6-1-2020

Learning Humor Through AI: A Study on New Yorker's Cartoon Caption Contests Using Deep Learning

Ray Tianyu Li
Dartmouth College

LEARNING HUMOR THROUGH AI: A STUDY ON NEW YORKER'S CARTOON CAPTION CONTESTS USING DEEP LEARNING

A Thesis

Submitted to the Faculty

in partial fulfillment of the requirements for the

degree of

High Honors of Bachelor of Arts

in

Computer Science

by

Ray Tianyu Li

DARTMOUTH COLLEGE

Hanover, New Hampshire

June 2020

Dartmouth Computer Science Technical Report TR2020-893

Thesis Advisor:

Professor Soroush Vosoughi

Abstract

My research focuses on predicting a cartoon caption's wittiness using multi-modal deep learning models. Nowadays, deep learning is commonly used in image captioning tasks, during which the machine has to understand both natural languages and visual pictures. However, instead of aiming to describe a real-world scene accurately, my research seeks to train computers to learn humor inside both natural languages and visual images. Cartoons are the artistic medium that supposes to deliver visual humor, and their captions are also supposed to be interesting to add to the fun. Thus, I decided to use research on cartoons' captions to see if deep learning models can, in some ways, learn human humor. I ended up using New Yorker's Cartoon Captioning Contests as the dataset to train a multi-modal model that can predict a cartoon's funniness. The model didn't beat the benchmark in terms of accuracy of the classification task, but it eliminated some unsuccessful attempts and set us up for the future study on this topic.

Preface

This paper could not be possible without many people who have helped and supported me in my research journey. First and foremost, I would express my greatest appreciation to my advisor, Professor Soroush Vosoughi. He showed me this extremely interesting topic and provided me many new ideas and directions for my research. Of the many things he had contributed to the thesis, I am most appreciative of his patience and trust. This paper wouldn't take its current shape without Professor Vosoughi, and I will never forget moving the chess pieces with him after our weekly meetings.

I am also grateful to everyone who introduced me to machine learning and its research. I want to thank Professor Qiang Liu, Professor Lorenzo Torresani, and Professor John Voight for introducing me to machine learning, deep learning, and the mathematical concepts behind them. I'm also grateful for the teachers of MIT course 6.867, Professor Devavrat Shah, Professor Suvrit Sra, and many others, for opening my eyes on the beauty of the advanced theories of machine learning. Finally, I would like to especially thank Professor Feng Fu for giving me the opportunity to first-author a machine learning related paper before the end of my sophomore year. His supervision led to the paper's eventual publication a year later. I wouldn't be doing a thesis for my computer science degree without that research experience.

Contents

Chapter 1

Introduction

In social media, people tend to spend more and more effort writing a good caption for their pictures. This is evidenced by the many online articles [1] that teach people how to write better captions for their posts on Instagram, a popular social media platform initially built for the sharing of mere photos. But competitions of writing good captions for pictures started way earlier than Instagram. Even back in the late 19th century, there were already magazines publishing caption contests, some even with rewards up to $1000[2]. Fast forwards to the 21st century, the most popular and competitive picture caption contests come from *The New Yorker* magazine. Its weekly cartoon caption contests first appeared in 1998 and have been published regularly since 2005[3], have attracted both billionaires and Pulitzer Prize winners[4]. The contest is extremely competitive, and according to a Wall Street Journal article, "Michael Bloomberg, self-made billionaire and three-term mayor of New York, has been heard to complain that no matter how hard he tries, he can't even come up with a contender."[4] What Michael Bloomberg struggled with was to give a title to a picture like this:

Figure 1.1: **The Most Recent Caption Contest on *The New Yorker Magazine* before the submission of this thesis [5]**

And what Mr. Bloomberg wanted to achieve is to be featured on *The New Yorker's* website like the winner of the latest contest:

Figure 1.2: **The Winner and His Caption for Contest #709[6]**

As computer scientists, we are intrigued to think about a way to design an algorithm to solve this problem. The algorithm can be rule-based but can also be

machine-learning-based. But before we jump into the issue, let's take a look at what people think is the key to win this competitive contest.

Section 1.1

What Makes a Good Caption

Don't let the section name fool you. In this part, we will discuss what makes a good caption for *The New Yorker's editors*. According to 7-time *The New Yorker* caption contest winner Larry Wood, pun is an awful choice, and he didn't think the good puns can "redeem the pun's well-deserved reputation as the easiest and lowest form of humor." He also talked about what makes a bad caption for the contests. "They're too long, they're too obvious, they don't end with the punch line, they don't recognize which character in the cartoon is speaking, and they're just not funny." But when it comes to what makes a good caption, Larry Wood is didn't quite articulate his point. He claimed that a good caption should "be brief, end with a punchline, and be funny." That barely helps us to construct a useful algorithm, since if we already know concretely what is "funny" we won't need to research on writing "funny" captions.[7]

