Crystalace

User Manual

User Manual Versions	Date	Description
2.1	**22 August 2019**	**Updated with Docker Image**
2.0	22 July 2019	Python SDK for Linux and Windows (Full SDK)
1.0	16 August 2018	Python SDK (Java Wrapper)

Table of Content

Section 1 Overview

Welcome to the Crystalace User Manual.

Detecting sarcasm is among one of the toughest natural language understanding problems in AI. In computational linguistics and NLP, sarcasm detection is receiving increasing research interest. While recent studies recognized the linkage between sarcasm and sentiment and have proposed various techniques for detecting sarcasm, none directly and systematically studied the impact of sarcasm detection on sentiment analysis.

Crystalace is a sarcasm detection method that is able to analyze the underlying meaning of a message and automatically determine if the message is meant to be sarcastic or not sarcastic. The output is a score indicating the degree of confidence of the sarcasm detection in the range of [0, 1]. The score has significant implication in sentiment analysis and opinion mining, and can be used to, for example, screen verbatim with sarcasm scores and improve the accuracy of sentiment analysis expressed from the text messages.

Section 2 Supported Platforms

The system employs both *database* mode (MySQL/PostgreSQL) and *server-client* mode.

The currently supported platforms include:

i) Server-side:
   - 64 bit Linux distributions (Ubuntu 16.xx or Ubuntu 18.xx)
   - Python 3.5 or above
   - MySQL or PostgreSQL (not required for server-client mode)

ii) Client-side:
   - Any programming language/platform that can communicate on a socket (for server-client architecture) or remote database

Section 3 SDK Installation

To run the Crystalace SDK, you need to install some base python packages. For database mode, you need to then create a database. This section describes the details of these python packages and database structure.

3.1 Required Base Packages

Please install the following Python packages:

```
$  pip3 install scikit-learn==0.20.3
$  pip3 install nltk==3.4.1
$  pip3 install bert-serving-server==1.9.3
$  pip3 install bert-serving-client==1.9.3
$  pip3 install setuptools==41.0.1
$  pip3  install  psycopg2      #  Required  only  if  you  will  use  SDK  with
   PostgreSQL database, not required in case of server-client mode
$  sudo apt-get install python3-mysqldb   # Required only if you will use
   SDK with MySQL database, not required in case of server-client mode
```

After installing the above python packages, download some nltk modules using these commands:

```
$  python3 # Login to python shell and run the following commands
      »  import nltk
      »  nltk.download('punkt')
      »  nltk.download('wordnet')
      »  nltk.download('averaged_perceptron_tagger')
      »  exit(0)  # Exit python shell
```

3.2 Database Structure

This step is not required if you are running the SDK in server-client mode. You will need to create a table 'sarcasm_data' with the following structure:

- For **MySQL database**:

```
»  CREATE DATABASE <db_name>;
»  ALTER DATABASE <db_name> CHARACTER SET = utf8mb4 COLLATE
   utf8mb4_unicode_ci;
»  USE <db_name>;
»  CREATE TABLE sarcasm_data(id INT NOT NULL AUTO_INCREMENT, tweet
   TEXT, timestamp INT, results varchar(64) DEFAULT '-', is_processed
   TINYINT(1) DEFAULT 0, primary key(id));
»  ALTER TABLE sarcasm_data CONVERT TO CHARACTER SET utf8mb4 COLLATE
   utf8mb4_unicode_ci;
```

- For **PostgreSQL database**:

```
»  CREATE DATABASE <db_name>;
»  \c <db_name>;  # connect to database
»  CREATE TABLE sarcasm_data(id serial primary key, tweet TEXT,
   timestamp INT, results varchar(64) DEFAULT '-', is_processed
   smallint DEFAULT 0);
```

3.3 Configure & Run SDK

The system configuration file *'system.cfg'* can be found in the root directory of the SDK. The file has the following structure:

```
[Network]
PORT_NUMBER 8248

#[MySql]
#SERVER_ADDRESS <db_server_address>
#DB_NAME <db_name>
#DB_USERNAME <db_user_name>
#DB_PASSWORD <db_user_password>

#[PostgreSQL]
#SERVER_ADDRESS <db_server_address>
#DB_NAME <db_name>
#DB_USERNAME <db_user_name>
#DB_PASSWORD <db_user_password>
```

The configuration file contains the details of three configurations:

- Network (for server-client mode),
- MySQL (to use SDK with MySQL database)
- PostgreSQL (to use SDK with PostgreSQL database)

Only one configuration can be active at a time (Network mode is active in the above configuration). To activate any configuration, please remove '#' from the beginning of that configuration and add '#' at the beginning to disable any active configuration. To activate MySQL or PostgreSQL configuration, remove the '#' (and add it in currently active configuration) from that configuration and update the corresponding SERVER_ADDRESS, DB_NAME, DB_USERNAME and DB_PASSWORD.

