Universidad del Valle de Guatemala

Facultad de Ingeniería

Deep Learning y Sistemas Inteligentes

Informe

Laboratorio 4

Mariana David 201055

Pablo Escobar 20936

Eduardo Ramirez 19946

Guatemala 10 de septiembre del 2023

1. Resultados obtenidos

```
Epoch 1/200
165/165 - 2s - loss: 0.5983 - accuracy: 0.7272 - val_loss: 0.5236 - val_accuracy: 0.7858 - 2s/epoch - 15ms/step
Epoch 2/200
165/165 - 1s - loss: 0.5454 - accuracy: 0.7809 - val_loss: 0.5043 - val_accuracy: 0.8009 - 980ms/epoch - 6ms/step
Epoch 3/200
165/165 - 1s - loss: 0.5231 - accuracy: 0.7886 - val_loss: 0.4883 - val_accuracy: 0.8022 - 1s/epoch - 7ms/step
Epoch 4/200
165/165 - 1s - loss: 0.5102 - accuracy: 0.7977 - val_loss: 0.4761 - val_accuracy: 0.8064 - 995ms/epoch - 6ms/step
Epoch 5/200
165/165 - 1s - loss: 0.5010 - accuracy: 0.7984 - val_loss: 0.4690 - val_accuracy: 0.8060 - 950ms/epoch - 6ms/step
Epoch 6/200
165/165 - 1s - loss: 0.4918 - accuracy: 0.8012 - val_loss: 0.4681 - val_accuracy: 0.8071 - 972ms/epoch - 6ms/step
Epoch 7/200
165/165 - 1s - loss: 0.4854 - accuracy: 0.8034 - val_loss: 0.4607 - val_accuracy: 0.8078 - 998ms/epoch - 6ms/step
Epoch 8/200
165/165 - 1s - loss: 0.4824 - accuracy: 0.8059 - val_loss: 0.4583 - val_accuracy: 0.8104 - 951ms/epoch - 6ms/step
Epoch 9/200
165/165 - 1s - loss: 0.4786 - accuracy: 0.8054 - val_loss: 0.4563 - val_accuracy: 0.8089 - 1s/epoch - 6ms/step
Epoch 10/200
165/165 - 1s - loss: 0.4768 - accuracy: 0.8089 - val_loss: 0.4575 - val_accuracy: 0.8104 - 979ms/epoch - 6ms/step
Epoch 11/200
165/165 - 1s - loss: 0.4728 - accuracy: 0.8083 - val_loss: 0.4556 - val_accuracy: 0.8091 - 961ms/epoch - 6ms/step
Epoch 12/200
165/165 - 1s - loss: 0.4688 - accuracy: 0.8076 - val_loss: 0.4521 - val_accuracy: 0.8138 - 1s/epoch - 8ms/step
Epoch 13/200
...
Epoch 200/200
165/165 - 1s - loss: 0.4150 - accuracy: 0.8238 - val_loss: 0.4462 - val_accuracy: 0.8133 - 1s/epoch - 6ms/step
141/141 [==============================] - 0s 2ms/step - loss: 0.4300 - accuracy: 0.8227
Accuracy on test data: 82.27%
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

Figura 1. Resultados

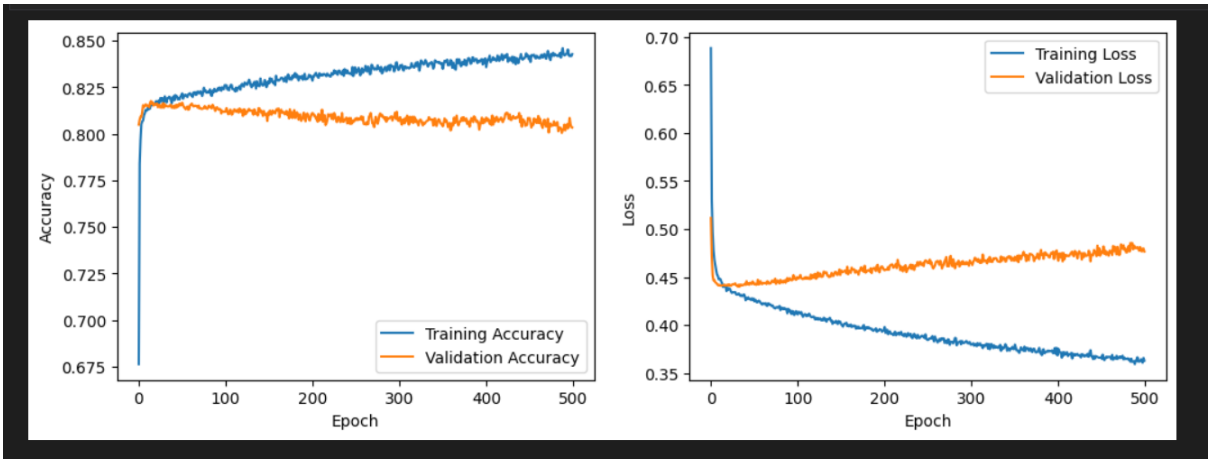

Figura 2. Gráfica de resultados

2. Descripción de la red neuronal y sus hiperparámetros

La red neuronal utilizada en este código es un modelo de clasificación binaria que consta de las siguientes capas y hiperparámetros:

Capa de Entrada

- Dense Layer

Esta capa tiene 512 unidades (neuronas) con una función de activación ReLU. Utiliza la inicialización "he_normal", que es una técnica de inicialización que puede ayudar a entrenar redes profundas de manera más efectiva. La capa tiene la misma forma que los datos de entrada.

Capas Ocultas

- BatchNormalization Layer

Esta capa se utiliza después de cada capa densa para normalizar las activaciones y mejorar la estabilidad del entrenamiento. Ayuda a mitigar los problemas de covariable interna en redes profundas.

- Dropout Layer

