ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL
CENTRO DE INVESTIGACIÓN CIENTÍFICA Y TECNOLÓGICA

Implementación y evaluación de un detector masivo de Web Spam

Jesús González[1], Washington Bastidas[2], Cristina Abad[3]
Facultad de Ingeniería en Electricidad y Computación
Escuela Superior Politécnica del Litoral
Campus Gustavo Galindo, Km. 30.5 Vía Perimetral
Apartado 09-01-5863, Guayaquil-Ecuador
jesus.gonzalezvera@gmail.com[1], wbastidas@fiec.espol.edu.ec[2], cabad@fiec.espol.edu.ec[3]

Resumen

Este trabajo presenta un mecanismo para detectar Web Spam de forma masiva, utilizando una arquitectura distribuida basada en el paradigma MapReduce para el procesamiento paralelo y las Máquinas de Vectores de Apoyo (SVM, por sus siglas en inglés) como algoritmo de aprendizaje para la clasificación. El Web Spam que es, la asignación injustificada de relevancia a páginas en la Web, se ha convertido en un tema muy abordado hoy en día, dado que las partes involucradas, las Máquinas de Búsqueda por un lado y por otro los usuarios que demandan información de ellas, pueden verse beneficiadas o perjudicadas según sea el tratamiento a dicho problema. Nuestra solución presenta una alternativa para detectar páginas Web Spam que combina el modelo de programación distribuida de MapReduce, implementado con Hadoop, con un modelo de SVM en cascada utilizando los servicios Web de Amazon que, brindan una forma muy práctica y no costosa de realizar el cómputo de grandes cantidades de información en la nube.

Palabras Claves: *MapReduce, Máquinas de Vectores de Apoyo, Web Spam, Computación en Nube.*

Abstract

This work presents a mechanism to detect Web Spam in a massive way, using a distributed architecture based on the paradigm MapReduce for the parallel processing and the Support Vectors Machines (SVM) as learning algorithm for the classification. The Web Spam that is, the unjustified assignment of relevance to pages in the Web, has become a topic very approached actually since the involved parts, the Searching Machines on one hand and for other the users that demand information of them, can be benefited or harmed by the treatment of this issue. Our solution presents an alternative to detect Web Spam pages that combine the programming pattern MapReduce, implemented with Hadoop, with a cascade model of SVM using the Amazon web services that, offer a very practical and not expensive form to carry out the computation of big quantities of information in the cloud.

Keywords: *MapReduce, Support Vectors Machine, Web Spam, Cloud Computing.*

1. Introducción

En el mundo actualmente globalizado el acceso, recuperación y reutilización de la información es una prioridad fundamental en el diario vivir de las personas. El uso de las máquinas de búsqueda para realizar dichas tareas han sido esenciales debido al crecimiento de la información existente en la Web; pero, a causa de las falencias existentes en dichas máquinas algunos internautas han logrado beneficiarse de los medios existentes, manipulando la información maliciosamente.

El mecanismo comúnmente usado para realizar esta forma ilícita de manipulación es el Web spam [4], que es simplemente la asignación injustificable de relevancia a una página produciendo resultados inesperados en las máquinas de búsqueda, deteriorando así la calidad de las consultas y su veracidad.

Como agravante a esta situación, tenemos que cada vez que se encuentra una solución parcial al problema de Web spam, los spammers [8] se encargan de buscar otra forma de eludir este mecanismo, lo que hace de esto una lucha interminable. Aunque no es muy difícil para una persona experta detectar un Web spam, la meta es poder suplir este tipo de métodos, debido a la gran cantidad de páginas existentes, con una solución automatizada que aprenda rápidamente de los nuevos métodos o formas de ataque.

2. Web Spam

El término Web spamming [4] es definido como cualquier acción humana deliberada que permite dar a lugar una relevancia injustificable de valor e importancia de algunas páginas.

La Web contiene muchas personas que buscan la forma de beneficiarse de los millones de usuarios en la red a un bajo costo. Por lo tanto hay un incentivo económico para manipular los motores de búsqueda mediante la creación de páginas que hacen que otras páginas destaquen y tenga un mejor ranking dentro de los resultados de los buscadores, independiente de su real mérito.

