

Conclusiones

El objetivo principal de este tutorial, es **clasificar imágenes** RGB de 32x32 pixeles, en 10 diferentes categorías: avión, automóvil, pájaro, gato, ciervo, perro, rana, caballo, barco y camión. Una vez se comienza a realizar el entrenamiento de la red, primero se procede a descargar un paquete de imágenes que se utilizarán para el entrenamiento (50 000 en total) y para pruebas (10 000 en total).

Según el modelo de entrenamiento:

- Se **entrena** la red con el algoritmo descendiente del gradiente estándar.
- El learning rate **decae** exponencialmente con el tiempo.
- Las imágenes se **cortan** a 24x24 pixeles, principalmente para evaluarlas o de forma aleatoria para utilizarlas como entrenamiento.
- Las imágenes se **blanquean** un poco para que el modelo sea insensible al 'Dynamic Range'.
- Aleatoriamente, se **rotan** las imágenes de izquierda a derecha.
- Aleatoriamente, se distorsiona el **brillo** de la imagen.
- Aleatoriamente, se distorsiona el **contraste** de la imagen.

Cuando comencé el entrenamiento, corrí el archivo de '**cifar10_train.py**' tal y como se muestra en el tutorial, me apareció el problema de que la computadora ya llevaba **más de 12 horas entrenando** la red neuronal, y a los cercanos 80 000 steps, no daba señal de terminar en un futuro cercano, por lo que decidí entrenar el programa y buscar una forma de disminuir ese tiempo de entrenamiento.

En el código, encontré un parámetro determinado como '**—max_steps**' que tenía como valor pre determinado 1 000 000.

```
parser.add_argument('--max_steps', type=int, default=1000000,  
                    help='Number of batches to run.')
```

Procedí a **disminuirlo** a 10 000, en donde calculé que la computadora podría durar en el entrenamiento unas **4 horas**.

Una vez realizado el entrenamiento, el cuál finalmente le tomó **3 horas** a la computadora, se obtuvo una **precisión de 81.8%**, un poco menor a la del tutorial que obtuvo 86% pero que igualmente es aceptable.

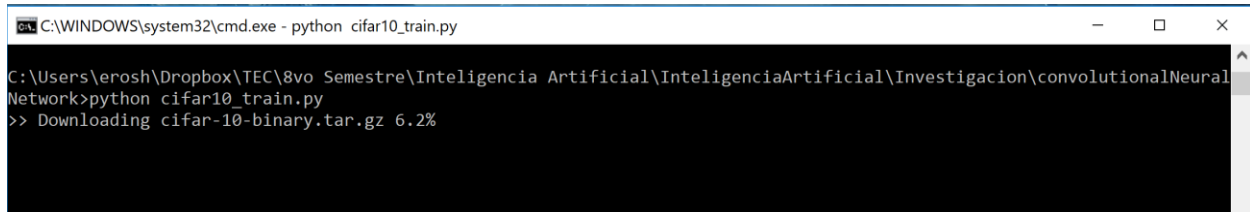
Cómo podría aplicarse a un problema real

Como este tutorial ha sido enfocado en clasificación de imágenes en 10 categorías, la aplicación al problema debe adaptarse a ella y no se le puede dar continuidad a las propuestas anteriormente.

Considero que una solución aplicable de esto, es **al reconocimiento de vehículos aéreos**. Se ha visto mucho en televisión o noticias sobre imágenes que las personas toman sobre un “objeto volador” en donde no se identifica qué puede ser, utilizando un entrenamiento para **clasificar estas imágenes** y determinar qué pueden ser los elementos que son capturados en los aires, primeramente, en términos de este tutorial, sería principalmente para pájaros o aviones, pero en un futuro las categorías podrían aumentarse.

