Aplicaciones de Redes Neuronales MLP, CNN y RNN

Antonio Everardo Navas Contreras
Universidad Galileo
navas604@gmail.com

Abstract

El artículo presenta la aplicación de 3 redes neuronales más utilizadas en Deep Learning, de forma separada e independiente, la primera aplicación que se presenta es una red neuronal MLP (Multi-Layer Perceptron), para la cual se utiliza un dataset que contiene información personal de diferentes personas como la edad, nivel de educación, género, etc. La cual será utilizada para entrenar una red MLP de forma supervisada y predecir si su ingreso anual es mayor o menor a $50,000.00.

La segunda aplicación que se presenta es una red neuronal CNN (Convolutional Neural Network) para la cual se utiliza un dataset de fotos de manos expresando una letra del abecedario en lengua de señas, este dataset contiene imágenes con diferentes variaciones a fin de tener suficientes datos para el entrenamiento de la red convolucional, lo que se busca es poder predecir que letra del abecedario está expresando la foto de una mano.

La tercera aplicación que se presenta es una red neuronal RNN (Recurrent Neural Network) para la cual se utilizan datos extraídos de la página de yahoo sobre distintas variables que se analizan en las acciones de distintas empresas, para está aplicación se trabaja con 3 empresas realizado un análisis exploratorio de las variables y buscando predecir el precio de cierre de las acciones de cada una de las 3 empresas en cuestión.

Palabras Clave: Multi-Layer Perceptron, Convolutional Neural Network, Recurrent Neural Network, Aprendizaje supervisado, Data Science, Procesamiento de imágenes.

1 Problema

Se busca realizar 3 aplicaciones de 3 redes neuronales que son las mas comunes, en problemas cotidianos.

1.1 Red MLP

Se busca predecir si el ingreso anual de una persona es mayor o menor de $50,000.00, esta aplicación nace de la necesidad de tener una idea del ingreso de ciertas personas que solicitan un crédito o préstamo bancario, para tener una primera aproximación aunque después se solicite toda la documentación necesaria para respaldar la información brindad, con ello se puede avanzar a mayor velocidad al momento de tomar una decisión si se niega o no la solicitud de préstamo, puede ser utilizado en análisis de riesgos, y muchas otras aplicaciones.

1.2 Red CNN

Se busca identificar en una foto en la cual una persona este expresando una letra del abecedario en lengua de señas, qué letra es la que desea expresar, esta aplicación nace de la necesidad de tener herramientas inclusivas y que puedan servir para mejorar la comunicación con personas que tienen algún tipo de discapacidad auditiva, y también puede servir como base para un programa de

aprendizaje humano en el que pueda servir como método de evaluación para aquellas personas que estén interesadas en aprender el lenguaje de señas.

1.3 Red RNN

Se busca predecir el cierre futuro de las acciones de una empresa, esto nace por la necesidad de tener cierto grado de certidumbre al momento de realizar inversiones en una bolsa de valores, es importante que además de una asesoría profesional, se tengan herramientas en las cuales puedan servir para analizar el histórico de crecimiento de las empresas a lo largo de los años y con esto poder predecir el comportamiento futuro de dicha empresa, esto es una muy buena aproximación inicial para empezar a formar un criterio de selección de la cartera de inversiones.

2 Metodología

Se realizan 5 pasos básicos para el análisis y creación de las redes neuronales:

2.1 Análisis exploratorio de los datos

Se utiliza estadística para explorar los datos, validando que tipos de datos tienen cada una de las variables, cantidad de observaciones, la distribución que presentan, si necesitan algún tipo de escalado, datos N/A, o casillas en blanco, si existe alguna correlación para el caso de datos estructurados. Es importante considerar que este análisis exploratorio es con el fin de conocer los datos con los que se va a estar trabajando y determinar qué tipo de tratamiento es el mas adecuado basado en las conclusiones a las que se llegue en este paso y el tipo de problema que se está trabajando.

2.2 Tratamiento y preparación de los datos

Después de conocer nuestros datos, se aplica tratamiento a cada uno de los datasets elegidos.

2.2.1 Red MLP

Para este caso fue necesario rellenar algunos datos faltantes o desconocidos en ciertas variables que presentaban estas fallas, se aplica codificación para las variables categóricas de tipo “object”, esto con la finalidad de no utilizar One Hot Encoding para que la cantidad de columnas del dataset no tuviera un incremento considerable, luego ya se separaron los datos para tener un dataset de train y otro de test, dentro del entrenamiento de la red neuronal también se configura que realice un split del 20% del dataset de train para que sea utilizado como validación, y tener así los datos de test aislado y que jamás sean presentados al modelo sino hasta el final cuando ya se esté evaluando.

