



---

ESCUELA DE COMPUTACIÓN

INTELIGENCIA ARTIFICIAL

## **PROYECTO DE INVESTIGACIÓN**

REALIZADO POR:

SOLÍS BARRANTES, ALEX ANDRÉ

PROFESOR:

HERNÁNDEZ VEGA, LUIS CARLOS

II SEMESTRE

11 DE NOVIEMBRE DEL 2017

## *Tabla de contenido*

I. Image Recognition .....	3
Conclusiones .....	3
Cómo podría aplicarlo a un problema real .....	3
Screenshots de resultados .....	3
II. Image Retraining .....	4
Conclusiones .....	4
Cómo podría aplicarlo a un problema real .....	4
Screenshots de resultados .....	4
III. A Guide to TF Layers: Building a Convolutional Neural.....	5
Conclusiones .....	5
Cómo podría aplicarlo a un problema real .....	5
Screenshots de resultados .....	6
IV. Convolutional Neural Networks .....	6
Conclusiones .....	6
Cómo podría aplicarlo a un problema real .....	7
Screenshots de resultados .....	7
V. Vector Representations of Words.....	8
Conclusiones .....	8
Cómo podría aplicarlo a un problema real .....	8
Screenshots de resultados .....	8
VI. Conclusión General de los Tutoriales.....	9

# **I. Image Recognition**

## **Conclusiones**

El tutorial de Image Recognition fue rápido y sencillo que explica cómo funciona el reconocimiento de imágenes. Este reconocimiento lo hace mediante la herramienta Tensorflow, la cual se encarga de agrupar librerías que son útiles para el desarrollo de proyectos de inteligencia artificial.

Se explicó cómo el programa Image Recognition iba aprendiendo con las imágenes, de tal forma que entre mayor fuera la cantidad de imágenes que aprendiera mejor va a ser la clasificación de las imágenes.

## **Cómo podría aplicarlo a un problema real**

Se puede utilizar en un sistema de reconocimiento de caligrafía y así reconocer qué persona escribió cierto documento. Esto porque todas las personas tienen ciertos patrones de escritura entonces esto ayudaría a la confiabilidad de las firmas en algún ente financiero o judicial.

## **Screenshots de resultados**

```
$ python classify_image.py
>> Downloading inception-2015-12-05.tgz 100.0%
Successfully downloaded inception-2015-12-05.tgz 88931400 bytes.
giant panda, panda, panda bear, coon bear, Ailuropoda melanoleuca (score = 0.89632)
indri, indris, Indri indri, Indri brevicaudatus (score = 0.00766)
lesser panda, red panda, panda, bear cat, cat bear, Ailurus fulgens (score = 0.00266)
custard apple (score = 0.00138)
earthstar (score = 0.00104)
2017-11-03 00:51:21.337076: I c:\tf_jenkins\home\workspace\rel-win\M\windows\PY\35\tensorflow\core\platform\cpu_feature_guard.cc:137] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX AVX2
2017-11-03 00:51:22.046277: W c:\tf_jenkins\home\workspace\rel-win\M\windows\PY\35\tensorflow\core\framework\op_def_util.cc:334] Op BatchNormWithGlobalNormalization is deprecated. It will cease to work in GraphDef version 9. Use tf.nn.batch_normalization().
```

