ISSN 1870-4069

Clasificación bi-clase de canciones infantiles aplicando inteligencia artificial y procesamiento de lenguaje natural

David Soto Osorio[1], Jesús Jaime Moreno Escobar[1],
Liliana Chanona-Hernández[1], Grigori Sidorov[2],
César Jesús Núñez-Prado[1]

[1] Instituto Politécnico Nacional,
Escuela Superior de Ingeniería Mecánica y Eléctrica,
México

[2] Instituto Politécnico Nacional,
Centro de Investigación en Computación,
México

davidsotoesime@outlook.com, jmorenoe@ipn.mx
{lchanona, cesar.jnprado}@gmail.com, sidorov@cic.ipn.mx

Resumen. Desde hace muchos años, las canciones infantiles han formado parte del crecimiento de los seres humanos, ya que ofrecen grandes beneficios tales como; el desarrollo de la inteligencia, la enseñanza de nuevos valores y buenos hábitos; pero es un hecho, que del mismo modo en que las canciones dirigidas a los niños pueden tener una influencia positiva, también lo pueden hacer de manera negativa. Por esta causa, en el presente trabajo se propone un algoritmo de clasificación bi-clase que emplea técnicas de procesamiento de lenguaje natural y el modelo de inteligencia artificial K vecinos más cercanos, para clasificar canciones infantiles en positivas y negativas.

Palabras clave: Canciones infantiles, inteligencia artificial, procesamiento de lenguaje natural, clasificación bi-clase.

Bi-Class Analysis and Classification of Songs for Children Applying Natural Language Processing and Artificial Intelligence

Abstract. For many years, children's songs have been part of the growth of human beings, since they offer great benefits such as; the development of intelligence, the teaching of new values and the learning of good habits; but it is

a fact, that in the same way that songs aimed at children can influence them in a positive way, they can also influence them in a negative way. For this reason, in the present work a bi-class classification algorithm is proposed that uses natural language processing techniques and the K nearest neighbors' artificial intelligence model to classify children's songs into positive and negative.

Keywords: Children's songs, artificial intelligence, natural language processing and bi-class classification.

1. Introducción

La música es considerada como un elemento cultural que fortalece el aprendizaje y la memoria de los seres humanos; inclusive cuando se trate de géneros diferentes. A partir del siglo XX, el género de canciones infantiles comenzó a ser considerado como una parte muy importante en el crecimiento del ser humano, ya que cuenta con grandes beneficios como son; el desarrollo de la inteligencia, la enseñanza de nuevos valores y permite que los niños aprendan buenos hábitos para su futuro.

En el siglo XXI el desarrollo tecnológico ha permitido que las personas puedan estar conectadas entre sí a través de dispositivos electrónicos, lo cual facilita la búsqueda de información y la interacción entre los seres humanos, sin embargo; las nuevas generaciones pueden aprender cosas tanto positivas como negativas y esto los convierte en una generación vulnerable, que puede ser influenciada por información errónea o por géneros musicales que no son aptos para su edad o inclusive pueden tener una mala influencia de aquellas canciones que son consideradas infantiles pero que transmiten un mensaje que puede perjudicar en su comportamiento o vocabulario.

Por lo anteriormente descrito, en este trabajo se propone desarrollar un corpus lingüístico digital con canciones dirigidas hacia niños y aplicar un algoritmo de aprendizaje supervisado basado en técnicas de procesamiento de lenguaje natural y el modelo k vecinos más cercanos para realizar una clasificación bi-clase de letras de canciones infantiles.

2. Trabajos relacionados

Los trabajos seleccionados en nuestro estado del arte realizan un análisis de letras de canciones de diferentes géneros musicales por medio de técnicas de procesamiento de lenguaje natural y algoritmos de inteligencia artificial.

En [1] se menciona la necesidad de la industria musical para mejorar la experiencia que tienen sus usuarios, es por ello que aplican un algoritmo de clasificación probabilístico clásico conocido como *Naive Bayes*, este algoritmo se implementó utilizando el módulo *AI de Perl*. Utilizan una clasificación de 5 diferentes clases, el etiquetado se realizó manualmente y se aplicó un método de validación cruzada para separar el corpus lingüístico digital en conjuntos de entrenamiento y prueba, y obtuvieron un resultado del 82 % de precisión.

