ECE324 Final Report

Andrew Wang 1003259305
Murtaza Latif 1004307065

Word Count: 2000
Penalty: 0%

Permission to post video: Wait until video seen
Permission to post final report: Yes
Permission to post source code: No

Introduction

Reddit is a platform where users can submit, view and vote on posts. Posts are prioritized by score (calculated as "upvotes minus downvotes") and organized in subsites called "subreddits".

We specifically focused on the subreddit "AskReddit" where posts are text-only questions. Our goal is to create a neural-net model to classify an AskReddit post into score categories based on its content and context. In particular, we focus on binary, ternary and quinary classification as seen below:

Table 1: *Class score boundaries for binary, ternary and quinary score classification*

Problem	**Class Score Boundaries and Names**				
Binary	0 - 99 "Low"	100+ "High"			
Ternary	0 - 1 "Low"	2 - 499 "Average"	500+ "High"		
Quinary	0 "Zero"	1 - 9 "Low"	10 - 99 "Average"	100 - 999 "High"	1000+ "Viral"

Neural networks are well-suited to this task given their popularity and success in natural language processing, the availability of data and data collection methods (scraping APIs), and the intrinsic labelling in posts (score).

The classifier would provide a means for estimating post/advertisement popularity without requiring users to publish content. Used as an early testing method, our classifiers could then help industry and users alike with improving their advertising or post quality.

Illustration / Figure

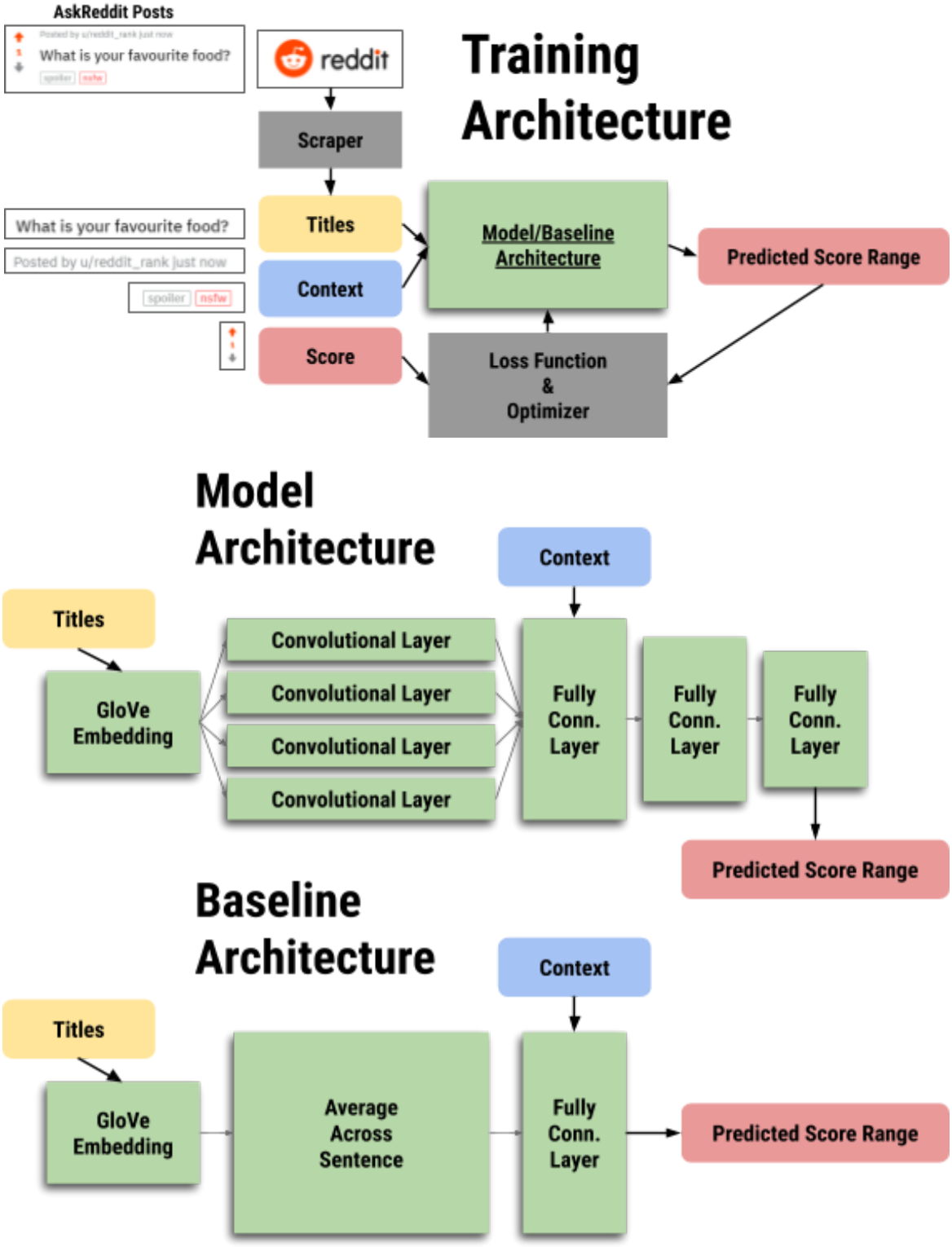

Figure 1: *Top To Bottom: High-level overview of data collection and training architecture, Model Architecture, Baseline Architecture*

Background & Related Work

One similar project compares the performance of Reddit comment score predictors [1]. The article outlines data collection processes before making comparisons between regression and tree models. Their project context is slightly different, as comment scores are often influenced by surrounding comments, whereas new post scores are independent from other posts. Additionally, they tackle regression (exact score) whereas we have scoped to classification. Regardless, a useful aspect of their work (which we borrowed) was the selection of features as input data. Since they also worked with Reddit, most of their analysis on features is transferable to our context.

Another related work is a thesis titled *Popularity Prediction of Reddit Texts* [2]. The primary takeaway is the challenges of classifying popularities in a forum where the culture is always changing. The paper presents many prediction hurdles, including a variety of variables that affect score regardless of post content. This work allowed us to identify important context components such as the time and day of posting, and user selected tags ("nsfw", "serious", "spoiler").