## II. Image Retraining

### Conclusiones

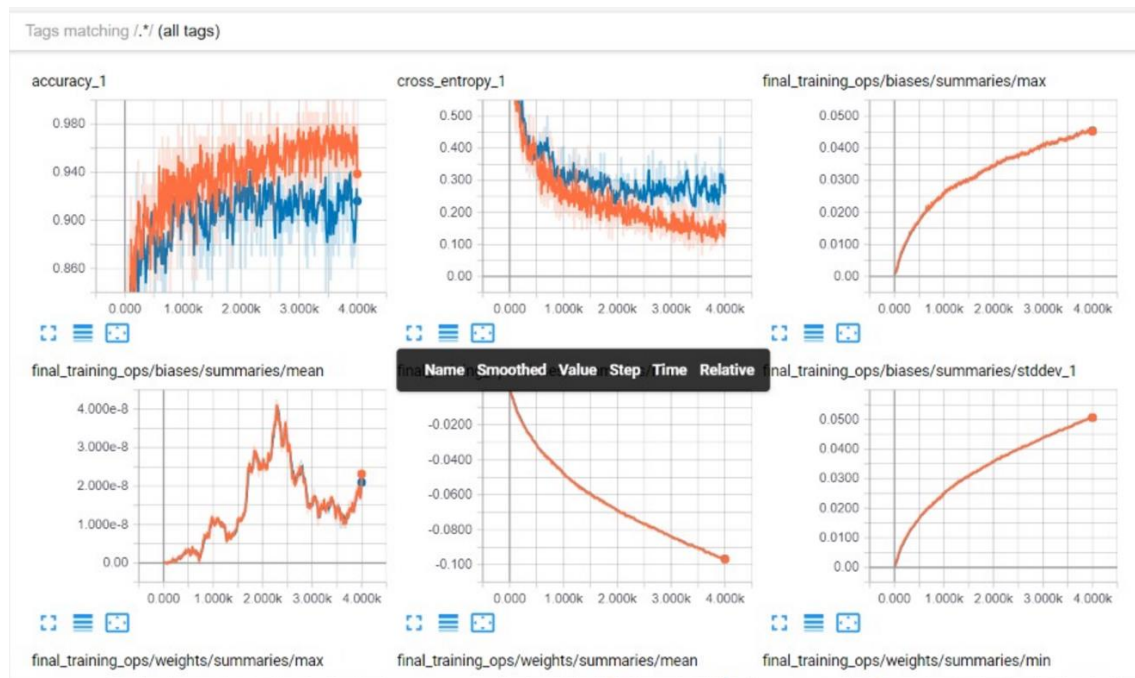
El tutorial de Image Retraining muestra lo sencillo que es cambiar los datos o agregar nuevos para el aprendizaje del programa el cual usa bases de la Inteligencia Artificial, y lo desglosa en gráficos entendibles para alguien que no maneje el concepto de Inteligencia Artificial. También se muestra como agregar una nueva categoría a estudiar por dicho programa.

### Cómo podría aplicarlo a un problema real

Se puede aplicar a un sistema de reconocimiento de pinturas, el cual necesita comprobar la originalidad de cada pintura con su respectivo autor, y de esta manera evitar cualquier tipo de fraude, además si se crea una nueva pintura para un nuevo pintor se podría agregar fácilmente al reconecedor.

### Screenshots de resultados

```
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\141479422_sasfa1fd1b_m.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\14171673834_1208e19be3_m.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\142218310_d06005030a_n.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\142235017_07816937c6.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\142235237_da662d925c.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\142235914_5419ff8a4a.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\14235021006_dd001ea8ed_n.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\14254839301_ffb10c445_n.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\14255917256_84c23c572b.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\14262354955_cc2ab3b112_m.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\14266093711_66d18a1e44_n.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\14270573963_f122c40438.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\14275234071_6e6f473356.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\14278331403_4c475f9a9b.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\14487705209_ea723109e1_m.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\14487762578_baba13d16a_m.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\14487943607_651e8062a1_m.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\14491997336_36ba524713.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\14651383746_419dc73634_m.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\14651383476_7ccbc0e394_m.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\14671196461_b725727229_m.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\14674071872_2df55466d5_m.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\14674388855_2da18e375a_m.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\14674389605_df3c0bcfa1_m.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\1474818178_40403cc57e.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\14836109101_1d07520932_m.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\14861513337_4ef0bfa40d.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\14866400927_3a59899df3_m.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\14957470_6a8c272a87_m.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\15029962436_3e50c1f30f_n.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\15069902081_dd85361f8c_m.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\15052586692_56a82de133_m.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\15082212714_ff87e8fcb1_m.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\15090146325_b7e1249e60.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\15147473067_7c5498eb0e_m.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\15275190769_0ed7bbf490.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\15275199229_9e2387f24d.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\15452908878_0c4941f729_m.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\15458787091_3edc6cd1eb.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\155097272_70feb13184.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\15516736553_b169b67195_n.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\15632065904_0d9caf174b.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\15861328302_f08007a01_n.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\15922772266_1167a06620.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\15976769174_1d50f46ca1_m.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\16055807744_000b07afc_m.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\16062072523_1be3c0b61f.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\16074109313_2cc14c7d16.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\16098264209_38fe491093.jpg_inception_v3.txt
INFO:tensorflow:200 bottleneck files created.
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\16110795216_b3e44697b4_m.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\16138212287_643bf336e1_m.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\16139439153_fbdee29a10_n.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\16169741783_deeab1a679_m.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\16265876844_0a149cd476.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\16265883604_92ba82b973.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\16282727784_b92776b194.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\tulips\16283125269_4cfae953f1.jpg_inception_v3.txt
```



