Modelos para detectar la polaridad de los mensajes en redes sociales

Yuvila M. Sanzón, Darnes Vilariño, María J. Somodevilla, Claudia Zepeda, Mireya Tovar

Benemérita Universidad Autónoma de Puebla,
Facultad de Ciencias de la Computación, Puebla,
México

{yuvisrosas, dvilarino, mariajsomodevilla, czepdac}@gmail.com, mtovar@cs.buap.mx

Resumen. En el presente artículo se presentan dos modelos para descubrir la polaridad de mensajes en redes sociales, en particular extraídos del Twitter. El primer modelo extrae las características léxico-sintácticas de cada tweet. El segundo modelo obtiene las características de cada tweet basándose en la centralidad de grafos.

Palabras clave: Análisis de sentimientos, redes sociales, grafos de co-ocurrencia.

1. Introducción

Gracias a la expansión de la Web 2.0 y a la participación activa de los usuarios en redes sociales, blogs, foros y páginas dedicadas a críticas *(reviews)* en los últimos años, se ha visto un crecimiento exponencial de la información subjetiva[1] disponible en Internet. Este fenómeno ha originado interés por detectar sentimientos, emociones y opiniones expresadas sobre tópicos u objetos diferentes. De esta manera, surge la necesidad de contar con herramientas que detecten, extraigan y estructuren dicha información subjetiva.

El análisis de sentimientos se ha abierto camino en años recientes y han aparecido multitud de escenarios de uso y aplicaciones. La relevancia de herramientas de este tipo radica en la posibilidad de evaluar las opiniones expresadas por los usuarios acerca de un tópico u objeto de interés y encontrar problemas, debilidades y fortalezas en diferentes aspectos de los productos y servicios que consumen los mismos. Es posible también medir el grado de satisfacción de los usuarios acerca de un fenómeno

[1] Es la información que se presenta desde un solo punto de vista. Generalmente, expresa la interpretación o perspectiva de una persona o de un grupo de personas.

y predecir su evolución (el "sentimiento del mercado") o incluso medir la tendencia de la preferencia política. Esa información es clave para identificar áreas de oportunidad de desarrollo o cambio en la imagen de un producto, una campaña o simplemente observar la opinión popular acerca de un tópico de interés general.

El análisis de sentimientos impone retos para el Procesamiento del Lenguaje Natural (PLN), principalmente debido a la diversidad de dominios sobre los cuales se expresan las opiniones, la informalidad en la escritura de los textos y la falta de conjuntos de datos muestra.

En el presente trabajo se presentan dos modelos que permiten detectar la polaridad de un mensaje, el primero utiliza características léxico- sintácticas y el segundo extrae las características a utilizar a través de un grafo de co-ocurrencia. La polaridad se refiere a la presencia o ausencia de partículas gramaticales que definen si la oración es positiva o negativa.

En la sección 2 se presentan los diversos trabajos acerca del análisis de sentimientos. En la sección 3 se describe la metodología que se empleó para realizar los modelos propuestos. Finalmente en la sección 4 se muestran los resultados obtenidos.

2. Estado del arte

Se han desarrollado distintos trabajos en el área de análisis de sentimientos, la mayoría de estos se han centrado en el idioma inglés, dado que existe una gran cantidad de herramientas de procesamiento del lenguaje disponibles y se dispone de conjuntos de datos que pueden ser usados para el entrenamiento y creación de modelos de clasificación.

En [1] se propone el desarrollo de un motor de detección (disponible en www.umigon.com) que está diseñado específicamente para detectar el sentimiento positivo, negativo o neutral en *tweets*, el cual consta de cuatro partes principales: detección de rasgos semánticos del tweet como emoticones y onomatopeyas; evaluación de los hashtags; descomposición del *tweet* en una lista de n-gramas, comparando cada n-grama con los términos léxicos, en caso de coincidencia se aplica una heurística; finalmente, aprovechando los rasgos semánticos detectados anteriormente, se aplica una serie de heurísticas a través de todo el *tweet*. Dado que los emoticones y onomatopeyas tienen fuertes indicios de sentimiento, pero también tienen una gran variedad ortográfica, se tiene una lista con las exclamaciones más comunes y se utilizan expresiones regulares para capturar la variedad de formas que pueden asumir. Para la evaluación de los *hashtags* se aplica una serie de heurísticas de forma que el *hashtag* (en caso de que sea un *hashtag* compuesto, éste se descompone) coincida con los términos léxicos. El *tweet* es descompuesto en una lista de unigramas, bigramas, trigramas y tetragramas, se recorren todos los *n-gramas* del *tweet*, y se realizan comprobaciones de su presencia en diccionarios de términos léxicos. Si un *n-grama* es encontrado en alguno de los diccionarios, se aplica la heurística adjunta a este término presente en aquel diccionario, regresando una clasificación (positivo, negativo o neutro) para ese n-grama. Para el análisis de sentimientos se utilizaron cuatro diccionarios: tono positivo, tono negativo, fuerza del

sentimiento y negaciones; los cuales fueron creados manualmente. Con este sistema se obtuvo una precisión promedio (positivo y negativo) de 69.02%.

