
Escuela Técnica Superior de Ingeniería

Universidad de Huelva

# Técnicas de *Deep Learning* Para la Detección Automática de Discurso Irónico Intencionado en Redes Sociales

Trabajo de Fin de Grado

Grado en Ingeniería Informática

Autor: Adrián Moreno Monterde
Tutores: Jacinto Mata Vázquez
Victoria Pachón Álvarez

Huelva, septiembre 2022

*Trabajo Fin de Grado*
*Grado en Ingeniería Informática*

*Título*: Técnicas de *Deep Learning* Para la Detección Automática de Discurso Irónico Intencionado en Redes Sociales

*Autor*: Adrián Moreno Monterde
*Tutores*: Jacinto Mata Vázquez
Victoria Pachón Álvarez

Departamento de Tecnologías de la Información
Escuela Técnica Superior de Ingeniería
Universidad de Huelva

# Agradecimientos

*A mi abuelo, que jamás se fue … simplemente se convirtió en mi estrella favorita del cielo.*

*Te llevo en cada uno de mis rasgos; soy tu prolongación en esta vida.*

*Sigue guiándome y dándome fuerzas allá donde estes, que yo seguiré cuidando a la abuela.*

*A ti te dedicaré cada uno de mis éxitos.*

*Te extraño.*

# Resumen

El sarcasmo es una forma de ironía verbal que se produce cuando existe una discrepancia entre el significado literal y el pretendido de un enunciado. A través de esta discrepancia, el hablante expresa su posición respecto a una proposición previa, a menudo, en forma de desprecio superficial o derogatoria.

El sarcasmo está omnipresente en las redes sociales y, debido a su naturaleza, puede ser altamente confuso para los sistemas computacionales que usan estos datos para realizar tareas de análisis de sentimientos.

El objetivo principal de este trabajo es el estudio e implementación de técnicas para la detección automática del discurso sarcástico intencionado en la red social *Twitter*. Para abordar el problema se han implementado tanto modelos clásicos de *Machine Learning* como modelos avanzados de *Deep Learning*. Concretamente se han desarrollado modelos de *Transformers*, que tan buenos resultados están obteniendo en tareas relacionadas con el Procesamiento del Lenguaje Natural (PLN).

El trabajo comienza con una contextualización del marco teórico, donde se definen los principales conceptos de *Machine Learning* y *Deep Learning*, así como los de las estrategias utilizadas para mejorar el rendimiento de los modelos: balanceamiento de datos, preprocesamiento, uso de recursos externo, entre otros.

Para evaluar los resultados de las técnicas desarrolladas, se ha participado en el "*16th International Workshop on Semantic Evaluation*", concretamente en la tarea "*iSarcasmEval. Intended Sarcasm Detection in English*". Se describe la tarea a resolver y los modelos presentados en la competición, analizando los resultados obtenidos.

Una vez concluida la competición se han propuesto un conjunto de nuevas propuestas para mejorar los resultados obtenidos en la competición. Se detalla y realiza una comparativa con los resultados previos.

El trabajo realizado y presentado en la competición ha culminado con un artículo científico que ha sido aceptado y publicado en las actas del "*16th International Workshop on Semantic Evaluation*".

**Palabras claves:** *Machine Learning, Deep Learning, Python, Data Augmentation*, Redes Bayesianas, *Transformers*.

# Abstract

Sarcasm is a form of verbal irony that occurs when there is a discrepancy between the literal and intended meaning of a statement. Through this discrepancy, the speaker expresses their position with respect to a previous proposition, often in a superficial contempt or derogatory manner.

Sarcasm is omnipresent in social networks and due to its nature, can be highly confusing for computational systems that use this data to conduct sentiment analysis tasks.

The main objective of this work is to study and implement techniques for the automatic detection of intentional sarcastic speech in the social network Twitter. Both classical Machine Learning models and advanced Deep Learning models have been implemented to address the problem. Specifically, Transformers models have been developed, which are obtaining such good results in tasks related to Natural Language Processing (NLP).

The work begins with a contextualization of the theoretical framework, where the main concepts of Machine Learning and Deep Learning are defined, as well as the strategies used to improve the performance of the models: data balancing, preprocessing, use of external resources, among others.

To evaluate the results of the developed techniques, we have participated in the 16$^{th}$ International Workshop on Semantic Evaluation, specifically in the task: iSarcasmEval. Intended Sarcasm Detection in English. The task to be solved and the models presented in the competition are described, analyzing the results obtained.

Once the competition is over, a set of new proposals have been proposed to improve the results obtained in the competition. A comparison with previous results is detailed and carried out.

The work conducted and presented in the competition has culminated in a scientific article that has been accepted and published in the proceedings of the 16$^{th}$ International Workshop on Semantic Evaluation.

**Keywords:** Machine Learning, Deep Learning, Python, Data Augmentation, Bayesian Networks, Transformers.

# Tabla de contenido

# Índice de tablas

# Índice de figuras

# 1 Introducción

El lenguaje natural es lo que nosotros, los seres humanos, usamos para expresar nuestros sentimientos o pensamientos. No importa lo que leas, hables, escribas o escuches que estará casi siempre en lenguaje natural [1].Desde el contenido de un libro hasta los mensajes que enviamos y recibimos diariamente en *WhatsApp* no son más que una forma del lenguaje natural.

Ahora que le hemos dado una acepción al lenguaje natural, podemos hablar del procesamiento del lenguaje natural (PLN), el cual no es más que la capacidad que tienen las tecnologías computacionales de procesar el lenguaje natural del ser humano.

El PLN abarca multitud de niveles como el fonológico, sintáctico y morfológico [2] entre otros, pero en el desarrollo de este trabajo nos vamos a enfocar en el nivel semántico, donde PLN aprende el significado de las palabras y su contexto para así saber el significado que se le dan a las frases.

## 1.1 Objetivos

El objetivo principal de este trabajo es el estudio e implementación de técnicas para la detección automática del discurso sarcástico intencionado en la red social *Twitter*. Para abordar el problema se han implementado tanto modelos clásicos de *Machine Learning* como modelos avanzados de *Deep Learning*.

Para evaluar los resultados de las técnicas desarrolladas, se ha participado en el "*16th International Workshop on Semantic Evaluation*", concretamente en la tarea "*iSarcasmEval. Intended Sarcasm Detection in English*" [3].

El trabajo realizado y presentado en la competición ha culminado con un artículo científico que ha sido aceptado y publicado en las actas del "*16th International Workshop on Semantic Evaluation*".

## 1.2 Estructura del documento

El documento sigue la siguiente organización:

- En el Capítulo 1, Introducción, se presenta la temática de este trabajo, se establecen los objetivos que se persiguen con su realización y se expone la estructura del documento.

- En el Capítulo 2, Marco Teórico, se expone la teoría de todo lo que se abarca en este trabajo, de manera que el lector pueda comprender los conceptos utilizados.

- En el Capítulo 3, Competición *iSarcasmEval*, se hace un seguimiento de lo realizado en la fase de evaluación de la competición, así como modelos, técnicas y métricas obtenidas.

- En el Capítulo 4, Propuestas de Mejora, se plantean y realizan una serie de propuestas de mejora, las cuales están encaminadas a mejorar los resultados conseguidos en la competición.

- En el Capítulo 5, Conclusiones y Trabajo Futuro, se enuncian las conclusiones a las que se ha llegado con la realización de este trabajo y se asientan futuros trabajos que podrían seguir teniendo esta investigación para seguir progresando.

- Finalmente, en el Anexo, se presenta el artículo científico publicado mientras se ha realizado este trabajo.

# 2 Marco Teórico

En este punto se describen los conceptos teóricos necesarios para saber cómo se ha ido abordando la tarea en la que se ha participado, la cual abarca desde algoritmos de *Machine Learning* clásicos como las Redes Bayesianas hasta técnicas de *Deep Learning* como los *Transformers*.

## 2.1 Clasificación automática de textos

### 2.1.1 Machine Learning

*Machine learning* (de aquí en adelante ML) es el estudio de algoritmos que permiten a las máquinas aprender a través de la experiencia de forma automática [4]. Es decir, a partir de datos una máquina puede identificar patrones de comportamiento y así puede llegar a tener la capacidad de predecir posibles comportamientos futuros para ciertas situaciones de las que ha obtenido previamente una experiencia.

El aprendizaje automático se categoriza en tres tipos de aprendizajes en función del algoritmo que sea empleado [5]:

- Aprendizaje supervisado: En el aprendizaje supervisado [6], el algoritmo analiza la relación entre las entradas y las salidas, es decir, aprende a base de ejemplos y contraejemplos para así ser capaz de predecir salidas en base a nuevas entradas con las que no ha sido entrenado previamente.

- Aprendizaje no supervisado: A diferencia del anterior, en el aprendizaje no supervisado [7], el algoritmo solo recibe entradas que están sin etiquetar y estudia distintos patrones de comportamiento que puedan tener los datos.

- Aprendizaje reforzado: En el aprendizaje reforzado, a diferencia de los aprendizajes supervisado y no supervisado, no se parte de un modelo algorítmico conocido inicialmente, sino de un escenario el cuál sirve como base para una experimentación dinámica en función de una recompensa final [8].

Uno de los algoritmos más conocidos dentro del campo de *Machine Learning* son los clasificadores de Redes Bayesianas.

### 2.1.2 Redes Bayesianas

Los clasificadores bayesianos son clasificadores estadísticos (función con probabilidades). Predicen las probabilidades de pertenecer a una clase, así como la probabilidad de que una muestra dada pertenezca a una clase en particular [9].

Estos están basados en el teorema de *Bayes* [10]. Los clasificadores de redes Bayesianas asumen que el efecto de un valor de un atributo perteneciente a clase dada es independiente a los valores de otros atributos. Esta característica se conoce con el nombre de independencia condicional de clase, por lo que el uso de los clasificadores de redes bayesianas resulta de mucha utilidad en clasificaciones multi-etiqueta con independencia entre las clases.

Los clasificadores bayesianos trabajan como se detalla a continuación [11]:

1. Partiendo del conjunto de muestras de entrenamiento $T$, las cuáles tienen sus etiquetas de clase. Hay $k$ clases, $C_1, C_2, \dots, C_k$. Cada muestra es representada por un vector de *n* dimensiones, $X = \{x_1, x_2, \dots, x_n\}$, que representa *n* valores de los *n* atributos $A_1, A_2, \dots, A_n$ , respectivamente.

2. Dada una muestra $X$, el clasificador predecirá que $X$ pertenece a la clase que tenga mayor probabilidad a posteriori, condicionada a $X$. Es decir, se predice que $X$ pertenece a $C_i$ sí y solo si:

   $$P(C_i|X) > P(C_j|X) \text{ para } 1 \leq j \leq m, j \neq i$$

   Por lo que encontramos la clase que maximiza $P(C_i|X)$. La clase $C_i$ para cual $P(C_i|X)$ es maximizada se denomina hipótesis máxima a posteriori. Por el teorema de Bayes:

   $$P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X)}$$

3. Como $P(X)$ es la misma para todas las clases, solamente $P(X|C_i)P(C_i)$ necesita ser maximizado. Si las probabilidades de una clase $P(C_i)$ no son conocidas a priori, entonces se asume que las clases son iguales en términos de probabilidad, lo que cual quiere decir que $P(C_1) = P(C_2) = \cdots = P(C_k)$, por lo tanto, se maximiza $P(X|C_i)$. En caso contrario, se maximiza $P(X|C_i)P(C_i)$. Hay que tener en cuenta que las probabilidades a priori de una clase pueden ser estimado por $P(C_i) = freq(C_i, T)/|T|$.

4. Para un conjunto de datos con muchos atributos, puede suponer una alta carga computacional el tener que calcular $P(X|C_i)$. Para reducir esta computación en evaluar $P(X|C_i)P(C_i)$, se supone la independencia condicional se las clases. Suponiendo que los valores de los atributos son condicionalmente independientes entre sí, dada la etiqueta de clase de la muestra. Matemáticamente, esto quiere decir que:

$$P(X|C_i) \approx \prod_{k=1}^{n} P(x_k|C_i)$$

Las probabilidades $P(x_1|C_i), P(x_2|C_i), \dots, P(x_n|C_i)$ pueden ser fácilmente calculadas a través del conjunto de entrenamiento. Hay que recordar que $x_k$ hace referencia al valor de un atributo $A_k$ de una muestra $X$.

a. Si $A_k$ es categórico, entonces $P(x_k|C_i)$ es el número de muestras de la clase $C_i$ en $T$, teniendo el valor $x_k$ para el atributo $A_k$, dividido por la frecuencia $freq(C_i, T)$ , el número de muestras de una clase $C_i$ en $T$.

b. Si $A_k$ es un valor continuo, entonces se asume que los valores tienen una distribución Gaussiana con una media μ y una desviación estándar σ definida por:

   $g(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} \exp - \frac{(x-\mu)^2}{2\sigma^2}$, por lo que $p(x_k|C_i) = g\left(x_k, \mu_{C_i}, \sigma_{C_i}\right)$.

   Se necesita calcular $\mu_{C_i}$ y $\sigma_{C_i}$ , las cuáles son la media y la desviación estándar de los valores del atributo $A_k$ para entrenar muestras de la clase $C_i$.

5. Para predecir la etiqueta de la clase de $X$, $P(X|C_i)P(C_i)$ es evaluada para cada clase $C_i$. El clasificador predice que la etiqueta de clase $X$ es $C_i$ si y solo si es la clase que maximiza $P(X|C_i)P(C_i)$ .

### 2.1.3 Deep Learning

Dentro del aprendizaje automático encontramos también el *Deep Learning*, el cual se considera como una nueva evolución del *Machine Learning* [12]. El *Deep Learning* aprende de una manera más similar a como aprenden los humanos en comparación con el *Machine Learning*, ya que imita lo que percibe nuestro cerebro conectándolo con las neuronas. De ahí a que la mayor parte de los métodos que usa el *Deep Learning* tienen una arquitectura de redes neuronales, las cuales están organizadas en capas sucesivas [13].

El uso de un mayor número de capas en una red neuronal nos aporta un mayor nivel de abstracción, permitiéndonos al mismo modo, que el algoritmo sea capaz de obtener una mayor cantidad de características e información de aquello para lo que lo estemos entrenando.

En la Figura 1, podemos observar la principal diferencia entre el *Machine* y el *Deep Learning*, que es que en algoritmos *Machine Learning* una persona humana se tiene que encargar de extraer características relacionadas con los datos de entrada para decirle al algoritmo que características son instructivas a la hora de tomar una decisión. Mientras tanto, en los modelos de *Deep Learning* ya se incluye esta extracción de características implícita con el uso del algoritmo [14].

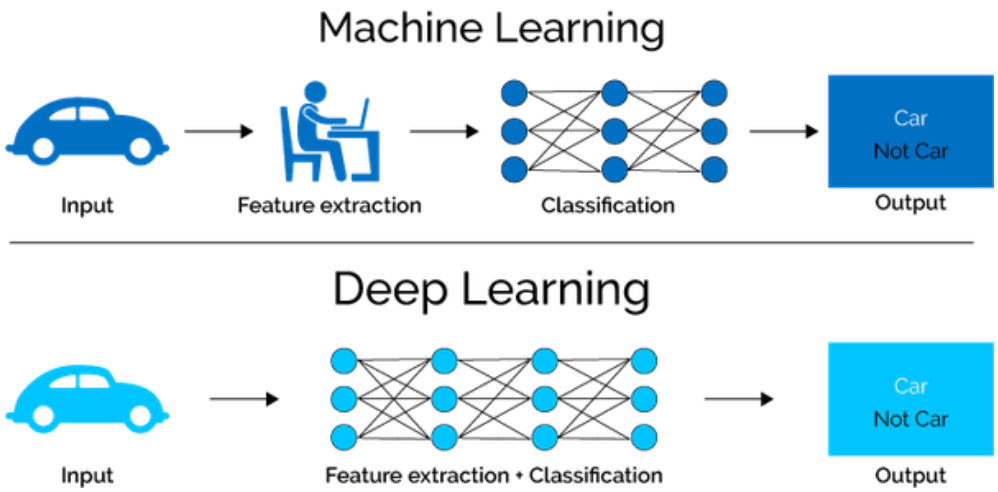


***Figura 1.*** *Principal diferencia entre Machine y Deep Learning*

### 2.1.4 Transformers

Dentro del *Deep Learning* encontramos que las Redes Neuronales Recurrentes [15] o las Redes *Long short-term memory* (LSTM) [16] han estado a la orden del día para abordar distintas tareas de clasificación, pero desde la presentación de los modelos *Transformers* [17], ha habido una revolución debido al gran impacto que ha tenido en el campo del procesamiento del lenguaje natural (PLN).