### III. A Guide to TF Layers: Building a Convolutional Neural

#### Conclusiones

El tutorial consistía en ejecutar un programa que está basado en tres capas, las cuales cada una tienen una función en específico:

- Convolutional: esta se encarga de filtrar las imágenes.
- Pooling: toma las imágenes y las disminuye a una menor cantidad de píxeles.
- Dense: es la capa encargada de realizar la clasificación de cada imagen.

Además, esta enseña a utilizar múltiples tarjetas gráficas para su funcionamiento con la pequeña limitante que no muestra cómo realizar si no se cuenta con más de una de estas. La problemática que esta presenta es que sobrecarga el procesador y esto evita que se puedan realizar otras tareas en el computador.

#### Cómo podría aplicarlo a un problema real

Podría ser utilizado en un programa para el análisis en placas de carros el cual requiera del almacenamiento de imágenes en múltiples categorías. Al utilizarlo se estaría optimizando el procesamiento.

## Screenshots de resultados

```
0.00417924 0.00071759 0.87615943 0.00796177]
0.00015174 0.9751634 0.00270424 0.00575363 0.00014304 0.00036732
0.00065469 0.00888754 0.00398971 0.00218452]
0.00038676 0.00000058 0.00005011 0.00001058 0.81317627 0.01881824
0.00198114 0.00142921 0.00891789 0.15522909]
0.000272 0.98923165 0.00193554 0.00290415 0.00042499 0.00031406
0.00055097 0.00058863 0.00374029 0.0000377 ]
0.00067168 0.00009268 0.96026301 0.02540899 0.00000034 0.00019784
0.0002906 0.00000041 0.01307065 0.00000382]
0.70861137 0.00014516 0.00825159 0.00081465 0.00112486 0.26200226
0.01504215 0.00020468 0.00354322 0.00026019]
0.00000329 0.00000022 0.00019194 0.00187271 0.00304772 0.00001396
0.00000414 0.36244705 0.00196259 0.63045639]
0.00163791 0.03599333 0.0049892 0.10855009 0.01469025 0.07910628
0.57450026 0.00031577 0.17511517 0.0051018 ]
0.00195329 0.00160929 0.00193686 0.00744415 0.02823939 0.00459713
0.00018932 0.14023326 0.01558721 0.79821008]
0.00531932 0.00000086 0.00031876 0.97812897 0.00000021 0.01283016
0.00000927 0.00000245 0.00338307 0.00000686]
0.00001729 0.0002956 0.00248779 0.98106259 0.00000137 0.00042973
0.0000002 0.01554972 0.00012685 0.00002889]
0.00000802 0.00000172 0.00108898 0.00036265 0.00014055 0.000289
0.00027069 0.00000376 0.99696702 0.00086754]] (6.617 sec)
```

```
Successfully downloaded train-images-idx3-ubyte.gz 9912422 bytes.
Extracting MNIST-data\train-images-idx3-ubyte.gz
Successfully downloaded train-labels-idx1-ubyte.gz 28881 bytes.
Extracting MNIST-data\train-labels-idx1-ubyte.gz
Successfully downloaded t10k-images-idx3-ubyte.gz 1648877 bytes.
Extracting MNIST-data\t10k-images-idx3-ubyte.gz
Successfully downloaded t10k-labels-idx1-ubyte.gz 4542 bytes.
Extracting MNIST-data\t10k-labels-idx1-ubyte.gz
{'loss': 0.095783487, 'global_step': 20000, 'accuracy': 0.9727}
2017-11-10 22:53:35.230963: I C:\tf_jenkins\home\workspace\rel-win\M\windows\Python35\tensorflow\core\platform\cpu_feature_guard.cc:137] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX AVX2
```

## IV. Convolutional Neural Networks

### Conclusiones

El tutorial consistía en descargar imágenes por categoría, luego este mismo se encargaba de filtrar las dichas imágenes de una manera óptima. Seguidamente se entrenaba a sí misma y el resultado terminaba en `/tmp/cifar10_train`.

