Procesamiento del lenguaje natural con tweets de desastres.

Deep Learning
Estudiantes: Giloc Delgado Cadavid, Juan Diego Quintero Urrea
Ingeniería de Sistemas

Resumen:
El procesamiento de lenguaje natural (NLP) ha surgido como una herramienta crucial para analizar grandes cantidades de datos no estructurados, como en este caso particular, los mensajes en redes sociales. El análisis de tweets utilizando técnicas de deep learning se ha vuelto esencial en la detección de desastres y eventos de emergencia en tiempo real. Los modelos de deep learning pueden identificar patrones sutiles en el lenguaje de los tweets que podrían indicar e identificar los desastres/catástrofes presentando oportunidades significativas, como identificar alertas tempranas de los desastres, pero esto también requiere una comprensión profunda de los desafíos inherentes como lo sería principalmente la detección de desastres basada únicamente en el lenguaje pudiendo generar falsos positivos, ya que algunos eventos podrían ser mencionados sin ser desastres reales.
El éxito de este procesamiento de lenguaje natural (NLP) depende en gran medida de la calidad y preparación de los datos. En el contexto de tweets para la detección de desastres, el preprocesamiento de datos junto con su limpieza desempeñará un papel crucial.

Introducción:
En la actualidad las plataformas de redes sociales han tomado mucha relevancia e importancia a la hora de informar y notificar a las personas sobre desastres, catástrofes, accidentes, que están ocurriendo. Esto ha generado en la comunidad científica la necesidad crítica de crear herramientas analíticas avanzadas. En este

contexto, el procesamiento de lenguaje natural (NLP) emerge como una herramienta esencial para extraer conocimientos valiosos de mensajes en redes sociales, con un enfoque particular en la detección de desastres y eventos de emergencia en tiempo real. Exploraremos la aplicación de técnicas de deep learning en el análisis de tweets, destacando tanto las oportunidades transformadoras como los desafíos inherentes. Por ejemplo, usaremos las Redes Neuronales(RNA) ya que son capaces de abordar muchos diversos problemas en el campo del Aprendizaje Automático. Las RNA tratan de emular el comportamiento y la capacidad de aprendizaje de las redes neuronales biológicas[1]. Por medio de Python junto con su versatilidad y flexibilidad para el procesamiento de datos, en este caso de lenguaje natural(NLP), nos enfrentamos al procesamiento y limpieza de tweets. Exploraremos no solo las oportunidades para una detección temprana de desastres, sino también los obstáculos que deben superarse, con un énfasis particular en la necesidad de comprender a fondo la ambigüedad del lenguaje y mitigar posibles falsos positivos. Este informe destila las complejidades de la aplicación práctica de NLP con el procesamiento de diferentes modelos y la implementación de redes neuronales, proporcionando una perspectiva integral sobre cómo esta disciplina revoluciona la capacidad de respuesta a eventos críticos en nuestra era digital.

Metodología:

- Estructura de los notebooks

Se tiene un notebook donde se realiza la exploración de los datos, con diferentes gráficos que nos ayudaron a identificar parámetros para la realización limpieza de los datos.
Y un segundo notebook con la limpieza de los datos, la preparación de los datos y la ejecución de los modelos de procesamiento de lenguaje natural con los resultados obtenidos.

- Arquitectura

Para los modelos que son variaciones de BERT se usó TensorFlow Hub para obtenerlos y se usaron como presets. En general siguen la siguiente arquitectura: la capa de entrada; seguido de una capa de preprocesado para los datos de entrada;

a partir de esta capa se introduce el modelo de TensorFlow Hub; a la salida se introduce una capa dropout y por último una capa densa para hacer la clasificación.

Para el modelo LTSM la capa de entrada fue dada por los embeddings, seguido de dos capas LSTM de 64 y 32 salidas y por último una capa densa para la clasificación.

- Recopilación de datos:

Se utilizó la plataforma Kaggle para acceder a conjuntos de datos relevantes para la detección de desastres en tweets. La base de datos seleccionada proporciona una amplia variedad de tweets etiquetados que abarcan situaciones de desastres y eventos de emergencia.
El total de datos a predecir es de 10875 tweets, que han sido clasificados manualmente, es decir, etiquetados por humanos como tweets sobre desastres reales o no. El tamaño de la dataset es de unos 1.3MB de información en los conjuntos de datos train.csv y test.csv.

