**Project Merme generator**

**Nicolas Di Costanzo - 21209060**

(nicolas.dicostanzo@ucdconnect.ie)

Contents

[Abstract 1](#_Toc99755748)

[Online Access 2](#_Toc99755749)

[1. Introduction 3](#_Toc99755750)

[2. The Core Idea 4](#_Toc99755751)

[3. Technical Approach: Architectural Perspective 3](#_Toc99755752)

[4. Data, Information, Knowledge 4](#_Toc99755753)

[5. Diversity and Divergence 5](#_Toc99755754)

[6. “Mere” Generation and “True” Creativity 5](#_Toc99755755)

[7. Evaluation, Self-Critique and Filtering 5](#_Toc99755756)

[8. Hits and Misses 5](#_Toc99755757)

[9. Conclusions 5](#_Toc99755758)

[References 5](#_Toc99755759)

List of Figures

Figure 1: Figure Caption ……………………………………………………………….. 1

Figure 2: Figure Caption ……………………………………………………………….. 2

Figure 3: Figure Caption ……………………………………………………………….. 3

Figure 4: Figure Caption ……………………………………………………………….. 4

Figure 5: Figure Caption ……………………………………………………………….. 5

Figure 6: Figure Caption ……………………………………………………………….. 6

List of Important Abbreviations Used Within

TLA Three Letter Acronym

FLW Four Letter Word

PFO Public Funding Option

# Abstract

# Online Access

https://github.com/NdcUcd/Merme-generation

# Introduction

"Merme generator" is about the simple generation of memes. My goal is to create a software that can generate memes that are meaningful representations of what human-created memes can be, while adding a creative touch that only a machine can simulate. What I plan to do is to create software that randomly determines an image and associates relevant text with that image. Of course, the images are all memes.

Then, what is a meme? Or at least, what definition of meme have I chosen to guide my work?

A meme is an image taken out of its context that is being associated with a new context for humorous purposes. There are other definitions, often broader than these, of what a meme is, but for the purposes of this project, this is the one I will stick to. Below, a good example of what the definition I will follow means:



Figure 1: Joker meme

This meme takes two images from the movie “The Joker” and applies a romantic context to it. The discrepancy between the dramatic aspect of the film scene and the tenderness of what a romantic relationship is creates the comic effect. This is particularly emphasized by the use of the word “fucking” in the first picture and the gun being replaced by a caricatural Valentine’s gift on the second one

However, from this meme can be guessed one of the drawbacks of memes: one needs to be aware of the image original context (or at least what kind of mood it is meant to represent) to understand it. This meme is probably far funnier for someone who watched the movie and can represent to oneself the mood of the original scene. I will talk later how I have addressed this issue.

This project has several advantages. First, it is relatively simple to set up technically. I spent about one week on the purely technical implementation of the project which means that I have been able to devote most of my time to work on the "creative" aspect of the project. Namely, to refine as much as possible the relevance of the outputs generated by the software. Also, I am a huge meme lover, so working on the theoretical aspects that underlies meme creation as well as trying to create a tool able to generate infinite memes is greatly valuable on a personal level. Finally, this project is relevant because the number of resources is huge. Every website, every Facebook, Twitter, Instagram profile can be a source of inspiration. It makes the

# 2. The Core Idea

The first step is to determine on which types of memes I will concentrate my work on. As mentioned above, there are lots of different definitions for this notion and the main thing to do is to narrow it down to a relatively small set of possibilities to push the creation as far as possible. I have therefore chosen to focus simply on memes which are images and their captions. However, even with this definition, there are still too many possibilities. So, I am still refining my work on only two types of memes: memes that illustrate situations and memes that illustrate reactions.

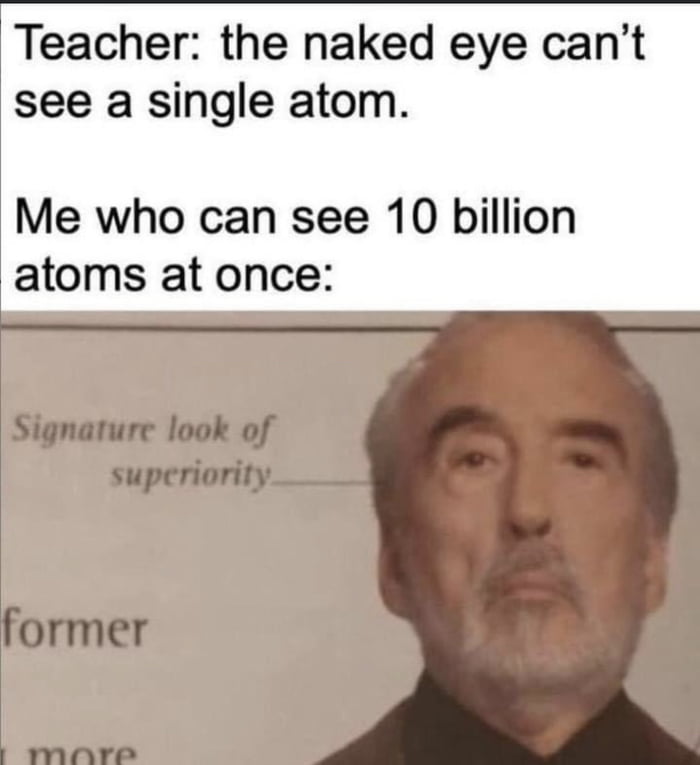


Figure 2: reaction meme



Figure 3: illustration meme

However, I would not use other types of memes such as the one which are jokes since they need a very accurate caption to work, such as the one below:



Figure 4: A “joke” meme

I would not generate memes which would need additional images or further work on the visual aspect of the original meme to be relevant since it would greatly complexify the technical challenge. As I said earlier, I want to focus on the creative aspect of the project and there is far enough to do without that kind of memes.



