PASOS PARA HACER FUNCIONAR DEEP LEARNING

1. Creamos las carpetas de imágenes que contendrán las imágenes de entrenamiento y las de test, la carpeta de entrenamiento debe contener un mínimo de entre 15000 y 25000 imágenes pare que el entrenamiento sea preciso.
   1. Las imágenes deben contener como nombre el tipo del cual se trata y un índice ejem: perro.4564.jpg
2. Ahora hay que crear el LMDB que contendrá los descriptores de las imágenes

Ejecutamos: ***python create\_lmdb.py*** puede tardar unos minutos

1. Una vez creado los lmdb ahora hay crear el archivo que contiene en binaryproto, que es un archivo que contiene un índice de los pixeles de las imágenes y los resta poner media cero.

Ejecutamos:

***/home/user\_cudnn/caffe/build/tools/compute\_image\_mean -backend=lmdb train\_lmdb /home/user\_cudnn/pruebanet/mean.binaryproto***

1. Ahora hay que definir el archivo que contiene la red para el entrenamiento.

En nuestro caso: /home/user\_cudnn/pruebanet/pruebanet\_train\_val\_1.prototxt

* 1. Podemos dibujar la red ejecutando:

***python /home/user\_cudnn/caffe/python/draw\_net.py /home/user\_cudnn/pruebanet/pruebanet\_train\_val\_1.prototxt ImagenRed.png***

1. Una vez tenemos el archivo con la red de entrenamiento hay que hacer uno para el solver que contiene los parámetros para entrenar nuestra red.

En nuestro caso: /home/user\_cudnn/pruebanet/pruebanet\_solver.prototxt

1. Con estos archivos creados ahora toca entrenar la red y crear un modelo pre-entrenado, también crearemos un log para posteriormente hacer la gráfica con la curva de entrenamiento.

Ejecutamos:

***nohup /home/user\_cudnn/caffe/build/tools/caffe train --solver /home/user\_cudnn/pruebanet/pruebanet\_solver.prototxt 2>&1 | tee /home/user\_cudnn/pruebanet/ pruebanet\_train.log***

Esto tarda en torno a 16 horas con las imágenes que hay ahora mismo en el servidor

1. Después de finalizar en entrenamiento cada 5000 iteraciones se nos guarda una imagen de modelo pre-entrenado y un log con todo el proceso, ahora podemos dibujar la curva de entrenamiento, que nos mostrara las perdidas y precisión que ha alcanzado nuestra red.

Ejecutamos:

***python /home/user\_cudnn/caffe/tools/extra/parse\_log.py caffe\_model\_1/model\_1\_train.log /home/user\_cudnn/pruebanet/ --delimiter " "***

***python /home/user\_cudnn/pruebanet/dibuja\_grafica.py /home/user\_cudnn/pruebanet/ pruebanet\_train.log grafica.png***

1. Como último paso, podemos hacer las predicciones, para ello vamos a usar las imágenes que hay en la carpeta test.

Para ello, necesitamos hacer un archivo que es parecido al que usamos para entrenar la red, pero ahora es para hacer predicciones.

En nuestro caso: /home/user\_cudnn/pruebanet/pruebanet\_deploy.prototxt

Ejecutamos: ***python /home/user\_cudnn/pruebanet/prediccion.py***

Esto nos mostrara un listado de las predicciones con los distintos porcentajes, también creara un archivo csv con los resultados.

Importante: Los comandos son de ejemplo, hay que revisar la localización y nombre de los archivos.

Los archivos adjuntos, son los usados para el ejemplo, hay que modificar los mean\_file, source y rutas dentro de los mismos para hacerlos coincidir.