Dentro del PLN los *Transformers* se definen como una arquitectura novedosa que tiene como objetivo resolver tareas de secuencia a secuencia mientras maneja dependencias de largo alcance con facilidad. Se basan por completo en la autoatención (*self-attention*) para calcular las representaciones de las entradas y salidas sin utilizar redes neuronales recurrentes [18].

Una de las mayores ventajas que posee usar los *Transformers* es la no necesidad de tener los datos ordenados de manera secuencial. Por lo que a la hora de que el modelo trabaje con una frase, no va procesando palabra por palabra, lo cual eso nos aporta una paralelización a la hora del entrenamiento o, lo que es lo mismo, mayor velocidad a la hora de trabajar con el modelo.

Como se puede observar en la arquitectura de los modelos basados en *Transformers* (véase Figura 2), el mismo presenta un sistema *encoding* y *decoding* por capas, junto con el mayor avance que tiene con respecto a las redes neuronales recurrente (RNN) que son los mecanismos de atención (*self-attention mechanism*).

Los mecanismos de atención relacionan diferentes posiciones de una simple secuencia para computar una representación de ésta. Es decir, se asigna una consulta y un conjunto de pares clave-valor a una salida, donde la consulta, las claves, lo valores y la salida son todo vectores. La salida se calcula como una suma ponderada de los valores, donde el peso asignado a cada valor se calcula mediante una función de compatibilidad de la consulta con la clave correspondiente.

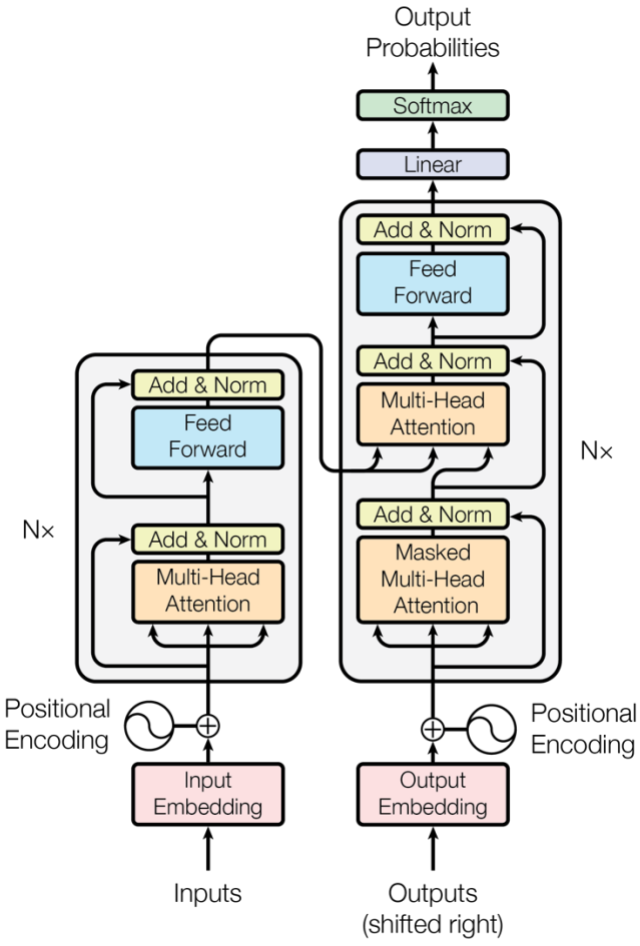


***Figura 2.*** *Arquitectura de los modelos Transformers*

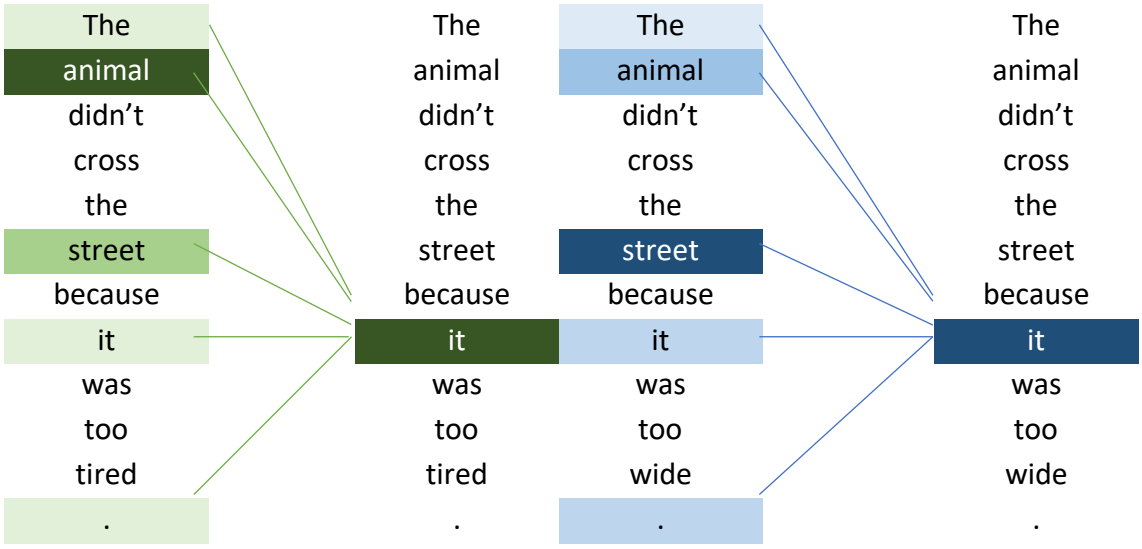


***Figura 3.*** *Ejemplos del funcionamiento del mecanismo de atención*

En la Figura 3 se aprecia dos ejemplos de cómo funcionan los mecanismo de atención en los transformers sobre el pronombre "*it*", en donde en un ejemplo deduciría que se hace referencia al animal (izquierda) mientras que en el otro deduce que hace referencia a la calle (derecha). No obstante, también existen otra palabras que son candidatas como "*the*" o el propio "*it*", pero su valor tendrá menos peso en el vector que las palabras con el relleno más oscuro.

Otra de las ventajas que poseen los modelos basados en *Transformers* con respecto a otros son el aprendizaje transferido (*Transfer Learning*) el cual permite trasladar la experiencia que se ha obtenido para la resolución de otros problemas para resolver nuestro actual problema que se quiera resolver.

*BERT (Bidirectional Encoder Representation from Transformers)*

La arquitectura del modelo BERT es un codificador *Transformer* bidireccional multicapa [19], lo que permite al modelo ser capaz de aprender el contexto de una palabra (secuencia antecesora y predecesora).

Al tener el modelo la característica de bidireccionalidad, BERT no usa los modelos tradicionales de izquierda-derecha o derecha-izquierda para pre-entrenarlo. En su lugar usa dos tareas de carácter no supervisado de manera simultánea.

1. Modelado del lenguaje enmascarado (MLM): Para entrenar una representación bidireccional profunda, simplemente se enmascara el 15% de todos los tokens de entrada en cada secuencia de forma aleatoria para después predecir esos tokens enmascarados. Esto al mismo tiempo puede crear un desajuste en el pre-

entrenamiento y el *fine-tuning*, debido a que el token [MASK] no aparece durante el ajuste fino. Para mitigar esto, BERT no siempre sustituye las palabras "enmascaradas" por el token [MASK] real, sino que sigue la siguiente distribución:

a. 80% de las veces → el token [MASK]

b. 10% de las veces → token aleatorio

c. 10% de las veces → el mismo token sin cambios

2. <u>Predicción de la siguiente frase (NSP):</u> Para entrenar un modelo que sea capaz de entender las relaciones entre oraciones, se hace un entrenamiento previo para una tarea de predicción de la siguiente oración binarizada. Es decir, se elige las frases A y B para cada ejemplo de pre-entrenamiento y el 50% de las veces B es la frase siguiente a A y el otro 50% es una frase aleatoria del corpus que se esté utilizando.

Las entradas del encoder de BERT requiere que las frases tengan el siguiente formato:

[CLS] + Frase A + [SEP] + Frase B,

siendo:

- [CLS]: el token de clasificación
- [SEP]: el token de separación entre las 2 frases (aunque la frase B es opcional)

En la Figura 4 se observa la forma que tienen las frases a la hora de introducirlas por el enconder:

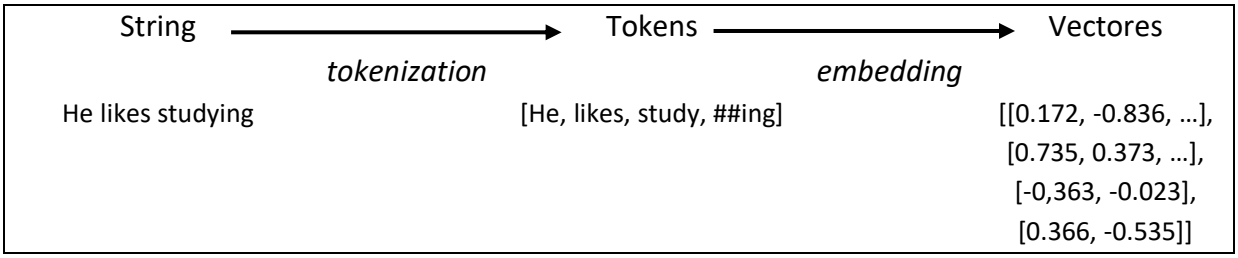


***Figura 4.*** *Forma de entrada de las frases BERT*

Por lo tanto, el enconder necesita las siguientes entradas (Véase la Figura 5):

- Vectores de palabras (*embedding*)
- Un indicador que diga si pertenece a la frase A o a la frase B
- Un *embedding* posicional

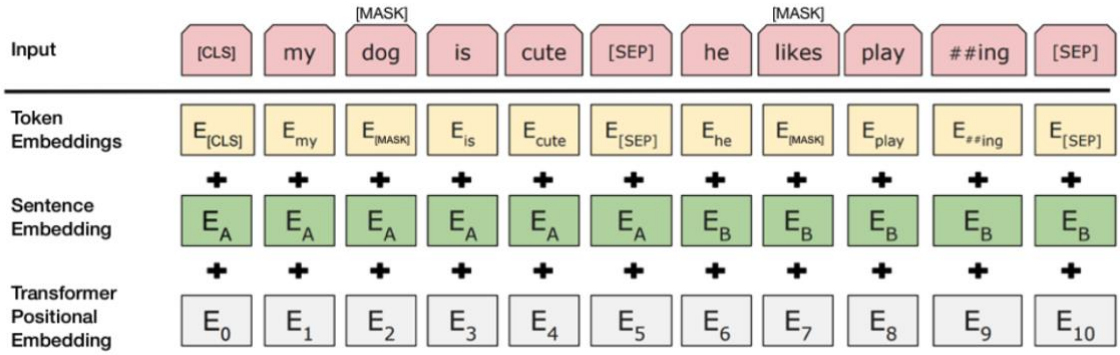


***Figura 5.*** *Entradas necesarias por el encoder*

En la Figura 6 se aprecian los dos tipos de salidas que se obtienen:

- Salida para el token [CLS], el cual es usado para la clasificación y es entrenada durante el NSP.

- Una representación para cada token, la cual es usada para tareas a nivel de palabras y es entrenada durante el MLM.

El modelo BERT utilizado en este proyecto es el *BERT-base-uncased*, pero es importante destacar que existen diferentes modelos de BERT, los cuales se pueden adaptar de mejor manera al problema que queramos resolver, como, por ejemplo: *BERT-base-cased*, *BERT-large-cased* o *BERT-large-uncased* entre otros.

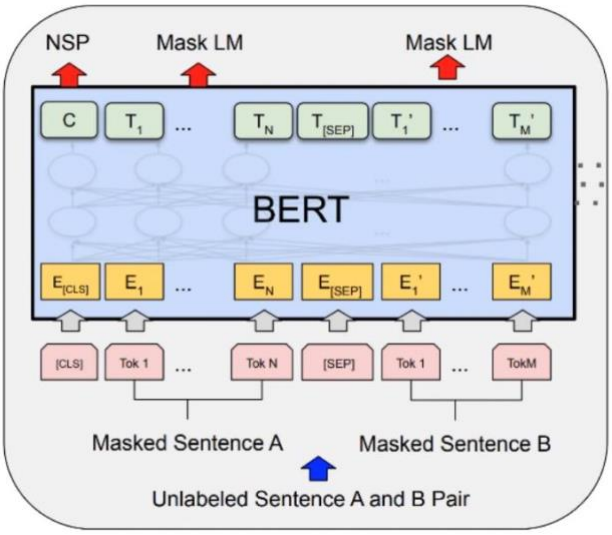


***Figura 6.*** *Arquitectura del modelo BERT*

*ROBERTA (Robustly Optimized BERT Approach)*

Podemos definir RoBERTa [20] como una readaptación del modelo BERT, con el fin de mejorar sus resultados obtenidos en base a una serie de modificaciones centradas principalmente en el procedimiento de su entrenamiento. De esta manera, el modelo RoBERTa igualará o superará el rendimiento de todos los métodos post-BERT.

Las modificaciones son las siguientes:

1. Entrenar el modelo más tiempo, con más datos y lotes más grandes. Se confirma que el uso de mayores cantidades de datos en el entrenamiento previo mejora el rendimiento en las tareas posteriores. Se demuestra asimismo que el entrenamiento con mini lotes muy grandes puede mejorar la velocidad de optimización y el rendimiento de la tarea final cuando la tasa de aprendizaje aumenta adecuadamente.

2. Eliminar el objetivo de predicción de la siguiente oración. La predicción de la siguiente oración (NSP), como ya se ha definido en el apartado anterior de BERT, es una pérdida de clasificación binaria para predecir si dos segmentos se suceden en el texto original. Los ejemplos positivos se crean tomando oraciones consecutivas del corpus de texto. Los ejemplos negativos se crean emparejando segmentos de diferentes documentos. Los ejemplos positivos y negativos se muestrean con igual probabilidad. Se demuestra que la eliminación de la NSP iguala o mejora el rendimiento en la tarea posterior.

3. Entrenamiento en secuencia más largas. Inicialmente se entrenó a *BERT-base* para 1 millón de pasos con un tamaño de lote de 256 secuencias. Posteriormente se entrena con 125 mil pasos de 2 mil secuencias y 31 mil pasos de 8 mil secuencias de tamaño de lote. Se observa que en el entrenamiento con mayores secuencias y lotes más grandes mejora la perplejidad para el objetivo de modelado del lenguaje enmascarado y la precisión de la tarea final.

4. Cambiar dinámicamente el patrón de enmascaramiento aplicado a los datos de entrenamiento. La implementación inicial de BERT realizó el enmascaramiento una vez durante el preprocesamiento de datos, lo que da como resultado una única máscara estática. Con el objetivo de evitar el uso de la misma máscara se duplican los datos de entrenamiento 10 veces. De esta forma, cada secuencia queda enmascarada de 10 maneras diferentes durante las 40 epochs de entrenamiento, por lo que cada secuencia de entrenamiento se vio con la misma máscara 4 veces durante el entrenamiento. Esta estrategia se compara con el enmascaramiento dinámico donde cada vez que alimentamos una secuencia al

modelo se genera el patrón de enmascaramiento. Esto se vuelve crucial cuando se entrena previamente para más pasos o con conjuntos de datos más grandes.

