Third Year Project Report

Adrien Schurger-Foy

Abstract What makes an image "memorable"? There has been a lot of interest in ways to automatically suggest hash tags for images. Can we build a detector that automatically determines if an image would likely be tagged as "memorable"? This is challenging because there are very many ways for a photo to be memorable. This project is a study of Twitter, more specifically of tweeted images. I was interested in identifying the characteristics of "memorable" images. I collected images tweeted with "memorable" hashtags (#unforgettable, #memories...), then I analysed them and determined which graphical features constituted a statistically significant difference between such images and any other tweeted image. After developing my project, I evaluated the certainty of my results and the correctness of my procedure, leading me to correct some conclusions. The purpose of this project was to serve as a foundation for building a detector capable of finding memorable images amongst the vast quantity of images posted on social media on a daily basis.

Results Amongst the graphical features I extracted, three revealed a significant difference between memorable and non-memorable images. The first result I obtained was that memorable images contain more faces on average. Following this, I noticed that nearly 15% of memorable images I collected contain some form of text presenting a "memorable" word or hashtag. I also determined that text density (a measure of how spread apart text is in an image) constitutes a statistically significant difference between memorable and non-memorable images. However, during the evaluation of my project, I found the certainty of this last result to be low.

Contents

1 Introduction

Classifiers are very well known and well studied, and as such they may easily come to mind even in situations they are not suitable for. A classifier has the goal of correctly assigning one of many labels to an item, with the highest accuracy possible. In situations with extremely unbalanced datasets, one's first reaction may be to use a classifier but will eventually come to the following conclusion: the best classifier is one that assigns the overwhelmingly more common label at every instance. However, if this is an unsatisfactory result, it means that a classifier is not a tool suitable for the problem in question.

In these cases, the goal may not be to assign as many correct labels as possible, but to detect something reliably. The purpose of a detector is to catch the instances where something of interest is present. Therefore, true positives are valued much more than true negatives, meaning that the goal is not to increase accuracy (unlike a classifier). Detection is sort of like a classification problem, but where the non-target category is far too heterogeneous to be meaningfully labelled. Detectors are much less studied; in many cases classification problems are tackled effectively but detection constitutes a bottleneck.

To illustrate this point, I will elaborate on the challenges encountered in the development of brain-computer interfaces. A brain-computer interface (BCI) uses brain activity as an input, allowing the user to control a computer without physical movement. For example, there is a BCI that outputs "left" or "right" depending on whether the user is thinking about moving their left or their right hand. This is clearly a classification problem: the BCI needs to label brain activity data as "left" or "right". However, this BCI will just output a random value whenever the user is thinking of something else, or is simply not attempting to control the BCI. This is the case with early BCIs, there is no distinction between an intentional control state and a non-control state.

Today, classifying brain activity data has been successful for the most part, but the issue of detecting when the user is intending to control the BCI is what constitutes a challenge. This detection is not something a classifier is suitable for. Indeed, in the vast majority of instants, the user is not intending to control the BCI at all, leading to an extremely unbalanced dataset. In general, solutions to the problem of detection are far behind the progress made towards classification.

The Viola–Jones object detection framework [1] is a competitive solution to a problem of detection, primarily oriented towards the detection of faces in video footage. In most frames, there is no face, so we can see why a classifier is not suitable for this problem. For such a task, it's necessary to identify features that characterise the object to detect. In the case of the Viola–Jones object detection framework, the characteristics of a human face are matched using a concept called Haar features.

In my project, I attempt to determine what makes an image memorable, in other words, I set out to find the characteristics of "memorable" images, and how they differ from any other image. This is in effect the prerequisite to building a "memorable" image detector. The task would be to find "memorable" images in a sea of "other" images as opposed to assigning labels to a class. While hashtag prediction is relatively well studied, fewer have worked on the task of hashtag detection, which is what my project aims to contribute to.

2 Overview

In this project, I am interested in identifying what distinguishes a memorable image from any other image. More specifically, I am studying images tweeted with hashtags such as "#memories", "#unforgettable", "#onceinalifetime", etc. Furthermore, I am only concerned with the difference between the images themselves, not the tweets or accounts that contain the images. The general procedure of the project was as follows:

- Collect data using the Twitter API and assemble a dataset of tweeted images selected as randomly as possible.
- Determine which images to label as "memorable", based on their hashtags (I will be calling these images "target images").
- Extract graphical features such as brightness, background color, number of faces, etc. . .
- Analyze the extracted data and identify which features reveal a difference between the two classes of images.

In terms of the scientific method, my project would be structured in the following manner:

- **Question:** are there intrinsic graphical differences between "memorable" and non "memorable" tweeted images?
- **Hypothesis:** if there are differences, features extracted from samples will reveal statistically significant differences between the two classes of images.
- **Prediction:** the number of faces, brightness, contrast, and other features will reveal differences.
- **Experiment:** collect samples and carry out statistical tests.
- **Results:** there are in fact intrinsic graphical differences revealed by certain features.

Figure 1 – Project timeline

This project has led me to conclude that target images contain more faces on average, and that a significant number of target images actually contain text that displays target ("memorable") hashtags. Additionally, I have found that text that is displayed in target images is less spread out than in their counterparts; however, the statistical certainty of this result was revealed to be low during the evaluation of my project. I have also uncovered some tangential results that are not directly relevant to my project, but are interesting enough to be discussed in this report.

3 Project Development

3.1 Data Collection

The first objective for this project was to collect a sample of tweeted images small enough to study, but most able to represent accurately the actual population of tweeted images. To achieve this, the sample should consist of unique and randomly selected tweeted images. The "random selection" that I needed here was the maximum-entropy kind, where any image is equally likely to be sampled: the sample should be as unbiased as possible.

Therefore, my initial approach to data collection consisted of collecting a random subset of all tweets, and keeping only those containing images. Furthermore, retweets need to be removed for if they are kept, popular tweets and accounts will be over-represented and lead to a biased sample. To obtain this sample, I first considered collecting tweets by picking a random Twitter account, and then sampling random tweets posted by that account. However, I quickly dismissed this idea, because it would clearly result in a random sampling greatly differing from a uniform distribution: there is a difference between "a random image" and "a number of random images from a random user".

A less biased approach consists of generating random tweet IDs and fetching the corresponding tweets. Twitter uses an internal service called snowflake [2] to generate unique 64-bit unsigned integers used as tweet IDs. These IDs are generated in function of time and are not sequential, leaving large gaps between any two IDs. In fact, these gaps are so large, that the aforementioned sampling approach is not feasible: $1.3e+12$ is a rough estimate of the total number of tweets in existence, whereas a tweet ID can take any of 2^{64} values, so only about 1 in $7e+8$ tweet IDs correspond to an actual tweet.

