Dropout en Redes Neuronales con TensorFlow v2.0

CHEAT SHEET

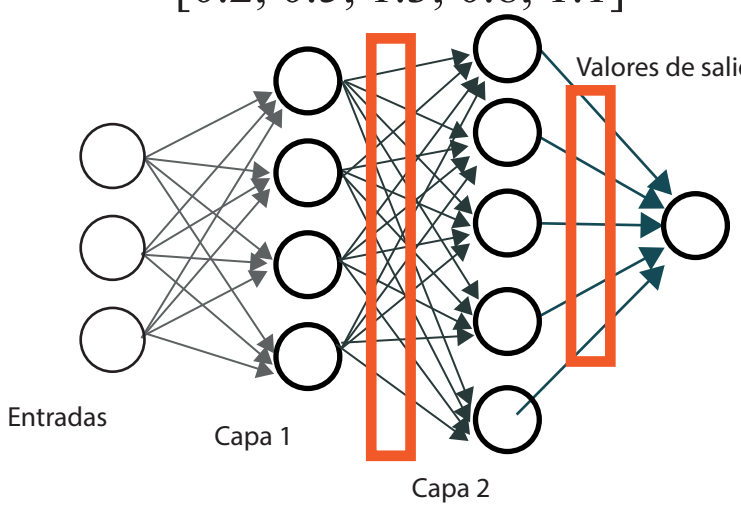
FEREBELL

DROPOUT

Es una de las técnicas de regularización más efectivas y utilizadas en redes neuronales. Dropout aplicado a una capa consiste en cambiar los valores de salida de una capa, de forma aleatoria, por ceros durante el entrenamiento.

Veamos un ejemplo, supongamos que la capa tiene los siguientes valores de salida, a manera de vector:

[0.2, 0.5, 1.3, 0.8, 1.1]



Para una entrada dada durante el entrenamiento, después de aplicar el dropout, el vector mostrado tendrá algunos valores en cero, de esta forma:

[0, 0.5, 1.3, 0, 1.1]

El dropout rate es la fracción del vector que se ha convertido en cero, usualmente se establece entre 0.2 y 0.5. Durante la validación a ningún valor se le aplica el dropout, en cambio los valores de la capa de salida disminuen en un factor igual al dropout rate, esto para balancear el hecho de que más unidades están activas que durante el entrenamiento.

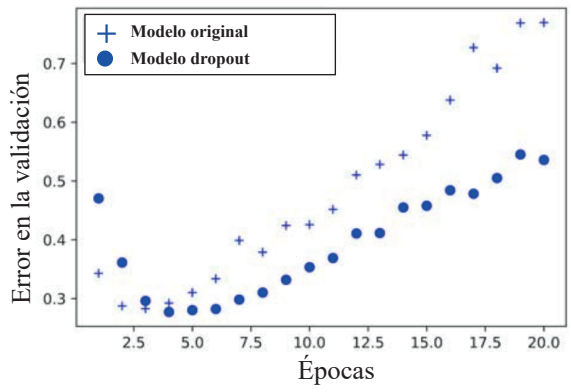
dropout rate = 0.5

0.3	0.2	1.5	0.0
0.6	0.1	0.0	0.3
0.2	1.9	0.3	1.2
0.7	0.5	1.0	0.0

50% DROPOUT

0.0	0.2	1.5	0.0
0.6	0.1	0.0	0.3
0.0	1.9	0.3	0.0
0.7	0.0	0.0	0.0

Es posible que esta técnica parezca rara y arbitraria y tal vez también estés pensando por qué esto ayudaría a evitar el sobre entrenamiento. Digamos que si removemos de forma aleatoria un conjunto de neuronas en cada capa previene posibles conspiraciones entre los datos y reduce la posibilidad de mostrar un sobre entrenamiento. La idea básica es que al introducir ruido a los valores de salida de una capa puede evitar la casualidad de identificar patrones insignificantes.



Como podemos ver al aplicar dropout al modelo el error en la validación disminuye, lo que nos dice que la red neuronal está haciendo bien su trabajo de clasificación o predicción.

EN TENSORFLOW

```
[1] import tensorflow as tf
    from tensorflow.keras.layers Dropout
    from tensorflow.keras import models, layers
    import numpy as np
```

```
[2] tf.random.set_seed(0)
    layer = tf.keras.layers.Dropout(.2, input_shape=(2,))
    data = np.arange(10).reshape(5, 2).astype(np.float32)
    print(data)
```

```
[3] outputs = layer(data, training=True)
    print(outputs)
```

METIENDONOS AL CODIGO

[1] Importando la librería de tensorflow y lo necesario para crear un modelo de red, con una sola capa.

[2] Creamos una matriz de números random, con _seed conseguimos que los mismos números aparezcan. Creamos una capa a la que aplicamos un dropout del 20% (rate) y con input_shape ordenamos que entre como un vector de 2 posiciones y redimensionamos los datos en una matriz de 5,2.

[3] Creamos la salida del modelo y lo entrenamos, finalmente impirmimos los resultados. Estos mostrarán la matriz creada anteriormente con algunas posiciones en cero, como parte de la regularización dropout.

Esta regularización puede ser aplicada a un modelo de red neuronal completo destinado a una tarea específica. Sirva lo anterior como ejemplo de aplicación con TensorFlow.