NETWORK NEWS: DESCIFRANDO LAS FAKE NEWS CON REDES NEURONALES PROFUNDAS. (marzo, 2024)

Daza Pereira Juan Pablo y Camargo Sanchez Juan Sebastian

Aprendizaje Profundo.

Escuela Colombiana de Ingeniería Julio Garavito

Resumen - En este artículo se aborda el problema de las noticias falsas en la era digital de los últimos años, enfocándose en su detección mediante el uso de redes neuronales profundas. Se utiliza un conjunto de datos de Kaggle llamado “Fake News Detection”, que consta de aproximadamente 40.000 datos de noticias, tanto falsas como verdaderas, después de un proceso de análisis y filtro de datos.

La metodología empleada implica un procesamiento de los datos, donde se realizó eliminación de atributos innecesarios y el análisis de los textos que contienen noticias para reducir algunos excesos. Se utiliza la tokenización para preparar los datos como entrada de la red neuronal.

Se diseña y entrena la red neuronal utilizando las funciones de activación ReLU para las capas intermedias de la red, y una función de activación Sigmoide para la salida. Durante el entrenamiento, se aplican técnicas de optimización y regularización para mejorar el rendimiento del modelo y evitar sobreajustes, para así tener un modelo definitivo.

El modelo ya entrenado, se espera tener una medida de precisión del 93.8%, decidiendo esa precisión si la técnica implementada fue o no la mejor para la detección de noticias falsas.

En resumen, este artículo presenta una solución nueva e innovadora que aborda el problema de las noticias falsas y los bots sociales, utilizando técnicas avanzadas de

la inteligencia artificial, teniendo un enfoque en el diseño y entrenamiento del modelo, y con los resultados obtenidos, destacar la importancia y viabilidad que tiene la inteligencia artificial en el tema de la reducción de la desinformación que se presenta actualmente.

Palabras clave **-** fake news, inteligencia artificial, redes neuronales profundas, entrenamiento, datos, detección, impacto, validación, entrenamiento, optimización, regularización.

I. INTRODUCCIÓN

En la actual era del siglo XXI, el fenómeno de las noticias falsas (Fake News) ha tenido un gran impacto. El surgimiento de las Fake News impulsadas por bots sociales, han sido una serie de amenazas para la integridad de la información y la sociedad en conjunto. La capacidad que estos bots sociales tienen para generar contenido convincente y engañoso a través de diferentes algoritmos, plantea un desafío significativo para la sociedad, la política, los medios de comunicación y su creencia.

Por ello, es esencial abordar este problema de manera precisa y eficaz, porque en el momento en que la información a fluir a través de diferentes medios, la veracidad y autenticidad se vuelve escaza.

Ya con este contexto, este artículo de investigación se propone presentar una solución nueva e innovadora para la solución de esta problemática haciendo uso de las redes neuronales profundas. Contamos con un conjunto de datos compuesto por aproximadamente 40.000 noticias, que abarca tanto noticias verdaderas como

falsas, que tendrán un riguroso proceso de entrenamiento de la red neuronal. Esta base de datos tendrá un trato cuidadoso donde se segmentará en tres conjuntos de entrenamiento (Train), desarrollo (Dev) y prueba (Test), lo anterior para garantizar eficacia y fiabilidad del modelo resultante.

Al implementar este desafío, queremos detectar, a través de la base de datos con la que contamos, mitigar la propagación de noticias falsas, y que a futuro este modelo pueda mejorar aplicando técnicas más avanzadas de inteligencia artificial para mitigar este problema de la desinformación.

II. METODOLOGÍA

En el desarrollo del proyecto, se abordó la problemática de la detección de noticias falsas mediante el uso de redes neuronales profundas. Para comenzar, se seleccionó y procesó un conjunto de datos de Kaggle llamado “Fake News Detection”. Este conjunto de datos inicialmente contaba con 44.954 datos, que incluían 23.537 datos de noticias falsas y 21.417 datos de noticias de datos verdaderos. Sin embargo, tras un proceso de filtrado y eliminación de datos sin información, se redujo a 40.000 datos viables para el entrenamiento de la red.

Durante este procesamiento de datos, se eliminaron atributos innecesarios como la fecha de publicación y el tema de la noticia, ya que no contribuían al proceso de entrenamiento de la red neuronal. Además, se realizó un análisis detallado de los textos de las noticias, identificando y recortando caracteres repetitivos para así, optimizar el rendimiento del modelo.

