Group Permissions

We grant permission to post our group video, the final report, and our source code

Pranav Agnihotri | Daniel Mogilny

Reddit Classifier

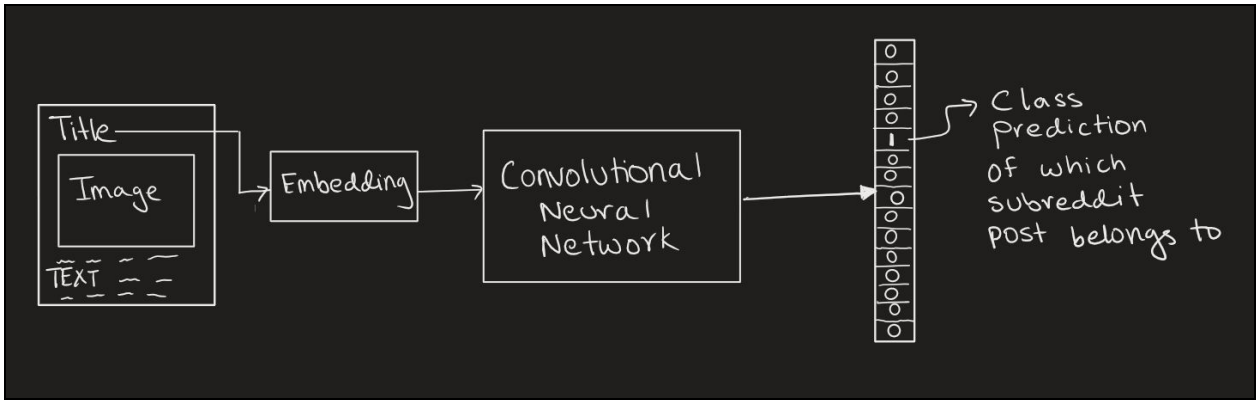
Word Count: 1996 Penalty: 0%

Introduction

Reddit is an online space that's divided into smaller communities called *subreddits*. Each subreddit has a moderator to remove off-topic posts and notify the poster where the post belongs. The process is manual, with potential to miss posts, and misidentify where they belong. The goal is to build a model to classify subreddit posts. This is well-suited to machine-learning since *understanding* a post and its topic with high accuracy isn't possible with conditional programming. Previous research like the medical research classifier[1], have shown that neural networks are well suited for context-based text classification over classical counterparts.

Illustration

A bird's eye view of the vision of the project is shown below.



Background and Related Work

The below sources demonstrate cutting-edge approaches to context-based classification, and motivate our approach with regards to architecture, and data processing.

- **Distributed Representations of Words and Phrases and their Compositionality.**[2] This paper turns words into vectors using neural networks, that capture semantic meaning. As an example, the encodings understood that "Montreal" : "Montreal Canadians" :: "Toronto" : "Toronto Maple Leafs". This allows better extraction of meaning from text in NLP tasks, and *gives us a groundwork to convert text into machine-understandable representations that preserve semantic relationships*.
- **Medical Text Classification Using Convolutional Neural Networks**[3]**.** This study applied CNNs to classify sentences from patient descriptions into health-related categories using word embeddings trained on a medical corpus. This method proved 15% more accurate than previous methods. *And as our goal is to also extract meaning*

[1] https://arxiv.org/ftp/arxiv/papers/1704/1704.06841.pdf
[2] https://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf
[3] https://arxiv.org/ftp/arxiv/papers/1704/1704.06841.pdf

from sentences within a post, it provides architectural inspiration (CNNs) for sentence processing tasks.