- Preprocesamiento de datos:

En Google Colab(herramienta de compilación de python), se implementaron técnicas de preprocesamiento de datos para preparar los tweets para el análisis:

- Tokenización y normalización para descomponer los tweets en unidades significativas y homogeneizar el texto.
- Eliminación de stopwords y puntuación para centrarse en términos relevantes.
- Lematización para simplificar la representación del lenguaje.

- Iteraciones realizadas:
    - En la primera iteración se buscó generar un modelo a partir de la información entregada por la página de Kaggle para confirmar el correcto funcionamiento de librerías y modelos, específicamente el modelo DistilBERT y generar un primer resultado. No se hizo ningún tipo de procesado o limpieza de datos y se obtuvieron resultados acordes a los esperados, basados en los resultados obtenidos en

Kaggle. También se hizo la exploración de los datos en bruto para interiorizarlos.

- Después de investigar más acerca del funcionamiento de los modelos de procesamiento de lenguaje natural y de conocer un poco más acerca de los modelos basados en BERT se trataron de probar algunos como en BERT Base, en donde no pudimos completar el entrenamiento pues dado su gran número de parámetros entrenables (109.483.778), el hardware no pudo afrontar el proceso de entrenamiento. Otros modelos que se probaron fueron el Small BERT y el ALBERT.
- En otra iteración añadimos un preprocesado más profundo de datos, quitando caracteres y secuencias irrelevantes de los textos, por ejemplo emojis, urls, y también estandarizando el texto para que fuera todo en minúscula. Adicional a esto se quitaron las stopwords en idioma inglés usando nltk, y por último se lematizaron las palabras para tratar de obtener raíces comunes. Después se volvió a hacer una exportación de datos, ahora más limpios.
- En la última iteración se añadió también un modelo LTSM, por lo cual se tokenizaron los textos (también se añadió la tokenización para algunos de los modelos de BERT).

Resultados:
En general, todos los modelos implementados tuvieron un desempeño similar, exceptuando el BERT base puesto que no se pudo hacer el entrenamiento en este caso. Usando la métrica de accuracy en los datos de validación del entrenamiento, y con épocas que iban desde 2 hasta 4, se obtuvieron resultados entre el 80% y el 84%, siendo los dos mejores el DistilBERT y el ALBERT.

Cabe resaltar que los resultados no tuvieron una variación realmente significativa antes y después del preprocesado de datos, otorgando resultados similares entre ambos conjuntos de datos.

Conclusión:
Con el desarrollo del proyecto se logra entender que es indispensable un buen conocimiento de los datos y la calidad de la limpieza de datos, se posicionan como

un elemento indispensable para garantizar la precisión de los modelos durante el entrenamiento. La selección adecuada de modelos desempeña un papel fundamental, y en este contexto, se destaca que modelos de aprendizaje profundo como las redes neuronales, representadas por DistilBERT y ALBERT, demostraron ser altamente efectivos en la tarea de predicción de desastres en tweets.
Los buenos resultados del proyecto nos resalta la comprensión sólida de los datos, una limpieza efectiva, y la elección acertada de modelos avanzados, subrayando la importancia de abordar los aspectos clave en el desarrollo de modelos de procesamiento de lenguaje natural para la clasificación de tweets.

Obtención de datos
Descargar y subir los siguientes archivos train.csv test.csv para los datos a entrenar desde acá: Kaggle Data

Tener en cuenta a la hora de leer los csv del drive cargarlos desde la carpeta correcta, en nuestro caso de esta manera:

python

```
df_train = pd.read_csv("/content/drive/MyDrive/Colab Notebooks/Deep Learning/Proyecto/disaster_train.csv")
df_test = pd.read_csv("/content/drive/MyDrive/Colab Notebooks/Deep Learning/Proyecto/disaster_test.csv")
```

Ser paciente, el entrenamiento de los modelos puede ser demorado(3h-8h)

Referencias:

- Audevart, A. (2023) *Kerasnlp starter notebook disaster tweets, Kaggle.* Available at: https://www.kaggle.com/code/alexia/kerasnlp-starter-notebook-disaster-tweets
- Thite, S. (2023) *NLP disaster tweets using bert*, *Kaggle*. Available at: https://www.kaggle.com/code/sunilthite/nlp-disaster-tweets-using-bert

- Sevval, I. (2023) *Text classification with BERT - Disaster tweets.* Available at: https://www.kaggle.com/code/ilhansevval/text-classification-with-bert-disaster-tweets#☁%EF%B8%8FWord-Cloud-in-Train-and-Test-Data-