Posteriormente, como se realizó anteriormente, se realizo el procesamiento del texto de las noticias utilizando técnicas de tokenización para organizar y clasificar la frecuencia de las palabras. Esto dio como resultado una matriz de tokens que se utilizó como entrada para el diseño y entrenamiento de la red neuronal. Se implementó una arquitectura de red neuronal profunda, con funciones de activación

ReLU para las capas intermedias y una función de activación sigmoide para la capa de salida.

Ahora, durante el proceso de entrenamiento de la red neuronal, se piensa aplicar diversas técnicas de optimización y regularización para la mejorar el rendimiento y evitar el sobreajuste. Una vez se tenga entrenado el modelo, se espera obtener una medida de precisión del 93.8%.

Aparte de todo lo anterior, se realizó una pequeña investigación acerca de los Fake News, donde encontramos antecedentes de los bots sociales [1], y de como estos son capaces de viralizar noticias falsas por medio de combinaciones de valores diferenciales, de la adaptación que tienes estos bots al lenguaje natural mediante los textos ficticios, y la automatización que estos usan. Aparte de esta pequeña investigación, se realizó otra acerca del tema de la lucha contra esta situación [2], donde se hayo que hay estudios recientes para la detección automática de bots, donde usan técnicas de aprendizaje continuo y análisis de redes sociales mediante la extracción de comentarios y metadatos, y patrones temporales de actividad, para la detección de las noticias falsas que circulan en los medios.

Finalmente, ya con el resultado obtenido, se analizarán y se considerará la posibilidad de realizar un nuevo procesamiento de entrenamiento según las necesidades y observaciones halladas. Esta metodología proporciona un enfoque que aborda el problema de las noticias falsas (Fake News), destacando la efectividad de las redes neuronales profundas en la detección y mitigación de este fenómeno global.

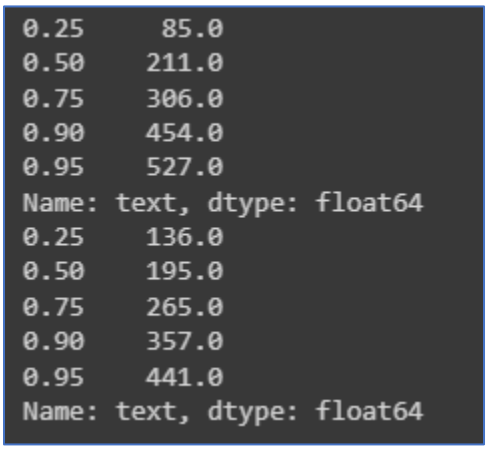
III. RESULTADOS

Tratamiento de datos

Empezamos analizando toda la data que tenemos. Se dividen en dos grandes data sets, Fake y True, cada una con más de veinte mil datos y que sumadas dan

44.689 cada una cuenta con cinco columnas. Como nuestro enfoque está en el contenido de una noticia, solo nos enfocaremos en la columna de texto de nuestra data. Hacemos un procesamiento para eliminar información duplicada y también eliminar información de valor nulo. Aunque la data esta balanceada de noticias tanto verdaderas como falsas, decidimos dejar ambas del mismo tamaño. Podemos observar el balance de número de caracteres del texto en la figura A. Creamos una nueva columna para identificar fácilmente las noticias falsas dándoles un 1. Concatenamos para procesar el texto. El proceso de esto está en la figura B.

Figura A

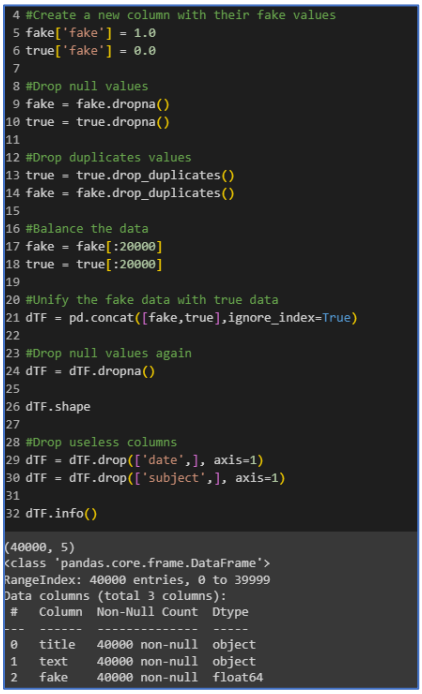


Porcentaje de datos con respecto al número de caracteres.