Data and Data Processing

Raw AskReddit post data is accumulated using the Pushshift.io API. It offers options to scrape posts within certain score ranges and submission dates. Using this, we easily populate our class ranges and ensure that all posts are a minimum 2 days old at the time of collection. An example of raw, scraped posts follows:

```
id,title,over_18,created_utc,link_flair_text,score
e02wdb,am i [F16] the redditor?,False,1574438321,,0
dz8nip,What loopholes do you take advantage of?,False,1574285777,,0
dzjkra,What is wrong with our society today?,False,1574344205,,0
dz8nff,"Blind people of reddit, what do you see?",False,1574285764,,0
dz8nzw,"People who dislike videos/posts, why do you do it?",False,1574285832,,0
```

Figure 2: *Sample rows of the raw dataset for the quinary classifier for posts that fall under the score class of 0 (zero score).*

After collection, we process and filter the data to a format that better suits our model. The filters applied to the dataset are the following:

Table 2: *Types and description of filters applied to the input data*

Category	Description	Example
String Manipulation	Lowercase	"Hello" -> "hello"
	Replace numbers with nearest order of magnitude	"123" -> "100"
	Replace symbols	"$" -> "dollars"
	Replace common Reddit phrases/acronyms	"r/..." -> "subreddit" "SO" -> "significant other" "OP" -> "original poster"
Post Removal	Remove duplicate inputs	posts with the same id
	Remove posts that cannot be made by users	moderator posts
Data Enhancement	Convert UNIX timestamp into one hot encoded hour vector	"13416434124" -> "001000..." (3:00 AM)
	Convert UNIX timestamp into one hot encoded day vector	"13416434124" -> "1000000" (Sunday)

After the filters are applied, the datasets are balanced so each score class has an equal number of samples. Some samples of the processed dataset are as shown in the following figure:

```
1  title,over_18,serious_flair,score,hour_0,hour_1,hour_2,hour_3,hour_4,hour_5,
   hour_6,hour_7,hour_8,hour_9,hour_10,hour_11,hour_12,hour_13,hour_14,hour_15,
   hour_16,hour_17,hour_18,hour_19,hour_20,hour_21,hour_22,hour_23,weekday_0,
   weekday_1,weekday_2,weekday_3,weekday_4,weekday_5,weekday_6
2  what is a universal truth that no-one can deny?,0,0,4,0,0,0,0,0,1,0,0,0,0,0,0,
   0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1
3  "quarantined subreddits don't show up in search bar, any idea how to visit
   those subs on mobile?",0,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
   0,0,0,1,0,0
4  "what's one thing you never want to experience in your lifetime, but probably
   will?",0,0,4,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1
5  golf caddies of reddit - do you overhear a lot of business gossip? if so what's
   the juiciest thing you've witnessed/overheard?,0,0,4,0,0,0,0,0,0,0,0,1,0,0,0,
   0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0
6  "what do people say say wrong, like 'vicea versa' instead of 'vice versa' that
   annoys you?",0,0,4,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0
```

Figure 3: *Sample rows of the cleaned dataset for the quinary classifier. Indices were included on the left hand side of the figure to help distinguish the different data samples due to the large quantity of parameters.*

The statistics obtained include lists of frequencies for the various attributes. In **Figure 4**, the histograms for the frequencies of posts over each weekday and hour are shown. The visualized data allows us to easily identify biases, such as the higher number of posts on Friday for this particular dataset. We also determined that the average length of a title was 8 words, hence we decided not to create a recurrent neural net based model. Additionally, we ignored the "spoiler" tag since we found that a negligible number of posts had it set.

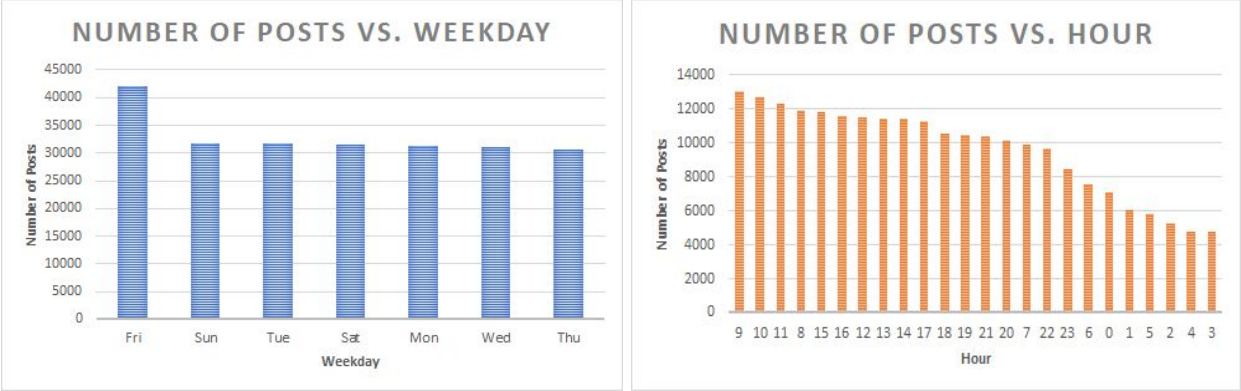

Figure 4: *Histograms showing the number of posts made in any particular weekday and hour.*

Architecture

Our final model consists of GloVe word embedding followed by a convolutional neural network (CNN) and then a multi-layer perceptron (MLP). This same architecture was used for all three classification problems, but with a different number of MLP output layer neurons:

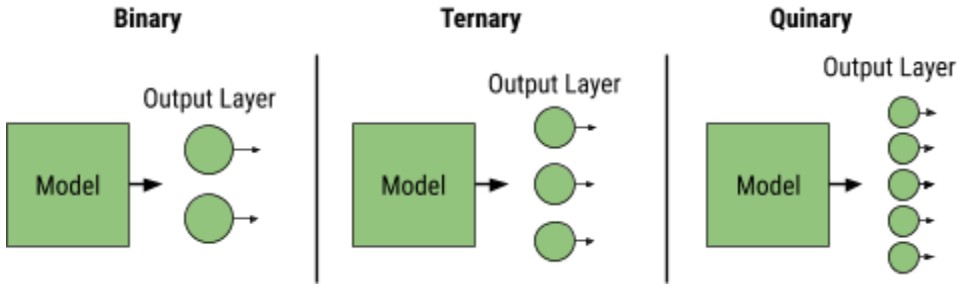

Figure 5: *Depiction of number of output neurons in output layer of MLP depending on number of prediction classes. From left to right, 2, 3 and 5 classes.*

The GloVe embedding layer produces (1, 100) word vectors. The CNN uses 4 parallel convolutional layers with outputs concatenated together. The layers have kernel sizes (k, 100), with k = (1, 2, 3, 4) per layer respectively. They, in essence, scan across titles for phrases of length k words. All convolutional layers use ReLu activation and max-pooling across the title. Batch normalization and dropout are applied before concatenation with the one-hot encoded categorical features. The final "with-context" vector is passed to a three layer MLP. The MLP uses a Leaky ReLu activation function, batch normalization and dropout on the two non-output layers and a Softmax output function.

