Permissions:

- permission to post video: **wait till see video**
- permission to post final report: **yes**
- permission to post source code: **yes**

ECE324

Where2Spreddit Project Final Report

Jason Li - 1004372813
Devansh Ranade - 1004221399

Due: 3 December 2019

Word Count: 1862

1 Introduction

Reddit is a website where users are free to openly share content and have discussions on the internet. Organized into communities called subreddits, Reddit can be thought of as a public internet forum to serve people with similar interests.

The goal of Where2Spreddit is to train a model for text classification. The model takes as input a Reddit text post and predicts the subreddit where it was originally posted (or should be posted). The idea of classifying text based on a theme is useful for a diverse applications. For example, such a model may be able to identify people with similar interests by analyzing one's online posts and introducing alike people to become friends.

In recent years, the use of machine learning has greatly advanced the technology of text classification, and the use of NNs is very effective in the field [1]. Furthermore, NNs provide the ease of scalability as vast amounts of labelled text is available from the internet, and the classification of text can be expanded upon without the need for feature engineering.

2 Project Structure Illustration

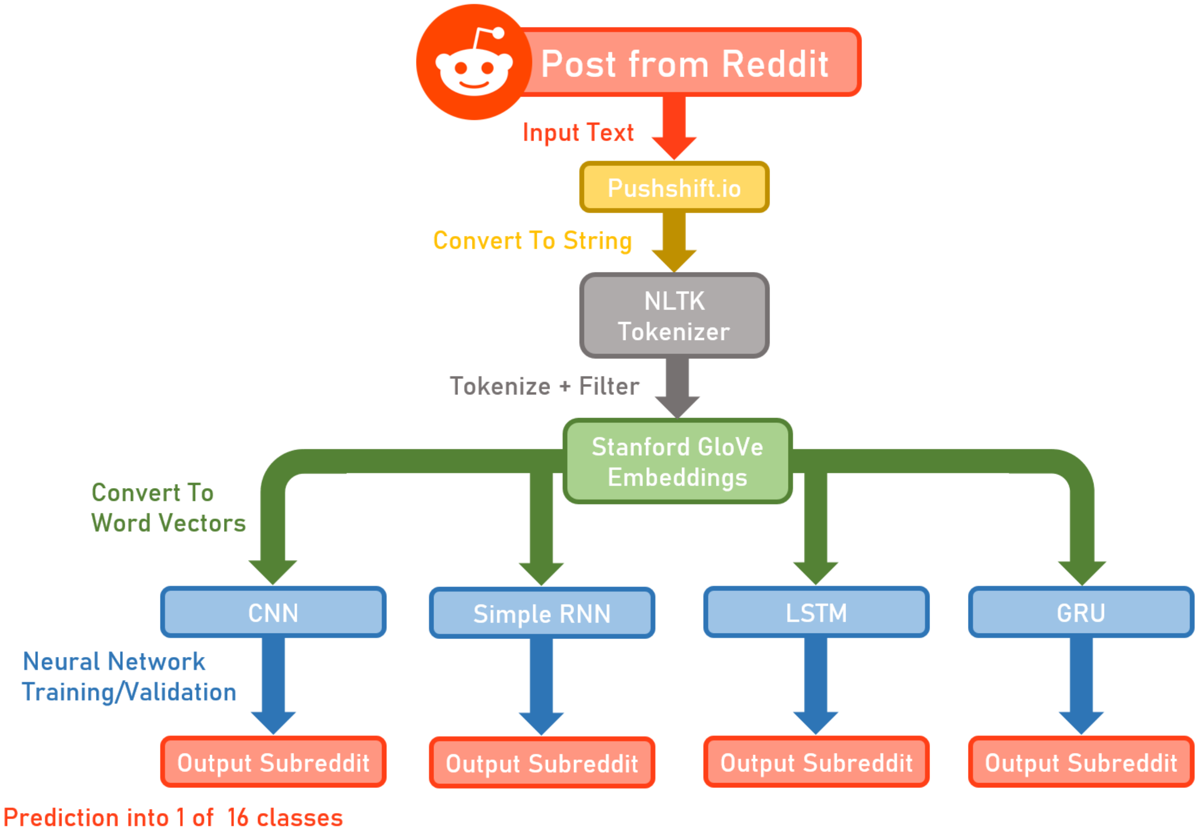


Figure 1: Overall structure of the project.

3 Background and Related Work

Due to the recency of popular work done in NLP techniques such as the repopularization of LSTM networks or creation of GRU networks, most related projects seek to solve similar problems on platforms using older techniques. An example of this is a project, 'Supervised Text Classification in Reddit Posts'[1], by Jacqueline Gutman. This project, from 2015, seeked to apply and evaluate a number of text classification techinques to categorize text-based reddit posts into one of five chosen categories. This project explored a Naive Bayes method, Multinomial Logistic Regression, Support Vector Machines, and Ensemble Methods. The project concluded that these approaches didn't improve much on the Naive Bayes approach. Since 2015, other models such as the GRU in NLP have been popularized and shown promising results in related fields. We applied these techniques to improve on the results reported by in this project.