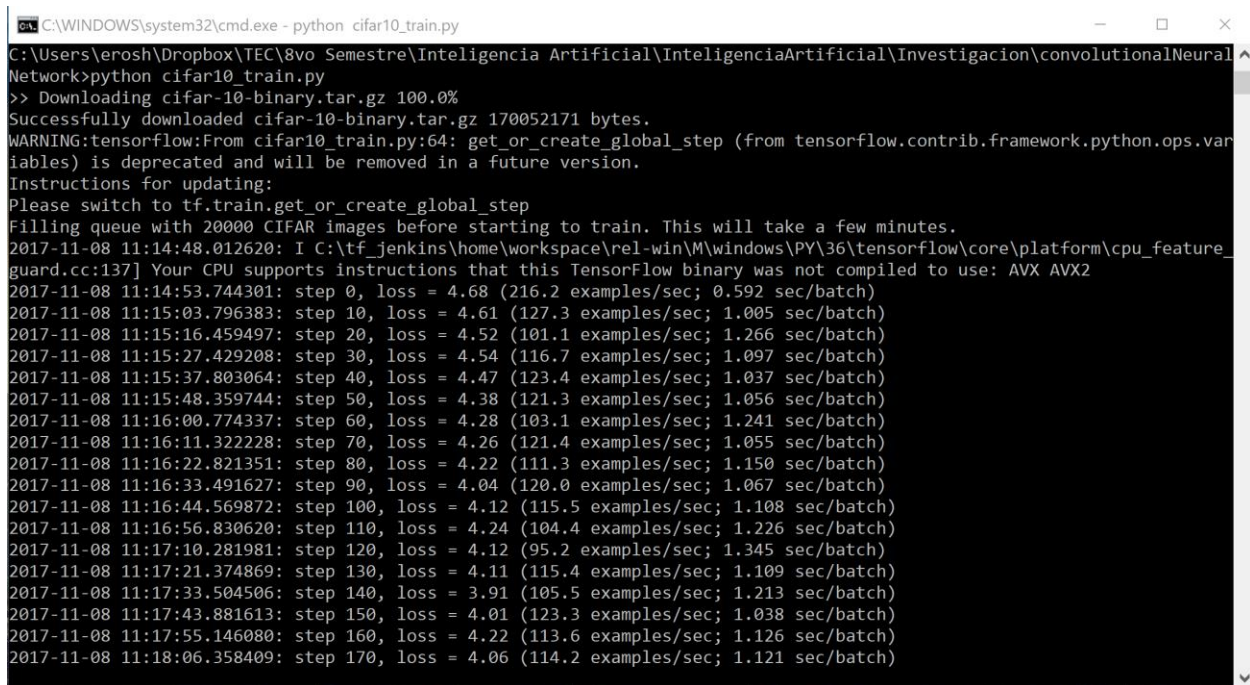
Screenshots de Resultados

Inicialmente, comencé ejecutando el programa 'cifar10_train.py' y el primer proceso que realiza es descargar el paquete de imágenes.



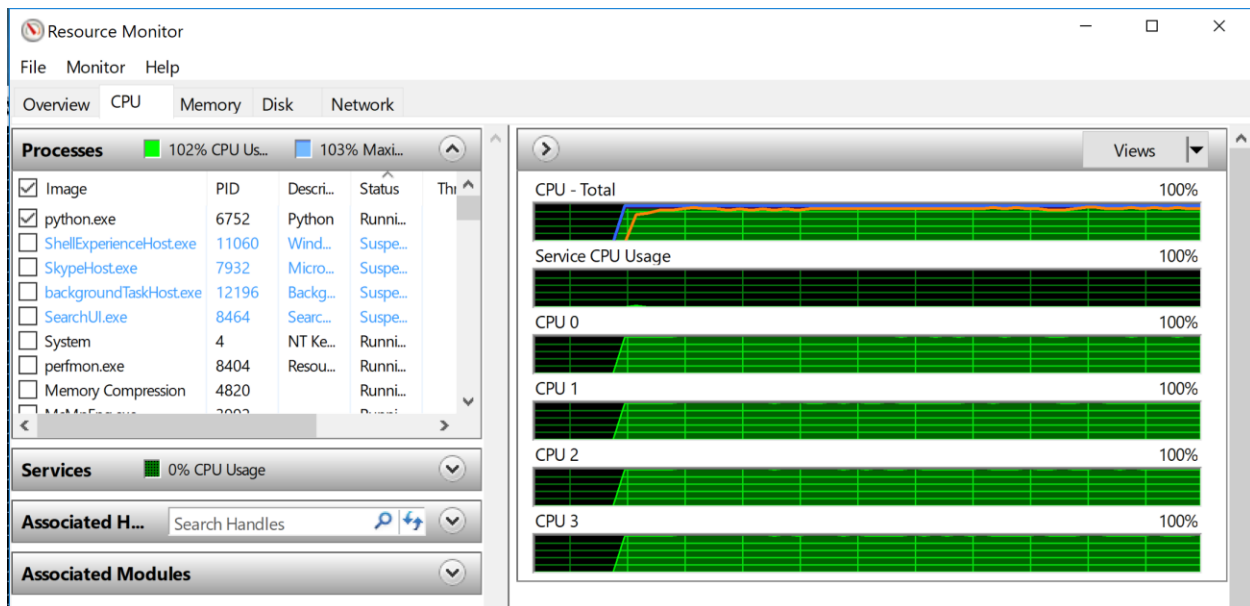
```
C:\WINDOWS\system32\cmd.exe - python cifar10_train.py
C:\Users\erosh\Dropbox\TEC\8vo Semestre\Inteligencia Artificial\InteligenciaArtificial\Investigacion\convolutionalNeural
Network>python cifar10_train.py
>> Downloading cifar-10-binary.tar.gz 6.2%
```

