Universidad del Valle de Guatemala
Facultad de ingeniería

Data Science
Laboratorio 2
Clasificacion de rotulos de tráfico utilizando CNN Le-Net

Andrés de la Roca
Jun Woo Lee

Guatemala, 2023

Índice

Exploración de datos

Para la exploración de datos de este laboratorio se realizó una división de datos a partir del conjunto de entrenamiento que contiene un total de 34799 imágenes a conjuntos de validación y de prueba, a partir de estos conjuntos se pudieron visualizar la información acerca de la dimensión de cada una de las imágenes, cada imagen se compone de 3 arreglos de 32x32 arreglos, cada uno de estos arreglos conteniendo la información del color de cada una de las imágenes para así formar una imagen de 32x32 pixeles.

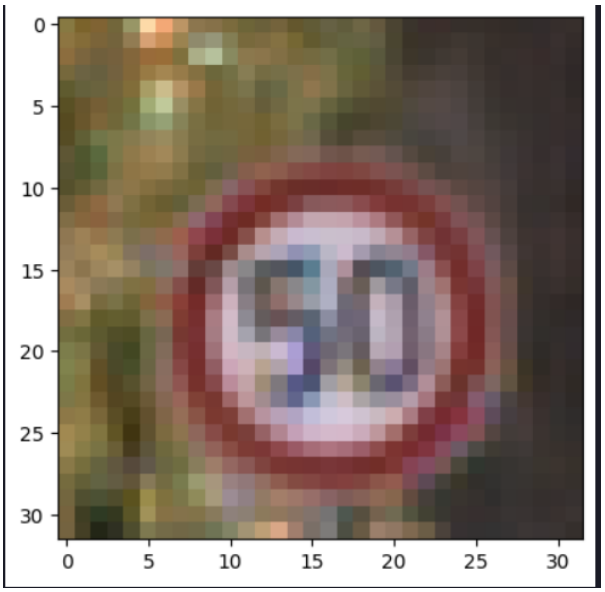

Fig. 1. Visualización de imagen del conjunto de entrenamiento

Vale la pena mencionar la normalización realizada a cada una de las imágenes para de esta forma facilitar más adelante el entrenamiento, así mejorando el rendimiento del modelo.

Adicionalmente, es de suma importancia recordar cada una de las clases de imágenes que conforman los conjuntos que utilizará el modelo. Cada clase tiene una forma, tonos o figuras únicas lo que hace que se diferencie uno del otro, lo cual facilitará al modelo poder diferenciar entre señales/imágenes.

0 - Limite velocidad (20km/h)
1 - Limite velocidad (30km/h)
2 - Limite velocidad (50km/h)
3 - Limite velocidad (60km/h)
4 - Limite velocidad (70km/h)
5 - Limite velocidad (80km/h)
6 - Fin de limite velocidad (80km/h)
7 - Limite velocidad (100km/h)
8 - Limite velocidad (120km/h)
9 - No rebasar
10 - No rebasar para vehículos mayores de 3.5 toneladas métricas

11 - Derecho-de-vía en la siguiente intersección
12 - Camino prioritario
13 - Ceda el paso
14 - Alto
15 - No vehículos
16 - Prohibido vehículos mayores de 3.5 toneladas métricas
17 - No hay entrada
18 - Precaución general
19 - Curva peligrosa a la izquierda
20 - Curva peligrosa a la derecha
21 - Doble curva
22 - Camino disparejo
23 - Camino resbaloso
24 - Camino se reduce a la derecha
25 - Trabajos adelante
26 - Señales de Tráfico -semáforos-
27 - Cruce de peatones
28 - Cruce de Niños
29 - Cruce de bicicletas
30 - Cuidado hielo/nieve
31 - Cruce de animales silvestres
32 - Fin de todos los límites de velocidad y rebase
33 - Gire a la derecha adelante
34 - Gire a la izquierda adelante
35 - Recto solo
36 - Vaya recto o a la derecha
37 - Vaya recto o a la izquierda
38 - Manténgase a la derecha
39 - Manténgase a la izquierda
40 - Vuelta en U obligada
41 - Fin de no rebasar
42 - Fin de no rebasar para vehículos mayores de 3.5 toneladas métricas

Implementación de Le-Net

Para entender la arquitectura utilizada hay que entender su historia y sus conceptos; Le-Net fue desarrollada en 1998 por Yann Lecun, es una de las primeras arquitecturas CNN y fue diseñada principalmente para la tarea de reconocimiento de dígitos escritos a mano en imagenes, como parte del proyecto de procesamiento de cheques del Banco Nacional de Francia.

Esta arquitectura consiste de siete capas, incluyendo tres capas de convolución, dos capas de submuestreo (pooling) y dos capas totalmente conectadas (feedforward layers). A continuación se detalla un poco más el funcionamiento de cada capa.

