03008915 C/ Ferrocaril, 22, 03570 La Vila Joiosa **Tel** 966870140 **Fax** 966870141 http://portal.edu.gva.es/iesmarcoszaragoza

1 DEEP LEARNING – Overfitting, Aum. de datos y Dropout

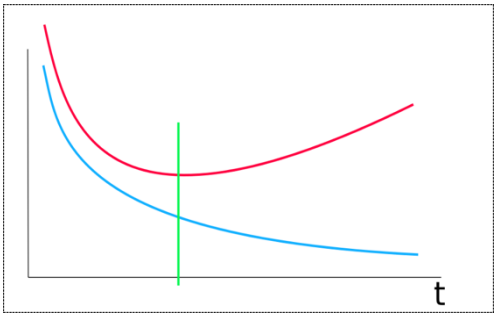
1.1 ¿Qué es el Overfitting (sobreajuste)?

El objetivo de los modelos de aprendizaje profundo es generalizar bien con la ayuda de los datos de entrenamiento a cualquier dato del dominio del problema. Esto es muy importante, ya que queremos que nuestro modelo haga predicciones sobre el conjunto de datos no visto, es decir, que nunca ha visto antes.

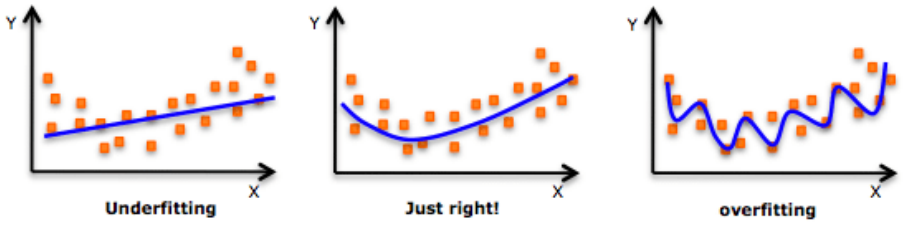
En el Overfitting, el modelo intenta **aprender demasiados detalles de los datos de entrenamiento** junto con el ruido de los datos de entrenamiento. Como resultado, el rendimiento del modelo es muy pobre en los conjuntos de datos desconocidos o de prueba. Por lo tanto, **la red no consigue generalizar** las características o patrones presentes en el conjunto de datos de entrenamiento.

1.2 ¿Cómo podemos detectar el sobreajuste mientras entrenamos nuestro modelo?

El Overfitting durante el entrenamiento puede detectarse cuando el error en los datos de entrenamiento disminuye a un valor muy pequeño, pero el error en los nuevos datos o datos de prueba aumenta a un valor grande.



Este gráfico representa la curva de error frente a iteración que muestra cómo una red neuronal profunda sobreajusta los datos de entrenamiento. Aquí, la curva azul representa el error de entrenamiento y la curva roja representa el error de prueba. El punto en el que se cruza la línea verde es el punto en el que la red empieza a sobreajustarse. Como se puede observar, el error de prueba aumenta bruscamente mientras que el error de entrenamiento disminuye.



An example of overfitting, underfitting and a model that's "just right!"

03008915 C/ Ferrocaril, 22, 03570 La Vila Joiosa **Tel** 966870140 **Fax** 966870141 http://portal.edu.gva.es/iesmarcoszaragoza

La figura anterior, para un modelo de regresión lineal simple, describe cómo el modelo intenta incluir todos los puntos de datos en el conjunto de entrenamiento. Por lo tanto, cuando un nuevo conjunto de puntos de datos esto dará lugar a un pobre rendimiento del modelo, ya que está muy cerca de todos los puntos de entrenamiento que son ruido y valores atípicos. El error en los puntos de entrenamiento es mínimo o muy pequeño, pero el error en los nuevos puntos de datos será alto.

1.3 Razones del sobreajuste

Las posibles razones del sobreajuste en las redes neuronales son las siguientes:

1.3.1 El tamaño del conjunto de datos de entrenamiento es pequeño

Cuando la red intenta aprender de un conjunto de datos pequeño, tenderá a tener un mayor control sobre el conjunto de datos y se asegurará de satisfacer exactamente todos los puntos de datos. Por lo tanto, la red intenta memorizar todos y cada uno de los puntos de datos y no consigue captar la tendencia general del conjunto de datos de entrenamiento.