This problem is a specific case of a Text Classification problems. Accordingly, another related work is a book which describes techniques on mining and processing text data. Specifically, the chapter called 'A Survey of Text Classification Algorithms' [2] discusses a number of techniques

[1] More info on `http://jgutman.github.io/`

for analyzing and classifying text such as Topic Representation. This work and others discussed in class suggested and influenced the architecture used by our models.

4 Data Processing

The source of data for this project is the Reddit website, where a vast amount of posts are pre-categorized into subreddits. This data is available through the pushshift.io API, which stores all posts from reddit in an online database. We can access this data in Python using the `psaw` package, a wrapper for the pushshift.io API. Using the `search_submissions()` method from `psaw`, we fetched to the top 4000 posts (3000 training, 300 validation, 500 testing) from popular subreddits in the form of Python strings. The data consists of posts from the following 16 subreddits, which will serve as the classes for this project:

- `r/askreddit`
- `r/askscience`
- `r/history`
- `r/jokes`
- `r/legaladvice`
- `r/LifeProTips`
- `r/movies`
- `r/personalfinance`
- `r/relationship_advice`
- `r/science`
- `r/space`
- `r/sports`
- `r/technology`
- `r/tifu`
- `r/todayilearned`
- `r/worldnews`

Thus we have $4000 \times 16 = 64000$ data samples in total.

To train the model, we used the titles of the posts only. This allows us to train and classify data from subreddits that are not exclusively text based (as all Reddit posts have a text title), and it also ensures that no text input will be of outstanding length.

Each post (input) has been processed to make NLP easier. This process included the following[2]:

- ignoring case sensitivity,
- expanding contractions (e.g. convert “don’t” to “do not”),
- removing all punctuation except periods and question marks,
- removing prefixes associated with specific subreddits (e.g. LPT for `r/LifeProTips` or TIL for `r/TodayILearned`), and
- lemmatizing words to root using NLTK (e.g. convert “corpora” to “corpus”).

After the process outlined above, the text is first tokenized using the Natural Language Toolkit (NLTK) and then each word is converted to a 100-dimensional word vector using GloVe word embeddings.

[2] `https://github.com/ece324-2019/Where2Spreddit/blob/master/filter.py`

Overall, an input Reddit post of n words is processed into a $100 \times n$ size tensor.

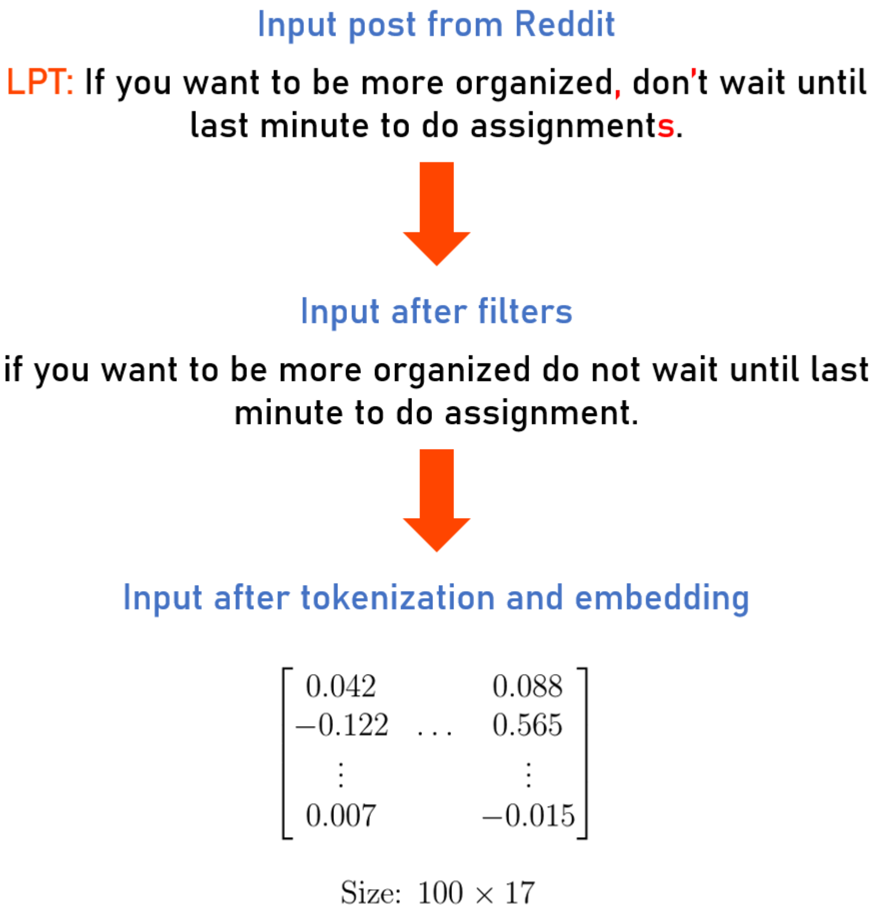

Input post from Reddit
LPT: If you want to be more organized, don't wait until last minute to do assignments.
Input after filters
if you want to be more organized do not wait until last minute to do assignment.
Input after tokenization and embedding
$\begin{bmatrix} 0.042 & & 0.088 \\ -0.122 & \dots & 0.565 \\ \vdots & & \vdots \\ 0.007 & & -0.015 \end{bmatrix}$
Size: 100 × 17

