The Report Committee for Olamide Temitayo Kolawole Certifies that this is the approved version of the following report:

Classification of Internet Memes

APPROVED BY SUPERVISING COMMITTEE:

Supervisor: ______________________________
Suzanne Barber

____
Kristen Grauman

Classification of Internet Memes

by

Olamide Temitayo Kolawole, B.S.

Report

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Master of Science in Engineering

The University of Texas at Austin

December 2015

Classification of Internet Memes

by

Olamide Temitayo Kolawole, M.S.E

The University of Texas at Austin, 2015

SUPERVISOR: Suzanne Barber

This paper explores a system that could be used to classify internet memes by certain characteristics. The anatomy of these viral images are explored to find the best indicators to classify an internet meme. Although more than one indicator was found, the paper focuses on the using image data to perform the classification. Further research is done to determine which type of feature descriptor would be used based on past successes of other projects. A dataset is a scraped from a popular repository of memes on the internet and their features extracted. Features are passed into a SVM classifier to derive a unique listing of potential labels that an image could have.

Although training times were very reasonable as the number of classes increased, result accuracy degrade with increase in number of classes trained on the same model.

Contents

Introduction

The objective of this work at its highest level is image classification, to be more specific, image classification of memes. A meme according to the Merriam-Webster dictionary is "an idea, behavior, or style that spreads from person to person within a culture". In 2015, an internet meme has come to represent viral images that exhibit certain attributes, usually accompanied with text in the form of a dialogue and a base image. For the purposes of this project, a "meme" refers to "internet meme". What can be defined as an internet meme, does not follow a singular style. Due to this variability, results attained on this project can vary over time, depending state of "internet meme affairs". Human creativity knows no bounds. This work will specialize on viral images that comprise of text at the bottom and top of the image. A "meme character" is a symbolic entity, usually at the center of an internet meme, which sets expectations on what kind of expression that an internet meme conveys. For example, in High Expectations Dad" [**Fig(2)**], the meme character is the Asian man with glasses. It is expected that the meme should convey discontent about a result if it is not the best. The expected input into the system is an image of a meme and nothing else. The end goal of this project is to explore a system that can correctly label an internet meme. Various websites such as memegenerator.com exist to auto-generate memes. The popularity of memes has warranted the existence of such sites. If this project is successful enough on a consumer level, a feedback loop can be created to not only create memes through an automated process like memegenerator.net but also identify them. Due to the open-endedness of viral images, there is no master list that will describe every meme ever created or a governing body that will label emerging memes in accordance to a convention. This means that the trained model cannot be trained on every label imaginable pre-emptively.

Related Work

In a paper by Bosch et al [9], the paper attempts to classify images using random forests and ferns classifiers and describe its acquisition of training data from the Caltech-101 dataset. Feature descriptions from the images takes two-pronged approaches derived from the shape and appearance of an image. The features describing appearance is sourced using SIFT (Scale-Invariant Feature Transform) descriptors. Multiple SIFT descriptors are computed to handle scale variation between images. The shape of the training set was represented by using Histogram of edge Orientation Gradients, HOG. In HOG, edges with orientations in similar ranges are grouped into bins. The outputs of both shape and appearance are concatenated and serve to describe the image. The authors find similarities between images in the Caltech dataset using a kernel function, finding a measure of similarity between two images was crucial to the authors. Feature descriptors (shape and appearance) in combination with the kernel function deliver fixed regions from many training images to be feed into the model. Similarity is calculated within the label instances to be classified, for example, all training images with the label: Watermelon. A random forest classifier is used as the main driver of the paper instead of the traditional SVM. A random forest classifier operates by creating random decision trees, test images are passed down random tree until it reaches a leaf node. Posterior probabilities are averaged to find the final classification of an image. A goal of the paper is to achieve faster training times while maintaining the same level of performance compared to the traditional SVM. A random ferns classifier was used to deliver speedy training times. The paper concluded that it is possible to have comparable performance while having faster training using alternate classifiers to SVM.

In “Understanding Image Virality” by Deza et al [1], the paper attempts to gain insight what makes images viral. They illuminate characteristics that determine how pervasive an image can be, and determine if there are certain features that can be used to predict virility of an image. The paper utilized Reddit to gain metrics and data on image virality. The authors decided to use Reddit instead of other social networks platforms due to the anonymity associated with using Reddit. The authors believe that the purest form of "internet trolling" will be less filtered due to the lack of direct consequences that come from being anonymous. Reddit employs a reputation based scheme to determine top rated posts and posters. The paper defines viral images, in the context of Reddit, as an image that has many upvotes, few downvotes and have been resubmitted several times by many users. A virality score function was determined with these factors in mind. Four forms of contexts were derived to better understand image virality.

