Meme Search

Abhishek Shah
ans556@nyu.edu

Mayank Singh
ms8599@nyu.edu

1 Introduction

Internet "memes" have become a very popular mode of communication in the past few years. The ability to put across a message along with some humor is appealing to a lot of people. Though technically memes mean a lot more, we'll mostly mean image macros when we use the word "meme". Image macros are some "viral" images with text superimposed over them.

Our report deals with building a search engine for these meme images. This could be useful for people looking to communicate on social media platforms such as "facebook.com", "reddit.com", etc. via such images. Overall, the task boils down to fetching the appropriate meme from a database pertaining to a given text query. The main challenges that we deal with in this project are extracting the text from a meme image and identifying the visual content of the image. The final search engine is situated here.

The rest of the report is organized as follows. We describe what methodology we used for crawling meme images in section 2. Sections 3 and 4 explain how we dealt with the challenges mentioned in the previous paragraph. Section 5 gives a full description of our search engine's architecture. In section 6, we discuss the kind of evaluations we performed on the search engine. Finally, in section 7, we present some ideas on what more can be done to improve this project.

2 Crawling

The first task for any search engine is to crawl and collect webpages relevant to the requirements of that search engine. Since we are building a meme search engine, we only need to crawl for meme images. One vital problem related to this task is that it is difficult to tell whether an image is a meme. Though this is an interesting problem, we decided to focus more on the other tasks in this project. Due to this reason, we crawl only those sites where the crawled images are bound to be memes. We do discuss a few techniques that we could have explored to solve this problem in the future work section.

A few websites that contain only meme images are the meme-specific subsections of "reddit.com" such as "\r\memes", "\r\meme" and "\r\AdviceAnimals". We crawled these sections (called subreddits) of the reddit website using the reddit API from the top and new subsections to assure that we have both the trending

and top memes. We crawled a total of 2517 images through this process. Some of the images were corrupt so we could index only 2252 images in the end.

3 Text Extraction

The superimposed text found in image memes often contains a lot of signal that might be useful to some user. To extract this text, we started off by exploring tesseract OCR [4], a readily available open-source OCR software. However, the results were not that impressive. One reason we speculate could be that tesseract doesn't generalize well to different font sizes, different colors, etc. A particular suggestion that we found while researching this issue was to apply stroke width transform (SWT) over an image before passing it to tesseract. However, due to time constraints, we couldn't explore this advice.

Since tesseract wasn't an option, we explored a few online vision APIs. Out of these APIs, we chose to use Microsoft Vision API [12]. Apart from being very accurate, it was also a cheaper alternative to other OCR APIs. We have used this API for extracting text from the meme images in our final work.

4 Identifying Meme Categories

The texts extracted from images provide a lot of information about them. However, certain information can only be deciphered from the visual content of an image. For example, consider a "bad luck brian" meme which has some text superimposed over it in figure 1. The extracted text doesn't inform us that this is a "bad luck brian" meme. To get access to this visual content, we need to employ an image recognizer. This image recognizer should be able to identify the type of meme image being used.

We have had a few excellent image recognizers invented in the past 4 years due to recent advances in neural networks and GPUs as evident in the "ImageNet Large Scale Visual Recognition Challenge"[1] (ILSVRC) competition. We use one of them - the "VGGNet"[2] for identifying the visual content of the meme images. Apart from its impressive accuracy in the ILSVRC, another reason for using the VGGNet is that the authors have released the learned weights of their network which relieves us from the task of correctly training this huge network (19 layers deep).

The original paper shows that VGGNet performs really well on the "Imagenet" dataset but it has been trained to work particularly on the "Imagenet"'s 1000 classes. However, the features generated by these deep neural networks such as the VGGNet have been shown to generalize well to other datasets as well [2]. The trick is to use the output of the layers before the final softmax regression (multinomial logistic regression) layer to generate features for the new dataset. Using these features, we can perform a new softmax regression on the categories of our new dataset. We employ this method for our task.

