# **MUStARD: Multimodal Sarcasm Detection Dataset**

This repository contains the dataset and code for our ACL 2019 paper:

[Towards Multimodal Sarcasm Detection (An *Obviously* Perfect Paper)](https://www.aclweb.org/anthology/P19-1455/)

We release the MUStARD dataset which is a multimodal video corpus for research in automated sarcasm discovery. The dataset is compiled from popular TV shows including *Friends*, *The Golden Girls*, *The Big Bang Theory*, and *Sarcasmaholics Anonymous*. MUStARD consists of audiovisual utterances annotated with sarcasm labels. Each utterance is accompanied by its context, which provides additional information on the scenario where the utterance occurs.

## **Example Instance**



Example sarcastic utterance from the dataset along with its context and transcript.

## Raw Videos

We provide a [Google Drive folder with the raw video clips](https://drive.google.com/file/d/1i9ixalVcXskA5_BkNnbR60sqJqvGyi6E/view?usp=sharing), including both the utterances and their respective context

## Data Format

The annotations and transcripts of the audiovisual clips are available at [data/sarcasm\_data.json](https://github.com/soujanyaporia/MUStARD/blob/master/data/sarcasm_data.json). Each instance in the JSON file is allotted one identifier (e.g. "1\_60") which is a dictionary of the following items:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | **Key** | **Value** | | utterance | The text of the target utterance to classify. | | speaker | Speaker of the target utterance. | | context | List of utterances (in chronological order) preceding the target utterance. | | context\_speakers | Respective speakers of the context utterances. | | sarcasm | Binary label for sarcasm tag. | |

**Example format in JSON:**

{  
 "1\_60": {  
 "utterance": "It's just a privilege to watch your mind at work.",  
 "speaker": "SHELDON",  
 "context": [  
 "I never would have identified the fingerprints of string theory in the aftermath of the Big Bang.",  
 "My apologies. What's your plan?"  
 ],  
 "context\_speakers": [  
 "LEONARD",  
 "SHELDON"  
 ],  
 "sarcasm": true  
 }  
}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## **Run the code**

1. Set up the environment with Conda:
2. conda env create  
   conda activate mustard  
   python -c "import nltk; nltk.download('punkt')"
3. Download [Common Crawl pretrained GloVe word vectors of size 300d, 840B tokens](http://nlp.stanford.edu/data/glove.840B.300d.zip) somewhere.
4. [Download the pre-extracted visual features](https://drive.google.com/open?id=1Ff1WDObGKqpfbvy7-H1mD8YWvBS-Kf26) to the data/ folder (so data/features/ contains the folders context\_final/ and utterances\_final/ with the features) or [extract the visual features](https://github.com/soujanyaporia/MUStARD/blob/master/visual) yourself.
5. [Download the pre-extracted BERT features](https://drive.google.com/file/d/1GYv74vN80iX_IkEmkJhkjDRGxLvraWuZ/view?usp=sharing) and place the two files directly under the folder data/ (so they are data/bert-output.jsonl and data/bert-output-context.jsonl), or extract the BERT features in another environment with Python 2 and TensorFlow 1.11.0 following ["Using BERT to extract fixed feature vectors (like ELMo)" from BERT's repo](https://github.com/google-research/bert/tree/d66a146741588fb208450bde15aa7db143baaa69#using-bert-to-extract-fixed-feature-vectors-like-elmo) and running:
6. # Download BERT-base uncased in some dir:  
   wget <https://storage.googleapis.com/bert_models/2018_10_18/uncased_L-12_H-768_A-12.zip># Then put the location in this var:  
   BERT\_BASE\_DIR=...  
     
   python extract\_features.py \  
    --input\_file=data/bert-input.txt \  
    --output\_file=data/bert-output.jsonl \  
    --vocab\_file=${BERT\_BASE\_DIR}/vocab.txt \  
    --bert\_config\_file=${BERT\_BASE\_DIR}/bert\_config.json \  
    --init\_checkpoint=${BERT\_BASE\_DIR}/bert\_model.ckpt \  
    --layers=-1,-2,-3,-4 \  
    --max\_seq\_length=128 \  
    --batch\_size=8
7. Check the options in python train\_svm.py -h to select a run configuration (or modify [config.py](https://github.com/soujanyaporia/MUStARD/blob/master/config.py)) and then run it:
8. python train\_svm.py # add the flags you want
9. Evaluation: We evaluate using weighted F-score metric in a 5-fold cross validation scheme. The fold indices are available at data/split\_incides.p . Refer to our baseline scripts for more details.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## **Citation**

@inproceedings{mustard,  
 title = "Towards Multimodal Sarcasm Detection (An \\_Obviously\\_ Perfect Paper)",  
 author = "Castro, Santiago and  
 Hazarika, Devamanyu and  
 P{\'e}rez-Rosas, Ver{\'o}nica and  
 Zimmermann, Roger and  
 Mihalcea, Rada and  
 Poria, Soujanya",  
 booktitle = "Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)",  
 month = "7",  
 year = "2019",  
 address = "Florence, Italy",  
 publisher = "Association for Computational Linguistics",  
}