

TUST: Twitter User Search Tool

Instrucciones de instalación

18.03.2016

Desarrolladores:

Daniel Garnacho y Diego Castaño
Universidad Autónoma de Madrid

Resumen

TUST (Twitter User Search Tool) es una aplicación web que permite buscar usuarios de la red social Twitter que sean similares a uno dado o que estén relacionados con un cierto texto. Esta herramienta emplea algoritmos de procesamiento del lenguaje natural y big data para analizar ingentes cantidades de tweets a gran velocidad.

Objetivos

Este documento tiene como objetivo detallar los pasos para instalar el software en otro servidor o servidores.

Pre - requisitos

Hardware utilizado para desarrollo:

- 32GB de memoria RAM
- Un procesador i7-4790K
- Un disco duro para sistema, Western Digital Black de 2 TB
- Un disco duro para copias de seguridad

Hardware mínimo recomendado, una sola máquina:

- 64GB de memoria RAM
- Un procesador de 4 a 8 núcleos - 8 a 16 hilos
- Sistema de discos RAID - 1 o 5

Hardware optimo:

- 3 máquinas para almacenamiento de bases de datos distribuidas, entre 16 y 32 GB de RAM. (Ampliable según la utilización)
- 1 máquina para almacenamiento de base de datos relacional. (Aplicación)
- 1 máquina para procesamiento de lenguaje natural 128 - 256GB de RAM y 2 procesadores.

Sistema operativo:

- Utilizado para desarrollo Ubuntu Server 14.04 LTS
- Se recomienda utilizar Ubuntu Server o sistema Unix basado en Debian.

Instalación de Cassandra DB

El nodo que vaya a tener Cassandra DB debe tener instalado Java, se recomienda por los desarrolladores de Cassandra utilizar Oracle-Java 8.

```
$ sudo add-apt-repository ppa:webupd8team/java  
$ sudo apt-get update  
$ sudo apt-get install oracle-java8-installer  
$ sudo apt-get install oracle-java8-set-default
```

Al ejecutar:

```
java -version
```

La salida debería contener algo parecido a:

```
java version "1.8.0_XX"
```

```
Java(TM) SE Runtime Environment (build 1.8.0_XXXXX)
```

```
Java HotSpot(TM) 64-Bit Server VM (build 25.72-b15, mixed mode)
```

Cassandra 2.1

Para el desarrollo se ha utilizado Cassandra 2.1, no se ha probado otra distribución pero no debería haber problemas con la última versión.

```
$ echo "deb http://debian.datastax.com/community stable main" | sudo tee -a  
/etc/apt/sources.list.d/cassandra.sources.list  
$ curl -L https://debian.datastax.com/debian/repo_key | sudo apt-key add -  
$ sudo apt-get update  
$ sudo apt-get install cassandra=2.1.x
```

Donde las x representan la versión de cassandra 2.1, la última a fecha de creación de este documento es 2.1.13

Dado que los sistemas Debian arrancan Cassandra automáticamente se deberán ejecutar las siguientes líneas, posteriormente reiniciar el sistema.

```
$ sudo service cassandra stop  
$ sudo rm -rf /var/lib/cassandra/data/system/*
```

Lucene Index Cassandra 2.1.XX

Lucene permite realizar un índice de búsqueda para poder encontrar tweets como si de un buscador como google se tratara. La herramienta web que se proporciona no utiliza esta funcionalidad pero se recomienda instalarlo para realizar búsquedas más complejas en la base de datos por tweets y no por usuarios.

<https://github.com/Stratio/cassandra-lucene-index>

Si no se instala se deberá modificar el fichero DBbridge/Cassandra/CreaTablas.py

Creación del Keyspace

El keyspace es similar esquema de las bases de datos relacionales. Al definirlo creamos una forma de acceso unificada un servidor o servidores.

```
$ cd /usr/bin/  
$ ./cqlsh
```

Una vez dentro de la consola de Cassandra ejecutamos

```
CREATE KEYSPACE twitter WITH REPLICATION = { 'class' : 'SimpleStrategy',  
'replication_factor' : X};
```

Donde X es la cantidad de veces que queremos replicar la información, por ejemplo si se estropea un nodo Cassandra y no queremos perder los datos.

Cuidado, es posible que se deba dar acceso a otros hosts para acceder a la máquina.

Instalación de Neo4J

Neo4J es una base de datos distribuida para almacenar grafos y conexiones, en este proyecto se utiliza para almacenar las relaciones dentro de la red social, seguimientos. Neo4J al igual que Cassandra utilizará Oracle Java 8

Instalamos el repositorio:

```
wget -O - https://debian.neo4j.org/neotechnology.gpg.key | sudo apt-key add -  
echo 'deb http://debian.neo4j.org/repo stable/' >/tmp/neo4j.list  
sudo mv /tmp/neo4j.list /etc/apt/sources.list.d  
sudo apt-get update
```

Instalamos Neo4J

```
sudo apt-get install neo4j
```

La primera vez que nos conectemos a la interfaz web de Neo4J nos pedirá cambiar la contraseña de acceso "localhost:7474" si no podemos acceder al servidor por localhost se deberá configurar que el servidor escuche a otros Hosts.