Here, we describe the method in detail. We crawled the popular memes section

of "imgflip.com" to gain blank meme images of a total of 780 meme categories. Then, we used a meme generator to generate 50 images per meme category with random text written at the top and bottom of the meme image. This served as the dataset to train our meme classifier. The actual classifier was built on the same principles as mentioned in the previous paragraph. We split the dataset generated above into training, validation and testing sets in a ratio of 60:20:20. We used the adam [3] optimizer to minimize the softmax loss function. After training for a few epochs over the training dataset, we got an accuracy of over 98% on test dataset. This was pretty good for our purposes. All of the machine learning work mentioned above was done using the torch package [11].

Even though we had a very good meme recognizer with us, it was limited by the fact that many of the crawled images from the web did not belong to any of the 780 categories. To solve this problem, we only use the output of the classifier for an image if the confidence of the classifier over its top prediction is over 98%. Though a more systematic approach could have been used here, we found this simple heuristic to be very effective for most memes. At the end of this process, we either get some label or an empty string for each candidate image from the crawled image data.

Figure 1: A sample "Bad Luck Brian" meme.

5 Search Engine Architecture

We describe the complete search engine pipeline in this section. After crawling the meme images from meme-related subreddits, we pass these images through the text extraction and meme classification processes to retrieve the superimposed text and the meme type. We create an index using Apache Lucene [8] with image name as the key; and the superimposed text and the meme type as the values. Note that, we concatenated the superimposed text and the text denoting the meme type together to use as the value for the lucene index. For scoring documents with respect to a query, we use the default scoring function available in lucene which is a combination of the vector space and boolean models of information retrieval. The concatenation made sense because this scoring

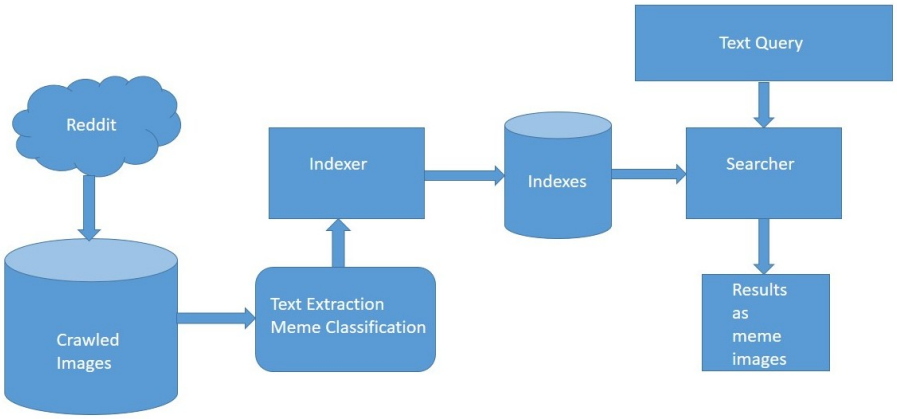


Figure 2: The full search engine pipeline

function is in the end a bag-of-words model and the ordering of words don't matter. Also, note that while displaying the results for a particular query, we have limited the number of displayed search results to 10.

Results for query interesting

Figure 3: Sample results from the search engine. More examples can be found in the appendix.

6 Evaluation

The relevance of a meme image may differ from person to person based on their information needs. For some, their information need maybe fulfilled if some part of the superimposed text matches their query; for others, it maybe fulfilled if they get all the memes of a certain category, and a few more may have other myriad needs. We evaluate our search engine by doing a personal evaluation with the help of an unbiased judge who is conversant in memes.

We select MAP(Mean Average Precision)[9] as an evaluation metric since we don't have a set of all relevant documents pertaining to a given query. MAP is

calculated as follows:

$$MAP = \frac{1}{|Q|} \sum_{j=1}^{|Q|} \frac{1}{5} \sum_{k=1}^{5} Precision(R_{jk}) \tag{1}$$

where $|Q|$ is number of queries, R_{jk} is the k^{th} ranked retrieved document (image in our case) from engine for jth query. The number 5 in the formula indicates that we limit the evaluated search results to the top 5 displayed images.

We selected a set of 7 queries to evaluate our search engine. Some of these are taxonomic in nature in that they are categories of a very broad class of things e.g. cats, dog, fish, etc. For queries such as "cat", "spiderman" and "pokemon" (from our actual query set), the user may be more interested in matching visual content. We also selected a few queries that are abstract in nature e.g. "good luck", "interesting", "success", etc. For such queries, the information need may differ from person to person. The user might expect a meme image with the query text in the image or some image signifying the abstract concept. The queries selected for evaluation and their average precision on the first five search results have been shown in Table 1.

