Can we (artificially) understand Seinfeld's humor?

Dan Kufra (dank) [dan.kufra@mail.huji.ac.il]
Yuval Jacoby (yuvalja) [yuval.jacoby@mail.huji.ac.il]
Alon Netser (31160253-6, alonnetser) [alon.netser@mail.huji.ac.il]

February 23, 2019

Contents

Abstract

Who among us isn't familier with the hilarious sitcom "Seinfeld"? We all laughed from Jerry's stand-up scenes (YouTube), George Costanza ridiculous behaviour (YouTube), Kramer facial expressions and bursting into the room (YouTube), and Elaine pushing everyone away and shouting "Get Out!" (YouTube).

The main question we ask is - can we build a model to understand this humor like we do?

1 Problem Descrpition

We want to build a model that "understand humor", i.e. given the text of a scene in Seinfeld, will predict funniness for each sentences said.

First, we validate the data. We find certain flaws and think about how to deal with them. Second, we analyze the data and visualize its content. Last, (just for the fun of it) we visualize the connections between the different characters in the show, and build an interactive graph describing these relations.

1.1 Difficulties

There are many difficalties regarding this problem.

Here are a few:

1.1.1 Funny is not taken only from the text

When we watch Seinfeld and laugh, it is not only from of the text. Most of the times it affected from the tone the person was saying that sentence, when he emphasized, and more generally the whole scene's video. This is the case most of the times in Kramer funny scenes when he enters the room demonstratively and "in a funny way". These aspects are not always obvious from the text iteself.

Indeed, if you try to read the subtitles for a specific Seinfeld's episode, you would not always realize what is so funny.

This leaves us with the question - if we, as human beings, can not always realize funniness from the text alone, how can we expect a computer to do that?

1.1.2 Subjectivity of Funny

The laugh-track does not necesarily mean that this is funny. After all, Seinfeld is a TV show and the goal of its writers it to profit. If you want people to enjoy your show it is essential to encourage them to laugh, and this is done by adding the "laugh-track".

First of all, this issue is not that severe. Indeed, the laugh-track does not always indicates funniness, but the writers who decided to add the laugh-track at a certain position certainly thought that this might be funny. Therefore, learning when a laugh-track will occur is also interesting because it is the same as given a sentence, predict if some human-being thought it might be funny.

Second, the accuracy of the laugh-track is quite good. Most of the times there is a laugh-track there is indeed something funny going on. Moreover, the coverage of the laugh-track is quite close to 100%, i.e. there is (almost) no funny line that is lacking a laugh-track.

Anyway, this issue is the reality and we can not change it. Even if we would label the sentences by ourself, it would still mean that some human-being (i.e. us) is thinking there are funny. it does not neccesarily mean that another person will think it's funny.

To conclude, we can be more precise and say that our goal is to predict when a sentence is supposed to be funny, as the Seinfeld authors decided.

1.1.3 Context

The other (many) difficulties ragarding natural-language-processing that were mentioned in the lecture also affect us (ambiguity, language is not static, etc).

Morever, the context is really important in understanding humor. When we watch an episode of Seinfeld we understand the overall topic, and remember events that occured previously in the episode (and even in other episodes). In order to make the computer understand the humor as well, we must use a model that is capable of understanding context and "remember" important stuff.

2 Data

2.1 Introduction

The data we use is basically Seinfeld's subtitles, coupled with the speaking character. In addition, we have the exact time in which every character's sentence begins and ends, and the exact times in which the laughter occured.

We downloaded the dataset from GitHub, and we thank Ran Yad Shalom and Yoav Golberg for the great dataset they built [1].

2.2 Specs

The dataset contains 96 humor annotated “Seinfeld” screenplays, along with the timing of the laughter and the timing of the dialog.

R. Y. Shalom and Y. Golberg [1] got the subtitles from opensubtitles.org, the scripts from seinology.com and used the audio tracks to extract the exact timing of the laugh-tracks. Furthermore, they used quite sophisticated techniches to align the subtitles with the exact timing, and attach the speaking character for every sentence using the scripts.

Due to the technic used by [1] to build the data (using the fact that the dialogs were recorded in mono, and the laugh-track in stereo) we have the episodes starting at season 4 episode 6. This is because the previous episodes were not recorded this way.

The final dataset contains 96 episodes, with 46,497 sentences in total. We took the raw data from the GitHub repository and inserted it into a pandas DataFrame, and added several useful attributes. Each sentence is associated with several properties:

- ’character’: The speaking character.
- ’txt’: The text.
- ’start’: Start time (in seconds).
- ’end’: End time (in seconds).
- ’is_funny’: Whether a laugh occured after this sentence or not.
- ’laugh_time’: If this sentence is funny, this is the duration of the laugh (in seconds). Note that if the sentence was not funny, this is set to NaN.
- Various meta-data about the episode, such as ’season’ (season’s number), ’episode_num’, ’episode_name’, ’total_lines’, ’global_episode_num’ (in the whole dataset).
- Useful features of the sentence, such as ’num_words’, ’length’ (in seconds), ’line_num’ (in the episode), ’avg_word_length’ (in letters).