2.1. Spam basado en contenidos

La primera generación de buscadores tenía básicamente mucha confianza en el clásico modelo de vectores espaciales del cual obtenían información. Así los primeros Web spammers manipulaban el contenido de las páginas repitiendo los keywords (claves) muchas veces. La mayoría de las páginas generadas de esta manera aún existen en la actualidad. Este tipo de resultados pueden caracterizarse como anómalos a través de un análisis estadístico [7], centrándose en la naturaleza de la estructura con la que cuenta la página. Además, éstas estructuras no solo cuentan con las palabras más usadas sino también con palabras escritas erróneamente tales como "googel", "accomodation" que suelen ser comúnmente escritas. Ntoulas [2] nos describe qué el tipo de características que se debería tomar en cuenta, son como el número de palabras, el tamaño del título, así como la fracción de palabras populares con las que cuenta la página.

Se ha demostrado, que en este tipo de Web spam tiene bastante éxito debido a que las máquinas de búsqueda no realizan filtros de spam en las consultas más populares y mejor pagadas [8]. Esta lista de consultas fácilmente puede ser encontrada en el servicio de pago por click de Google AdWords (http://adwords.google.com).

2.2. Spam basado en hiperenlaces

Google a través de su algoritmo PageRank realiza el ranking de las páginas en la búsqueda, marcó la pauta para que los otros buscadores comenzaran a usar este tipo de algoritmos. A partir de este momento también nació una gran cantidad problemas con el spam, debido a la facilidad que otorga este algoritmo para realizar este tipo de acciones.

Una de las principales maneras de realizar este tipo de spam es creando granjas de enlaces [6], que son un conjunto de páginas densamente conectadas, creadas explícitamente con el fin de engañar a los algoritmos basados en el ranking de los enlaces (ver ejemplo en Figura 1). Esto también es definido como colusión [10] que es "la manipulación de la estructura de los enlaces por un grupo de usuarios que intentan mejorar el ranking de uno o más usuarios en el grupo".

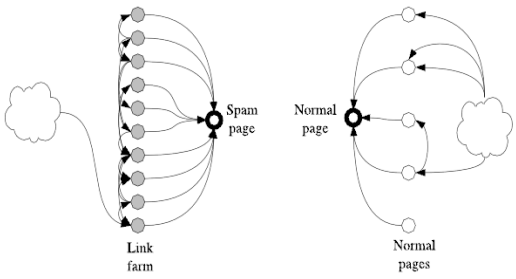

Figura 1. Esquema de los vecinos de una página participando en una granja de enlaces [6].

En la Figura 1 se ilustra la diferencia entre una página a la que se está tratando de elevar el ranking a través de una granja de enlaces. Por lo general, las

páginas que son spam no tienen mucha relación con la granja de enlaces, para evitar ser detectadas.

3. Metodología

A causa de la continua evolución de las técnicas usadas por spammers para crear nuevas formas de spamming, ha sido necesario, en el mecanismo de detección, utilizar una solución que aprenda y responda a través del tiempo, adaptable a futuros cambios siendo escalable para enfrentar futuros problemas de Web Spam. Para esto a continuación se presentan las metodologías usadas para implementar la solución y al final de todas éstas, el diseño principal del sistema detector.

3.1. Aprendizaje automático

El aprendizaje automático es el estudio de métodos para permitir a las computadoras aprender [11]. El área de máquinas de aprendizaje tiene que lidiar con el diseño de programas que puede aprender de los datos, adaptarse a los cambios y mejorar el desempeño con la experiencia.

En general, para tener un problema de aprendizaje bien definido [21], debemos identificar 3 características:

- El tipo de tarea
- La medida del desempeño a ser mejorada
- La fuente de experiencia

El presente trabajo utiliza técnicas de aprendizaje supervisado y hablaremos más al respecto a continuación.

3.2. Aprendizaje Supervisado y Clasificación

El Aprendizaje automático supervisado es la búsqueda de algoritmos que razonen a partir de ejemplos provistos externamente para producir hipótesis generales, que entonces hagan predicciones acerca de ejemplos futuros. En otras palabras la meta del aprendizaje supervisado es construir un modelo conciso de distribución de las etiquetas de clases en términos de un predictor de características [29].