Por otra parte, en [2] aplican un algoritmo de aprendizaje supervisado con el que se efectúa la clasificación de sentimientos en las canciones hindi en 5 clases; para la generación del corpus lingüístico usaron diferentes fuentes de internet, se hizo una asignación de clases de forma manual, para este caso; solicitaron el apoyo de dos estudiantes a los que se les asignó la tarea de leer la canción y clasificarla. Una vez que se contó con el corpus lingüístico etiquetado se aplicó la técnica de preprocesamiento de eliminación de *stop-words*. Se realizó la separación en conjunto de entrenamiento y prueba aplicando un método de validación cruzada de 10 iteraciones.

Para la clasificación de las letras se usó el módulo *LibSVM* de la herramienta *WEKA*[1], al probar el algoritmo se logró obtener un porcentaje de precisión del 38.49 %, y concluyeron que este porcentaje se debe a que no consideraron varias características textuales es por ello que como propuesta a futuro buscan aumentar las características textuales para incrementar el porcentaje de precisión. En [3] aplican una clasificación bi-clase de canciones de diferentes géneros musicales.

Para la generación del corpus lingüístico utilizaron la lista de *Jamrock Entertainment* de las 100 mejores canciones por año. Se generó un corpus de 420 letras, de las cuales 210 letras son positivas y 210 letras son negativas, para realizar la clasificación se usó el algoritmo *Present In One*. En este algoritmo realizan dos conteos, el primero almacena el número de palabras que son consideradas como positivas mientras que el segundo almacena el número de palabras que son clasificadas como negativas, al final dependiendo del conteo mayor, se asigna la clase a la letra, como resultado de esta metodología obtuvieron un porcentaje de precisión del 66%.

En [4] se realiza un análisis de un corpus lingüístico conformado por letras de canciones de artistas que se han suicidado, este trabajo tiene como propósito identificar tendencias suicidas por medio de las letras escritas por algunos autores o artistas. Se generó un corpus lingüístico de 533 canciones de las cuales 253 letras fueron escritas por 4 artistas que no han tenido tendencias suicidas y 280 escritas por 5 artistas que cometieron suicidio.

Para realizar la clasificación se realizó la separación del corpus en conjunto de prueba y conjunto de entrenamiento, el conjunto de prueba estuvo conformado por 63 letras de artistas que se suicidaron y 46 letras de artistas que no han presentado tendencias suicidas. Sobre el corpus creado se aplicaron técnicas de preprocesamiento, tales como; tokenización y lematización usando la librería *OpenNLP*[2].

Para el algoritmo de clasificación se usó la librería *WEKA* y se ejecutaron diferentes algoritmos de clasificación y se determinó que el algoritmo *SimpleCart* obtiene el mayor porcentaje de precisión con un 70.6%, concluyen que el procesamiento de lenguaje natural puede ser una herramienta muy poderosa para realizar diferentes análisis dentro de las letras.

Por último, en el trabajo [5] plantean el hecho de que la música forma parte de la vida diaria de todos los seres humanos y el impacto que ésta puede producir en ello y es por ello que realizan un análisis de letras de canciones en idioma tailandés. Para la clasificación consideran sólo dos clases (feliz y triste), para generar el corpus

[1] https://www.cs.waikato.ac.nz/ml/weka/

[2] https://opennlp.apache.org/

lingüístico consultaron la página de *Chord Café*[3] y obtuvieron un corpus lingüístico de 427 canciones felices y 317 canciones tristes. Para el preprocesamiento se usó la segmentación de las letras y para el clasificador se consideró una red neuronal multicapa, y con ello obtuvieron un 68 % de precisión.

3. Desarrollo del proyecto

Para el desarrollo del algoritmo de clasificación bi-clase se consideran un conjunto de etapas que tienen como propósito pre-procesar las letras de canciones infantiles para su posterior clasificación en clase positiva y negativa. A continuación, se explican las etapas que se llevaron a cabo para la clasificación.

3.1. Corpus lingüístico digital

Un corpus lingüístico es un conjunto de textos que tienen un mismo origen y que tiene por objetivo almacenar un conjunto de documentos o textos con el fin de usar estos datos para realizar un análisis. Para el desarrollo del corpus lingüístico digital del presente trabajo, se consultaron diferentes páginas web en las que se presentan letras de canciones infantiles de diferentes artistas, películas y programas de televisión. El corpus lingüístico integra canciones en español de la década de los cincuenta hasta la actualidad. Todas las letras descargadas fueron almacenadas en un solo directorio en texto plano con una numeración consecutiva. Como resultado de esta metodología de búsqueda se generó un corpus lingüístico de 220 letras[4].

