Emoter

emoter.me (http://emoter.me) (The information on this website is not up to date, as Emoter has gone open-source)

try-emote.ml (http://try-emote.ml)

This repository contains the full source code for a sentiment analyzer library called Emote, and its companion program Emoter, a chatbot intergrated with Emote that allows it to empathize with the users talking to it. Emote is based off TextBlob's (NLTK's) naive Bayes probability system, and is able to detect reasonably accurate values for 3-6 different emotional tones, from a total range of 36 classifications. Emoter chatbot agents use Emote to analyze user messages, then choose an emotionally appropriate response from interchangable "conversations", based around designed personalities / personas. This project was (or, is still currently being) developed for my undergraduate thesis at Parsons School of Design.

Note

This is a student project that is still in development. It is also the first library I've developed, and the most ambitious project I've done yet. I don't claim this to be the best or most optimal code, but I think it does work well enough to demonstrate basic proof-of-concept of an important but often neglected idea in AI: higher level thinking. Emote / Emoter are open-sourced under the MIT License. The back end code (outside the libraries and database) totals only around ~1500 lines right now, so many features are lacking still. I've mapped out full design specs in diagrams located in the /docs directory.

Screenshots

Prerequisites

- Windows / Mac (Untested on Linux)
- Python 3.5 or higher
- Pip / virtual environments (Pip not required if installing via setup.py)

Introduction

Emote uses the TextBlob / NLTK, NumPy, SciPy, pandas, and scikit-learn libraries to build a probabilistic sentiment analyzer for 26 different classifications. These classifications have been divided into 13 pairs of opposites, and are designed to be grouped together to create tone clusters that can then encompass more values as well as decrease false positives. Based off these tone clusters, a further 10 additional tone classifications are derived.

Emoter is a basic but functional chatbot platform intergrated with Emote, in order to give chatbot agents the ability to empathize with users and give back emotionally appropriate responses. Emoter agents thus can operate on a "higher level of thinking", by first categorizing messages and then choosing specific, interchangable "conversations" (lists of text responses) to respond from based on certain emotional tones. Within these conversations, Emoter looks for matching text in its database and compares it with the user input on a sliding threshold, outputting the corresponding response if the threshold is met.

Both Emote and Emoter can be run offline.

Full List of Emotional Tone Classifications

```
Positive, negative; love, hate; joy, anger; certainty, confusion; amusement, boredom; intensity, regret; challenging, agreeable; desire, calm
; sarcastic, emphatic; instructive, accusative; admiration, inquisitive; modest, pride; ambivalence, vulgarity
```

```
10 additional tones can be derived from the 26 base tones through combining associations: obedience, assertiveness; attraction, disgust; inf
ormative, malevolent; anxiety, excitement; hopeful, horror.
```

Installing

Once you create your virtual environment, run the setup file or install the dependencies via pip.

All packages should install fine, but for NumPy and SciPy, you will need to manually download the packages with mkl, here:

http://www.lfd.uci.edu/~gohlke/pythonlibs/#numpy (http://www.lfd.uci.edu/~gohlke/pythonlibs/#numpy)

http://www.lfd.uci.edu/~gohlke/pythonlibs/#scipy (http://www.lfd.uci.edu/~gohlke/pythonlibs/#scipy)

In command line or Terminal, cd to the directory where the whl files are, and run this command and install both packages:

```
pip install package_name.whl
```

Then, download the necessary corpus to use TextBlob:

```
python -m textblob.download_corpora
```

Because GitHub has a file upload limitation, and to save space, I've uploaded a pickled version of the probability classifier, so that it doesn't have to rebuild every time Emote is reloaded.

https://www.mediafire.com/?c18ll802ynb7s3c (https://www.mediafire.com/?c18ll802ynb7s3c"Download pickled classifier")

Put the pickle file in the /data directory of Emoter, or build it yourself with your own database by running emote.py.

Using Emote / Emoter

Emote can be run as a script with a CLI, or imported as a module / library. Run.py starts Emote's web app interface, where you can demo the sentiment analysis and mass analyze CSV files. Emoter.py contains the base template for a blank chatbot agent, and currently can only be run as a script. Emoter_trainer_wrapper.py is meant to be used to train and test Emote's database, but is unfinished and not functional.

Emote's (currently) very basic API..

Start by importing emote.

```
from emote import emote
```

Send text message to be analyzed.

```
message = "Duct tape. I need it for... taping something."
result = emote.runAnalysis(message)
```

You will get a list of tuples of all the emotional tone values in descending order as a result:

```
[('desire', 100.0), ('emphatic', 14.7), ('certainty', 9.6), ('challenging', 5.0), ('agreeable', 3.1), ('instructive', 2.4), ('intensity', 2.3
), ('accusative', 2.0), ('anger', 1.4000000000000001), ('inquisitive', 0.7000000000000001), ('confusion', 0.4), ('hate', 0.1), ('negative', 0
.0), ('calm', -0.0), ('regret', -0.0), ('modest', -0.1), ('vulgarity', -0.1), ('pride', -0.1), ('love', -0.1), ('positive', -0.1), ('amuseme
nt', -0.1), ('joy', -0.1), ('admiration', -0.1), ('sarcastic', -0.1), ('boredom', -0.1), ('ambivalence', -0.1)]
```

You can get values from the result like this:

```
result[0][0]
```

returns the strongest tone classification-value pair:

```
('desire', 100.0)
```

and

```
result[0][0][0]
```

returns the strongest tone classification:

```
'desire'
```

while

```
result[0][0][1]
```

returns the strongest tone value:

```
100.0
```

And so

```
result[0][1]
```

returns the 2nd strongest tone classification-value pair:

```
('emphatic', 14.7)
```

and so on for all 26 base tones, in descending order of value.