1. Intrinsic context: "visual content related to the pixels of the image".
2. Vicinity context: "visual content of images surrounding the image"
3. Temporal context: "visual content of images seen before the image"
4. Textual context: "non-visual context referring to title or caption of an image"

As it relates to internet memes in this project, the intrinsic and textual contexts will be most relevant. Intrinsic context will be in reference to the background image or the meme character present in an internet meme. This context alone should be able to provide enough features to be able to classify an internet meme. The secondary context for internet meme would be its textual context. The text or captions of the meme will often relate to the ongoing characteristic that allows the meme to be viral. With that said, it is not a written rule for the caption of an internet meme to conform to its meme background every time. Human creativity knows no bounds, it will be difficult for an artificially intelligent system to determine if a deviant use of a caption on an image is wrong or just being more creative. The paper determined that a high-level of understanding of the image is key in predicting virality. Also, virality can be predicted more accurately as a relative concept.

On the website, PyImagesearch.com, it comprises of multiple avenues to search images for content strictly related to using the Python language. One of the more useful blog was "Histogram of Oriented Gradients and Object Detection"[12] by Adrian Rosebrock. In this blog, it was able to reinforce the use of histogram of gradients to detect subject matter in images. The author focuses on guided steps to achieve the intended results using available tools rather than a broad exploration of Object detection. The author maintains a dissatisfactory stance on using a Haar-Cascade classifier from the OpenCV library [13]. Haar-Cascade classifiers were introduced for object detection by Viola and Jones in "Rapid Object Detection using a Boosted Cascade of Simple features"[5]. Three Haar-like features [14] are used by Viola. A two-rectangle feature is the difference between the sums of the pixels within two rectangular regions. A three-rectangle feature is the sum within two outside rectangles subtracted from the sum in a center rectangle. A four-rectangle feature is the difference between diagonal pairs of rectangles. There are a lot of features extracted from an image, Viola et al. proposed to use AdaBoost [15] to select the best features. Rosebrock found that the performance of the Haar-cascade classifier could be problematic in his experience. He states that a lot of time is used in tuning the parameters of the classifier to get useful results. Often, more tuning most occur on other training datasets to yield comparable results. Rosebrock admits that while the Haar-Cascade classifier's performance could be blamed on its age, the Histogram of Oriented Gradients methods is “old" as well but it is still in use today. On Rosebrock's blog, he delves into a six step process to achieve relatively decent performance for object detection using Histogram of Oriented Gradients and Linear Support Vector Machines.

1. Sampling positive images
2. Sampling negative images
3. Training a Linear SVM
4. Performing hard-negative mining
5. Re-training Linear SVM using hard-negative samples
6. Performing non-maximum suppression to remove redundant, overlapping regions identified by the classifier.

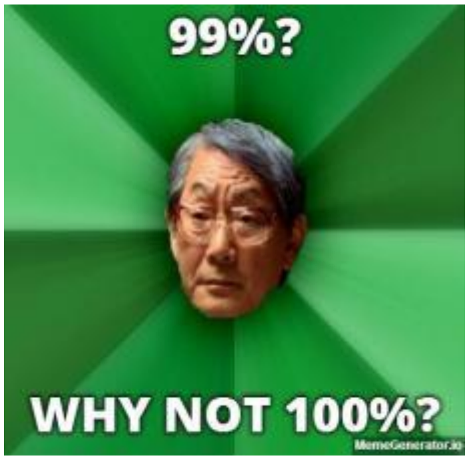
The end result of Rosebrock's process is very visual. It is meant to identify the explicit region of interest in a test image. Hence, non-maximum suppression is crucial to his goals. Non-maximum suppression attempts to remove redundant overlapping classifications over the same area of an image. The decision function of the classification output is used to prioritize desired results, the higher the confidence (decision function) - the better. Rosebrock’s aim is to designate a region on a test image and reveal them to stakeholders/users as the object of interest. A unique listing of all output of the classifier (labels) as it traverses over a test image is desired for classification of internet memes in our instance. Our output to the consumer will be a list of internet meme names that are present in the test image.