On the other hand, scientists provided us a more specific set of criteria to focus on in order to write better captions for the contest. According to professor Peter McGraw, "you'd do well to mind these four factors: novelty, length, punctuation, and 'abstractness and imaginability.'" [8] McGraw is a professor of marketing and psychology at University of Colorado Boulder. The conclusion came from research he has done with Phil Fernbach, a cognitive scientist from Brown University. During their research on more than 5,000 captions for contest #281, they find that "captions with uncommon words were more likely to make the shortlist," and also "captions with fewer punctuation marks fared better than others, as did captions that were harder to visualize." These findings are helpful, but even professor McGraw himself is unsure

of the absolute applicability of his discovery. In an interview with *Wired*, professor McGraw said, "My guess is that we have to be a little bit careful about the conclusions we can draw." He also said: "This is only one contest. As a good scientist I would want to replicate these effects with many different contests." Professor McGraw's hesitation is valid, not only because he only did his research on one contest, but also because he only compared the 42 shortlisted titles with the rest 5000 captions. In other words, his research only helps predict which caption will get shortlisted instead of predicting which caption will be the finalist or even the winner. Whether his rules apply to contests winners are still largely unknown.

Apparently, a rule-based algorithm will not lead us to a satisfactory result, since neither the 7-time winner nor the professors can articulate a set of "rules" that can help them find the winner or even a list of finalists. So we are forced to look into the machine-learning-based algorithm. The only requirement is that I need more data than professor McGraw's 5000 captions, and fortunately, there is such a dataset available with more than a million captions. In the next chapter, I will detail its features and collection so that we can train a machine learning model on it to hopefully understand humor.

Chapter 2

Dataset

With the help from *The New Yorker* magazine itself, we have a public dataset[9] that collects crowdsourced users' ratings on over 1 million captions for more than 100 contests.

Section 2.1

Dataset High-Level Description

The dataset is collected using data provided by *The New Yorker* magazine. Every time a cartoon caption contest is held, the magazine sends the cartoon and its many potential captions to the organizers of this database. Then the organizers used *Next Active Machine Learning System*, "a cloud based machine learning system for applying state-of-the-art adaptive data collection techniques[10]," to get ratings of captions from users online. And finally, they score the users' responses and decide which caption is the best.

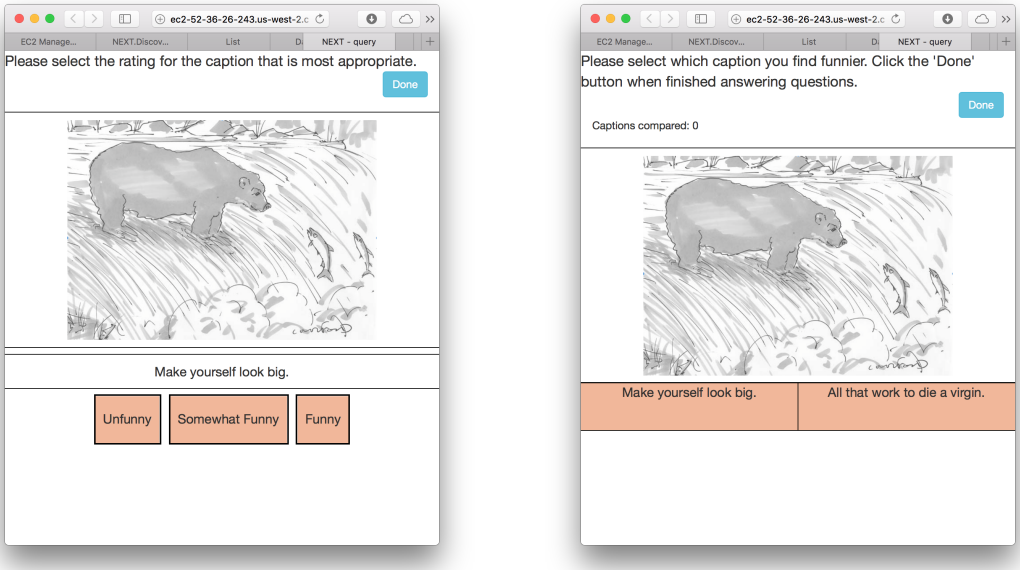
Section 2.2

Data Collection

There are two types of questions for online users and also two types of algorithms for the cloud-based platform to release these captions to the users.

2.2.1. Sample Query

There are two types of queries that are provided to the users. The first type is to rate "how funny is this caption." Users are given three choices to choose from: "unfunny," "somewhat funny," and "funny." The dataset calls this type of query "Cardinal." The other type of query is "which caption is funnier?" Two captions for the same cartoon are given to the users, and they are asked to choose the better one. This type of question is titled "Dueling" by the dataset.



(a) Cardinal (b) Dueling

Figure 2.1: **Sample Query**

2.2.2. Collection Algorithms

Since the users have to give responses to a series of captions or caption pairs, the order of the sequence in which the users receive these questions matters. The queries are generated by the platform in two different ways. Specifically, there is a random algorithm that generates the queries randomly, and there is an adaptive algorithm that generates the queries based on users' previous responses to decide which query to ask next.