Al igual que ocurría con BERT, podemos encontrar diferentes modelos de RoBERTa que pueden adaptarse mejor al problema que estemos resolviendo, como, por ejemplo: *RoBERTa-base* o *RoBERTa-large*.

## 2.2 Word Embeddings

En términos generales, definimos *Word Embedding* [21] como las representaciones vectoriales de una palabra en particular. Éste nos permite conocer el contexto de una palabra en un documento, similitud semántica y sintáctica y relación con otras palabras entre otras características.

### 2.2.1 Word2Vec

*Word2Vec* [22] es un método para construir dicha incrustación. Puede obtenerse utilizando dos métodos (ambos con redes neuronales): *Skip-Gram* y *Common Bag of Words* (*CBOW*).

- *CBOW*: Este método toma el contexto de cada palabra como entrada e intenta predecir la palabra correspondiente al contexto (Véase Figura 7).

- *Skip-Gram*: Este método [23], hace lo contrario a *CBOW*, es decir, intenta predecir las palabras del contexto utilizando la palabra principal (Véase Figura 7).

En este trabajo se ha utilizado el *CBOW* en los algoritmos clásicos de *Machine Learning* utilizados, es decir, en todos los modelos construidos con clasificadores de Redes Bayesianas.

A la hora de tener que escoger entre uno de los dos modelos para trabajar, podemos tener en cuenta lo que se dice en la publicación científica original [24]: *Skip-Gram* trabaja mejor con conjuntos de datos pequeños y puede representar mejor las palabras menos frecuentes. Sin embargo, *CBOW* entrena de manera más rápida que *Skip-Gram* y representa mejor las palabras con más frecuencia.

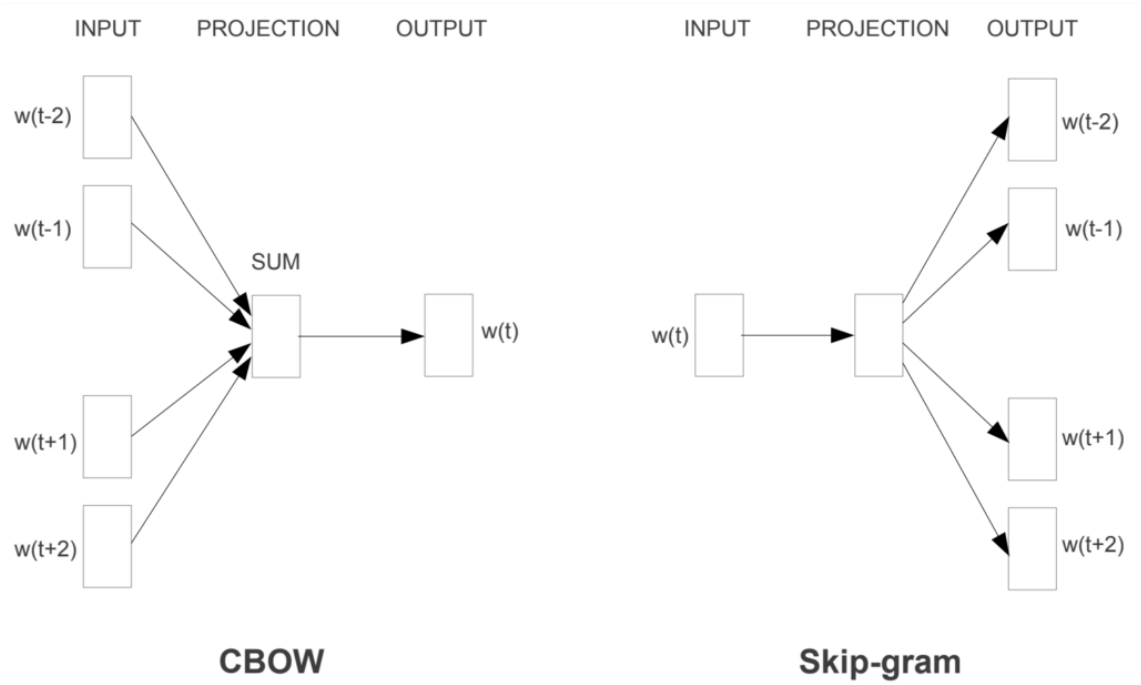


***Figura 7.*** *Comparativa de arquitecturas de CBOW y Skip-Gram*

### 2.2.2 Global Vectors for Word Representation (GloVe)

*GloVe* es un nuevo modelo de representación de palabras que combina las ventajas de las dos principales familias de modelos de la literatura: la factorización matricial global y los métodos de ventana de contexto local. Gracias a ello, encontramos solución al principal inconveniente de *Word2Vec*, que no llega a conocer el contexto global de una palabra en particular. Sin embargo, *GloVe* captura las estadísticas globales del corpus directamente a través del modelo.

## 2.3 Conjuntos de datos desbalanceados

Cuando se habla de conjuntos de datos desbalanceados [25], se hace referencia a un problema de descompensación de muestras de cada clase, donde existe una clase con una gran cantidad de muestras (denominada clase mayoritaria) y otra clase donde el número de muestras es considerablemente menor (clase minoritaria).

Para mejorar el rendimiento cuando trabajamos con conjuntos de datos que presentan desbalanceo, existen diversas técnicas que nos permiten equilibrar los datos antes de entrenar al modelo con el que se esté trabajando. Las propuestas que se presentan en este trabajo para corregir el problema anterior son: la incrustación de datos a partir de una base de datos externas y la aplicación de técnicas de aumento de datos (*Data Augmentation*).

### 2.3.1 Bases de datos externas para entrenamiento

Una de las alternativas que se utiliza para enfrentarse a un conjunto de datos desbalanceado es utilizar otros conjuntos de datos que nos aporte información útil para el problema con el que se esté trabajando [26]. En otras palabras, esta técnica se basa en recopilar más datos de las clases minoritarias para así poder equilibrar las clases y poder solucionar el problema de desbalanceo en las clases.

### 2.3.2 Data Augmentation

Existen diferentes técnicas para poder hacer un aumento de los datos a partir de los ya existentes, pero en este proyecto se ha hecho énfasis en la sustitución léxica y en inyección de ruido.

*Sustitución Léxica*

La sustitución léxica [27] consiste en seleccionar diferentes palabras de una frase para cambiarlas por otras de forma que el significado de la frase no se vea alterado. Concretamente en este proyecto se han utilizado dos tipos de sustituciones léxicas [28]:

- *Random Word Augmenter*: las palabras que selecciona las sustituye por otras palabras de carácter aleatorio.

| **Original** | The quick fox jumps over the lazy dog |
|---|---|
| **Texto Aumentado** | Quick the Brown fox jumps over the lazy dog |

***Tabla 1.*** *Ejemplo de Random Word Augmenter*

- *Synonym Word Augmenter*: las palabras que selecciona las sustituye por otras que se consideran sinónimas.

| **Original** | The quick fox jumps over the lazy dog |
|---|---|
| **Texto Aumentado** | The speedy brown fox completes the lazy dog |

***Tabla 2.*** *Ejemplo de Synonym Word Augmenter*

Ambas técnicas seleccionan palabras que no son consideradas *Stopwords* y las sustituyen usando el corpus *WordNet* [29].

*Inyección de ruido*

Esta técnica consiste en añadir ruido al texto. Es decir, añade faltas de ortografía o erratas a palabras de forma aleatoria, lo que resulta en una gran preparación a los modelos de cara a enfrentarse a textos reales. Y al igual que en la sustitución léxica, las palabras que selecciona no pertenecen al conjunto de palabras *Stopwords*.

En este proyecto se ha usado la sustitución aleatoria de caracteres en frases que pertenecen a las clases minoritarias.

| **Original** | The quick fox jumps over the lazy dog |
|---|---|
| **Texto Aumentado** | Hte quikc borwn fxo jupms ovre teh lzay dgo |

***Tabla 3.*** *Ejemplo de Random Character Augmenter*

## 2.4 Preprocesamiento del texto

El preprocesamiento del texto [30] es una técnica muy importante a la hora de sintetizar el contenido del texto con el que se está trabajando. La eliminación caracteres especiales, emoticonos, links o partición de palabras en tokens, entre otros, son ejemplos de técnicas de preprocesamiento del texto y hacen que el modelo con el que se está trabajando pueda aprender mejor ya que se le está eliminado contenido que no les aporta gran significado a los textos.

Concretamente, en este proyecto se trabaja con tweets, los cuáles pueden llevar menciones (@user), etiquetas (#hashtag), links, fotos o videos adjuntos en el propio tweet. Por lo tanto, esta técnica será fundamental a la hora de eliminar información que no es efectiva para los modelos y quedarnos con lo verdaderamente importante del tweet.

## 2.5 Eliminación de stopwords

Podemos definir las *stopwords* [31] como aquellas palabras que no aportan ningún tipo de información relevante dentro de una frase. Estas son normalmente artículos, preposiciones, pronombres, conjunciones, etc.

Como se ha comentado en el punto anterior, esta técnica hace que eliminemos información que no es de utilidad para el modelo y, por ende, no solo va a entrenar de manera más rápida, sino que aprenderá de una manera más efectiva al haber eliminado "ruido" de la frase.

# 3 Competición ISarcasmEval

## 3.1 Descripción de la tarea a resolver

Tal como se adelantó en la introducción, para la realización de este proyecto se ha participado en la subtarea B de la Tarea 6 de la competición SemEval2022 cuyo nombre es *Detección del Sarcasmo Intencionado en inglés.*

El sarcasmo a menudo se expresa a través de varias señales verbales y no verbales, por ejemplo, un cambio de tono, énfasis excesivo en una palabra, una sílaba prolongada o una cara seria [32].

Sin embargo, los sistemas de análisis de redes sociales existentes están limitados por su incapacidad para detectar e interpretar con precisión el lenguaje figurativo. Las personas suelen utilizar el sarcasmo para expresar opiniones sobre asuntos complejos y sobre objetivos específicos.

Los primeros modelos computacionales para la detección de la ironía y el sarcasmo se han basado en métodos superficiales que explotan las regularidades del recuento condicional de tokens. Pero no solo podemos guiarnos por las pistas léxicas, ya que son insuficientes para discernir la intención del sarcasmo. Para ello, es fundamental apreciar el contexto de la expresión, incluso para los seres humanos [33]. De hecho, la misma frase puede interpretarse como literal o sarcástica dependiendo del hablante.

Esta tarea consiste en la clasificación binaria de tweets en inglés basada en la categoría de discurso irónico. Al tratarse de una clasificación multi-etiqueta, un tweet puede pertenecer a varias categorías o a ninguna. La investigación realizada en la elaboración de este proyecto es importante porque las redes sociales han cambiado nuestras vidas en los últimos años. Para las empresas, esta realidad social se ha convertido en una obligación para establecer canales de comunicación y marketing a través de las redes sociales. Además, las empresas requieren el feedback de los consumidores, a través de comentarios o mensajes que marquen la aceptación o el rechazo de cada una de las propuestas, productos o servicios.

## 3.2 Análisis del conjunto de datos

El conjunto de datos original que nos aportó la organización de la competición tiene 3467 tweets en inglés con un tamaño máximo de 280 caracteres (límite caracteres de un tweet). Pero de ese corpus, solo los primeros 867 tweets son útiles para nuestra tarea porque el resto de los tweets no tiene las etiquetas con las que vamos a trabajar.

Para un mejor análisis y comprensión de las múltiples etiquetas que tenemos en nuestro conjunto de datos es importante mencionar el discurso irónico expuesto en [34]:

- *Sarcasm*: tweets que contradicen el estado de las cosas y son críticos hacia un destinatario.

- *Irony*: al igual que los tweets con carácter sarcástico, estos contradicen el estado de las cosas, pero no son obviamente críticos con un destinatario.

- *Satire*: tweets que parecen apoyar a un destinatario, pero con tienen un desacuerdo y una burla de forma implícita.

- *Understatement*: tweets que restan importancia al estado de las cosas al que se refieren.

- *Overstatement*: tweets que describen el estado de cosas en términos evidentemente exagerados.

- *Rhetorical question*: tweets que incluyen una pregunta cuya inferencia (implicación) invitada es obviamente contradictoria con el estado de cosas.

En la Tabla 4 se describe una muestra del conjunto de datos de entrenamiento.

En cuanto a las múltiples etiquetas (categorías) del sarcasmo, las Tablas 5, 6, 7, 8, 9 y 10 nos muestran la distribución de los tweets en estas categorías. Como puede verse el conjunto de datos está desequilibrado, por lo que se tendrán que aplicar técnicas de balanceo de cara a las distintas soluciones que se proponen en la competición.

No solo tenemos el problema del desequilibrio de datos, sino que también tenemos categorías en donde apenas tenemos tweets con los que entrenar nuestros modelos (*Understatement*, por ejemplo), por lo que aquí es donde aumento de datos puede jugar un papel fundamental a la hora de trabajar con este conjunto de datos.

| Index | tweet | sarcasm | irony | satire | under-statement | over-statement | rhetorical_question |
|---|---|---|---|---|---|---|---|
| *382* | So now that it's not getting banned, should I download TikTok? | 1 | 0 | 0 | 0 | 0 | 0 |
| *662* | Man PepsiCo loves to add 750g of sugar to a mixture of their existing flavors of sodas and call it a new flavor of dew or Pepsi... | 1 | 0 | 0 | 0 | 1 | 0 |
| *150* | I still can't believe England won the World Cup | 1 | 0 | 0 | 0 | 0 | 0 |
| *255* | Hate this site [CHIRPBIRDICON] | 1 | 0 | 0 | 0 | 0 | 0 |
| *314* | I miss walking up 3 flights of stairs for class and having to catch my breath in the bathroom 😩 | 1 | 0 | 0 | 0 | 0 | 0 |
| *409* | Thank you to @OrchardMead for organising pizza and ice cream! It was a lovely end of year gesture 😁💖 Happy holidays everyone - time to collect our time owed in lieu! 😅 #summerholidays https://t.co/324izGSEli | 1 | 0 | 0 | 0 | 0 | 0 |

***Tabla 4.*** *Muestra del conjunto de datos*

| Sarcasm | N.º de tweets |
|---|---|
| **0** | 154 |
| **1** | 713 |

***Tabla 5.*** *Distribución tweets con etiqueta "Sarcasm"*

| Irony | N.º de tweets |
|---|---|
| **0** | 712 |
| **1** | 155 |

***Tabla 6.*** *Distribución tweets con etiqueta "Irony"*

| Satire | N.º de tweets |
|---|---|
| **0** | 842 |
| **1** | 25 |

***Tabla 7.*** *Distribución tweets con etiqueta "Satire"*

| Understatement | N.º de tweets |
|---|---|
| **0** | 857 |
| **1** | 10 |

***Tabla 8.*** *Distribución tweets con etiqueta "Understatement"*

| Overstatement | N.º de tweets |
|---|---|
| **0** | 827 |
| **1** | 40 |

***Tabla 9.*** *Distribución tweets con etiqueta "Overstatement"*

| Rhetorical_question | N.º de tweets |
|---|---|
| **0** | 766 |
| **1** | 101 |

***Tabla 10.*** *Distribución tweets con etiqueta "Rhetorical_question"*

*Frecuencia de palabras del conjunto de datos en función de las categorías*

A continuación, se exponen diferentes gráficas donde se pueden apreciar las palabras que tienen mayor frecuencia en las diferentes etiquetas (sin contar las *stopwords*).