In understanding the psychological ability for a human to classify memes, one can find that it is possible decipher the internet meme just by its captions alone.   This is highly dependent on the meme and its characteristics that allow it to be easily identifiable and the vastness of pop culture knowledge of the actor inferring from the image. For example, the meme "Good Guy Greg" signifies a meme character who does/thinks well regardless of the situation that he is put in. On the other side of the spectrum is "Scumbag Steve", this internet meme positions itself to always behave wrong/inappropriate in every situation regardless of the good will that has been put forth to him. Examples captions include

Scumbag Steve		
Good Guy Greg		EATS PIZZA WITH YOU PRETENDS TO BE FULL SO YOU CAN HAVE THE LAST SLICE memegenerator.net

Fig (1)

A person well versed could classify these captions with some accuracy without seeing the image. This means that features could be extracted from the captions to classify the image solely or in conjunction with image data. In the field of Natural Language Processing, research efforts exists to be able to build better consumer review systems, online help systems [6] etc. Research efforts must be able to decipher speech traits and patterns such as irony and sarcasm from pure text. In a paper [16] by Davidov et al, the authors sought to build a model that can recognize sarcastic sentences on twitter and reviews on Amazon. Understanding sarcasm plays a big role in understanding most internet memes and it is a reason for success in terms of virality. Davidov et al build a system by finding groups of words that represent sarcasm with a #sarcasm hashtag on twitter, extract patterns and punctuation-based features, then pass through a KNN-like classifier. This is a step in the right direction but there are more barriers ahead to be able to classify internet memes based on their caption alone. Other factors such as oxymoron, irony and witticism come into play when digesting internet memes in a human sense. Acquisition of datasets accurately portraying these factors come from twitter [7][16] or large scale human-based classification projects like Amazon Mechanical Turk. The original challenge to extract words from an internet meme image (Optical Character Recognition) should be another project in itself. Therefore, this project will only be focusing on classification based on image data.



Fig(2)

In the “High expectations dad” meme, [**Fig(2)**], if the texts on the image does not relate to a father who demands the best, the image would still be considered as the "High Expectations Dad" meme because of the region of interest with the Asian dad with glasses. The meme character carries the most weight, a mislabeled meme is a wrong meme. Meme characters carry most of the visual context needed to identify a meme.

Datasets

At this time during our research, there were no available resources that store internet memes in a format that could be easily consumed for this project. Known image database used in the field of Machine Learning store images with intentional diversity and varying scales, viral images are not part of them. There are a lot of sites that contain millions of memes but they may not be named appropriately for the use of training a model. To remedy this, images are scraped from a website, memegenerator.net, by a Python script and stored categorically. On memegenerator.com, brief synopses of internet memes are described followed by sample images of said memes. As the name of the website implies, it allows templated creation of meme images. As described in the introduction, the meme usually follow a formula of a base image, followed by words at the top and bottom of the screen. Memegenerator.net monetizes this concept, allowing almost anyone to create any meme to any situation that demands a witty response without knowledge of photo editing tools. This project will focus on a list of memes curated by the administrators of the website tracking the most popular memes. Whether the list is very accurate is not very important at this time, all that is needed is a base set of memes to use. A scraper has been written in Python to scrape the list of most popular memes and organize them into a format that can be easily consumable for a training model. At the end of a scraping session, the intent is to have numerous folders named appropriately that will denote what meme images are contained within. The name of the folder will form labels that will be used during training.

At the time of writing, images scraped from meme generator are of JPEG type and are sized uniformly at 250 by 250 pixels. Each image is in color and have an average size of 40 kilobytes. The convenience of the pre-emptive scaled images is due to the scraping of gallery images used for navigation to the real "Original Sized" images.

A total of twenty memes where culled from memegenerator.net. In each meme, ten images are selected as they appear on the website. A certain segregation of memes were handpicked as a “worst case scenario” for edge case testing. This special set exhibit gross occlusion by the captions or mutation of the meme characters into other forms.