To use Emote's web interface and mass analyzer feature, start run.py, and go to localhost:5000 in your browser:

Emoter chatbot agents right now only work as scripts through CLI.

Use emoter.py as a template to write in custom conversations / databases (documented within comments in the code).

Run emoter_fitness_coach.py to demo the sample persona 'fitness coach'.

Training / Adding Your Own Database

Follow this quick visual guide to understand how the base corpus has been trained.

Also, see the file 'alice_classification_training_sample.txt' to see passages from Alice in Wonderland classified through with Emote's tones.

Current Limitations

- System for deriving 10 additional tones from base 26 has not yet been implemented
- Database is too young (<8000 classifications) to be consistently accurate
- Emote / Emoter agents are only deployable with Python
- Emoter agents have no ability to remember or learn new things
- Emoter agents only search for one matching database, not multiple. If the threshold fails, then Emoter will just search the entire database.
- No API has been started yet for Emoter agents; you can only chat via command line interface right now
- Conversations / texts are just lists of tuples, which is too limiting of a data structure

Future Plans

- Functionality to automatically load interchangable databases / corpuses (Pickled files) to narrow down contexts in Emote
- Incorporate machine learning and accuracy testing into Emote's database
- Develop automated way of training databases (described in architecture flowchart in /docs)
- Build out a full RESTful API for Emote, and offer plans for developers / businesses for API calls via Emoter website
- Develop a GUI web interface to create / customize Emoter chatbot agents
- Fix database matching to multiple (going in descending order) instead of just one
- Rework conversations / texts data structure into SQL, with added frequency / weight values for more complexity
- Implement short-term memory functionality for Emoter agents
- Improve sequence matching by incorporating automatic addition of synonyms for words and phrases
- Build a Unity SDK for Emote / Emoter
- Incorporation with speech recognition and speech synthesis
- Incorporation with computer vision

Authors

- **Johnny Dunn** - *Initial work* - jddunn (https://github.com/jddunn)

See also the list of contributors (https://github.com/jddunn/emoter/contributors) who participated in this project.

License

This project is licensed under the MIT License - see the LICENSE.md (LICENSE.md) file for details

References

https://cloud.google.com/prediction/docs/sentiment_analysis (https://cloud.google.com/prediction/docs/sentiment_analysis)

http://venturebeat.com/2016/08/21/how-higher-emotional-intelligence-will-help-chatbots/ (http://venturebeat.com/2016/08/21/how-higher-emotional-intelligence-will-help-chatbots/)

https://www.theguardian.com/technology/2016/oct/26/black-mirror-episode-playtest-predicted-future-video-games-augmented-reality/ (https://www.theguardian.com/technology/2016/oct/26/black-mirror-episode-playtest-predicted-future-video-games-augmented-reality/)

https://www.theguardian.com/technology/2016/oct/12/video-game-characters-emotional-ai-developers/ (https://www.theguardian.com/technology/2016/oct/12/video-game-characters-emotional-ai-developers/)

https://www.researchgate.net/publication/306091953_A_Meta-Framework_for_Modeling_the_Human_Reading_Process_in_Sentiment_Analysis/ (https://www.researchgate.net/publication/306091953_A_Meta-Framework_for_Modeling_the_Human_Reading_Process_in_Sentiment_Analysis/)

https://www.researchgate.net/publication/308401450_A_Novel_Approach_to_Big_Data_Veracity_using_Crowdsourcing_Techniques_and_Bayesian_Predictors/ (https://www.researchgate.net/publication/308401450_A_Novel_Approach_to_Big_Data_Veracity_using_Crowdsourcing_Techniques_and_Bayesian_Predictors/)

https://www.researchgate.net/publication/308017468_Domain-specific_sentiment_classification_via_fusing_sentiment_knowledge_from_multiple_sources/ (https://www.researchgate.net/publication/308017468_Domain-specific_sentiment_classification_via_fusing_sentiment_knowledge_from_multiple_sources/)

http://www.julianjaynes.org/evidence_summary.php/ (http://www.julianjaynes.org/evidence_summary.php/)

Credits

- TextBlob
- NLTK
- NumPy / SciPy
- scikit-learn
- Flask
- MetroUI (for HTML / CSS template)

List of Texts / Sources Used in Database (Incomplete)

- Fight Club
- Pride And Prejudice
- Freakonomics