Análisis de Sentimientos sobre un Corpus en Español: Experimentación con un Caso de Estudio

Luciana Dubiau, Juan M Ale

Facultad de Ingeniería, Universidad de Buenos Aires, Argentina
ldubiau@fi.uba.ar, ale@acm.org

Resumen. En este artículo se presenta la investigación, evaluación y comparación experimental de técnicas de procesamiento de lenguaje natural para análisis de información subjetiva como opiniones, sentimientos y emociones en textos no estructurados en idioma español. Se implementó una herramienta para la extracción de opiniones de documentos con el objetivo de clasificarlos según polaridad de sentimientos (positivos o negativos) utilizando como corpus de datos la base de comentarios de un sitio de crítica gastronómica al que se le aplican distintas técnicas de preprocesamiento. La principal contribución de este artículo es la experimentación y evaluación de técnicas de clasificación subjetiva de textos para el idioma español en función del tamaño de corpus, tipos de atributos extraídos y preprocesamientos aplicados.

Keywords: procesamiento de lenguaje natural, análisis de sentimientos, minería de opiniones, extracción de opiniones, análisis subjetivo, minería de sentimientos.

1 Introducción

Análisis de Sentimientos (AS), también conocido como Extracción de Opiniones, Minería de Opiniones, Minería de Sentimientos o Análisis Subjetivo se define como el estudio computacional de opiniones, sentimientos y emociones expresadas en textos [1].

El objetivo principal del análisis computacional de sentimientos consiste en determinar la actitud de un escritor ante determinados productos, situaciones, personas u organizaciones (target); identificar los aspectos que generan opinión (features); quién las posee (holder); y cuál es el tipo de emoción (me gusta, me encanta, lo valoro, lo odio) o su orientación semántica (positiva, negativa, neutra) [2].

El tipo de información que puede obtenerse utilizando sistemas de AS incluye: polaridad de sentimientos en críticas sobre arte, productos o servicios; nivel de fidelización de clientes; opinión pública sobre representantes políticos o situaciones de interés social; predicciones sobre resultados de elecciones; tendencias de mercado, etc.

Existen distintas tareas de procesamiento que pueden realizarse en sistemas de AS: la más simple es la clasificación binaria de la actitud de un texto, en positiva o negativa (también puede existir el neutro); una tarea un poco más compleja es la multiclasificación de un texto según el grado de polaridad de la actitud dentro de

una escala; y la tarea más avanzada es la identificación de los aspectos mencionados en un texto y sus sentimientos asociados.

En este trabajo nos enfocamos en la clasificación binaria de textos en idioma español utilizando métodos de aprendizaje supervisado y no supervisado y aportamos resultados basados en experiencias sobre la efectividad de algoritmos de clasificación que ya han sido utilizados con éxito en trabajos previos para el idioma inglés [3, 4].

Existen antecedentes de trabajos de clasificación subjetiva para el idioma español utilizando técnicas de machine learning [5, 6] donde se analizan distintos clasificadores y tipos de atributos. En esta experiencia trabajamos con los algoritmos de Naïve Bayes, Modelos de Máxima Entropía, Decision Trees, Support Vector Machines y una adaptación del algoritmo no supervisado de Turney para el idioma español, con el objetivo de comparar y evaluar performance en función de distintos preprocesamientos de textos propuestos, tipos de atributos extraídos y tamaños de corpus que van desde 500 hasta 22000 documentos.

El artículo está organizado de la siguiente manera: en la sección 2 introducimos brevemente las técnicas de clasificación y preprocesamiento de textos que se utilizan actualmente en tareas de AS; en la sección 3 describimos el software desarrollado y las herramientas utilizadas para realizar esta experiencia; en la sección 4 presentamos el caso de estudio, los criterios adoptados para la construcción del corpus de datos, los parámetros de las experiencias realizadas, los resultados obtenidos acompañados por gráficas comparativas y un análisis de los mismos; y por último en la sección 5 presentamos las conclusiones y el trabajo futuro.

2 Estado del Arte

En esta sección describimos las técnicas más utilizadas de clasificación subjetiva de textos y preprocesamientos que pueden aplicarse para mejorar los resultados del clasificador.

2.1 Técnicas de Clasificación Supervisada

Naïve Bayes es un método de clasificación supervisado y generativo que se basa en el teorema de Bayes y en la premisa de independencia de los atributos[1] para obtener la probabilidad de que un documento pertenezca a una determinada clase como se indica en la ecuación que sigue [7]:

$$P(C_i|D) \propto P(C_i) \prod_{k=1}^{n} P(f_k|C_i) \tag{1}$$

Donde f_k son los atributos del documento, C_i es la clase y $P(f_k|Ci)$ es la probabilidad de ocurrencia del atributo en la clase dada. La clase seleccionada por el clasificador sera la que maximice la probabilidad anterior.