Not only would this make sampling impossibly slow, the Twitter API's request rate limit would be quickly reached. This approach poses another problem from a theoretical perspective: the "gaps" may have an uneven distribution, thus leading to a biased sample. For example, let us suppose that there are larger gaps between high IDs: the previous sampling method will collect fewer tweets with high IDs, giving a sample that does not represent the true population very well.

Sampling over all tweets in existence is not feasible, but the Twitter API provides a way to collect live tweets. The random tweets studied in this project were collected with a python script that performs the following:

while running **do**
  Collect all live tweets for 5 seconds
  Discard retweets and tweets without images
  Sleep for a random (uniform distribution) amount of time under one hour
end while

This method ensures a less biased sample over the time period that the script is running, but I cannot collect tweets dating from before my project started.

Figure 2 – Samples collected on 5/1/2020

Figure 2 shows gaps during which no images were sampled, which means that no images were tweeted during the 5 second sampling windows occuring during these

periods. This reflects the uneven activity of Twitter over time, which makes the situation ideal: I will have fewer samples from low-activity hours, resulting in a sample that is more representative of the true population.

The other dataset needed for this project is a varied selection of target images. The ideal dataset is one that is randomly collected and unbiased, as discussed previously, except here, the true population consists of target images. This dataset is not simply a subset of the previous randomly-sampled dataset: this would result in a pitifully small dataset (in my experience, non-existent) due to the fact that target images are so rare.

Although I could not collect random past tweets, it is possible to search for past tweets with specific properties. I collected a target dataset by searching for past tweets that have certain properties (these properties will be discussed below). Note that I could not collect non-target images in the same way: the search function requires specific queries (text, hashtag, view count...) so I could not simply search for random tweets. Not all tweets are indexed or made available via the search interface (the search index has a 7-day limit) so my sampling script ran queries regularly with the aim of collecting every target image tweeted during the sampling period. Like with the previous dataset, I could not sample from past tweets but those that are collected accurately represent the true population (if not perfectly) during the sampling period.

To avoid storing tens of thousands of images, all the data that I collected consists of tweet IDs and image URLs. This way, I only download a given image when needed for processing.

3.2 The Definition of "Memorable Image"

The next aspect to consider for this project is how to define a "target image", that is, how do I decide what images should be labeled as "memorable". I previously described the collection of such a dataset by searching Twitter for tweets with certain properties, here I will describe these "properties" and how they were determined.

You may notice an instance of recursion here: the goal of the project is to find what defines "memorable image", and yet to complete this project, we first need a definition of "memorable image". For this project to have any worth, conclusions must be drawn from my initial definition of "memorable image" and not the other way around.

Therefore, I tried to define these "#unforgettable" images in an objective manner: an image is "#unforgettable" if and only if it was tweeted with the hashtag "#unforgettable". In other words, an image is a target image if the user tweeting it intends to label it as such. Why the user has such an intention, and whether the image is actually memorable to the user is outside the scope of this project.

This project now resembles a specific case of hashtag prediction: if we know what makes an image memorable, we can try to predict what images are likely to be tweeted with the hashtag #unforgettable. However, the resulting target dataset was very small, so I decided to form a list of target hashtags instead of the unique "#unforgettable". Any image tweeted with any hashtag present in this target tag list, I now considered a target image.

A rather objective way of compiling this list is to look for hashtags correlated with "#unforgettable", and there are a number of online tools that can find correlated hashtags. These tools are usually called "hashtag generators" and are used by people seeking to gain more engagement and discoverability by adding as many related hashtags as possible to their social media posts. While these tools have helped me find related terms (such as "#memories"), the hashtags generated are engineered to increase

engagement and discoverability and therefore tend to diverge significantly from the original hashtag, possibly in order to reach a wider audience. Best-hashtags.com [3] for example, says that the "best" hashtags to post alongside "#unforgettable" are "#love" and "#happy", these differ greatly from the topic studied in this project. Finding the results from these tools unsatisfactory, I attempted to find correlations and related hashtags within my own dataset with a script that does the following:

1: $T \leftarrow \#unforgettable$
2: **while** running **do**
3: Find the tag N most correlated with T
4: Add N to target tag list
5: $T \leftarrow N$
6: **end while**

I implemented a few different ways of completing line 3:

- finding the tag with the highest number of occurrences in target images (without taking into account non-target images).
- finding the tag that is present in the most target images and in the least non-target images.

I completed the second method by selecting the tag that maximises the following formula:

$$\frac{taggedTarget}{totalTarget} - \frac{taggedNonTarget}{totalNonTarget}$$

Where *taggedTarget* and *taggedNonTarget* represent the number of target tweets (and non-target tweets respectively) containing the tag, *totalTarget* and *totalNonTarget* represent the total number of target and non-target tweets respectively. Both approaches give almost identical results and have the same problem as online tools: the script provides hashtags that are vastly different from the focus of this project. For example, the script returns "#Mugabe" referring to Zimbabwean revolutionary Robert Mugabe. To complete line 3, there are also statistical tools such as the chi-squared statistic or phi coefficient that can be used, but it seems clear that this type of script will not return compelling results.

In similar projects, I have seen authors subjectively deciding on related terms in addition to finding correlated terms; the former is how I determined my target-hashtag list. More precisely, I found related terms and synonyms (some of which came from online tools) and included them in my target-hashtag list if their meaning corresponds to the focus of this project and if they have a significant presence on Twitter (excluding tags such as #instamemories found only on Instagram).

After broadening my definition of a target image, the data I could collect remained rather limited. After all, there are only so many "unforgettable" moments to post on social media! In fact, out of the nearly 20 000 randomly sampled images studied in this project, only a handful happened to also be target images, and the 400 or so target images studied were collected by the searching script. Therefore it became obvious that building an image classifier for this project was futile: simply predicting "non-target" for every randomly-tweeted image will result in an accuracy of over 99.99%.

3.3 Feature Extraction and Data Analysis

Before starting data analysis, it can be useful to view some data without analysis tools. For this project, I found that looking at some collected images was not very informative, but it did give me ideas on which graphical features to extract. I also created image "averages" from my datasets in an attempt to notice any obvious differences between target and non-target images. These averages were made in two different ways:

- Padded average: adding black padding so that all images have the same dimensions (no scaling involved); adding all these padded images together to produce a sum matrix, and finally dividing the sum matrix by the number of images.
- Pixel by pixel average: creating a sum matrix in the same way, and then dividing each element in the sum matrix by the number of pixels contributing towards that element.