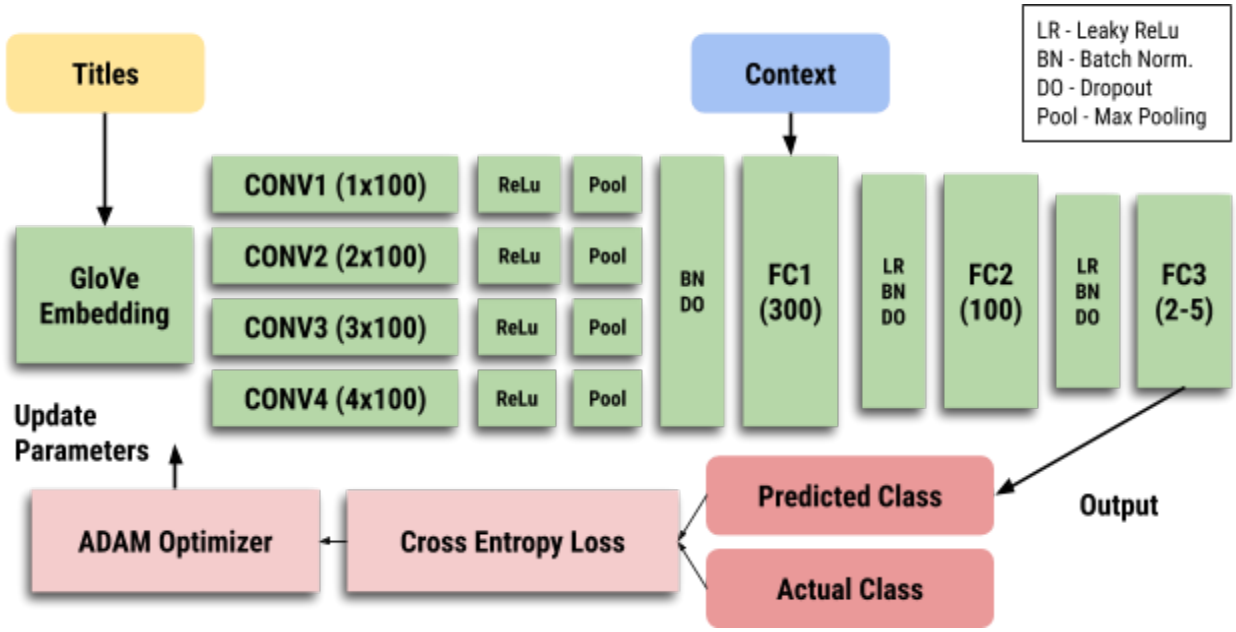

Figure 6: *Full model architecture with flow from left to right if not otherwise indicated. CNN kernel sizes and MLP layer sizes indicated in brackets per module.*

Both this model and the following baseline model are trained on batches using "Cross Entropy Loss" and the "ADAM Optimizer".

Baseline Model

The baseline model follows the same structure as above. It uses a GloVe embedding layer, but then averages the word vectors across the title. As before, context is concatenated before passing it to an MLP. The MLP uses two layers with sigmoid activation on the input layer and a softmax output function.

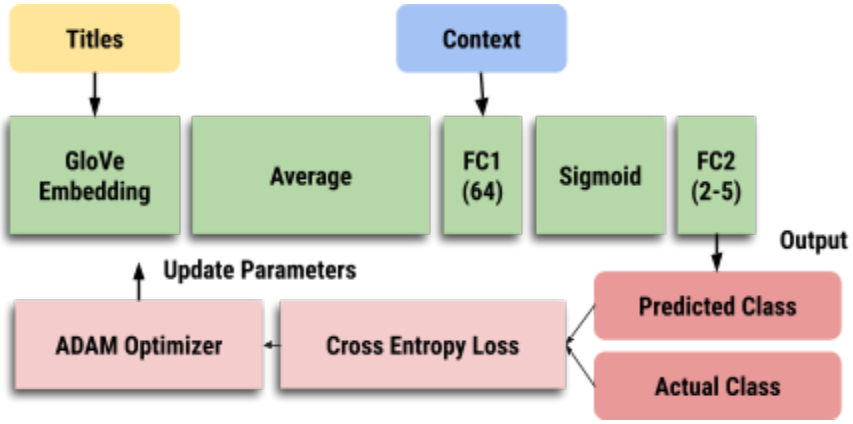

Figure 7: *Baseline model architecture with flow from left to right if not otherwise indicated. MLP layer sizes indicated in brackets per module.*

Quantitative Results

The following hyperparameters were used for training the baseline model:

Table 3: *Baseline model hyperparameters*

Classification Problem	Epochs	Batch Size	Embedding Dimension	FC1 Size	FC2 Size
Binary	100	10,000	100	64	2
Ternary	100	10,000	100	64	3
Quinary	100	10,000	100	64	5

To determine the hyperparameters used in the CNN architecture, a grid search was conducted manually. The following hyperparameters were the final selections used to train the CNN models.

Table 4: *CNN Model hyperparameters*

Classification Problem	Binary	Ternary	Quinary
Epochs	35	35	35
Batch Size	10,000	10,000	10,000
Embedding Dims	100	100	100
Conv1 Kernels	300	300	700
Conv2 Kernels	200	200	600
Conv3 Kernels	100	200	400
Conv4 Kernels	100	100	300
Convolution Dropout	0.1	0.2	0.1
FC1 Size	300	300	300
FC1 Dropout	0.5	0.5	0.6
FC2 Size	100	100	100
FC2 Dropout	0.5	0.5	0.5
FC3 Size	2	3	5

Training/validation accuracy graphs were created for each classification problem. Note, for the ternary and quinary case, models were overtrained to 100 epochs to illustrate the models' ability to reach high training accuracies. Model parameters were saved at 35 epochs.

