Escuela de Ingeniería en Computación

Inteligencia Artificial - IC6200 - Grupo 2

Proyecto de Investigación: Convolutional Neural Networks

Estudiante:

Alanis Pineda, Stephannie - 201263425

Profesor: Luis Carlos Hernández Vega

Cartago, Costa Rica
II Semestre, 2017

Conclusiones

- La idea del tutorial fue realizar una red neuronal convolucional que pudiera clasificar imágenes de 32x32 (RGB) en 10 categorías diferentes: avión, automóvil, pájaro, gato, venado, perro, rana, caballo, barco y camión.

- El archivo con el que se entrena y prueba la red neuronal consta de 60000 imágenes, 10000 de cada categoría, 50000 son para entrenamiento y 10000 para prueba. Las imágenes son a color y de un tamaño definido (32px X 32px).

- La red es entrenada con el algoritmo descendiente del gradiente.

- Como el entrenamiento es bastante pesado, se implementaron 16 hilos para acelerar el pre-procesamiento de las imágenes, entrando cada uno en una cola para entrenar la red la cola la provee TensorFlow.

- La red se entrenó con 10000 steps, teniendo una duración de 4 horas con 30 minutos aproximadamente. Sin embargo, en el TensorBoard aparace el Learning Rate como un escalar constante.

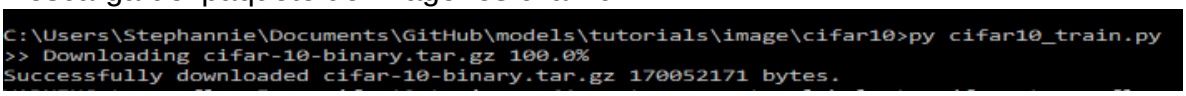
- La red neuronal es bastante confiable, cuando se probó retornó los resultados con una precisión del 81% (aproximadamente), lo cual es una buena estadística.

Aplicación a un problema real

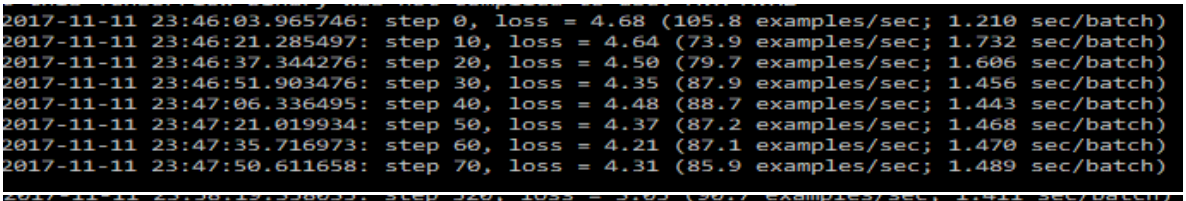
En medios de transporte para controlar la cantidad de distintos tipos de medios de transporte que transitan sobre una carretera, se puede utilizar esta red neuronal para el reconocimiento de los mismos y saber que tanto transita dicho vehículo. Se podría realizar un análisis estadístico y mejorar las carreteras para facilitar el transporte del lugar más transitado y así lograr evitar que haya tanto tráfico.