### Clase Sarcasm

En la Figura 8 se observa las palabras que se usan más frecuentemente en relación con la clase *Sarcasm*. De esta manera podemos ver qué términos son más usados cuando una clase es catalogada como sarcástica y cuando no.

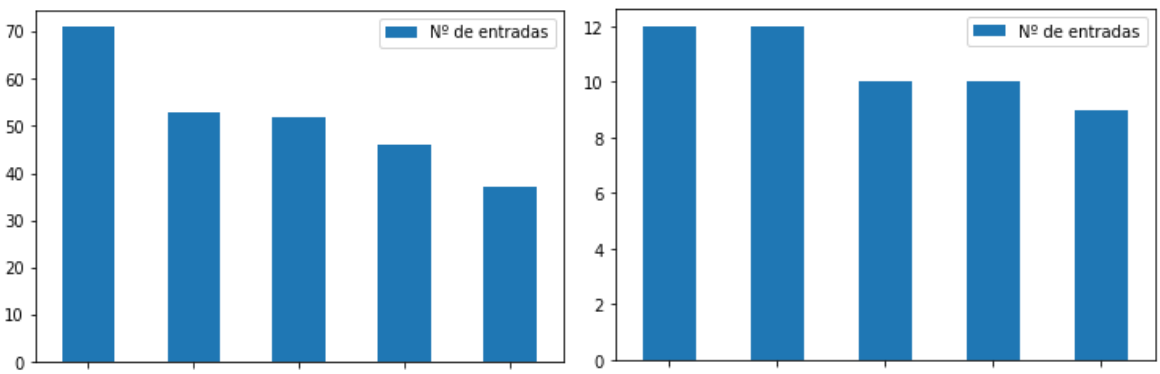


***Figura 8.*** *Palabras frecuentes cuando un tweet es Sarcástico (izquierda) y no es Sarcástico (derecha)*

Clase Irony

En la Figura 9 se observa las palabras que son usadas de manera más frecuente en relación con la clase *Irony*.

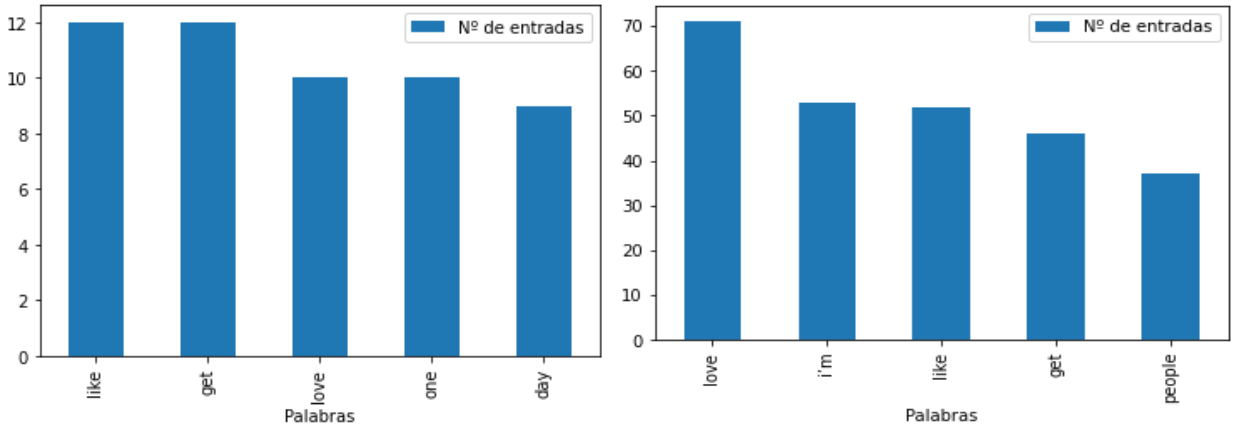


***Figura 9.*** *Palabras frecuentes cuando un tweet es Irónico (derecha) y cuando no es Irónico (izquierda)*

Clase Satire

En la Figura 10 se observa las palabras que son usadas de manera más frecuente en relación con la clase *Satire*.

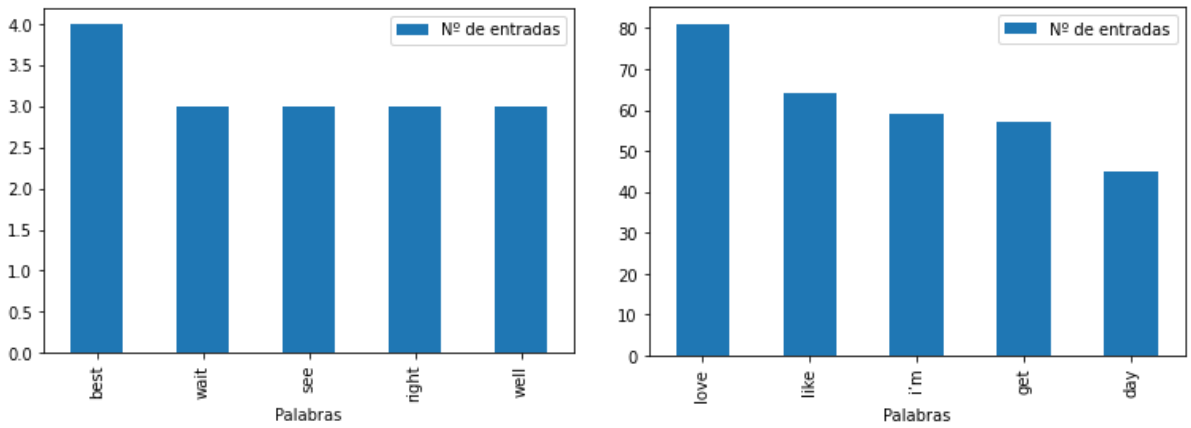


***Figura 10.*** *Palabras frecuentes cuando un tweet es Satírico (izquierda) y cuando no es Satírico (derecha)*

Clase Understatement

En la Figura 11 se observa las palabras que son usadas de manera más frecuente en relación con la clase *Understatement*.

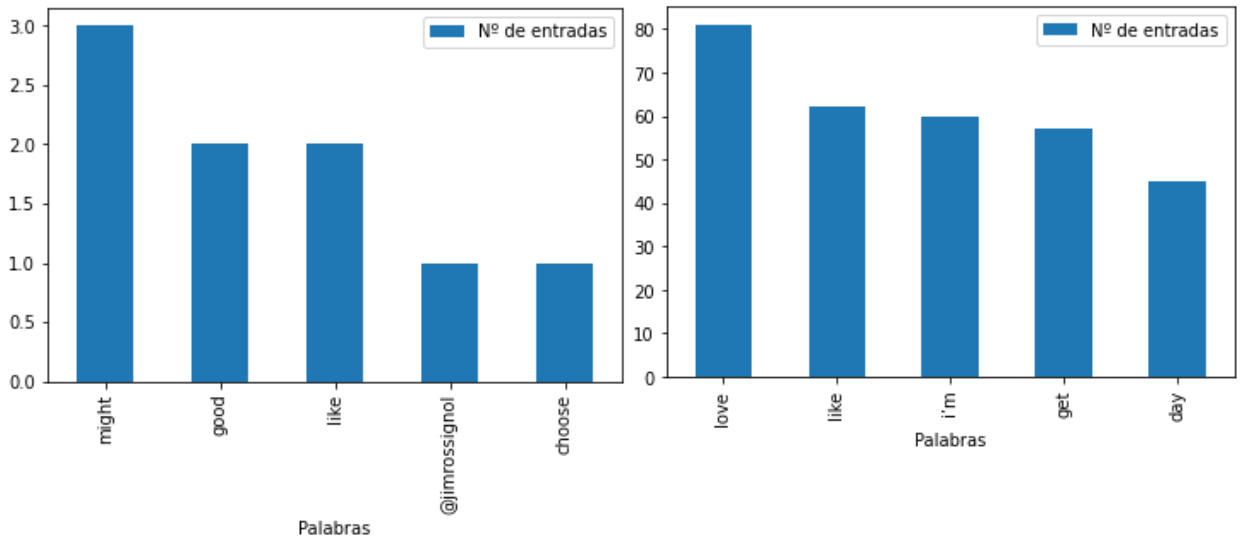


***Figura 11.*** *Palabras frecuentes cuando un tweet es Subestimación (izquierda) y no es Subestimación (derecha)*

## Clase Overstatement

En la Figura 12 se observa las palabras que son usadas de manera más frecuente en relación con la clase *Overstatement*.

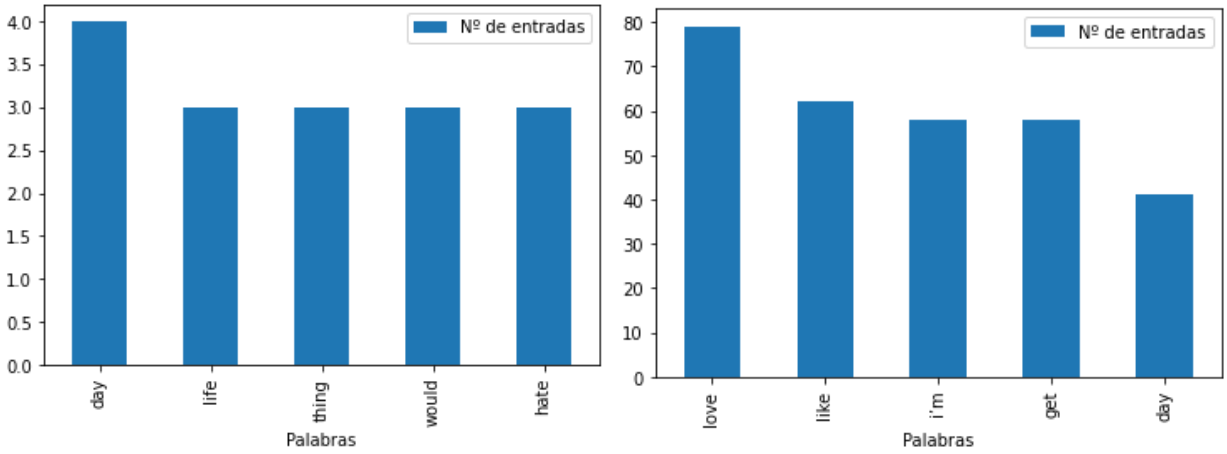


***Figura 12.*** *Palabras frecuentes cuando un tweet es Exageración (izquierda) y no es Exageración (derecha)*

## Clase Rhetorical Question

En la Figura 13 se observa las palabras que son usadas de manera más frecuente en relación con la clase *Rhetorical Question*.

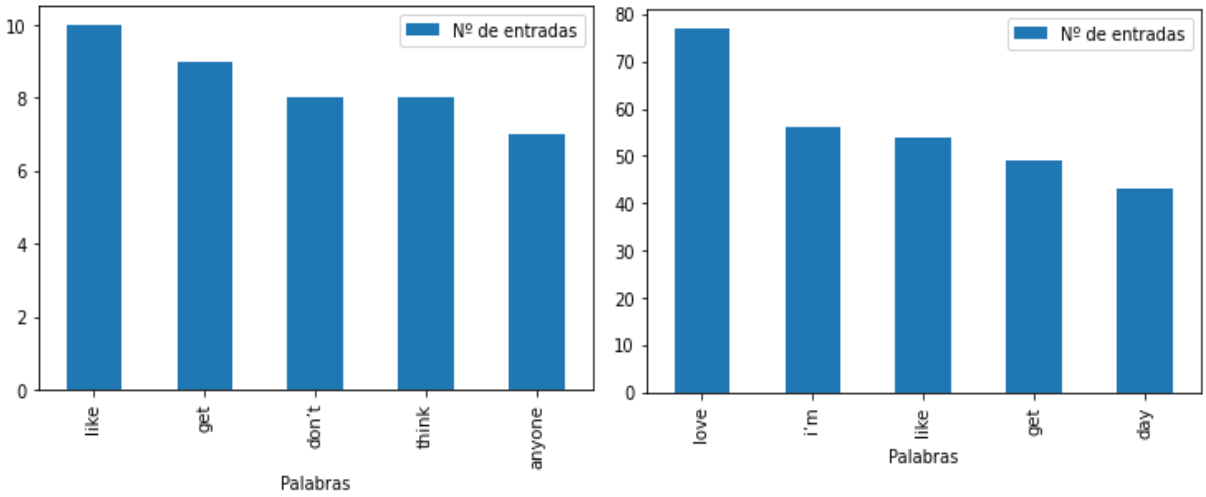


***Figura 13.*** *Palabras frecuentes cuando un tweet es una Pregunta Retórica (izquierda) y no es una Pregunta Retórica (derecha)*

Con las frecuencias de palabras se puede observar la complejidad de esta tarea, ya que hay muchas palabras que se comparten tanto en las clases positivas como en las clases negativa como "*get*", "*like*" o "*love*". Por lo tanto, el contexto de las palabras en las frases será clave para determinar la clasificación.

## 3.3 Implementación

Se he optado por utilizar como lenguaje de programación *Python*[1], debido a la gran variedad de posibilidades que nos aporta para el Procesamiento del Lenguaje Natural, *Machine Learning* y *Deep Learning* [35].

Para el desarrollo de los modelos con los que se ha participado en la competición se usaron las siguientes librerías:

- Para el aumento de datos, la librería *nlpaug*[2] [36].
- Para el relleno de las secuencias de los identificadores de entrada; la biblioteca *keras*[3].
- La biblioteca *sklearn*[4] para las métricas y la división del conjunto de datos [37].
- Librería *pandas*[5] para trabajar con *dataframes* [38].
- Librería *Transformers* de *huggingface*[6] para todo lo relacionado con BERT.

[1] https://www.python.org/
[2] https://nlpaug.readthedocs.io/en/latest/augmenter/word/synonym.html
[3] https://keras.io
[4] https://scikit-learn.org/stable/
[5] https://pandas.pydata.org
[6] https://huggingface.co/

Para el desarrollo y entrenamiento de todos los modelos con los que se ha ido trabajando a lo largo de la realización de este proyecto y de la competición se ha usado la plataforma *Google Colaboratory*[7]. Este permite la posibilidad de trabajar con programas usando el lenguaje *Python* desde el navegador, y permite usar GPUs y TPUs de forma gratuita, aunque de forma limitada.

Para resolver la tarea que se nos propone en la competición, se utilizaron dos modelos/estrategias:

- El primer modelo es un clasificador binario multi-etiqueta que utiliza redes bayesianas.
- El segundo modelo y la presentación final en la competición consistió en seis clasificadores binarios diferentes usando *BERT-base-uncased*. Es decir, una para cada etiqueta evaluable: *Sarcasm, Irony, Satire, Understatement, Overstatement y Rhetorical Question.*

## 3.4 Soluciones propuestas

En esta sección se describen los dos tipos de modelos presentados y las técnicas y métodos aplicados a cada uno de ellos para mejorar su rendimiento y métricas.

Como se ha podido observar en el apartado donde se ha analizado el conjunto de datos de entrenamiento y, como se ha comentado anteriormente en otras secciones, los mayores retos que presenta esta tarea son: el desbalanceado de datos de las distintas categorías de sarcasmo y la falta de datos para entrenamiento en algunas de las clases (en la categoría *Understatement* solamente contamos con 10 tweets en total para entrenamiento y validación).

### 3.4.1 Métricas utilizadas

En esta sección se describen las métricas utilizadas para el análisis de los resultados de los modelos que fueron enviados en la competición.

*F1-score* fue la métrica con la que se nos evaluaba en la competición y por lo tanto en la que más énfasis se ha hecho. No obstante, también se definen otras métricas importantes como *Precision*, *Recall* o la Matriz de Confusión.