Por otro lado, fue necesario modificar el código debido a que este pedía un millón de repeticiones, cuando se notó que iba por 150 000 repeticiones y habían pasado cuatro horas se tomó la decisión de cancelar el tutorial y modificar estas repeticiones a 10 000. Al hacer esto, el tiempo de espera se redujo considerablemente.



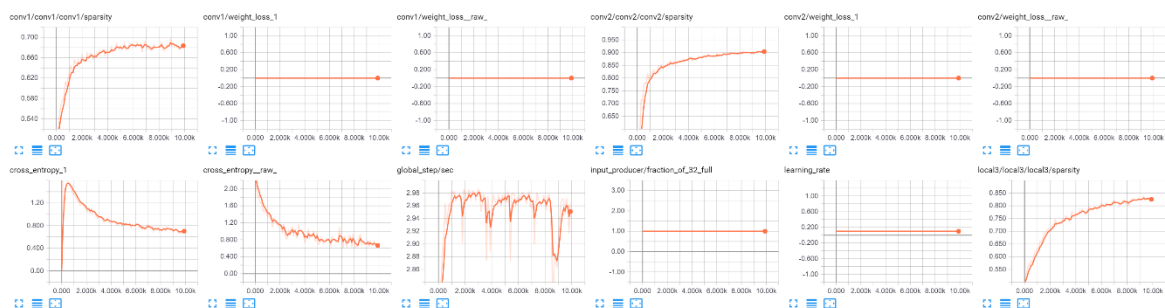
Se observó que utilizó el 100% del CPU en lugar de mandar ese procesamiento al GPU lo que no permitía que la máquina pudiera realizar otra tarea además del tutorial.

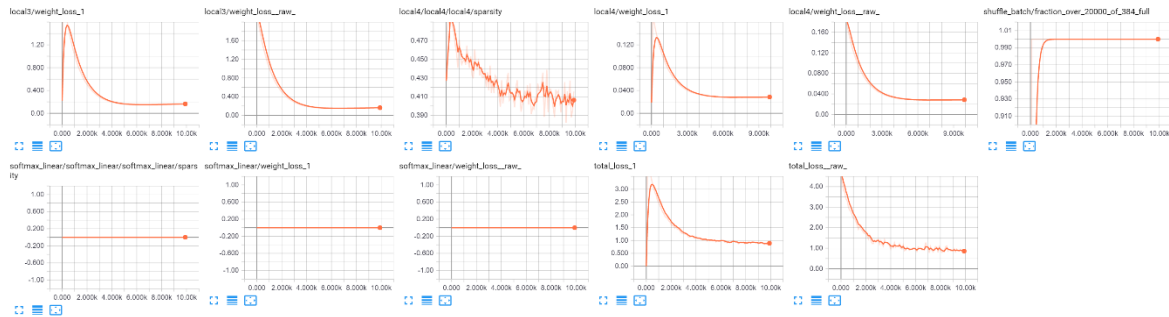
## Cómo podría aplicarlo a un problema real

Se podría generar un software que ayude al reconocimiento facial de rápido procesamiento, como en una cárcel, la cual ocupa saber en qué celda va cada reo. Además de esto se podría calcular cuál es la cantidad exacta de reos y que no se den fraudes.