Una vez termina la descarga, se comienza con el entrenamiento, en donde se puede destacar que el **loss** con el que se comienza es de **4.68**.

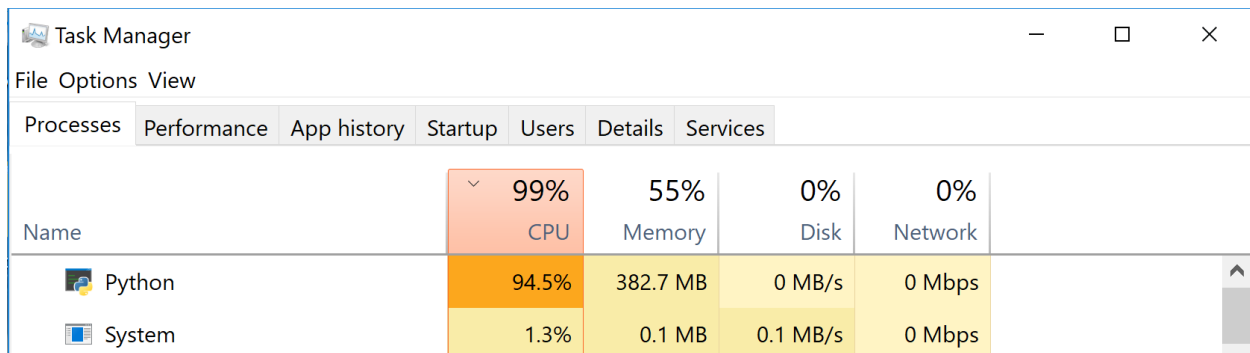


```
C:\WINDOWS\system32\cmd.exe - python cifar10_train.py
C:\Users\erosh\Dropbox\TEC\8vo Semestre\Inteligencia Artificial\InteligenciaArtificial\Investigacion\convolutionalNeural
Network>python cifar10_train.py
>> Downloading cifar-10-binary.tar.gz 100.0%
Successfully downloaded cifar-10-binary.tar.gz 170052171 bytes.
WARNING:tensorflow:From cifar10_train.py:64: get_or_create_global_step (from tensorflow.contrib.framework.python.ops.var
iables) is deprecated and will be removed in a future version.
Instructions for updating:
Please switch to tf.train.get_or_create_global_step
Filling queue with 20000 CIFAR images before starting to train. This will take a few minutes.
2017-11-08 11:14:48.012620: I C:\tf_jenkins\home\workspace\rel-win\M\windows\PY\36\tensorflow\core\platform\cpu_feature_
guard.cc:137] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX AVX2
2017-11-08 11:14:53.744301: step 0, loss = 4.68 (216.2 examples/sec; 0.592 sec/batch)
2017-11-08 11:15:03.796383: step 10, loss = 4.61 (127.3 examples/sec; 1.005 sec/batch)
2017-11-08 11:15:16.459497: step 20, loss = 4.52 (101.1 examples/sec; 1.266 sec/batch)
2017-11-08 11:15:27.429208: step 30, loss = 4.54 (116.7 examples/sec; 1.097 sec/batch)
2017-11-08 11:15:37.803064: step 40, loss = 4.47 (123.4 examples/sec; 1.037 sec/batch)
2017-11-08 11:15:48.359744: step 50, loss = 4.38 (121.3 examples/sec; 1.056 sec/batch)
2017-11-08 11:16:00.774337: step 60, loss = 4.28 (103.1 examples/sec; 1.241 sec/batch)
2017-11-08 11:16:11.322228: step 70, loss = 4.26 (121.4 examples/sec; 1.055 sec/batch)
2017-11-08 11:16:22.821351: step 80, loss = 4.22 (111.3 examples/sec; 1.150 sec/batch)
2017-11-08 11:16:33.491627: step 90, loss = 4.04 (120.0 examples/sec; 1.067 sec/batch)
2017-11-08 11:16:44.569872: step 100, loss = 4.12 (115.5 examples/sec; 1.108 sec/batch)
2017-11-08 11:16:56.830620: step 110, loss = 4.24 (104.4 examples/sec; 1.226 sec/batch)
2017-11-08 11:17:10.281981: step 120, loss = 4.12 (95.2 examples/sec; 1.345 sec/batch)
2017-11-08 11:17:21.374869: step 130, loss = 4.11 (115.4 examples/sec; 1.109 sec/batch)
2017-11-08 11:17:33.504506: step 140, loss = 3.91 (105.5 examples/sec; 1.213 sec/batch)
2017-11-08 11:17:43.881613: step 150, loss = 4.01 (123.3 examples/sec; 1.038 sec/batch)
2017-11-08 11:17:55.146080: step 160, loss = 4.22 (113.6 examples/sec; 1.126 sec/batch)
2017-11-08 11:18:06.358409: step 170, loss = 4.06 (114.2 examples/sec; 1.121 sec/batch)
```

