Snarkbot: A Technical Solution to Deal with a Face-To-Face Bullying Problem

Tess DiStefano
Drexel University
3141 Chestnut St, Philadelphia
tess.distefano@gmail.com
(610) 996-3386

ABSTRACT

Updated 6/8/2016. The Snarkbot project followed a process of development ensuring that everything created matches an original specification. The goal of the application is to provide a tool for children to stand up to bullies.

Author Keywords

Drexel; Computer Science; Female; BSMS; Undergraduate.

ACM Classification Keywords

Algorithms; Design; Documentation; Human Factors; Machine Learning; Graphical User Interface; Grid Format; Children; Bullying.

INTRODUCTION

A problem many people face is not knowing how to respond when somebody says something mean to them. Currently, this problem is approached by either thinking on the spot and spitting out whatever a person thinks of first, or not responding at all. For either option, people often probably their choice later on.

My approach for solving this problem is to design an application people can use to input a mean phrase and receive advice based on how the system responds to this phrase. Ideally, the extremely simple AI in this system will come up with an amazing comeback that the user can repeat to respond to whoever said something mean to them. However, this is not always the case.

In this paper, Tess DiStefano describes the system implementation, changes based on feedback, evaluation, results, and future work for the Snarkbot project.

RELATED WORK

Bullying is a problem online as well as in person. While many technological solutions to bullying specifically cater to cyberbullying, this solution is intended to help kids stuck in a face to face scenario in which time to think before responding isn't given.

In a paper on cyberbullying with teenagers [3], teenagers were asked to prototype systems to prevent cyberbullying. One of the suggested solutions uses string parsing to detect when a user is being bullied. I discuss in the "Future Work" section how I think this idea could really help the Snarkbot application even easier to use. Another prototype designed in this study is intended to help users after being bullied, however the purpose of this application is to use therapy with the user rather than actually help them face the bully head on. Finally, one application discussed touches on some of the ideas used in designing Snarkbot. The paper discusses a prototype designed to read online posts to Twitter and fire back at bullies by auto-generating responses when somebody "roasts" another user. I think this would be a great solution, but my system has the potential to do a better job of responding to bullies by allowing users to add more personal information about the bully. Also, my solution deals with face to face bullying as opposed to these proposed prototypes.

Another paper is focused on figuring out what a bully's purpose is [1]. The purpose of the paper was to evaluate and discuss potential ways to prevent cyberbullying by matching what a bully says to a script outlining the bully's goal. This doesn't really help the user with any sort of call to action though.

Finally, "Feelbook" is a system whose prototype was discussed in another paper presented at a previous conference [2]. The purpose of this application is to fix the system rather than fix one user's experiences. So users would include bullies or potential bullies, and the prototyped system is designed to help them fix their attitudes and act in a more respectful and respectable way. The goals of this system are to help teenagers reflect, empathize, feel empowered, and help the greater good of their communities.

My design is unique because it will handle face to face events and will help users respond right away to what a bully might say. Details added to responses about the user and antagonist are both used to personalize responses. Rather than helping an online community, my application Snarkbot will help a physical community by ensuring that no person is seen as inferior. This is done by helping people keep from being targets of bullying. By helping somebody with how they handle being bullied, we can prevent them from being bullied again in the future.

SYSTEM

The system details are discussed in the following subsections, focusing on how the user will use the application and what will be going on in the back end when the user interacts with a form.

Implementation

The system provides an interface and functionality for the user to find out how to respond in a snarky way to an insult given to them. In addition to inputting the phrase to respond to, the user should also specify (using GUI elements) what level of snarkiness to use. The system calculates a response quickly based on user input which was either input previously as profile data or input on the fly. It should have a diverse enough pool of possible responses that responses are not repeated often if at all.