Entre los tipos de aprendizaje supervisado se encuentra el de clasificación, el cual dado un conjunto de datos donde cada dato pertenece a una clase, construye un modelo que permite predecir la clase de un nuevo dato, en general el modelo es mejor si el error de predicción es menor. En nuestro proyecto se utiliza este algoritmo para clasificar entre las clases spam y no spam los datos no etiquetados.

3.3. Máquina de vectores de apoyo

Para resolver el problema de detección de Web Spam, utilizamos la alta capacidad de clasificación de las máquinas de vectores de apoyo, las mismas que fueron desarrolladas por Vapnik [3] y están basadas en la teoría de aprendizaje estadístico.

Considerando el caso más básico de clasificación, en el que hay que decidir entre dos clases diferenciables en el sentido de que se pueden separar mediante un hiperplano se lo conoce como clasificación binaria con datos linealmente separables, y consiste en:

Dados los vectores de entrenamiento:

$$x_i \in R^d \quad i = 1,.....,l$$

Etiquetados en dos clases en Y tal que:

$$y_i \in \{-1,1\}$$

Tenemos entonces:

$$\{x_i, y_i\};\ i = 1,....,l; y_i \in \{-1,1\}; x_i \in R^d$$

Además un hiperplano, que divide a las dos clases:

$$w \cdot x + b = 0 \quad (1)$$

Donde w es un vector normal al plano, ver Fig. 2.

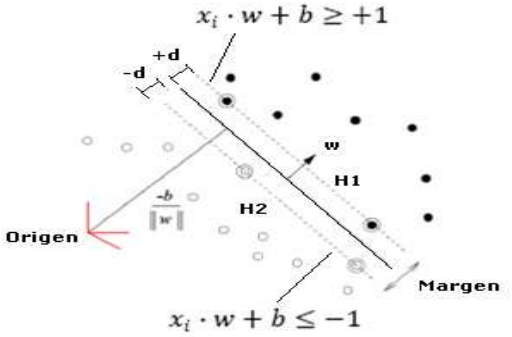

Figura 2. Muestra del hiperplano que separa las clases.

Sean $\frac{-b}{\|w\|}$ la distancia que existe desde el plano al origen y $\|w\|$ la norma euclidiana de w, siendo ésta el producto escalar $\sqrt{w \times w}$.

Sean $+d$ y $-d$ las distancias que existe entre el hiperplano al punto más cercano de ambas clases, satisfaciendo las siguientes condiciones:

$$x_i \cdot w + b \geq +1 \quad para \quad y_i = +1 \quad (2)$$

$$x_i \cdot w + b \leq -1 \quad para \quad y_i = -1 \quad (3)$$

Teniendo así una sola ecuación:

$$y_i (x_i \cdot w + b) - 1 \geq 0 \forall_i \quad (4)$$

Cabe señalar que H1 y H2 son hiperplanos paralelos y no contienen vectores entre ellos.

Dado que la suma de $+d$ y $-d$ es $\frac{1}{\|w\|^2}$ se pueden obtener los hiperplanos que maximicen el margen mediante la minimización de $\|w\|^2$ sujeto a las restricciones en (2 y 3).

Estos vectores que satisfacen las restricciones en (2 y 3), es decir que están en el borde de los hiperplanos, se conocen como vectores de soporte de la máquina.

Consideremos ahora el problema dual para la minimización:

$$\min_{\alpha} \frac{1}{2}\alpha^{\tau} Q\alpha$$

Sujeto a:

$$0 \le \alpha^{\tau} \le \frac{1}{l} \quad , \quad i = 1,.....,l$$

$$e^{\tau}\alpha \ge v \quad , \quad y^{\tau}\alpha = 0$$

Donde:

$$Q_{ij} \equiv y_i y_j K(x_i, x_j)$$

Entonces la función de decisión es:

$$F(x) = \text{sgn}(\sum_{i=1}^{1} y_i \alpha_i (K(x_i, x) + b)) \quad (5)$$

La ecuación 5, que está definida por la función sgn(x), sirve para la clasificación de futuros vectores y tiene como argumentos a los multiplicadores de LaGrange α_i y a los vectores de apoyo implícitos en la función Kernel $k(x_i, x)$.