Binary Classification

Baseline CNN Model

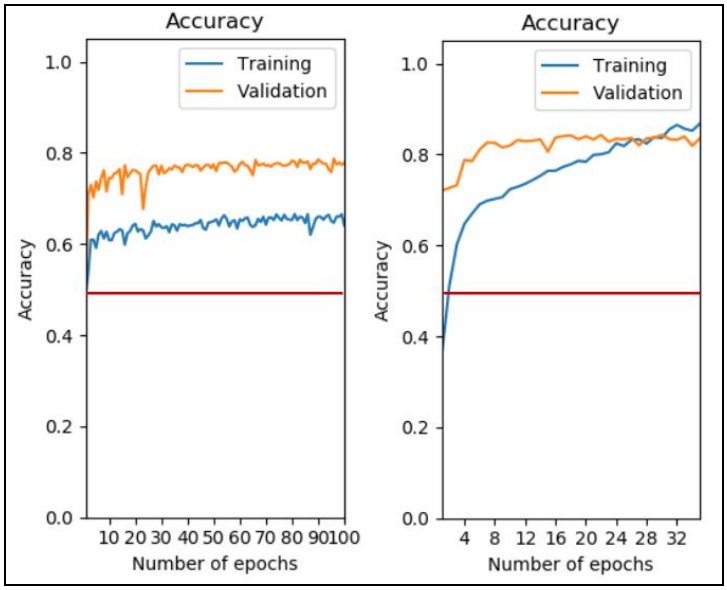

Figure 8: *Training/validation accuracy graphs for the Baseline and CNN Model respectively in the binary classification problem. A random guess has expected accuracy 50% (red line).*

Ternary Classification

Baseline CNN Model

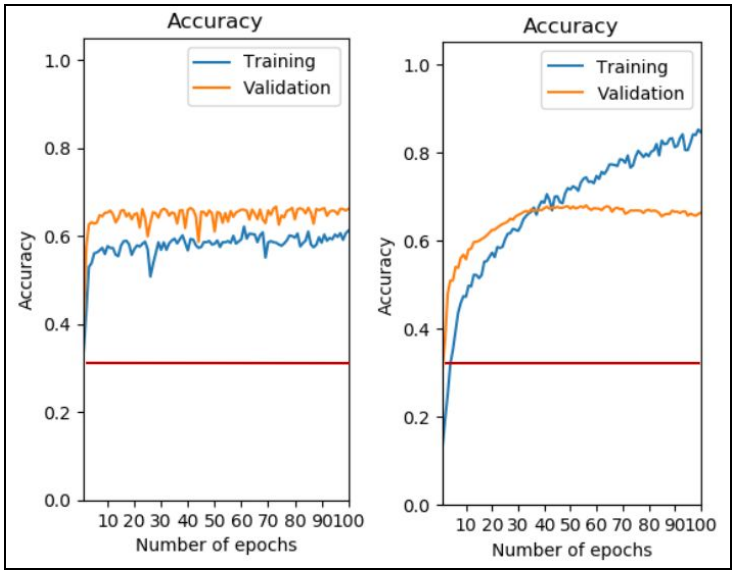

Figure 9: *Training/validation accuracy graphs for the Baseline and CNN Model respectively in the ternary classification problem. A random guess has expected accuracy 33.33% (red line).*

Quinary Classification

Baseline CNN Model

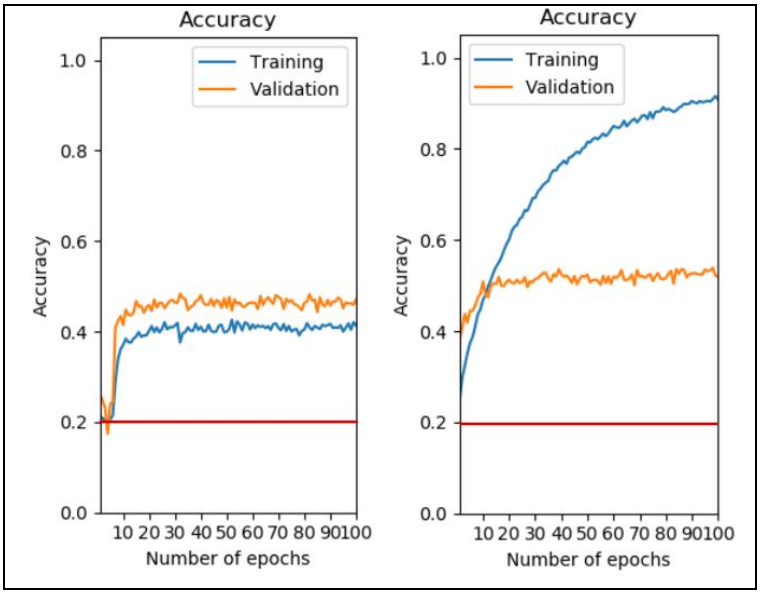

Figure 10: *Training/validation accuracy graphs for the Baseline and CNN Model respectively in the quinary classification problem. A random guess has expected accuracy 20% (red line).*

The final validation and test accuracies are summarized in the following graphs. A random guesser is also included for comparison.

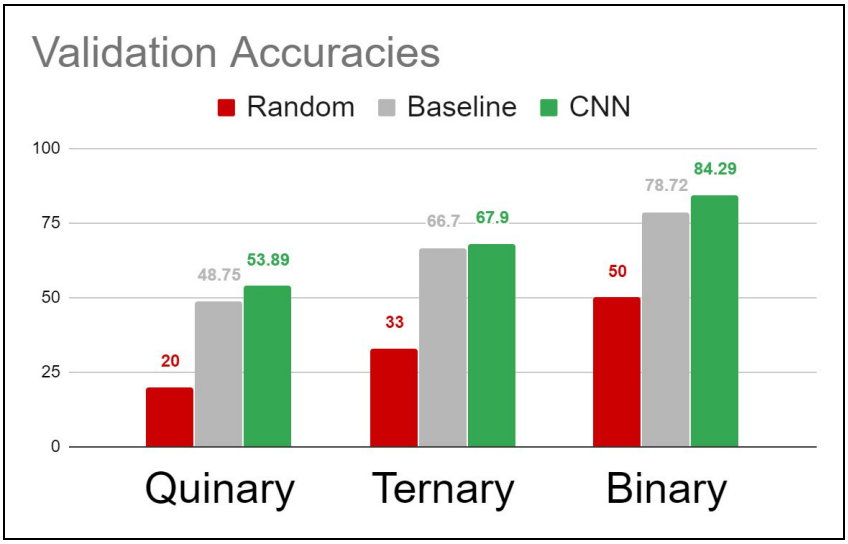

Figure 11: *Final validation accuracies for a random guess (expected), the baseline model and the CNN model for the three classification problems*

