IITB-Monash Research Academy

Sarcasm Suite

http://www.cfilt.iitb.ac.in/sarcasmsuite

Aditya Joshi, **Diptesh Kanojia**
Pushpak Bhattacharyya, Mark Carman
adityaj@cse.iitb.ac.in, diptesh@cse.iitb.ac.in




Overview

- Sarcasm Suite is a browser-based engine that deploys five of our past papers in sarcasm detection and generation.
- The sarcasm detection modules use four kinds of incongruity:
  - *sentiment incongruity,*
  - *semantic incongruity,*
  - *historical context incongruity and*
  - *conversational context incongruity.*
- The sarcasm generation module is a *chatbot* that responds sarcastically to user input.

Introduction

- Sarcasm detection gained attention from the sentiment analysis (SA) community for the challenges that sarcasm poses to typical SA systems.
- Several approaches to detect sarcasm have been reported Joshi et. al. (2016).
- This demonstration deploys five of our papers on sarcasm detection and generation. The demonstration titled '*Sarcasm Suite*' is a browser-based engine that allows users to test these systems.

Detection: Sentiment Incongruity

This demonstration is based on our paper: **Aditya Joshi, Vinita Sharma, Pushpak Bhattacharyya**, '***Harnessing Context Incongruity For Sarcasm Detection***', ACL-IJCNLP 2015, Beijing.

- In Joshi et. al. (2015), we use **sentiment incongruity** to detect sarcasm. We experiment with two sets of features: explicit incongruity features,
  - *number of sentiment flips,*
  - *sentiment subsequence lengths, etc.,*

and implicit incongruity features (which are phrases with implicit sentiment).

This detector is expected to work for sarcastic sentences bearing sentiment. The classifier uses features based on a linguistic concept called context incongruity. For example, 'I love being ignored'.

Detection: Semantic Incongruity

This demonstration is based on our paper: **Aditya Joshi, Vaibhav Tripathi, Kevin Patel, Pushpak Bhattacharyya and Mark J Carman**, '*Are Word Embedding-based Features Useful for Sarcasm Detection?*', EMNLP 2016, Austin.

- In Joshi et al. (2016b), we capture **semantic incongruity** via word embeddings in order to detect sarcasm.
- We experiment with two kinds of features:
  - *regular features and,*
  - *distance-weighted features.*
- This detector is expected to work for sarcastic sentences that may not bear sentiment. It uses features based on word embedding similarities between words. For example, 'A man needs a woman like a fish needs bicycle' is incongruent (and hence, sarcastic) because of the distance between `fish' and `bicycle'.

Detection: Historical Context Congruity

This demonstration is based on our paper: **Anupam Khattri, Aditya Joshi, Pushpak Bhattacharyya, Mark J Carman**, '*Your sentiment precedes you: Using an author's historical tweets to predict sarcasm*', WASSA workshop at EMNLP 2015, Lisbon.

- In Khattri et al. (2015), we use an author's historical context in the form of their twitter timeline to detect sarcasm in their tweets.
- This is a rule-based technique that calculates surface sentiment of a tweet and compares it with the author's sentiment towards phrases in the tweet, in the past.
- So, if a user says 'X is the best Presidential candidate we have ever seen', this detector looks up the user's twitter timeline to determine their sentiment towards the given person X, and predicts sarcasm.

Detection: Conversational Context Congruity

This demonstration is based on our paper: **Aditya Joshi, Vaibhav Tripathi, Pushpak Bhattacharyya, Mark J Carman**, '*Harnessing Sequence Labeling for Sarcasm Detection in Dialogue from TV Series `Friends*", CONLL 2016, Berlin.