## Screenshots de resultados

```
c/batch)
2017-11-11 03:14:12.743492: step 9920, loss = 0.78 (385.2 examples/sec; 0.332 se
c/batch)
2017-11-11 03:14:16.081564: step 9930, loss = 0.92 (383.5 examples/sec; 0.334 se
c/batch)
2017-11-11 03:14:19.414474: step 9940, loss = 0.93 (384.0 examples/sec; 0.333 se
c/batch)
2017-11-11 03:14:22.744997: step 9950, loss = 1.06 (384.3 examples/sec; 0.333 se
c/batch)
2017-11-11 03:14:26.092473: step 9960, loss = 0.92 (382.4 examples/sec; 0.335 se
c/batch)
2017-11-11 03:14:29.411810: step 9970, loss = 1.01 (385.6 examples/sec; 0.332 se
c/batch)
2017-11-11 03:14:32.762407: step 9980, loss = 0.89 (382.0 examples/sec; 0.335 se
c/batch)
2017-11-11 03:14:36.078499: step 9990, loss = 0.88 (386.0 examples/sec; 0.332 se
c/batch)
2017-11-11 02:18:47.921257: I C:\tf_jenkins\home\workspace\rel-win\M\windows\PY\
35\tensorflow\core\platform\cpu_feature_guard.cc:137] Your CPU supports instruct
ions that this TensorFlow binary was not compiled to use: AVX AVX2
```





## V. Vector Representations of Words

### Conclusiones

Este tutorial representa las palabras con vectores y además utiliza como herramienta de análisis una ventana. Esta ventana es como una variable, de forma que va tomando la cantidad de palabras que tenga tanto a la derecha e izquierda de la palabra que se le fue dada como target.

Utiliza el método de incrustación, el cual se encarga de introducir palabras por porcentaje de probabilidad y así estimar el parentesco entre palabras

### Cómo podría aplicarlo a un problema real

Podría ser utilizado en un software de biblioteca en el cual reconozca quien es el escritor de cada uno de los libros mediante el análisis de su prosa.

### Screenshots de resultados



```


Nearest to to: thaler, would, can, nine, should, pulau, kapoor, dasyprocta,
Average loss at step 92000 : 4.65530025625
Average loss at step 94000 : 4.71656356466
Average loss at step 96000 : 4.68934567225
Average loss at step 98000 : 4.59293352735
Average loss at step 100000 : 4.69437941957
Nearest to was: is, had, has, became, were, been, busan, kapoor,
Nearest to it: he, this, she, there, which, they, ursus, thaler,
Nearest to after: before, during, when, while, in, following, for, clifford,
Nearest to four: five, three, six, seven, eight, two, zero, ursus,
Nearest to war: agouti, speciality, pat, puritan, shatila, continual, kapoor, ve
rtically,
Nearest to can: may, would, could, should, will, might, must, cannot,
Nearest to state: pontificia, kapoor, constituci, zimri, sakhalin, dasyprocta, m
atrices, trade,
Nearest to than: or, khoisan, cranmer, janitor, but, no, genus, and,
Nearest to one: six, four, seven, eight, two, five, three, kapoor,
Nearest to system: pontificia, abet, systems, tejada, tempe, coppola, thibetanus
, encyclopedia,
Nearest to use: dasyprocta, kapoor, operatorname, most, upanija, akita, ursus, a
gouti,
Nearest to they: there, he, we, it, you, these, not, pulau,
Nearest to be: been, have, being, by, were, is, was, become,
Nearest to d: b, globemaster, f, operatorname, thibetanus, bryozoans, six, pulau
,
Nearest to if: when, where, since, nvidia, thumb, is, although, operatorname,
Nearest to to: thaler, would, kapoor, dasyprocta, pulau, can, should, circ,
Please install sklearn, matplotlib, and scipy to show embeddings.
No module named 'sklearn'
2017-11-11 20:32:05.617774: I C:\tf_jenkins\home\workspace\rel-win\M\windows\PY\
35\tensorflow\core\platform\cpu_feature_guard.cc:137] Your CPU supports instruct
ions that this TensorFlow binary was not compiled to use: AVX AVX2

```

## VI. Conclusión General de los Tutoriales

Los tutoriales son complejos en la parte de código, no hay explicaciones respecto al funcionamiento de cada parte, pero son muy sencillos de implementar porque solo dicen qué hay que ejecutar.

Para mejorar los tutoriales, lo ideal sería que el procesamiento lo realizara en el GPU y no en el CPU. Como en la imagen se muestra, el CPU estaba a su máximo y la temperatura rondaba los 71°C, cosa que si se hubiera utilizado en el GPU habría durado la mitad del tiempo y la temperatura no sería tan alta.

Nombre	100% CPU
 Python	40,2%
 Python	53,0%