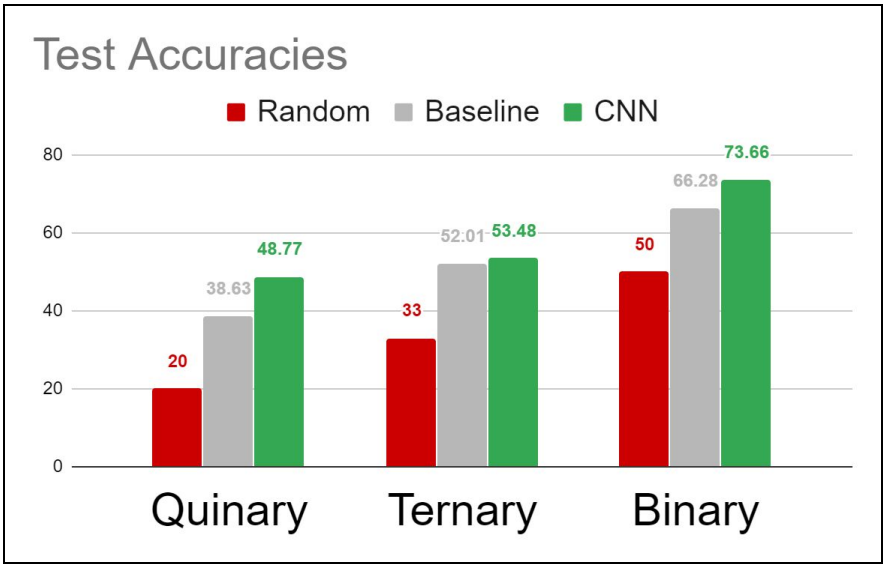

Figure 12: *Final test accuracies for a random guess (expected), the baseline model and the CNN model for the three classification problems*

The following tables are confusion matrices for each classifier:

Table 5: *Confusion matrix for binary classification. "Low" corresponds to scores in [0, 99] and "High" to scores in [100+]*

Binary	Truth Low	Truth High
Prediction Low	13104	5113
Prediction High	4389	13466

Table 6: *Confusion matrix for ternary classification. "Low" corresponds to scores in [0, 1], "Average" to scores in [2,499] and "High" to scores in [500+].*

Ternary	Truth Low	Truth Average	Truth High
Prediction Low	4173	1066	541
Prediction Average	890	3352	1189
Prediction High	411	914	3279

Table 7: *Confusion matrix for quinary classification. "Zero" corresponds to scores in [0], "Low" to scores in [1, 9], "Average" to scores in [10, 99], "High" to scores in [100, 999] and "Viral" to scores in [1000+].*

Quinary	Truth Zero	Truth Low	Truth Avg.	Truth High	Truth Viral
Prediction Zero	1981	311	163	399	136
Prediction Low	1410	1966	731	301	611
Prediction Average	550	851	3370	1009	442
Prediction High	636	1324	506	2566	605
Prediction Viral	281	620	133	1283	3053

We value the recall metric with high importance because it factors in how often we predict a high-scoring post to have a low score. This is the worst case because the user may be dissuaded from posting good content. For classifiers with more than two classes, we take the rate of correctness for our model when the post's ground-truth score is high (and viral for quinary classifier) and denote this metric as the "True High Rate" (THR). The following table organizes these metrics for each classifier.

Table 8: *Recall and THR values per classification problem. Essentially, how often is the model correct when the ground truth score of the post is high.*

Classification Problem	Random Guess Recall/THR	CNN Recall/THR
Binary	50%	72.48%
Ternary	33.33%	65.46%
Quinary	20%	51.16%

The following table summarizes our results:

Table 9: *Final summary of model results*

Classification Problem	Random Guess	Baseline		CNN		
		Validation Accuracy	Test Accuracy	Validation Accuracy	Test Accuracy	Recall or THR
Binary	50%	78.72%	66.28%	84.29%	73.66%	72.48%
Ternary	33.33%	66.7%	52.01%	67.9%	53.48%	65.46%
Quinary	20%	48.75%	38.63%	53.89%	48.77%	51.16%

Qualitative Results

Sample outputs of the model can be evaluated using our website UI, designed to look identical to Askreddit's post submission page.

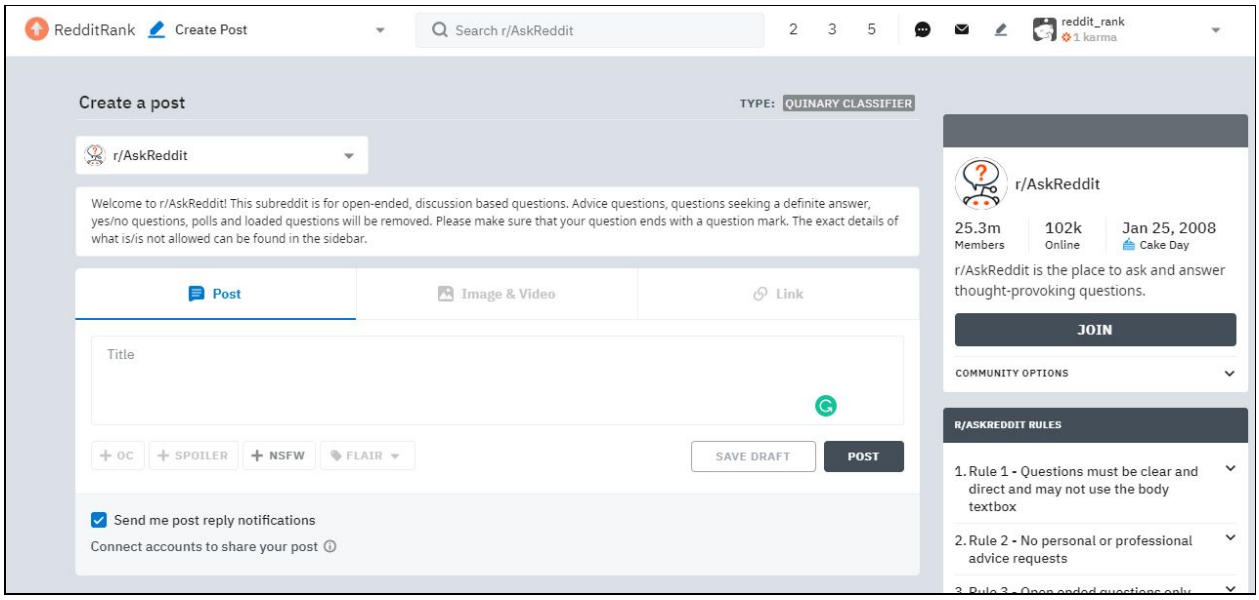

Figure 13: *Website interface for evaluating model outputs. The title of the post is input into the text area in the middle, and the maturity rating context can be added directly below the textbox. The classifier can be selected at the top, with the choice between a 2, 3 and 5 class model. The model is evaluated when the 'POST' button is clicked.*

The model's input also requires the hour and weekday context, which is taken directly from the device running the interface before conversion to a global (UTC) hour and weekday.