Figure 2: Input text is filtered to remove subreddit prefixes

5 Architecture

As discussed in the proposal, we decided to attempt this problem by cerating and training four ML models (CNN, RNN, GRU, LSTM). Each of our models was chosen to attempt a certain approach at text processing. The CNN was chosen to be able to detect patterns in syntax and word order. The RNN was chosen to use short-term memory to detect patterns of word emphasis in post structure. The GRU was chosen to use selective long-term memory to pick out otherwise hidden patterns throughout longer posts. The LSTM was added later to compare results with the GRU, as it also a popular model for NLP. Through training, the GRU model performed the best, and is thus described below.

The GRU network we trained has hidden dimension 100. The output of the GRU is then fed into a single linear output layer with 16 neurons and Softmax activation and Categorical Cross-Entropy Loss. The model does not utilize more FC layers nor further regularization methods (Batch-norm/dropout) since a negligible effect on accuracy was observed. Performance and results of this model will be discussed in **Section 7**.

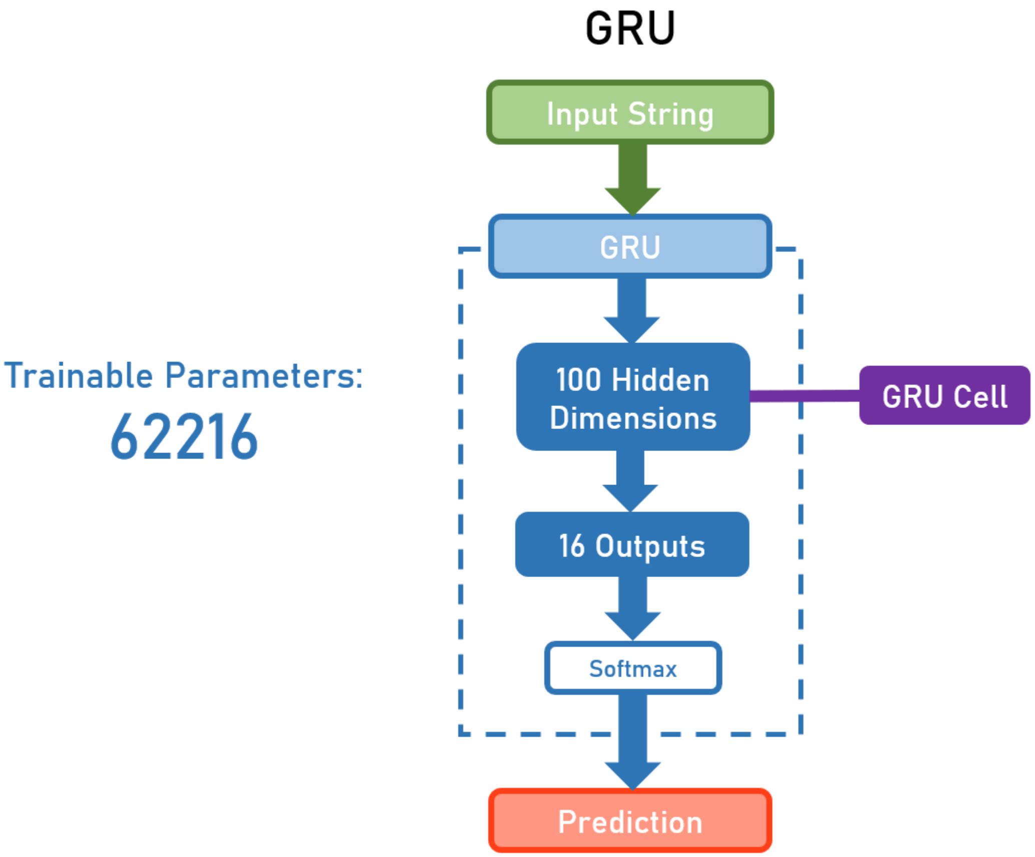


Figure 3: Structure of the GRU model

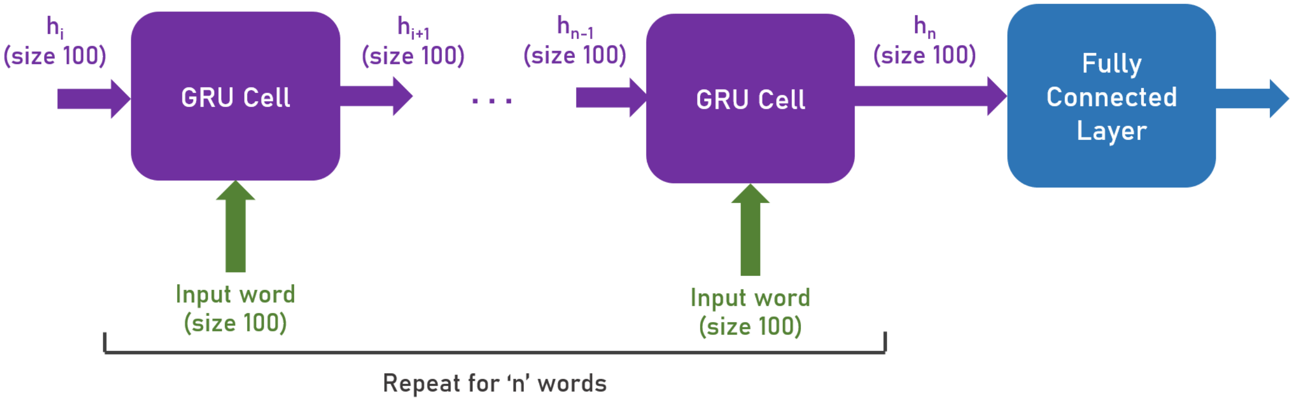


Figure 4: Detailed diagram for GRU cells