Las implementaciones del algoritmo de Naïve Bayes difieren principalmente en la aproximación de $P(f_k|Ci)$ y las técnicas de smoothing[2] utilizadas para el tratamiento de probabilidades bajas o nulas.

[1] Esta premisa es conocida como *"Naïve Assumption"*.

[2] Se conoce como *smoothing* a la técnica para suavizar o unificar la distribución de probabilidad ajustando las bajas o nulas hacia arriba y las altas hacia abajo.

Modelo de Máxima Entropía (MaxEnt) es un método de clasificación discriminativo donde los documentos del conjunto de datos son descriptos a partir de una lista de atributos, siendo cada uno una restricción del modelo. Este método se basa en seleccionar la distribución de probabilidad que satisfaga todas las restricciones del modelo y maximice la entropía. Esto apunta a preservar la incertidumbre tanto como sea posible [8].

En modelos de máxima entropía la probabilidad de que un documento pertenezca a una clase se define como sigue:

$$P(c|x) = \frac{exp(\sum_{i=0}^{N} w_{ci} f_i)}{\sum_{c' \in C} exp(\sum_{i=0}^{N} w_{c'i} f_i)} \tag{2}$$

Donde, c es la clase que se desea evaluar, x es el documento, f_i es cada atributo, w_{ci} es el peso de ese atributo para la clase c que se está evaluando y $w_{c'i}$ es el peso del atributo en cada una de las posibles clases.

El cálculo de pesos es un problema complejo conocido como **convex optimization** [9] que busca maximizar la verosimilitud del modelo.

Support Vector Machines es un método supervisado de clasificación binaria en el cual el entrenamiento consiste en encontrar un hiperplano[3] que separe los vectores de atributos que representan los documentos del conjunto de datos en dos grupos, siendo esta separación la más grande posible. Aquellos vectores que definen los márgenes de la máxima separación entre las clases se conocen como *support vectors* [10].

Para la predicción de la clase utilizando este modelo se define la ecuación que sigue:

$$f(\boldsymbol{x}) = sign(\sum_{i} \alpha_i \boldsymbol{x_i} \cdot \boldsymbol{x} + b) \tag{3}$$

Siendo, x el vector de atributos del documento a clasificar, α_i cada uno de los pesos que ponderan los vectores de atributos identificados como *support features*, x_i cada uno de los *support features* y b el término independiente. Un valor de -1 indicará que el documento pertenece a una clase y un valor de $+1$ a la otra, lo que representa de qué lado del hiperplano se encuentra $\boldsymbol{x}$.

Decision Trees es un método de clasificación supervisado en el que el entrenamiento consiste en la construcción de un árbol de decisión de múltiples caminos en el que para cada nodo se busca el atributo que provee mayor ganancia de información para la clase [11]. El árbol crece hasta su tamaño máximo y luego es acotado para mejorar su capacidad de generalización para los datos que no ocurren en el conjunto de datos de entrenamiento. A partir de este modelo se infieren reglas de decisión sobre las cuales se basa la clasificación.

En este trabajo utilizamos una implementación optimizada del algoritmo de CART[4] basado en árboles de decisión.

[3] En geometría un hiperplano es una división del espacio en dos partes. En $\mathbb{R}^1$, será un punto que divide una recta; en $\mathbb{R}^2$ será una recta que divide un plano; en $\mathbb{R}^3$ será un plano que divide el espacio. Esto mismo puede ser generalizado para espacios de más dimensiones.

[4] Classification And Regression Trees

2.2 Técnicas de Clasificación No Supervisada

Polaridad de Turney es un algoritmo no supervisado de clasificación subjetiva de textos que se basa en predecir la polaridad de un documento a partir de su orientación semántica [4].

Este método consiste en extraer de los documentos aquellos bigramas que cumplen con determinados patrones de opinión y luego calcular la orientación semántica a partir de la distancia de cada bigrama a los términos del idioma inglés, *"excelent"* y *"poor"* que sugieren referencias positivas y negativas respectivamente según la siguiente ecuación:

$$SO(phrase) = log_2[\frac{hits(phrase\ NEAR\ "excellent")hits("poor")}{hits(phrase\ NEAR\ "poor")hits("excellent")}] \quad (4)$$

2.3 Preprocesamiento de Textos

Con el objetivo de mejorar los resultados del clasificador y dependiendo de la tarea de procesamiento de texto que se esté realizando y el idioma en el que esté escrito el texto, pueden requerirse algunas de las siguientes transformaciones en el corpus de datos de entrada antes de su procesamiento.

Normalización: Consiste en unificar términos que representan la misma información y pueden ser escritos en distintos formatos. Por ejemplo, *"restaurante", "restaurant", "restorán", "restó".*

Tokenización: Separación de oraciones y palabras de un documento a partir de *tokens*, o caracteres especiales, que indican el fin de una oración o palabra y el comienzo de la que sigue.

