Modelos de Aprendizaje Profundo

Carlos Eduardo García Lemus
Universidad Galileo
Ciudad de Guatemala
carlos.garcialemus@galileo.edu

Abstract

Deep Learning es una de las técnicas que se han beneficiado de los avances de la tecnología, permitiendo generar modelos end-to-end que se asemejan al modelo de aprendizaje humano. De esta forma el proyecto busca aplicar tres de los principales tipos de redes neuronales en el área de aprendizaje supervisado. Generando análisis de regresión basados en modelos MLP (Perceptron multi capa), clasificación de basura por medio de análisis de imágenes en redes CNN (Red neural convolucional), y finalmente aplicando modelos secuenciales para el procesamiento de texto y análisis de sentimiento a través de modelos RNN (Redes neurales recurrentes). Todo esto aplicando un enfoque experimental que evalua y ajusta parámetros para alcanzar un resultado superior al 85%.

1 Perceptron Multicapa (MLP)

1.1 Dataset

El dataset utilizado para el modelo MLP es tomado de:

`https://www.kaggle.com/mirichoi0218/insurance`

Este datset contiene información sobre 1,338 personas entre las edades de 18 a 64 años, la cual se describe a través de 7 distintas variables. Las variables independientes presentan información sobre su género, uso de cigarrillo, region, hijos, edad e índice de masa corporal (BMI), con el fin de describir la variables dependiente que estima los cargos asociados al seguro de salud de la persona.

1.2 Metodología

El modelo parte del análisis de los datos en los cuales se observa que es necesario realizar la codificación de las variables categóricas presentes (género, region, uso de cigarrillo), para lo cual se hace uso de codificación One-Hot. Adicionalmente, dado las distintas dimensionalidades presentes para cada una de las variables la data es escalada a través de normalización. Una vez la data ha sido preprocesada se aplica un train-test split que conserva 70% de los datos para entrenamiento y 30% de estos para evaluación, aplicado un split adicional en los datos de entrenamiento que toma el 20% de estos para validación durante el entrenamiento.

Para la generación del modelo se aplica una red de tipo MLP la cual, tras experimentar con distintos valores de hiperparámetros y arquicturas, se define con la siguiente estructura:

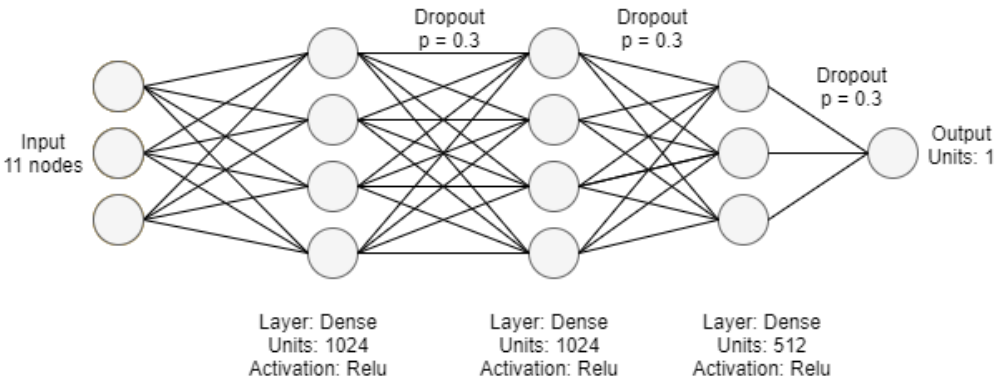

Figure 1: Arquitectura MLP

En el modelo final la red neural es entrenada durante 50 epochs aplicando mini-batch size de 64. Adicionalmente, el modelo aplica early stopping a través de callbacks en TensorFlow, para detener la ejecución y evitar overfitting.

1.3 Resultados

Los resultados para los dos modelos evaluados son los siguientes:

Table 1: Resultados MLP

Model ID	R^2	MSE
1	0.8404	22799556.49
2	0.8509	21305968.76

A partir de la siguiente tabla se selecciona el modelo número 2 como el mejor modelo entrenada para el esquema MLP al cumplir con un coeficiente de determinación R^2 = 85.09%

2 Red Convolucional (CNN)

2.1 Dataset

El dataset utilizado para el modelo CNN es tomado de:

`https://www.kaggle.com/asdasdasasdas/garbage-classification`

Este dataset contiene 2,527 imágenes de 6 diferentes tipos de residuos, con dimensiones de 512x384 pixeles. Las categorías de residuos presentadas en el dataset permiten la clasificación de:

- Cartón
- Vidrio
- Metal
- Papel
- Plástico
- Basura

2.2 Metodología

Como parte del preprocesamiento fue necesario crear un script de Python que permitiera reordenar las imágenes de forma que se mantuviera una estructura de directorios que separa las imágenes en sets de entrenamiento, validación y evaluación, cada uno con sus respectivos directorios

pertenecientes a las 6 categorías de residuos. Esto con el fin de facilitar el flujo de los datos al utilizar *keras.preprocessing.image.ImageDataGenerator* el cual se beneficia de la estructura anteriormente mencionada.

Dada la limitada cantidad de imágenes pertencientes al set de entrenamiento, luego de realizar el split de los datos para los distintos set de datos, se utiliza Síntesis de Imágenes para generar variaciones en los datos y permitir distintas visualizaciones de las imágenes de entrenamiento, a la vez que se escalan (150x150) y normalizan.