6 Baseline Model

We initially proposed a deep NN as a baseline model with many layers to see how well a normal NN could perform. From feedback received during the proposal and duration of training time, we decided our baseline model was too complex. The new baseline model (inspired from Assignments 4 and 5) takes an average of the tokenized word vectors from the input. This average vector is then fed into a Multilayer Perceptron (MLP)[3]. The MLP currently uses two hidden linear layers with 64 and 32 neurons with ReLU activation. This feeds into an output linear layer with 16 neurons (classes) with Softmax activation and Categorical Cross-Entropy Loss.

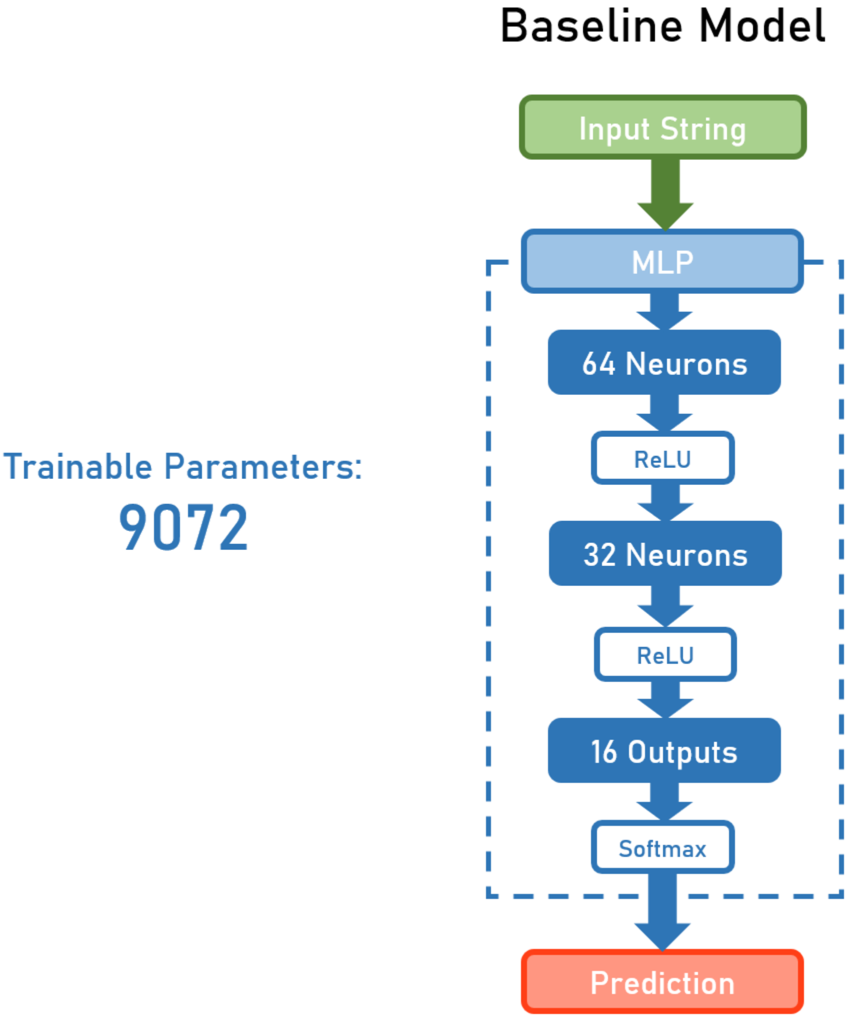


Figure 5: Structure of the baseline model (MLP)

[3] `https://github.com/ece324-2019/Where2Spreddit/blob/master/models.py (Line 6)`

7 Results

For a multi-classification problem such as this one, analyzing accuracy as a metric is a fast and convenient means of evaluating model performance. Other classification metrics such as Precision or Recall would require comparisons between each combination of classes. The data output of such a representation would present far too much data to be meaningful (${}_{16}C_2 = 120$ values to be specific, one for each distinct pair of subreddits). As such, accuracy was chosen as the metric for the models.