<http://neo4j.com/docs/stable/security-server.html>

Instalación PostgreSQL

PostgreSQL se utiliza en la interfaz web y para mantener la coherencia del uso de consultas en la API de Twitter. Podría utilizarse para guardar datos de usuarios y tweets, se ha demostrado que es más lento.

Instalación:

```
sudo apt-get update  
sudo apt-get install postgresql postgresql-contrib
```

Configuración del usuario de la base de datos:

```
sudo -i -u postgres  
createuser --interactive (Necesita permisos de creación y eliminación de tablas)
```

Desde dentro del usuario postgres creamos las bases de datos, necesitamos dos de ellas:

```
createdb datos  
createdb aplicacion
```

Ficheros de Configuración

Hay dos ficheros de configuración en la aplicación, uno controla el modelo y otro defecto creado por Django (Servidor python), se ha configurado este segundo para que utilice el primer fichero.

El fichero relevante se encuentra en la carpeta Config. Se proporciona un ejemplo de configuración en **Conf_default.py**, se debe copiar y **renombrar por Conf.py**

A continuación se detallan las variables de la clase que hay que modificar.

1. Domain: Cadena de texto que contiene el dominio del proyecto, por ejemplo "www.ejemplo.com" se utilizará para generar la url de los correos.
2. Cassandra_keyspace: Contiene el nombre del Key Space, en este documento lo llamamos twitter.
3. Sql_database: Contiene el nombre de la base de datos sql, que contiene los datos de la aplicación y la coherencia de las consultas a la API de twitter.
4. Sql_database_policia: Contiene el nombre de la base de dato SQL que contiene los datos del servidor web, usuarios de la aplicación y tareas creadas.
5. Sql_user, sql_password, sql_host, sql_port: parámetros de configuración de la base de datos.
6. Abspath: es una cadena de texto que especifica dónde se almacenarán los datos de las tareas luigi, generación de modelos de entrenamiento y similitudes de usuarios.
7. Neo4j_password: contraseña de Neo4j
8. dimVectors: número entero que define cuál es la dimensión de los vectores de entrenamiento, cuanto mayor sea este número mayor precisión pero a la vez mayor consumo de memoria.

Instalar las librerías

En esta sección se detalla la instalación tanto de librerías externas como de los módulos del proyecto.

Se recomienda la utilización de virtual environment, sobre todo en entornos de desarrollo o de aplicaciones compartidas.

Librerías externas

Estando en la carpeta raíz del proyecto. Si no se usa venv, se deberá usar sudo

```
pip install -r requirements.txt
```

Módulo internos

```
sudo python setup.py install
```


Creación de tablas y relaciones

Se deben crear las tablas, índices y relaciones. Cada base de datos utiliza un fichero.

```
python DBbridge/Cassandra/Creatablas.py
python DBbridge/Neo4j/CreaRelaciones.py
python DBbridge/PostgreSQL/CreaTablas.py
```

Inserción API Key Twitter

Al menos debe existir una API Key de twitter para hacer las consultas. Para insertarla en la base de dato se puede usar DBbridge/PostgreSQL/NewApiKey.py

Se deben rellenar en el main las variables apik, apiks, acstoken, acstokens, todas estas se pueden consultar en la web de la API de Twitter.

Creación de tablas Django

Django se encarga de crear los modelos que se traducen en tablas de una base de datos a grandes rasgos, para crearlos debemos:

```
cd website
python manage.py makemigrations
python manage.py migrate
```

Apache y Django

Apache gestionará múltiples instancias Django, se debe instalar en el servidor:

```
sudo apt-get install apache2
```

En `/etc/apache2/sites-enabled`, debemos o crear un fichero o modificarlo para tener una configuración similar a la siguiente donde (path del proyecto) es la localización global del proyecto.

```
ServerAdmin webmaster@localhost


ErrorLog ${APACHE_LOG_DIR}/error.log
CustomLog ${APACHE_LOG_DIR}/access.log combined

Alias /static/ (path del proyecto)/website/policia/static/

<Directory (path del proyecto)/website/policia/static/>
    Require all granted
</Directory>

WSGIApplicationGroup %{GLOBAL}
WSGIScriptAlias / (path del proyecto)/website/website/wsgi.py
WSGIProxyPath (path del proyecto)/website

<Directory (path del proyecto)/website/website>
    <Files wsgi.py>
        Require all granted
    </Files>
</Directory>
```



Dentro de la carpeta del proyecto debemos dar la propiedad de la carpeta LuigiTasks a apache que es donde se almacenarán los ficheros de las tareas Luigi.

```
chown -R www-data:www-data LuigiTasks/
```