En este entrenamiento, el uso de los núcleos de la computadora vuelve a estar en el máximo.



Se puede apreciar como aproximadamente un **94% del CPU** está siendo utilizado por el programa.



A medida que avanza el entrenamiento y se realizan más steps, se puede ver como el **loss va disminuyendo**, comenzó en aproximadamente 4 y ya va siendo menos de 2.

```
C:\WINDOWS\system32\cmd.exe - python cifar10_train.py
2017-11-08 11:54:23.470003: step 1090, loss = 2.31 (81.9 examples/sec; 1.562 sec/batch)
2017-11-08 11:54:41.969718: step 1100, loss = 2.13 (69.2 examples/sec; 1.850 sec/batch)
2017-11-08 11:54:57.481554: step 1110, loss = 2.42 (82.5 examples/sec; 1.551 sec/batch)
2017-11-08 11:55:08.017583: step 1120, loss = 2.15 (121.5 examples/sec; 1.054 sec/batch)
2017-11-08 11:55:21.996784: step 1130, loss = 2.17 (91.6 examples/sec; 1.398 sec/batch)
2017-11-08 11:55:34.054361: step 1140, loss = 2.19 (106.2 examples/sec; 1.206 sec/batch)
2017-11-08 11:55:46.050276: step 1150, loss = 2.34 (106.7 examples/sec; 1.200 sec/batch)
2017-11-08 11:55:57.849164: step 1160, loss = 2.34 (108.5 examples/sec; 1.180 sec/batch)
2017-11-08 11:56:13.589038: step 1170, loss = 2.17 (81.3 examples/sec; 1.574 sec/batch)
2017-11-08 11:56:27.398777: step 1180, loss = 2.74 (92.7 examples/sec; 1.381 sec/batch)
2017-11-08 11:56:38.989112: step 1190, loss = 2.19 (110.4 examples/sec; 1.159 sec/batch)
2017-11-08 11:56:51.009089: step 1200, loss = 2.18 (106.5 examples/sec; 1.202 sec/batch)
2017-11-08 11:57:02.827542: step 1210, loss = 2.20 (108.3 examples/sec; 1.182 sec/batch)
2017-11-08 11:57:14.376262: step 1220, loss = 2.22 (110.8 examples/sec; 1.155 sec/batch)
2017-11-08 11:57:24.715769: step 1230, loss = 2.32 (123.8 examples/sec; 1.034 sec/batch)
2017-11-08 11:57:35.200622: step 1240, loss = 2.03 (122.1 examples/sec; 1.048 sec/batch)
2017-11-08 11:57:48.236341: step 1250, loss = 2.42 (98.2 examples/sec; 1.304 sec/batch)
2017-11-08 11:57:59.737490: step 1260, loss = 2.25 (111.3 examples/sec; 1.150 sec/batch)
2017-11-08 11:58:11.303812: step 1270, loss = 1.96 (110.7 examples/sec; 1.157 sec/batch)
2017-11-08 11:58:22.838596: step 1280, loss = 2.14 (111.0 examples/sec; 1.153 sec/batch)
2017-11-08 11:58:34.374363: step 1290, loss = 2.14 (111.0 examples/sec; 1.154 sec/batch)
2017-11-08 11:58:48.191139: step 1300, loss = 2.21 (92.6 examples/sec; 1.382 sec/batch)
2017-11-08 11:59:03.368521: step 1310, loss = 2.23 (84.3 examples/sec; 1.518 sec/batch)
2017-11-08 11:59:17.401857: step 1320, loss = 2.05 (91.2 examples/sec; 1.403 sec/batch)
2017-11-08 11:59:31.521417: step 1330, loss = 2.18 (90.7 examples/sec; 1.412 sec/batch)
2017-11-08 11:59:46.631616: step 1340, loss = 2.04 (84.7 examples/sec; 1.511 sec/batch)
2017-11-08 11:59:59.946552: step 1350, loss = 2.08 (96.1 examples/sec; 1.331 sec/batch)
2017-11-08 12:00:11.413082: step 1360, loss = 2.05 (111.6 examples/sec; 1.147 sec/batch)
2017-11-08 12:00:21.990722: step 1370, loss = 1.92 (121.0 examples/sec; 1.058 sec/batch)
```