Extracting Features

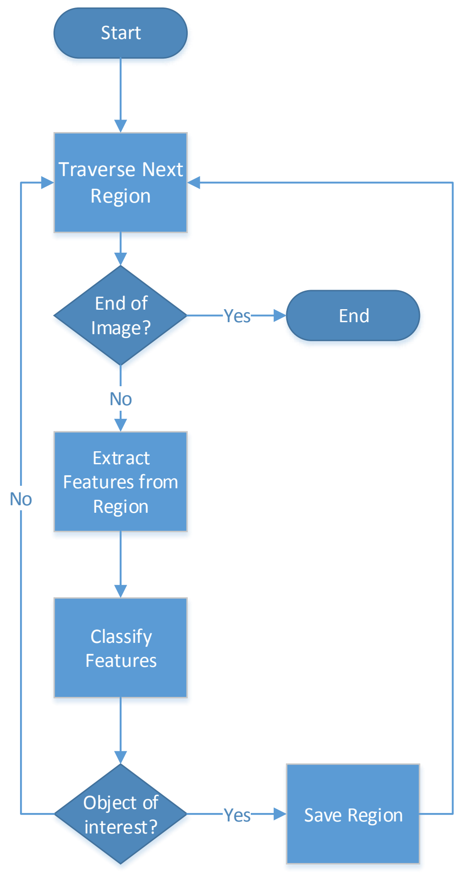
Certain properties have to be computed into relevant features that will be used to train a machine learning model. Most times, data mining datasets are represented with numbers. Different methods of feature extraction exists in the computer vision today with varying complexity. Image features sort to explain phenomenon such as lines, edges, and contrast or interest points such as blobs or corners.

One method for determining edges in an image is canny edge detection [17]. Named after its creator, it is a multistage algorithm that is used to determine where edges are located in an image. This will serve to explain the shape of an image. A Gaussian filter is used to smooth the image to remove noise. The standard deviation of this filter is crucial to determine how "edge-y" the output may become. Too much smoothening can remove some vital features that may have been able to differentiate two similar images. Image gradients are calculated on the smoothened image with different filters to capture gradients with different directions. Consequently, weak gradients (edges) are removed from the image with non-maximum suppression and double thresholding. The output is an image of the same size as the original with only edges highlighted and nothing else. The Scikit image library [19] has an implementation of canny edge detection that outputs a two dimensional array representation. This representation will only contain zeros and ones. It is important to note that the original image must have grayscaling applied to work. The three channels in a color image (red, green and blue) prove to be too high in dimension space. Grayscaling serves to retain intensity information of every pixel in an image.

The histogram of oriented gradients [11] is another descriptor that was explored in this project. The descriptor serves to compute gradients of local sections of an image and forming a histogram of gradients per orientations for the local cell of pixels. The Scikit image library has an implementation of Histogram of oriented gradients algorithm. This implementation flattens the resultant histogram into a feature vectors. Both Canny and Histogram of Oriented Gradients were evaluated during the testing process.

Training the Model

The classification methodology of internet memes used in the project is similar to practices used in Object Detection [9][12].



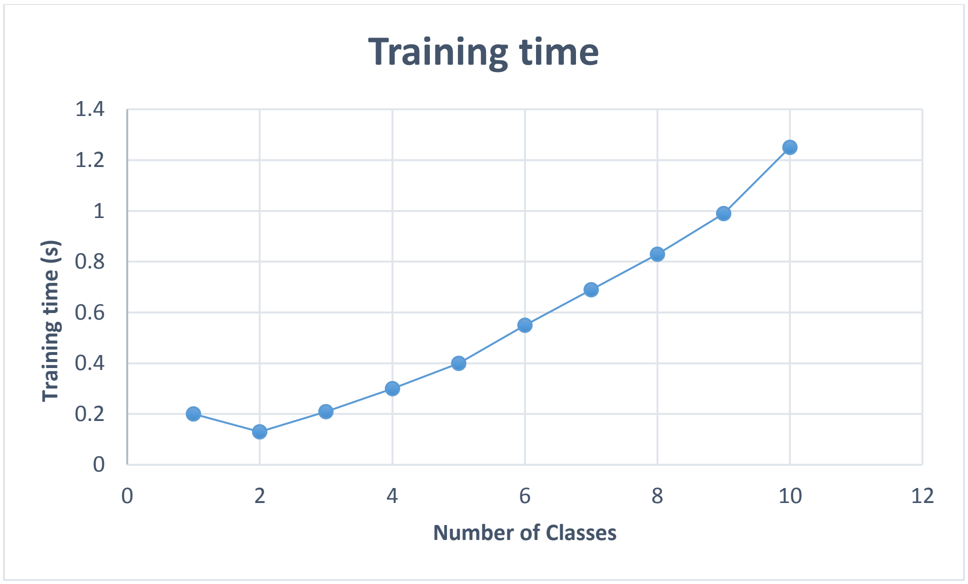
Fig(3)