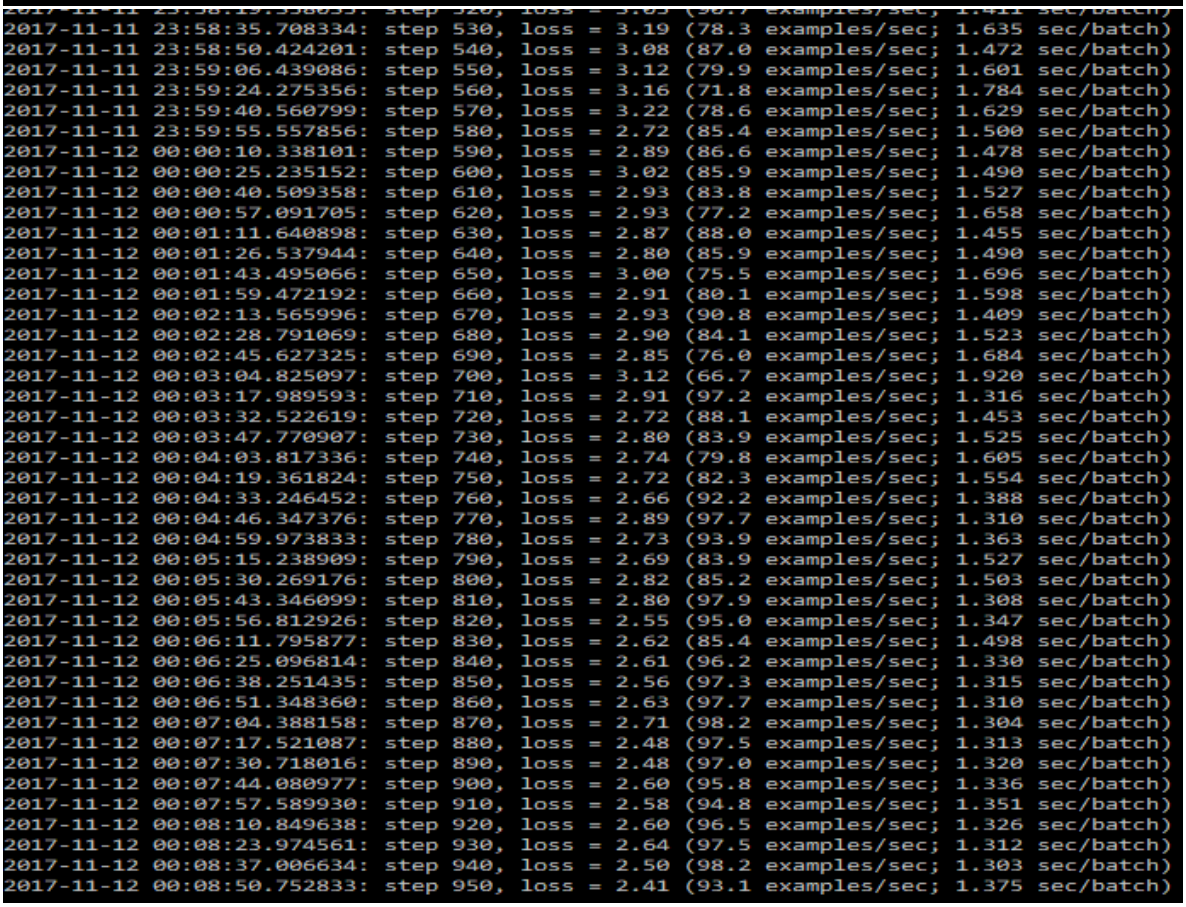
Screenshots

Descarga del paquete de imágenes cifar10.