Below, we show Loss and Accuracy Plots for both the baseline and GRU[4] models. Both models trained successfully, achieving validation accuracies of **67.6%** and **84.6%** for the baseline and GRU respectively. Their test accuracies were similar at **67.4%** and **84.5%** , which is much better than the random chance probability of 1/16 = 6.25%.

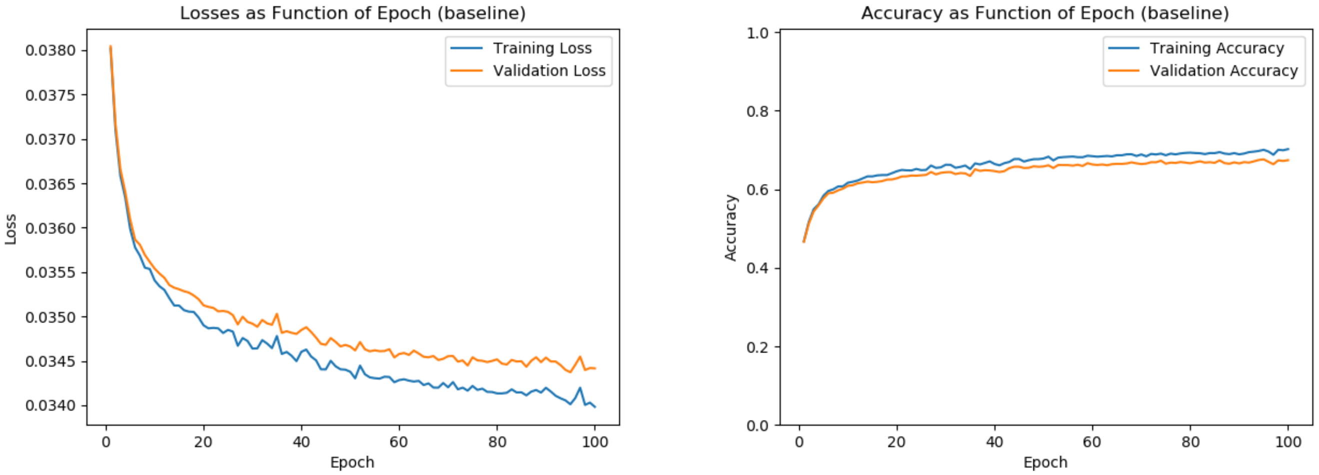


Figure 6: Loss and Accuracy Plots for Baseline Model
Optimizer: Adam
Loss Function: Cross-entropy Loss
Learning Rate: 0.001

[4] `https://github.com/ece324-2019/Where2Spreddit/blob/master/models.py (Line 30)`