The main functionalities of this system are: entering and modifying user information, entering and modifying adversary information, and requesting responses. The syntactic level design is listed below in the form of state diagrams to show the control flow of the system for each scenario. Screenshots of the prototype are also displayed to show how the user should be able to interact with the system in order to complete tasks.

Enter and Modify User Information

Because the user is intended to be represented in the actual application, there has to be a way for the user to describe the character representing them. They do this by adding and modifying their user profile.

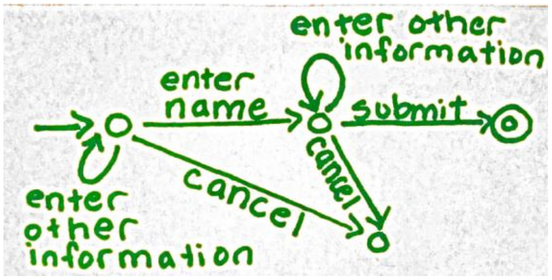

Figure 1. When first entering a user profile, the user should be forced to enter a name to represent them. This helps with linking the user interface to the real world by showing that the profile is intended to be a representation of the user. Other information, such as age and gender, are optional but can be used to help the user feel a relation to the profile they are creating. These data fields will also be used, if filled out by the user, to personalise responses.

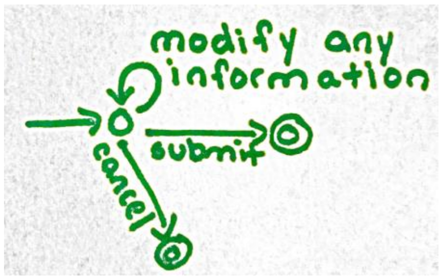

Figure 2. The control flow should be similar but not the same when first creating a user profile versus modifying a user profile. Since the user should have already entered their name when creating their profile, there is no need to enforce that when modifying the profile.

Enter and Modify Antagonist Information

Whether or not the antagonist will actually be given any filled out information, the user does have to create the antagonist. Whether or not the use choses to add a name or other details, the user still has to submit the antagonist entry. This is so that the antagonist can be represented as an object in the program.

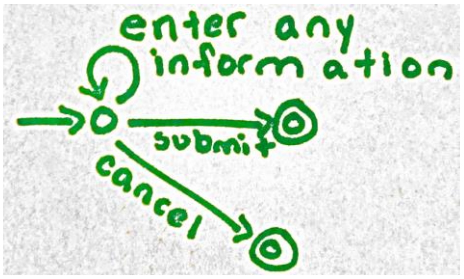

Figure 3. The antagonist's information is less crucial than the user's own profile. Because of the nature of being teased, it's possible that the user doesn't have any information on the antagonist. So using this state diagram to show how the user should interact with the system to complete the task of detailing their adversary, it is simple to show that the user can choose to enter information or leave it out. Whatever information is entered will help the user relate the system to the real world and can be used when generating responses that the user should say to the adversary.

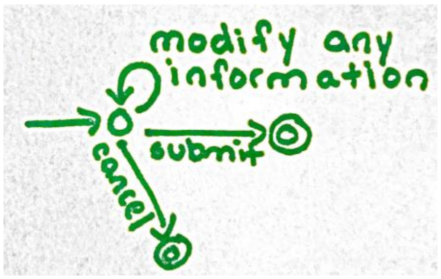

Figure 4. Modifying antagonist information is actually treated the same way as creating an antagonist for the first time, since none of the data is required.

Request Response

In order to optimize communication between the user and the system, I tried to make the process of requesting responses as simple as possible. Assuming that the user has already completed the previously mentioned steps of creating their own profile as well as creating a profile for the antagonist (even if it isn't filled out at all), it should be ideally very quick and easy to calculate a response given an insult from the adversary.