In server-client mode, the server listens at a default port number 8248. You can configure the server to listen another port by updating the variable PORT_NUMBER in the configuration file.

After updating the system configuration file (*system.cfg*), open a terminal and start the BERT server using the following command:

```
$  bert-serving-start -model_dir SDK_PATH/model/uncased_L-24_H-1024_A-
   16/ -num_worker=2 -max_seq_len=100
```

Please note that CrystalFeel and Crystalace use the same BERT model so you don't need to start BERT server again if it is already running on the same machine.

After successfully starting the BERT server, open another terminal window to start the sarcasm detection engine. To start, run the python script *'server.py'* in the bin folder.

```
SDK_PATH/bin$ python3 server.py
```

On successful execution of this python script, it will display *"System Ready!"* message. After receiving this message, the client scripts can be used to analyze the text.

Section 4 Sarcasm Detection

After successfully starting the sarcasm detection engine, you can analyze any text to obtain sarcasm probability score. Based on the configuration in *"system.cfg"* file, the text can be analyzed as follows:

4.1 Server-Client Mode

If the "[Network]" configuration is active in "system.cfg" before starting the sarcasm detection engine, the text can be analyzed by sending it to the port (PORT_NUMBER) mentioned in the configuration file. Here is the python3 script to analyze the text:

```
import socket

PORT_NUMBER = 8248; # Update the port number according to system.cfg
IP_ADDRESS  = 'localhost'; # IP address of machine running CrystalFeel
SDK

data = "Another cheap attempt to in news or he was really drunk when he
was giving that statement"; # Text to be analysed

data = bytes(data, 'utf-8');

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM);
s.connect((IP_ADDRESS, PORT_NUMBER));

s.send(data);

result = '';
out = s.recv(1);
while(out != b'\n'):
      result = result + str(out, 'utf-8');
      out = s.recv(1);

s.close();
print(result.strip());
```

Please update the ***PORT_NUMBER*** (based on the PORT_NUMBER specified in SDK configuration file *'system.cfg'*), ***IP_ADDRESS*** (with the address of the machine running Crystalace SDK) and ***data*** (with the text that to be analyzed) in the above sample code.

4.2 PostgreSQL Mode

If the "[PostgreSQL]" configuration is active in "system.cfg" before starting the sarcasm detection engine, the text can be analyzed using the following python3 script:

```
import time
import psycopg2

dbInfo = {'SERVER_ADDRESS': 'localhost', 'DB_NAME': 'db_name',
'DB_USERNAME': 'db_user_name', 'DB_PASSWORD': 'db_password'};

data = "Another cheap attempt to in news or he was really drunk when he
was giving that statement";

def connect_db(dbInfo):
      try:
            connect_str = "dbname='" + dbInfo['DB_NAME'] + "' user='" +
dbInfo['DB_USERNAME'] + "' host='" + dbInfo['SERVER_ADDRESS'] + "'
password='" + dbInfo['DB_PASSWORD'] + "'";
            db = psycopg2.connect(connect_str);
            db.autocommit = True;
            cur = db.cursor();
            #cur.execute("SET NAMES utf8mb4;");
            return db, cur;
      except:
            print("FAILED to connect database.")
            exit(-1);


db, cur = connect_db(dbInfo);

query = "INSERT INTO sarcasm_data(tweet, timestamp) VALUES('" + data +
"', " + str(int(time.time())) + ") RETURNING id;";
cur.execute(query);
insert_id = cur.fetchone()[0];

query = "SELECT results FROM sarcasm_data WHERE id=" + str(insert_id) +
" AND is_processed=1";
while(1):
      is_processed = 0;
      time.sleep(0.3);
      cur.execute(query);
      for row in cur.fetchall():
            print("Result: " + str(row[0]));
            is_processed = 1;

      if is_processed:
            break;

cur.close();
db.close();
```

Update the ***dbInfo*** (with database name, server address, database user name and password) and ***data*** (with the text to be analyzed) in the above source code.

