Universidad Sergio Arboleda

Facultad de Ciencias Exactas e Ingeniería

Taller 3 - Deep Learning

Computación 2

Autor:

Mariana Rodríguez Pérez

Mayo 2025

1. **¿Por qué existe una diferencia entre 'accuracy' o 'loss' en las fases de entrenamiento y pruebas?**

   La diferencia entre 'accuracy' y 'loss' en las fases de entrenamiento y prueba se debe a que el modelo analiza distintos tipos de datos en cada fase. Durante el entrenamiento, el modelo ajusta sus parámetros con base en los datos que conoce, y por eso suele tener un mejor rendimiento (mayor accuracy y menor loss) en ese conjunto. Sin embargo, cuando se evalúa en la fase de prueba, se enfrenta a datos nuevos que no ha visto antes, por lo que puede equivocarse más y mostrar una precisión un poco más baja o una pérdida más alta.

   Además, el 'loss' (pérdida) y el 'accuracy' (precisión) no siempre se comportan igual porque miden cosas diferentes; el 'loss' se basa en cuán lejos está la predicción del valor real (como un margen de error), mientras que el 'accuracy' solo mide si acertó o no. Por eso, puedes tener un modelo con una precisión aceptable, pero que todavía tenga un valor de pérdida relativamente alto si las predicciones acertadas fueron por poco.

2. **Por qué considera que algunas de las imágenes no lograron ser categorizadas correctamente? ¿Qué las podría diferenciar?**

   Una de las principales razones por las que algunas imágenes no fueron clasificadas correctamente ocurre ya que ciertas imágenes pueden ser visualmente muy parecidas entre clases. Por ejemplo, una camiseta de manga larga puede parecerse bastante a una blusa, o unos zapatos deportivos a unas sandalias, especialmente si están en blanco y negro y con baja resolución como lo es el dataset usado.

   En esta situación también influye la calidad de la imagen: algunas pueden estar borrosas, mal centradas, rotadas o tener detalles poco visibles, lo que dificulta que el modelo capte los patrones correctos. Además, si durante el entrenamiento el modelo no vio suficientes ejemplos similares a esas imágenes difíciles, es natural que no haya aprendido a clasificarlas bien.

   Otro factor importante es que las redes neuronales no "entienden" el contexto como lo haría una persona. Ellas se basan en patrones numéricos, así que si dos clases comparten características similares en píxeles, es fácil que se confundan.

3. **¿Qué ocurre cuando incrementamos la cantidad de neuronas intermedias de 128 a 500? ¿Qué le ocurre a la diferencia de los valores de train vs test? Explique su respuesta.**

   Cuando aumentamos la cantidad de neuronas en la capa oculta de 128 a 500, el modelo se vuelve más complejo y con mayor capacidad para aprender patrones. Esto puede ayudar a que se ajuste mejor a los datos de entrenamiento, logrando una precisión más alta en esa fase. Sin embargo, esa misma complejidad puede hacer que el modelo empiece a memorizar demasiado los datos de entrenamiento, en lugar de aprender patrones generales.

   Como resultado, la diferencia entre el rendimiento en entrenamiento (train) y prueba (test) puede aumentar. Es decir, el modelo puede tener una muy buena precisión en los datos que ya vio, pero un rendimiento más bajo en los nuevos, lo que se conoce como sobreajuste (overfitting).

4. **¿Qué ocurre cuando incrementamos la cantidad de épocas (iteraciones) de 10 a 20? ¿Qué le ocurre a la diferencia de los valores de train vs test? Explique su respuesta.**

   Al aumentar el número de épocas de 10 a 20, el modelo tiene más tiempo para aprender y ajustar sus pesos con base en los datos de entrenamiento. Esto puede hacer que la precisión en el conjunto de

entrenamiento siga mejorando, ya que el modelo se adapta cada vez más a esos datos específicos.
Sin embargo, a partir de cierto punto, entrenar más no siempre significa que el modelo va a generalizar mejor. De hecho, puede empezar a memorizar los datos en lugar de aprender patrones útiles, lo que lleva nuevamente al problema del sobreajuste. Esto se nota cuando la precisión en entrenamiento sigue subiendo, pero la precisión en prueba se estanca o incluso empieza a bajar.

5. **Explique como es la arquitectura del siguiente modelo:**
   Este modelo tiene una arquitectura pensada para tareas de clasificación de imágenes como las del dataset *Fashion MNIST*. Se construye con la API `Sequential` de Keras, lo que significa que las capas están organizadas de forma secuencial, una después de la otra.
   La primera capa es `Flatten`, que transforma la imagen de 28x28 píxeles (2D) en un vector de 784 elementos (1D). Esta transformación es necesaria porque las capas densas (completamente conectadas) requieren vectores de entrada. La siguiente capa es una capa `Dense` con 128 neuronas y función de activación `ReLU`. Esta es una capa oculta que se encarga de procesar la información y detectar patrones importantes. Al tener 128 neuronas, tiene una capacidad razonable para aprender sin volverse demasiado compleja. Por último, la capa de salida también es `Dense`, con 10 neuronas (una por cada clase posible en *Fashion MNIST*). Esta capa no tiene activación porque el modelo se entrena con la función de pérdida `SparseCategoricalCrossentropy(from_logits=True)`, lo cual ya considera que las salidas son valores çrudos"(logits). En resumen, es una red neuronal *feedforward*, compuesta por una capa de entrada (implícita), una capa oculta de 128 neuronas con `ReLU`, y una capa de salida con 10 neuronas que representa las clases posibles.