Section 2.3

Content of Data

There are two types of data. The first type is **response**, which is a CSV file and include the following metrics: Participant ID, Response Time (s), Network delay (s), Timestamp, and algorithm label. Each of them is quite self-explanatory, and these five measurements are recorded every time a user rate a caption. Depending on whether the caption is *cardinal* or *dueling*, another 2 measurements are recorded as well. If a caption is *cardinal*, then the caption the user is asked to rate, and the rating (funny, unfunny, or somewhat funny) is recorded. If the caption is *dueling*, then the whether the caption appears on the left or right side and the funnier caption are recorded.

The focus of my research is on the other type of data in the dataset called **summary**. They are basically the summary of responses from every contest. The header of **summary** looks like this. The headers are quite self-explanatory, but the *score* is calculated through the following equation:

$$score = \frac{(\#funny * 3 + \#somewhat\_funny * 2 + \#unfunny * 1)}{(\#funny + \#somewhat\_funny + \#unfunny)}$$

and the *precision* is just the standard deviation with all these input of funny levels and *score* being the mean.

	caption	contest	count	funny	precision	rank	score	somewhat_funny	target_id	unfunny
0	They must have made a fortune from the wheel.	682	9234	2956	0.007887	1	2.062053	3895	3586.0	2383
1	We are going to go with Noah, but thanks.	682	11024	3356	0.007501	2	1.988480	4185	2858.0	3483
2	I guarantee he brings up the whole discovering...	682	10148	3035	0.007865	3	1.969551	3769	399.0	3344
3	What'd you expect? She hunts-gathers, too , a...	682	10164	2890	0.007816	4	1.944805	3823	816.0	3451
4	Of course it's pretentious- didn't you see the...	682	9492	2767	0.008210	5	1.939633	3385	1986.0	3340

Figure 2.2: ***Summary* Data**

Section 2.4

Previous Research

Besides the general research on *The New Yorker* cartoon contests mentioned in Chapter1.1, there is one previous research that focuses specifically on this dataset, and several more quantitative research focuses on a smaller dataset than this one, which contains only 5000 captions for one contest.

The research done on this larger dataset was delivered on GitHub by a user with user name **RajeshThevar**[11]. He purposed to use two neural networks in tandem to generate captions. Specifically, he claimed to use word2vec[12] to extract embeddings from captions and a pretrained VGG network[13] to extract embeddings from the cartoons. But in his code on GitHub, he only used LSTM[14] model to generate somewhat less "random" words as his caption and claim that he has solved the captioning problem. Since he didn't utilize the images and the ratings from the dataset, unfortunately, I don't think his work is satisfactory by any standards.

Other research has been done on a smaller dataset. One research from Yale[15] created a language model based on the seven-time winner Larry Wood's advice[7]. In his interview with Cartoon Collections Blog, Larry Wood gave out one practical tip for writing a good caption. "Most of the drawings used in the contest contain two frames of reference that you have to connect through the caption," says the

7-time caption contest winner. The Yale research, based on the same idea, built a system where users can input "two parts of the "incongruity" apparent in the image." Then the system will generate a group of captions for users to choose from. This is an interesting research, but it didn't really automate the process of captioning the picture. Another research from Yale[16] attempted to predict the "funny score" for each caption by using linear regression. The model, unfortunately, didn't work, but it pointed a good direction for me to move forward - using a more complicated model to achieve a better prediction of the humor level of the captions. But before we dive into my research, I would like to first discuss my use of the dataset.

Section 2.5

Our Usage of the Dataset

Since there aren't enough *Dueling* contests (there are only less than 10 of them), I used *Cardinal* contests exclusively. The range of the data I used is from contest 508 to contest 689, a total of 182 contests, with more than 1 million captions. I didn't make a difference based on the captions' collection algorithms. That is, I treated users ratings from both adaptive algorithm and random algorithm in the same manner. The reason for ignoring the algorithms' potential impact is that the difference is not that relevant to the current phase of my research. Finally, I only used the *summary* data because the *response* data are too granular for my research.

Chapter 3

Captions Side: BERT Model

I intended to build a multi-modal model that put together features from images and captions and train the joint embedding on the funny score. To do that, I need to first get the embeddings from captions. The task was given to BERT model.

Section 3.1

What is BERT and Why BERT

BERT is the abbreviation for Bidirectional Encoder Representations from Transformers[17]. It is a natural language processing model developed by Google in 2018 that built on the ideas of Transformers[18] and ELMo[19], among other similar ideas. I chose BERT instead of other models like word2vec[12] because BERT can understand the context of a word. For example, in the sentence "I took money from the bank and then walked to the river bank," the word "bank" has different meanings in its two appearances. Since a BERT model is pretrained on both directions and is trained in semi-supervised learning fashion[20] to understand the context of each word, it can recognize the difference between the meanings of the two "bank"s, while traditional NLP models mostly fail to detect the difference. When compared to newer models, BERT is the most well-established model that has lengthy documentation and a

sizeable community. Besides, BERT broke records in most downstream NLP tasks and is regarded as the new benchmark.[17] Thus, BERT is my natural choice for feature extraction on the captions.