Stemming: Los algoritmos de Stemming (o Stemmers) permiten obtener la raíz o *stem* de una palabra eliminando terminaciones, con el objetivo de unificar aquellos términos que aportan la misma información al clasificador. Por ejemplo, los términos "recomendable, recomendamos, recomendar, recomendación" son reemplazados por su stem, "recomend". El stem de una palabra no necesariamente será un término válido del vocabulario.

Lematización: Otra forma de unificar los términos que aportan la misma información al clasificador es reemplazando cada palabra por su *lema*. El lema de una palabra es un término válido del vocabulario que por convención es la representación de todas las formas flexionadas de la palabra, es decir, para hallar el lema de un término se eliminan todas las flexiones (conjugaciones, grado, persona, género, número, etc). Por ejemplo, lema("pésimo") = "malo"; lema("empieza") = "empezar"; lema("primeras") = "primero"; lema("estas") = "este".

Tratamiento de Negaciones: Con el objetivo de distinguir aquellos términos que siguen a una negación suele ser de utilidad indicar que la palabra aparece negada agregando un prefijo a todos los términos que siguen a la negación hasta el siguiente signo de puntuación.

Otros Preprocesamientos: Otras transformaciones que pueden representar una mejora en la efectividad del clasificador incluyen: corregir errores ortográficos; agregar a cada término la etiqueta gramatical para eliminar ambigüedades[5];

[5] La tarea de asignar etiquetas gramaticales a un texto es conocida como *Part Of Speech (POS) Tagging.*

eliminar términos o secuencias de palabras que no aporten información a la tarea de procesamiento que se está realizando (signos de puntuación, fechas, números, capitalizaciones[6], caracteres especiales, emoticones, caracteres repetidos más de dos veces, stopwords[7], palabras de menos de N letras, acentos, etc).

3 Implementación y Herramientas

Para analizar, comparar y evaluar la efectividad de las técnicas investigadas se implementó una herramienta en lenguaje Python[8] para clasificación automática de textos según polaridad de sentimientos que se ejecuta en función de los siguientes parámetros:

- Tamaño y cantidad de folds: se implementó el algoritmo de k-fold cross validation para métodos supervisados.
- Algoritmo de clasificación: Naïve Bayes, MaxEnt, SVM, Decision Trees, Adaptación del Algoritmo de Turney.
- Tipo de atributo a utilizar para el entrenamiento en métodos supervisados: presencia de unigramas, frecuencia de unigramas, presencia de bigramas y presencia de adjetivos.
- Preprocesamientos: se implementaron preprocesadores para remover stop words, filtrar palabras con mínimo de longitud parametrizable, remover caracteres duplicados, preprocesar negaciones, stemming, lematización, transformar caracteres a minúscula, remover signos de puntuación, remover acentos, o una combinación de ellos.

Para la implementación de preprocesamientos y *features extractors*[9] se utilizaron herramientas provistas por los frameworks NLTK[12] y Freeling [13].

Para la implementación del algoritmo de Naïve Bayes se utilizó el framework NLTK[12]; para el cálculo de pesos en el algoritmo de Máxima Entropía se utilizó la herramienta Megam[14]; y para la clasificación de textos utilizando los métodos SVM y Decisions Trees se utilizaron las implementaciones provistas por el framework sci-kit learn[15].

Para adaptar el clasificador de Turney al idioma español se tuvieron las siguientes consideraciones:

- El operador *NEAR* se definió como la ocurrencia conjunta de los términos en la misma oración.
- Para la extracción de bigramas se utilizaron los siguientes patrones de opinión definidos en trabajos previos para el idioma español [6]:
  - Adjetivo + Nombre
  - Nombre + Adjetivo + (No_Nombre)
  - Adverbio + Adjetivo + (No_Nombre)

[6] En algunas tareas de clasificación mantener las capitalizaciones puede resultar útil para identificar nombres propios.

[7] Se consideran stopwords aquellas palabras que no agregan información al texto que se está procesando. Por ejemplo, en español: de, la, el, que, a, etc.

[8] La herramienta desarrollada se encuentra disponible en el siguiente repositorio: `https://github.com/ldubiau/sentiment_classifier`

[9] Extractores de atributos del conjunto de datos.

  - Adverbio + Verbo
  - Verbo + Adverbio
- Los términos utilizados para representar polaridad fueron:
  - Positivos: *excelente, excelentes, bueno/a, buenos/as, buenisimo/a, buenisimos/as, rico/a, ricos/as, espectacular, genial.*
  - Negativos: *mal, malo/a, malos/as, feo/a, feos/as, horrible, horribles, pesimo/a, pesimos/as, desastre, mediocre.*

4 Caso de Estudio y Experimentación

El caso de estudio elegido es el sitio de crítica gastronómica online `www.guiaoleo.com`. En este sitio los usuarios emiten opiniones sobre restaurantes y proveen una calificación en las categorías comida, ambiente y servicio asignado puntajes del 1 al 4 (malo/regular, bueno, muy bueno o excelente respectivamente).