- **Deep Convolutional Neural Networks for Twitter Sentiment Analysis[4].** This study shows the success of deep learning in understanding *sentiment* behind text. Using a vector embedding system that focuses on characters and applying the embeddings to a CNN architecture, this study was able to quite accurately judge tweet sentiment. *This motivates a different form of english representation on a character-by-character basis, not just a word-embedding approach.*

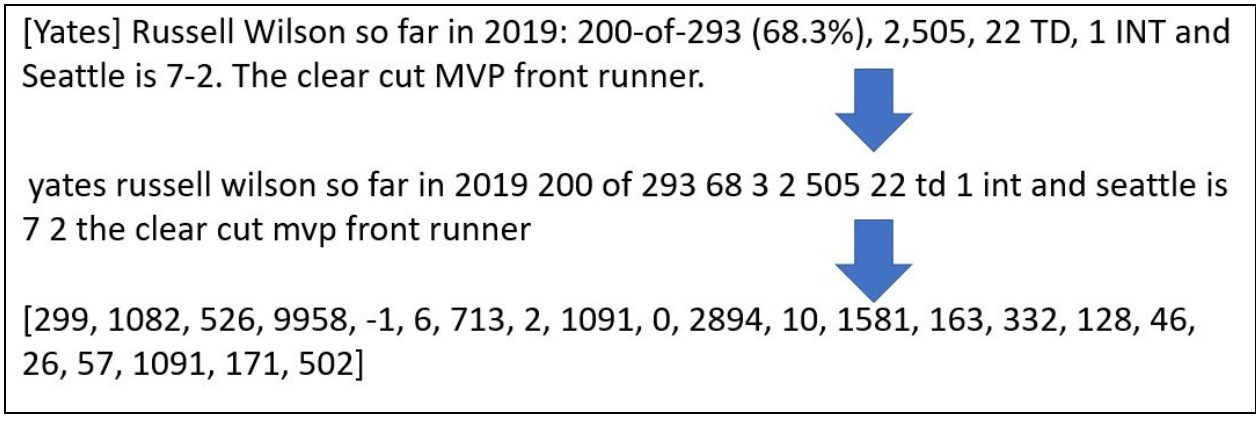
Data and Data Processing

We scraped 50,000 posts across 20 subreddits, ensuring at least 50 net "upvotes" on each post. This confirmed the post belonging to the subreddit.

We only use the post's title as we propose it provides sufficient information, as well as reduces memory and computational resources required to build a large dataset. Furthermore, we decided to not use pre-trained word embeddings due to the possibly reddit-specific vocabulary that would be overlooked if not present in the pretrained-embedding vocabulary. Hence, we formed our own word embeddings and character embeddings (inspired by background research).

Word-embedding pipeline:

1. Strip all punctuation and non alphanumeric characters
2. Turn everything lowercase
3. Assign each unique word that occurs more than 100 times a unique integer, while assigning the rest a -1 value

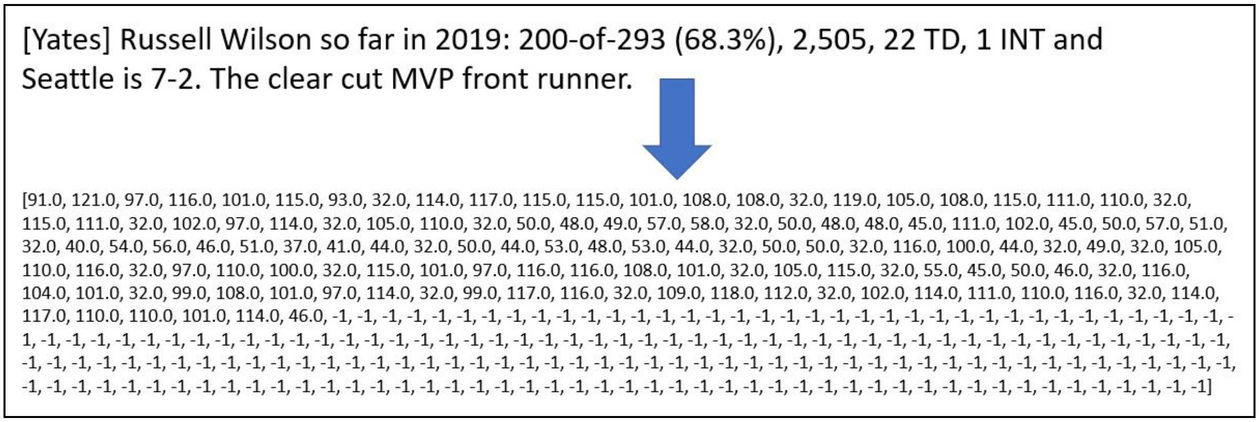


Character embedding pipeline:

1. Replace all non-ascii characters with -1
2. Turn everything lowercase
3. Assign each unique character a unique integer

[4] https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8244338

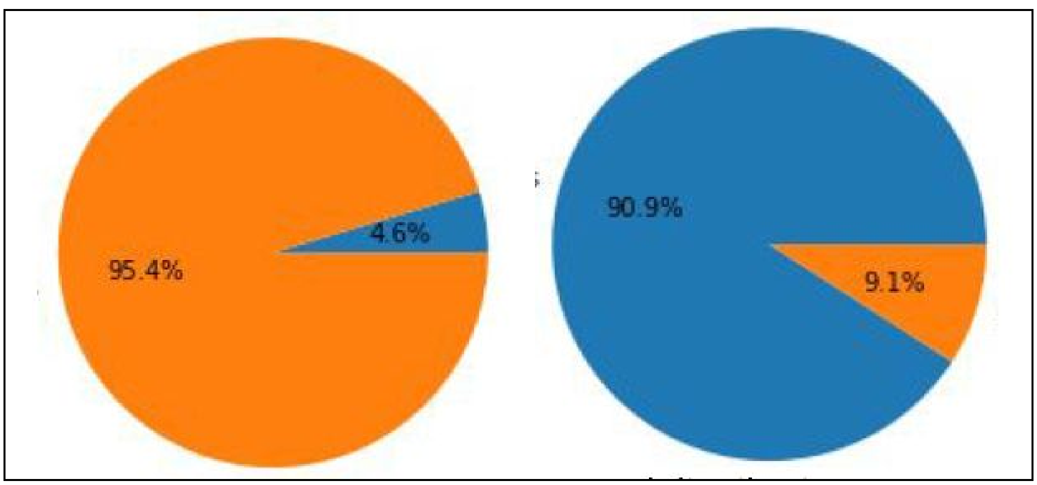
4. Pad all titles to the longest title-length in the dataset with -1


[Yates] Russell Wilson so far in 2019: 200-of-293 (68.3%), 2,505, 22 TD, 1 INT and Seattle is 7-2. The clear cut MVP front runner.

[91.0, 121.0, 97.0, 116.0, 101.0, 115.0, 93.0, 32.0, 114.0, 117.0, 115.0, 115.0, 101.0, 108.0, 108.0, 32.0, 119.0, 105.0, 108.0, 115.0, 111.0, 110.0, 32.0, 115.0, 111.0, 32.0, 102.0, 97.0, 114.0, 32.0, 105.0, 110.0, 32.0, 50.0, 48.0, 49.0, 57.0, 58.0, 32.0, 50.0, 48.0, 48.0, 45.0, 111.0, 102.0, 45.0, 50.0, 57.0, 51.0, 32.0, 40.0, 54.0, 56.0, 46.0, 51.0, 37.0, 41.0, 44.0, 32.0, 50.0, 44.0, 53.0, 48.0, 53.0, 44.0, 32.0, 50.0, 50.0, 32.0, 116.0, 100.0, 44.0, 32.0, 49.0, 32.0, 105.0, 110.0, 116.0, 32.0, 97.0, 110.0, 100.0, 32.0, 115.0, 101.0, 97.0, 116.0, 116.0, 108.0, 101.0, 32.0, 105.0, 115.0, 32.0, 55.0, 45.0, 50.0, 46.0, 32.0, 116.0, 104.0, 101.0, 32.0, 99.0, 108.0, 101.0, 97.0, 114.0, 32.0, 99.0, 117.0, 116.0, 32.0, 109.0, 118.0, 112.0, 32.0, 102.0, 114.0, 111.0, 110.0, 116.0, 32.0, 114.0, 117.0, 110.0, 110.0, 101.0, 114.0, 46.0, -1]

We justify ignoring all words that occur under a 100 times within the corpus with the following data-statistics. Below, orange represents words that occur less than 100 times in the corpus, and blue represents words that occur more than 100 times in the corpus.