Figure 14: *The input into our website predictor, showing the title as "What's your favourite food?" and the NSFW context tag selected.*

One sample input we tested was a basic question with NSFW flag on, as seen above. Normally, this tag is used when the thread contains mature content.

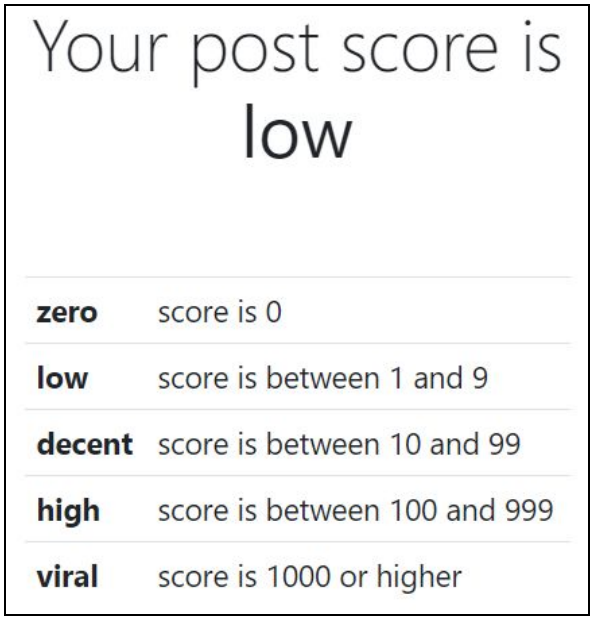

Figure 15: *The prediction corresponding to the input shown above in* ***Figure 14****. The post was predicted to have a 'low' score, which is classified as a post with a score between 1 and 9.*

In the quinary classifier, the post was predicted to have a score of between 1 and 9 as shown in **Figure 15**. We posted this on the actual subreddit [3]. After 2 days, the thread had no new activity and the score steadied.

Figure 16: *Status of the Reddit post made on AskReddit made with the exact same input parameters as the prediction. The score of the post after a couple days was 4.*

Despite the odd context of the input, the thread received a score of 4, in agreement with our prediction.

There are several types of posts that our model consistently predicts correctly. Titles without a question mark or consisting of incomprehensible words are correctly predicted as low scoring. This is entirely expected, as the purpose of AskReddit posts are to post questions. Take the top scoring set of posts below for example:

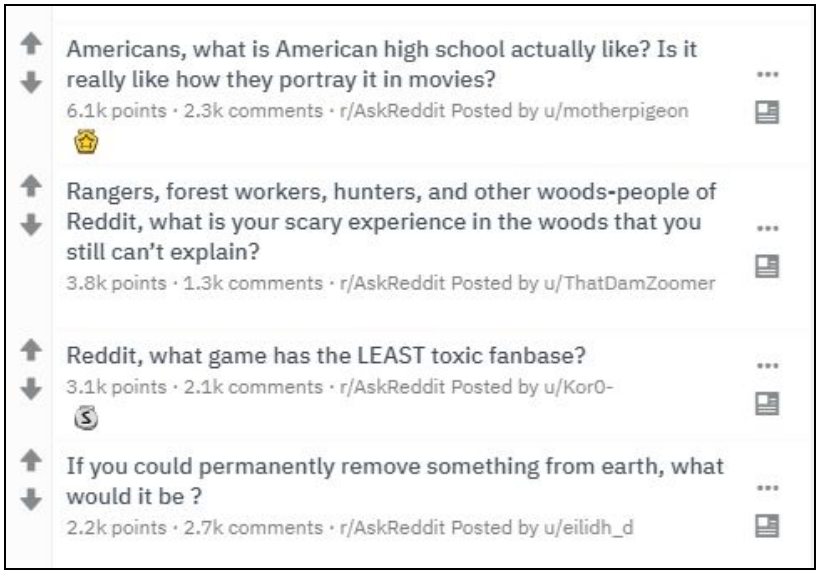

Figure 17: *Set of 4 hot posts on the AskReddit subreddit, each featuring a question mark at the end of their titles.*

When predicting posts we know are popular, often the result is incorrect. This can be attributed to the fact that the same content (question), posted multiple times usually has a singular highly scored post followed by multiple low-scored repeats. See the following post for example:

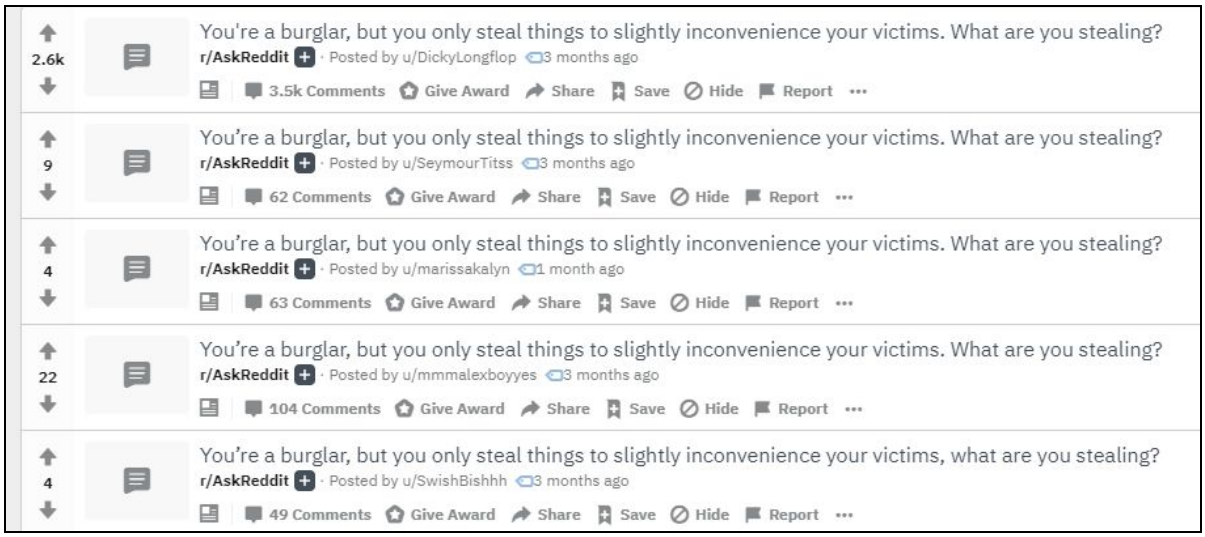

Figure 18: *The same post title made with different contexts, with only one of the posts having a high score and the rest having very low scores.*

A low predicted score is reasonable, as only 1 of the posts with the same title received a high score.