In the first case the corners of the image average will be darker as most images consist of black padding in that area; whereas in the second case the black padding is not taken into account in the average.

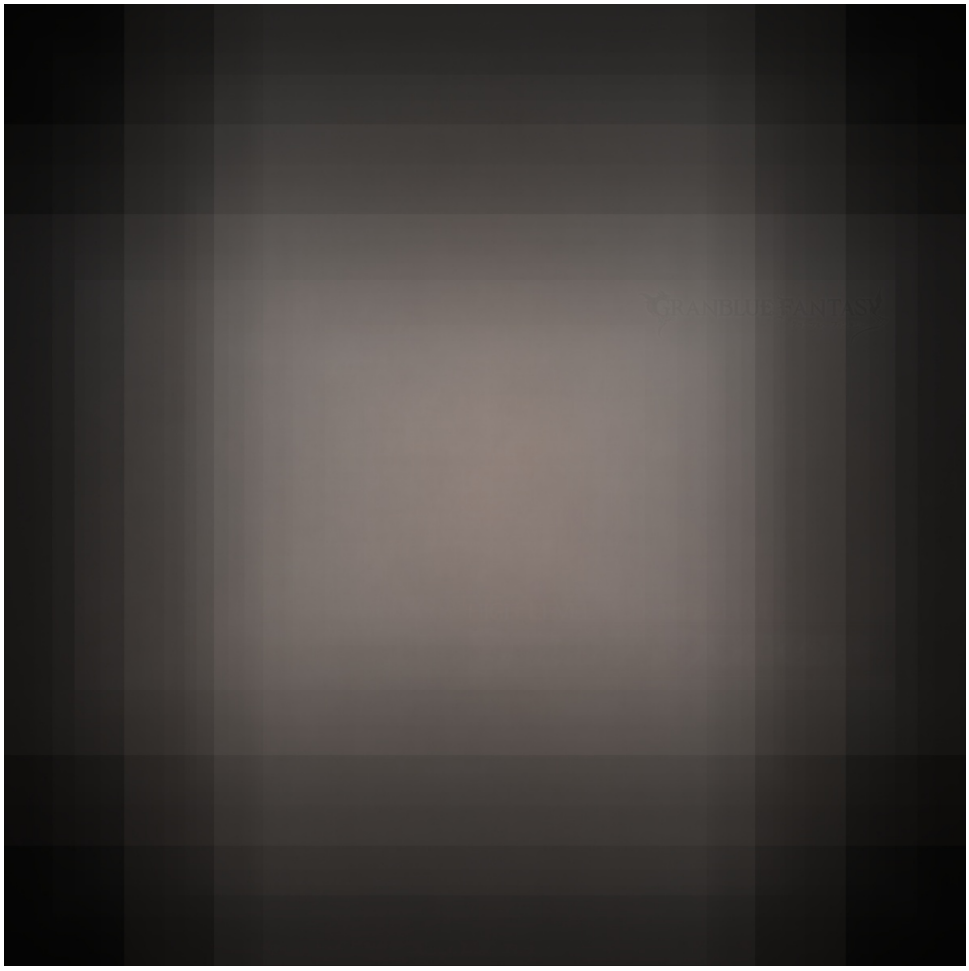

Figure 3 – Non-target general average

Figure 4 – Non-target pixel-by-pixel average

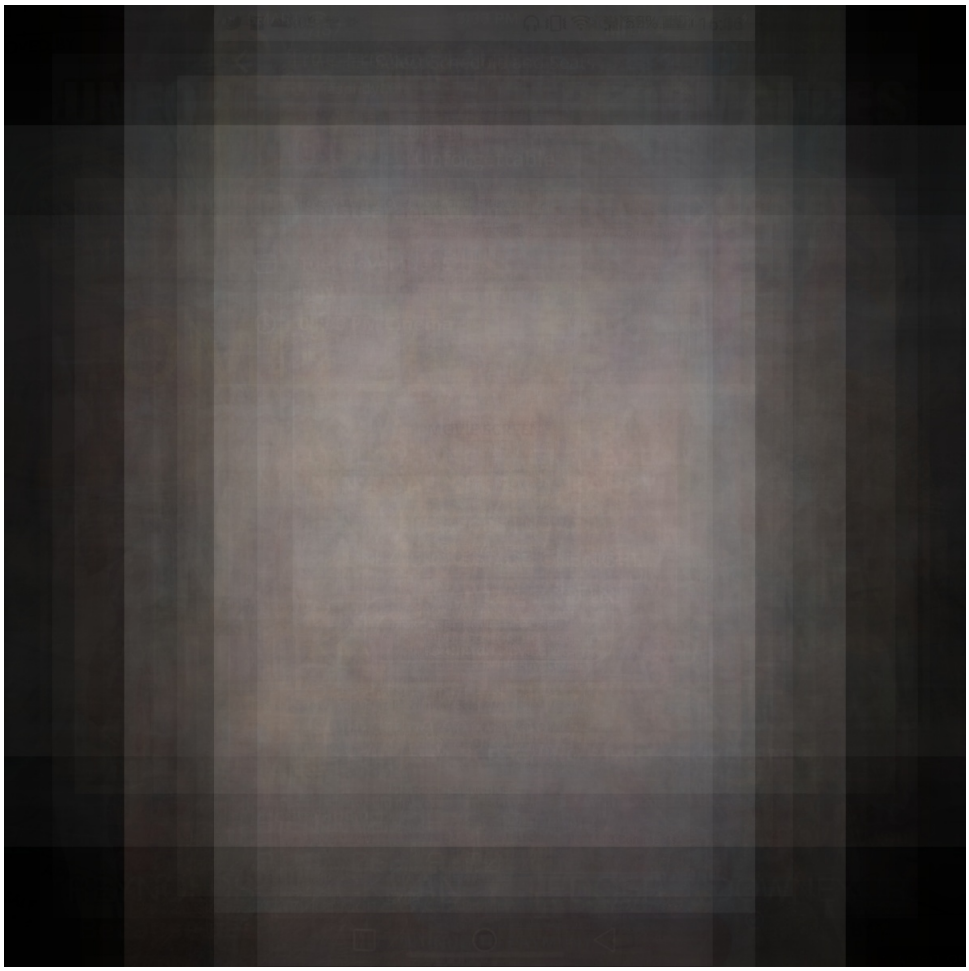

Figure 5 – Target general average

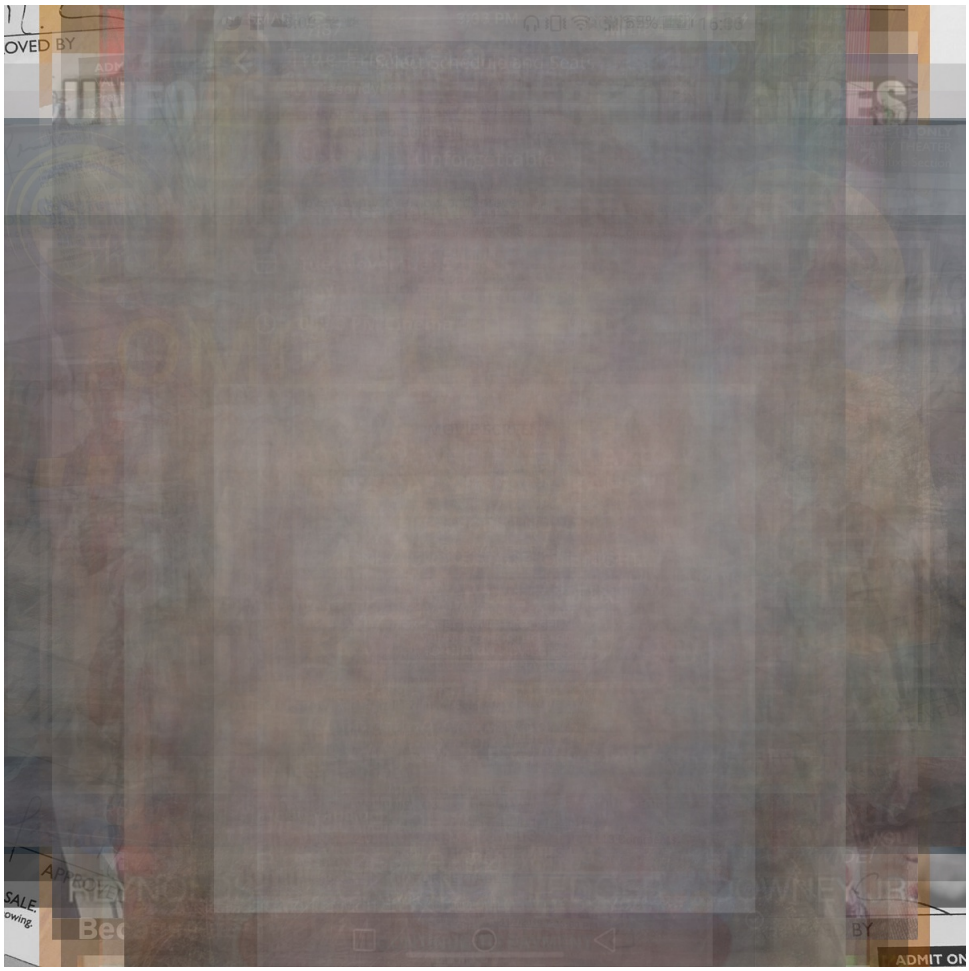

Figure 6 – Target pixel-by-pixel average

As you might expect, these averages do not reveal any obvious difference between target and non-target images. A few interesting things can be noticed however:

- The averages show clear lines representing "clusters" of image sizes. This means that there are a few common resolutions for tweeted images.
- In the non-target average, text is visible. This means that similar images with the exact same dimensions are tweeted in large quantities, with the same text and in the same position. This can only be observed for non-target images, indicating that text may be a differentiating characteristic between target and non-target images.

I used ImageMagick [4] to extract graphical features such as brightness, contrast, resolution, etc..., and used facedetect [5] and PyTesseract [6] to count the number of faces and read any text present in the images. When I started feature extraction, I found many images that my data processing scripts were unable to download. This happened because many tweets were deleted, and their images were no longer hosted on Twitter.

The goal here is to find which of these features reveal a significant difference between the two classes of images. I looked for correlations between features and class, and tried training a linear classifier (with Scikit-learn) but as explained previously, this approach did not yield any compelling results. However, running statistical tests on every feature did reveal some differences between non-target and target images.