2.2.2 Red CNN

Se realiza un cambio en el tamaño de la imagen para estandarizarlos y que la red puede funcionar de una mejor forma, también se hace un escalado o normalización de los datos, este paso es el primero que se realiza dentro de la red neuronal, se hace un split inicial del 20% del dataset de train para que sea utilizado como validación, y el dataset de test se mantiene aislado.

2.2.3 Red RNN

Se realiza una descarga de los datos directamente del sitio de yahoo especializado en recopilar esta información de fuentes oficiales de las distintas bolsas de valores. Se utilizan datos de 3 compañías “MSFT”, “GOOG” y “AMZN”, se realiza normalización sobre los datos y se separan de igual forma dejando el 20% para test. También se realiza la conversión de la serie de tiempo a datos estructurados que se utilizarán para el entrenamiento del modelo.

2.3 Creación de la red neuronal

2.3.1 Red MLP

Se utiliza una red neuronal secuencial de 1 capa de entrada de 8 neuronas. 3 capas ocultas con 256, 128 y 32 neuronas respectivamente con función de activación “relu”, inicialización con distribución

uniforme. Y por último una capa de salida de 1 neurona con función de activación "sigmoid" para tener una clasificación binaria basado en probabilidad. Se utiliza Dropout como técnica de regularización para que desactive el 50% de las neuronas y así combatir el overfitting, para que las predicciones se realicen de mejor forma.

2.3.2 Red CNN

Se tiene una primera capa de aumentado del dataset de imágenes con ligeras variaciones de las imágenes para tener un mejor proceso de entrenamiento y no se tengo overfitting que luego nos pueda afectar al momento de realizar predicciones, después se realiza un escalado de los pixeles de cada imagen, se utilizan 3 capas de convolución espacial 2D para conocer las características, dos con 32 neuronas y 1 con 64 neuronas, cada una con tamaño de filtro de 3x3 y función de activación "relu", seguidas de MaxPool2D para reducir el tamaño de nuestra matriz de valores de cada pixel, se utiliza padding para dejar la imagen con las mismas dimensiones en cada capa oculta, también se utiliza "Dropout" como técnica de regularización para que desactive el 40% de las neuronas en las últimas 2 capas de convolución y así combatir el overfitting, para que las predicciones se realicen de mejor forma. Por último, se realiza un aplanado de la matriz para dejarla como un vector y pasar ese vector por una capa densa de 128 neuronas y por último una capa densa de 5 neuronas para realizar la clasificación de las imágenes.

2.3.3 Red RNN

Se utiliza el método LSTM (Long Short-Term Memory) que es muy utilizado en Deep Learning por la facilidad de entrenamiento para arquitecturas muy grandes, se utilizan 2 capas LSTM con 128 y 64 neuronas respectivamente, en la capa de entrada se configura que nos retorne la secuencia completa que luego se utilizará en la siguiente capa que también es LSTM pero en esta no nos interesa la secuencia completa sino solo el ultimo valor para ir reduciendo la dimensionalidad y luego pasar por una capa densa de 32 neuronas para finalmente llegar a la capa de salida de 1 neurona en la que nos interesa el valor como tal.

2.4 Entrenamiento de la red neuronal

2.4.1 Red MLP

Esta red por ser una clasificación binaria se utiliza "BinaryCrossentropy" como función de costo y optimizador "Nadam" que brindó mejor desempeño en el modelo, con Learning Rate de 0.01, y se valida con métricas Accuracy y AUC, se corre por 300 épocas el entrenamiento, y se utiliza un callback para realizar checkpoints en caso suceda cualquier cosa durante el proceso de entrenamiento.

2.4.2 Red CNN

Para esta red que es una clasificación multiclase y que se tienen varios pixeles en valores de 0 se utiliza una función de costo "SparseCategoricalCrossentropy" y optimizador "Adam" que brindó mejor desempeño en el modelo, con Learning Rate de 0.001, y se valida con métricas Accuracy, se corre por 200 épocas el entrenamiento, y se utiliza un callback para realizar checkpoints en caso suceda cualquier cosa durante el proceso de entrenamiento, ya que este entrenamiento si fue de aproximadamente 4 horas.