1.3.2 El modelo intenta hacer predicciones sobre datos ruidosos

El Overfitting también se produce cuando el modelo intenta hacer predicciones sobre datos que tienen mucho ruido, lo que se debe a un modelo demasiado complejo con demasiados parámetros. Por ello, el modelo sobreajustado es inexacto, ya que la tendencia no refleja la realidad presente en los datos.

1.4 Reducir la complejidad del modelo

1.4.1 ¿Por qué las redes neuronales profundas son propensas al sobreajuste?

Las redes neuronales profundas son propensas al sobreajuste porque aprenden millones o miles de millones de parámetros mientras construyen el modelo. Un modelo con tantos parámetros puede sobreajustar los datos de entrenamiento porque tiene capacidad suficiente para hacerlo.

La idea básica para resolver el problema del sobreajuste es **reducir la complejidad del modelo.** Para ello, podemos hacer la red más pequeña simplemente eliminando capas o reduciendo el número de neuronas, etc.

1.4.2 ¿Cómo se reduce el Overfitting cuando eliminamos las capas o reducimos el número de neuronas?

Al eliminar algunas capas o reducir el número de neuronas, la red se vuelve menos propensa al sobreajuste, ya que se eliminan o desactivan las neuronas que contribuyen al sobreajuste.

03008915 C/ Ferrocaril, 22, 03570 La Vila Joiosa **Tel** 966870140 **Fax** 966870141 http://portal.edu.gva.es/iesmarcoszaragoza

Por lo tanto, la red tiene un menor número de parámetros que aprender, por lo que no puede memorizar todos los puntos de datos y se verá obligada a generalizar.

1.5 Aumento de datos

Una de las mejores técnicas para reducir el sobreajuste es aumentar el tamaño del conjunto de datos de entrenamiento. Cuando el tamaño de los datos de entrenamiento es pequeño, la red tiende a tener un mayor control sobre los datos de entrenamiento.

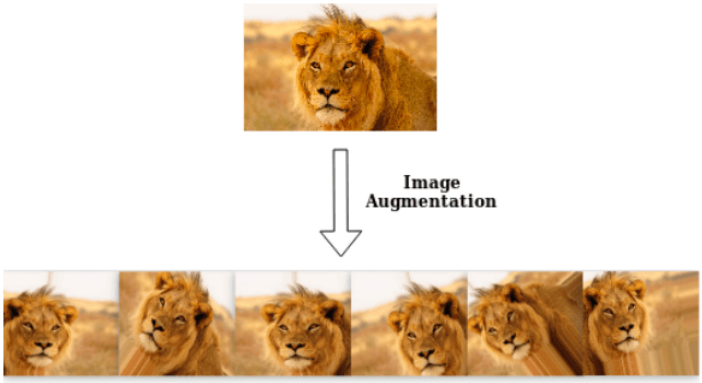
Por lo tanto, para aumentar el tamaño de los datos de entrenamiento, es decir, aumentar el número de imágenes presentes en el conjunto de datos, podemos utilizar el **aumento de datos**, que es la forma más fácil de diversificar nuestros datos y hacer que los datos de entrenamiento sean más grandes.

Algunas de las técnicas más populares de aumento de imágenes son el volteo, la traslación, la rotación, el escalado, el cambio de brillo, la adición de ruido, etcétera, etcétera.

Pero, ¿por qué nos centramos en el aumento de datos en lugar de recopilar más datos en el paso de recopilación de datos?

En el mundo real, recopilar grandes cantidades de datos es una tarea tediosa y lenta, por lo que la recopilación de nuevos datos no es una opción viable.

Esta técnica se muestra en el siguiente diagrama.



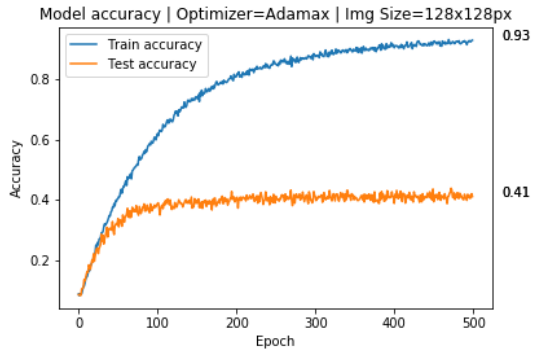
Como podemos ver, utilizando el aumento de datos, podemos generar una gran cantidad de imágenes similares, y la red se entrena en múltiples instancias de la misma clase de objetos desde diferentes perspectivas. Esto ayuda a aumentar el tamaño del conjunto de datos y, por tanto, reduce el sobreajuste, ya que, **a medida que añadimos más y más datos, el modelo no puede sobreajustar todas las muestras y se ve obligado a generalizar**.