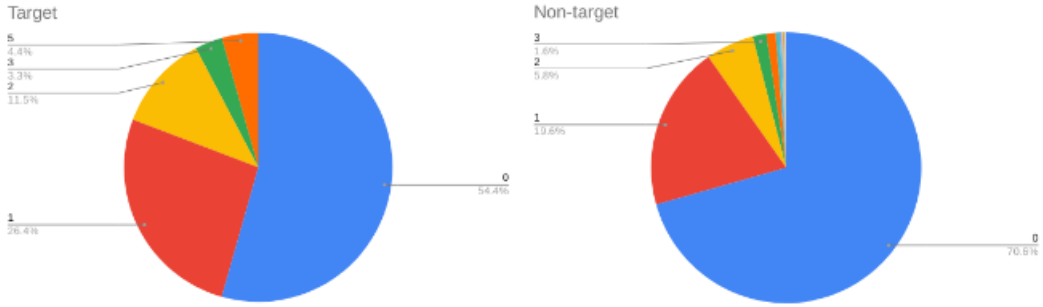

Figure 7 – Distribution of the number of faces present in target and non-target images

The number of faces constitutes a statistically significant difference: a t-test gives a p-value of 0.006,which is well below the commonly-used threshold of 0.05. We can conclude that "memorable" tweeted images contain more faces on average (around 0.8 faces per image) than their counterparts (around 0.5 faces per image). One of the assumptions that needs to hold in order for a t-test to be valid is that the data should follow a normal distribution. It is easy to see that the data in question does not follow a normal distribution, calling into question the validity of a t-test. The t-test is based on the means of the two groups, and according to the central limit theorem, the distribution of these means (in repeated sampling) converges to a normal distribution, irrespective of the distribution of faces in the studied population. Since I have a large amount of data, the result of a t-test can still be considered relevant.

When analysing text extracted from tweeted images (using Google's Tesseract OCR), I found very strange results: 23% of randomly-sampled images contain exactly 2 characters of text. Such a large proportion indicates that I should have noticed images with exactly two characters after looking at a few dozen collected images. However, this was not the case. After looking into this issue, I found that of the scripts

involved in extracting and processing text data, some did not work together correctly even though they were individually correct. To solve this problem, I wrote a single new script that handles the data from start to finish and obtained coherent results after reprocessing all studied images.