Section 3.2

Choosing the Best Layer

Since a pretrained BERT model comes with many layers, we need to decide which layer will become our eventual embedding in the final model. Due to constraints of time and computational power, I decided to use the "BERT-base-uncased" model.[21] It has 12 layers, 768 hidden units, and 110 million parameters, and is trained on lower-cased English text. Due to the constraints again, I ran the model on a smaller dataset that is randomly sampled from over 1 million captions in the original dataset. The smaller dataset has 10739 captions. I calculated the caption's embedding by averaging the embedding on each word in that caption, which is one of the most standard ways to get sentence embedding.[22] This gives me a 12x768 tensor for each caption. That tensor represents 12 layers, in which there is a 1x768 vector representing this caption's embedding. Now I have to decide which layer is the best layer for my final task of rating how interesting a picture is. I need to compare each of the layer's embedding on a downstream classification task. And random forest algorithm comes handily as the classifier.

Section 3.3

Random Forest

Random forest[23] is an algorithm based on decision tree algorithm[24]. It adds bootstrap aggregating to tree learners, and sample from the dataset with replacement to feed to each of its tree learners. It also randomly select participating features, so in

some trees feature A aren't selected in the training set. This de-correlated the many tree learners, and together the trees predict values or help classifying a certain input. By using ensemble learning, random forest has advantage over decision tree as it is less likely to overfit to the data. Comparing to other classification methods, random forest has the following advantages -

(a) It is designed for multi-class classification, unlike SVM[25] which is inherently binary. We don't need to make any modification on random forest to let it classify the three classes

(b) It is agnostic to the scale of its data. That is, if a feature in the 768 features generated by BERT is significantly larger than the rest, we don't need to scale it down.

(c) It is not as interpretable as Decision Tree method, but is generally more interpretable than other classification methods.

Hence, I proceed with random forest as the classifier for our task.

Section 3.4

Classification

There are three types of labels in our classification task. A caption is either "funny," "unfunny," or "somewhat funny." We determine a caption's funny level by getting the mode among the responses. If most people think the caption is "funny," the label of the caption will be "funny."

3.4.1. Class Imbalance

However, there is a clear issue with the class imbalance in the dataset. It is not surprising that most of the data regarded as "unfunny" by most online users. In

fact, if we take a look at the distribution of the different funny levels in our sampled dataset, we can see that the level of imbalance is pretty extreme.

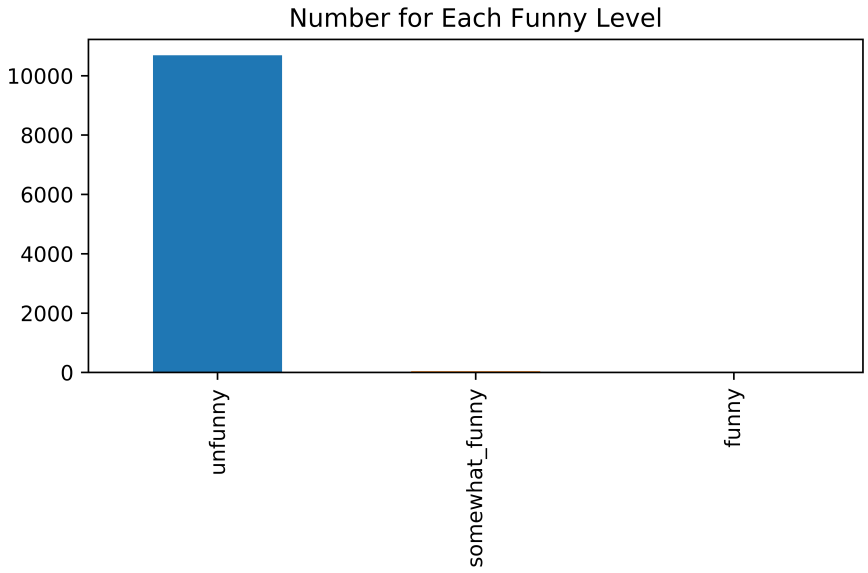


Figure 3.1: **Number of Captions on Different Funny Level**

Although we can barely see any *somewhat funny* and *funny* post, we actually know that there are some of the posts that fit into these two categories. Specifically, the composition is shown in the table below:

unfunny	10688	99.53%
somewhat_funny	45	0.4%
funny	6	0.06%

To effectively train our dataset, I have to combat the imbalance.

3.4.2. Upsampling

Since there are only 6 *funny* posts, we would suffer huge information loss if we down-sample the majority class. Thus, our only way out is to upsample the minority classes. I attempted two methods here:

Resampling. I first split my data into 75% training and 25% testing. Then in the training data, I resample the minority classes with replacement to let them equal the amount of majority classes. Then I trained my random forest model on the training data and tested it on the testing data.