Como argumentos positivos encontrados en base a la práctica se tiene que las máquinas de vectores de apoyo muestran un excelente desempeño tanto en escenarios donde existan vectores con pocas características [16] como en donde se necesiten significativamente menos cantidad de datos para el entrenamiento [17], asunto que es diferencial al momento de comparar este con otros métodos de clasificación.

3.4. Función Kernel

Visto anteriormente el problema de clasificación entre dos clases (+1,-1) se torna muy manejable cuando los datos son linealmente separables, es decir que pueden ser divididos por un hiperplano, pero un gran problema surge cuando aparecen escenarios en donde los datos no son linealmente separables. Para estos casos Vapnik presenta una solución al problema, mapeando la dimensión de los vectores de entrada a otra dimensión más alta, con el fin de tratar los vectores mapeados como un caso lineal separable. Dichos mapeos se los realizan a través de funciones denominadas Kernels [3].

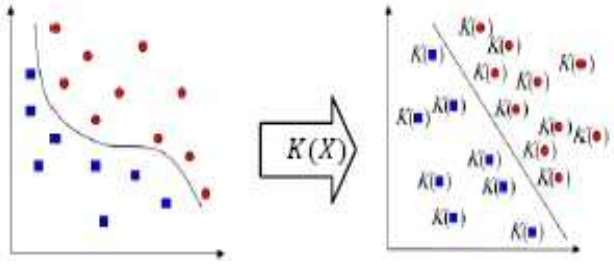

Figura 3. Ilustración del proceso de una función Kernel, llevando un espacio de una dimensión a otro de una dimensión más alta convirtiéndose en un caso linealmente separable.

Figura 4. Ejemplos del desempeño de las máquinas de vectores de apoyo con mapeos a través de las funciones Kernel, generados con LibSVM [18].

Como muestran la Figura 4 la distribución de los vectores en este caso no tiene un comportamiento lineal, es muy dispersa, debido a esto el uso de Kernels que lleva a los vectores de entrada del espacio a otro espacio de una mayor dimensión facilitando y permitiendo así la clasificación.

3.5. SVM en cascada

Las máquinas de vectores de apoyo a pesar de ser una excelente alternativa para los problemas de clasificación, han mostrado en la práctica que, a medida que se incrementan los datos de entrenamiento muestran un consumo de recursos demasiado elevado (memoria, procesamiento, etc.), agravando así el problema de escalabilidad de la solución [20].

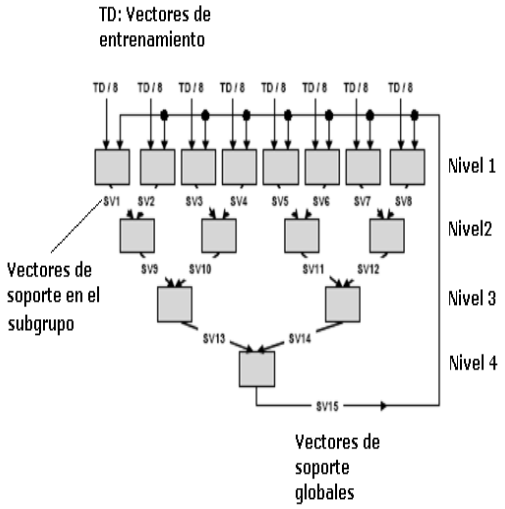

Figura 5. Modelo en cascada de SVM [20], los datos son divididos en subgrupos de datos donde cada grupo es evaluado de forma independiente.

Tal falencia de rendimiento con gran cantidad de datos, exige un mecanismo de paralelización que acelere los tiempos de procesamiento o entrenamiento del algoritmo. Este mecanismo es encontrado en [20] y presenta un modelo de varios niveles denominado SVM en cascada. (Ver Figura 5).