# words	# unique words	# words occuring more than 100 times (common words)
14,855,063	244,734	11249



Left: Word Distribution Over Vocabulary *Right: Word Distribution Over Corpus*

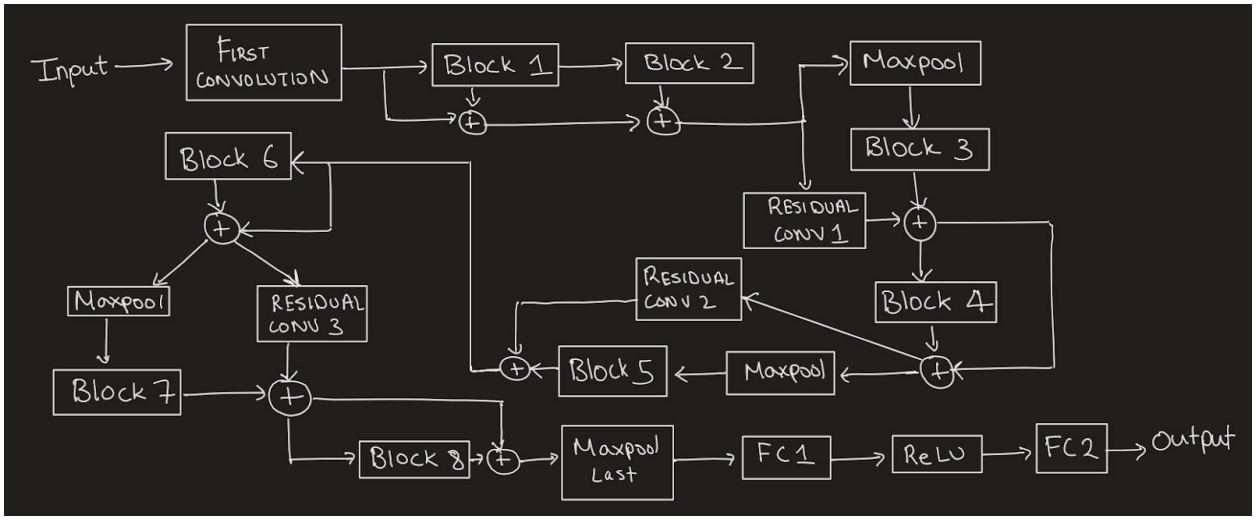
While words that occur a 100 times or less make up the majority of the vocabulary, they are the minority in the actual corpus. Their lack of prevalence allows for the safe assumption that they don't play a significant role as a deciding factor in the classification, and it allows us to narrow the search for key-words in our titles.

Moreover, within each subreddit, the top 100 most frequently occurring words share the majority with our overall top 100 most common words, with just enough of a difference to allow for subreddit nuances.