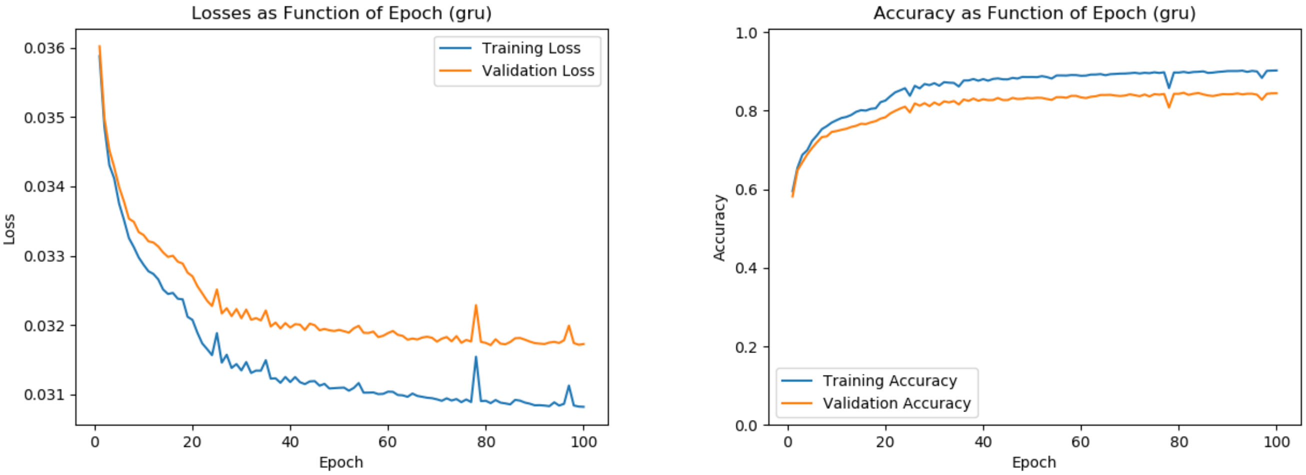
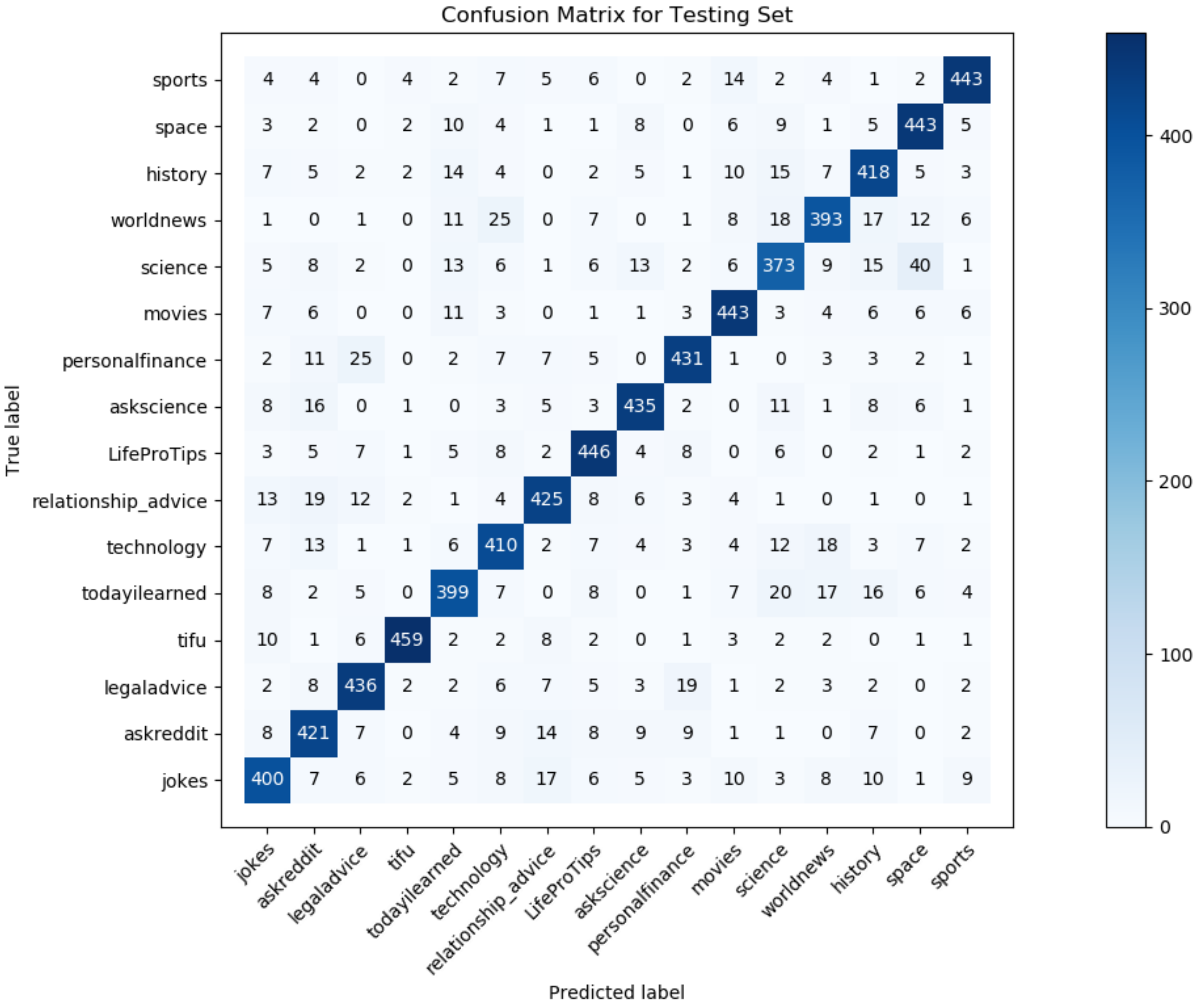


Figure 7: Loss and Accuracy Plots for GRU Model
Optimizer: Adam
Loss Function: Cross-entropy Loss
Learning Rate: 0.0001

As seen in **Figures 6** and **7**, both models converged and trained successfully, and our GRU model outperformed the Baseline significantly. Similarities in shape between training and validation can be explained by a large number of reposts, resulting in some data that is repeated in the training and validation set. However, we left reposts in the training process as we believe it is representative of the whole of Reddit.