Otro sistema desarrollado que se debe destacar es el presentado en [2], para su implementación se hizo uso del kit de herramientas MALLET (paquete basado en Java para el procesamiento de lenguaje natural). Para la normalización de los *tweets* se realizaron las siguientes tareas: todas las palabras se convierten a su forma minúscula (utilizando el algoritmo de *Porter Stemming*), se sustituyen @ y # por las notaciones `[usuario]`, `[tag]` respectivamente, los emoticones se clasifican en positivos y negativos, se remueven caracteres innecesarios, en el caso de palabras que contienen repeticiones de caracteres se reduce la longitud, sólo teniendo en cuenta una secuencia de tres caracteres, con el fin de unificar estas repeticiones, finalmente se realizó un filtrado de palabras (palabras no significativas). Después de la normalización de los mensajes se determinó la polaridad de cada palabra utilizando el diccionario de sentimientos *SentiWordNet*, se consideró una palabra como positiva si el valor positivo relacionado es mayor a 0.3; como negativa, si el valor negativo relacionado es mayor a 0.2 y como neutral si el valor relacionado es mayor a 0.8. Una vez calculada la polaridad se contemplan tres características para cada *tweet* que son: el número de palabras positivas, negativas y objetivas respectivamente; se verifica si a una palabra positiva le precede una negación, si es así la polaridad se invierte; se utilizó un diccionario de siglas y para cada sigla se utilizó una polaridad. El mejor resultado que se reporta es de una precisión del 54%, esto usando un modelo basado en la obtención de características y la normalización de los tweets, además de usar como clasificador de aprendizaje de máxima entropía.

Encontrar las características o elementos relevantes presentes en un texto, es una parte fundamental en el proceso de clasificación supervisada. En la literatura existe una amplia variedad de trabajos relacionados a la extracción de características, pero de los trabajos que hacen énfasis en la extracción de características para el análisis de sentimientos se pueden encontrar como relevantes los descritos en [3] y [4]. En [3] y [4] se muestra que el uso de las categorías gramaticales[2] (PoS tag, Part of Speech tag) de las palabras como características sintácticas puede ayudar de forma simple a desambiguar la polaridad de las palabras. También se muestra que el uso de frecuencia de aparición de los sustantivos y los adjetivos es importante para identificar la subjetividad de las oraciones para todas las categorías gramaticales. Además, en [5] se muestra que el uso de ciertas reglas asociadas a las categorías gramaticales puede ser de utilidad para detectar patrones de sentimiento dentro de los textos. Entre los patrones más relevantes se muestra que el uso de adverbios puede ayudar a detectar la negación de las oraciones (asociado normalmente a un sentimiento positivo).

La representación de la información permite conjuntar todas las características de un texto en un esquema específico, el cual facilita la construcción de modelos de clasificación para descubrir el sentimiento de un texto dado. En el resto de esta sección se mencionan algunos de los artículos relacionados con las representaciones usadas para el análisis de sentimientos.

[2] Proceso de asignar una etiqueta gramatical a cada una de las palabras de un texto según su categoría léxica.

En [6] se presenta una comparación sobre el uso de distintas formas de representación del conocimiento para la minería de opinión. El artículo destaca el uso de una representación vectorial basada en la frecuencia de aparición de los elementos, como una forma de encapsular información numérica relevante de una forma fácil y optimizada, pero que ignora la información estructural y semántica presente en los textos, es decir, toda la información acerca de cómo están relacionadas las palabras entre sí, ya sea dentro de un párrafo o una oración, así como la relación que guardan las palabras por medio del contexto que las rodea.

En [7] se describe el uso de una representación vectorial basada en la presencia de las características como una forma reducida de la frecuencia de aparición en el contexto de la clasificación supervisada. En el artículo se muestra que una representación basada en la presencia de las características puede ser de utilidad en problemas donde no exista una gran cantidad de documentos asociados al entrenamiento ya que sólo es necesario cuantificar una sola ves un elemento para que tenga una fuerte presencia dentro del vector.

En [8] se introduce el uso de grafos para representar la información de los textos en la fase de entrenamiento. En específico se propone el uso de una representación basada en la ocurrencia de términos en una ventana de tamaño predeterminado llamada co-ocurrencia, usada para determinar relaciones de proximidad semántica relevante. Como aportación adicional se propone el uso de técnicas de ranking sobre los nodos del grafo asociado al uso de medidas de similaridad como la distancia euclidiana, entre otros.

En [9] se presenta el uso de técnicas de análisis de redes sociales para determinar la importancia de los nodos a través de las relaciones que estos forman y los caminos posibles entre ellas. En específico se propone el uso de varias métricas novedosas entre las que se encuentran las centralidades de grado, de cercanía, entre otras; con el propósito de analizar el comportamiento de una red de nodos y cuáles son los nodos más importantes y centrales por los que debe de propagarse la información.