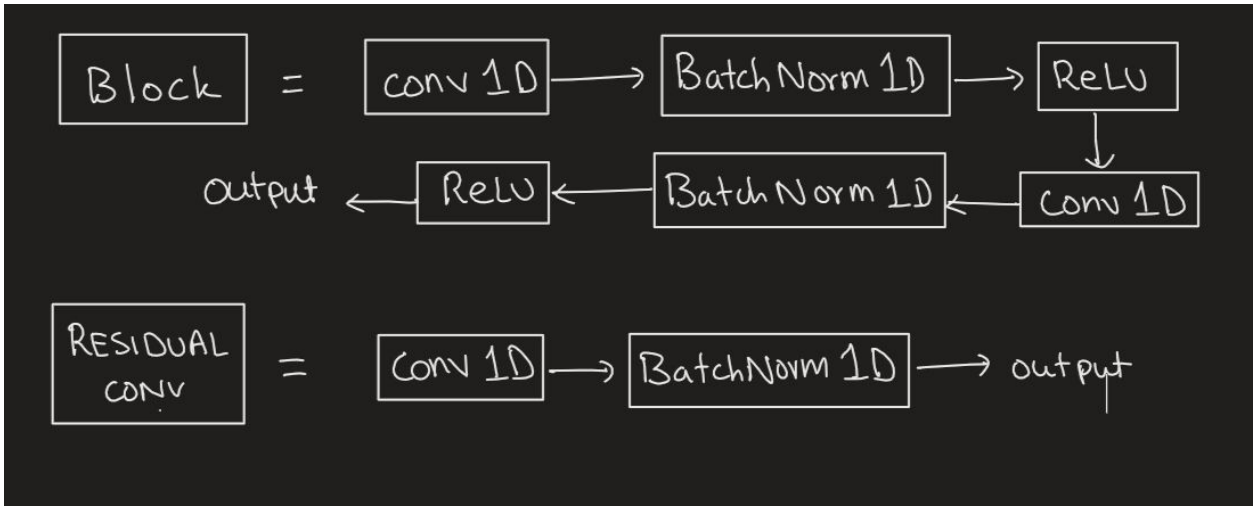
Subreddit	# of top 100 words in common with corpus top 100 words
0	69
1	67
2	59
3	58
4	66
5	61
6	65
7	68
8	72
9	72
10	48
11	71
12	74
13	64
14	66
15	59
16	30
17	68
18	67
19	61

Architecture

After testing several versions of deep CNN and LSTM networks, we arrived upon our final model which is a ResNet Inspired Convolutional Neural Network. It uses character-embeddings to obtain the best results, with a cross entropy loss function, adam optimizer, and learning rate of 0.001. The network's architecture is depicted below. Note that all convolutions and pooling are 1-Dimensional. As well, a benefit of having an extremely large number of words in our dataset is that we do not have to explicitly train our embeddings. The model is able to learn the appropriate representations over the course of its training cycle.



Each “Block” and “Residual Conv” layer has the below structure:



Note how at several instances the output from previous blocks is brought forward and added to the output of the current block. However, because of pooling, the dimension of a past output won’t match with the current output. The residual convolutional layers stride across the previous output to convert it into a compatible size for the current output. This system decreases the amount of information lost to pooling, and allows earlier layers to learn. A common problem with very deep networks is that the gradient is very small by the time it reaches the early layers, and these residual connections help alleviate that issue by carrying forward early values.

The hyper-parameters for each building block is listed in the tables below:

Block #	In Channels	Out Channels	Kernel Size	Stride	Padding
Block 1	64	64	3x3	1	0
Block 2	64	64	3x3	1	0
Block 3	64	128	3x3	1	0
Block 4	128	128	3x3	1	0
Block 5	128	256	3x3	1	0
Block 6	256	256	3x3	1	0
Block 7	256	512	3x3	1	0
Block 8	512	512	3x3	1	0

	In Channels	**Out Channels**	**Kernel Size**	**Stride**	**Padding**
Residual Conv 1	64	128	1	2	0
Residual Conv 2	128	256	1	2	0
Residual Conv 3	256	512	1	2	0
First Conv	100	64	3	1	1

	Inputs	**Outputs**
Fully Connected Layer 1	512	1024
Fully Connected Layer 2	1024	20