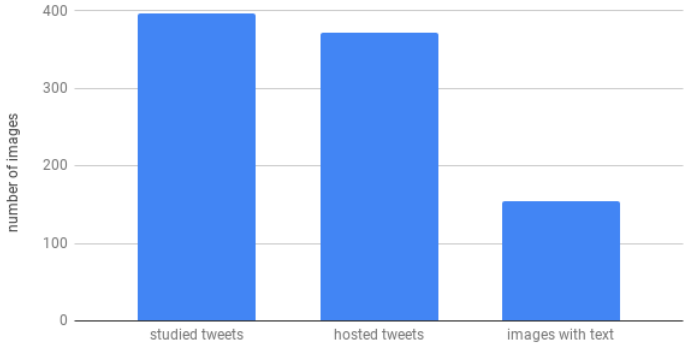

Figure 8 – Number of target images studied, hosted on Twitter, and containing text

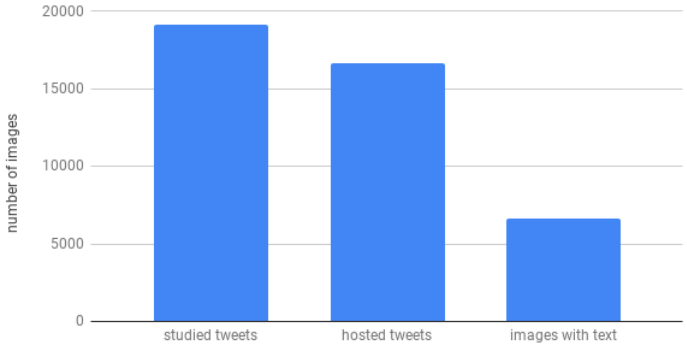

Figure 9 – Number of non-target images studied, hosted on Twitter, and containing text

The figures above show similar proportions of images with text, but the proportion of images hosted on Twitter at the time of processing is lower in the non-target dataset. A chi-squared test gives a p-value of $6e-5$, revealing that this difference in proportions is significant. For this test I use a 2×2 contingency table, the two samples being "target" and "non-target" and the two categories being "hosted tweets" and "deleted tweets".

Since target and non-target images were collected over the same time period, we can conclude that target images are hosted on Twitter for longer, and that in a sense, "memorable" images are in fact less quickly "forgotten" by Twitter. This difference is not between the images themselves but between metadata attached to the images, so it is not directly relevant for this project, although it is an interesting result. By focusing only on images containing text, I found other features that show a significant difference between target and non-target images. After extracting the text from images, I proceeded to look at features like the number of characters, new lines, etc...

I measured text "density", which is an indication of how words and characters are spaced apart, and found that target images seem to have a higher text density on average. Indeed, a t-test for this feature gave a p-value of 0.016. I also measured the minimum Levenshtein distance (MLD) of any word in an image to any hashtag in my target tag list. If an image reads "an unforgettable journey", the resulting MLD will

be 0. An MLD of 1 indicates that the image contains a word just one character off from a target hashtag (“memorale” for example, where the “b” is missing).

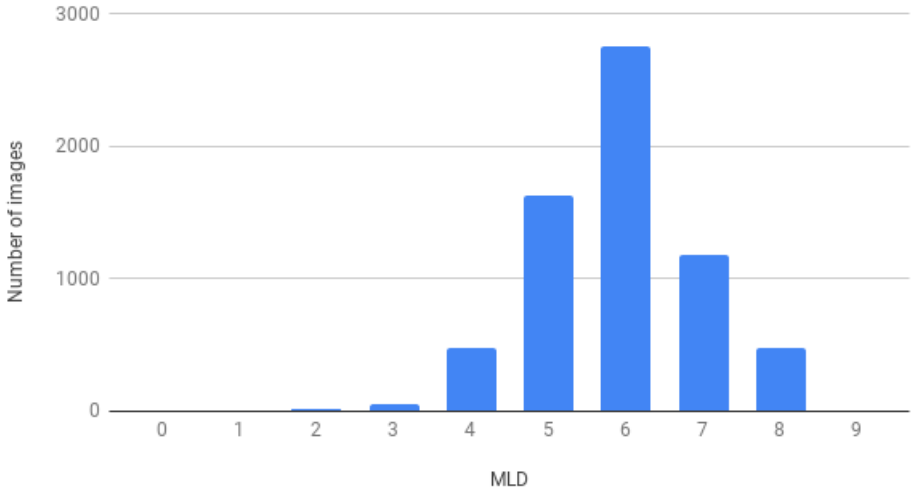

Figure 10 – MLD distribution of non-target images

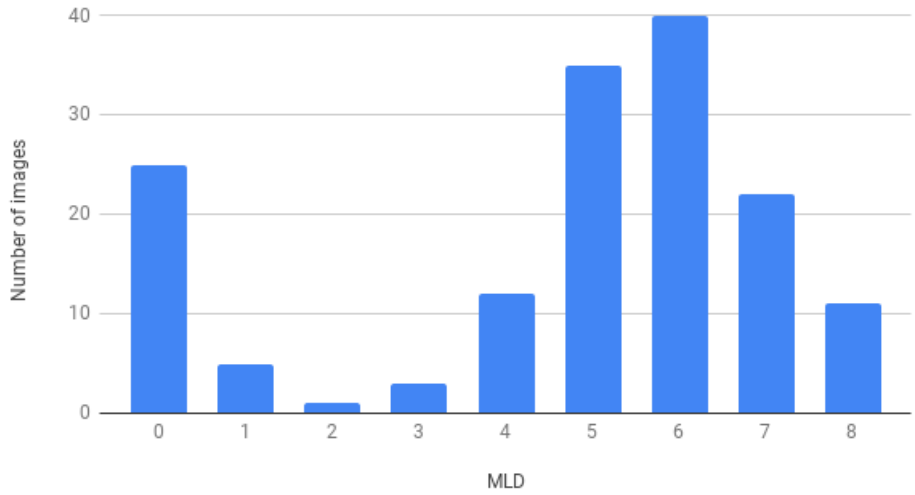

Figure 11 – MLD distribution of target images

We can see a truly large difference between the distributions of target and non-target images. Text in target images unquestionably contain more target tags than text in non-target images (recall this regards text present in the graphical image itself, not in the tweet). This result leaves no room for doubt, as a chi-square test gives a p-value on the order of 10^{-38}. The categories I use for the chi-square test are $MLD \leq 1$ and $MLD > 1$.

4 Conclusions

This kind of project in general can help reveal the intricacies of online interactions, and my project specifically provides hints that give rise to some interesting questions and hypotheses.

In order to fight platform manipulation, Twitter automatically removes about 10 "fake" accounts every second [7], and I suspect this to be the main cause of tweet deletion observed in my dataset. If this is indeed the case, the difference in image lifespan can be interpreted as a difference in how target tweets tend to be less "fake" compared to their counterparts. Alternatively, using "memorable" tags might be a way for platform manipulators to avoid detection. I've found in this project that images that people like to label as memorable tend to involve more people than others, which may indicate that important memories tend to involve people.