The search results related to these seven queries have been listed in the ap-

Table 1: Queries and Average Precision

Query	AP@5
cat	0.45
pokemon	1
success	1
one does not simply	1
good luck	0.4
girlfriend	1
spiderman	1
MAP	**0.83**

pendix section. Here, we analyze these results. For the query 'good luck', the search engine should have retrieved memes with some quotes wishing good luck or images with inspiring visual content. But it retrieved certain images of the meme type 'bad luck brian' (because it contained the word luck) and also images with the word 'good' contained in the superimposed text. For the query 'cat', the search engine retrieved the image of a man speaking something about cats. This particular image was found to be irrelevant by our evaluator as it didn't have any cat in the image. We also found a case of false negative for the query 'pokemon'. An image of 'team rocket' (an entity related to the pokemon franchise) wasn't retrieved for this query. Further, for a lot of queries entered by the evaluator, we did not get any results. Most of these problems arise due to the small amount of crawled data that we use as the backbone of our search engine. We hope these results will improve with an increase in the amount of this data.

We also evaluated the performance of the Microsoft Computer Vision API for

our task. We generated some meme images by superimposing text on some standard meme templates (similar to the dataset generation process of the meme classifier) and used the OCR API to extract text from these images. We built two search engines to help with the evaluation. The first contained images indexed with texts retrieved from the API and the second contained the same images indexed with the original texts. The results of the second search engine were used as the gold standard. Here are the precision, recall and F-1 scores for the first 10 results of the first search engine for the set of 5 queries which can be seen in Table 2.

Table 2: Queries and Precision

Query	Precision	Recall	F-1
anime Kannagi	1	0.88	0.93
Avatar	1	1	1
iphone	1	0.69	0.82
cooking	1	0.78	0.78
taylor swift	1	0.55	0.71

The results indicate that Microsoft's OCR API gives a good precision score and a decent recall. The recall goes down for queries where there are chances of confusing between similar characters such as when 'o' is detected as '0' in the word 'cooking'. Also phrases like "taylor swift" were not detected at all for certain images.

7 Future Work and Discussion

There are many areas in which this project can be improved by more work. Here, we mention a few of them.

The crawling part of our search engine depends on sites that rarely have images other than memes. This isn't true crawling of the web. To actually crawl the web, we need a meme detector that tells a meme image from a non-meme image. We have a few ideas on how this could be built. We can use the same classifier we trained earlier for meme classification. If the classifier generates some labels, we accept it as a meme otherwise we reject it. However, this classifier is limited by the meme categories available at the external site that we crawled for meme categories. New memes and memes not cataloged by these external sites will not detected by such a detector.

To solve the above problem, we would need to build a live meme image tracker that informs of meme image categories that are getting viral. There have been attempts made by [5] in this regard which were specifically aimed at text memes from news blogs. They adopt a mutational variant of phrases model to detect how over the time information flows in the form of text. A similar kind of model for image memes can be considered for our task.

We would also like to improve on the text extraction front as there were certain texts that were not detected at all by Microsoft's OCR API. In this regard, we

would be interested in exploring the new advances in the field of text-spotting (which is a more general form of OCR and can identify text from natural images) such as [7].

References

[1] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg and Li Fei-Fei., *ImageNet Large Scale Visual Recognition Challenge.*, arXiv:1409.0575, 2014

[2] Karen Simonyan and Andrew Zisserman, *Very Deep Convolutional Networks for Large-Scale Image Recognition*, arXiv:1409.1556, 2014

[3] Diederik P. Kingma and Jimmy Ba, *Adam: A Method for Stochastic Optimization*, arXiv:1412.6980, 2014

[4] Smith, Ray, *An overview of the Tesseract OCR engine.*, icdar. IEEE, 2007

[5] Leskovec, Jure, Lars Backstrom, and Jon Kleinberg., *Meme-tracking and the dynamics of the news cycle.* Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2009

[6] Epshtein, Boris, Eyal Ofek, and Yonatan Wexler, *Detecting text in natural scenes with stroke width transform.*, IEEE Conference on Computer Vision and Pattern Recognition (CVPR)

[7] Max Jaderberg, Karen Simonyan, Andrea Vedaldi and Andrew Zisserman, *Reading Text in the Wild with Convolutional Neural Networks*, arXiv:1412.1842, 2014

[8] McCandless, Michael, Erik Hatcher, and Otis Gospodnetic. *Lucene in Action: Covers Apache Lucene 3.0. Manning Publications Co., 2010.*

[9] Manning, Christopher D., Prabhakar Raghavan, and Hinrich Schütze. *Introduction to information retrieval.* Vol. 1. No. 1. Cambridge: Cambridge university press, 2008.