A pesar de que se han realizado diversas investigaciones que permiten detectar la polaridad de un cierto mensaje, los resultados que se han obtenido dependen mucho de las características del corpus de entrenamiento, en este sentido es conveniente buscar la forma de extraer características que no estén asociadas directamente al corpus, sino a la forma en que se expresan los usuarios en las redes sociales. Esto justifica perfectamente la presente investigación.

3. Metodología

Para descubrir la polaridad de mensajes en redes sociales y en específico los mensajes provenientes del tweeter, se han desarrollado dos modelos implementados en el lenguaje Python, con ayuda de las herramientas Network X y CLIPS Pattern. El primero se basa en las características léxico – sintácticas de cada *tweet* y consta de tres fases (normalización, entrenamiento y prueba). El segundo modelo obtiene las características de cada *tweet* basándose en la centralidad de grafos aplicada a todo el corpus de entrenamiento, este modelo se compone de cinco fases (normalización, representación del grafo, selección de características y representación vectorial, entrenamiento y prueba). En esta sección se detallan ambos modelos.

3.1. Modelo léxico-sintáctico

Este modelo esta implementado en tres fases (Fig. 1). Para el desarrollo del mismo se utilizó el corpus de la competencia SemEval 2014, el cual contiene un conjunto de *tweets* etiquetados con cinco sentimientos diferentes (positivo, negativo, neutral, objetivo y objetivo o neutral). Sólo se trabajó con las clases: positivo, negativo y neutral. Las tres fases mencionadas anteriormente se describen a continuación:

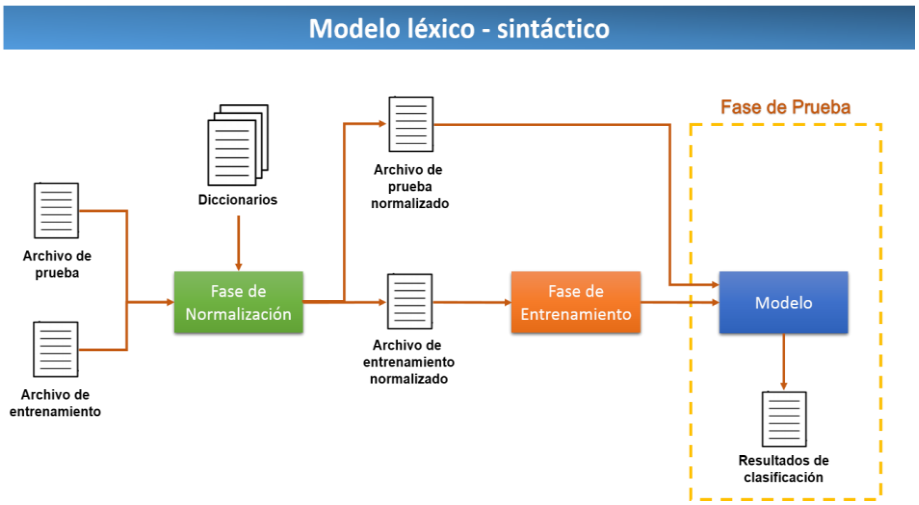

Fig. 1. Arquitectura del modelo léxico-sintáctico.

Fase de normalización. En esta fase se realiza el pre-procesamiento de los datos de entrenamiento y de los datos de prueba, para esto se desarrollaron dos diccionarios de forma manual, el primero contiene emoticones y una palabra representativa de su significado; el otro diccionario contiene algunas de las siglas empleadas en redes sociales y su significado. Esta fase comienza convirtiendo el contenido de los archivos a su representación en minúsculas, posteriormente se reemplazan los emoticones y abreviaturas, por la palabra o palabras correspondientes. Se eliminaron elementos como las *URLs*, los *hashtags* y los nombres de usuario, debido a que se consideró que no eran candidatos a características que permitiera detectar algún sentimiento. Al finalizar esta fase se obtienen los archivos de entrenamiento y prueba que son utilizados en la fase de entrenamiento.

Fase de entrenamiento. En esta fase se utilizan los clasificadores Naive Bayes y Máquina de Soporte Vectorial, proporcionados por la herramienta CLIPS Pattern. El modelo de clasificación es desarrollado con cada uno de los *tweets* contenidos en el archivo de entrenamiento normalizado, este modelo se utiliza para clasificar los datos en la fase de prueba.

Fase de prueba. En esta fase se emplea el archivo de prueba normalizado, de éste se extraen los *tweets* que serán enviados al modelo quien se encargará de asignar un sentimiento a cada *tweet*.

3.2. Modelo sobre grafos

Este modelo está formado por cinco fases como se aprecia en la Fig. 2, al igual que en el modelo anterior se emplearon los corpus proporcionados en la tarea 9: análisis de sentimientos en Twitter del SemEval 2014. A continuación se describe de forma detallada el proceso empleado en este modelo.