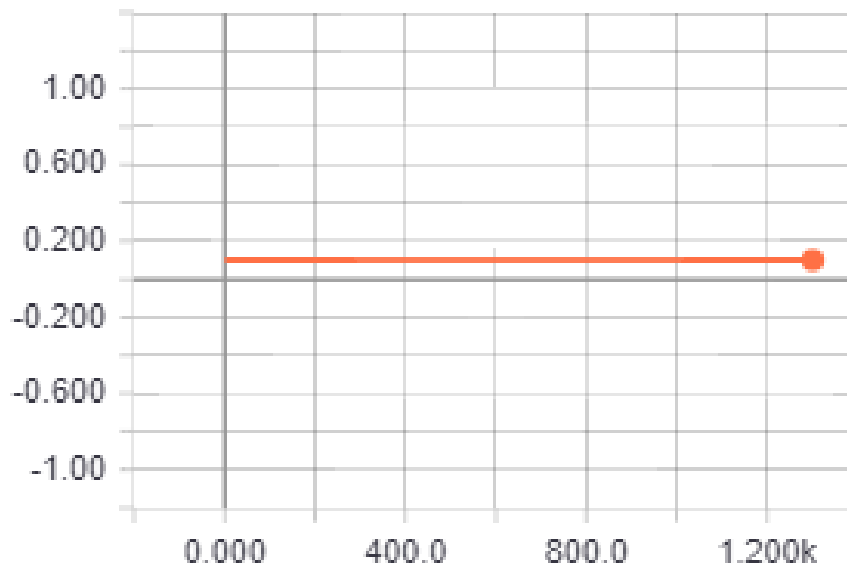
El loss continúa disminuyendo

```
C:\WINDOWS\system32\cmd.exe - python cifar10_train.py
2017-11-08 13:40:59.437454: step 7080, loss = 0.97 (127.5 examples/sec; 1.004 sec/batch)
2017-11-08 13:41:10.177894: step 7090, loss = 0.83 (119.2 examples/sec; 1.074 sec/batch)
2017-11-08 13:41:20.025155: step 7100, loss = 0.83 (130.0 examples/sec; 0.985 sec/batch)
2017-11-08 13:41:29.943083: step 7110, loss = 0.77 (129.1 examples/sec; 0.992 sec/batch)
2017-11-08 13:41:40.044393: step 7120, loss = 0.97 (126.7 examples/sec; 1.010 sec/batch)
2017-11-08 13:41:50.752081: step 7130, loss = 0.97 (119.5 examples/sec; 1.071 sec/batch)
2017-11-08 13:42:01.553381: step 7140, loss = 0.72 (118.5 examples/sec; 1.080 sec/batch)
2017-11-08 13:42:11.893468: step 7150, loss = 0.87 (123.8 examples/sec; 1.034 sec/batch)
2017-11-08 13:42:22.256751: step 7160, loss = 0.96 (123.5 examples/sec; 1.036 sec/batch)
2017-11-08 13:42:32.570754: step 7170, loss = 0.84 (124.1 examples/sec; 1.031 sec/batch)
2017-11-08 13:42:42.884754: step 7180, loss = 1.10 (124.1 examples/sec; 1.031 sec/batch)
2017-11-08 13:42:53.252892: step 7190, loss = 0.77 (123.5 examples/sec; 1.037 sec/batch)
2017-11-08 13:43:03.777459: step 7200, loss = 0.89 (121.6 examples/sec; 1.052 sec/batch)
2017-11-08 13:43:14.158924: step 7210, loss = 0.92 (123.3 examples/sec; 1.038 sec/batch)
2017-11-08 13:43:24.510994: step 7220, loss = 0.94 (123.6 examples/sec; 1.035 sec/batch)
2017-11-08 13:43:35.094677: step 7230, loss = 0.99 (120.9 examples/sec; 1.058 sec/batch)
2017-11-08 13:43:45.555033: step 7240, loss = 0.83 (122.4 examples/sec; 1.046 sec/batch)
2017-11-08 13:43:55.948243: step 7250, loss = 0.96 (123.2 examples/sec; 1.039 sec/batch)
2017-11-08 13:44:06.298384: step 7260, loss = 0.98 (123.7 examples/sec; 1.035 sec/batch)
2017-11-08 13:44:16.628495: step 7270, loss = 0.83 (123.9 examples/sec; 1.033 sec/batch)
```