Discussion and Learnings

First consider the accuracy graphs. We can see that the Baseline Model and CNN are capable of training for all three problems given their training accuracy curves.

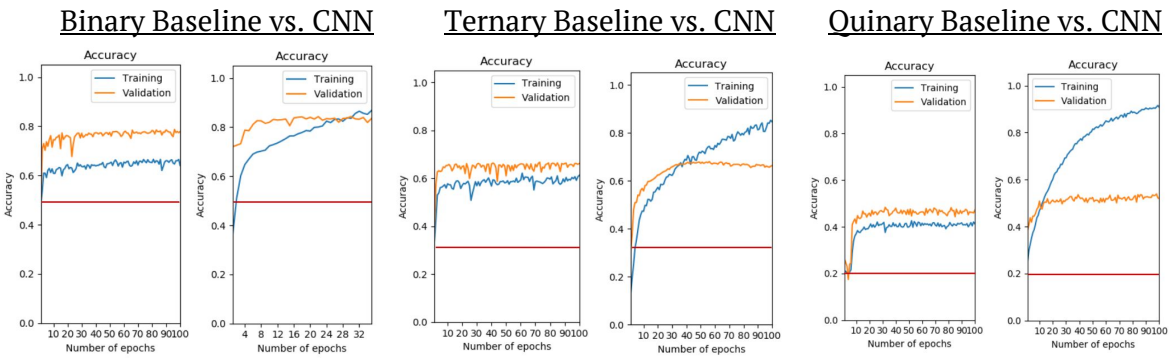

Figure 19: *Repeated accuracy graphs. This figure's purpose is to see the shape of the accuracy curves. See figures 8, 9, 10 for full size renders.*

For validation accuracy, both models easily beat the expected accuracy of a random guess and the CNN consistently beats the Baseline. We note that the final training accuracy is 30% and 50% higher than the validation accuracy for the ternary and quinary problems respectively. This indicates that our model overfits. To combat this overfitting, we learned and applied several techniques including batch normalization, dropout and early stopping.

We conclude our model has generalized given its ability to classify score more accurately than both the Baseline and a random guess on data it has never seen before. The corresponding confusion matrices (**Table 5, 6, 7**) similarly reflect this conclusion, with high values along the diagonals and increasingly smaller values away from it. This additionally indicates that our model rarely confuses high scoring posts for low scoring ones and vice-versa. Looking at the THR metric (**Table 8**) we again see our classifier vastly outperforms a random guess.

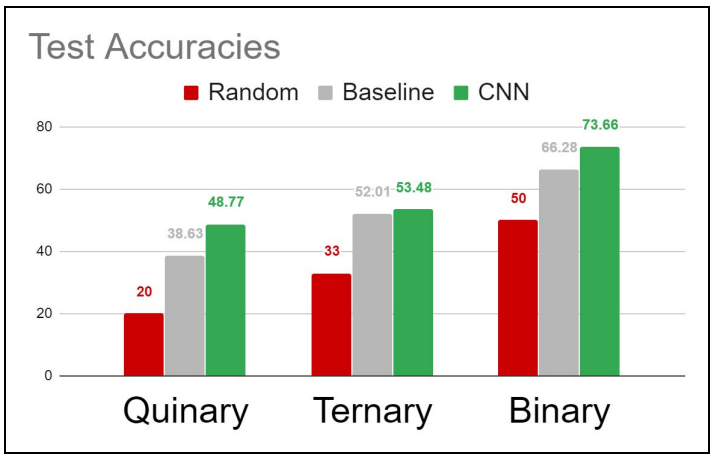

Figure 20: *Repeated summarized test accuracy results.*

As expected, we see that our model's absolute test accuracies (**Figure 20**) improve as we move from more classes to fewer classes (quinary 48.77%, ternary 53.48%, binary 73.66%). But, surprisingly, the models' test accuracies relative to a random guess decrease! The quinary, ternary and binary classifiers are 2.5x, 1.65x and 1.45x more accurate than a random guess respectively. From this lens, the model actual performs better (relative to guessing) as the classification problem is more complex. Perhaps, each class has "easily learnable" and "complicated" characteristics with respect to predicting score. Then, it would be easier for a model to achieve a low accuracy per class across multiple classes than a high accuracy on a singular class.

For future work, we hope to increase our dataset size to further reduce overfitting. Given that we scraped all AskReddit posts (bottlenecked by posts over 1000 score), we could achieve this using data augmentation (e.g. using synonyms for sentences). Second, we would also begin training with GPUs. By employing GPU processing we were able to see speed ups of 400-500%, drastically saving time.

Ethical Framework

The stakeholders of this model are Reddit's users ("redditors"), advertisers, the company (Reddit), and model creators. We consider scenarios in which our model is accurate and used widely.

Foremost, advertisers can predict to what degree their advertisements will be appreciated by the Reddit community (+beneficence). Smaller businesses and studios also equally have this opportunity. This reduces the advantages larger businesses have due to their larger budgets for marketing and advertisement analysis (+justice). In the same vein, redditors can equally improve their grasp on what content is "popular" through trial and error using the predictor (+justice).