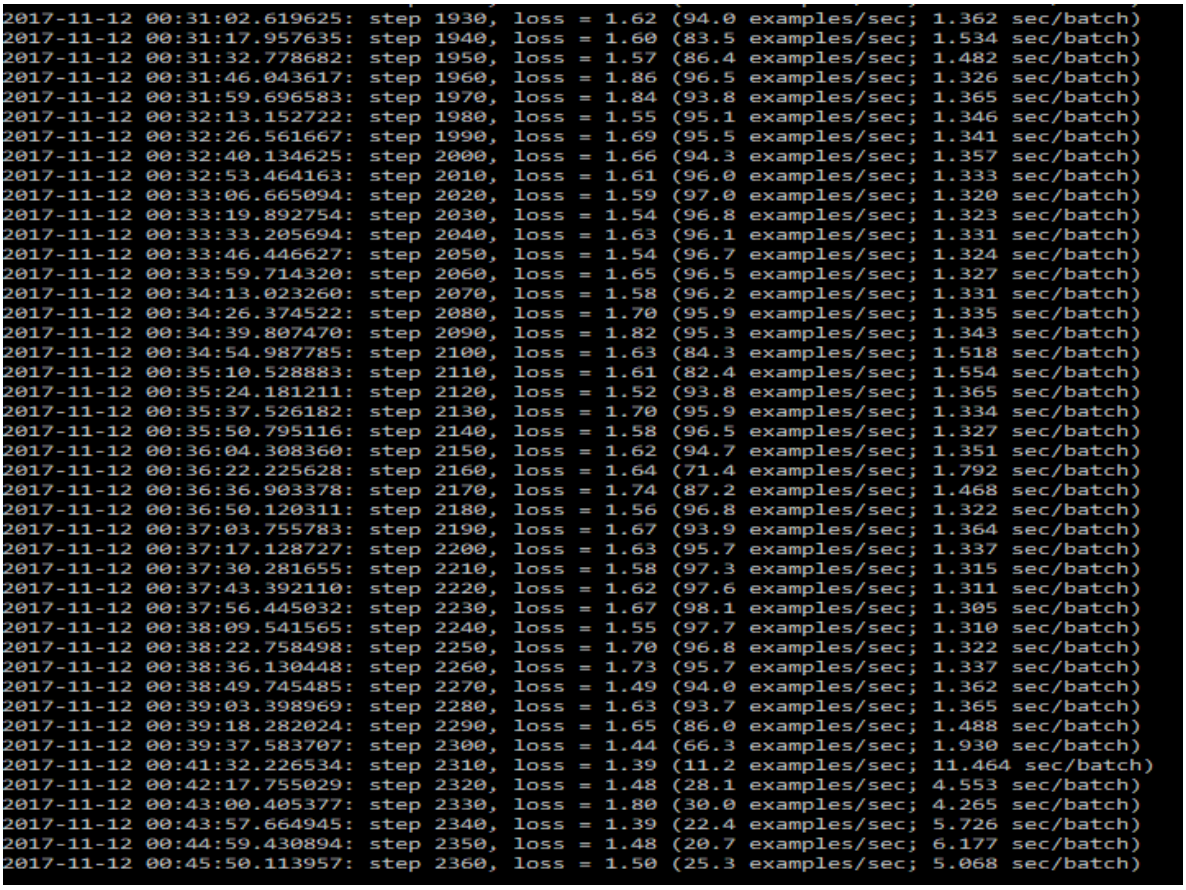


Comienzo del entrenamiento, el error es bastante alto.

