803 bytes (118.33, 96.67 y 76.33 segundos) y 761 bytes (69.67, 65 y 52.5) segundos se presentó mejoría en los tiempos al aumentar el número de máquinas disponibles en el clúster. Sin embargo, para el archivo de 437.643 bytes se presentó una desmejoría entre usar una y tres máquinas (tiempos de 80 y 87 segundos respectivamente) pero una mejora sustancial al usar finalmente las cinco máquinas con un tiempo de 54 segundos.

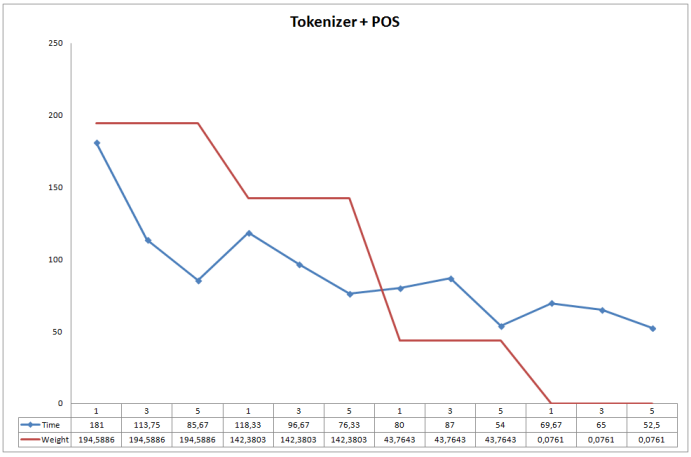

Figure 6. Validación - Tokenizer + POS

Para la tarea de Tokenización sumada a la de Stemming usando el algoritmo Snowball (Figura 7) se obtuvieron los siguientes tiempos para una, tres y cinco máquinas respectivamente:

- 1.945.886 bytes: 69.15, 53.86 y 49 segundos
- 1.423.803 bytes: 47.05, 45.4 y 43.71 segundos
- 437.643 bytes: 46.46, 43.6 y 40.67 segundos
- 761 bytes: 50.21, 47.42 y 42.25 segundos

Pudiéndose identificar una leve tendencia a la baja a la medida que se fueron adicionando máquinas disponibles al clúster.

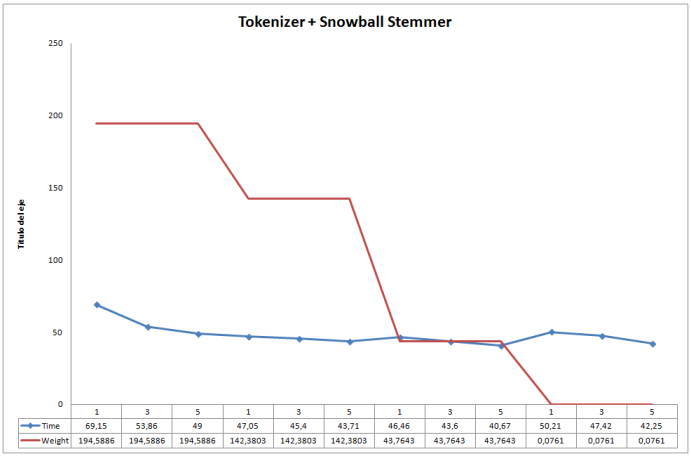

Figure 7. Validación - Tokenizer + Snowball

Como puede observarse los tiempos para un mismo archivo, en una misma tarea, difieren dependiendo del número de esclavos disponibles, mejorando a medida que se iban adicionando nodos esclavo. También se puede observar que la mejoría es mucho más notoria cuando se trabaja con tareas complejas como el RegexNER y Part Of Speech. Adicionalmente, se puede ver que entre más grande es el archivo, mayor es la velocidad de procesamiento, ya que por ejemplo para el de 761 bytes los tiempos para los diferentes escenarios (1,3 y 5 máquinas) es muy similar.

V. Conclusiones y Trabajos Futuros

Con la realización del proyecto se pudo observar que la adición de una capa de abstracción a todo el ecosistema de tecnologías de Big Data y Minería de Texto es un aporte

muy valioso, ya que la instalación, configuración e integración de los componentes es bastante dispendiosa. Esto se debe a que por ejemplo, las tecnologías de Big Data, aunque son poderosas, también son precarias en documentación precisa y confiable de instalación y configuración.

En cuanto a los resultados de la validación se pudo entender que vale la pena usar toda la infraestructura de Big Data siempre y cuando se tengan archivos lo suficientemente grandes o tareas lo suficientemente complejas, ya que se comprueba que para archivos grandes o tareas complejas, la adición de nodos en el cluster mejora notoriamente el rendimiento, pero que para archivos pequeños, por ejemplo, los tiempos son bastante similares a los obtenidos con una sola máquina.