- Capa de entrada: La entrada son imágenes en escala de grises de tamaño fijo (generalmente 32x32 píxeles), que representan las señales de tránsito.
- Capa de convolución (C1): Esta es la primera capa convolucional. Utiliza kernels para realizar convoluciones en la imagen de entrada. Los filtros detectan características simples, como bordes y esquinas. Se aplican funciones ReLU de activación después de cada convolución.
- Capa de submuestreo (S2): Se aplica un proceso de pooling para reducir la dimensionalidad y la cantidad de parámetros de la red. Esto se hace utilizando operaciones de agrupación en regiones locales de la imagen.
- Capa de convolución (C3): En esta capa nuevamente se realizan convoluciones, sin embargo, en esta ocasión se tienen conexiones con regiones más grandes de la imagen.
- Capa de submuestreo (S4): Al igual que la capa S2, esta capa de submuestreo reduce la dimensionalidad de la representación y enfatiza las características más prominentes de la imagen.
- Capa totalmente conectada (FC5): Después de las capas convolucionales y de submuestreo, la red se aplana y se conecta a una capa totalmente conectada. En esta capa, se realizan operaciones de multiplicación de matrices para aprender características más abstractas y complejas.
- Capa totalmente conectada (FC6): La última capa totalmente conectada toma las características aprendidas y las mapea a las clases de salida. En este caso siendo 42 salidas por cada tipo de señal de tránsito.

En esta ocasión la función de pérdida utilizada es la entropía cruzada categórica, ideal para este problema de clasificación.

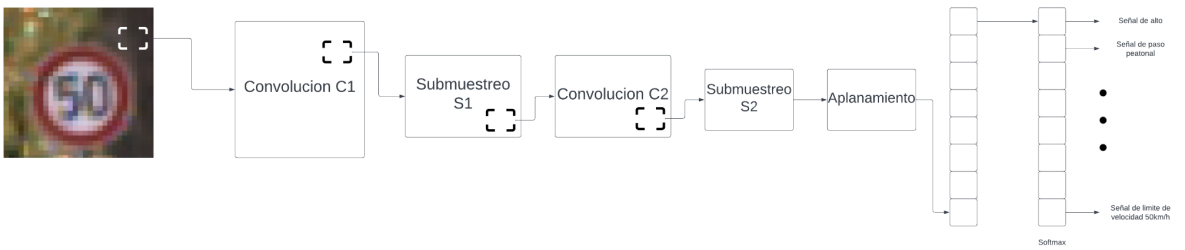

Fig. 2. Diagrama de CNN Le-Net

Construcción del modelo

Siguiendo la arquitectura planteada de Le-Net, se utilizaron las siguientes capas para crear el modelo:

1. Capa de convolución (C1):

Esta es la primera capa de convolución. Tiene 6 filtros de tamaño 5x5, utiliza la función de activación ReLU y espera una entrada de imágenes con un tamaño de 32x32.

2. Capa de submuestreo (S2):
   Se agrega una capa de submuestreo con ventanas de 2x2 para reducir la dimensionalidad de las características

3. Capa de convolución (C3):
   Tiene 16 filtros de tamaño 5x5 y también utiliza la función ReLU.

4. Capa de submuestreo (S4):
   Con las mismas características que la S2, reduce aún más la dimensionalidad.

5. Capa de aplanamiento:
   Se aplana la salida de la última capa convolucional para prepararla para la alimentación a las capas totalmente conectadas

6. Capa totalmente conectada (FC5):
   Tiene 120 neuronas y utiliza ReLU.

7. Capa totalmente conectada (FC6):
   Otra capa totalmente conectada con 84 neuronas y ReLU.

8. Capa de salida:
   La última capa, con 43 neuronas correspondientes a las clases de las señales de tráfico, utiliza la función softmax para obtener probabilidades de clasificación.

Entrenamiento del modelo

El proceso de entrenamiento de nuestro modelo de detección de señales de tráfico sigue varios pasos esenciales. Iniciamos con el preprocesamiento de nuestros datos, donde normalizamos las imágenes del conjunto para que sus valores oscilen entre 0 y 1, lo cual se logra dividiendo cada pixel de la imagen por 255. Además, separamos el conjunto de datos en dos partes: una para entrenar el modelo y otra para validar su rendimiento. Este paso es crucial para garantizar que el modelo no solo memorice los datos sino que también pueda generalizar bien a datos no vistos.

Una vez preparados los datos, definimos ciertos hiper parámetros que guiarán el proceso de entrenamiento. Establecemos una tasa de aprendizaje, definimos un número de épocas y seleccionamos un tamaño de lote específico. Estos hiper parámetros son esenciales para garantizar un entrenamiento eficiente y efectivo.