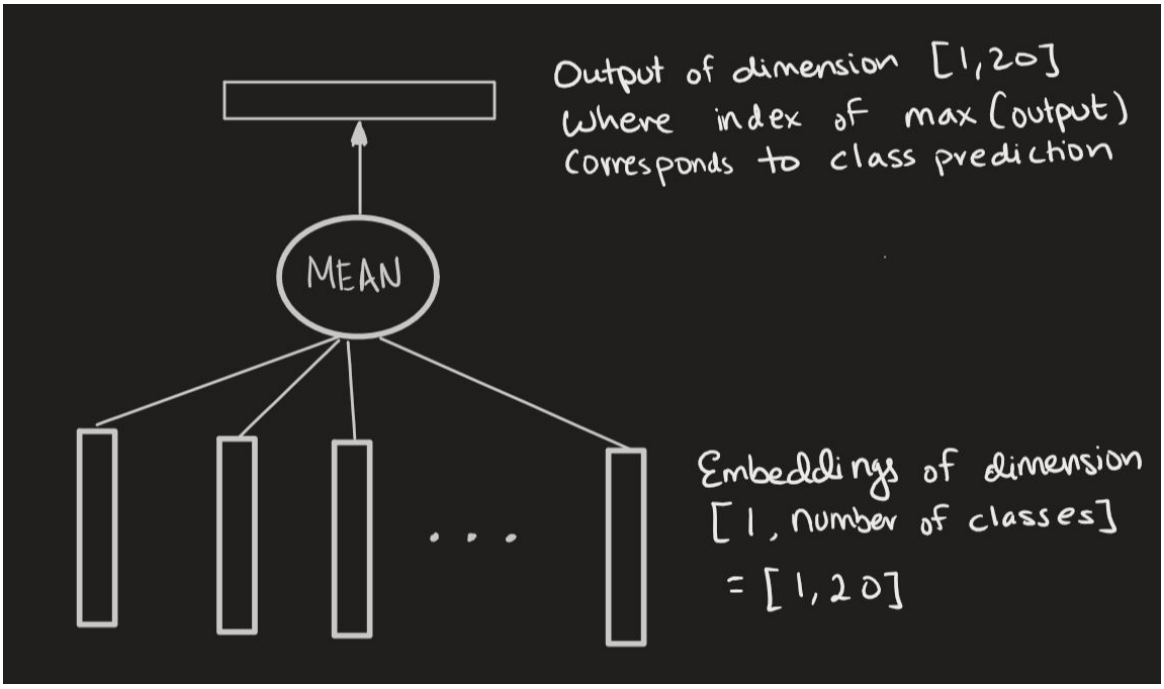
	Size	**Stride**	**Padding**
Maxpool	3	2	1
Maxpool Last	36	-	-

Baseline Model

We use two baselines that are appropriate as they encapsulate a simplistic method of trying to *understand* a sentence.