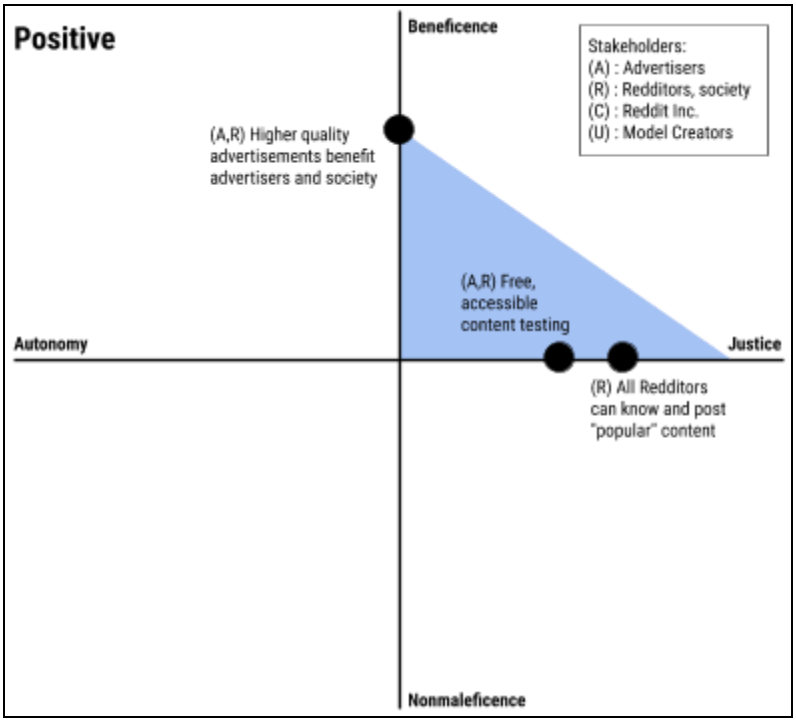

Figure 21: *Positive reflexive principlism graph*

Conversely, we consider negative consequences in terms of the four ethical principles. Redditors will post content less frequently because low score predictions may dissuade them from submitting their posts (-nonmaleficence). This decreases the amount of content posted overall which negatively affects users' browsing experiences and hence, the site's traffic (-nonmaleficence).

Since Reddit is a dynamic platform (where "what's popular" changes), the model must be frequently retrained to maintain accuracy. This invokes a computing cost for the creators and is an economic burden (time/money), and an environmental waste (-nonmaleficence). If the

model algorithm is not open source, biases (unconscious or conscious) may be introduced, skewing prediction results and potentially influencing certain types of posts (-nonmaleficence). An example is predicting left-leaning content to have a lower score, effectively reducing posts of that opinion.

The "RedditClassifier" project (Agnihotri, Mogilny 2019) has created a model to classify which Subreddit a given post belongs to. Utilizing both our models as discriminators for a Generative Adversarial Network (GAN), one could create a GAN capable of generating high-scoring posts for a particular subreddit. With a good enough GAN and enough generated post submissions, a user may see only artificial posts whilst browsing a subreddit. In this way, the GAN's creators could direct what users see on Reddit, resulting in a loss in redditor autonomy (-autonomy). Such a GAN could create false reports, politically skewed posts, etc. in large enough quantities to affect the reddit user base mentality (-nonmaleficence).

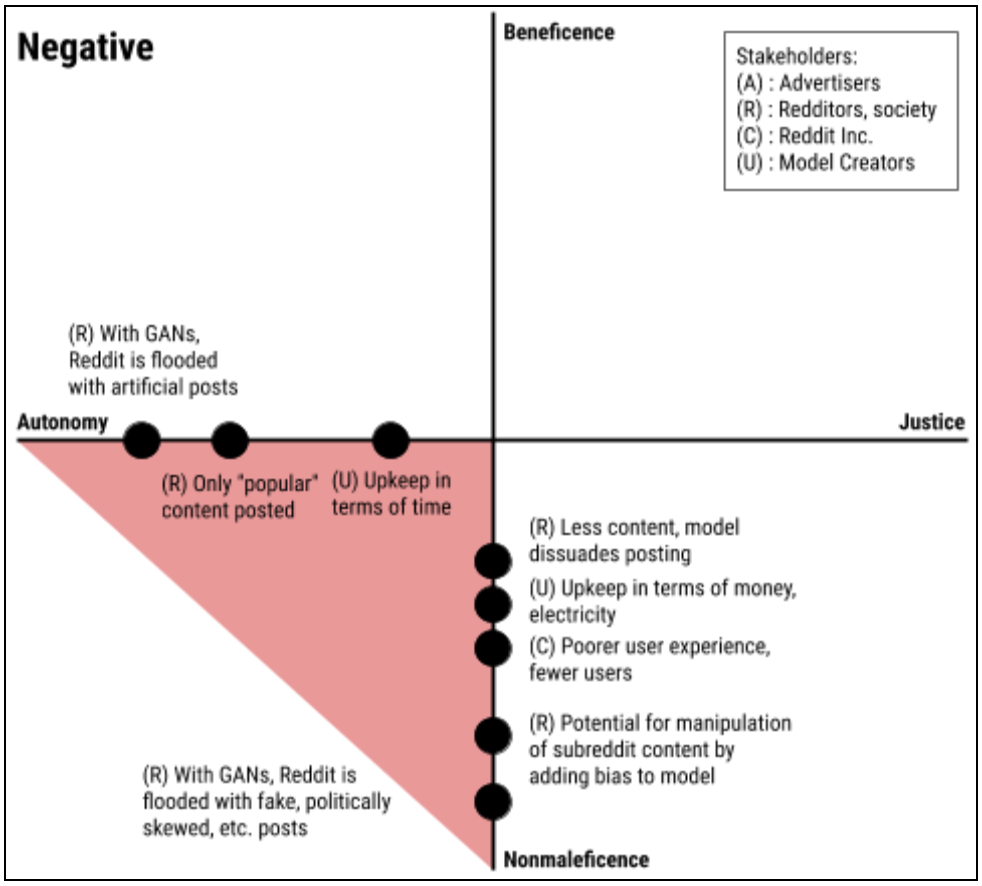

Figure 22: *Negative reflexive principlism graph*

Project Difficulty & Model Complexity

To predict the score of a Reddit post is intrinsically difficult. First, the culture of the Reddit community is always changing. Second, the context of a post is complex, touching on factors like submission time, author post history, current Reddit trends, and even current societal trends.

Given the complexity of our problem, our model performs with a good accuracy. All three of our classifiers perform significantly better than a random guess (**Figure 20).** To get our models to this level of accuracy required techniques outside of the scope of the course. Chiefly, dropout was used to decrease the degree of overfitting and Leaky ReLu activation was used to combat the vanishing gradient problem (given our large neural network).

References

[1] A. Reevesman, “Predicting Reddit Comment Upvotes with Machine Learning,” Medium, 09-Jan-2019. [Online]. Available: https://towardsdatascience.com/predicting-reddit-comment-karma-a8f570b544fc

[2] T. Rohlin, “Popularity Prediction of Reddit Texts,” n.d.. [Online]. Available: https://scholarworks.sjsu.edu/cgi/viewcontent.cgi?article=8251&context=etd_theses.

[3] “r/AskReddit - What's your favourite food?,” reddit. [Online]. Available: https://www.reddit.com/r/AskReddit/comments/e2z8k4/whats_your_favourite_food/.