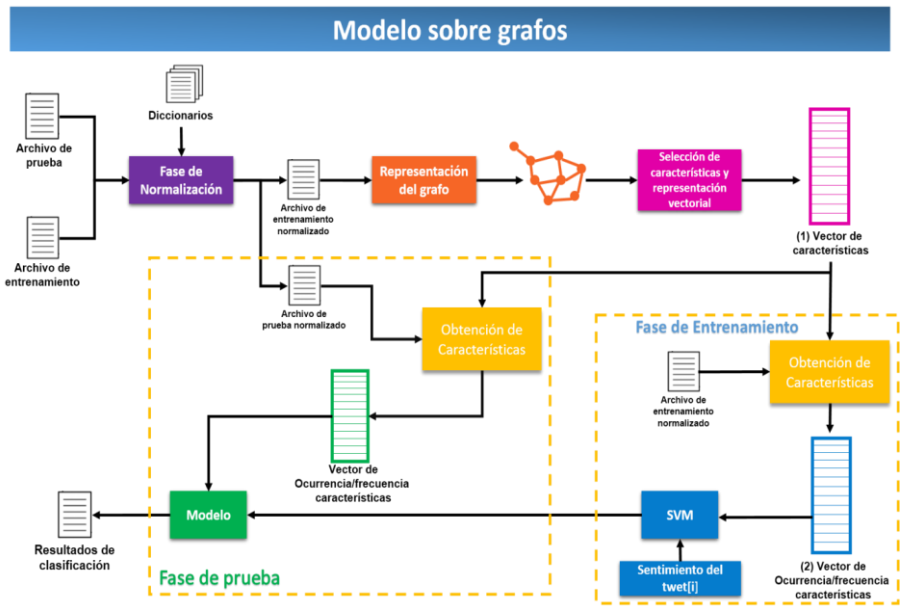

Fig. 2. Arquitectura del modelo sobre grafos.

Fase de normalización: al igual que en el modelo anterior se realizó un pre-procesamiento de los datos de entrenamiento y de prueba. Todos los tweets se llevan a minúsculas, posteriormente se reemplazan los emoticones y abreviaturas por su significado, utilizando los diccionarios antes mencionados, por último se eliminan *URLs*, *hashtags* y los nombres de usuario.

Representación del grafo:
Un grafo se define como un par (V, E), donde V es un conjunto no vacío cuyos elementos son denominados vértices o nodos y E es un subconjunto de pares no ordenados de vértices y que reciben el nombre de aristas o arcos [10].
Entre las distintas propuestas para la representación de grafos en el análisis de textos, la co-ocurrencia de palabras se ha convertido en una forma simple, pero eficaz de representar la relación de un término con respecto a otros en un grafo. Formalmente dos términos co-ocurren si están presentes en una ventana[3] de texto N [8]. Tomando en cuenta lo anterior, un grafo de co-ocurrencia no dirigido, es representado por:

[3] Se refiere a la cantidad de n sucesivas palabras con las que tendrá una conexión cada palabra en el texto.

G = (V, E), donde:

- V, es un conjunto de vértices que está formado por los términos contenidos en uno o varios textos.

- E, es un subconjunto de pares de vértices, que representa la relación entre los términos que forman dichos vértices.

En esta fase, se construyó la representación del corpus de entrenamiento por medio de un grafo, empleando el siguiente procedimiento.

1. Se crea un grafo vacío no dirigido.
2. Se obtienen las palabras que componen cada *tweet*.
3. Para cada palabra que se obtuvo se agrega una arista dentro del grafo, que una a esta palabra y a las siguientes *n* palabras, donde *n* es el valor de la ventana.
4. Se repite el procedimiento a partir del paso 2 para cada *tweet* dentro del corpus.

De manera que cada palabra distinta en el corpus se convierte en un nodo dentro del grafo, así bien los nodos se conectarán con otros nodos, si las palabras que representan dichos nodos co–ocurren dentro del valor de la ventana. En las Fig. 3 y 4 se ejemplifican la representación del grafo de co–ocurrencia con un ancho de ventana igual a 2 y un ancho de ventana igual a 3 respectivamente, utilizando los siguientes tres *tweets*, a los cuales ya les fueron removidas las *stopwords.*

1. count day tomorrow shouldnt winans crazy,
2. finish watching vow tomorrow cute movie,
3. excited nuggets game tomorrow.

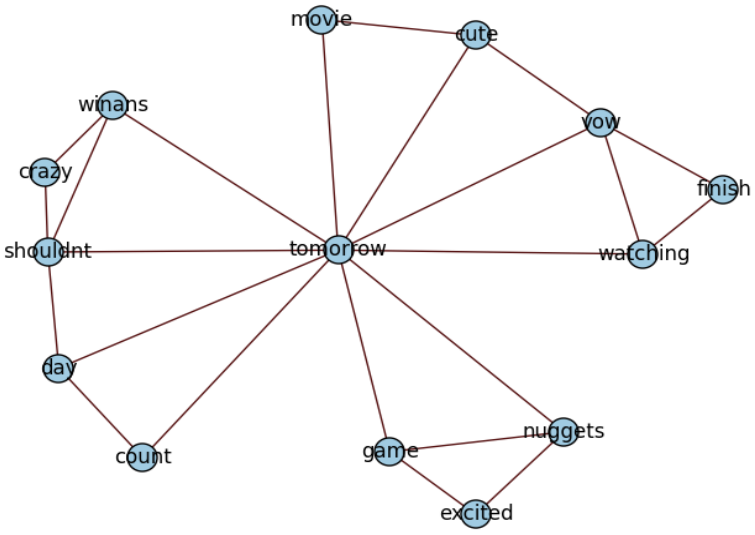

Fig. 3. Grafo con ventana igual a 2.