In our implementation, each sub section window is 100 pixel by 100 pixel image, this window slides from left to right and then moves 10 pixels down to start the iteration all over again. This means that this method does not work well for images that are smaller than 100 pixels by 100 pixels. On the other end of the scale, very large images take a long time to process. In addition to the sliding window, the test image is scaled in decreasing steps to capture meme characters that may occur at larger or smaller scales than the training scale. Scale Invariance is an ever present challenge in computer vision, algorithms such as SIFT and SURF [18] tackle this problem in a more holistic manner.

The classifier used is a traditional support vector machine. SVM seems to be the goto classifier in object detection [9]. Training times are very fast on the order of seconds. A key component to a classifier that is chosen is its ability to persist after training. The model must have the ability to be serialized. This allows re-instantiation of the model to be used at a later time. Decision trees were considered and had showed decent performance in previous explorations [9], unfortunately there is a bug that disallowed random forest, decision trees to be serialized in our python implementation. This was quite unfortunate as the fast and moderate performance of Random Forest used in [9] was a great inspiration in terms of model selection.

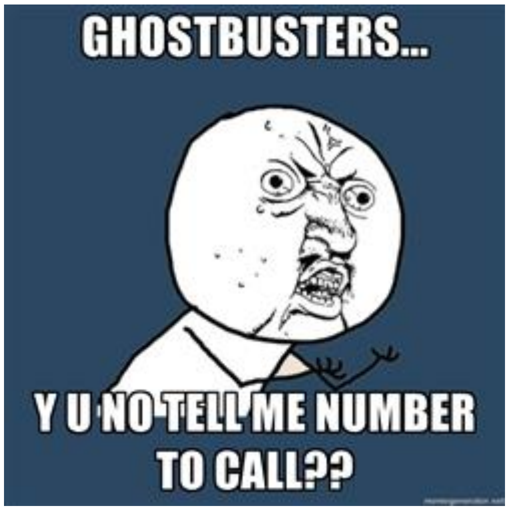
Results

Classification performance proved very interesting depending on certain scenarios. Initially, two variants to the training dataset selection was used. The first variant used cropped images of the meme characters. Each crop focused further and further into the meme character and then resized to the standard size of 250 x 250 pixels for consumption into a model. This process was done automatically by a python script. For some memes, the cropped image contained parts of the caption/text. This could further skew the results by introducing features that do not represent the meme character. The second variant of the training dataset involves hand picking the best area of the image that is least likely to be occluded by texts/captions on an image. A major drawback to the second variant is its compulsory use of human intervention. Time to train the classifier increased slightly per class available for classification as shown in [**Fig(4)**]. There appears to be a linear relationship between the training time and the number of classes to be classified. At this rate, the training time is very reasonable and would not need to adopt a faster method of classification like a random forest used in [9]. Training time for a thousand class is estimated to be about 120 seconds.



Fig(4)

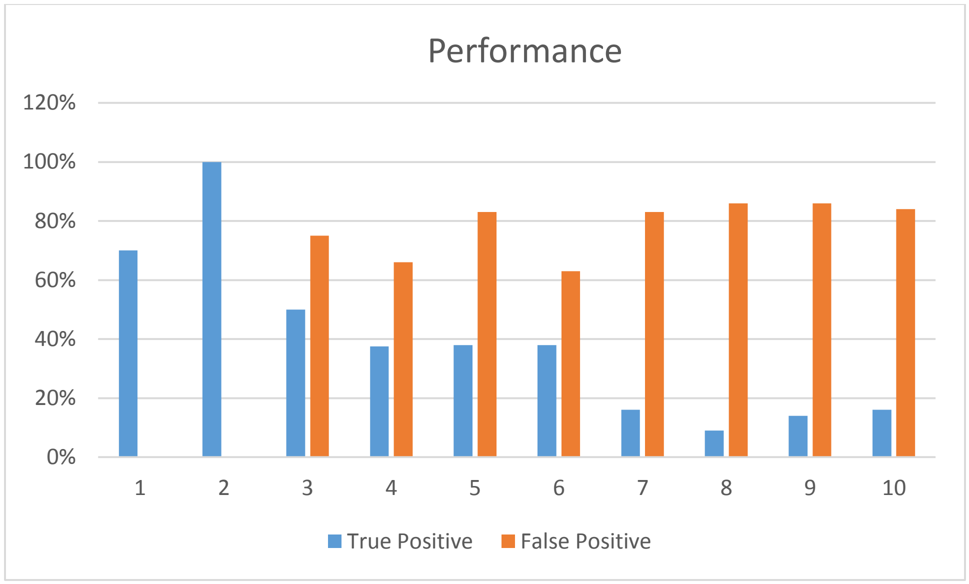
The output of our python classification module is a potential list of memes that were believed to be the meme. That means there can be more than one result. When this occurs, some of its labels can be designated as a false positive. A particular test instance can have both the correct label and other false positive labels, this test instance will count towards both true positive and false positive metrics in **[Fig(6)**]. Classification performance with 4 carefully chosen classes was at 90%, while false positive of the set was 11%. It is important to remember that a test instance can count in both the true positive and false positive sets if the classification module outputs more than one label. In **Fig(5)**, the module outputs “Y-U-No” and “The-Most-Interesting-Man-In-The-World”, the only correct label should be “Y-U-No”.