Se aplica una capa de Dropout después de cada capa densa con una tasa de dropout del 50%. El Dropout es una técnica de regularización que ayuda a prevenir el sobreajuste al apagar aleatoriamente algunas neuronas durante el entrenamiento.

Capa de Salida

- Dense Layer

Esta capa tiene una sola unidad con una función de activación sigmoide, que es típica para problemas de clasificación binaria.

Compilación del Modelo

- Optimizador

Se utiliza el optimizador Adam con una tasa de aprendizaje de 0.001. Adam es un optimizador eficiente y popular para entrenar redes neuronales.

- Función de Pérdida

La función de pérdida utilizada es “binary_crossentropy”, que es adecuada para problemas de clasificación binaria.

- Métricas

Se utiliza la métrica de precisión ('accuracy') para evaluar el rendimiento del modelo.

3. Proceso para llegar a esta red

- El proceso para llegar a esta red neuronal implicó los siguientes pasos:
- Carga de datos: Se cargaron los datos desde un archivo Excel y se separaron en características (X) y etiquetas (Y).
- División de datos: Los datos se dividieron en conjuntos de entrenamiento, validación y prueba utilizando “train_test_split”.
- Normalización: Se normalizaron los datos utilizando “StandardScaler” para asegurar que todas las características tengan una escala similar.
- Definición de la arquitectura: Se definió la arquitectura de la red neuronal con múltiples capas densas intercaladas con Batch Normalization y Dropout. Se utilizó una inicialización adecuada para los pesos.
- Compilación del modelo: Se configuró el modelo con el optimizador Adam, la función de pérdida “binary_crossentropy” y la métrica de precisión.
- Entrenamiento del modelo: El modelo se entrenó con los datos de entrenamiento y se utilizó el conjunto de validación para supervisar el proceso de entrenamiento.
- Evaluación del modelo: Finalmente, el modelo se evaluó en el conjunto de prueba para medir su rendimiento.

4. Ideas para mejorar la red

- Ajustar hiperparámetros: Experimentar con diferentes valores de hiperparámetros como la tasa de aprendizaje, el tamaño del lote, el número de unidades en las capas ocultas y la tasa de dropout para encontrar la combinación óptima.
- Regularización adicional: Además del dropout, se podría agregar regularización L1 o L2 a las capas densas para reducir aún más el sobreajuste.
- Aumento de datos: aplicar técnicas de aumento de datos para generar más ejemplos de entrenamiento artificiales y mejorar la capacidad de generalización.
- Experimentación con arquitectura: arquitecturas de redes neuronales diferentes, como redes convolucionales (CNN) o redes recurrentes (RNN), si son relevantes para su problema.
- Sintonización de hiperparámetros avanzada: usar técnicas de sintonización de hiperparámetros como búsqueda de hiperparámetros aleatoria o búsqueda en cuadrícula para encontrar la mejor combinación de hiperparámetros.
- Recopilación de más datos: recopilar más datos para mejorar la capacidad de entrenamiento del modelo y reducir el riesgo de sobreajuste.