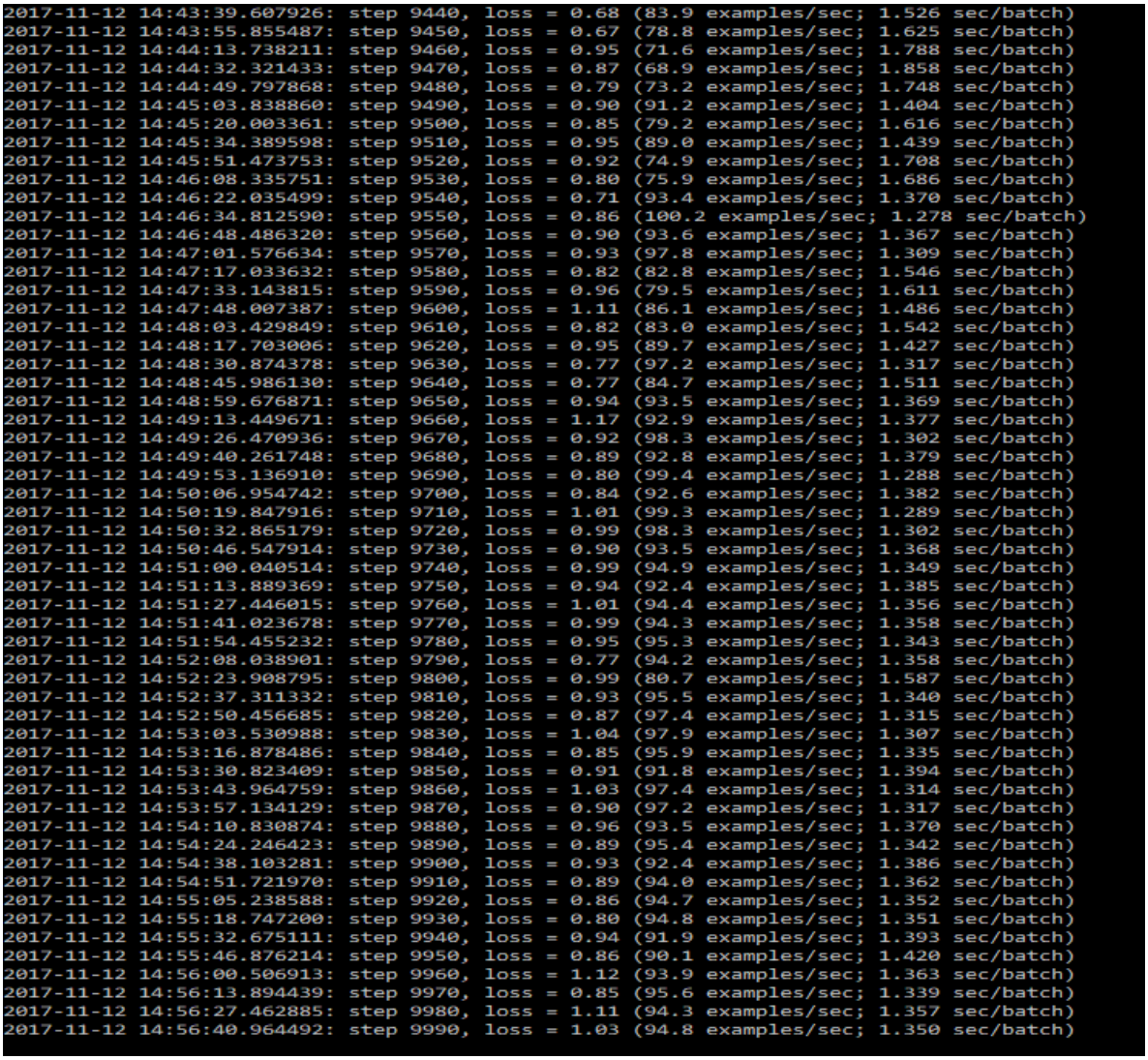




Se observa como el error va disminuyendo.