Below, we also include the confusion matrix for the GRU which shows the subreddits that were most often confused by the model. We see that the confusion matrix is noticeably symmetric. This suggests that confusion between two classes is bidirectional, suggesting the classes themselves are inherently similar. An example ambiguous post title "*How much does it cost to hire a lawyer?*" can be fairly classified for both `r/personalfinance` and `r/LegalAdvice`. Other points of confusion such as similarities between `r/science` and `r/space` are also understandable. In such cases, Reddit users may post the same post to different subreddits (known as cross-posting), and thus a limitation of our model is that it does not support multiple labels for one input sample.

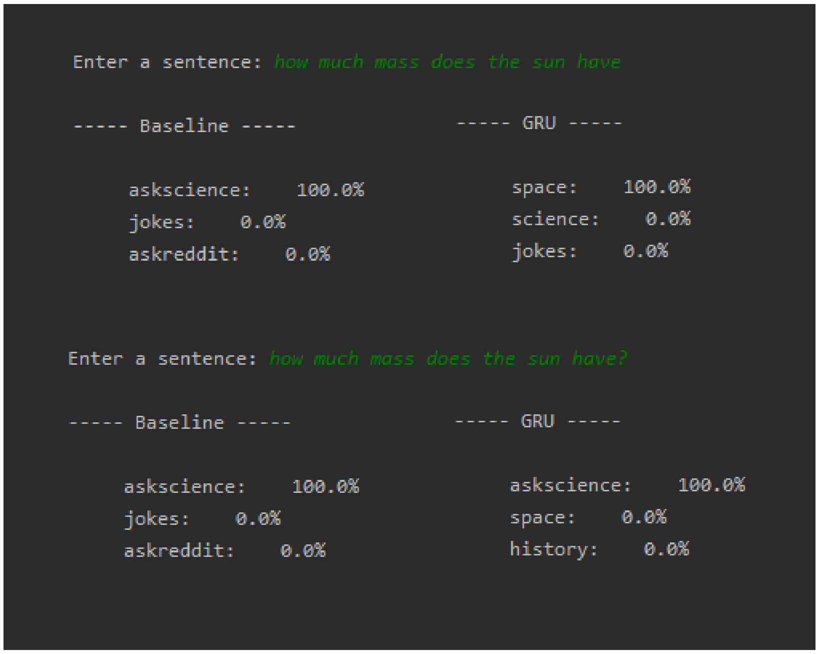


True label	jokes	askreddit	legaladvice	tifu	todayilearned	technology	relationship_advice	LifeProTips	askscience	personalfinance	movies	science	worldnews	history	space	sports
sports	4	4	0	4	2	7	5	6	0	2	14	2	4	1	2	443
space	3	2	0	2	10	4	1	1	8	0	6	9	1	5	443	5
history	7	5	2	2	14	4	0	2	5	1	10	15	7	418	5	3
worldnews	1	0	1	0	11	25	0	7	0	1	8	18	393	17	12	6
science	5	8	2	0	13	6	1	6	13	2	6	373	9	15	40	1
movies	7	6	0	0	11	3	0	1	1	3	443	3	4	6	6	6
personalfinance	2	11	25	0	2	7	7	5	0	431	1	0	3	3	2	1
askscience	8	16	0	1	0	3	5	3	435	2	0	11	1	8	6	1
LifeProTips	3	5	7	1	5	8	2	446	4	8	0	6	0	2	1	2
relationship_advice	13	19	12	2	1	4	425	8	6	3	4	1	0	1	0	1
technology	7	13	1	1	6	410	2	7	4	3	4	12	18	3	7	2
todayilearned	8	2	5	0	399	7	0	8	0	1	7	20	17	16	6	4
tifu	10	1	6	459	2	2	8	2	0	1	3	2	2	0	1	1
legaladvice	2	8	436	2	2	6	7	5	3	19	1	2	3	2	0	2
askreddit	8	421	7	0	4	9	14	8	9	9	1	1	0	7	0	2
jokes	400	7	6	2	5	8	17	6	5	3	10	3	8	10	1	9

Figure 8: Confusion matrix for the test set with 500 samples of each class

The biggest significant improvement of our model over the baseline is its ability to interpret and classify contextual clues, syntax, and meaning of symbols.