Utilizando **TensorBoard** durante el proceso de entrenamiento, se pueden destacar los siguientes puntos:

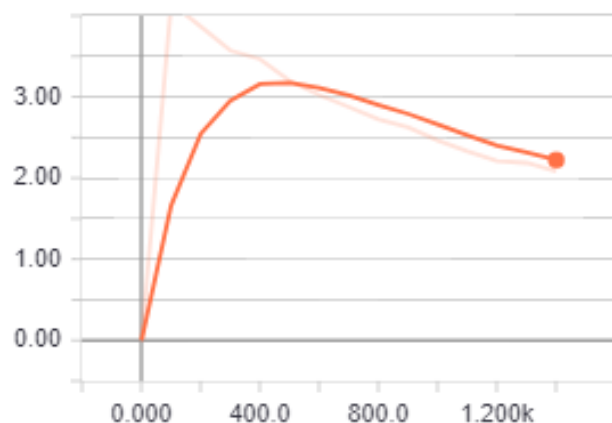
- A pesar de que en el tutorial se demuestra que durante el entrenamiento, el **learning rate** irá decreyendo de manera exponencial, en el gráfico de se muestra que mantiene un valor constante durante el entrenamiento.

learning_rate

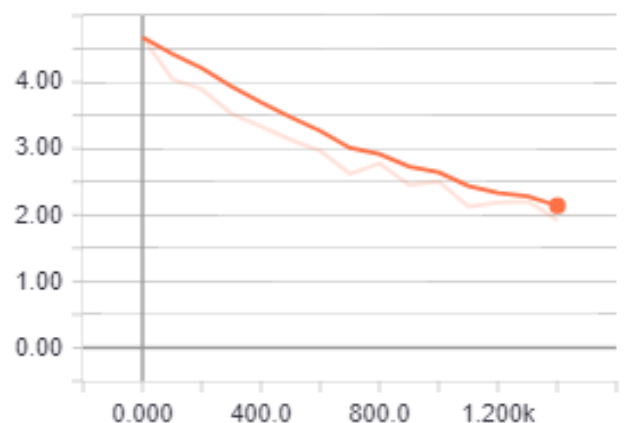


- Se puede apreciar como el **loss efectivamente va disminuyendo** a medida que pasa el tiempo con el entrenamiento.

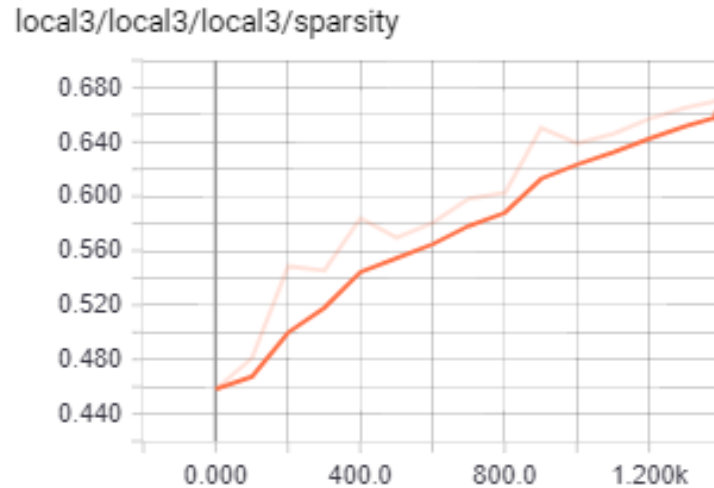
total_loss_1



total_loss_raw_



- También se puede visualizar como la distribución de activaciones y el grado de dispersión aumentan a medida que pasa el tiempo con el entrenamiento.