3.6. Vectores de características

Los vectores de características son vectores los cuales están constituidos por datos numéricos, cada uno de estos datos representan las características que se extraen de las páginas Web, mediante expresiones regulares, que se encuentran en nuestro dataset más una etiqueta que indica si es spam o no. Un pilar fundamental previo al entrenamiento de la máquina, es la apropiada selección del tipo de características a evaluar en las páginas, optimizando de esta manera el proceso de clasificación.

En este trabajo se utilizaron características basadas en los contenidos basándonos en el trabajo de Ntoulas [2]. A continuación se presenta la lista de las características que se han considerado para llenar nuestro vector:

Número de palabras en la página (f1): Para esta característica nosotros contamos la cantidad de palabras existentes en la página sin tomar en cuenta el contenido de los tags de las páginas.

Número de palabras en el título (f2): Se tome en consideración las palabras que se encuentran incluidas en tag “TITLE”, y se cuenta el número de palabras de este.

Promedio de palabras (f3): Se encuentra un promedio del tamaño de las palabras dentro de la página.

Fracción del texto anclado (f4): Para esta característica se hace una relación entre la cantidad de palabras que hay dentro de texto de anclas, palabras que se encuentra entre el tag “a”, dividido para la cantidad de palabras visibles dentro de la página.

Porcentaje de texto oculto (f5): Se hace una relación entre la cantidad de palabras de texto oculto y la cantidad total de palabras. Las palabras de texto oculto están localizadas dentro del contenido de alt que es una propiedad del tag “img” así como el texto dentro del tag “input” cuando este es del tipo hidden.

3.7. Modelo de programación MapReduce

Es un modelo de programación para procesamiento de grandes cantidades de datos implementado en ambientes con arquitecturas distribuidas, que permite computar los datos masivamente en forma paralela.

Como vemos en [30], la computación paralela con MapReduce se lo realiza tomando un conjunto de datos de entrada representados como duplas clave/valor y produciendo un conjunto de datos de salida también expresados como duplas clave/valor. El usuario del modelo MapReduce expresa la computación como la simple ejecución de dos funciones: Map y Reduce.

Un maper es el que realiza un proceso Map, el cual está escrito por un usuario. Es el que toma las duplas de entrada y produce una dupla clave/valor intermedia. La librería MapReduce agrupa todos los valores intermedios asociados con la misma clave intermedia y los pasa a la función Reducer.

La función Reducer, también escrita por el usuario, acepta una clave intermedia y un set de valores para la clave. Este junta los valores para formar un posible set de valores más pequeño. Típicamente cero y un valor de salida es producida por una invocación reducer. Los valores intermedios son proveídos al reducer del programa a través de un iterador. Esto permite manejar una lista de valores que son muy grandes para ser manejados por la memoria. La Figura 6 muestra de forma más clara como trabaja este modelo.

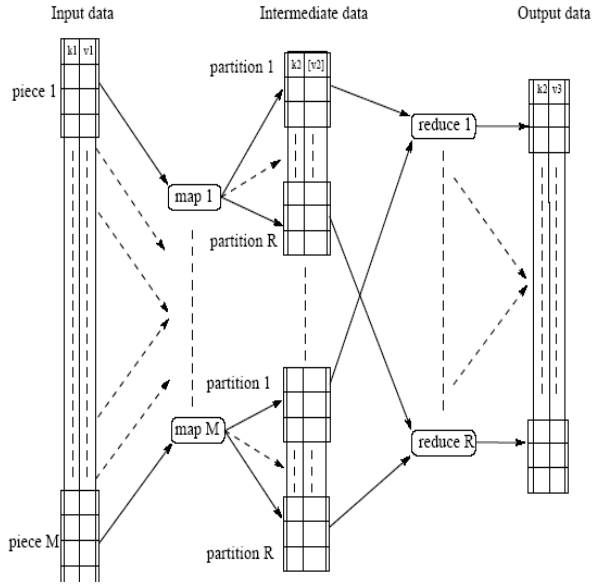

Figura 6. Modelo de trabajo MapReduce.

3.8. Amazon Web Services S3

El Amazon Simple Storage Service (S3) [24] es un repositorio para Internet. Fue diseñada para hacer la computación Web escalable más fácil para los desarrolladores.