4.1 Construcción del Corpus de Datos

Para la construcción del corpus de datos se extrajeron los comentarios junto con el puntaje asignado por el usuario en cada categoría y se asignaron etiquetas a los documentos utilizando siguiente criterio: se asignó la etiqueta "POSITIVO" a aquellos comentarios cuya suma de puntos en las categorías mencionadas sea 10 o superior. Esto significa que seleccionamos aquellos comentarios que tienen 4 puntos (excelente) en al menos una categoría y 3 puntos (muy bueno) o más en las restantes; se asignó la etiqueta "NEGATIVO" a aquellos comentarios cuyo puntaje en la categoría "comida" (identificada como la más relevante para el dominio estudiado) sea 1 (malo/regular) o bien sea 2 (bueno) y que el resto de las categorías tengan ambas puntaje 1 (malo/regular); el resto de los comentarios no se incluyeron en el corpus de datos[10].

4.2 Experimentación

En esta sección presentamos los resultados obtenidos a partir de la clasificación del corpus de datos descripto en la sección 4.1 en función de los preprocesos aplicados, el algoritmo de clasificación, los distintos tamaños de corpus y los atributos extraídos. Para calcular la efectividad de los clasificadores nos basamos en la medida de *accuracy* que se obtiene a partir de la relación entre la cantidad de casos clasificados correctamente y el total de casos.

Todas las experiencias presentadas en esta sección fueron ejecutadas utilizando un corpus balanceado, el método de validación 5-fold cross validation y un tamaño de corpus entre 500 y 22000 documentos del conjunto de datos[11]. Teniendo en

[10] El dataset final se encuentra disponible en la siguiente url: `http://www.fi.uba.ar/~ldubiau/datasets/restaurante-review-dataset.zip` e incluye un total de 34808 comentarios positivos y 16912 negativos agrupados dentro de los directorios "pos" y "neg" respectivamente.

[11] Este tamaño máximo fue seleccionado en base al costo de entrenamiento de los métodos supervisados y teniendo en cuenta que la mejora obtenida para corpus mayores no se consideró representativa.

cuenta el costo de entrenamiento de los métodos supervisados se adoptó el siguiente criterio para la selección de atributos: para unigramas se extrajeron aquellos cuya frecuencia de aparición es mayor a 10; en el caso de adjetivos y bigramas la frecuencia mínima requerida fue de 4 y en todos los casos se utilizó un máximo de 3000 atributos.

En la tabla 1 podemos observar una muestra de los documentos clasificados con los algoritmos en estudio utilizando el tamaño máximo de corpus y unigramas como atributos para los métodos supervisados.

Tabla 1. Ejemplos de Documentos Clasificados

Documento	**Clase**	**Clase Asignada**				
		NB	ME	SVM	DT	Turney
"Muy buen ambiente, buena música. La atención es rápida. Las pizzas son muy ricas, hay muchas variedades y sus precios no son desorbitados."	POS	POS	POS	POS	POS	POS
"Increible, no se conoce baires ni la pizza sino comiste en el cuartito...una noche especial para disfrutar con amigos..."	POS	NEG	POS	POS	POS	POS
"Atendido por sus dueños, ambiente y atención muy agradable. Calidad 10 puntos, nada que envidiarle a los de Puerto Madero."	POS	POS	POS	POS	POS	NEG
"Es muy caro para lo poco que sirven y tampoco es de tan buena calidad. La atencion es buena, pero no profesional"	NEG	POS	NEG	NEG	NEG	POS
"Comida impresentable"	NEG	POS	POS	POS	POS	NEG
"Todo impecable, lindo, fashion y glamuroso. La comida? En la Farola de Belgrano se come mejor. Creo que para dar de comer tan mal hay que poner ganas y esfuerzo"	NEG	POS	NEG	NEG	POS	NEG

Para definir los preprocesamientos más efectivos ejecutamos la clasificación aplicando cada preprocesamiento en forma aislada y en conjunto para analizar el impacto en los resultados. Esta experiencia se ejecutó con los siguientes parámetros: algoritmo de Naïve Bayes, unigramas como atributos y tamaño máximo de corpus. En la tabla 2 se observan los valores de accuracy y porcentajes de mejora obtenidos aplicando cada preprocesamiento y combinaciones de ellos. Para las experiencias que siguen utilizamos la combinación de preprocesadores que representa el porcentaje de mejora más alto según se observa en la figura 1, es decir, eliminación de stopwords, tratamiento de negaciones, filtrado por tamaño de palabra, eliminación de signos de puntuación, reemplazo de caracteres especiales y transformación a minúsculas.