Fig(5)

<u>Classification Output</u>:
Y-U-No
The-Most-Interesting-Man-In-The-World

The following diagram, [**Fig(6)**], was built by using internet meme classes with no curation. The python classification module is retrained with the increasing number of classes. The negative class is included in every retrain but does not count towards the number of classes(X axis) in [**Fig(6)**]. Classification performance is very good when a minimal number of classes are involved. As the number of classes increase, the false positive rate increases greatly with the true positive rate on a decline. Real world usage of this model would yield incorrect results most of the time. It would be an expectation of this model to deliver likely suggestions that would be in close proximity to success. A true positive rate lower than twenty percent for higher order of classes would miss the mark for consumer grade applications.


Performance
120%
100%
80%
60%
40%
20%
0%
1
2
3
4
5
6
7
8
9
10
True Positive
False Positive

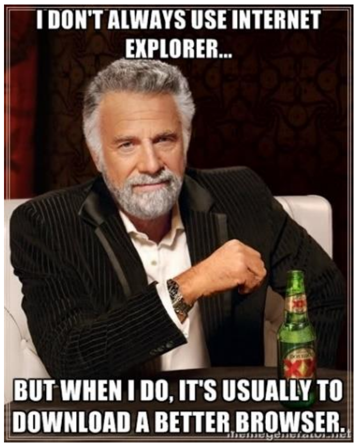
Fig(6)

Discussion

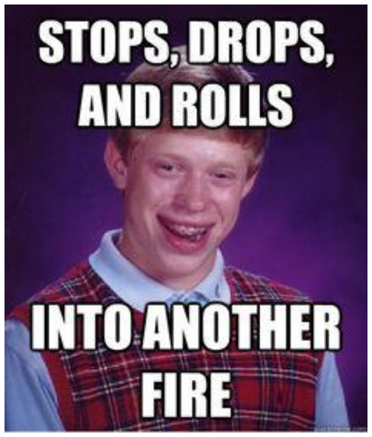
It is determined that the model at the current stage of sophistication cannot evaluate every variation of an internet meme, even ones in the same class. Classifier performance decreases with an increase in the number of classes added. This poses a problem for developing a real world application that is meant to classify a large number of memes. To retain optimal accuracy, a minimum amount of classes could be trained by the classification module, possibly three, excluding the negative class. Ensemble techniques can be used to pass test images through a pipeline of classifiers trained on different sub groups of classes to obtain a palpable result. Training time for the classifier was less than three seconds for the maximum number of classes tested in [**Fig(4)**] and [**Fig(6)**]. The training of possibly three hundred classifiers would still be within the realm of possibility.

Alternatively, specificity of numerous models of the above proposed technique can be eliminated by using the latest advances in Machine Learning, Convolutional Neural Networks (ConvNet). An advantage of ConvNet is the ability to use datasets as they are with minimal amount of preprocessing. Applications of ConvNet can benefit more in terms of speed from distributed processing with GPUs. Implementations like LeNet and Caffe allow the option to run on CPU or GPUs. It is proven to work for classification attempts with 1000 classes/labels [20].