Amazon S3 provee una interfaz de servicios Web que puede ser usada para almacenar y recuperar cualquier cantidad de información, en cualquier momento, desde cualquier lugar de la Web. Este da al desarrollador acceso a la misma alta escalabilidad, confiabilidad, rapidez, infraestructura de almacenamiento barata que Amazon usa para su propia red global de sitios Web. El servicio permite maximizar los beneficios de escala y pasar esto a los desarrolladores.

3.9. Amazon Web Services EC2

El Amazon Elastic Compute Cloud (EC2) [26] es un servicio Web que provee capacidad computacional reajustable en la nube. Está diseñado para hacer la computación Web escalable más fácil para los desarrolladores.

Amazon EC2 es una interfaz de servicios Web simple que te permite la capacidad de obtención y configuración con un mínimo desgaste. Te provee un completo control de tus recursos computacionales y los permite correr en un ambiente computacional probado de Amazon.

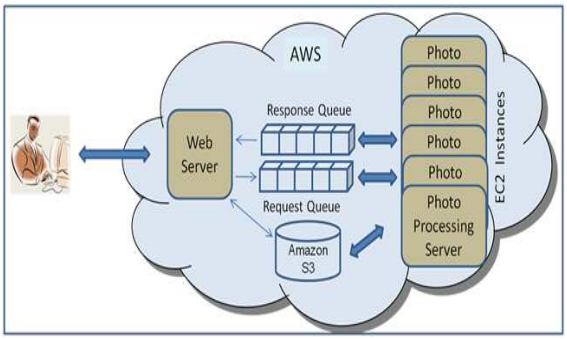

Figura 7. Ejemplo de EC2 y Amazon S3 trabajando juntos [28].

Amazon EC2 reduce el tiempo requerido para obtener y arrancar nuevos instancias de servidores al minuto, permitiendo así subir y bajar la capacidad escalable rápidamente. Amazon EC2 cambia la economía de la computación permitiéndote pagar solo por la capacidad que se usa. Amazon EC2 provee a los desarrolladores las herramientas para crear aplicaciones flexibles para errores y aisladas de posibles escenarios de error.

3.10. Modelo General

Para resolver la problemática de detección, se utilizaron procesos de tipo MapReduce, basando así el diseño en dos subprocesos principales: extracción, donde se excava la información en las páginas y entrenamiento, donde se generan los vectores de apoyo útiles para la clasificación.

En el subproceso de "extraer y cuantificar" se realizan los siguientes pasos:

- Se inicia una tarea MapReduce para la extracción y se almacena el dataset en el sistema de archivos distribuido de Hadoop (HDFS, por sus siglas en inglés) [22].
- Los mapers reciben las páginas almacenadas en el HDFS (Contenido HTML) como datos de entrada.
- Los mapers excavan y cuantifican la información en las páginas.
- Los mapers arman los vectores de características y los distribuyen a los reducers.
- Los reducers escriben los vectores en el HDFS.

En el subproceso de "Entrenamiento" se realizan los siguientes pasos:

- Se inicia una tarea MapReduce para el entrenamiento y se asigna la ubicación en el HDFS, de donde se sacan los datos de entrada

para la tarea (la ubicación debe ser el directorio de salida del subproceso de extracción).

- Los mapers reciben los vectores armados desde el HDFS.
- Los mapers distribuyen los vectores a los reducers, formando así grupos de vectores.
- Los reducers reciben los grupos vectores (un grupo por reducer) y se eligen los vectores de apoyo para dichos grupos.