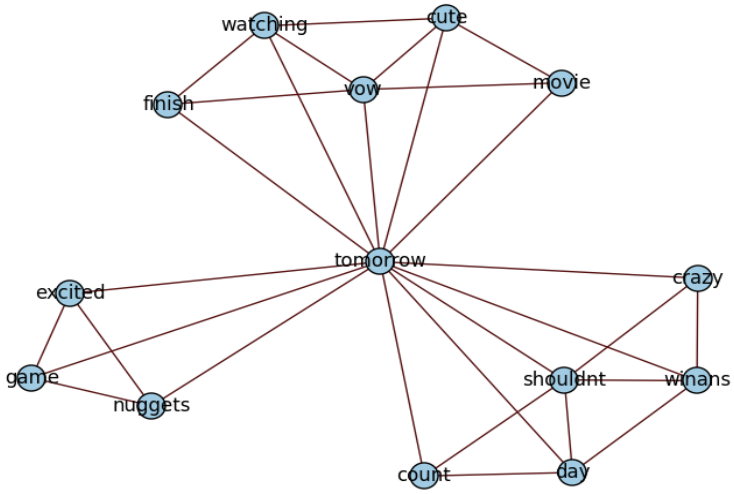

Fig. 4. Grafo con ventana igual a 3.

Se realizaron dos variaciones de este modelo, la primera utilizando todo el corpus de entrenamiento y la segunda separando el corpus de entrenamiento en tres archivos (positivo, neutral y negativo) obteniendo así tres grafos. Estas representaciones se discuten a continuación.

a) Para la representación utilizando todo el corpus de entrenamiento, en primer lugar se eliminaron las *stopwords*[4] del archivo de datos. Con el archivo resultante de este pre-procesamiento se construyó el grafo de co–ocurrencia con un ancho de ventana igual a 2 y ancho de ventana igual a 3, usando la herramienta Network X.
b) De igual manera se eliminaron las *stopwords* del corpus de entrenamiento. El archivo de entrenamiento sin *stopwords*, se separó en tres archivos: *tweets* positivos, *tweets* neutros, *tweets* negativos. Para cada uno de estos archivos se construyeron los grafos de co–ocurrencia con un ancho de ventana igual a 2 y un ancho de ventana igual a 3.

Selección de características y representación vectorial: En esta fase se describen dos variaciones una con el grafo conformado por todo el corpus de entrenamiento y otra con los tres grafos obtenidos por la separación del corpus con respecto a su etiqueta de sentimiento. A los grafos les fue aplicado un algoritmo de centralidad. Para las pruebas se emplearon cuatro de estos algoritmos:

1. Centralidad de grado: Corresponde al número de enlaces que posee un nodo con los demás [11].
2. Centralidad de cercanía: La suma o bien el promedio de la distancias más cortas desde un nodo hacia todos los demás en un grafo [11].

[4] Palabras sin significado como artículos, pronombres, preposiciones, etc.

3. Centralidad de intermediación: Es una medida que cuantifica la frecuencia o el número de veces que un nodo actúa como un puente a lo largo del camino más corto entre otros nodos [11].
4. Centralidad de vector propio: Mide la influencia de un nodo en una red. Los nodos que poseen un valor alto de esta medida de centralidad están conectados a muchos nodos que a su vez están bien conectados [11].

Dichas medidas las posee la herramienta Network X, estos algoritmos retornan un diccionario[5] de Python con los nodos (palabras) y su valor de centralidad, ya sea de grado, cercanía, intermediación o vector propio. A continuación se describe el resto del proceso para las dos alternativas de selección de características.

a) A partir de los valores contenidos en el diccionario, éstos se ordenaron de mayor a menor, se eligieron los 300 nodos (palabras) más centrales, tomando en cuenta sólo: adjetivos, adverbios, verbos, adverbios comparativos, adverbios superlativos, adjetivos comparativos, adjetivos superlativos, interjecciones, verbos en tercera persona, en presente, en pasado, en gerundio y sustantivos, estas fueron las palabras seleccionadas para obtener el vector de características de tamaño 300.
b) Para esta variación se tienen tres grafos, uno para cada sentimiento, a cada grafo le fue aplicado un algoritmo de centralidad, obteniendo así un diccionario con los valores de centralidad. Los valores contenidos en cada diccionario se ordenaron de mayor a menor, y se seleccionaron 100 nodos (palabras) más centrales de cada diccionario, tomando en cuenta sólo: adjetivos, adverbios, verbos, adverbios comparativos, adverbios superlativos, adjetivos comparativos, adjetivos superlativos, interjecciones, verbos en tercera persona, en presente, en pasado, en gerundio y sustantivos. Finalmente se unieron las 100 palabras más centrales de cada diccionario para formar el vector de 300 características.