During testing, it was found that some classes were more inclined to be falsely labelled more frequently than others. A prime example is the "The most interesting man in the world" [**Fig(7)**]. On the other side of the spectrum, "Bad Luck Brian"[**Fig(8)**] was very difficult to classify correctly, even though it was trained as the only class besides the negative class. It is not understood what phenomenon the features of these special classes exhibit to fall on extreme ends of the spectrum. The meme characters cannot be changed to suite training, hence more study must occur to determine factors that lead to extremism of these special classes.


I DON'T ALWAYS USE INTERNET EXPLORER...
BUT WHEN I DO, IT'S USUALLY TO DOWNLOAD A BETTER BROWSER.
memegenerator.net

Fig(7)



Fig(8)

Conclusion

This project has created its own dataset for consumption from a popular website into a Machine Learning model. Attributes that could make up and identify a viral internet meme was determined. Identification of internet memes can be gleamed solely its caption or its base image. Much work is still to be done to reach a consumer grade application to give a user with an accurate interpretation of a meme. Some memes in the training dataset were difficult to classify successfully such as "The most interesting man in the world" and “Bad Luck Brian”. It might be possible to use different feature sets that can successfully differentiate these tricky classes with clearer decision boundaries. Online training i.e. adding more labels to a previously trained model would be ideal if newer memes are introduced into the wild.

References

1. A. Deza, D. Parikh, “Understanding Image Virality”, arXiv:1503.02318v3 [cs.SI] 26 May 2015
2. C. Papageorgiou, M. Oren, T. Poggio, "A general framework for object detection", International Conference on Computer Vision, 1998
3. Q. Le, M. Ranzato, R. Monga, M. Devin, K. Chen, G. Corrado, J. Dean, and A. Ng. “Building high-level features using large scale unsupervised learning”. In International Conference in Machine Learning, 2012
4. H. Agrawal, N. Chavali, M. C., Y. Goyal, A. Alfadda, P. Banik., and D. Batra. Cloudcv: Large-scale distributed computer vision as a cloud service, 2013
5. P. Viola, M. Jones, “Rapid Object Detection using a Boosted Cascade of Simple Features”, Conference on Computer Vision and Pattern Recognition, 2001
6. A. Kongthon, C. Sangkeettrakarn, S. Kongyoung and C. Haruechaiyasak “Implementing an online help desk system based on conversational agent” Proceeding, MEDES '09 Proceedings of the International Conference on Management of Emergent Digital EcoSystems, ACM New York, NY, USA
7. F. Barbieri, H. Saggion, “Modelling Irony in Twitter”, Conference of the European Chapter of the Association for Computational Linguistics, 2014
8. J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. arXiv preprint arXiv:1310.1531, 2013
9. A. Bosch, A. Zisserman, X. Munoz, “Image Classification using Random Forests and Ferns”, ICCV 2007, 2007.
10. L. Breiman. Random forests. Machine Learning, 45:5–32, 2001
11. N. Dalal and B. Triggs. Histogram of oriented gradients for human detection. CVPR, 2005.
12. A. Rosebrock, “Histogram of Oriented Gradients and Object Detection” [Web log post]. Retrieved August 31, 2015 from http://www.pyimagesearch.com/2014/11/10/histogram-oriented-gradients-object-detection/
13. G.Bradski, The OpenCV Library, Dr. Dobb’s Journal of Software Tools, 2000
14. Haar A. *Zur Theorie der orthogonalen Funktionensysteme*, Mathematische Annalen, **69**, pp. 331–371, 1910
15. Y. Fruend, R. Schapire, “A decision-theoretic generalization of on-line learning and an application to boosting". *Journal of Computer and System Sciences* **55**. 1997
16. D. Davidov, O.Tsur, “Semi-Supervised Recognition of Sarcastic Sentences in Twitter and Amazon”, Fourteenth Conference on Computational Natural Language Learning, 2010.
17. Canny, J., *A Computational Approach To Edge Detection*, IEEE Trans. Pattern Analysis and Machine Intelligence, 8(6):679–698, 1986.
18. H. Bay, T. Tuytelaars, L. Van Gool, "Speeded Up Robust Features", ETH Zurich, Katholieke Universiteit Leuven
19. Scikit-image library, http://scikit-image.org
20. A. Krizhevsky, I. Sutskever, G. Hinton “ImageNet Classification with Deep Convolutional Neural Networks”, Advances in Neural Information Processing Systems, 2012