[7] https://colab.research.google.com/

*Matriz de Confusión*

Como su propio nombre indica, la matriz de confusión (también conocida como la matriz de error) [39], es una matriz donde podemos visualizar el rendimiento del algoritmo, ya que nos permite ver qué aciertos y errores tiene el modelo.

| | | ***Prediction*** | |
|---|---|---|---|
| | | **Positives** | **Negatives** |
| ***Real Values*** | **Positives** | True positives (TP) | False negatives (FN) |
| | **Negatives** | False positives (FP) | True negatives (TN) |

***Tabla 11.*** *Matriz de Confusión*

*Precision*

La *precision* expresa el porcentaje de valores que el modelo ha predicho como positivos y son realmente positivos. Por lo que nos indica la calidad del modelo.

$$precision = \frac{TP}{TP + FP}$$

*Recall*

El *recall*, también conocido como sensibilidad, es la relación entre las predicciones positivas correctas y el número total de predicciones positivas.

$$recall = \frac{TP}{TP + FN}$$

*F1-score*

La medida f1 es la combinación de las medidas expuestas anteriormente (*precision* y *recall*) para producir un único valor evaluable. Esta métrica es la utilizada en la competición.

$$f1 - score = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

### 3.4.2 Balanceo de datos

Tratando de poner solución al principal defecto que nos encontramos con el conjunto de datos, se han aplicado dos técnicas para balancear los datos: *Data Augmentation* y adición de tweets de una base de datos externa de las clases minoritarias.

*Data Augmentation*

Al disponer un número reducido de tweets para entrenar nuestros modelos (solamente 867 tweets), hemos aplicado una técnica de aumento de datos. En concreto, se utilizó un aumentador de sinónimos usando el corpus *WordNet* en inglés [40] para así crear un nuevo tweet, pero intercambiando únicamente una palabra aleatoria por su sinónimo y manteniendo sus etiquetas. Esta técnica se ha aplicado utilizando la librería *nlpaug,* con todos sus parámetros por defecto.

Se decidió utilizar esta técnica una vez solamente porque, si se usaba más veces, los modelos sobre-ajustaban los datos y producían resultados sobrevalorados de las métricas en la fase de entrenamiento y peores valores a la hora de evaluar el conjunto de datos de test.

*Uso de base de datos externa*

Otra de las técnicas aplicada en los modelos propuestos para aumento de datos fue la inserción manual de tweets y etiquetas [41]. Se ha usado la base de datos de entrenamiento de la competición llamada *iSarcasm*[8], la cual aparece en "*Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*" del año 2020. Esta base de datos está compuesta por 3 etiquetas:

- *tweet_id*, donde aparece el número de identificación del tweet
- *sarcasm_label*, donde aparece si la frase es catalogada como sarcástica o no.
- *sarcasm_type*, donde, en caso de que la frase sea sarcástica, indica el tipo de sarcasmo que tiene el tweet.

Entonces, lo que se ha hecho ha sido buscar e insertar manualmente tweets de todas las clases excepto del tipo *Sarcasm*, buscando tweet por tweet usando a partir de la siguiente url:

$$https://twitter.com/user/status/\{insertarIDTweet\}$$

Finalmente, una vez añadidos manualmente los nuevos tweets con sus respectivas etiquetas, el conjunto de datos pasó de tener de 867 a 904 tweets.

[8] https://github.com/silviu-oprea/iSarcasm/blob/master/isarcasm_train.csv

En la Tabla 12 se puede apreciar las nuevas distribuciones de las clases, tras haber aplicado el aumento al conjunto de datos original y la inserción de la base de datos externa.

| | ***Dataset original*** | ***Dataset original + Base de datos externa*** | ***Dataset original + Aumento*** | ***Dataset original + Aumento + Base de datos externa*** |
|---|---|---|---|---|
| ***Sarcasm*** | Clase 0: 154<br>Clase 1: 713 | Clase 0: 192<br>Clase 1: 713 | Clase 0: 308<br>Clase 1: 713 | Clase 0: 384<br>Clase 1: 713 |
| ***Irony*** | Clase 0: 712<br>Clase 1: 155 | Clase 0: 737<br>Clase 1: 168 | Clase 0: 712<br>Clase 1: 310 | Clase 0: 737<br>Clase 1: 336 |
| ***Satire*** | Clase 0: 842<br>Clase 1: 25 | Clase 0: 871<br>Clase 1: 34 | Clase 0: 842<br>Clase 1: 50 | Clase 0: 871<br>Clase 1: 68 |
| ***Understatement*** | Clase 0: 857<br>Clase 1: 10 | Clase 0: 889<br>Clase 1: 16 | Clase 0: 857<br>Clase 1: 20 | Clase 0: 889<br>Clase 1: 32 |
| ***Overstatement*** | Clase 0: 827<br>Clase 1: 40 | Clase 0: 860<br>Clase 1: 45 | Clase 0: 827<br>Clase 1: 80 | Clase 0: 860<br>Clase 1: 90 |
| ***Rhetorical_question*** | Clase 0: 766<br>Clase 1: 101 | Clase 0: 799<br>Clase 1: 106 | Clase 0: 766<br>Clase 1: 202 | Clase 0: 799<br>Clase 1: 212 |

***Tabla 12.*** *Distribuciones después de aumentar el conjunto de datos*

### 3.4.3 Preprocesamiento del texto

De cara a la competición se han aplicado tres versiones de preprocesamiento del texto para limpiarlo y simplificarlo, basándonos en el trabajo descrito en [42].

Preprocesamiento de texto v1.0: es el preprocesamiento más básico. Para esta versión, se aplicaron las siguientes directrices:

- Conversión de todos los caracteres a minúsculas.
- Extensión de todas las contracciones posibles en inglés (por ejemplo, *what's* → *what is*).
- Eliminación de los emojis.
- Eliminación de los caracteres especiales.
- Eliminación de espacios múltiples entre caracteres.

Preprocesamiento de texto v2.0: es la versión intermedia. Además de las características descritas en la v1.0, se añadieron las siguientes funcionalidades:

- Eliminación de los emojis hecho a partir de caracteres de teclado.
- Eliminación de las menciones
- Eliminación de enlaces

Preprocesamiento de texto v3.0: es la versión más completa. Además de las funciones presentadas en la v2.0, se han añadido la eliminación de las *stopwords* en inglés.

### 3.4.4 Entrenamiento

*Primer modelo: Redes Bayesianas*

El primer modelo que se desarrolló se basó en el algoritmo clásico de redes bayesianas [43] para estudiar el patrón de comportamiento que pueden presentar las categorías de sarcasmo en nuestro conjunto de datos.

Se ha utilizado un clasificador de Bayes ingenuo (NBC), el cuál asume que los atributos son independientes entre sí. Es decir, la probabilidad se puede obtener calculando el producto de las probabilidades condicionales individuales de cada atributo dado el nodo de la clase, como puede verse en la Figura 14.

En este modelo, se proporciona una entrada (un solo tweet) y devuelve un vector de tamaño seis (uno por cada etiqueta) con la etiqueta predicha.

Se enviaron dos modelos a la competición utilizando las Redes Bayesianas: el primero con el preprocesamiento de texto v1.0 y el segundo con el preprocesamiento de texto v3.0. Véase la Tabla 13 para ver los resultados obtenidos durante la fase de entrenamiento.

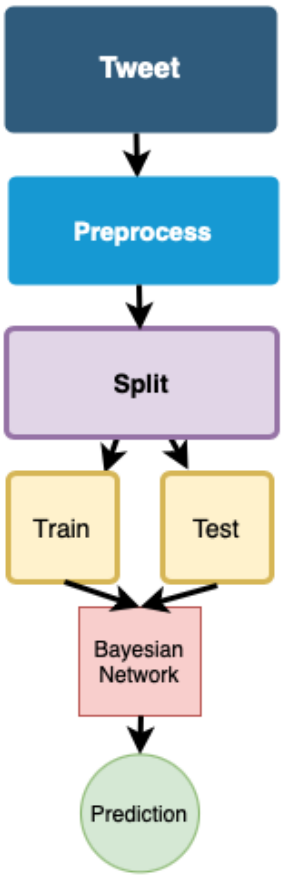


***Figura 14.*** *Pasos seguidos en el modelo de Redes Bayesianas*

| ***Metrics*** | ***V1.0. Text processing*** | ***V3.0. Text processing*** |
|---|---|---|
| ***Macro F1-Sarcasm*** | 0.89 | 0.84 |
| ***Macro F1-Irony*** | 0.26 | 0.20 |
| ***Macro F1-Satire*** | 0.38 | 0.51 |
| ***Macro F1-Overstatement*** | 0.17 | 0.48 |
| ***Macro F1-Understatement*** | 0.47 | 0.47 |
| ***Macro F1-Rhetorical Question*** | 0.12 | 0.20 |
| ***Macro F1-Score*** | 0.38 | 0.45 |

***Tabla 13.*** *Entrenamiento modelo Redes Bayesianas*

*Segundo modelo: BERT*

Como ya se comentó anteriormente, las representaciones codificadoras bidireccionales a partir de *Transformers* (BERT) son una técnica de aprendizaje automático basada en *Transformers* para el pre-entrenamiento del procesamiento del lenguaje natural (PLN). El modelo *Bert-base-uncased* se pre-entrena a partir de datos no etiquetados extraídos de *BooksCorpus* [44] que tienen 800 millones de palabras y de Wikipedia en inglés con 2500 millones de palabras.

BERT utiliza *Transformers* como mecanismo de atención que aprende las relaciones contextuales entre las palabras (o sub-palabras) dentro de un texto. Transformers incluyen dos mecanismos separados: un codificador que lee la entrada del texto y un decodificador que produce una predicción de la etiqueta.

Para este modelo, se utilizó la estrategia de relevancia binaria (BR) [45], la cual se basa en dividir el proceso de aprendizaje del conjunto de datos en un conjunto de tareas de clasificación binaria, es decir, una por cada etiqueta. El principal inconveniente de esta estrategia es que la BR ignora cualquier dependencia de las etiquetas y podría fallar en la predicción de alguna combinación de etiquetas que presenta alguna dependencia.

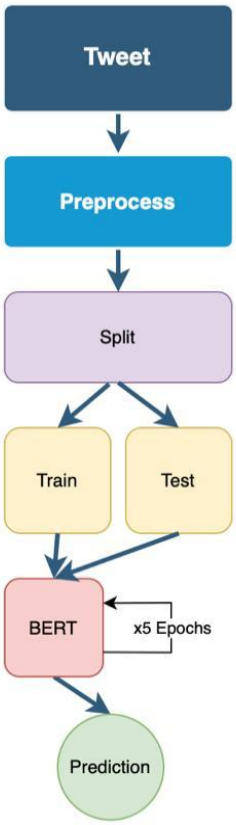


***Figura 15.*** *Pasos seguidos en el modelo BERT*

Se ha entrenado este segundo modelo con un lote de 32 instancias y 5 epochs. En la Figura 15 se puede observar la estrategia que sigue este modelo.

En cuanto al tercer y último envío, el cual es basado en *BERT-base-uncased*, se utilizaron diferentes enfoques. La Tabla 14 muestra los resultados obtenidos en la fase entrenamiento. Los enfoques fueron:

- v1.0. No se aplicó nada adicional.
- v2.0. Versión anterior + aumento de datos en las clases minoritarias.
- v3.0 Versiones anteriores + inserción de datos de una base de datos externa.
- v4.0 Versiones anteriores + v2.0 del preprocesamiento de texto
- v5.0 Versiones anteriores + v3.0 del preprocesamiento de texto

| ***Metrics*** | ***v1.0*** | ***v2.0*** | ***v3.0*** | ***v4.0*** | ***v5.0*** |
|---|---|---|---|---|---|
| ***Macro F1-Sarcasm*** | 0.45 | 0.78 | 0.81 | 0.80 | 0.70 |
| ***Macro F1-Irony*** | 0.45 | 0.72 | 0.77 | 0.77 | 0.64 |
| ***Macro F1-Satire*** | 0.49 | 0.63 | 0.66 | 0.64 | 0.57 |
| ***Macro F1-Overstatement*** | 0.50 | 0.60 | 0.63 | 0.62 | 0.52 |
| ***Macro F1-Understatement*** | 0.49 | 0.65 | 0.69 | 0.71 | 0.60 |
| ***Macro F1-Rhetorical Question*** | 0.84 | 0.90 | 0.91 | 0.89 | 0.87 |
| ***Macro F1-Score*** | 0.54 | 0.71 | **0.75** | 0.74 | 0.65 |

***Tabla 14.*** *Resultados entrenamiento BERT Transformers*

Tanto para los experimentos realizados con este modelo BERT, como para el primer modelo definido de Redes Bayesianas, los tweets se preprocesaron y luego se dividieron aleatoriamente utilizando un método estratificado (80% de entrenamiento y 20% de test). Esto significa que la proporción de valores en las muestras producidas es la misma que la proporción de valores proporcionada gracias a el parámetro a estratificar (*stratify*).

Observando las Tablas 13 y 14, se puede ver que en la v3.0 de la Tabla 14, se obtiene la mejor *Macro F1-score*. Así que ese fue nuestro último y definitivo envío.

## 3.5 Resultados de la etapa de evaluación

Cuando comenzó la etapa de evaluación, los organizadores de la competición habilitaron un conjunto de datos compuesto únicamente de los tweets de los que teníamos que hacer predicción. También se habilitó un enlace donde subir las predicciones a través de la plataforma *Codalab*[9].

Los modelos se entregaron de manera secuencial en el mismo orden en el que se han ido explicando a lo largo de esta memoria ya que, la competición, permitía subir todos los modelos que quisiéramos siempre y cuando se estuviese dentro del rango de la fecha de evaluación.

En este apartado solamente se ha expuesto la *F1-score* porque era la métrica con la que se evaluaba en la competición. Concretamente la *F1-score* de la clase a predecir (etiqueta=1).

Según las métricas oficiales, como se puede apreciar en la Tabla 15, se consiguió una *Macro F1-score* de 0.0699, obteniendo el puesto 10 de los 22 equipos que han participado en esta subtarea.

| ***Macro F1-score*** | ***F1 Sarcasm*** | ***F1 Irony*** | ***F1 Satire*** | ***F1 Understatement*** | ***F1 Overstatement*** | ***F1 Rhetorical question*** |
|---|---|---|---|---|---|---|
| *0.0699* | 0.2430 | 0.0485 | 0.0000 | 0.0000 | 0.0000 | 0.1280 |

***Tabla 15.*** *Resultados obtenidos en la competición*

Cabe destacar que obtuvimos la segunda mejor puntuación en la etiqueta *rhetorical question*, en la cual la mejor puntuación de todos los participante fue de 0.1556 y la tercera fue de 0.1091.

También obtuvimos la tercera mejor puntuación en la etiqueta *sarcasm*, en la cual la segunda mejor puntuación de esa etiqueta fue de 0.2480.

Analizando nuestros modelos, podemos afirmar que el principal problema encontrado en la subtarea fue el desequilibrio en el conjunto de datos provocado la falta de datos, por lo que hemos estado aplicando constantemente el aumento de datos en las clases minoritarias e incluso insertando datos de una base de datos externa. Aplicando estas dos técnicas, se observa una gran mejora en el rendimiento de nuestros modelos.

[9] https://codalab.lisn.upsaclay.fr

Además, nuestra investigación demuestra que cualquier tipo de técnica de preprocesamiento de texto empeora los resultados, ya que cualquier carácter o letra mayúscula, podría ser el factor determinante para reconocer el discurso irónico.

Aunque los resultados no fueron nada malos en comparación con el resto de los participantes, se optó por seguir trabajando para poder mejorarlos. Los conocimientos adquiridos durante la participación en la competición y el margen de mejora existente nos motivaron para desarrollar nuevos modelos y estrategias que se describen en los siguientes apartados.

# 4 Propuestas de mejora

Una vez finalizada la competición, los organizadores aportaron a los equipos el conjunto de datos de test etiquetado, el cual estaba formado por los tweets que proporcionaron durante la etapa de evaluación con sus respectivas etiquetas de discurso irónico.