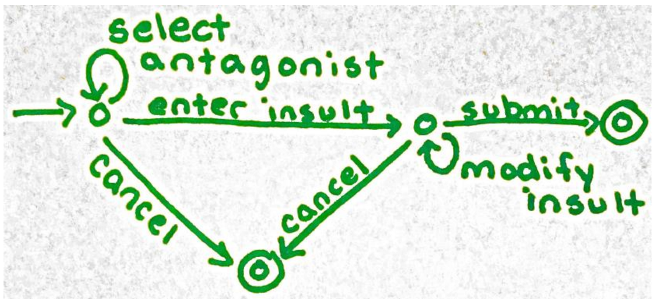

Figure 5. The ideal flow of events would be to enter the insult and submit the insult. However, in order to allow modifications to input based on different user circumstances, of course the options of changing the antagonist, fixing a mistyped insult, and simply cancelling a request are also options

Changes Made Based on Original Feedback

For my initial design, I had the snark level as a profile level attribute. However, upon getting suggestions from users and seeing how confused this made them, it became clear to me that the snark level should be modified on the same screen as where the user enters the other input to be used to generate a response. This is because, depending on a user's current mood, their snark level could affect how "snarkily" they want to respond to their adversary. In response to similar comments, I also added a tooltip showing users what a snark level is when they first open the home page so that they know why it's there and what it means.

Some other user interface requests I used to modify the prototype were to give more feedback to the user. Specifically, I used feedback to make it clear that, on the home page, the way to modify a user or adversary is to click on their respective "profile" image (I do not want this term to be misleading, as the "profile" images are not editable by the user). In the same school of thought, I was asked to add gray text telling the user what to write in each input location. This also serves to make it clear from the user interface which text boxes are for users and which ones will be filled out by the system. My last modification was to get rid of "back" buttons and replace them with "cancel" buttons. This way, when the user is not on the home page, it is clear how to submit anything they've done versus how to go back to the home page without saving any information.

EVALUATION

The system was presented to four potential users to evaluate it for functionality and usability.

Process

After the preliminary tests done in class, I got four other users to test the system. Because the users are intended to be elementary or middle school aged children, I tried to get as many of those as possible. I was able to reach two 12-year-old children in 8th grade, as well as two college level adults aged 21 and 22. Unfortunately all of my testers were male, so I was not able to obtain data regarding how females would use the system.

The method I used to conduct my test was to sit down with the user and ask them to speak their mind out loud narrating their thought process. I also informed them that I would not answer any questions, as all I wanted to know was how easy my user interface was to use. My goals were made clear to testers, in that I wanted to see how much difficulty the system brought users when attempting to achieve their goals.

Starting off the test, I opened the prototype and told user to begin personalizing it. Because usage of the system is based off of outside events (being insulted by some sort of adversary), I acted as the adversary myself giving an insult to the users so as to simulate a real life scenario to use the system. I used the same insult for each person "Your shirt makes babies cry." I also measured times it took to complete tasks, as described in more detail below.

Results

There were some similarities between user reactions and some reactions that were unique. The two younger kids seemed amused by the scenario which may have been because they aren't familiar with the idea of testing a user interface (also the topic of my project might be amusing to kids). After getting past the original amusement, the kids seemed to understand what was intended by personalizing the application and filled out their profiles completely rather than just filling out their names. Conversely, one of the college aged students just filled out his name and the other filled out his name and gender.