SMOTE. Instead of copying the data in the minority dataset, I decided to use SMOTE[26], or Synthetic Minority Over-sampling Technique, to augment my data in the minority classes. SMOTE augments the minority data in a nearest neighbor fashion. According to its original paper,"the minority class is over-sampled by taking each minority class sample and introducing synthetic examples along the line segments joining any/all of the k minority class nearest neighbors. Depending upon the amount of over-sampling required, neighbors from the k nearest neighbors are randomly chosen." As usual, I started with a train-test-split, and then augment the minorities in the training set using SMOTE. Then, again, I trained my random forest model on the training data and tested it on the testing data.

3.4.3. Result of Upsampling

Recall and Precision. Unfortunately, the upsampling methods didn't improve the performance of the data when the number of estimators is more than 5. And even though I tried a bunch of hyperparameters, none of them have yielded results of none zero precision and recall on the minority classes. Remember that recall is the fraction of true positives over total predicted positives, and precision is the fraction of true positives over total true positives, so ideally, we would want to have none zero values for these two metrics so that we at least have some chance of correct classification. The results (using layer 8 out of 12, which is randomly chosen) are shown in the following charts:

```
unfunny    2684
funny         1
Name: funny_level, dtype: int64
0.9936685288640595
                precision    recall  f1-score   support

         funny      0.000     0.000     0.000         2
somewhat_funny      0.000     0.000     0.000        14
       unfunny      0.994     1.000     0.997      2669

      accuracy                          0.994      2685
     macro avg      0.331     0.333     0.332      2685
  weighted avg      0.988     0.994     0.991      2685
```

Figure 3.2: **Metrics Report when Train on Original Traning Set**

```
unfunny    2685
Name: funny_level, dtype: int64
0.9940409683426443
                precision    recall  f1-score   support

         funny      0.000     0.000     0.000         2
somewhat_funny      0.000     0.000     0.000        14
       unfunny      0.994     1.000     0.997      2669

      accuracy                          0.994      2685
     macro avg      0.331     0.333     0.332      2685
  weighted avg      0.988     0.994     0.991      2685
```

Figure 3.3: **Metrics Report when Train on Resampled Traning Set**

```
unfunny    2685
Name: funny_level, dtype: int64
0.9940409683426443
                precision    recall  f1-score   support

         funny      0.000     0.000     0.000         2
somewhat_funny      0.000     0.000     0.000        14
       unfunny      0.994     1.000     0.997      2669

      accuracy                          0.994      2685
     macro avg      0.331     0.333     0.332      2685
  weighted avg      0.988     0.994     0.991      2685
```

Figure 3.4: **Metrics Report when Train on SMOTE Traning Set**

The above figured explained why the accuracy of the prediction didn't change regardless of our data augmentation. The tree predicted that all the captions are "unfunny" in the case of two upsampling training sets, and only 1 "funny" post in the case of the original dataset. Since in none of the three cases we correctly predicted any minorities class, we might have to shift our method to something else. But before we do that, let's take a look at our tree structures.