Ante esta oportunidad, se decidió seguir trabajando para aportar nuevas propuestas e intentar mejorar los modelos entregados en la competición. El objetivo era de mejorar los resultados que se obtuvieron en el ranking de la competición.

En este apartado, se describen las diferentes propuestas de mejora que se han llevado a cabo.

## 4.1 Descripción del marco de experimentación

En este subapartado se expone cuáles han sido las líneas de trabajo que se han seguido en los intentos de mejora.

En primer lugar, las Tablas 16, 17, 18, 19, 20 y 21 reflejan la distribución de los tweets con respecto a las categorías en el conjunto de datos de test etiquetado proporcionado por los organizadores y está compuesto por un total de 1400 tweets.

| Sarcasm | N.º de tweets |
|---|---|
| **0** | 1220 |
| **1** | 180 |

***Tabla 16.*** *Distribución tweets con etiqueta "Sarcasm" del fichero test*

| Irony | N.º de tweets |
|---|---|
| **0** | 1380 |
| **1** | 20 |

***Tabla 17****. Distribución tweets con etiqueta "Irony" del fichero test*

| Satire | N.º de tweets |
|---|---|
| **0** | 1351 |
| **1** | 49 |

***Tabla 18****. Distribución tweets con etiqueta "Satire" del fichero test*

| Understatement | N.º de tweets |
|---|---|
| **0** | 1399 |
| **1** | 1 |

***Tabla 19.*** *Distribución tweets con etiqueta "Understatement" del fichero test*

| Overstatement | N.º de tweets |
|---|---|
| **0** | 1390 |
| **1** | 10 |

***Tabla 20.*** *Distribución tweets con etiqueta "Overstatement" del fichero test*

| Rhetorical_question | N.º de tweets |
|---|---|
| **0** | 1389 |
| **1** | 11 |

***Tabla 21.*** *Distribución tweets con etiqueta "Rhetorical_question" del fichero test*

En segundo lugar, hay que indicar que se han aplicado otros dos modelos similares a BERT como *RoBERTa-base* y *RoBERTa-large*, los cuáles, como se ha comentado anteriormente, son una mejora con respecto a BERT.

Los modelos de RoBERTa se han aplicado con los siguientes valores de los hiperparámetros:

- Número de *epochs* = 3
- *early_stopping* con *patience* = 1
- *learning_rate* = $5 \cdot 10^{-5}$

En tercer lugar, en cada modelo de RoBERTa se han realizado dos estrategias de aumento de datos usando la librería *nlpaug*:

- Data Augment Tipo 1: Sustituye una palabra de manera aleatoria de un tweet por otra palabra completamente aleatoria del corpus WordNet.

- Data Augment Tipo 2: Este es el mismo que se usó en la competición. Sustituye una palabra aleatoria de un tweet por otra sinónima del corpus WordNet.

En cuarto lugar, a cada modelo de RoBERTa se ha aplicado las mismas técnicas que se aplicaron en la competición: preprocesamiento del texto simple (eliminación de menciones, links, fotos y videos), ya que para los transformers es importante mantener expresiones y palabras porque aprenden el contexto de las palabras e inserción de la misma base de datos externa, pero con *46* nuevos tweets pertenecientes a la clase *Satire*.

**Inciso:** En las tablas que aparecen en los siguientes apartados (Tablas 22, 23, 24 y 25) se puede apreciar que cuando se hace aumento de datos en la etiqueta *Sarcasm,* la celda

aparece de varios colores: naranja, verde o gris oscuro. Esto se debe a que si se observa la distribución que tiene la etiqueta *Sarcasm* en el conjunto de entrenamiento (154 tweets no sarcásticos y 713 tweets sarcásticos) se da el caso contrario al resto de las etiquetas, es decir, tenemos más tweets de la clase a predecir. Este desbalanceo de la clase a predecir puede provocar que el modelo no termine de aprender correctamente, por lo que ha habido veces que se ha obtenido mejores resultados haciendo aumento de datos en la clase 0 (celdas naranjas), otras veces se ha obtenido mejores resultados haciendo aumento de datos en la clase 1 (celdas verdes) o incluso se ha obtenido el mismo resultado haciendo el aumento de las 2 clases (celdas grises oscuras).

## 4.2 Modelo ROBERTA-BASE

### 4.2.1 Data Augment Tipo 1

| ***Metrics*** | ***Competición*** | ***Baseline*** | ***AugmentTipo1*** | ***AugmentTipo1 + DB Externa*** | ***AugmentTipo1 + DB Externa + Preprocesamiento*** |
|---|---|---|---|---|---|
| ***F1 Sarcasm*** | 0.2430 | 0.2278 | 0.2464 | 0.2306 | 02403 |
| ***F1 Irony*** | 0.0485 | 0.0000 | 0.0526 | 0.0737 | 0.0000 |
| ***F1 Satire*** | 0.0000 | 0.0000 | 0.0000 | 0.0930 | 0.0000 |
| ***F1 Understatement*** | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| ***F1 Overstatement*** | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| ***F1 Rhetorical Question*** | 0.1280 | 0.0952 | 0.1224 | 0.0808 | 0.0727 |
| ***Macro F1-Score*** | 0.0699 | 0.0538 | 0.0702 | **0.0797** | 0.0522 |

***Tabla 22.*** *Resultados RoBERTa-base con Augment Tipo 1*

En la Tabla 22 se puede observar que el mejor valor de *Macro F1-score* se obtiene haciendo el aumento de datos de tipo 1 obteniéndose un valor de 0.0797. Con este resultado ya se mejora al obtenido en la competición.

Si tratamos cada etiqueta de manera individual, se observa que el mejor resultado de la etiqueta *Sarcasm* se obtiene haciendo solamente el aumento de datos, ya que ni con la inserción de la base de datos externa ni con el preprocesamiento se ha conseguido mejorar los resultados.

Sin embargo, con la inserción de la base de datos externa si se ha conseguido un mejor resultado de la etiqueta *Irony* y *Satire* con respecto a los obtenidos en la competición.

Con respecto a las etiquetas *Understatement*, *Overstatement* y *Rhetorical question*, no se ha conseguido mejorar los resultados con respecto a la competición.

### 4.2.2 Data Augment Tipo 2

| ***Metrics*** | ***Competición*** | ***Baseline*** | ***AugmentTipo2*** | ***AugmentTipo2 + DB Externa*** | ***AugmentTipo2 + DB Externa + Preprocesamiento*** |
|---|---|---|---|---|---|
| ***F1 Sarcasm*** | 0.2430 | 0.2278 | 0.2377 | 0.2434 | 0.2431 |
| ***F1 Irony*** | 0.0485 | 0.0000 | 0.0000 | 0.0000 | 0.0246 |
| ***F1 Satire*** | 0.0000 | 0.0000 | 0.0000 | 0.0628 | 0.0566 |
| ***F1 Understatement*** | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| ***F1 Overstatement*** | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| ***F1 Rhetorical Question*** | 0.1280 | 0.0952 | 0.1238 | 0.1085 | 0.0681 |
| ***Macro F1-Score*** | 0.0699 | 0.0538 | 0.0603 | **0.0691** | 0.0654 |

***Tabla 23.*** *Resultados RoBERTa-base con Augment Tipo 2*

En la Tabla 23 se puede observar que el mejor valor *Macro F1-score* se obtiene haciendo solamente el aumento de datos tipo 2 con un valor de 0.0691. No obstante, no se consigue mejorar el resultado obtenido en la competición.

Hablando de cada etiqueta individualmente, se observa que en la etiqueta *Sarcasm* con el aumento de datos de datos tipo 2 y con la inserción de la base de datos externa si se consigue una ligera mejoría con respecto al resultado obtenido en la competición.

Con respecto al resto de las etiquetas, no se ha conseguido mejorar ninguna métrica haciendo uso de ninguna de las técnicas enunciadas.

## 4.3 Modelo ROBERTA-LARGE

### 4.3.1 Data Augment Tipo 1

| ***Metrics*** | ***Competición*** | ***Baseline*** | ***AugmentTipo1*** | ***AugmentTipo1 + DB Externa*** | ***AugmentTipo1 + DB Externa + Preprocesamiento*** |
|---|---|---|---|---|---|
| ***F1 Sarcasm*** | 0.2430 | 0.2278 | 0.2278 | 0.2278 | 0.2278 |
| ***F1 Irony*** | 0.0485 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| ***F1 Satire*** | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| ***F1 Understatement*** | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| ***F1 Overstatement*** | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| ***F1 Rhetorical Question*** | 0.1280 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| ***Macro F1-Score*** | 0.0699 | **0.0380** | **0.0380** | **0.0380** | **0.0380** |

***Tabla 24.*** *Resultados RoBERTa-large con Augment Tipo 1*

En la Tabla 24 se puede observar que las técnicas que se van aplicando al modelo no reflejan mejorías, sino que se mantienen los valores constantes.

Con respecto a la etiqueta *Sarcasm*, se puede observar que cuando aplicas el modelo por defecto (sin usar ninguna técnica extra), si se mantiene el valor con respecto a cuando se usa *RoBERTa-base*, pero no refleja ningún tipo de incremento o decremento de las métricas en ninguna de las posteriores técnicas aplicadas.

### 4.3.2 Data Augment Tipo 2

| *Metrics* | *Competición* | *Baseline* | *AugmentTipo2* | *AugmentTipo2 + DB Externa* | *AugmentTipo2 + DB Externa + Preprocesamiento* |
|---|---|---|---|---|---|
| ***F1 Sarcasm*** | 0.2430 | 0.2278 | 0.2278 | 0.2278 | 0.2278 |
| ***F1 Irony*** | 0.0485 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| ***F1 Satire*** | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| ***F1 Understatement*** | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| ***F1 Overstatement*** | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| ***F1 Rhetorical Question*** | 0.1280 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| ***Macro F1-Score*** | 0.0699 | **0.0380** | **0.0380** | **0.0380** | **0.0380** |

***Tabla 25.*** *Resultados RoBERTa-large con Augment Tipo 2*

En la Tabla 25, al igual que en la Tabla 24, se pueden observar que las aplicaciones de las diferentes técnicas al modelo no reflejan mejorías, sino que se vuelven a mantener los valores constantes.

Por lo tanto, tras haber realizado la experimentación al completo podemos decir que los modelos que usan *RoBERTa-large* no son adecuados para esta tarea porque en todas las ejecuciones las métricas se han mantenido constantes. Las experimentaciones satisfactorias han sido en las que se ha usado el modelo *RoBERTa-base*, es donde la aplicación de las diferentes técnicas ha hecho que se mejoren los resultados con respecto a la competición.

Cabe destacar que se ha conseguido mejorar individualmente el resultado de las etiquetas *Sarcasm, Irony* y *Satire*, donde esta última ha sido de manera notable. Esto se debe a que, con la adición de la base de datos externa, esta etiqueta ha sufrido un aumento notorio del número de tweets catalogados como *Satire* y junto con el aumento de los datos se ha conseguido incluso superar el resultado de otra etiqueta como *Irony*, la cual contaba con más muestras en el conjunto de datos por defecto.

## 4.4 Métricas del mejor modelo

Aunque el mejor *Macro F1-score* se ha obtenido usando el modelo *RoBERTa-base* con las técnicas de aumento de datos tipo 1 + inserción de base de datos externa, al haber estado siguiendo la estrategia de relevancia binaria (BR) anteriormente explicada, podemos considerar las mejores métricas de manera individual e independiente al modelo y/o técnicas empleadas.

Por lo tanto, las mejores métricas de manera individual son:

- *Sarcasm*: Modelo *RoBERTa-base*. Técnica: aumento de datos tipo 1.
- *Irony*: Modelo *RoBERTa-base*. Técnicas: aumento de datos tipo 1 + inserción de base de datos externa.
- *Satire*: Modelo *RoBERTa-base*. Técnicas: aumento de datos tipo 1 + inserción de base de datos externa.
- *Understatement*: No se ha conseguido mejorar las métricas de esta etiqueta.
- *Overstatement*: No se he conseguido mejorar las métricas de esta etiqueta.
- Rhetorical Question: No se ha conseguido mejorar las métricas de esta etiqueta.

En la Tabla 26 se detallan los nuevos resultados obtenidos con las mejoras que se han aplicado junto con los resultados que se obtuvieron en la competición, para así poder comparar.

| | ***Macro F1-score*** | ***F1 Sarcasm*** | ***F1 Irony*** | ***F1 Satire*** | ***F1 Understatement*** | ***F1 Overstatement*** | ***F1 Rhetorical question*** |
|---|---|---|---|---|---|---|---|
| **Competición** | 0.0699 | 0.2430 | 0.0485 | 0.0000 | 0.0000 | 0.0000 | 0.1280 |
| ***Mejoras*** | 0.0902 *(+29.04%)* | 0.2464 (+1.40%) | 0.0737 (+51.96%) | 0.0930 (+∞) | 0.0000 (+0.00%) | 0.0000 (+0.00%) | 0.1280 (+0.00%) |

***Tabla 26.*** *Comparativa de los resultados obtenido en la competición con las mejoras realizadas y porcentaje de mejora*

Haciendo primero una comparativa individual en cada una de las etiquetas, hemos mejorado la métrica de la etiqueta *Sarcasm*, aunque no se ha escalado ninguna posición en la clasificación de esa etiqueta. Por otra parte, hemos conseguido una mejora también en la etiqueta *Irony*, donde se ha escalado desde la posición 11ª a la 8ª posición en su respectivo ranking. Por último, pero no menos importante porque es lo que más

se ha conseguido mejorar, hemos pasado de ser los 11ª en la etiqueta *Satire* (lo cual quiere decir que solamente 10 participantes habían podido desarrollar un modelo el cuál no tuviese una puntuación de 0) a ser 3º en esa etiqueta.

Aunque con el resto de las etiquetas (*Understatement*, *Overstatement* y *Rhetorical question*) no se ha conseguido ninguna mejoría, cabe destacar que en *Rhetorical question* ya se obtuvo la segunda mejor puntuación en la competición por lo que era complicado mejorarla partiendo de una *baseline*. Por otra parte, con las que si se ha conseguido, ha significado un incremento también en la *Macro F1-score*. La nueva *Macro F1-score* conseguida nos sitúa en un 2º puesto del ranking general de la competición, por lo tanto, se puede afirmar que las propuestas de mejora han sido un éxito puesto que no solo se ha mejorado la marca anterior, si no que se ha escalado desde la 10º posición a la 2ª de manera virtual.

En la Tabla 27 se observa las diferentes métricas que se han obtenido de los nuevos resultados con respecto a la clase a predecir por cada etiqueta.

| ***Etiqueta*** | ***Precision*** | ***Recall*** | ***F1-score*** | ***Matriz de confusión*** |
|---|---|---|---|---|
| ***Sarcasm*** | 0.1438 | 0.8611 | 0.2464 | $\begin{bmatrix} 297 & 923 \\ 25 & 155 \end{bmatrix}$ |
| ***Irony*** | 0.0406 | 0.4000 | 0.0737 | $\begin{bmatrix} 1191 & 189 \\ 12 & 8 \end{bmatrix}$ |
| ***Satire*** | 0.1081 | 0.0816 | 0.0930 | $\begin{bmatrix} 1318 & 33 \\ 45 & 4 \end{bmatrix}$ |

***Tabla 27.*** *Métricas completas de los nuevos resultados*

Como se puede observar en las matrices de confusión de las diferentes etiquetas en casi ninguna de las matrices se predice bien ni la clase positiva ni negativa exceptuando la clase positiva de *Sarcasm* (de ahí a que tenga un buen recall). Esto se debe a la falta de datos para entrenar que tiene el conjunto de datos sumado al desbalanceo que tienen las clases.