4.3 MySql Mode

If the “[MySql]” configuration is active in “system.cfg” before starting the sarcasm detection engine, the text can be analyzed using the following python3 script:

```
import time
import MySQLdb

dbInfo = {'SERVER_ADDRESS': 'localhost', 'DB_NAME': 'db_name',
'DB_USERNAME': 'db_user_name', 'DB_PASSWORD': 'db_password'};

data = "Another cheap attempt to in news or he was really drunk when he
was giving that statement";

def connect_db(dbInfo):
    try:
          db = MySQLdb.connect(host=dbInfo['SERVER_ADDRESS'],
user=dbInfo['DB_USERNAME'], passwd=dbInfo['DB_PASSWORD'],
db=dbInfo['DB_NAME']);
          db.autocommit(True);
          cur = db.cursor();
          cur.execute("SET NAMES utf8mb4;");
          return db, cur;
    except:
        print "FAILED to connect database."
        exit(-1);

db, cur = connect_db(dbInfo);

query = "INSERT INTO sarcasm_data(tweet, timestamp) VALUES('" + data +
"', " + str(int(time.time())) + ");";
cur.execute(query);
insert_id = db.insert_id();

query = "SELECT results FROM sarcasm_data WHERE id=" + str(insert_id) +
" AND is_processed=1";
while(1):
      is_processed = 0;
      time.sleep(0.3);
      cur.execute(query);
      for row in cur.fetchall():
            print("Result: " + str(row[0]));
            is_processed = 1;

      if is_processed:
            break;

cur.close();
db.close();
```

Update the ***dbInfo*** (with database name, server address, database user name and password) and ***data*** (with the text to be analyzed) in the above source code.

4.4 Output

For any of the three configurations, the client script will generate the output for the sample text ("Another cheap attempt to in news or he was really drunk when he was giving that statement") in the following format:

```
{
    "sarcasm_score": 0.1016,
    "status": "success"
}
```

Section 5 Docker Image of SDK

The Crystalace SDK is also available in Docker image form. This section describes the details of loading and running the Docker image.

5.1 Loading the Docker Image

Use the following commands to load the docker images (*bertserver* and *crystalace*):

```
$  docker load --input bertserver-x.x.x.tar
$  docker load --input crystalace-x.x.x.tar
```

5.2 Running the Docker Image

After loading the two docker images (*bertserver* and *crystalace*) successfully, the *bertserver* can be started using the following command:

```
$  docker run -p 5555:5555 -p 5556:5556 bertserver <NUM_OF_WORKERS>
```

The default parameter **NUM_OF_WORKERS** value is **1**. If you are also running the *CrystalFeel Emotion Analysis* SDK using the same *bertserver*, please set the value of this parameter as **2**.

After successfully starting the *bertserver* (if you see “all set, ready to serve request!”), the *Crystalace* docker image can be started using the following command:

```
$  docker run -p 8248:8248 -it crystalace <BERT_SERVER_IP_ADDRESS>
```

The parameter **BERT_SERVER_IP_ADDRESS** should contain the IP address of the machine running the *bertserver*. If *bertserver* is running on the same machine, the IP address of the host should be used to connect to the *bertserver*.

After starting the *crystalace* docker image, the client script described in Section 4.1 can be used to process the text messages.

Section 6 Interpretation of Sarcasm Detection Output

Crystalace produces a sarcasm confidence score between the range of 0 and 1, where 0 indicates no likelihood of sarcasm and 1 indicates 100% likelihood of sarcasm. For example,

```
        "sarcasm_score": 0.1016
```

The output indicates the probability of sarcasm. The following table describes the output codebook.

Crystalace Output / Feature	Meaning	Scoring Range
sarcasm_score	Sarcasm is often expressed in a seemingly positive way in the literal sense which involves a negative emotional connotation. The Crystalace sarcasm score indicates the probability or degree of confidence if a given text message is meant to be sarcastic.	[0 - this text is very unlikely meant to be sarcastic; 1 - this text is very likely meant to be sarcastic]

Crystalace sarcasm score can be converted into binary value as follows. Note that 0.5 is a general heuristic, and users may adjust the threshold as they see appropriate for their applications.

[0 - 0.5]	(0.5 - 1]
This text is not meant to be sarcastic	This text is meant to be sarcastic

Broadly speaking, Crystalace sarcasm score has two categories of possible applications:

i. The sarcasm score can be directly applied to pre-screen text messages (e.g., tweets, Facebook comments, reviews), before applying sentiment analysis. Those with high scores are likely to be sarcastic, and sarcasm is mostly implying negative meaning.
ii. The sarcasm scores can also serve as a special feature for training and developing new machine learning methods for advanced applications.

The following provides three examples to illustrate Crystalace's sarcasm detection outputs. To try your own text, visit www.crystalace.socialanalyticsplus.net.

SN	Source	Examples	Crystalace Output
1	Twitter (SemEval18)	I graduated yesterday and already had 8 family members asking what job I've got now 😂 #nightmare	0.2386 (no sarcasm)
2	Facebook	WHEREVER they go and NEVER use disposable takeaway containers and cups anymore? GET REAL.	0.1768 (no sarcasm)
3	Amazon	Great at teaching the youngsters how to behave in a police state. What better way to teach the next generation how to behave in a police state with a tyy such as this? I'm really hoping that they come out with a toy in which the kids can play "interregator". Think of all the fun the little folks can have waterboarding those who "hate our freedom".	0.8634 (sarcasm)