2.3 Data Validation - no dataset is perfect in our world...

Remark. This section was done in an interactive Jupyter Notebook named ’Data_Cleaning.ipynb’. You are more than welcome to take a look!

The first task we addressed was analyzing the dataset and validate it.

We watched several episodes and looked at the dataset at the same time.

We saw that the text is pretty accurate, subject to minor changes the man (or machine) who translated it did. These changes are not really an influence to our task, because most of the time they have the same meaning. For example, "You met her in the supermrket. How did you do that?" is shorten to "You met her in the supermrket. How?". For humor detection, this is not really a severe issue.

The timing of the talking are also pretty accurate, and by reading the paper of Ran Yad-Shalom [1] who created this dataset he addressed this issue specifically and payed extra attention to take several subtitles and choose the one that is best aligned with the audio. His job here is pretty impressive.

2.3.1 The first flaw: the laugh-tracks

The laugh tracks are not perfect. It's mostly labeled correctly, but there are some false-positives. Also, the timing of the laughs are not measured perfectly. It's quite understandable, because in the show it's also not so obvious when the laugh start and when it ends. Extracting it automatically by analyzing the audio (like Ran Yad-Shalom did) seems pretty challenging. However, because most of the time it's actually labels correctly, we treat this as acceptable noise in our dataset and continue with our work.

2.3.2 The second flaw: the speaking character is sometime mislabeled.

We noticed that the speaking characters sometime is not correct. We found an attribute that might be the cause for mistakes in the speaking character labels - one character speaking a lot of sentences in a row. It seems that while building the dataset, some characters labels "get stuck" and the same character (possibly wrongly) is labeled as the one speaking.

We found that this is often when some character is speaking more that 10 sentences in a row. Indeed, this is quite rare for the same character to speak more than 10 sentences in a row. There are exceptions of course, for example when a person is talking over the phone (and the other side is not heard), or when Jerry is doing his stand-up scenes.

We tried to capture the mistakes in the talking character by the amount of sentences he is saying in a row. To do this, we collected all these cases, and analyzed them statistically. We saw that the mean sentences-in-a-row is around 1, and that 99% of the times it is below 8.

Our solution was to mark these speaking characters as 'unknown', but keep the other attributes (such as text, duration, funniness, etc). Indeed, there is no reason to remove these line from the dataset - most of it is correct, the only part that is not correct is the talking character. We can still use the funniness and the words in the sentence. In models where we take the talking character into account, we need to ignore these. But in other models - it's perfectly OK.

2.4 Data Visualizations

Remark. This section was done in an interactive Jupyter Notebook named 'Visualizations.ipynb'. You are more than welcome to take a look!

We tried to visualize several aspects of the data, in order to get more insight about it and get ideas about how to solve it.

First of all, we tried to see if funny sentences are longer (generally speaking). We saw that the distributions of the length / #words is different between funny and not-funny sentences. However, these differences are not so significant, as the standard deviations of the length is also pretty high. Nevertheless, we can say that the length of the sentence is a features that might help in classifing it as funny/not-funny. Moreover, we found the the length in seconds and the length in #word behave different. For example, Elaine's mean #words is quite similar in funny/not-funny sentences (about 5.71), but the length in seconds is quite different (2 v.s. 2.5).

Our conclusions from this visualization is to add more features to the sentence, such as #words, length (in seconds), speech-rate (words per second), etc. Afterwards, we saw that the results actually improved when we gave them these features.

Another nice visualtions we did was to analyze the speaking characters. One thing was to make a word-cloud for each character, as well as to separate for funny/not-funny sentences and see if we can see any difference. Beside everything, these word-clouds are just fun to look at. We can see that there are many common frequent words between funny/not-funny sentences. However, we can see some interesting issues. For example, in Jerry's not-funny sentences the word 'Elaine' is more common. Can we learn that when Jerry says 'Elaine' in a sentence it is less likely to be funny?

Another nice visualization was the characters-graph, containing the different characters and their connections. Each edge's weight is proportional to the amount of times the characters spoke to each other. One can see perfectly the “community” of the 4 main characters (Jerry, George, Kramer and Elaine), as well as other secondary characters (such as George parents Monty and Helen).

3 Experiments

TODO

4 Future Work

TODO

5 Conclusion

TODO

References

[1] Ran Yad-Shalom and Yoav Goldberg, The Seinfeld Corpus: A Method for Generating A Humor Annotated Corpus and An Analysis of That Corpus. Computer Science Department, Bar-Ilan University, Israel, 2017.