No obstante, a pesar de obtenerse resultados tan bajos, con las propuestas de mejora se hubiese obtenido el segundo puesto en la clasificación. Estos valores indican la enorme dificultad que tenía la tarea propuesta por la organización *SemEval2022*.

# 5 Conclusiones y trabajo futuro

En este apartado se exponen las conclusiones a las que se ha llegado con la realización de este trabajo y, seguidamente, se plantean futuras líneas de investigación para poder mejorar los resultados obtenidos.

## 5.1 Conclusiones

Con la realización de este trabajo, se ha demostrado una vez más la importancia del PLN a la hora de identificar patrones de comportamiento, en nuestro caso la detección del sarcasmo intencionado en el ámbito de las redes sociales. No obstante, aparte de poder usar el PLN para identificar factores sociales y actitudes, también puede usarse en reconocimiento y análisis del habla e incluso de reconocimiento de imágenes.

El ganador de la competición obtuvo una *Macro F1-score* de 0.1630 y nosotros, después de las propuestas de mejora hemos conseguido escalar a la segunda posición virtual con una *Macro-F1 score* de 0.0902. Esto quiere decir que queda mucho margen de mejora para resolver esta tarea porque el análisis del sarcasmo en redes sociales tiene una gran complejidad ya que no se conoce la intención del emisor del mensaje.

Los mejores resultados se consiguieron con la utilización de *RoBERTa-base*, el cuál es un modelo que funciona mejor que el que usamos en la competición (BERT), junto con un conjunto de técnicas de aumento de datos e inserción de base de datos externa para cada etiqueta.

En el anexo de este trabajo se adjunta el artículo científico publicado con la consecución de este trabajo, el cuál aparece en actas del "*16th International Workshop on Semantic Evaluation*".

Por último, se ha demostrado la importancia de aplicaciones de técnicas de balanceo en conjunto de datos desbalanceados, la importancia de buscar recursos externos en caso de escasez de datos y la facilidad que nos aportan los *Transformers* a este tipo de tareas.

## 5.2 Trabajo futuro

Planteamientos de trabajos futuros sobre esta tarea existen una infinidad, pero los principales problemas nos quedan claro: la falta de datos en el conjunto de datos de entrenamiento y test, y su desbalaceamiento. Al igual que se ha hecho con la etiqueta *Satire*, se podría hacer un uso completo de la base de datos externa que se ha ido utilizando con otras etiquetas que cuentan con potencial en su conjunto de datos estándar, como por ejemplo *Rhetorical Question* (esto también podría dar lugar a poder aplicar otras técnicas de balanceo como el *UnderSampling*).

Otra futura experimentación futura sería la utilización de la técnica de *Underbagging*, la cual consiste en entrenar distintos conjuntos de *Undersampling*, para después someterlos a una votación en donde se combinan y promedian las predicciones que se han obtenido en una única predicción. El uso de esta técnica aporta una mayor probabilidad a la hora de obtener mejores resultados, ya que las predicciones no dependen solamente de un único modelo entrenado.

Finalmente, y puesto que el área de la PLN está en auge y existen numerosos recursos a disposición de los investigadores, sería interesante trabajar con otros modelos basados en Transformers que estén pre-entrenados con textos con contenido sarcástico.

# 6 Bibliografía


[1] J. Thanaki, *Python natural language processing*. Birmingham, England: Packt Publishing, 2017.

[2] M. Á. Camacho-Álvarez and E. Navarro-Álvarez, "Procesamiento del lenguaje natural con Python," *Revista de Computo Aplicado*, pp. 24–28, 2020.

[3] I. Abu Farha, S. V. Oprea, S. Wilson, and W. Magdy, "SemEval-2022 task 6: iSarcasmEval, intended sarcasm detection in English and Arabic," in *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, 2022.

[4] T. M. Mitchell, "Machine Learning," *Ufpe.br*. [Online]. Available: https://www.cin.ufpe.br/~cavmj/Machine%20-%20Learning%20-%20Tom%20Mitchell.pdf. [Accessed: 15-Sep-2022].

[5] *Wikipedia.org*. [Online]. Available: https://es.wikipedia.org/wiki/Aprendizaje_automático. [Accessed: 15-Sep-2022].

[6] H. C. Arteaga, "Técnicas de aprendizaje supervisado y no supervisado para el aprendizaje automatizado de computadoras," in *Memorias del primer Congreso Internacional de Ciencias Pedagógicas: Por una educación integral, participativa e incluyente*, 2015, pp. 549–564.

[7] E. M. Rojas, "Machine Learning: análisis de lenguajes de programación y herramientas para desarrollo," *RISTI - Rev. Ibér. Sist. Tecnol. Inf.*, vol. 28, pp. 586–599, 2020.

[8] J. Machicao, "El aprendizaje reforzado: un punto de encuentro entre las políticas públicas y la inteligencia artificial." Unpublished, 2020.

[9] K. M. Leung, "Naive Bayesian Classifier," *Nyu.edu*, 2007. [Online]. Available: https://cse.engineering.nyu.edu/~mleung/FRE7851/f07/naiveBayesianClassifier.pdf. [Accessed: 15-Sep-2022].

[10] "ENSEÑANZA DEL TEOREMA DE BAYES CON APOYO TECNOLÓGICO," *Ugr.es*. [Online]. Available: https://www.ugr.es/~batanero/pages/ARTICULOS/Thales2006.pdf. [Accessed: 15-Sep-2022].

[11] J. Han, M. Kamber, and J. Pei, “Data Mining: Concepts and Techniques (3 rd ed.) -Chapter 1,” *188.242:8080*. [Online]. Available: http://14.99.188.242:8080/jspui/bitstream/123456789/12925/1/Module%201%20%26%202.pdf. [Accessed: 15-Sep-2022].

[12] M. Gorini, “¿Cuál es la diferencia entre el machine learning y el deep learning?,” *Bismart.com*. [Online]. Available: https://blog.bismart.com/diferencia-machine-learning-deep-learning. [Accessed: 15-Sep-2022].

[13] G. Chassagnon, M. Vakalopolou, N. Paragios, and M.-P. Revel, “Deep learning: definition and perspectives for thoracic imaging,” *Eur. Radiol.*, vol. 30, no. 4, pp. 2021–2030, 2020.

[14] OpenWebinars, “DIFERENCIAS ENTRE MACHINE LEARNING Y DEEP LEARNING,” 02-Oct-2019. [Online]. Available: https://www.youtube.com/watch?v=EEe9xcrq2-o. [Accessed: 15-Sep-2022].

[15] T. Mikolov, M. Karafiát, L. Burget, J. Cernocký, and S. Khudanpur, “Recurrent neural network based language model,” *In Interspeech*, vol. 2, no. 3, pp. 1045–1048, 2010.

[16] B. Lindemann, B. Maschler, N. Sahlab, and M. Weyrich, “A survey on anomaly detection for technical systems using LSTM networks,” *Comput. Ind.*, vol. 131, no. 103498, p. 103498, 2021.

[17] A. Vaswani *et al.*, “Attention is all you need,” *arXiv [cs.CL]*, 2017.

[18] R. Kulshrestha, “Transformers in NLP: A beginner friendly explanation,” *Towards Data Science*, 29-Jun-2020. [Online]. Available: https://towardsdatascience.com/transformers-89034557de14. [Accessed: 15-Sep-2022].

[19] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional Transformers for language understanding,” *arXiv [cs.CL]*, 2018.

[20] Y. Liu *et al.*, “RoBERTa: A robustly optimized BERT pretraining approach,” *arXiv [cs.CL]*, 2019.

[21] D. Karani, “Introduction to word embedding and Word2Vec,” *Towards Data Science*, 01-Sep-2018. [Online]. Available: https://towardsdatascience.com/introduction-to-word-embedding-and-word2vec-652d0c2060fa. [Accessed: 15-Sep-2022].

[22] X. Rong, “word2vec Parameter Learning Explained,” *Arxiv.org*, 2016. [Online]. Available: https://arxiv.org/pdf/1411.2738.pdf. [Accessed: 15-Sep-2022].

[23] *Baeldung.com*. [Online]. Available: https://www.baeldung.com/cs/word-embeddings-cbow-vs-skip-gram. [Accessed: 15-Sep-2022].

[24] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” *arXiv [cs.CL]*, 2013.

[25] L. Puente-Maury, A. López-Chau, W. Cruz-Santos, and L. López-García, “Método rápido de preprocesamiento para clasificación en conjuntos de datos no balanceados,” *Research in Computing Science*, vol. 73, no. 1, pp. 129–142, 2014.

[26] profesorDATA, “Cómo manejar datos desbalanceados en Aprendizaje Automático,” *profesordata.com*, 16-Jul-2020. [Online]. Available: https://profesordata.com/2020/07/16/como-manejar-datos-desbalanceados-en-aprendizaje-automatico/. [Accessed: 15-Sep-2022].

[27] *Upm.es*. [Online]. Available: https://oa.upm.es/68623/1/TFM_GUILLEM_GARCIA_SUBIES.pdf. [Accessed: 15-Sep-2022].

[28] E. Ma, “Data Augmentation library for text,” *Towards Data Science*, 20-Apr-2019. [Online]. Available: https://towardsdatascience.com/data-augmentation-library-for-text-9661736b13ff. [Accessed: 15-Sep-2022].

[29] G. A. Miller and C. Fellbaum, “WordNet then and now,” *Lang Resources & Evaluation*, vol. 41, no. 2, pp. 209–214, 2007.

[30] A. Krouska, C. Troussas, and M. Virvou, “The effect of preprocessing techniques on Twitter sentiment analysis,” in *2016 7th International Conference on Information, Intelligence, Systems & Applications (IISA)*, 2016, pp. 1–5.

[31] C. Khanna, "Text pre-processing: Stop words removal using different libraries," *Towards Data Science*, 10-Feb-2021. [Online]. Available: https://towardsdatascience.com/text-pre-processing-stop-words-removal-using-different-libraries-f20bac19929a. [Accessed: 15-Sep-2022].

[32] P. Carvalho, L. Sarmento, M. J. Silva, and E. de Oliveira, "Clues for detecting irony in user-generated contents: Oh...!! it's 'so easy' ;-)," in *Proceeding of the 1st international CIKM workshop on Topic-sentiment analysis for mass opinion - TSA '09*, 2009.

[33] B. C. Wallace, D. K. Choe, L. Kertz, and E. Charniak, "Humans Require Context to Infer Ironic Intent (so Computers Probably do, too)," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2014.

[34] John S. Leggitt & Raymond W. Gibbs, "Emotional Reactions to Verbal Irony", Discourse Processes, 2000.

[35] N. Madnani, "Getting started on natural language processing with Python," *Crossroads*, vol. 13, no. 4, pp. 5–5, 2007.

[36] I. Sarhan, P. Mosteiro, and M. Spruit, "UU-Tax at SemEval-2022 Task 3: Improving the generalizability of language models for taxonomy classification through data augmentation," in *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, 2022.

[37] J. Hao and T. K. Ho, "Machine learning made easy: A review of Scikit-learn package in Python programming language," *J. Educ. Behav. Stat.*, vol. 44, no. 3, pp. 348–361, 2019.

[38] H. Stepanek, "Introduction," in *Thinking in Pandas*, Berkeley, CA: Apress, 2020, pp. 1–7.

[39] S. V. Stehman, "Selecting and interpreting measures of thematic classification accuracy," *Remote Sens. Environ.*, vol. 62, no. 1, pp. 77–89, 1997.

[40] J. P. McCrae, A. Rademaker, F. Bond, E. Rudnicka, and C. Fellbaum, "English WordNet 2019 -- An Open-Source WordNet for English," in *Proceedings of the 10th Global Wordnet Conference*, 2019, pp. 245–252.

[41] S. Oprea and W. Magdy, "iSarcasm: A Dataset of Intended Sarcasm," *arXiv [cs.CL]*, 2019.

[42] E. Alzahrani and L. Jololian, “How different text-preprocessing techniques using the BERT model affect the gender profiling of authors,” 2021.

[43] D. Heckerman and M. P. Wellman, “Bayesian networks,” *Communications of the ACM*, vol. 38, p. 27+, Mar-1995.

[44] J. Bandy and N. Vincent, “Addressing ‘documentation debt’ in machine learning research: A retrospective datasheet for BookCorpus,” 2021.

[45] O. Luaces, J. Díez, J. Barranquero, J. J. del Coz, and A. Bahamonde, “Binary relevance efficacy for multilabel classification,” *Prog. Artif. Intell.*, vol. 1, no. 4, pp. 303–313, 2012.

# Anexo

## Paper *SemEval* 2022 Task 6: Intended Sarcasm in English using Deep Learning Techniques

**I2C at SemEval-2022 Task 6:**
**Intended Sarcasm in English using Deep Learning Techniques**

**Adrián Moreno Monterde**
Escuela Técnica Superior de Ingeniería
Universidad de Huelva (Spain)
adrian.moreno521@alu.uhu.es

**Laura Vázquez Ramos**
Escuela Técnica Superior de Ingeniería
Universidad de Huelva (Spain)
laura.vazquez005@alu.uhu.es

**Victoria Pachón Álvarez**
Escuela Técnica Superior de Ingeniería
Universidad de Huelva (Spain)
vpachon@dti.uhu.es

**Jacinto Mata Vázquez**
Escuela Técnica Superior de Ingeniería
Universidad de Huelva (Spain)
mata@uhu.es

**Abstract**

Sarcasm is often expressed through several verbal and non-verbal cues, e.g., a change of tone, overemphasis in a word, a drawn-out syllable, or a straight looking face. Most of the recent work in sarcasm detection has been carried out on textual data. This paper describes how the problem proposed in *Task 6: Intended Sarcasm Detection in English* (Abu Arfa et al. 2022) has been solved. Specifically, we participated in *Subtask B: a binary multi-label classification task*, where it is necessary to determine whether a tweet belongs to an ironic speech category, if any. Several approaches (classic machine learning and deep learning algorithms) were developed. The final submission consisted of a BERT based model and a macro-F1 score of 0.0699 was obtained.

### 1 Introduction

Existing social media analysis systems are limited by their inability to accurately detect and interpret figurative language. Sarcasm is often used by individuals to express opinions on complex matters and regarding specific targets (Carvalho et al. 2009).

Early computational models for verbal and irony and sarcasm detection have relied on shallow methods exploiting conditional token count regularities. But lexical clues alone are insufficient to discern sarcasm intent. Appreciating the context of expression is critical for this; even for humans (Wallace et al. 2014). Indeed, the exact same sentence can be interpreted as literal or sarcastic, depending on the speaker. Consider the sarcastic tweet in Figure 1 (ignoring for the moment the attached *#sarcasm* hashtag). Without knowing the author's political inclination, it would be difficult to conclude with certainty whether the tweet was intended as sarcastic or not.


Figure 1: An illustrative tweet.

This task is about the binary classification of tweets in English based on the category of ironic speech. As a Multilabel-Classification, a tweet can belong to multiple categories or none. This research is important because social networking sites have changed our lives in recent years. For companies, this social reality has turned into an obligation to set up communication and marketing channels through social networks. Besides, companies require feedback from consumers, through comments or messages that mark the acceptance or rejection of each of the proposals, products or services.

For this competition 2 models/strategies were used:

- The first model is a binary multilabel classifier using Bayesian networks.
- The second model and final submit on the competition consisted of six different binary classifiers using BERT. In other words, one for each evaluable label: sarcasm, irony, satire,

understatement, overstatement and rhetorical question.

In this task we have analysed the sarcastic behaviour of people on social networks, in this case twitter, and the ways in which people express it. The dataset proposed by the organization was very unbalanced, so we had to apply several data balancing techniques in order to achieve good results.