Fase de entrenamiento: Para esta fase se crean dos tipos de vector: uno de ocurrencia y otro de frecuencia. Los tipos de entrenamiento realizados se describen a continuación.

a) Con el vector de características obtenido en la fase anterior, se procede a obtener características. Se calcula el vector de ocurrencia para cada *tweet* que se encuentre en el archivo de entrenamiento sin *stopwords*. Si la palabra *i* en el vector de características se encuentra en el *tweet*, se coloca un *1* en la posición *i* en el vector de ocurrencia, una vez terminado este proceso se entrena una máquina de soporte vectorial usando el vector de ocurrencia junto con la etiqueta de sentimiento de cada *tweet*. Obteniendo asi un modelo de clasificación supervisado.
b) Basados en el vector de características se calcula el vector de frecuencia para cada *tweet* que se encuentre en el archivo de entrenamiento sin *stopwords*. Si la palabra *i* en el vector de características se encuentra en el *tweet*, se incrementa en *1* el contenido en la posición *i* en el vector de

[5] Contenedor de pares clave – valor

frecuencia, de igual forma se entrena una máquina de soporte vectorial usando el vector de frecuencia junto con la etiqueta de sentimiento de cada *tweet*. Se obtinene un modelo de clasificación supervisado.

Fase de prueba: De la misma manera que en la fase de entrenamiento se obtiene el vector de ocurrencia o frecuencia, según sea el caso. Con el modelo obtenido se procede a clasificar los tweets que contiene el archivo de prueba normalizado, obteniendo el archivo de resultados de la clasificación.

4. Resultados

En esta sección se presentan los resultados obtenidos por los modelos para la detección de sentimiento en los *tweets*, se describen el clasificador empleado, los aciertos, y su porcentaje de precisión, finalmente se muestran gráficas para la comparación de los mejores resultados.

Como se planteó con anterioridad, el corpus tanto de entrenamiento como de prueba fue obtenido de la Conferencia Semeval 2014. El corpus de entrenamiento se compone de 6364 *tweets* (Tabla 1); se hizo una variante de este corpus pero balanceado, en el cual se igualaron la cantidad de elementos por clase, se tomaron 905 tweets de cada clase, eliminando aleatoriamente tweets de las clases que tenían un sobrante (Tabla 2). El corpus de prueba tiene un total de 8987 *tweets* (Tabla 3). Para las pruebas de ambos modelos se realizó el entrenamiento, con el corpus de entrenamiento completo y con el corpus de entrenamiento balanceado es decir, éste corpus se formó con 905 *tweets* negativos, positivos y neutros, obteniendo un total de 4525 *tweets*.

Tabla 1. Composición del corpus de entrenamiento.

Corpus de entrenamiento			
Tweets positivos	Tweets neutros	Tweets negativos	Total de tweets
2319	905	3140	6364

Tabla 2. Composición del corpus de entrenamiento balanceado.

Corpus de entrenamiento			
Tweets positivos	Tweets neutros	Tweets negativos	Total de tweets
905	905	905	2715

Tabla 3. Composición del corpus de prueba.

Corpus de prueba			
Tweets positivos	Tweets neutros	Tweets negativos	Total de tweets
3506	1541	3940	8987

4.1. Resultados del modelo léxico-sintáctico

En las pruebas de éste modelo se trabajó con el corpus de entrenamiento completo y con el corpus de entrenamiento balanceado, y los clasificadores que se utilizaron fueron Naïve Bayes y SVM.

En la Tabla 3 se muestran los resultados obtenidos con el clasificador Naïve Bayes para el corpus de entrenamiento completo, aplicando el modelo léxico – sintáctico.

Tabla 3. Resultados de precisión, corpus de entrenamiento completo, con Naïve Bayes.

Naïve Bayes Corpus de entrenamiento completo					
Positivos	Tweets Neutrales	Negativos	Total de aciertos	Total de Tweets	% de Precisión
1813	562	1873	4248	8987	47.26

Los resultados obtenidos al balancear el corpus se puede observar en la Tabla 4.

Tabla 4. Resultados de precisión, corpus balanceado, con Naïve Bayes.

Naïve Bayes Corpus de entrenamiento balanceado					
Positivos	Tweets Neutrales	Negativos	Total de aciertos	Total de Tweets	% de Precisión
1771	1434	818	4023	8987	44.76

Los resultados obtenidos con el clasificador máquina de soporte vectorial y el corpus de entrenamiento completo se describen en la Tabla 5. El único resultado significativo fue el de la prueba con el kernel lineal que arrojó una precisión del 56.58%, los demás resultados no tienen relevancia, puesto que todos los *tweets* son asignados a la clase neutral.

Tabla 5. Resultados de precisión, corpus de entrenamiento completo, con SVM.

Máquina de Soporte Vectorial Corpus de entrenamiento completo						
Kernel	Positivos	Tweets Neutrales	Negativos	Total de aciertos	Total de tweets	% de Precisión
Lineal	1929	2704	452	5085	8987	56.58
Polinomial grado 2, polinomial grado 3 y radial	0	3940	0	3940	8987	43.84

Tabla 6. Resultados de precisión, corpus de entrenamiento balanceado, con SVM.

