Reporte 4: LeNet reconocimiento de señales de tránsito

Efren Lopez, CIDETEC-IPN

I. Introducción

Redes neuronales convolucionadas son similares a una red neuronal ordinaria, cada neurona recibe alguna entrada. Las redes neuronales convolucionales aprovechan el hecho de que la entrada consiste en imágenes y restringen la arquitectura de una manera más sensata. En particular, a diferencia de una red neuronal normal, las capas de un ConvNet tienen neuronas dispuestas en 3 dimensiones: ancho, alto, profundidad.

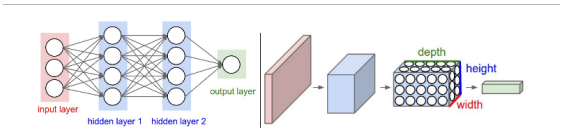

Fig. 1. De izquierda, una red neuronal simple y a la derecha una red neuronal convolucionada

A. Arquitectura de una red convolucionada

- INPUT [32x32x3] mantendrá los valores de píxel en bruto de la imagen, en este caso una imagen de ancho 32, alto 32 y con tres canales de color R, G, B.
- La capa CONV calculará la salida de las neuronas que están conectadas a regiones locales en la entrada, cada una calculando un producto de punto entre sus pesos y una región pequeña a la que están conectados en el volumen de entrada. Esto puede generar un volumen como [32x32x12] si decidimos usar 12 filtros.
- La capa CONV calculará la salida de las neuronas que están conectadas a regiones locales en la entrada, cada una calculando un producto de punto entre sus pesos y una región pequeña a la que están conectados en el volumen de entrada. Esto puede generar un volumen como [32x32x12] si decidimos usar 12 filtros.
- La capa RELU aplicará una función de activación de elemento, como el $max(0, x)$umbral en cero. Esto deja el tamaño del volumen sin cambios ([32x32x12]).
- La capa POOL realizará una operación de reducción de muestreo a lo largo de las dimensiones espaciales (ancho, alto), lo que generará un volumen como [16x16x12].
- La capa FC (es decir, totalmente conectada) calculará los puntajes de clase, lo que resulta en un volumen de tamaño [1x1x10], donde cada uno de los 10 números corresponde a un puntaje de clase, como entre las 10 categorías de CIFAR-10. Como ocurre con las redes neuronales ordinarias, y como su nombre lo indica, cada neurona de esta capa estará conectada a todos los números del volumen anterior.

jmzelectronica@gmail.com

II. Desarrollo

Para esta tarea, se retomó algunos ejemplos realizados previamente, en donde se describen algunas caracteristicas.

```
Number of training examples = 34799
Number of validation examples = 4410
Number of testing examples = 12630
Image data shape = (32, 32, 3)
Number of classes = 43
```

Fig. 2. Características de la red neuronal

En donde tenemos 34799 ejemplos de entrenamiento, 4410 ejemplos de validación, 12630 ejemplos de validación, las imágenes son de 32x32x3 y 43 números de clases.
En la siguiente imagen se muestra las diferentes clasificaciones de las señales de transito.

```
Most common signs:
'Speed limit (50km/h)'  train samples: 2010
'Speed limit (30km/h)'  train samples: 1980
'Yield'  train samples: 1920
'Priority road'  train samples: 1890
'Keep right'  train samples: 1860
'No passing for vehicles over 3.5 metric tons'  train samples: 1800
'Speed limit (70km/h)'  train samples: 1770
'Speed limit (80km/h)'  train samples: 1650
'Road work'  train samples: 1350
'No passing'  train samples: 1320

Most rare signs:
'Go straight or left'  train samples: 180
'Dangerous curve to the left'  train samples: 180
'Speed limit (20km/h)'  train samples: 180
'End of no passing'  train samples: 210
'End of no passing by vehicles over 3.5 metric tons'  train samples: 210
'End of all speed and passing limits'  train samples: 210
'Pedestrians'  train samples: 210
'Bicycles crossing'  train samples: 240
'Road narrows on the right'  train samples: 240
'Keep left'  train samples: 270
```

Fig. 3. Clasificación de las imágenes

En la siguiente parte se muestra algunos de los ejemplos de entrenamiento.