2.4.3 Red RNN

Para las series temporales se utiliza una función de costo "mean squared error" y optimizador "Adam" que brindó mejor desempeño en el modelo, con Learning Rate de 0.001, se corre por 50 épocas el entrenamiento, y se utiliza un callback para realizar checkpoints en caso suceda cualquier cosa durante el proceso de entrenamiento.

2.5 Evaluación de la red neuronal

Para la evaluación de las 3 redes neuronales se utiliza el dataset de validación para que se valide de una vez durante el entrenamiento y el dataset de test que fue separado desde el inicio de la

experimentación y nunca fue presentado al modelo, por lo que esos datos no los ha visto en ningún momento durante la experimentación sino hasta el final cuando el modelo ya está entrenado.

3 Resultados

3.1 Red MLP

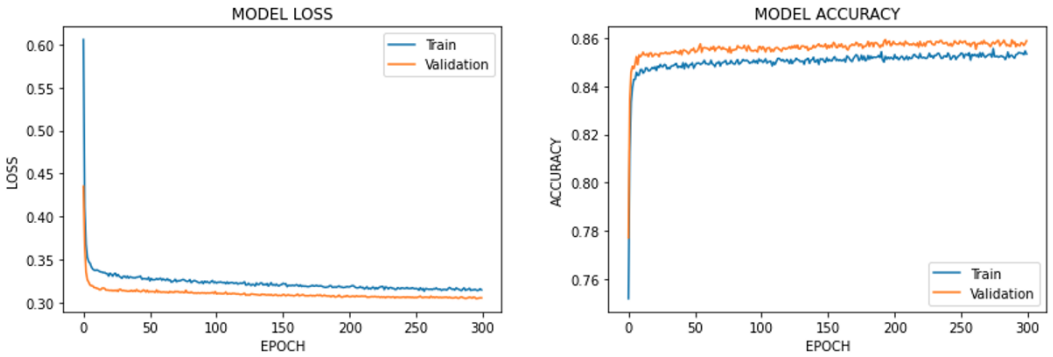

Figure 1: Loss y Accuracy MLP.

Se puede observar que durante el proceso de entrenamiento se logra la estabilidad o convergencia bastante rápido, en las primeras 100 épocas ya se tiene un costo bastante bajo y una exactitud bastante alta. El resultado final es una exactitud del 85.34% con los datos de entrenamiento, 85.89% con los datos de validación y 85.03% con los datos de Test.

3.2 Red CNN

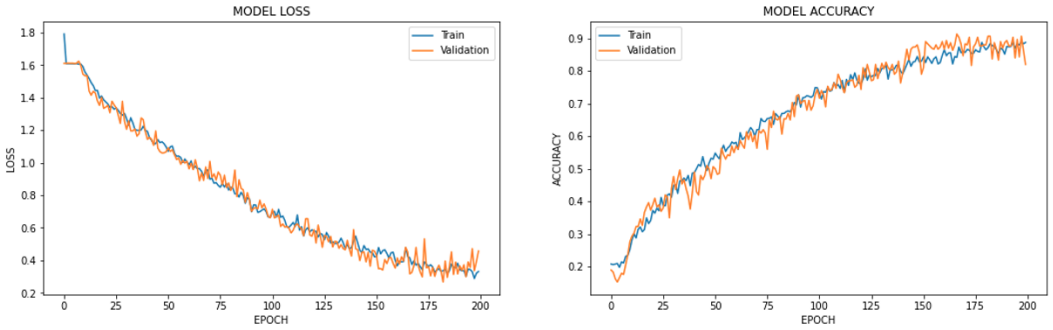

Figure 2: Loss y Accuracy CNN.

Podemos observar que la función de costo continúa disminuyendo y la gráfica de exactitud sigue aumentando, este experimento de realizó para 200 épocas derivado de limitaciones de cómputo ya que procesar las imágenes es bastante pesado por lo que conlleva mucho tiempo, para efectos de la prueba se utilizaron 200 épocas y un batch size de 32 con lo que se logró llegar a una exactitud de 88.75% con los datos de entrenamiento, 82% con los datos de validación y 88% con los datos de Test.

3.3 Red RNN

En esta red se puede observar que la función de costo también disminuye a medida que el modelo se va entrenando y en la segunda imagen se puede apreciar la predicción del precio de cierre de las acciones para las 3 compañías, estas predicciones se realizaron para un período de 1 año.