The rest of this paper is organized as follows: in section 2 we explain the dataset, the meaning of the labels and data distribution. Therefore, we refer to other research that has helped us in our approach to this one. In section 3, several techniques and methods applied to our models to improve their performance and metrics are described. In section 4 we explain the libraries used and their usefulness. Finally, in section 5, the scores obtained with our proposed approaches are presented.

## 2 Background

As mentioned above, this paper is focused on Subtask B: binary multilabel classification. The original dataset has 3467 tweets in English with a maximum size of 280 characters (tweet limit). As a matter of fact, only the first 867 tweets will be useful for our task because the rest of the tweets do not have the labels that we are going to work with. Furthermore, the columns "Unnamed:0", "rephrase" and "sarcastic" have been removed because they are useless for our task perfomance.

For a better analysis and understanding of the multiple labels that we have in our dataset it is important to mention the *ironic speech* exposed in (Leggitt and Gibbs 2000):

1. *Sarcasm*: tweets that contradict the state of affairs and are critical towards an addressee
2. *Irony*: tweets that contradict the state of affairs. but are not obviously critical towards an addressee.
3. *Satire*: tweets that appear to support an addressee but contain underlying disagreement and mocking.
4. *Understatement*: tweets that undermine the importance of the state of affairs they refer to
5. *Overstatement*: tweets that describe the state of affairs in terms that are obviously exaggerated.
6. *Rhetorical question*: tweets that include a question whose invited inference (implication) is obviously contradicting the state of affairs.

In Figure 2, two samples of the dataset with the information used in our approaches can be seen.

**Tweet:** The only thing I got from college is a caffeine addiction
**Sarcasm:** [0]
**Irony:** [1]
**Satire:** [0]
**Understatement:** [0]
**Overstatement:** [0]
**Rhetorical question:** [0]

**Tweet:** do i just blast maneskin to get hyped for my osce or??
**Sarcasm:** [1]
**Irony:** [0]
**Satire:** [0]
**Understatement:** [0]
**Overstatement:** [0]
**Rhetorical question:** [1]

Figure 2: Example of two rows

When it comes to the multiple labels (categories) of sarcasm, Table 1 shows the distribution of the tweets into these categories. As can be seen, the dataset is imbalanced so, in the next section, we explain how the dataset was balanced for a better performance.

| Category | Number of tweets |
|---|---|
| Sarcasm | 713 |
| Irony | 155 |
| Satire | 25 |
| Understatement | 10 |
| Overstatement | 40 |
| Rhetorical question | 101 |

Table 1: Distribution of tweets in each category

This challenge has been approached by different researchers. In (Davidov et al., 2010), experiments with semi-supervised sarcasm identification on a Twitter dataset (5.9 million tweets) were carried out using 50 Twitter tags and 15 emojis as sentiment labels. They used a 5-fold cross validation on their classifier getting a F1-score of 0.55.

In addition, in (Tsur et al.,2010), they propose a semi supervised system for sarcasm recognition over 66,000 products reviews from Amazon. They used the same strategy as in the previous mention and obtained and F-score of 0.83 on the product reviews dataset.

More recently, other approaches have been developed to solve the task of sarcasm detection. In (Ashwita et al. 2021), the authors experimented by varying the amount of context used along with the response (text to be classified) and found that including the last utterance in the dialogue along with the response improved the performance of their system.

In (Khatri, P, Pranav, y M, Dr. Anand Kumar 2020), a model using machine learning techniques with BERT and GloVe embeddings to detect sarcasm in tweets was proposed.

## 3 System Overview

This section describes the two types of models that were submitted and the techniques and methods applied to each model to improve their performance and metrics.

### 3.1 Data augmentation

One of the main problems with the dataset is the small number of tweets to train our models (only 867 tweets). To solve this, a data augmentation technique was applied. In particular, a synonym augmenter (Wordnet, English) (McCrae et al. 2019) was used to create a new tweet but only swapping one random word by its synonym and keeping their labels. An example of this technique can be seen in Table 2.

| | |
|---|---|
| **Original tweet** | The quick brown fox jumps over the lazy dog |
| **New tweet** | The quick gray fox jumps over the lazy dog |

Table 2: Example of data augmentation

We suggest applying this technique only once because our model could overfitting the data and could yield overrated results of the metrics.

### 3.2 External databases

Another technique applied in the proposed models for the data augmentation was the manual insertion of tweets and labels (Oprea and Magdy 2019). Most of the tweets inserted belong to minority labels (we can see the minority classes on Table 1) such as satire, overstatement or understatement. Finally, once the new tweets were manually added, the dataset consisted of 904 tweets.

### 3.3 Text Processing

We have applied three versions of text processing to clean and simplify the text based on the work described in (Alzahrani and Jolonian 2021).

**Text processing v1.0:** this is the most basic pre-process. For this version, the following guidelines were applied:

- Conversion of all characters to lowercase.
- Extent of all possible contractions in English (e.g., what's  what is).
- Removal of emojis.
- Removal of special characters.
- Removal of multiple spaces between characters.

**Text processing v2.0:** this is the intermediate version. In addition to the features described at v1.0, the following features were added:

- Removal of emojis made from keyboard characters
- Removal of mentions
- Removal of links

**Text processing v3.0:** this is the full version. In addition to the features presented in v2.0, a removal of stopwords in English was added.

### 3.4 First model: Bayesian networks

The first model that was developed involves the classic algorithm of Bayesian networks (Heckerman and Wellman 1995) to study the pattern of behavior that the categories of sarcasm may present in our dataset.

We used a naive Bayes classifier (NBC) which assumes that the attributes are independent of each other. That is to say, the probability can be obtained by calculating the product of the individual conditional probabilities of each attribute given the class node as it can be seen on Figure 3.

In this model, an input (a single tweet) is provided, and it returns a vector of size six (one for each tag) with the predicted label.

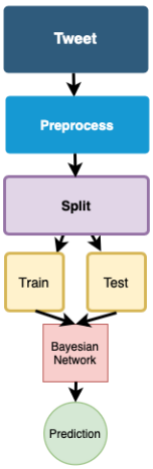


Figure 3: Steps followed on the first model

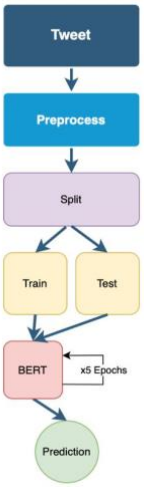


Figure 4: Steps followed on the second model

### 3.5 Second model: BERT

Bidirectional Encoder Representations from Transformers (BERT) is a transformer-based machine learning technique for natural language processing (NLP) pretraining developed by (Devlin et al. 2018). *BERT-base-uncased* model is pretrained from unlabeled data extracted from BooksCorpus (Bandy and Vincent 2021) which have 800M of words and from English Wikipedia with 2,500M of words.

BERT uses Transformers (Wolf et al. 2019) as an attention mechanism that learns contextual relations between words (or sub-words) within a text. Transformers includes two separate mechanisms: an encoder that reads the text input and a decoder that produces a prediction of the label.

For this model, the binary relevance (BR) strategy (Luaces et al. 2012) was used, which splits the learning process of the dataset into a sets of binary classification tasks, in other words, one per label. The main disadvantage of this strategy is that BR ignores any label dependency and could fail in predicting some combination of labels that presents any dependency.

We have trained our second model with a batch of 32 instances and 5 epochs. Figure 4 represents the strategy of this model.

## 4 Experimental Setup

To obtain the above models, some libraries were used:

- For the data augmentation, the *nltk* library (Wang and Hu 2021, 1041-1049).
- For padding the sequences of the inputs id's, the *keras* library.
- *Sklearn* library for the metrics and splitting the dataset (Hao and Ho 2019).
- *Pandas* library for working with dataframes (Stepanek 2020).
- *Transformers* library for everything related with BERT.

For all the experiments, tweets were preprocessed and then they were randomly split using a stratified method (80% training and 20% testing). That means that the proportion of values in the samples produced is the same as the proportion of values provided for the parameter to stratify.

During the training phase of the competition, we have only focused on the macro-F1 score as the key metric of the Subtask B.

## 5 Results

Two submissions were sent using Bayesian Networks: the first one with the text processing v1.0 and the second one with the text processing v3.0. Table 3 shows the results obtained during the training phase.

| Metrics | v1.0. Text processing | v3.0. Text processing |
|---|---|---|
| F1 Sarcasm | 0.89 | 0.84 |
| F1 Irony | 0.26 | 0.20 |
| F1 Satire | 0.38 | 0.51 |
| F1 Overstatement | 0.17 | 0.48 |
| F1 Understatement | 0.47 | 0.47 |
| F1 Rhetorical Question | 0.12 | 0.20 |
| Macro F1-Score | 0.38 | 0.45 |

Table 3: Results obtained using Bayesian Networks

| Metrics | v1.0 | v2.0 | v3.0 | v4.0 | v5.0 |
|---|---|---|---|---|---|
| F1 Sarcasm | 0.45 | 0.78 | 0.81 | 0.80 | 0.70 |
| F1 Irony | 0.45 | 0.72 | 0.77 | 0.77 | 0.64 |
| F1 Satire | 0.49 | 0.63 | 0.66 | 0.64 | 0.57 |
| F1 Understatement | 0.50 | 0.60 | 0.63 | 0.62 | 0.52 |
| F1 Overstatement | 0.49 | 0.65 | 0.69 | 0.71 | 0.60 |
| F1 Rhetorical Question | 0.84 | 0.90 | 0.91 | 0.89 | 0.87 |
| Macro F1-Score | 0.54 | 0.71 | **0.75** | 0.74 | 0.65 |

Table 4: Results obtained using BERT

Regarding the final submission using BERT-base-uncased, different approaches were used. Table 4 shows the results obtained during the training phase. The approaches were:

- **v1.0.** Nothing extra applied
- **v2.0.** Previous versions + Data Augmentation in minority class only.
- **v3.0.** Previous versions + insert data of an external database.
- **v4.0.** Previous versions + v2.0 of text processing.
- **v5.0.** Previous version + v3.0 of text processing.

Taking a look at Table 3 and Table 4, can be seen that in v3.0 of Table 4, the best macro F1-score is obtained. So that was our final submission.

According to the official metrics, as was mentioned before, we achieved a macro F1-score of 0.0699 and we were ranked 10$^{th}$ among 22 teams that participated on this subtask.

Analyzing our systems, we can state that the main problem found in the subtask was the lack of data towards unbalanced data at the dataset, which is why we have been constantly applying data augmentation on the minority classes and even inserting data from an external database. Applying these two techniques, a big improvement in the performance of our systems can be seen.

Furthermore, our research shows that any kind of preprocessing technique is mostly useless because any character, capital letter, overextended word or symbol, could be the determining factor in recognizing ironic speech.

## 6 Conclusion

In this paper our approach to solve *Task 6 (iSarcasmEval) – Subtask B: Given a text, determine which ironic speech category it belongs to, if any; in English*, has been described.

Our best result was reached with a deep learning algorithm (BERT) model, with which we achieved a macro F1-score of 0.0699. We obtained the 10th position in the ranking.

For future works, an improved version of our BERT model could be developed by training with a bigger dataset. It is also possible to look for new preprocessing techniques that enable the removal of information that is useless to the meaning of the tweet but still maintain the ironic speech patterns (if any).

## References

Abu Farha, Ibrahim, Silviu Oprea, Steven Wilson, Walid Magdy. "SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic". In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics, 2022.

Carvalho, Paula, Luís Sarmento, Mário J. Silva, and Eugénio de Oliveira. 2009. "Clues for Detecting Irony in User-Generated Contents: Oh...!! It's 'so Easy' ;-)." In *Proceeding of the 1st International*

*CIKM Workshop on Topic-Sentiment Analysis for Mass Opinion - TSA '09*. New York, New York, USA: ACM Press.

Wallace, Byron C., Do Kook Choe, Laura Kertz, and Eugene Charniak. 2014. "Humans Require Context to Infer Ironic Intent (so Computers Probably Do, Too)." In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 512–16. Stroudsburg, PA, USA: Association for Computational Linguistics.

Leggitt, John S., and Raymond W. Gibbs. 2000. "Emotional Reactions to Verbal Irony." *Discourse Processes* 29 (1): 1–24. https://doi.org/10.1207/s15326950dp2901_1.

Dmitry Davidov, Oren Tsur, and Ari Rappoport. 2010. Semi-supervised recognition of sarcastic sentences in twit- ter and amazon. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, CoNLL '10, pages 107–116, Stroudsburg, PA, USA. Association for Computational Linguistics.

Oren Tsur, Dmitry Davidov, and Ari Rappoport. 2010. ICWSM - a great catchy name: Semi-supervised recog- nition of sarcastic sentences in online product reviews. In William W. Cohen and Samuel Gosling, editors, *Proceedings of the Fourth International Conference on Weblogs and Social Media, ICWSM 2010, Washington, DC, USA, May 23-26, 2010*. The AAAI Press.

Oprea, Silviu, and Walid Magdy. 2019. "ISarcasm: A Dataset of Intended Sarcasm." *ArXiv [Cs.CL]*. https://paperswithcode.com/paper/isarcasm-a-dataset-of-intended-sarcasm.

Alzahrani, Esam, and Leon Jololian. 2021. "How Different Text-Preprocessing Techniques Using the BERT Model Affect the Gender Profiling of Authors." *ArXiv [Cs.CL]*. http://arxiv.org/abs/2109.13890.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. "BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding." CoRR abs/1810.04805.

Bandy, Jack, and Nicholas Vincent. 2021. "Addressing 'Documentation Debt' in Machine Learning Research: A Retrospective Datasheet for BookCorpus." *ArXiv [Cs.CL]*. http://arxiv.org/abs/2105.05241.

Wolf, Thomas, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, et al. 2019. "HuggingFace's Transformers: State-of-the-Art Natural Language Processing." ArXiv [Cs.CL]. http://arxiv.org/abs/1910.03771.

Luaces, Oscar, Jorge Díez, José Barranquero, Juan José del Coz, and Antonio Bahamonde. 2012. "Binary Relevance Efficacy for Multilabel Classification." *Progress in Artificial Intelligence* 1 (4): 303–13. https://doi.org/10.1007/s13748-012-0030-x.

Wang, Meng and Fanghui Hu. 2021. "The Application of NLTK Library for Python Natural Language Processing in Corpus Research." Theory and Practice in Language Studies 11 (9): 1041-1049. doi:10.17507/tpls.1109.09.

Hao, Jiangang and Tin Kam Ho. 2019. Machine Learning made Easy: A Review of Scikit-Learn Package in Python Programming Language. Vol. 44. Los Angeles, CA: SAGE Publications. doi:10.3102/1076998619832248. https://journals.sagepub.com/doi/full/10.3102/1076998619832248

Stepanek, Hannah. 2020. Thinking in Pandas : How to use the Python Data Analysis Library the Right Way. Berkeley, CA: Apress. doi:10.1007/978-1-4842-5839-2. https://library.biblioboard.com/viewer/087e187a-b660-11ea-a44d-0a7fc7c4e64f

McCrae, John Philip, Alexandre Rademaker, Francis Bond, Ewa Rudnicka, and Christiane Fellbaum. 2019. "English WordNet 2019 – An Open-Source WordNet for English." In *Proceedings of the 10th Global Wordnet Conference*, 245–52.

Ashwitha, Shruthi, Shruthi, Makarand Upadhyaya, Abhra Pratip Ray, y Manjunath. 2021. "Sarcasm Detection in Natural Language Processing". Materials Today: Proceedings 37: 3324–31. https://doi.org/10.1016/j.matpr.2020.09.124.

Khatri, Akshay, P, Pranav, y M, Dr. Anand Kumar. 2020. "Sarcasm detection in tweets with BERT and GloVe embeddings". https://doi.org/10.48550/ARXIV.2006.11512.

Heckerman, David, and Michael P. Wellman. "Bayesian networks." *Communications of the ACM*, 1995, 38(3), 27-31