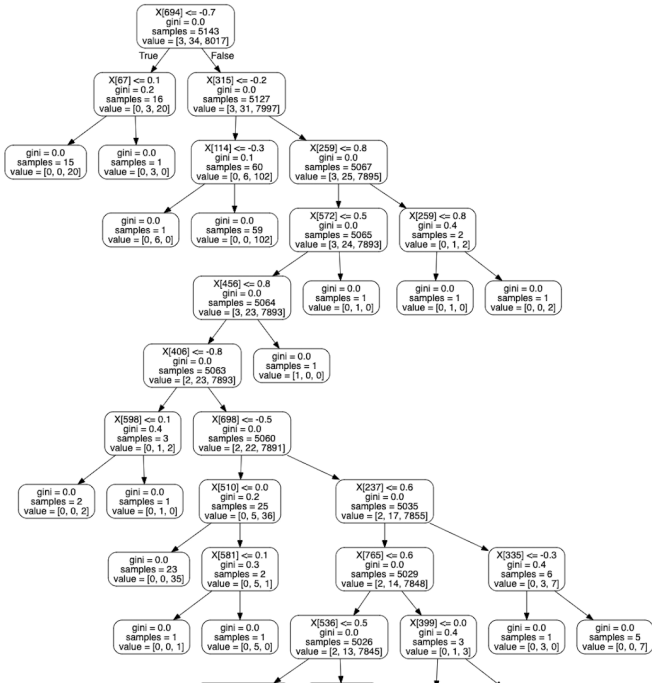


Figure 3.5: **A Peak at the RF trained on Original Traning Set**

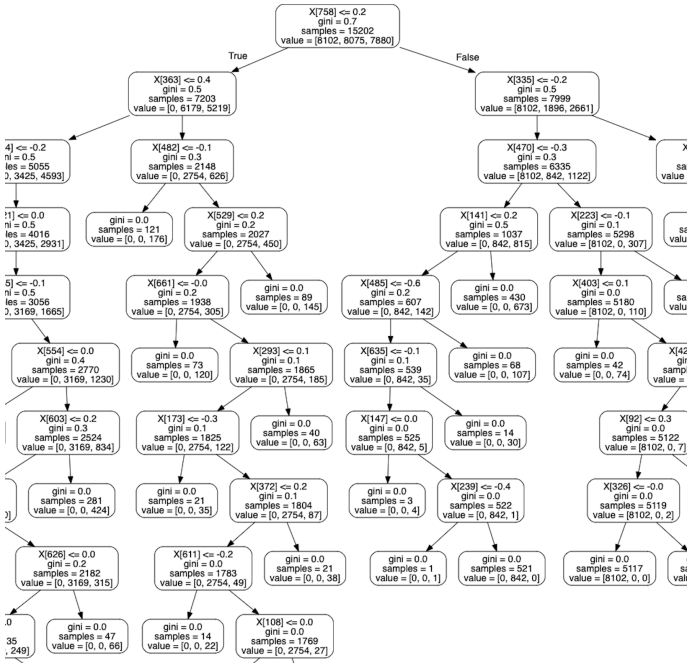


Figure 3.6: **A Peak at the RF trained on Resampled Traning Set**

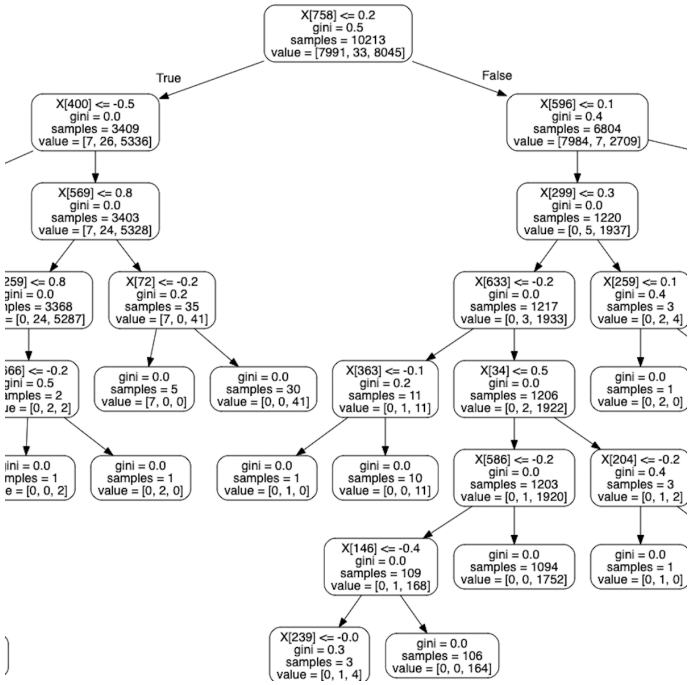


Figure 3.7: **A Peak at the RF trained on SMOTE Traning Set**

Trees Structure. According to these three snippets of three structures of the three situations, I am glad to know that the data upsampling does change the composition of the trained tree. However, this further indicated that we might have to move away from this machine prediction approach due to the lack of minority class. But before we formally say goodbye to our approach, maybe we can do a regression task based on the *score*?

Section 3.5

Regression

The reluctance to start our machine learning task on a regression task lies in the distribution of our predicted value.