A continuación, preparamos el modelo para el entrenamiento. Usamos el optimizador Adam, ampliamente reconocido por su eficiencia, con la tasa de aprendizaje que definimos anteriormente. El modelo se compila con una función de pérdida específica para tareas de clasificación multiclase y se configura para medir la precisión durante el entrenamiento.

Para mejorar la robustez de nuestro modelo, introducimos una técnica llamada "Detección Temprana" o "Early Stopping" en inglés. Esta técnica monitorea el rendimiento del modelo en el conjunto de validación y detiene el entrenamiento si no observa mejoras después de un cierto número de épocas, en nuestro caso, dos épocas. Esta estrategia ayuda a prevenir el sobreajuste y ahorra tiempo y recursos.

Una vez entrenado, guardamos el modelo en un archivo para su uso futuro. Este modelo ahora está listo y puede ser implementado en aplicaciones prácticas de detección de señales de tráfico.

Finalmente, para visualizar y entender el comportamiento del modelo durante el entrenamiento, generamos gráficos de su precisión y pérdida en función de las épocas. Estas visualizaciones son fundamentales para evaluar cómo evoluciona el modelo y para identificar posibles áreas de mejora.

Resultados

Los resultados del entrenamiento del modelo de detección de señales de tráfico muestran un rendimiento prometedor con una precisión general del 97.74%. Al analizar las métricas detalladas, observamos que la mayoría de las clases tienen una alta precisión, recall y F1-score, acercándose o incluso alcanzando el 100% en varias categorías. Por ejemplo, la clase 0 tiene una precisión de 100% y un recall del 97%, dando como resultado un F1-score del 99%. Esta tendencia de alto rendimiento es consistente en la mayoría de las clases, como se refleja en la precisión ponderada y el F1-score ponderado, ambos con un 98%. Estos resultados indican que el modelo tiene una capacidad robusta para identificar correctamente las señales de tráfico en el conjunto de validación.

```
218/218 [==============================] - 1s 2ms/step
Accuracy: 0.9774

              precision    recall  f1-score   support

           0       1.00      0.97      0.99        38
           1       0.99      0.98      0.99       425
           2       0.98      0.98      0.98       394
           3       1.00      0.84      0.91       242
           4       0.99      1.00      0.99       373
           5       0.91      0.96      0.93       338
           6       0.97      1.00      0.98        56
           7       0.97      0.99      0.98       242
           8       0.99      0.98      0.99       237
           9       0.97      1.00      0.98       253
          10       0.98      1.00      0.99       332
          11       0.97      0.97      0.97       229
          12       1.00      1.00      1.00       417
          13       0.99      0.99      0.99       386
          14       1.00      0.99      1.00       130
          15       0.98      0.95      0.97       125
          16       1.00      0.94      0.97        77
          17       1.00      1.00      1.00       203
          18       0.98      0.96      0.97       219
          19       0.93      1.00      0.96        26
...
    accuracy                           0.98      6960
   macro avg       0.97      0.97      0.97      6960
weighted avg       0.98      0.98      0.98      6960
```

Fig. 3. Resultados

Reflexiones

Una estrategia común para mejorar el rendimiento de los modelos de aprendizaje profundo, especialmente en tareas de visión, es la ampliación de datos. Esta técnica incrementa artificialmente el tamaño del conjunto de entrenamiento aplicando varias transformaciones a las imágenes originales, tales como rotación, acercamiento y volteo horizontal. Esto ayuda al modelo a generalizar mejor y reduce el riesgo de sobreajuste.

En situaciones del mundo real, especialmente en el ámbito de reconocimiento de señales de tráfico, el sistema puede encontrarse con señales bajo diversas condiciones, ya sea debido a efectos climáticos, diferentes condiciones de iluminación o incluso obstrucciones físicas. Al usar la ampliación de datos, simulamos estas variaciones durante el entrenamiento. Esto garantiza que nuestro modelo esté preparado para reconocer señales incluso bajo condiciones no ideales.

Por ejemplo, una imagen rotada o ampliada del conjunto de entrenamiento puede simular un escenario donde la cámara que capta la señal de tráfico está en un ángulo o a una distancia variable del letrero. El volteo horizontal, aunque no es completamente realista para las señales de tráfico, puede ser considerado como un método de regularización potente, fortaleciendo la robustez de la red neuronal.

Referencias

Dive Into Deep Learning (2021) Convolutional Neural Networks (LeNet). Extraido de d2l.ai

Alake, R. (2020) Understanding and Implementing LeNet-5 CNN Architecture (Deep Learning). Extraído de Medium

Das, S. (2017) CNN Architectures: LeNet, AlexNet, VGG, GoogLeNet, ResNet and more… Extraido de Medium