Ya después de hacer una limpieza a nuestra data queda es procesar el texto para que pueda ser entendida por una red neuronal. Para eso usamos una función de tokens, que va a asignarle un identificador del vocabulario en inglés, a cada palabra para después convertir todo el texto en una matriz de TF-IDF (expresa lo relevante que es una palabra en un documento o texto, el cual forma parte de un corpus.). Que si puede ser leída como entrada de una red. Antes de eso debemos

limpiar todos lo que no sea una palabra y llevar cada palabra a su raíz (sin conjugaciones, diminutivo, etc.). Una vez se procesa todos los textos lo vectorizamos par a la entrada a la red La limpieza está en la figura C.

Figura B



```
#Create a new column with their fake values
fake['fake'] = 1.0
true['fake'] = 0.0

#Drop null values
fake = fake.dropna()
true = true.dropna()

#Drop duplicates values
true = true.drop_duplicates()
fake = fake.drop_duplicates()

#Balance the data
fake = fake[:20000]
true = true[:20000]

#Unify the fake data with true data
dTF = pd.concat([fake,true],ignore_index=True)

#Drop null values again
dTF = dTF.dropna()

dTF.shape

#Drop useless columns
dTF = dTF.drop(['date',], axis=1)
dTF = dTF.drop(['subject',], axis=1)

dTF.info()
```

```
(40000, 5)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 40000 entries, 0 to 39999
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   title   40000 non-null  object
 1   text    40000 non-null  object
 2   fake    40000 non-null  float64
```

Figura C

```
from nltk.stem import SnowballStemmer

# SnowballStemmer
stemmer = SnowballStemmer('english')

def stem_text(text):
    # Tokenize the input text into individual words
    tokens = nltk.word_tokenize(text)

    # Stem each token using the SnowballStemmer
    stemmed_tokens = [stemmer.stem(token) for token in tokens]

    # Join the stemmed tokens back into a single string
    return ' '.join(stemmed_tokens)

# Apply stemming to the text data
dTF['text']=dTF['text'].apply(stem_text)
```

La función stem_text genera el token de la palabra y la ubica en una matriz TF-IDF.

Algunos métodos de limpieza de data y método para hacer tokens son tomados de los trabajos de SHOYOMBO RAPHAEL y ATRIJ TALGERY. Sobre sus trabajos en esta base de datos.

Ya teniendo esto automatizamos el proceso de dividir nuestra data en 80/10/10 para Training, Developin y Testing, respectivamente. La figura D muestra la ejecución y cantidad de datos para cada entrada.

Figura D

```
#Prepare the data Stage: Training, Developing and Testing
y=dTF['fake']

X_train, X_devtest, y_train, y_devtest = train_test_split(X, y, test_size=0.2, random_state=42)

X_trainStage, X_trainTestStage, y_trainStage, y_trainTestStage = train_test_split(X_train, y_train, test_size=0.2, random_state=42)

X_dev,X_test,y_dev,y_test = train_test_split(X_devtest, y_devtest, test_size=0.5, random_state=42)

X_devStage,X_devTestStage,y_devStage,y_devTestStage = train_test_split(X_dev, y_dev, test_size=0.2, random_state=42)

X_testStage,X_testTestStage,y_testStage,y_testTestStage = train_test_split(X_test, y_test, test_size=0.2, random_state=42)
```

```
Data for every Stage:
Data for training (32000, 158267)
Data for dev (4000, 158267)
Data for testing (4000, 158267)
Data for used in the stage :
Data (32000, 158267) for training stage  (25600, 158267) (6400, 158267)
Data (4000, 158267) for dev stage  (3200, 158267) (800,)
Data (4000, 158267) for dev stage  (3200, 158267) (800,)
```

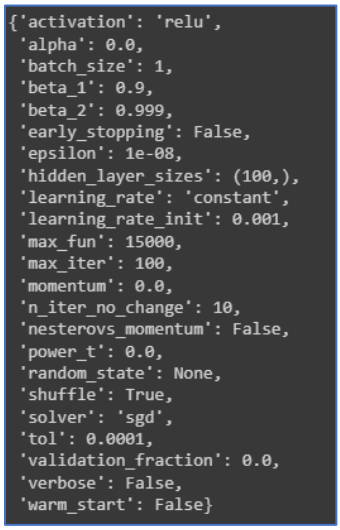
El código muestra como dividimos la data por etapas y la consola nos muestra el tamaño por cada etapa junto a su entrenamiento 80/20 en cada etapa.