[10] Davidov, Dmitry, Oren Tsur, and Ari Rappoport. *Semi-supervised recognition of sarcastic sentences in twitter and amazon.* Proceedings of the Fourteenth Conference on Computational Natural Language Learning. Association for Computational Linguistics, 2010.

[11] *Torch: A Scientific Computing framework for Luajit*, `http://torch.ch`

[12] *Microsoft Cognitive Services*
`http://www.microsoft.com/cognitive-services/en-us/computer-vision-api`

Appendix

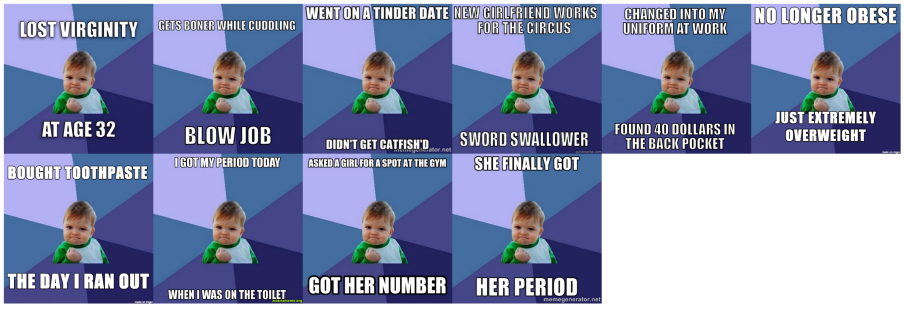
Results for query one does not simply

Results for query pokemon

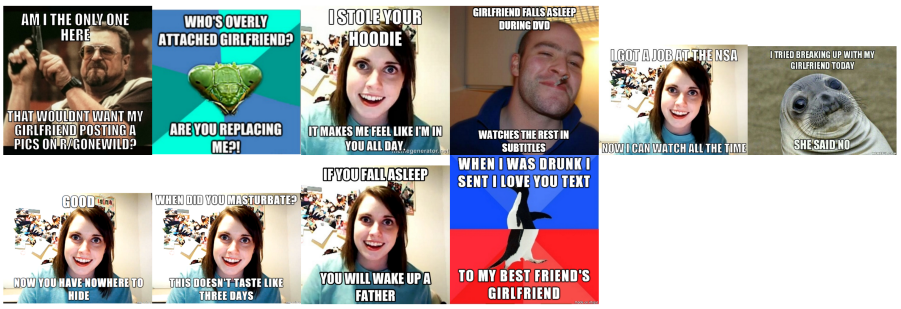
False negative for the query 'pokemon'

Results for query spiderman

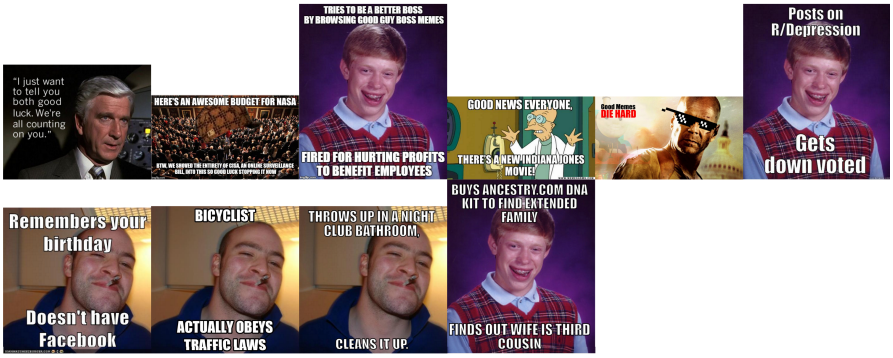
Results for query success



Results for query girlfriend



Results for query good luck



Results for query cat