Tras algunos intentos de generar la red neural desde una construcción inicial, se llegó a la conclusión de necesitar un mayor poder computacional y tiempo para poder entrenar una red que cumpliera con las condiciones necesarias del proyecto, por lo cual el modelo presentado aplica Transfer Learning tomando ventaja de los pesos entrenados por la arquitectura de red InceptionV3. De esta forma los pesos de esta arquitectura son congelados durante el entrenamiento y el modelo elaborado para el presente proyecto integra dos capas ocultas totalmente conectadas y una capa de salida para la clasificación de imágenes.

Basado en las ideas presentadas a continuación se presenta el esquema final de la arquitectura utilizada:

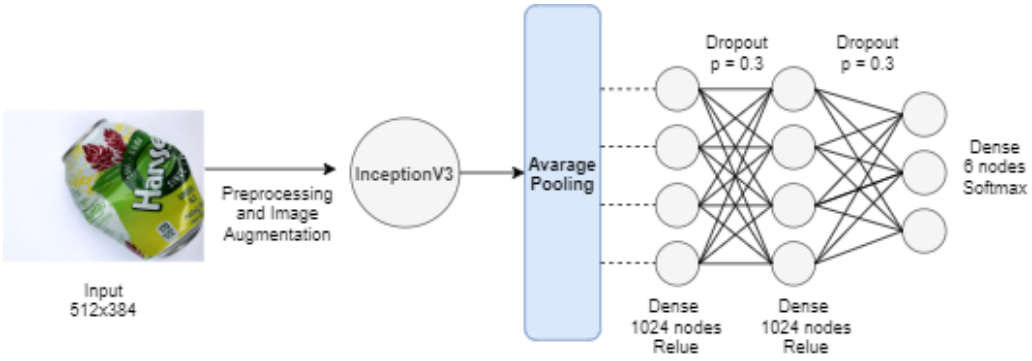

Figure 2: Arquitectura CNN

El modelo final se entrena por 30 epochs aplicando mini-batch size de 64, logrando un accuracy del 88% para los datos de validación.

2.3 Resultados

Los resultados para los cuatro modelos evaluados son los siguientes:

Table 2: Resultados CNN

Model ID	Accuracy
1	65.66%
2	77.03%
3	84.69%
4	85.61%

A partir de la siguiente tabla se selecciona el modelo número 4 como el mejor modelo entrenada para el esquema CNN al cumplir con un accuracy = 85.61%

3 Red Recurrente (RNN)

3.1 Dataset

El dataset utilizado para el modelo RNN es tomado por parte del autor Luis Fernando Leal, a partir de un proceso de recolección de tweets de diferentes países latinoamericanos durante los primeros meses de la pandemia provocado por el virus SARS-CoV-2 (2020).

Este dataset contiene 10 archivos perteneciente a distintos país de latinoamérica para más de 3 millones de tweets registrados. De los datos presentados se destacan la existencia de emoticones en el texto los cuales son utilizados como indicadores de emociones, permitiendo completar el proceso de análisis de sentimiento a partir del estudio de estos datos.

3.2 Metodología

El preprocesamiento de los datos abarca la generación de un dataset consolidado que incluye la información de los 10 países sobre los cuales se recolectaron los tweets. Una vez generado el dataset consolidado se procedió a eliminar todos aquellos comentarios vacíos ya que no aportan valor al análisis.

A partir de este punto se trabaja en la detección de emoticones como parte de los caracteres especiales identificados, generando un listado de 11 clases de distintos estados representado por varios emoticones como se demuestra en la siguientes tabla:

Figure 3: Mapeo sentimientos-emoticones

A partir de este diccionario de emoticones y emociones se trabaja en la extracción de emoticones en los textos, asi como las operaciones de procesamiento de lenguaje natural (NLP), las cuales consisten en la normalización y eliminación de stopwords y caracteres especiales de los comentarios.

Para finalizar el preprocesamiento de los datos se aplican las operaciones de tokenización y padding, para generar las secuencias numéricas que servirán como datos de entrada en el modelo implementado.

El modelo secuencial utilizado aplica una capa de embedding para mejorar la interpretación de los datos al utilizar un espacio de 32 dimensiones, a lo cual se le agregan una capa recurrente simple, una capa completamente conectada con dropout y una capa de salida. La arquitectura final es la siguiente:

Table 3: Arquitectura RNN

Capa	Configuración
Embedding	(vocab size = 127, embedding dim 32)
SimpleRNN	(units = 32)
Dense	(units = 32, activation = 'relu')
Dropout	(rate = 0.3)
Dense	(units = 11, activation = 'softmax')

El modelo final se entrena por 1,373 iteraciones a lo largo de 3 epochs aplicando mini-batch size de 128 dada la alta cantidad de datos, logrando un accuracy del 93.78% para los datos de validación durante el primer epoch de entrenamiento.

3.3 Resultados

Los resultados para los tres modelos evaluados son los siguientes:

Table 4: Resultados RNN

Model ID	Accuracy
1	90.31%
2	48.58%
3	93.53%

A partir de la siguiente tabla se selecciona el modelo número 3 como el mejor modelo entrenada para el esquema RNN al cumplir con un accuracy = 93.53%