During this project, I have found that a large proportion of target images contain target words, and in many cases, this is due to the fact that the target image is a screenshot of a different target tweet. The tweets already contain a target hashtag, so why provide redundant information? In fact, why tweet a screenshot of another tweet when the retweet feature exists? This is undoubtedly the most puzzling result obtained from this project and I cannot fathom any realistic hypothesis to explain it.

This does indicate that users may be interested in a tool incorporated into their social media platform that reads text off images posted and automatically adds the hashtags already on the image. Nearly 15% of images I have collected that are tagged "memorable" already contain words such as unforgettable","memories" (in the form of a caption for example), so there is evidence to suggest that a demand exists for such a tool.

5 Evaluation

After the completion of my project, I proceeded to evaluate the certainty and coherence of my results, which has led me to correct mistakes in my reasoning and to consider in hindsight what I could have done better.

5.1 Evaluation of my Procedure

The more complex the analysis, the more unit- and integration testing becomes necessary. As mentioned previously, I had a bug in my text processing scripts, and I had to simplify my analysis process in order to fix it. In hindsight, I think it would have been better to implement simple, one-off scripts used for a single analysis instead of having intermediary processing scripts that are used for multiple analyses.

My random sampling can be seen as a point process similar to the Poisson point process, except instead of an event following an exponential distribution (radioactive decay for example), the sampling follows a uniform distribution. I'm interested in

knowing if the "points" at which sampling is attempted are uniformly distributed over the entire duration that the sampling script is run. The sampling "points" are in fact 5-second intervals so it is not quite a point process. However, 5 seconds is negligibly small compared to the entire sampling interval, so we can consider them to be points.

> If all intervals of the same length on the distribution's support have identical probability distribution, then the distribution is uniform.

The above is true for my sampling, perhaps with the exception of the start of the process: the probability distribution of having at least one sample point in the first hour is uniform (by design), whereas in subsequent hours I am not confident that the distribution is the same (more than one "random delay" needs to be taken into consideration).

For any other two intervals of equal length (far from the start of the process), it is clear that the probability of sampling once is the same in both intervals, the probability of sampling twice is the same in both intervals... etc (the distributions are identical). We can conclude that, excluding the start, my sampling achieves maximum entropy with regards to time.

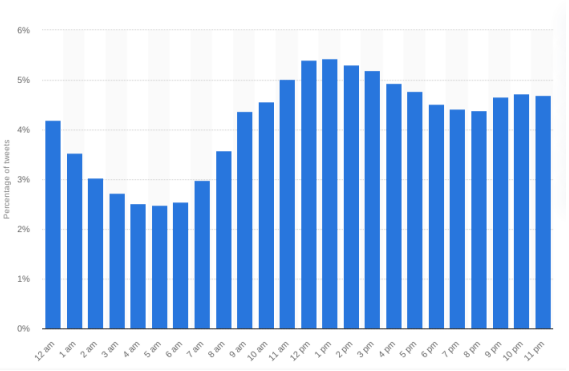

Figure 12 – Twitter activity over a 24h period

According to www.statista.com, the above graph represents the distribution of tweets in a day (measured in 2014). Tweeted images will also have some distribution in function of time. Since my sampling points are uniformly distributed, the sample I have collected will follow the same distribution thus accurately representing the true population.

Were I to redo this project, I would set the random intervals between sample attempts to follow an exponential distribution so that the resulting sampling would be a Poisson point process. The Poisson point process is well studied and documented, and it has been proven to lead to uniformly distributed points.

5.2 Evaluation of my Results

To ensure that my conclusions are coherent, I will estimate the underlying distributions for a few features using maximum likelihood estimation. Of the statistically significant features, I will only explore the maximum likelihood estimation of the text density feature, as it is the only continuous variable. The other variables are categorical, so maximum likelihood estimation will simply produce a scaled version of the observed distribution (see MLD distribution of non-target images for example).

In fact, during the development of the project, I treated the number of faces (face count) as a continuous variable, but after evaluation I have come to the conclusion that it is best to view face count as categorical data.

I use the mean pixel value as a measure of brightness of an image; this was a graphical feature for which no statistically significant difference was found between target and non-target images. Brightness takes a value of 0 or greater, so I will use the gamma distribution as a model for maximum likelihood estimation. The log likelihood function for the gamma distribution is:

$$\ell(\alpha, \beta) = -n\alpha \ln \beta - n \ln \Gamma(\alpha) - \frac{1}{\beta} \sum_{i=1}^{n} x_i + (\alpha - 1) \sum_{i=1}^{n} \ln x_i$$

Figure 13 – The gamma distribution's log likelihood function

I use the non-linear minimization function in R (nlm) on the negative log likelihood function (with my brightness data in the place of x_i), and hence find the maximum of the likelihood function. I obtain the following maximum likelihood estimates for image brightness:

	α	β
Target brightness	5.90	21.16
Non target brightness	4.59	27.48

Figure 14 – Maximum likelihood estimates for image brightness

In other words, the most likely distribution for target brightness is (5.90, 21.16) and non-target brightness most likely follows (4.59, 27.48).

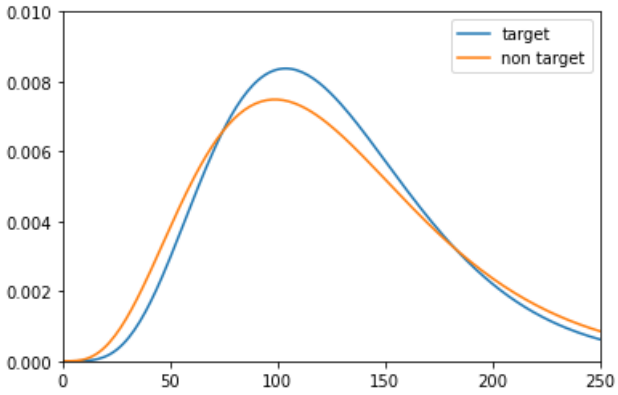

Figure 15 – MLE distributions of brightness in target and non-target images

Text density takes a value between 0 and 1, so I will use the beta distribution as a model. I use the same procedure in R except this time with the log likelihood function for the beta distribution.