En las figuras 2 y 3 observamos los valores de accuracy obtenidos para cada clasificador supervisado, en función del tamaño de corpus y atributos extraídos, presentados en gráficas por atributo y por algoritmo respectivamente.

Por último en la figura 4 comparamos los valores máximos de accuracy alcanzados por el clasificador de Turney y por los clasificadores supervisados para cada tipo de atributo, tamaño máximo de corpus y siguiendo el mismo criterio ya explicado para selección de atributos y preprocesamientos.

Tabla 2. Valores de Accuracy obtenidos utilizando Naïve Bayes, Unigramas, Tamaño máximo de Corpus y distintos Preprocesamientos

Preproceso	**Accuracy**	**Mejora (%)**
NP (Sin Preproceso)	0.868	
SW (Eliminación de stopwords)	0.895	3.11%
NEG (Tratamiento de negaciones)	0.875	0.81%
WL (Filtrado de palabras de menos de 3 caracteres)	0.883	1.73%
DC (Eliminación de caracteres repetidos más de 2 veces)	0.867	-0.12%
STEM (Stemming)	0.861	-0.81%
LC (Transformación de capitalizaciones)	0.868	0.00%
PUNCT (Eliminación de signos de puntuación)	0.871	0.35%
SC (Transformación de caracteres especiales)	0.869	0.12%
LEMMA (Lematización)	0.867	-0.12%
Combinación 1: SW + NEG	0.905	4.26%
Combinación 2: SW + NEG + WL	0.911	4.95%
Combinación 3: SW + NEG + WL + PUNCT	0.914	5.3%
Combinación 4: SW + NEG + WL + PUNCT + SC	0.918	5.76%
Combinación 5: SW + NEG + WL + PUNCT + SC + LC	**0.924**	**6.45%**
Combinación 6: SW + NEG + WL + PUNCT + SC + LC + LEMMA	0.92	5.99%
Combinación 7: SW + NEG + WL + PUNCT + SC + LC + DC	0.924	6.45%
Combinación 8: SW + NEG + WL + PUNCT + SC + LC + STEM	0.918	5.76%

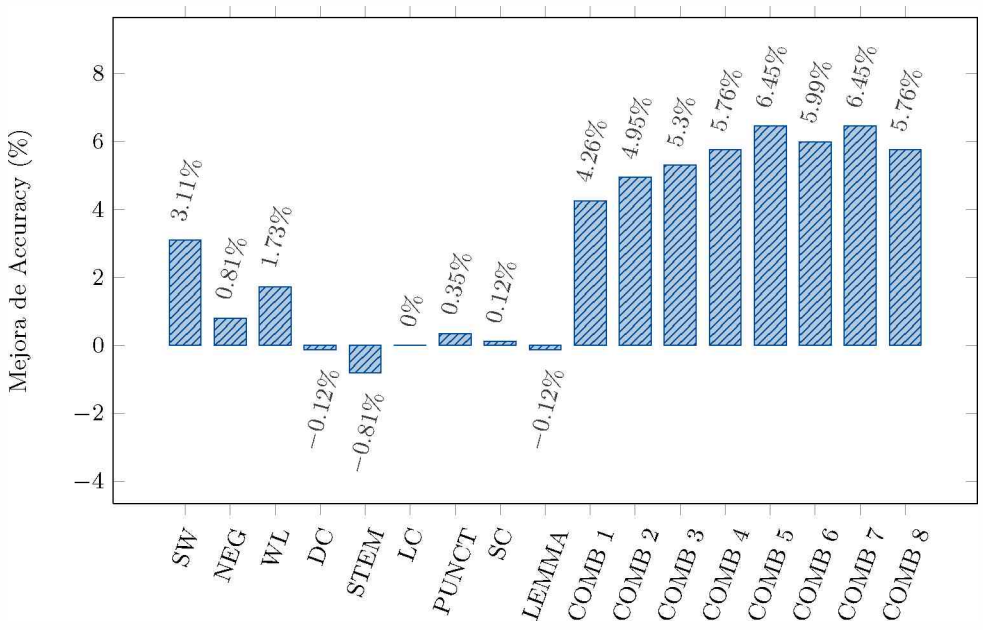

Fig. 1. Mejora de Accuracy (%) vs Preprocesos utilizando Naïve Bayes, Unigramas y Tamaño máximo de Corpus