3.2. Etiquetado del corpus lingüístico

Se decidió realizar el etiquetado del corpus lingüístico digital sólo con dos posibles clases: positivo y negativo. Para realizar esta tarea, se solicitó el apoyo de 5 profesionales en el campo de la psicología infantil de las cuales son todas mujeres con un rango de edad de 28 a 40 años y una experiencia laboral de 5 a 10 años. La tarea asignada a las psicólogas fue determinar si la canción transmite un mensaje positivo o negativo a los niños con base en su propia experiencia profesional y que justificara la clasificación redactando un pequeño párrafo.

Para realizar la asignación de la clase final de cada una de las letras, se consideró la siguiente métrica: Si por lo menos 3 especialistas clasificaron como positiva la misma canción, a ésta se le asignaría la clase positiva, en caso contrario se clasificaría como negativa. Se consideraron como letras negativas aquellas que transmiten un mensaje con lenguaje inapropiado o que difunde el miedo hacia algunos personajes o también incentiva el machismo, la desigualdad y el racismo. Al terminar la asignación de clases se tiene que de las 220 letras que forman el corpus lingüístico digital a 172 canciones se les asignó la clase positiva y 48 fueron consideradas negativas.

[3] https://en.chordcafe.com/

[4] Proximamente estará disponible en: https://www.cic.ipn.mx/~sidorov/

Fig. 1 Tokenización.

Fig. 2 Eliminación de Stop-words.

3.3. Preprocesamiento del corpus lingüístico

En esta etapa, se aplicaron técnicas de preprocesamiento sobre el corpus lingüístico con el propósito de reducir el tiempo de procesamiento y aumentar el porcentaje de precisión del algoritmo de clasificación. Para aplicar estas técnicas se utilizó el lenguaje de programación *Python* junto con las librerías *NLTK, Pandas y Spacy.*

De manera general, la tokenización consiste en separar las oraciones en palabras, a estas palabras se les conoce como tokens. En la Figura 1 se puede observar la frase "La niña está en su cama de madera", al aplicar la tokenización se tiene el siguiente resultado.

Una vez que se realizó la tokenización se procede a realizar la eliminación de *stop-words*. Se consideran como *stop-words* a todas aquellas palabras que aparecen con una frecuencia muy alta en los textos.

La eliminación de *stop-word* consiste en eliminar todas aquellas palabras que no se consideran para el análisis del texto, como los conectores, artículos, pronombres, preposiciones o conjunciones; para aplicar la eliminación de stop-words se utilizaron los diccionarios de las librerías *NLTK (Natural Language Toolkit)* y *Spacy*.

En la Figura 2 se puede observar la frase "La niña está en su cama de madera", los recuadros verdes indican que pertenecen al conjunto de stop-words y los recuadros amarillos indican que nos quedaremos con estos tokens.

Por último, se aplicó la lematización, la cual consiste en llevar a todas las palabras a su forma base, es decir; a su forma infinitiva. A las palabras obtenidas del proceso de lematización se les conoce como lemas. Para realizar este proceso se usó un diccionario de la librería *Spacy* en español. En la Figura 3, la primera columna corresponde a verbos conjugados y la segunda columna a los verbos lematizados.

3.4. Asignación de números de identificación y conteo de palabras

En esta etapa se realizó la asignación de números de identificación y conteo de palabras en las letras, con la función *CountVectorizer()* de la librería *Scikit-learn*. Estos identificadores fueron utilizados posteriormente para hacer un conteo del número de

Palabras	Lemas
Corriendo	Correr
Comiendo	Comer
Saltaba	Saltar

Fig. 3 Lematización

Tabla 1. Porcentaje de precisión.

No. de vecinos	Porcentaje de precisión
1	78%
3	80%
5	84%
7	81%
9	83%

veces que aparece una palabra en una canción y de esta manera se generó una bolsa de palabras.

3.5. Aplicación del método de validación simple

Un método de validación simple consiste en dividir de manera aleatoria el conjunto de datos en dos conjuntos, una se usa para el entrenamiento del algoritmo y el otro para realizar las pruebas. Para llevar a cabo la separación del corpus lingüístico, se utilizó la función "*train_test_split()*" de la librería *Scikit-learn*.

Para el presente proyecto, se aplicó la métrica 70-30, es decir; 70 % del corpus lingüístico se utilizó para el entrenamiento del algoritmo K vecinos más cercanos y el 30 % del corpus lingüístico, para realizar las pruebas de funcionamiento y para determinar el porcentaje de precisión.