	α	β
Target text density	12.62	3.98
Non target text density	16.74	4.72

Figure 16 – Maximum likelihood estimates for text density

The most likely distribution for target text density is B(16.74, 4.72), and the most likely distribution for non-target density is B(12.62, 3.98)

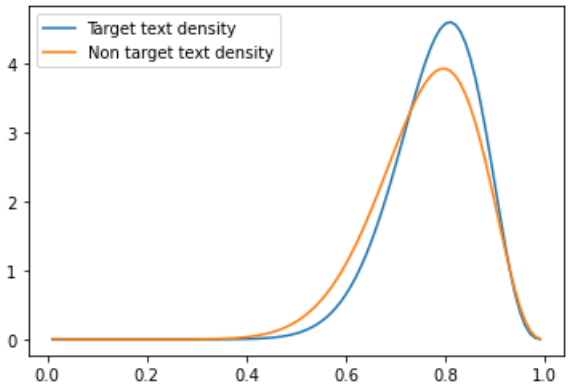

Figure 17 – MLE distributions of text density in target and non-target images

In both cases the estimated distributions are very similar, so what makes the difference in text density statistically significant when the difference in brightness is not?

	Brightness	Text density
Target mean	124.8	.78
Non Target mean	130.1	.76
P-value from t-test	0.8160	0.0163

Figure 18 – Brightness and text density comparison

For the t-tests I have carried out, the p-value shows the probability of obtaining my target sample, assuming that the true distribution is that of my non-target sample. A t-test is limited in that it does not show how likely it is for both samples to originate from the same population with the same mean. You could conceive of a situation where sample A has a mean of 0, sample B has a mean of 1, and it is unlikely that B originates from a population of mean 0; so you conclude that A and B originate from different populations. However, It may be possible that both A and B originate from the same population of mean 0.5.

I had previously concluded that text density reveals a statistically significant difference between target and non-target images, but after seeing the similarity in

MLE distributions, I decided to cross-validate my results using a statistical tool more advanced than a t-test. BEST [8] is a statistical tool that gives the probability distribution of the difference of means, allowing me to see how likely it is that target and non-target populations have the same mean.

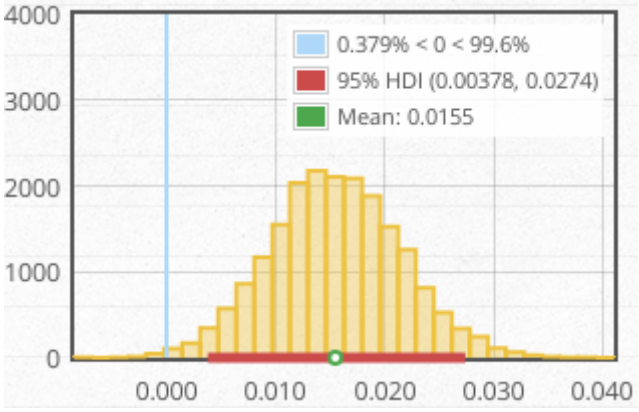

Figure 19 – Probability distribution of the difference of means of text density

I used an online implementation of BEST [9] to produce the above image. We can see that despite the t-test giving a p-value of 0.016, BEST shows that it is actually not very unlikely that there is a negligible difference between means. The red bar shows that there is a 95% likelihood that the difference in means is between 0.00378 and 0.0274. Additionally, we can see that there is a 50% chance that the difference in means is less than 0.0155 (about a 2% difference). I must concede that text density showing a difference between target and non-target images is a rather uncertain result.

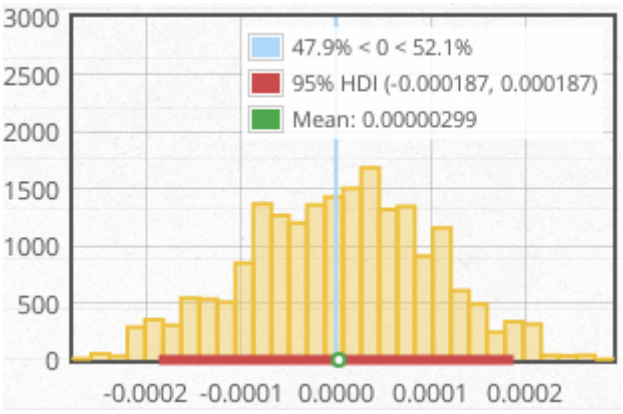

Figure 20 – Probability distribution of the difference of means of face count

Evaluating face count with BEST leads to the conclusion that face count does not constitute a statistically significant difference between the two classes of images. This is a very surprising conclusion, as the observed distributions of face count show obvious differences (see figure 7). This led me to realize that I had been treating face count as a continuous feature when it should in fact be considered categorical. Both a t-test and BEST are exclusively for continuous data, so it makes sense that they are not suitable for testing face count.

Instead of a continuous variable, face count can be seen as being in one of the

following 5 categories: 0 faces, 1 face, 2 faces, 3 faces, 4 or more faces. I have set "4 or more faces" as the final category since the number of samples collected with more than 4 faces is negligibly small. In this way, an appropriate statistical test is no longer a t-test but instead a 5 category chi-squared test; this gives a p-value of 2.9^{-5}.

I am inclined to conclude that I had used the wrong statistical test for face count and that there is in fact a statistically significant difference between the number of faces in target images and the number of faces in non-target images.

5.3 Project Limits and Scope

The Twitter API search function searches against a sampling of recent Tweets published in the past 7 days. In other words, there are tweets within the past 7 days that were never indexed for searching, only a subset is. Unfortunately, I have not been able to find how this sample is made and can only hope that it accurately represents the true population. The validity of my results depends on the search API returning an unbiased set of "memorable" images.

The scope of my project is an obvious limitation. Twitter was the only platform studied, and my samples are taken from a limited timeframe.

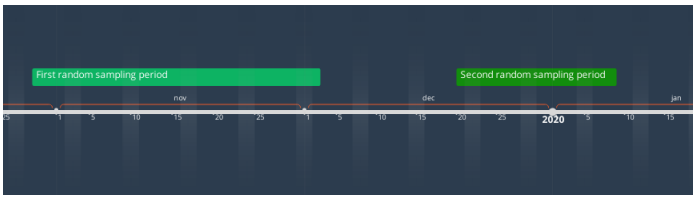

Figure 21 – Sampling period timeline

The dataset I studied in this project is a uniform mix of tweets from different sampling periods, which only covers about 2 months. The way I have defined "memorable" is also not all inclusive: the subjectively chosen group of hashtags are all in English! It is important to remember that my project is limited to the study of users who use English hashtags, when in fact only 20% of the tweets I have collected are primarily in English. Furthermore, many non-English tweets contain English hashtags, and it is not uncommon for a word to change meaning when incorporated into a foreign language.