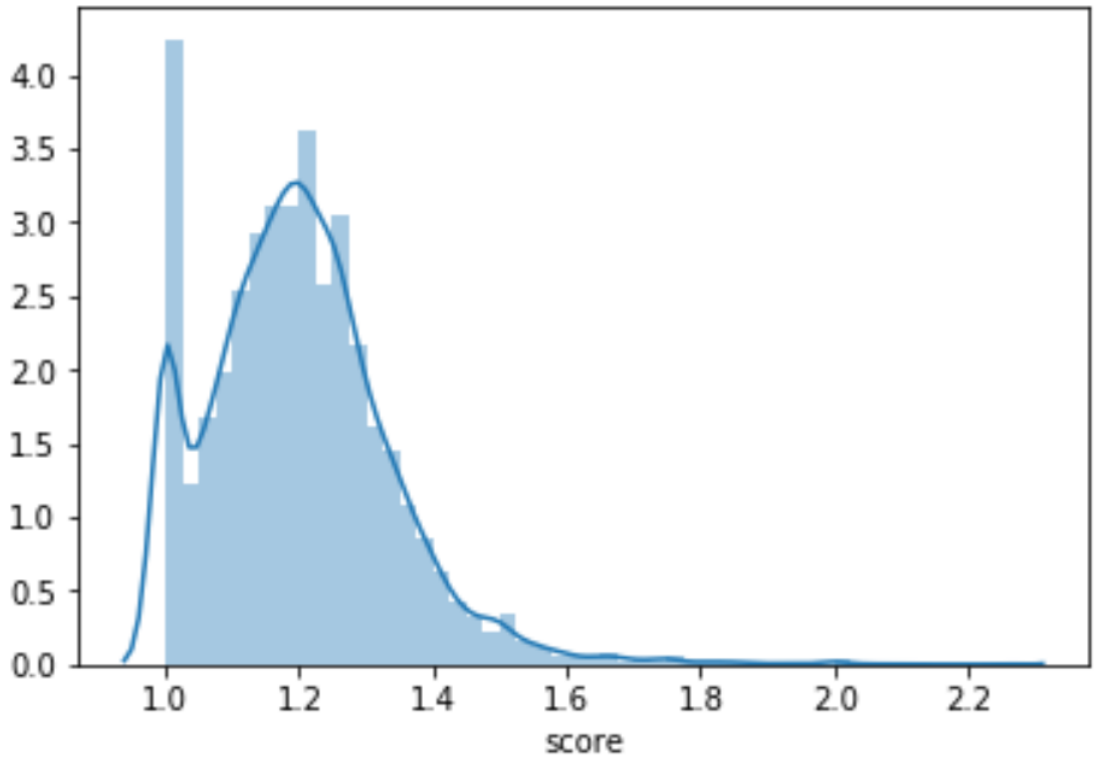


Figure 3.8: **Distribution of the Humor Score**

The distribution of the *funny_level score* is heavily distorted, which is not a surprise consider this score is an average of people's categorization of the funny level of the caption. Since people can only vote 1, 2, or 3, the value of a caption's score is only between 1-3, and a majority of them have an average of 1 - everyone thinks the caption is unfunny. Another problem that we have is the numerical value of the regression target - it is extremely small, which creates numerical difficulties for us to discern our final MSE(min square error) result of the regression.

Nevertheless, I conducted a series of regression tasks, and the difficulties turn out to be true. Here is the result of regression on 32 estimators. There are 12 groups of data, and each represents a model trained on one of the 12 layers. The third value of each group is min square error.

```
Mean Absolute Error: 0.11 point.
Accuracy: 91.14 %.
0.019736949467170414
Mean Absolute Error: 0.11 point.
Accuracy: 91.23 %.
0.01931236385351616
Mean Absolute Error: 0.11 point.
Accuracy: 91.17 %.
0.019527663087014466
Mean Absolute Error: 0.11 point.
Accuracy: 91.17 %.
0.019686288659007765
Mean Absolute Error: 0.11 point.
Accuracy: 91.22 %.
0.019455740845459382
Mean Absolute Error: 0.1 point.
Accuracy: 91.27 %.
0.019238702232210875
Mean Absolute Error: 0.1 point.
Accuracy: 91.3 %.
0.01923732572668897
Mean Absolute Error: 0.11 point.
Accuracy: 91.17 %.
0.019586992395310705
Mean Absolute Error: 0.11 point.
Accuracy: 91.26 %.
0.01945955289063481
Mean Absolute Error: 0.11 point.
Accuracy: 91.23 %.
0.01935402989611361
Mean Absolute Error: 0.11 point.
Accuracy: 91.19 %.
0.01935135353033566
Mean Absolute Error: 0.11 point.
Accuracy: 91.24 %.
```

Figure 3.9: **Regression Result**

It is unfortunate that we can't really choose a better group and nor did meaningfully improved our baseline prediction. It definitely does better on MSE value compare with taking the average of the labels. But the improvement is not really convincing to advance further study.

Chapter 4

BERT Model and VGG19 network

Unfortunately, since the class imbalance problem still hovers around, adding VGG19[13] network didn't help. I used the third from last layers of a pretrained VGG network combine with the BERT layer and conducted a random forest classification. Note that since VGG is trained on ImageNet, which consists of colored pictures of 3 channels instead of the black-and-white cartoons that only have one, I populated the one channel to 3 channels so that VGG19 can be used on my images. The result is still as disappointing as before, as the classifier classifies everything to the category of "unfunny." It seems that before solving the problem of class imbalance, we can hardly move forward to a multi-modal model that can constructively improve our result.

Chapter 5

Limitation and Future Work

There are a couple of improvements I can make to potentially come up with more conclusive results.

(a) According to the original SMOTE paper[26], another technique to boost the upsampling result is simply downsampling the majority class. Moving forward, this is something to be attempted.

(b) Although many classifiers and regressors don't make sense in the current setting, I should've attempted to use them should there be more time. There could be amazing insights in other models that the random forest models somehow failed to detect.

(c) With more computational power and time, I would like to train on the whole dataset that consists of 1 million captions instead of just the 1% of them, even though the smaller dataset has a similar score distribution with the original dataset.

(d) Fine-tuning both BERT and VGG could massively help improve the quality of the embeddings.

Chapter 6

Summary

In this thesis, I explored the possibility of categorizing a cartoon caption's level of fun by leveraging a New Yorker cartoon captions databased that contains more than 100 million ratings and more than one million captions. I attempted to conduct both classification tasks and regression tasks on the dataset, but neither yielded meaningful results due to the extreme class imbalance. To solve the class imbalance, I attempted many data augmentation techniques to upsample my minority classes. Unfortunately, due to the extremely small amount of the minority class, the upsampling method didn't improve the performance of the classification task. The research set out to combine both visual embedding and verbal embedding to generate a multimodal model, but the class imbalance made the added VGG[13] embedding just as futile. The research shows that to generate a cartoon caption by building a discriminative machine learning algorithm and turning it into a generational one might not be practical due to the class imbalance, especially not with my proposed upsampling technique. But we could potentially leverage other non-supervised or semi-supervised learning techniques to achieve this final goal of writing captions for cartoons.