A. Modelo de arquitectura

En esta parte del programa fue donde se propuso un modelo en donde se describe lo siguiente:

- Tiene una capa convolucional y como entrada tiene una imagen de 32x32x3 y de salida es 28x28x6
- Se tiene una etapa de pooling en donde se tiene una entrada de 28x28x6 y una salida de 14x14x3

Fig. 4. Ejemplos de entrenamiento

- Posteriormente se realiza una segunda capa convolucional en donde tiene una salida de 10x10x16
- Posteriormente tiene una capa de activación RELU y toma los parámetros de la convolucion 2.
- Posteriormente necesitamos una capa de flatten esto nos ayuda para convertir la parte convolucional de la CNN en un vector de características de 1 dimensión.
- Posteriormente se realiza una etapa de activación RELU.
- Y finalmente se realiza una capa convolucional totalmente conectada con una entrada de 84 y una salida de 10

En la siguiente imagen muestra el proceso de entrenamiento y la reducción máxima de error, teniendo 0.197, en donde se ocupan 10 épocas

```
2017-10-31 10:48:19.944630 Training... dropout = 0.3 , batch_size = 128 , learning rate = 0.0001

2017-10-31 10:50:46.775255 EPOCH 1 - 147 sec ...
2017-10-31 10:50:46.775696 Training error = 0.752 Validation error = 0.802

2017-10-31 10:53:18.450302 EPOCH 2 - 299 sec ...
2017-10-31 10:53:18.458327 Training error = 0.546 Validation error = 0.620

2017-10-31 10:55:53.240634 EPOCH 3 - 454 sec ...
2017-10-31 10:55:53.241362 Training error = 0.414 Validation error = 0.490

2017-10-31 10:58:39.169958 EPOCH 4 - 620 sec ...
2017-10-31 10:58:39.170251 Training error = 0.328 Validation error = 0.407

2017-10-31 11:01:04.311927 EPOCH 5 - 765 sec ...
2017-10-31 11:01:04.312472 Training error = 0.272 Validation error = 0.337

2017-10-31 11:03:37.297539 EPOCH 6 - 918 sec ...
2017-10-31 11:03:37.298028 Training error = 0.232 Validation error = 0.289

2017-10-31 11:06:14.542173 EPOCH 7 - 1075 sec ...
2017-10-31 11:06:14.542662 Training error = 0.204 Validation error = 0.258

Model saved 1075 sec
2017-10-31 11:08:42.797727 EPOCH 8 - 1223 sec ...
2017-10-31 11:08:42.798196 Training error = 0.178 Validation error = 0.234

2017-10-31 11:11:11.306690 EPOCH 9 - 1372 sec ...
2017-10-31 11:11:11.307231 Training error = 0.165 Validation error = 0.227

2017-10-31 11:13:42.444021 EPOCH 10 - 1523 sec ...
2017-10-31 11:13:42.444971 Training error = 0.143 Validation error = 0.197

Model saved 1523 sec
```

Fig. 5. Características de la red neuronal

A continuación se muestra la curva obtenida, comparando los parámetros reales y los obtenidos.

En la siguiente imagen se muestra la eficiencia obtenida durante el entrenamiento e la red Para validar los resultados y asegurarnos de que la red entrenó de forma adecuada, se tiene la siguiente imagen.

III. Conclusión

En este ejercicio me fue complicado determinar en que momento ocupar una capa convolucional o una activación

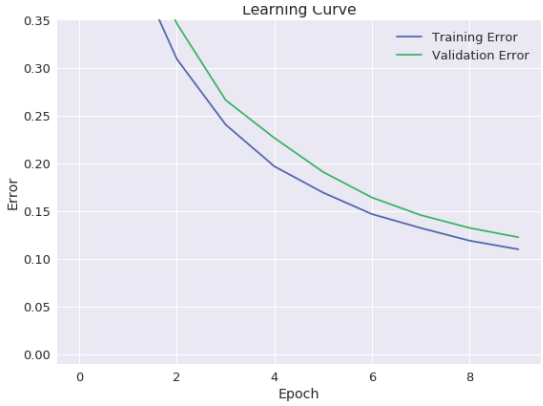

Fig. 6. Características de la red neuronal

```
INFO:tensorflow:Restoring parameters from ./models/lenet
Accuracy Model On Training Images: 0.89
Accuracy Model On Validation Images: 0.88
Accuracy Model On Test Images: 0.86
```

Fig. 7. Características de la red neuronal

```
./images_signs/double_curve.png
./images_signs/go_straigth_or_left.jpg
./images_signs/children_crossing.jpg
./images_signs/80-Km-limit.jpg
./images_signs/50-Km-limit.jpg
```

Fig. 8. Características de la red neuronal

RELU, sin embargo a prueba error reduje el error de tal forma que la eficiencia de la red pudiera reconocer las características de las señales de transito, ademas me di cuenta que entre mas épocas son, la reducción de error se va acercando al 100%, el problema es que el aprendizaje es mas lento.

Referencias

$https://www.quora.com/What-is-the-meaning-of-flattening-step-in-a-convolutional-neural-network$