El error varía mucho entre un paso y otro, sube o baja pero termino en 1.03



```
2017-11-12 14:43:39.607926: step 9440, loss = 0.68 (83.9 examples/sec; 1.526 sec/batch)
2017-11-12 14:43:55.855487: step 9450, loss = 0.67 (78.8 examples/sec; 1.625 sec/batch)
2017-11-12 14:44:13.738211: step 9460, loss = 0.95 (71.6 examples/sec; 1.788 sec/batch)
2017-11-12 14:44:32.321433: step 9470, loss = 0.87 (68.9 examples/sec; 1.858 sec/batch)
2017-11-12 14:44:49.797868: step 9480, loss = 0.79 (73.2 examples/sec; 1.748 sec/batch)
2017-11-12 14:45:03.838860: step 9490, loss = 0.90 (91.2 examples/sec; 1.404 sec/batch)
2017-11-12 14:45:20.003361: step 9500, loss = 0.85 (79.2 examples/sec; 1.616 sec/batch)
2017-11-12 14:45:34.389598: step 9510, loss = 0.95 (89.0 examples/sec; 1.439 sec/batch)
2017-11-12 14:45:51.473753: step 9520, loss = 0.92 (74.9 examples/sec; 1.708 sec/batch)
2017-11-12 14:46:08.335751: step 9530, loss = 0.80 (75.9 examples/sec; 1.686 sec/batch)
2017-11-12 14:46:22.035499: step 9540, loss = 0.71 (93.4 examples/sec; 1.370 sec/batch)
2017-11-12 14:46:34.812590: step 9550, loss = 0.86 (100.2 examples/sec; 1.278 sec/batch)
2017-11-12 14:46:48.486320: step 9560, loss = 0.90 (93.6 examples/sec; 1.367 sec/batch)
2017-11-12 14:47:01.576634: step 9570, loss = 0.93 (97.8 examples/sec; 1.309 sec/batch)
2017-11-12 14:47:17.033632: step 9580, loss = 0.82 (82.8 examples/sec; 1.546 sec/batch)
2017-11-12 14:47:33.143815: step 9590, loss = 0.96 (79.5 examples/sec; 1.611 sec/batch)
2017-11-12 14:47:48.007387: step 9600, loss = 1.11 (86.1 examples/sec; 1.486 sec/batch)
2017-11-12 14:48:03.429849: step 9610, loss = 0.82 (83.0 examples/sec; 1.542 sec/batch)
2017-11-12 14:48:17.703006: step 9620, loss = 0.95 (89.7 examples/sec; 1.427 sec/batch)
2017-11-12 14:48:30.874378: step 9630, loss = 0.77 (97.2 examples/sec; 1.317 sec/batch)
2017-11-12 14:48:45.986130: step 9640, loss = 0.77 (84.7 examples/sec; 1.511 sec/batch)
2017-11-12 14:48:59.676871: step 9650, loss = 0.94 (93.5 examples/sec; 1.369 sec/batch)
2017-11-12 14:49:13.449671: step 9660, loss = 1.17 (92.9 examples/sec; 1.377 sec/batch)
2017-11-12 14:49:26.470936: step 9670, loss = 0.92 (98.3 examples/sec; 1.302 sec/batch)
2017-11-12 14:49:40.261748: step 9680, loss = 0.89 (92.8 examples/sec; 1.379 sec/batch)
2017-11-12 14:49:53.136910: step 9690, loss = 0.80 (99.4 examples/sec; 1.288 sec/batch)
2017-11-12 14:50:06.954742: step 9700, loss = 0.84 (92.6 examples/sec; 1.382 sec/batch)
2017-11-12 14:50:19.847916: step 9710, loss = 1.01 (99.3 examples/sec; 1.289 sec/batch)
2017-11-12 14:50:32.865179: step 9720, loss = 0.99 (98.3 examples/sec; 1.302 sec/batch)
2017-11-12 14:50:46.547914: step 9730, loss = 0.90 (93.5 examples/sec; 1.368 sec/batch)
2017-11-12 14:51:00.040514: step 9740, loss = 0.99 (94.9 examples/sec; 1.349 sec/batch)
2017-11-12 14:51:13.889369: step 9750, loss = 0.94 (92.4 examples/sec; 1.385 sec/batch)
2017-11-12 14:51:27.446015: step 9760, loss = 1.01 (94.4 examples/sec; 1.356 sec/batch)
2017-11-12 14:51:41.023678: step 9770, loss = 0.99 (94.3 examples/sec; 1.358 sec/batch)
2017-11-12 14:51:54.455232: step 9780, loss = 0.95 (95.3 examples/sec; 1.343 sec/batch)
2017-11-12 14:52:08.038901: step 9790, loss = 0.77 (94.2 examples/sec; 1.358 sec/batch)
2017-11-12 14:52:23.908795: step 9800, loss = 0.99 (80.7 examples/sec; 1.587 sec/batch)
2017-11-12 14:52:37.311332: step 9810, loss = 0.93 (95.5 examples/sec; 1.340 sec/batch)
2017-11-12 14:52:50.456685: step 9820, loss = 0.87 (97.4 examples/sec; 1.315 sec/batch)
2017-11-12 14:53:03.530988: step 9830, loss = 1.04 (97.9 examples/sec; 1.307 sec/batch)
2017-11-12 14:53:16.878486: step 9840, loss = 0.85 (95.9 examples/sec; 1.335 sec/batch)
2017-11-12 14:53:30.823409: step 9850, loss = 0.91 (91.8 examples/sec; 1.394 sec/batch)
2017-11-12 14:53:43.964759: step 9860, loss = 1.03 (97.4 examples/sec; 1.314 sec/batch)
2017-11-12 14:53:57.134129: step 9870, loss = 0.90 (97.2 examples/sec; 1.317 sec/batch)
2017-11-12 14:54:10.830874: step 9880, loss = 0.96 (93.5 examples/sec; 1.370 sec/batch)
2017-11-12 14:54:24.246423: step 9890, loss = 0.89 (95.4 examples/sec; 1.342 sec/batch)
2017-11-12 14:54:38.103281: step 9900, loss = 0.93 (92.4 examples/sec; 1.386 sec/batch)
2017-11-12 14:54:51.721970: step 9910, loss = 0.89 (94.0 examples/sec; 1.362 sec/batch)
2017-11-12 14:55:05.238588: step 9920, loss = 0.86 (94.7 examples/sec; 1.352 sec/batch)
2017-11-12 14:55:18.747200: step 9930, loss = 0.80 (94.8 examples/sec; 1.351 sec/batch)
2017-11-12 14:55:32.675111: step 9940, loss = 0.94 (91.9 examples/sec; 1.393 sec/batch)
2017-11-12 14:55:46.876214: step 9950, loss = 0.86 (90.1 examples/sec; 1.420 sec/batch)
2017-11-12 14:56:00.506913: step 9960, loss = 1.12 (93.9 examples/sec; 1.363 sec/batch)
2017-11-12 14:56:13.894439: step 9970, loss = 0.85 (95.6 examples/sec; 1.339 sec/batch)
2017-11-12 14:56:27.462885: step 9980, loss = 1.11 (94.3 examples/sec; 1.357 sec/batch)
2017-11-12 14:56:40.964492: step 9990, loss = 1.03 (94.8 examples/sec; 1.350 sec/batch)
```

Se probó la red neuronal con 10000 imágenes y se obtuvo una precisión promedio de 81%.

```
C:\Users\Stephannie\Documents\GitHub\models\tutorials\image\cifar10>py cifar10_eval.py
2017-11-12 15:17:59.356712: I C:\tf_jenkins\home\workspace\rel-win\M\windows\PY\36\tenso
t this TensorFlow binary was not compiled to use: AVX AVX2
2017-11-12 15:18:24.259289: precision @ 1 = 0.818
```

Graficos