Bibliography

[1] Aynsley, Michael. "How to Write Good Instagram Captions: Tips, Ideas, and Tools." Hootsuite Social Media Management, HootSuite, 17 Sept. 2019, blog. hootsuite.com/instagram-captions-drive-engagement/.

[2] National Endowment for the Humanities. "New-York Tribune. [Volume] (New York [N.Y.]) 1866-1924, November 07, 1920, Page 3, Image 25." News about Chronicling America RSS, New York Tribune, 2012, chroniclingamerica.loc.gov/lccn/sn83030214/1920-11-07/ed-1/seq-25/.

[3] "Your Caption Here." The New Yorker, Condé Nast, 25 Apr. 2005, www.newyorker.com/magazine/2005/05/02/your-caption-here.

[4] Adams, Russell. "How About Never-Is Never Good for You? Celebrities Struggle to Write Winning Captions." The Wall Street Journal, Dow Jones & Company, 10 Sept. 2011, www.wsj.com/articles/SB10001424053111904199404576538831896448132.

[5] "Cartoon Caption Contest." The New Yorker, www.newyorker.com/cartoons/contest#thisweek.

[6] "Cartoon Caption Contest." The New Yorker, www.newyorker.com/cartoons/contest#winner.

[7] "SEVEN-TIME New Yorker Caption Contest Winner Larry Wood Shares His Insights." Cartoon Collections Blog, 21 Dec. 2018, more.cartooncollections.com/lawrence-wood-caption-contest-winner/.

[8] Watercutter, Angela. "Cracking the Code of The New Yorker's Cartoon Caption Contest." Wired, Conde Nast, 3 June 2017, www.wired.com/2011/05/new-yorker-cartoon-caption-contest/.

[9] Nextml. "Nextml/Caption-Contest-Data." GitHub, 28 Jan. 2020, github.com/nextml/caption-contest-data.

[10] "NEXT." NEXT, nextml.org/.

[11] Thevar, Rajesh. "RajeshThevar/Funny-Caption-Generation." GitHub, Aug. 2018, github.com/RajeshThevar/Funny-caption-generation.

[12] Mikolov, Tomas; Sutskever, Ilya; Chen, Kai; Corrado, Greg S.; Dean, Jeff (2013). *Distributed representations of words and phrases and their compositionality.* Advances in Neural Information Processing Systems. arXiv:1310.4546

[13] K. Simonyan, A. Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition.* International Conference on Learning Representations, 2015

[14] Klaus Greff; Rupesh Kumar Srivastava; Jan Koutník; Bas R. Steunebrink; Jürgen Schmidhuber (2015). *"LSTM: A Search Space Odyssey".* IEEE Transactions on Neural Networks and Learning Systems. 28 (10): 2222–2232.

[15] Prince, Rachel, and Dragomir Radev. "Humorous Caption Generation for New Yorker Cartoons." Semantic Scholar, corpus no. 202549495, 2017

[16] Anbarasu, Hari. "Humor Detection in New Yorker Cartoon Captions." Yale Lily, https://yale-lily.github.io/public/harianbarasu.pdf

[17] Devlin, Jacob; Chang, Ming-Wei; Lee, Kenton; Toutanova, Kristina (11 October 2018). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". arXiv:1810.04805v2 [cs.CL].

[18] Polosukhin, Illia; Kaiser, Lukasz; Gomez, Aidan N.; Jones, Llion; Uszkoreit, Jakob; Parmar, Niki; Shazeer, Noam; Vaswani, Ashish (2017-06-12). "Attention Is All You Need". arXiv:1706.03762 [cs.CL]

[19] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, Luke Zettlemoyer. "Deep contextualized word representations". arXiv:1802.05365

[20] Andrew M. Dai, Quoc V. Le. "Semi-supervised Sequence Learning." arXiv:1511.01432

[21] "Bert-Base-Uncased · Hugging Face." Hugging Face – On a Mission to Solve NLP, One Commit at a Time.,
huggingface.co/bert-base-uncased.

[22] Bin Wang, Student Member, IEEE, and C.-C. Jay Kuo "SBERT-WK: A Sentence Embedding Method by Dissecting BERT-based Word Models." JOURNAL OF LATEX CLASS FILES, VOL. 14, NO. 8, AUGUST 2020

[23] Ho, Tin Kam (1995). *Random Decision Forests*. Proceedings of the 3rd International Conference on Document Analysis and Recognition, Montreal, QC, 14–16 August 1995. pp. 278–282.

[24] Quinlan, J. R. (1986). "Induction of decision trees" (PDF). Machine Learning. 1: 81–106. doi:10.1007/BF00116251

[25] Cortes, Corinna; Vapnik, Vladimir N. (1995). "Support-vector networks". Machine Learning. 20 (3): 273–297. CiteSeerX 10.1.1.15.9362. doi:10.1007/BF00994018.

[26] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, W. Philip Kegelmeyer. "SMOTE: Synthetic Minority Over-sampling Technique" Journal of Artificial Intelligence Research 16 (2002), 321 – 357.