In the example below, we see that the Baseline outputs hardly change when a question mark is added to the input. The GRU, however understands that the question mark increases the probability of being in question/answer subreddits. Accordingly, the addition of just a question mark greatly changes the output of the GRU. This is coherent with the fact that words at the end of the sentence are more significant for the GRU model.



```
Enter a sentence: how much mass does the sun have

----- Baseline -----                ----- GRU -----


     askscience:    100.0%              space:    100.0%
     jokes:    0.0%                     science:    0.0%
     askreddit:    0.0%                 jokes:    0.0%


Enter a sentence: how much mass does the sun have?

----- Baseline -----                ----- GRU -----


     askscience:    100.0%              askscience:    100.0%
     jokes:    0.0%                     space:    0.0%
     askreddit:    0.0%                 history:    0.0%
```

Figure 9: Baseline vs GRU predictions differ for punctuation

8 Discussion and Learning

Overall, the model is performing quite well. The confusion matrix shows a very strong diagonal which is ideal. Most of the errors are also symmetric across the main diagonal, as a number of our chosen classes are similar by nature. Because of this, we have an accuracy above 80% despite the fact that accuracy ignores secondary and tertiary predictions in cases which are nearly tied (as is the case with ambiguous posts). As such, the model is performing quite well. While RNN advantages were discussed briefly in class, we saw in this project just how powerful they are when dealing with language. As predicted, the GRU is able to interpret and remember language patterns, contexts, and important clues which help make a prediction. However, the extent to which the GRU was able to decide on inputs was far better than we had ever imagined. For many ambiguous posts, the GRU is actually able to outperform humans in deciding the source subreddit of a post.

When starting this project, our team was very new to the field of machine learning in practice. From our experience, training and optimizing a chosen model can be done reliably with enough time a thorough hyperparameter search. As such, choosing the model becomes a crucial step to success. In our case, the lack of inexperience led us to choose three different main models to optimize. Through the hyperparameter search, we learned that, for this task, the GRU outperformed the CNN which outperformed the RNN. In the future, when starting similar tasks, we would make sure to use anecdotal evidence, reference materials, and expert opinions to make sure the proper model was selected for the task. This would make the process much more efficient when compared with experimenting with multiple models as more time could be spent on data processing, conditioning, and optimization.

Lastly, we would like to emphasize the importance of pre-processing the data **before** commencing training. We orginally used a simplified pre-processing that did not include lemmatization, and the after tokenization there were 40000 unique “words”, many of which had null word vectors as they were not recognized. Adding new input samples was sometimes detrimental to the results as more unqiue words were added to the vocab, further complicating the training. After lemmatization and contraction expansion, the number of unique words dropped to 27000, and the accuracies increased by at least 5% for all models. If given opportunity for another project, we will definitely start with optimizing pre-processing first so in the later stages we can focus on the model itself, rather than other potential improvements elsewhere.

9 Ethical Framework

Beneficence	• As mentioned in the introduction, the text classification capabilities of Where2Spreddit can benefit users who post online by identifying others with similar interests. This could result in expanded social circles and new friend groups for users, uniting those with similar interests (beyond just one common subreddit) • If this tool is implemented, moderators of subreddits and users who frequently check the newest posted content will not have to see nearly as many uninteresting or unrelated posts when browsing subreddits. • This tool also helps users who wish to post content if they unsure about the proper subreddit. This tool would help them as they can avoid their post being removed by moderators of incorrectly chosen subreddits. They can also avoid negative comments from the users of the incorrect subreddits they may post in.
Autonomy	• Inappropriate use for the network can also limit autonomy. Specifically, consider the earlier example where the model is used to connect people with similar interests. A social media platform implementing this might content suggestions are based on classification of posts. This can drastically limit the online social network of individuals by only letting them interact with content that users are already engaged with. This could result in a more closed-minded public. • Since this model would perform some duties currently handled by subreddit moderators (removing unrelated posts), it moves some of the subreddit governance to our automated model. This can potentially reduce autonomy for the communities as the model may have been mistrained or performing poorly as the owners/creators of the model would be the ones handling governance.

Justice	• Because our model is consistent, using it to evalaute a piece of text may be more 'fair' than the current (potentially biased) moderators. Outside of Reddit, application of such a model can be useful for employers screening applicants for a specific skillset based on a résumé. • However, one must be wary of potential biases associated with such a model. While true that it is impartial, it's possible that the model be trained improperly. In the hiring example, the model may neglect certain traits of applicants, depending on the learned parameters. This could unintentionally discriminate or lead to unjust hiring practices.
Non-Maleficence	• In any context in consideration, organizations who are considering implementing such a system always have the choice of not implementing it. As such, from an implementing organization's standpoint, this model scores highly on non-maleficence. • Consider the implementation instead from a user standpoint. In the Reddit example, users might feel as if such a model might remove some of their freedom of speech if the model rejects one of their posts in a specific subreddit. However, the model can be implemented instead to only suggest alternative subreddits where users might post content instead. In this way, the user isn't stopped from posting to their desired subreddit, but the model would only recommend more applicable subreddits (where their post is less likely to be banned or receive negative feedback). This implementation scores highly on non-maleficence since the user's actions are not affected negatively in any way.

References

[1] D. W. Otter, J. R. Medina, and J. K. Kalita, “A Survey of the Usages of Deep Learning for Natural Language Processing,” *IEEE Transactions on Neural Networks and Learning Systems*, Sep. 2019.

[2] Aggarwal, CC., Zhai, C. 2012. A Survey of Text Classification Algorithms. *Mining Text Data*, pp. 163-222.