3.6. K vecinos más cercanos

El algoritmo de inteligencia artificial k vecinos más cercanos utiliza la distancia euclidiana para determinar qué datos están más cercanos al dato a clasificar. Dependiendo del conteo mayor de datos, se asignará la clase al nuevo dato. Para aplicar el algoritmo de clasificación KNN se utilizó la librería *Scikit-Learn*.

En esta última etapa se ingresó el número de vecinos a considerar para la clasificación, de acuerdo con el estado del arte, al tratarse de una clasificación bi-clase se recomienda que el número de vecinos para la clasificación sea impar para evitar empates.

Posteriormente se realizó el entrenamiento del algoritmo k vecinos más cercanos con el 70 % del corpus lingüístico previamente procesado y etiquetado. Una vez que se entrenó al algoritmo de k vecinos más cercanos se procedió a verificar el funcionamiento del algoritmo con el conjunto de prueba, como resultado se obtuvo el porcentaje de precisión del algoritmo de clasificación y las letras clasificadas.

4. Resultados

Al tratarse de un algoritmo que considera los vecinos más cercanos al dato a clasificar se consideraron 1, 3, 5, 7 y 9 vecinos más cercanos, los porcentajes de precisión que se obtuvieron con cada métrica se presentan en la Tabla 1.

Como se observa en la Tabla 1 al variar el número de vecinos para la clasificación de la letra, también varía el porcentaje de precisión. Observando las pruebas se puede identificar que se obtiene un mayor porcentaje de precisión al considerar solamente a los 5 vecinos más cercanos al dato a clasificar.

5. Conclusiones y trabajo a futuro

Al variar el número de vecinos para la clasificación de la letra varía el porcentaje de precisión por lo que en este algoritmo se consideran los 5 vecinos más cercanos. Para lograr este porcentaje de precisión se realiza un conjunto de técnicas de preprocesamiento que permiten eliminar palabras no necesarias para el análisis del texto, así como permiten reducir el tiempo de procesamiento del corpus, de esta forma se pueden ahorrar recursos computacionales.

El algoritmo propuesto para la clasificación de canciones infantiles solamente funciona con letras en español debido a que los diccionarios que se usan para la eliminación de stop-word y lematización se encuentran en español, por lo que una palabra en inglés será desconocida para los procesos de eliminación de stop-words y lematización. También este algoritmo es muy vulnerable a los errores ortográficos, ya que si una palabra está mal escrita será completamente diferente para el algoritmo, estos errores ortográficos pueden afectar el porcentaje de precisión del algoritmo de clasificación.

Debido a los problemas antes mencionados para trabajos futuros se propone complementar las técnicas de preprocesamiento con diccionarios en otros idiomas y por último se propone que las letras que son usadas para el desarrollo del corpus serán verificadas anteriormente de forma manual para corregir las faltas ortográficas.

Referencias

1. Mahedero, J. P. G., Martinez, A., Cano, P.: Natural language processing of lyrics. In: Proceedings of the 13th annual ACM International Conference on Multimedia, pp. 475–478 (2005) doi: 10.1145/1101149.1101255.

2. Patra, B., Das, D., Bandyopadhyay, S.: Mood Classification of Hindi Songs based on Lyrics. In: Conference on Natural Language Processing, pp. 261–267 (2015)
3. Ashley, M., Sarah, E.: Identifying the Emotional Polarity of Songs Lyrics through Natural Language Processing. (2010)
4. Mulholland, M., Quinn, J.: Suicidal Tendencies: The Automatic Classification of Suicidal and Non-Suicidal Lyrics Using NLP. International Joint Conference on Natural Language Processing, pp. 680–684 (2013)
5. Srinilta, C., Sunhem, W., Tungjitnob, S., Thasanthiah, S., Vatathanavaro, S.: Lyric-based Sentiment Polarity Classification of Thai Songs. In: Proceedings of the International MultiConference of Engineers and Computer Scientists vol. 1 (2017)
6. Bird, S., Klein, E., Loper, E.: Natural Language Processing with Python, Analyzing Text with the Natural Language Toolkit. (2009)
7. Hervás, M., Barrio, G.: Influencia de las actividades audio musicales en la adquisición de la lectoescritura en niños y niñas de cinco años. (2017)
8. Orantes, A.: La influencia de la música en nuestros niños. (2001)
9. Martínez, B.: Canciones infantiles de siempre con un nefasto mensaje para los niños. (2020)
10. Pandas. [Online] Available: https://pandas.pydata.org/
11. Numpy [Online] Available: https://numpy.org/
12. ScikitLearn [Online] Available: https://scikit-learn.org/