Máquina de Soporte Vectorial Corpus de entrenamiento balanceado						
Kernel	Positivos	Tweets Neutrales	Negativos	Total de aciertos	Total de tweets	% de Precisión
Lineal	1865	1691	910	4466	8987	49.70
Polinomial de grado 2	210	1	1508	1719	8987	19.12
Polinomial de grado 3	3351	0	239	3590	8987	39.94
Radial	121	3882	67	4070	8987	45.28

En la Tabla 6 se muestran los resultados, empleando como clasificador una máquina de soporte vectorial con los *kernel* lineal, polinomial de grado 2, de grado 3

y radial, el corpus de entrenamiento utilizado fue balanceado. Contrario a las pruebas anteriores y a pesar de que no son resultados altos, la mayoría de estos predicen las tres clases, como lo es la SVM con *kernel* lineal, polinomial de grado 2 y radial.

Como puede apreciarse el resultado más alto obtenido por este modelo es de 56.58% de precisión empleando el corpus de entrenamiento completo y una máquina de soporte vectorial con *kernel* lineal.

4.2. Resultados del modelo sobre grafos

Para este modelo se realizaron 64 experimentos, incluyendo ambas variaciones del modelo, además de utilizar el vector de frecuencia y el de ocurrencia, también se probó con grafos de co – ocurrencia con ancho de ventana igual a dos y tres; se aplicaron los distintos algoritmos de centralidad; finalmente como en el modelo anterior se utilizó el corpus completo y el corpus balanceado para el entrenamiento. Cabe mencionar que en la mayoría de los experimentos con el corpus de entrenamiento completo sólo se predecía la clase neutral, por lo cual no era de utilidad. Para éste modelo, los mejores porcentajes se obtuvieron entrenando con el corpus de entrenamiento balanceado, contrario al modelo léxico – sintáctico.

Como se puede observar en la Fig. 5, los resultados más altos se obtienen utilizando el vector de ocurrencia, aplicando SVM de *kernel* lineal y centralidad de vector propio con un porcentaje de precisión de 47.34%; le sigue con vector de frecuencia, SVM de *kernel* lineal y centralidad de grado con 46.85%, finalmente con el vector de ocurrencia, SVM de *kernel* lineal y centralidad de intermediación con 46.68% de precisión.

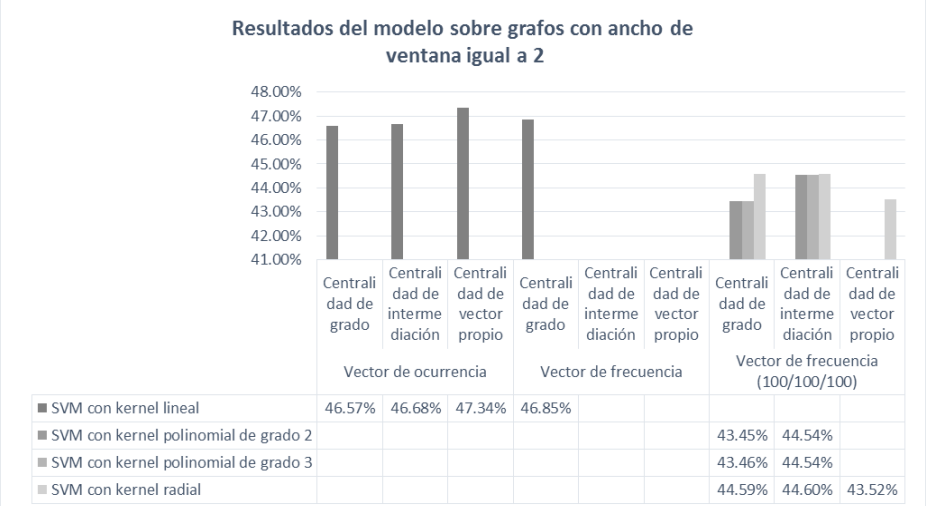

Fig. 5. Resultados del modelo sobre grafos con ancho de ventana igual a 2.

En la Fig. 6 se presentan los resultados con ancho de ventana igual a 3, con los vectores de ocurrencia (100/100/100), de frecuencia y de frecuencia (100/100/100), todos empleando el corpus de entrenamiento balanceado. El experimento con mayores

resultados fue el de vector de frecuencia (100/100/100). A continuación se enuncian los tres resultados que destacan de este conjunto de pruebas. Con vector de frecuencia, centralidad de grado y SVM de *kernel* polinomial de grado 2 y 3 se obtuvo un porcentaje de 45.64%; utilizando el vector de ocurrencia (100/100/100), SVM de *kernel* radial y centralidad de grado marco el porcentaje de 44.86%. Finalmente con el vector de frecuencia (100/100/100), centralidad de grado y SVM de *kernel* lineal fue de 44.78%.