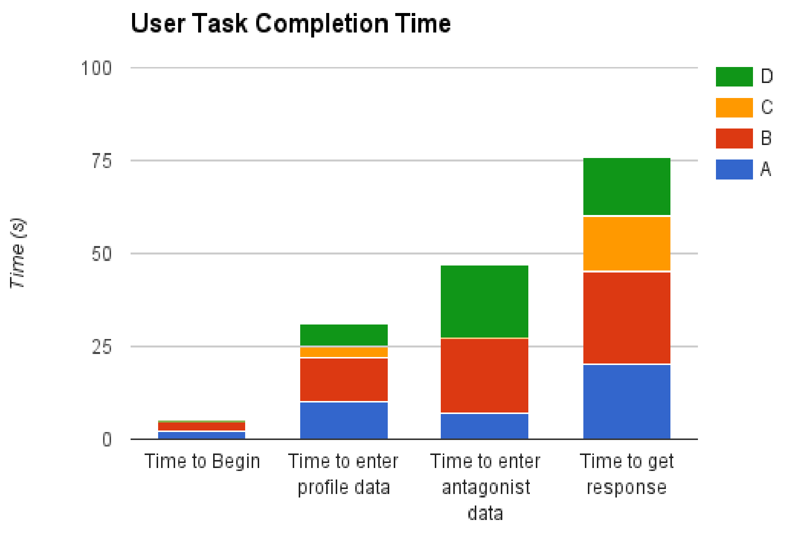

Figure 6. This graph shows the difference between how long it took different testers to complete specific tasks. The legend should help identify the demographic of each tester.

Discussion

My reaction to details like amusement was that maybe I hadn't thought about the idea of people not taking this system seriously. To some, the concept of needing help responding to insults seems crucial. But to others, it might seem unnecessary at least at first.

As you can see from Figure 8, the time to get started is the smallest amount of time because two of the users got started right away, as mentioned earlier. Based on this information, it's easy to tell that users A and B are the 12-year-olds. Overall, it took users more time to input their own data as opposed to the antagonists, but with user D more time was spent on the antagonist. I think that this is because user D asked if I was the antagonist or if he was supposed to imagine a real person who would really insult him. As you can see, tester C didn't enter any antagonist data which was expected as it is perfectly valid to not care about antagonist information. Tester B also took extra time on the antagonist page, and I observed that they looked confused about where to go next as they just stared at the screen for a bit. And finally, as expected, the time to get responses is the largest because users actually need to type up what the antagonist said.

FUTURE WORK

In regards to responses given during my tests, I think I might modify the design in the future. By changing the tone of the application to be more humorous, I think I could reach a wider audience. Instead of just kids who have issues thinking of responses, I could make this a fun game that any kid or adult could use for entertainment.

Also in regards to how long it took users overall to enter the insult, I think it would be very helpful and also interesting to, given time in the future, add integration with a speech-to-text system so that the user can simply press "record" to have the system parse the insult.

I also would like to implement some ideas found in other works, such as using language parsing to detect when a user is being bullied. This could be used with the speech-to-text functionality to have a response ready as soon as a rude remark is heard. That way, the user has nothing to do but pull out their phone and glance at it.

CONCLUSION

The Snarkbot application can be a very useful tool for helping prevent bullying by helping users handle being bullied in a more conclusive way. In a more short term respect, the application will help users get through situations of being bullied by aiding them through proper ways to respond to a bully.

ACKNOWLEDGMENTS

This was a project for Drexel class CS530, Developing User Interfaces. So I would like to thank my professor, Erin Solovey, for helping me through the development and design of this system.

REFERENCES

1. Jamie Macbeth, Hanna Adeyema, Henry Lieberman, and Christopher Fry. 2013. Script-based story matching for cyberbullying prevention. In CHI '13 Extended Abstracts on Human Factors in Computing Systems (CHI EA '13). ACM, New York, NY, USA, 901-906. DOI=http://dx.doi.org/10.1145/2468356.2468517

2. Mingyue Fan, Liyue Yu, and Leanne Bowler. 2016. Feelbook: A Social Media App for Teens Designed to Foster Positive Online Behavior and Prevent Cyberbullying. In Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '16). ACM, New York, NY, USA, 1187-1192. DOI=http://dx.doi.org/10.1145/2851581.2892398

3. Zahra Ashktorab and Jessica Vitak. 2016. Designing Cyberbullying Mitigation and Prevention Solutions through Participatory Design With Teenagers. In Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16). ACM, New York, NY, USA, 3895-3905. DOI=http://dx.doi.org/10.1145/2858036.2858548.