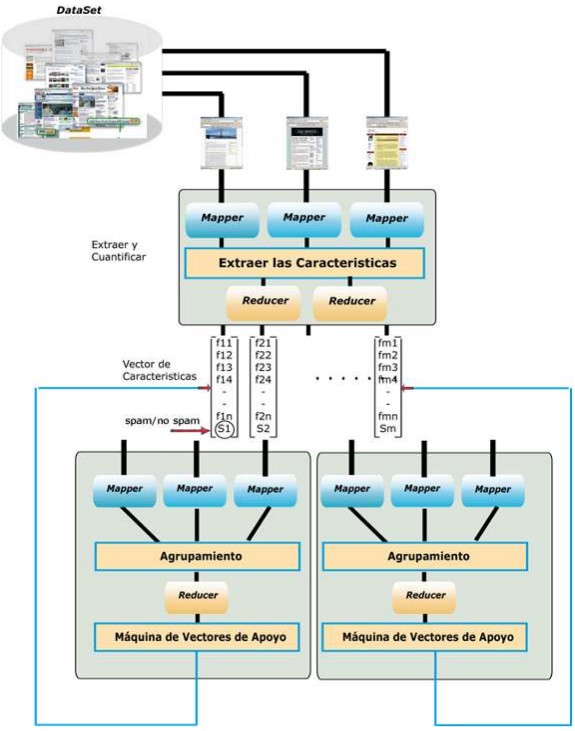

Figura 8. Modelo general de procesos.

4. Implementación

El almacén de datos fue utilizado fue obtenido usando el software UbiCrowler [13] y consta de un total de aproximadamente 120,000 páginas, que representa solo un volumen de los 8 obtenidos Cada volumen tiene un tamaño aproximado de 1.7 GB.

Para la implementación de las máquinas de vectores de apoyo se utilizó LibSVM [18], expresiones regulares para extraer la información de las páginas y Hadoop [22] como marco para trabajar con el modelo MapReduce.

5. Resultados

En la Tabla 1, que muestran resultados del procesamiento paralelo utilizando los clusters de Amazon, se obtuvo utilizando un Kernel de Base radial (RBS, por sus siglas en inglés) computando un total de 67,577 vectores (páginas Web extraídas) y obteniendo 20,338 vectores de apoyo necesarios para la clasificación.

Tabla 1. Resultados del procesamiento paralelo.

Nodos EC2	Extracción Tiempo (seg.)	Entrenamiento Tiempo (seg.)
3	83.127	302
8	50.54	250

En la tabla 2 observamos el comportamiento en eficiencia de la máquina de vectores de apoyo visto como una matriz de confusión.

Tabla 2. Matriz de confusión.

		Clases predichas		
		1	0	-1
Clases conocidas	1	89,70%	0	0
	0	0,30%	0	0
	-1	10,00%	0	0

Analizando la matriz de confusión los resultados muestran un buen desempeño en el detector teniendo en cuenta lo siguiente:

- El dataset de entrenamiento está compuesto en su mayoría por páginas etiquetadas como no spam.
- Por lo expuesto en el punto anterior, el detector puede predecir con certeza cuando una página no es Web spam, más no lo contrario.

5. Conclusiones

Hadoop es una herramienta muy poderosa y de gran utilidad en la actualidad por ser una excelente alternativa para el procesamiento distribuido.

El uso de los servicios Web de Amazon permiten ahorrar costos de infraestructura, y es de gran ayuda para empresas que recién comienzan.

SVM es una muy capaz herramienta de clasificación, adaptable a cualquier problema. Para el presente trabajo mostró resultados muy aceptables.

Los problemas de rendimiento generados por el consumo de recursos de SVM como tal, pueden ser solucionados con técnicas de paralelismo como con el paradigma MapReduce, trabajado en este proyecto.

5. Trabajos Futuros

En nuestro proyecto utilizamos SVM en cascada, es posible implementar otro tipo de paralelismo como la solución de "Sub-problemas cuadráticos".

Extender la dimensión de los vectores de características, para asegurar una mejor clasificación. Tales características aumentadas que incrementan la dimensión de los vectores, podrían ser analizadas y cuantificadas de las relaciones de una página con sus vecinos más cercanos.

Se pueden utilizar mecanismos de validación cruzada para el ajuste de parámetros en el modelo de la máquina de vectores de apoyo (SVM).

5. Referencias Bibliográficas

[1]. C. Castillo, D. Donato, L. Becchetti, P. Boldi, M. Santini, and S. Vigna. A reference collection for web spam detection. Technical report, DELIS, September 2006.