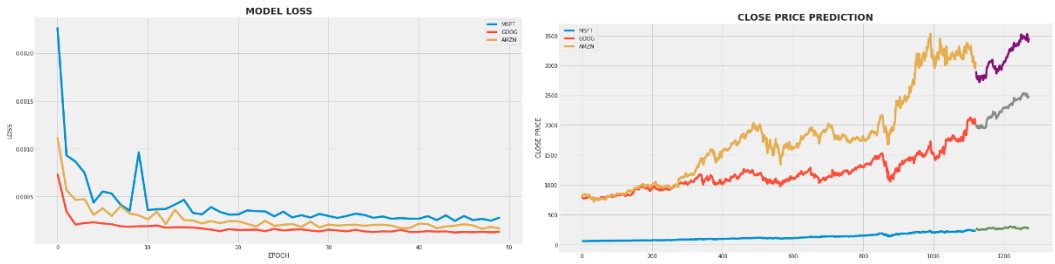

Figure 3: Loss y Accuracy RNN.

4 Conclusiones

4.1 Red MLP

- Se realizó un análisis exploratorio inicial de todas las variables independientes para tener mejor visibilidad de los datos y saber como trabajarlos, se consideran todas las variables del dataset, se creó una Red MLP para analizar las variables independientes y poder crear una red neuronal capaz de aprender el comportamiento de dichas variables respecto a la que queremos predecir, posterior a ello se realizan las predicciones, y se puede observar que se obtiene un Accuracy de 85.34% con los datos de entrenamiento, 85.89% con los datos de validación y 85.03% con los datos de Test.
- Es importante realizar tratamiento de las variables antes de meterlas a nuestra red neuronal para entrenarla, por ejemplo, tratar los datos N/A, o datos faltantes, realizar un escalado de la data, normalización, etc. para que la red neuronal tenga un mejor proceso de aprendizaje.

4.2 Red CNN

- Se pudo experimentar con distintos datasets, los cuales tenías distinta cantidad de imágenes, primero se experimentó con uno que contenía 11 imágenes por cada letra, a pesar de utilizar técnicas para realizar variaciones de dichas imágenes y aumentar el dataset, no fue posible conseguir Accuracy mayor al 30
- En el segundo dataset utilizado se tienen 300 imágenes de cada letra, se utilizan técnicas para variación de las imágenes y aumentar el dataset, inicialmente se utilizan pocas epochs y batch size de 64, para validar que corra bien, ya que es pesado entrenar los modelos con imágenes.
- Al tener el modelo ya optimizado se deja corriendo por 200 epochs y un batch size de 32, con lo cual se logra obtener un accuracy de 88.75% con los datos de entrenamiento, 82% con los datos de validación y 88% con los datos de Test.

4.3 Red RNN

- Se realizó un análisis exploratorio de distintas variables para las acciones de 3 compañías que cotizan en bolsa, y se creó una red neuronal recurrente para analizar los datos históricos y poder predecir el precio de cierre de las acciones, por un período de 1 año a partir de la fecha actual en que se realice el análisis.
- Mientras mas tiempo hacia atras se tenga de data histórica, podrá realizarse una mejor tendencia y predicción.
- Se puede observar que la red RNN, se puede modelar como aprendizaje supervisado transformando los datos a una forma estructurada.

References

[1] Hüsken, M., & Stagge, P. (2003). Recurrent neural networks for time series classification. Neurocomputing, 50, 223-235.

[2] Ienco, D., Gaetano, R., Interdonato, R., Ose, K., & Minh, D. H. T. (2019, July). Combining Sentinel-1 and Sentinel-2 Time Series via RNN for object-based land cover classification. In IGARSS 2019-2019 IEEE International Geoscience and Remote Sensing Symposium (pp. 4881-4884). IEEE.

[3] Ni, L., Li, Y., Wang, X., Zhang, J., Yu, J., & Qi, C. (2019). Forecasting of forex time series data based on deep learning. Procedia computer science, 147, 647-652.

[4] Huang, J., Zhou, W., Li, H., & Li, W. (2015, June). Sign language recognition using 3d convolutional neural networks. In 2015 IEEE international conference on multimedia and expo (ICME) (pp. 1-6). IEEE.

[5] Cui, R., Liu, H., & Zhang, C. (2019). A deep neural framework for continuous sign language recognition by iterative training. IEEE Transactions on Multimedia, 21(7), 1880-1891.