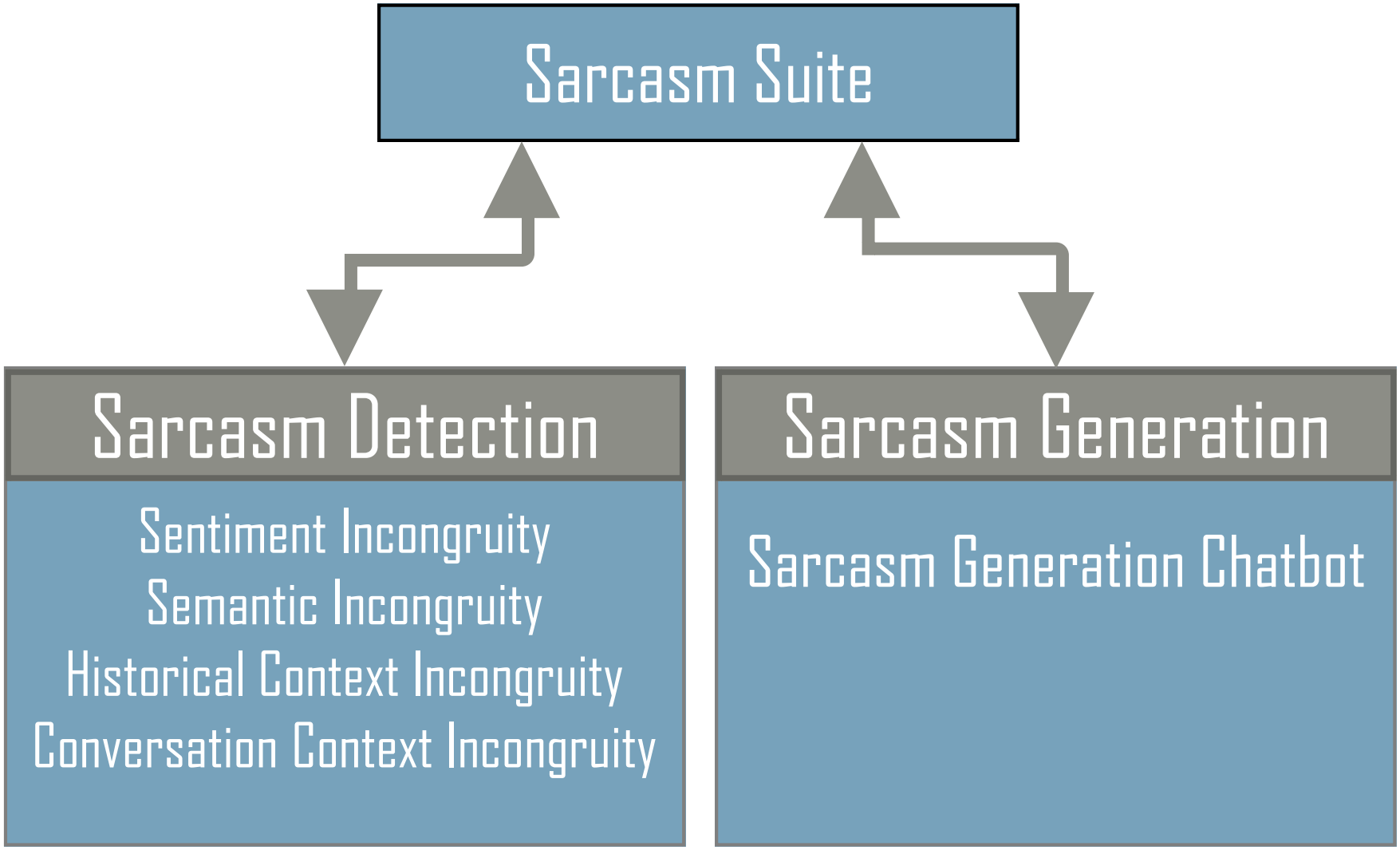
- Sarcasm detector that uses conversational context: This detector is a supervised classification system that takes into account sequential nature of a conversation, in order to predict sarcasm.
- It takes as input a conversation (a series of utterances), and predicts sarcasm in each utterance using a sequence labeling algorithm, and a set of features.

Sarcasm Generation

This demonstration is based on our paper: **Aditya Joshi, Anoop Kunchukuttan, Pushpak Bhattacharyya, Mark J Carman**, '*SarcasmBot: An open-source sarcasm-generation module for chatbots*', WISDOM workshop at SIGKDD 2015, Sydney.

- In Joshi et al. (2015), we present a sarcasm generation module that responds sarcastically to user input.
- The module implements eight sarcasm generators, each covering a peculiar form of sarcasm.
- SarcasmBot is an innovative, open-source (github link in the paper) chatbot module that generates sarcastic responses to a user input. It is a rule-based sarcasm generator that implements eight forms of sarcastic expressions.

Architecture



Implementation

- We use server side scripting in PHP to create the engine. We have used Bootstrap CSS Framework released by Twitter Inc. for the front end, and jQuery-based AJAX requests that query the modules.
- The implementation of sarcasm generation was done in Java.
  - *Available at the github repository at: https://github.com/adityajo/sarcasmbot/*
- The implementation of sarcasm detection modules was done in Python, with calls machine learning tools: SVM-perf Joachims (2006), SVM-HMM Altun et al. (2003) and SVM-Light Joachims (1999).

Conclusion

- Sarcasm Suite is a unique engine that demonstrates five of our past works related to computational sarcasm:
  - *four sarcasm detection approaches and*
  - *one sarcasm generation approach.*
- The engine offers a variety of approaches that use sentiment flips, word vectors, historical context based on twitter timelines, etc. as cues for sarcasm detection.
- Sarcasm Suite will enable the research community working in sentiment analysis to identify challenges encountered in sarcasm detection.

Also See

- Aditya Joshi, Pushpak Bhattacharyya, Mark James Carman, Automatic Sarcasm Detection: A Survey, arXiv preprint arXiv:1602:03426 (2016).and
- **Our upcoming tutorial on "Computational Sarcasm" at EMNLP 2017, in September 2017. (Speakers: Pushpak Bhattacharyya, Aditya Joshi)**