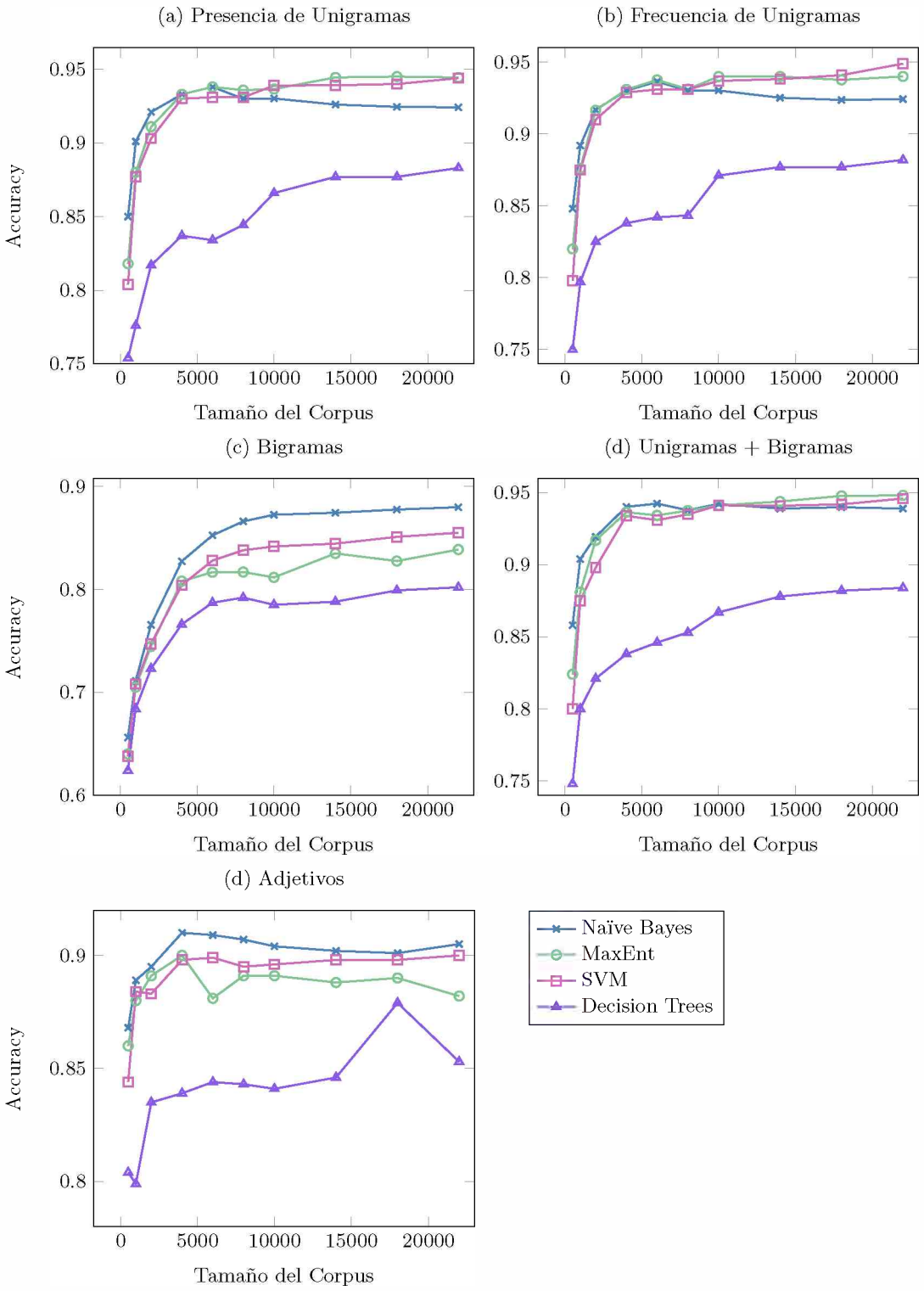

Fig. 2. Efectividad de Clasificadores por Atributo: Accuracy vs Tamaño de Corpus

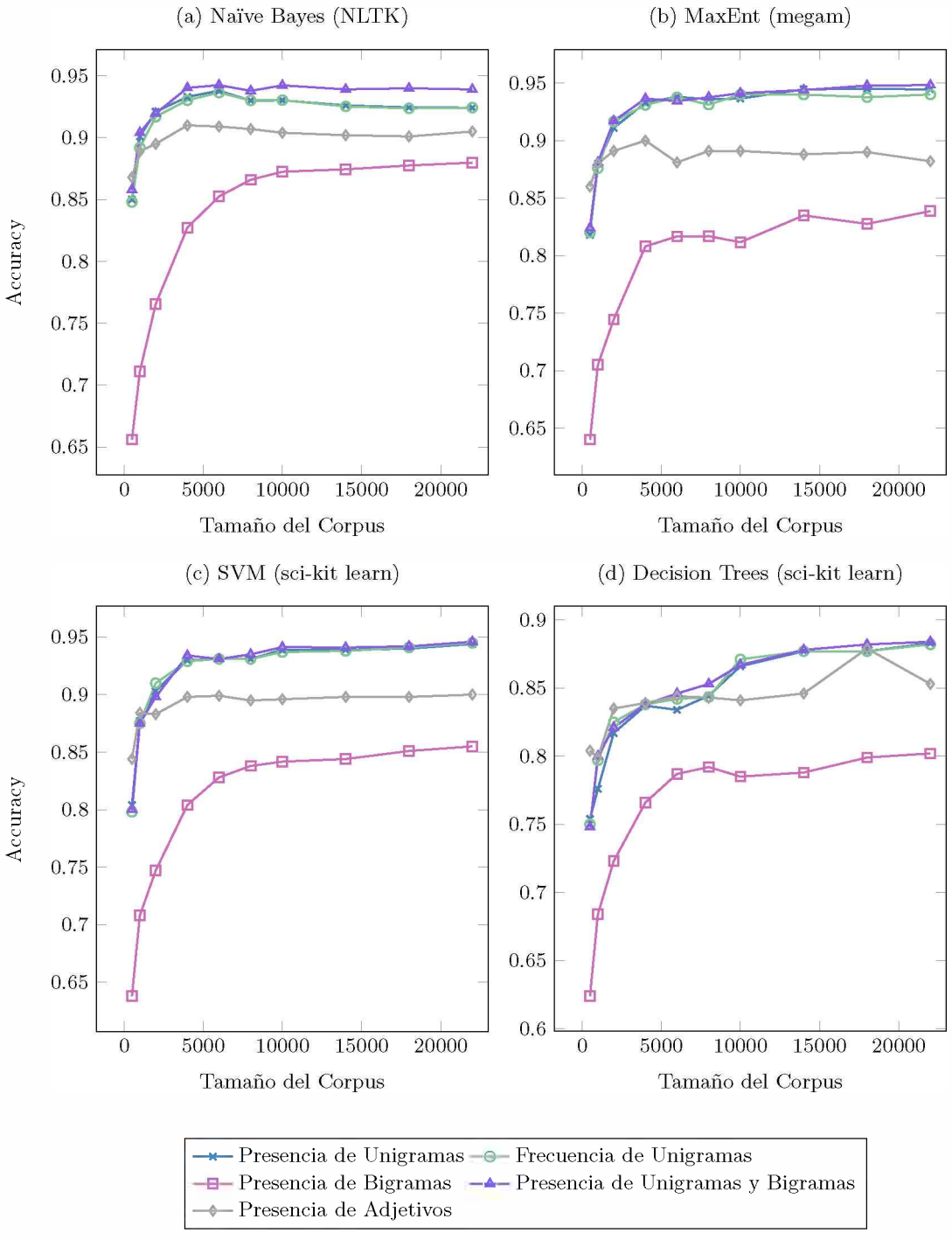

Fig. 3. Efectividad de Clasificadores por Algoritmo: Accuracy vs Tamaño de Corpus

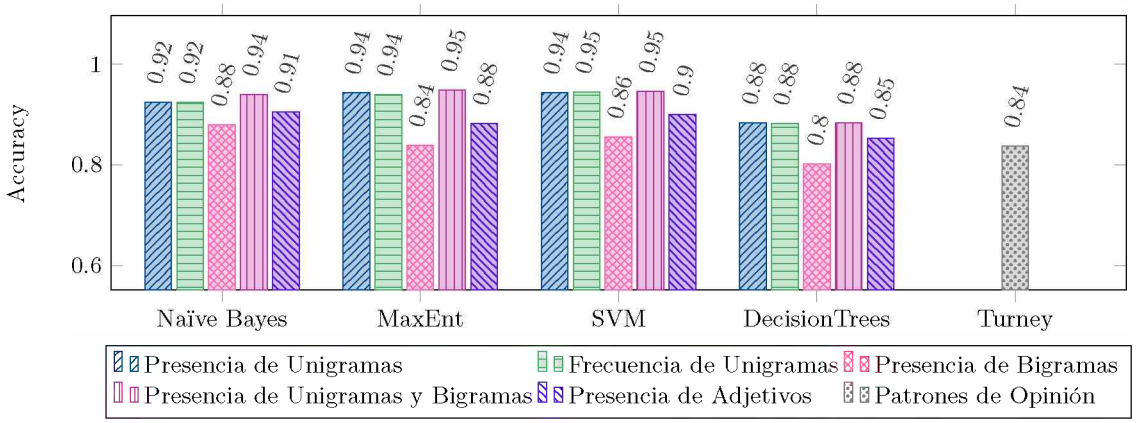

Fig. 4. Efectividad de Clasificadores para Máximo Tamaño de Corpus: Algoritmo vs Accuracy

4.3 Análisis de Resultados

La primera observación que hacemos es que hay preprocesamientos que aplicados en forma aislada no representan una mejora pero sí lo hacen cuando son aplicados en combinación con otros, como ocurre en el caso de transformación a minúscula.