Finalmente, después de aproximadamente **3 horas de entrenamiento**, se terminó de entrenar la red neuronal, en donde el **loss final es de 0.7 aproximadamente**. Comparado con el **loss inicial de 4.68**, este **disminuyó aproximadamente un 4 sólido**.

```
C:\WINDOWS\system32\cmd.exe
2017-11-08 14:27:59.888736: step 9800, loss = 1.13 (118.5 examples/sec; 1.080 sec/batch)
2017-11-08 14:28:10.352670: step 9810, loss = 0.82 (122.3 examples/sec; 1.046 sec/batch)
2017-11-08 14:28:20.704973: step 9820, loss = 0.70 (123.6 examples/sec; 1.035 sec/batch)
2017-11-08 14:28:31.089869: step 9830, loss = 0.93 (123.3 examples/sec; 1.038 sec/batch)
2017-11-08 14:28:41.441462: step 9840, loss = 0.84 (123.7 examples/sec; 1.035 sec/batch)
2017-11-08 14:28:51.774181: step 9850, loss = 0.83 (123.9 examples/sec; 1.033 sec/batch)
2017-11-08 14:29:02.146503: step 9860, loss = 0.92 (123.4 examples/sec; 1.037 sec/batch)
2017-11-08 14:29:12.881754: step 9870, loss = 0.87 (119.2 examples/sec; 1.074 sec/batch)
2017-11-08 14:29:23.317767: step 9880, loss = 0.89 (122.7 examples/sec; 1.044 sec/batch)
2017-11-08 14:29:33.655198: step 9890, loss = 0.86 (123.8 examples/sec; 1.034 sec/batch)
2017-11-08 14:29:44.142321: step 9900, loss = 0.77 (122.1 examples/sec; 1.049 sec/batch)
2017-11-08 14:29:54.514295: step 9910, loss = 0.75 (123.4 examples/sec; 1.037 sec/batch)
2017-11-08 14:30:04.913511: step 9920, loss = 0.86 (123.1 examples/sec; 1.040 sec/batch)
2017-11-08 14:30:15.241998: step 9930, loss = 0.89 (123.9 examples/sec; 1.033 sec/batch)
2017-11-08 14:30:25.608691: step 9940, loss = 0.88 (123.5 examples/sec; 1.037 sec/batch)
2017-11-08 14:30:35.926659: step 9950, loss = 0.78 (124.1 examples/sec; 1.032 sec/batch)
2017-11-08 14:30:46.216229: step 9960, loss = 0.71 (124.4 examples/sec; 1.029 sec/batch)
2017-11-08 14:30:56.519667: step 9970, loss = 0.80 (124.2 examples/sec; 1.030 sec/batch)
2017-11-08 14:31:07.098812: step 9980, loss = 0.76 (121.0 examples/sec; 1.058 sec/batch)
2017-11-08 14:31:18.064012: step 9990, loss = 0.79 (116.7 examples/sec; 1.097 sec/batch)

C:\Users\erosh\Dropbox\TEC\8vo Semestre\Inteligencia Artificial\InteligenciaArtificial\Investigacion\convolutionalNeuralNetwork>
```

Luego, se procede a ejecutar el programa '**cifar10_eval.py**', esto es para probar la red neuronal ya entrenada con las 10 000 imágenes anteriormente mencionadas.

```
C:\Users\erosh\Dropbox\TEC\8vo Semestre\Inteligencia Artificial\InteligenciaArtificial\Investigacion\convolutionalNeuralNetwork>python cifar10_eval.py
2017-11-08 23:08:43.526672: I C:\tf_jenkins\home\workspace\rel-win\M\windows\PY\36\tensorflow\core\platform\cpu_feature_guard.cc:137] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX AVX2
2017-11-08 23:09:01.160492: precision @ 1 = 0.818
```

Como **resultado** se obtuvo una **precisión de 81.8%**.