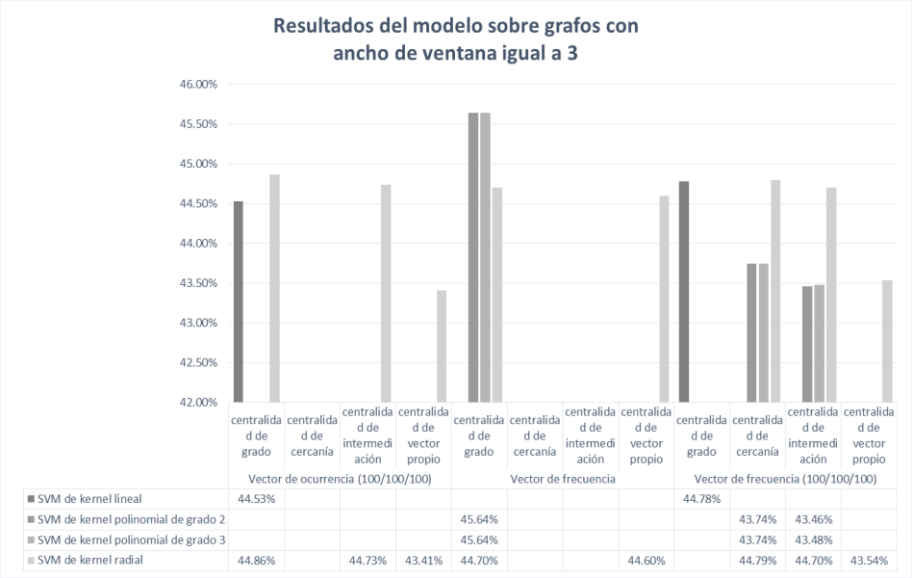

Fig. 6. Resultados del modelo sobre grafos con ancho de ventana igual a 2.

Como se puede apreciar el mejor resultado fue de 56.58% con el modelo léxico-sintáctico, en comparación con los resultados en el estado del arte, el rendimiento del modelo es bajo, pues en [1] se tiene un porcentaje de precisión 69.02%

5. Conclusiones y trabajo futuro

Se desarrollaron dos modelos de aprendizaje, uno utilizando características léxico-sintáctica y el otro extrayendo características a partir de la representación mediante grafos de los datos de entrenamiento. Además, se llevaron a cabo varios experimentos llegando a las siguientes conclusiones:

1. Es necesario balancear el corpus de entrenamiento para que ambos modelos logren descubrir las tres clases (negativo, positivo y neutro).
2. El modelo Léxico-Sintáctico arrojó mejores resultados.
3. El modelo desarrollado a partir de la selección de características utilizando los grafos de co-ocurrencias, no arrojó buenos resultados, se piensa que por

las características de los datos de entrenamiento y prueba, ya que muchos tweets son pequeñas oraciones y sin sentido.

4. El mejor comportamiento fue dado por la máquina de soporte vectorial.

Se planea seguir afinando los modelos ampliando los diccionarios de emoticones y siglas, además de crear un diccionario de *hashtags* en el que se incluya la polaridad de cada uno. También se planea la implementación de modelos que hagan uso de redes neuronales esperando un mejor resultado en la clasificación.

Referencias

1. Levallois, C.: Sentiment Analysis for Tweets based on Lexicons an Heuristics. http://www.cs.york.ac.uk/semeval-2013/accepted/27_Paper.pdf (2013)
2. Hangya, V., Berend, G., Farkas, R.: Sentiment Detection on Twitter Messages. http://www.cs.york.ac.uk/semeval-2013/accepted/102_Paper.pdf (2013)
3. Wilks, Y., Stevenson, M.: The Grammar of Sense: Using part-of-speech tags as a first step in semantic disambiguation. Journal of Natural Language Engineering, Vol. 4, No. 2, pp. 135–143 (1998)
4. Whitelaw, C., Garg, N., Argamon, S.: Using appraisal groups for sentiment analysis. In: Proceedings of the 14th ACM international conference on Information and knowledge management (CIKM'05), pp. 625–631 (2005)
5. Nasukawa, T., Yi, J.: Sentiment Analysis: Capturing Favorability Using Natural Language Processing. In: Proceedings of the 2nd international conference on Knowledge capture (K-CAP'03), pp. 70–77 (2003)
6. Serrano, J., Del Castillo, M.: Text Representation by a Computational Model of Reading. Neural Information Processing 13th International Conference, pp. 237–246 (2006)
7. Wrobel, S., Scheffer, T.: Text Classification beyond the Bag-of-Words Representation. (2002)
8. Sonawane, S., Kulkarni, P.: Graph based Representation and Analysis of Text Document: A Survey of Techniques. International Journal of Computer Applications, Vol. 96, No. 19, pp. 1–8 (2014)
9. Freeman, L.: Centrality in social networks: Conceptual clarification. Journal Social Networks – SOC NETWORKS, Vol. 1, No. 3, pp. 215–239 (1979)
10. Grafos. [Online]. http://www.ual.es/~btorreci/tr-grafos.pdf
11. Jimeng, S., Jie, T.: A survey of models and algorithms for social influence analysis. En Chary C. Aggarwal. Social Network Data Analytics, Springer, pp. 177–214 (2011)