[2]. Ntoulas, M. Najork, M. Manasse and D. Fetterly. Detecting spam Web pages through content analysis. In Proceedings of the World Wide Web conference, pages 83–92, Edinburgh, Scotland, May 2006.

[3]. V. Vapnik. The Nature of Statistical Learning Theory. Springer, N.Y., 1995. ISBN 0-387-94559-8.

[4]. Z. Gyongyi and H. Garcia-Molina. Web spam taxonomy. In AIRWeb, 2005.

[5]. B. Jansen and A. Spink. An Analysis of Web Documents Retrieved and Viewed. In International Conference on Internet Computing, June 2003.

[6]. Becchetti and Carlos Castillo and Debora Donato and Stefano Leonardi and Ricardo Baeza-Yates. Link-Based Characterization and Detection of Web Spam, In AIRWeb 2006.

[7]. Dennis Fetterly and Mark Manasse and Marc Najork. Spam, Damn Spam, and Statistics: Using statistical analysis to locate spam web pages. In Proceedings of WebDB 2004.

[8]. Carlos Castillo and Debora Donato and Vanessa Murdock and Fabrizio Silvestri. Know your neighbors: Web spam detection using the web topology. In Proceedings of SIGIR 2007.

[9]. L. Page, S. Brin, R. Motwani and T. Winograd. The PageRank Citation Ranking: Bringing Order to the Web. Stanford Digital Library Technologies Project, 1998.

[10]. H. Zhang, A. Goel, R. Govindan, K. Mason, and B. Van Roy. Making eigenvector-based reputation systems robust tocollusion. In Proceedings of the third Workshop on Web Graphs (WAW), volume 3243 of Lecture Notes in Computer Science, pages 92–104, Rome, Italy, October 2004. Springer.

[11]. Machine Learning, Thomas G. Dietterich.

[12]. C. Castillo, D. Donato, L. Becchetti, P. Boldi, M. Santini, and S. Vigna. A reference collection for web spam detection. Technical report. DELIS, September 2006.

[13]. P. Boldi, B. Codenotti, M. Santini, and S. Vigna. Ubicrawler: a scalable fully distributed web crawler. Software, Practice and Experience, 34(8):711–726, 2004

[14]. http://www.niso.org/ .

[15]. http://www.dmoz.org/.

[16]. Thorsten Joachims, Fachbereich Informatik, Lehrstuhl Viii. Text Categorization with Support Vector Machines: Learning with Many Relevant Features.

[17]. Zhenchun Lei, Yingchun Yang, Zhaohui Wu .Ensemble of Support Vector Machine for Text-Independent Speaker Recognition. IJCSNS International Journal of Computer Science and Network Security, VOL.6 No.5A, May 2006

[18]. http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[19]. Máquinas de soporte vectorial en el reconocimiento automático del habla.

[20]. Hans Peter Graf and Eric Cosatto and Leon Bottou and Igor Durdanovic and Vladimir Vapnik. Parallel support vector machines: The cascade SVM. In Advances in Neural Information Processing Systems, pages 521--528, 2005

[21]. Tom Mitchell, Machine Learning, 1997, Mc-Graw Hill

[22]. Doug Cutting, Hadoop the definitive Guide, published by O'Railly, 2009

[23]. Jeffrey Dean and Sanjay Ghemawat. MapReduce: Simplified Data Processing on Large Clusters. In OSDI 04, 6th Symposium on Operating Systems Design and Implementation, Sponsored by USENIX, in cooperation with ACM SIGOPS, pages 137 – 150, 2004

[24]. http://aws.amazon.com/s3/

[25]. http://docs.amazonwebservices.com/AmazonS3/latest/images/devpay_installations.gif

[26]. http://aws.amazon.com/ec2/

[27]. http://www.yr-bcn.es/webspam/datasets/

[28]. http://developer.amazonwebservices.com/connect/entry.jspa?externalID=1464.

[29]. S. B. Kotsiantis Supervised Machine Learning: A Review of Classification Techniques, Department of Computer Science and Technology University of Peloponnese, Greece.

[30]. Ralf Lämmel, Google's MapReduce programming model, published by Elsevier North-Holland, Inc, 2007.