03008915 C/ Ferrocaril, 22, 03570 La Vila Joiosa **Tel** 966870140 **Fax** 966870141 http://portal.edu.gva.es/iesmarcoszaragoza

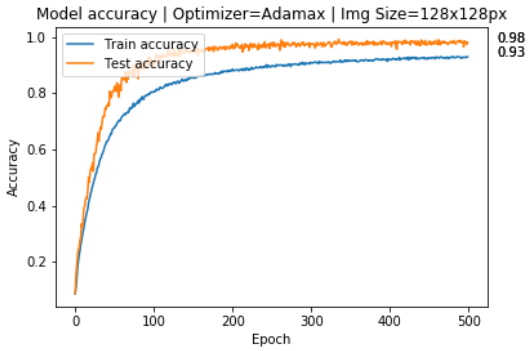
Por lo tanto, la idea que subyace al aumento de datos es que, al aumentar el tamaño del conjunto de datos de entrenamiento, la red no puede sobreajustar todo el conjunto de datos de entrenamiento (imágenes originales + imágenes aumentadas) y, por lo tanto, se ve obligada a **generalizar**. Pero la pérdida global de entrenamiento aumenta, ya que la red no predice con precisión las imágenes aumentadas, lo que incrementa la pérdida de entrenamiento, y el optimizador (algoritmo de optimización) ajusta la red para captar la tendencia generalizada presente en el conjunto de datos de entrenamiento.

La representación gráfica de lo anterior es la siguiente:

Antes del aumento de datos:



Después del aumento de datos:



1.6 Dropout

Es otra técnica de regularización que evita el sobreajuste de las redes neuronales. Esta técnica modifica la propia red para evitar que se sobreajuste.

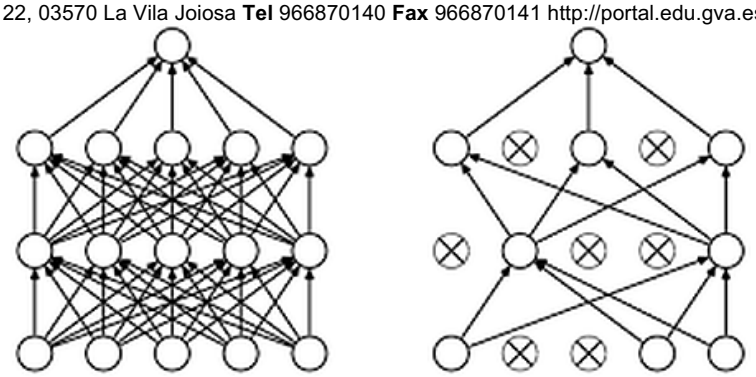
1.6.1 Principio de funcionamiento de esta técnica

Elimina aleatoriamente algunas neuronas, excepto la capa de salida, de la red neuronal durante el entrenamiento en cada iteración o podemos asignar una probabilidad *p* a todas las neuronas de una red para que sean ignoradas temporalmente de los cálculos.

donde *p* se conoce como Tasa de abandono y suele estar entre 0.2 y 0.5.

Entonces, a medida que avanza cada iteración, las neuronas de cada capa con la probabilidad más alta se descartan. Esto resulta en la creación de una red más pequeña con cada pasada en el conjunto de datos de entrenamiento (epoch). Como en cada iteración se puede eliminar un valor de entrada aleatorio, la red intenta equilibrar el riesgo y no favorecer ninguna de las características y reduce el sesgo y el ruido.

03008915 C/ Ferrocaril, 22, 03570 La Vila Joiosa **Tel** 966870140 **Fax** 966870141 http://portal.edu.gva.es/iesmarcoszaragoza



(a) Standard Neural Net (b) After applying dropout.

Ejemplo de creación de dropout mediante Tensorflow y Python:

```
model = Sequential()
model.add(Dense(60, input_shape=(60,), activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(30, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(1, activation='sigmoid'))
```

En este caso, en las capas de 60 y de 30 nodos se elimina el 20% de los mismos.