Como trabajo futuro se propone usar Cloud Computing para abordar el problema planteado en el presente proyecto, con el fin de evaluar el contraste de esfuerzo de desarrollo, uso de nuevas tecnologías y escalabilidad de un enfoque como ese, comparado con el usado en BigTexts. Adicionalmente, otro campo de trabajo es el desarrollo de más UDFs para incrementar el catálogo de tareas de pre-procesamiento de BigTexts. Se propone también la instalación de un cluster Hadoop en la infraestructura del centro de computación de alto desempeño de la Pontificia Universidad Javeriana (ZINE)[12] y realizar pruebas con muchos más nodos.

REFERENCES

[1] Apache Software Foundation. Hadoop. http://hadoop.apache.org/.

[2] Benjamin H. Brinkmann, Mark R. Bower, Keith A. Stengel, Gregory A. Worrell, and Matt Stead. Large-scale electrophysiology: Acquisition, compression, encryption, and storage of big data. *Journal of Neuroscience Methods*, 180(1):185–192, May 2009.

[3] Edward Capriolo, Dean Wampler, and Jason Rutherglen. *Programming Hive*. O'Reilly Media, 1 edition edition, September 2012.

[4] Nitesh V. Chawla and Darcy A. Davis. Bringing big data to personalized healthcare: A patient-centered framework. 28:660–665.

[5] T.K. Das and P Mohan Kumar. Big data analytics: A framework for unstructured data analysis. *School of Information Technology and Engineering, VIT University*.

[6] Ronen Feldman and James Sanger. *The Text Mining Handbook*. Cambridge University Press, 1 edition edition, March 2013.

[7] Bill Fox. Using big data for big impact. leveraging data and analytics provides the foundation for rethinking how to impact patient behavior. *Health management technology*, 32(11):16, November 2011. PMID: 22141243.

[8] Alan Gates. *Programming Pig*. O'Reilly Media, 1 edition edition, September 2011.

[9] Lancaster University. What is stemming? http://www.comp.lancs.ac.uk/computing/research/stemming/general/.

[10] Harshana Liyanage, Siaw-Teng Liaw, and Simon de Lusignan. Accelerating the development of an information ecosystem in health care, by stimulating the growth of safe intermediate processing of health information (IPHI). *Informatics in primary care*, 20(2):81–86, 2012. PMID: 23710772.

[11] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, 2014.

[12] Viktor Mayer-Schonberger and Kenneth Cukier. *Big data: a revolution that will transform how we live, work, and think*. Houghton Mifflin Harcourt.

[13] Andrew McAfee and Erik Brynjolfsson. Big data: The management revolution. *Harvard business review*, 90(10):p60–68.

[14] William Q. Meeker and Yili Hong. Reliability meets big data: Opportunities and challenges. *Quality Engineering*, 26(1):102–116, January 2014.

[15] Keith D Moore, Katherine Eyestone, and Dean C Coddington. The big deal about big data. *Healthcare financial management: journal of the Healthcare Financial Management Association*, 67(8):60–66, 68, August 2013. PMID: 23957187.

[16] Travis B. Murdoch. The inevitable application of big data to health care. 309(13):1351.

[17] Martin Porter. Snowball. http://snowball.tartarus.org/.

[18] Saptarshi Purkayastha and Jørn Braa. Big data analytics for developing countries – using the cloud for operational BI in health. *The Electronic Journal of Information Systems in Developing Countries*, 59(0), October 2013.

[19] Partho P. Sengupta. Intelligent platforms for disease assessment. *JACC: Cardiovascular Imaging*, 6(11):1206–1211, November 2013.

[20] V. Tablan, I. Roberts, H. Cunningham, and K. Bontcheva. GATECloud.net: a platform for large-scale, open-source text processing on the cloud. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 371(1983):20120071–20120071, December 2012.

[21] The Stanford NLP Group. Coreference resolution. http://nlp.stanford.edu/projects/coref.shtml.

[22] The Stanford NLP Group. Stanford CoreNLP. http://nlp.stanford.edu/software/corenlp.shtml.

[23] The Stanford NLP Group. Stanford log-linear part-of-speech tagger. http://nlp.stanford.edu/software/tagger.shtml.

[24] The Stanford NLP Group. Stanford named entity recognizer (NER). http://nlp.stanford.edu/software/CRF-NER.shtml.

[25] The University of Waikato. Weka 3 - data mining with open source machine learning software in java. http://www.cs.waikato.ac.nz/ml/weka/.

[26] WhatIs.com. What is data analytics (DA)? http://searchdatamanagement.techtarget.com/definition/data-analytics.

[27] WhatIs.com. What is NoSQL (not only SQL)? http://searchdatamanagement.techtarget.com/definition/NoSQL-Not-Only-SQL.

[28] Paul Zikopoulos and Chris Eaton. *Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data*. McGraw-Hill Osborne Media, 1 edition edition, October 2011.

[12] Al inicio del presente proyecto se realizó el contacto para usar dicha infraestructura, lo cual fué negado porque era un ambiente de producción.