Creación del modelo

Para la creación del modelo de nuestra red decidimos utilizar sklearn como librería, por la manera intuitiva de usarla y su documentación completa. Usaremos su modelo de MLPClassifier , que es Multy Layer Perceptron Classifier. Un modelo de clasificación, con una red a capas con un perceptrón (Aunque esto se puede modificar a más neuronas) y para clasificación.

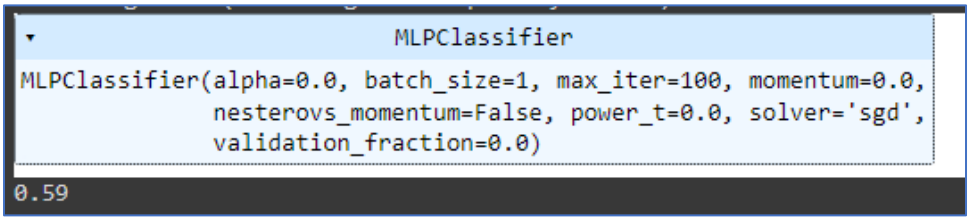
Construiremos un modelo básico sin ninguna optimización y regularización aplicada. En la figura E vemos en más detalle la configuración de hiperparametros para el modelo.

Figura E



Training

El primer resultado de la experimentación con este modelo duro más de dos horas y media y además sin tener ningún resultado ya que el entorno de ejecución Colab llego a su límite. Por lo que empezaremos por ejecutar el modelo y terminar su ejecución a la hora. Este fue el primer resultado:



Por lo que no está acorde con nuestra métrica de 93,8 que elegimos, modificamos el modelo optimizándolo. Aplicamos mini batch, momentum, nesterovs momentum, cambiamos el valor de rango de aprendizaje y cada capa tendrá 3 2 perceptrones más. Lo que nos dio como resultado:

```
MLPClassifier
MLPClassifier(alpha=0.0, hidden_layer_sizes=(100, 3), max_iter=100, power_t=0.0,
              solver='sgd', validation_fraction=0.0)
0.9540625
```

Ya con un 95.40 podremos avanzar a la etapa de desarrollo.

Developing

Ejecutamos el modelo para developing. Y los resultados fueron los siguientes:

```
1 #Test dev
2 MLPC_modelTrain.score(X_dev, y_dev)
0.9555
```

La precisión sigue siendo mayor a la métrica por lo que seguimos a la siguiente fase de Testing.

Testing

Ejecutamos el modelo para testing. Y los resultados fueron los siguientes:

```
1 #Test Training
2 MLPC_model.score(X_test, y_test)
0.959
```

La precisión sigue siendo mayor a la métrica cumpliendo con nuestro objetivo de proyecto.

IV. CONCLUSIONES

En el transcurso de este proyecto, logramos alcanzar los resultados esperados gracias a un buen desarrollo que se hizo en el modelo, un modelo que fue respaldado por buenas técnicas de optimización. Aunque la regularización no fue empleada directamente, el proceso de desarrollo en etapas y la aplicación de la optimización contribuyeron a la mejora de la precisión del modelo.

La implementación de esta red neuronal, especializada en la detección de noticias falsas representa una posible solución para esta problemática. Al usar la red feedforward de clasificación, hemos proporcionado una herramienta efectiva para identificar y distinguir entre noticias falsas y verdaderas, lo que contribuye en algo a los conocimientos sobre la detección de desinformación.

Nuestros resultados ofrecen una base sólida para futuras investigaciones en este campo, destacando la importancia de seguir explorando este tema y poder abordar más a fondo para buscar nuevos enfoques innovadores en el área de la inteligencia artificial y de la sociedad digital.

V. REFERENCIAS

[1]: ¿Cómo se viralizan las noticias falsas? Algoritmos y bots sociales.

Tomado de: https://www.iic.uam.es/innovacion/como-se-viralizan-noticias-falsas-algoritmos-bots-sociales/

[2]: Cómo funcionan los bots (y cómo contribuyen a difundir información falsa).

Tomado de: https://ijnet.org/es/story/c%C3%B3mo-funcionan-los-bots-y-c%C3%B3mo-contribuyen-difundir-informaci%C3%B3n-falsa

[3]: Librería de Sklean MLPClassifier.

Tomado de: https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html#sklearn.neural_network.MLPClassifier