En cuanto a los algoritmos de clasificación observamos que con Naïve Bayes se obtienen los mejores resultados para corpus pequeños pero su performance decrece levemente para los tamaños de corpus más grandes en comparación con MaxEnt y SVM donde los resultados mejoran a medida que crece el tamaño de corpus y se obtiene la máxima performance alcanzada por la experiencia. También observamos que si se utilizan adjetivos o bigramas como atributos Naïve Bayes presenta los mejores resultados. En el caso del clasificador basado en árboles de decisión, la performance obtenida es notablemente peor que en los otros clasificadores analizados.

En la figura 3 se observa que para todos los clasificadores supervisados los mejores resultados se obtienen utilizando como atributos la combinación de presencia de unigramas y bigramas aún cuando para todos los clasificadores la performance al utilizar sólo bigramas no resulta favorable. Así también comprobamos, como ya se ha mostrado para el inglés en trabajos anteriores [3], que en términos de información subjetiva considerar frecuencia de unigramas como atributos no representa una mejora notable con respecto a presencia.

En la figura 4 podemos observar la comparación de métodos supervisados con el algoritmo de Turney, que si bien no alcanza la efectividad máxima de éstos para presencia de unigramas, se obtienen muy buenos resultados comparables con los obtenidos utilizando bigramas como atributos en métodos supervisados, con la ventaja de que no requiere un corpus de entrenamiento.

5 Conclusiones y Trabajo Futuro

En este artículo presentamos un caso de estudio y una extensiva experimentación en tareas de análisis de sentimientos sobre un corpus en idioma español, utilizando

diversas técnicas de preprocesamiento y una variedad de métodos supervisados y no supervisados de clasificación de textos.

Este estudio muestra el impacto en la performance de los clasificadores ante la variación de parámetros de entrada como transformaciones de textos, tipos de atributos extraídos y tamaño de corpus utilizado. Hallamos que la aplicación de preprocesamientos influye considerablemente en los resultados del clasificador y que la máxima precisión se obtiene utilizando MaxEnt y SVM para corpus grandes y Naïve Bayes para corpus más pequeños.

En trabajos futuros planeamos evaluar el comportamiento de los clasificadores para otros dominios utilizando como entrenamiento el corpus de datos propuesto en este trabajo; realizar experiencias utilizando corpus desbalanceados; y analizar otros preprocesamientos y tipos de atributos que permitan mejorar los resultados y generalizar los clasificadores.

Referencias

1. Bo Pang and Lilian Lee: Opinion Mining and Sentiment Analysis. Department of Computer Science, Cornell University (2008)
2. Bing Liu: Sentiment Analysis and Subjectivity. Department of Computer Science, University of Illinois at Chicago (2010)
3. Bo Pang, Lillian Lee and Shivakumar Vaithyanathan: Thumbs up? Sentiment classification using machine learning techniques. Department of Computer Science, Cornell University (2002)
4. Peter Turney: Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews. Institute for Information Technology, National Research Council of Canada (2002)
5. Grigori Sidorov, Sabino Miranda-Jiménez, Francisco Viveros-Jiménez, Alexander Gelbukh, Noé Castro-Sánchez, Francisco Velásquez, Ismael Díaz-Rangel, Sergio Suárez-Guerra, Alejandro Treviño, and Juan Gordon: Empirical Study of Machine Learning Based Approach for Opinion Mining in Tweets. LNAI 7629 (2012)
6. Fermín L. Cruz, Jose A. Troyano, Fernando Enriquez, Javier Ortega: Clasificación de documentos basada en la opinión: experimentos con un corpus de críticas de cine en español, RUA. Repositorio Institucional de la Universidad de Alicante (2008)
7. Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze: Introduction to Information Retrieval. Cambridge University Press (2008)
8. Christopher D. Manning and Hinrich Schütze: Foundations of Statistical Natural Language Processing. MIT Press, Cambridge (1999)
9. Daniel Jurafsky and James H. Martin: Speech And Language Processing. An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. Prentice Hall (2009)
10. Cortes, C. and Vapnik, V.: Support-vector network. Machine Learning, 273-297 (1995)
11. Alexander Clark, Chris Fox and Shalom Lappin: The handbook of computational linguistics and natural language processing. Wiley-Blackwell (2010)
12. Steven Bird, Edward Loper and Ewan Klein: Natural Language Processing with Python. O'Reilly Media Inc (2009)
13. Lluís Padró and Evgeny Stanilovsky: FreeLing 3.0: Towards Wider Multilinguality, Proceedings of the Language Resources and Evaluation Conference (LREC 2012). ELRA (2012)
14. Hal Daumé III: Notes on CG and LM-BFGS Optimization of Logistic Regression (2004)
15. Pedregosa, F. and Varoquaux, G. and Gramfort, A. and Michel, V. and Thirion, B. and Grisel, O. and Blondel, M. and Prettenhofer, P. and Weiss, R. and Dubourg, V. and Vanderplas, J. and Passos, A. and Cournapeau, D. and Brucher, M. and Perrot, M. and Duchesnay, E.: Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12:2825–2830 (2011)