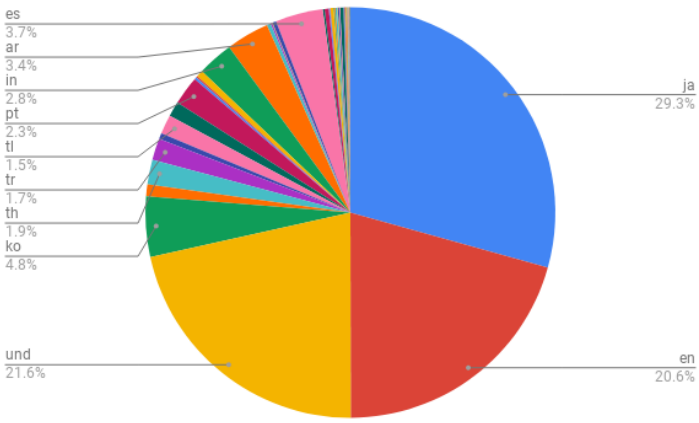

Figure 22 – Language distribution of collected tweets

In my opinion, the issue that introduces the most uncertainty is the large quantity of tweeted images that I was unable to process due to the fact that Twitter was no longer hosting said images. This effectively skews my dataset; there are so many deleted images that incorporating them into my study may lead to significantly different conclusions. Therefore, my project can only be said to be a study of tweeted images with a "lifespan" of more than a few months.

6 Directions for Continuation

As I have explained, my project as it is now has several limitations. If this project were to be expanded on, these limitations could be avoided and different applications as well as broader results would ensue.

The most straightforward improvement to this project would be expanding its scope: compiling datasets that are greater in quantity and variety (from different platforms and time periods), and creating a larger, more comprehensive and multilingual target hashtag list. In addition to enlarging the quantity of data studied, there are always more graphical features that can be extracted and analysed. Using some form of object recognition (with Google's vision API for example) would produce new features (such as the presence and position of an object) that could reveal new differences.

There is room for more text analysis as well: emotion recognition, language, coordinates of text, and even things like the color, font and size of text would produce interesting features in my opinion.

An interesting direction for this project would be to collect survey data, asking users what they intend to communicate by tagging posts as "memorable", or what makes such images actually memorable to the user. I would also be interested in asking users why it is preferred to have a memorable hashtag both in the image and in the tweet. To help elucidate this matter, I would suggest presenting varied tweets to users, some of which have redundant hashtags, and some of which don't. I would then ask them to select their preferred (or most attention-grabbing) tweets. I suspect this redundant information improves engagement, and there are always available methods to measure engagement for any platform (views, likes, shares...).

Another direction in which to expand this project would be to incorporate the study of other groups of hashtags. For example, it could be interesting to study an "opposite" group of images that are labeled with hashtags of opposing meaning to the original "memorable" group of hashtags.

Since a large proportion of tweeted images contain faces, I think it would be a good idea to focus on faces as a sub-problem, in the same way I focused on text for part of my project. The question would then become: what are the differences between "memorable" images with faces and "non memorable" images with faces?. There are a number of tools that can be used to evaluate facial expressions in images, and I would expect to see stronger emotion in memorable images.

One could also transform the collected data by aligning each image around the face, resizing, shifting and rotating until each image is perfectly centered around the face. Following this, results could be found by studying the vicinity of the face, and how other objects are positioned relative to the face. Of course, it would be necessary to devise a different procedure for images that contain multiple faces.

The restriction to graphical information in this project is a rather arbitrary one. In fact, the similar projects I have seen are not restricted in this manner. A good

direction for this project could be to study the tweets containing the images as well as the accounts that post them, instead of considering only the image itself.

In the early stages of my project, I had actually considered changing the focus of my study from "memorable" images to images tweeted with any hashtag at all. The same analysis scripts, statistical tests and procedures could be reused to this end, simply by changing the input datasets. I have found that about one fifth of my randomly sampled images are tweeted with a hashtag, so building a classifier would be interesting in this case.

In the end, the purpose of this project is to provide a foundation to build an image detector and shed light on the less studied problem of detection in general. To this end, I would use a cascade architecture: layers of filters that remove non-target images. It is difficult to find memorable images, but there certainly are classes of images that are known to be non-target images and are easy to identify.

For example, the image averages (see figure 4) revealed that a large quantity of images were being tweeted with a certain media property's name present in the exact same position (upper right hand corner). As such, a task for the first filter in this cascade architecture would be removing these easily identifiable commercial images. I expect that other easily identifiable non-target images exist and can be removed by the first few filters.

After these filters, I would put each image in a category (people, natural scene, advertisement...) allowing for a different "memorable" detector in each category. In this way, the detector can be fine tuned for specific images with predetermined characteristics. I am not certain what implementation of an image classifier is ideal for the task of putting images into said categories; it would depend on what the categories are. The classifier could be in the form of a neural network that takes every pixel as input, or a different method could be used (SVM for example); the classifier may work better with hand-crafted features rather than taking an entire image as input, or perhaps a combination of both would prove to be ideal.

After the classifier completes its task, a different detector can be used for each class of image. Each detector would be built in a similar manner to a classifier, except the error function used would not represent the number of misidentified images. Instead, the goal would be to catch almost every target image, while the number of false positives is not as important (just having fewer false positives than true positives would already be quite good). This could be implemented in the form of a weighted error function, where false negatives are weighted many times heavier than false positives.

The result I found regarding MLD seems to indicate that if an image is found to contain a memorable word, it should have a much higher chance of being a target image than any other image. I would consider incorporating this into the first filters as a form of "white-list" condition: if an image contains a memorable word it should not be rejected by the initial filters. As the output of the detector, one should obtain a subset of the input that contains almost all target images, but that is multiple orders of magnitude smaller than the input set.

My third year project was not extensive enough to include the creation of a memorable hashtag detector, but I hope to see complete projects regarding hashtag detection in the future.

References

[1] https://www.cs.cmu.edu/~efros/courses/LBMV07/Papers/viola-cvpr-01.pd

[2] https://developer.Twitter.com/en/docs/basics/Twitter-ids

[3] http://best-hashtags.com/hashtag/unforgettable/

[4] https://imagemagick.org/index.php

[5] https://www.thregr.org/~wavexx/software/facedetect/

[6] https://pypi.org/project/pytesseract/

[7] https://www.fastcompany.com/90331696/Twitter-is-automatically-removing-about-10-accounts-every-second

[8] https://www.semanticscholar.org/paper/Bayesian-estimation-supersedes-the-t-test.-Kruschke/dea60927efbd1f284b4132eae3461ea7ce0fb62a

[9] http://sumsar.net/best_online/