Figure 5: meme containing additional visual work

A meme which requires additional work on the original image. It would be far too complex to generate such kind of memes.

# 3. Technical Approach: Architectural Perspective

This software has been developed using both Unity and Tracery. I decided to use Unity since I have experience making games and other kind of software with it. Also, I did find a plug-in to use Tracery inside a Unity project which greatly eased my work.

Basically, there are only two things that happen on a technical level.

The first one is the randomized choice of image. Internet links of these images are stored in two different ArrayList of string. The first ArrayList contains images which correspond to “illustration memes” and the second one, memes which are “situation memes”. It is important to separate those two types of memes in term of data structure as they are, on a programming level, treated differently. So, being able to call different functions and methods according to the list used has been very handy.

The other thing that happens is the grammar generation, which is obviously determined by the image that has just been chosen by chance. The generated grammar will be determined by which list the image belongs to (so if it corresponds to an illustration or reaction meme). Through the Tracery plug-in, the software will “ask” for a different output.

Text

Description automatically generated

On a programming level, a meme is represented by 3 attributes:

An image which is randomly chosen by the machine. This image will define the other two attributes’ values. The input given to the Tracery plug-in will depend on the image chosen. For example, if it is an image from the first list (when rand\_list = 0 it is a “reaction meme”), it will ask Tracery to generate a title. While if it is an image from the second list (“an illustration meme”), it will generate words to integrate into the image.

Finally, the last attribute is only useful for the “illustration memes” since it gives the positions at which the words must be placed on the image.

|  |
| --- |
| Meme |
| RawImage image  String traceryAttributes  GameObject[] textsLocations |

# 4. Data, Information, Knowledge

One of the main advantages of such a project is that there is almost an infinite number of resources. The whole internet is my resource. However, for obvious reasons I must limit my area of research.

I found a website, <imgflip.com>, which collects a wide range of different memes. The big advantage of this website is that in top of having blank versions of memes, it also stores users’ creations which allows oneself to understand how it is used. So, in top of being a resource of images, it can also be an inspiration to get ideas on how to use a certain meme.

I intend to use sixty different images. Thirty illustrations and thirty situations’ memes. Reactions’ memes have only one caption which is the title while illustrations memes can have several different captions. To chose which images to pick I had two criteria in mind. The first one is that the image does not need any knowledge background to be understood. For example, if one needs to have seen a particular movie, show or video to understand it, I will exclude it. I want the generator to be as universal as possible, so I tried to pick only images that are independent from their original source to be understood. A very good example of that is the famous Pikachu meme. One does not need to have watched any Pokemon episode, nor to even know Pokemon. Indeed, the image speaks for itself, and Pikachu’s “facial expression” is talkative enough not to need anymore context to be understood. This is exactly the kind of images I want to use for that project.

The second criteria for an image to be picked is that it needs to be unique from the other memes already picked. I have chosen to integrate the Pikachu meme in my project, but I will not use the ????? meme in top of that one since they show the same reaction. I tried to choose 60 different memes that have some uniqueness in the way they depict a situation or a reaction which describes something new from the other images of the set.

However, even if memes’ “mood” can be very different, the generation can be irrelevant. Because of poor grammar, some memes which are originally different can be looked alike which would dull the set.

# 5. Diversity and Divergence

# 6. “Mere” Generation and “True” Creativity

# 7. Evaluation, Self-Critique and Filtering

# 8. Hits and Misses

# 9. Conclusions

Working on this project made me realize how relevant it is: I have understood why so many outputs were funny while they are completely random. The fact that the outputs are generated by a machine is funny in its own way. We humans have a reflex to look for patterns in what we experience. Therefore, for example we are so good at spotting face shapes in nature. And that is also why this project works well: even though there is no intention from the machine to do something funny or even meaningful, when a new output is generated, we try to find a meaning to it. Of course, it is not always obvious nor possible, but what I have personally experienced is that it is often the case. I think that this project, or at least this idea was worth to be worked on in a “mere generation” perspective in that sense. At least, this is what I have understood and retained from this module.

I would like to conclude that report by thanking the whole internet for its imagination and for being able to create such an amazing content that never cease to amaze me! Thank you, internet user!

# Acknowledgements

# References

Dantes, E. (1762). Escaping from tight corners. Monte Cristo press, Paris, France.

Squarepants, S.B. and Tentacles, S. (2003). Hygiene Issues In The Crabby Patty. Journal of Aquatic Foodstuffs, vol. 7, no. 6, pp 23-32.

Drumpf, D. J. (2016). The Effects of Magnetism on Cats. Phys. Rev. Letters D., vol. 203, no. 8, pp 56-59.

Turing, A. M. (2019). The Turing Test: A History of a Misunderstood Idea. Journal of Paranormal Communications., vol. 17, no. 18, pp 13-29.