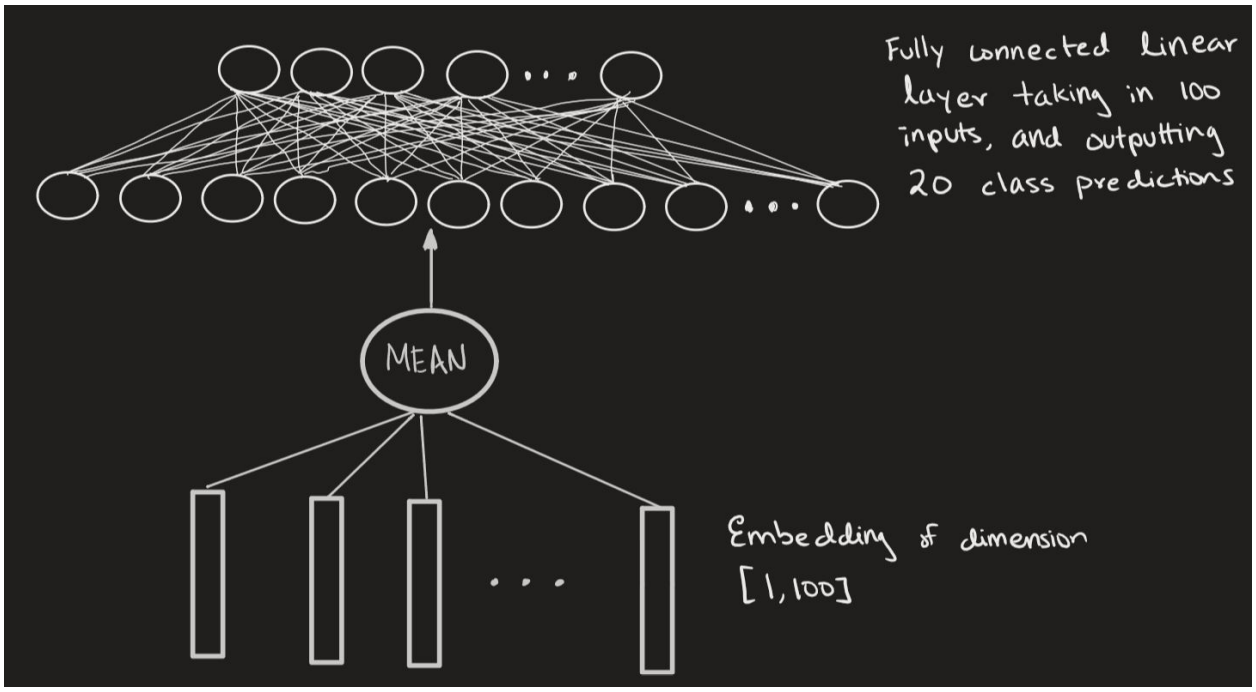
Bag-of-Words (BOW):

- Encode a document with untrained embedding, where the dimension of the embedding is the number of classes (20)
- Calculate the average of the embedding-vectors
- The average is the output, where the highest-valued index corresponds to prediction of that class



Assignment 5 “Method of Means”

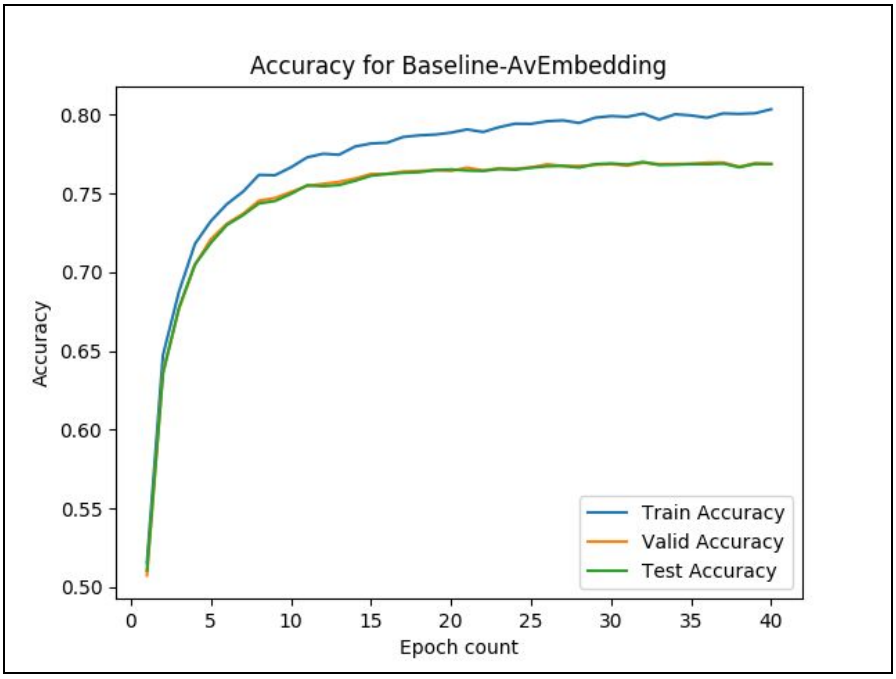
- Encode a document with untrained embedding, where the dimension of the embedding is 100
- Calculate the average of the embedding-vectors
- Feed the average through a fully connected layer that outputs a multi-class probability



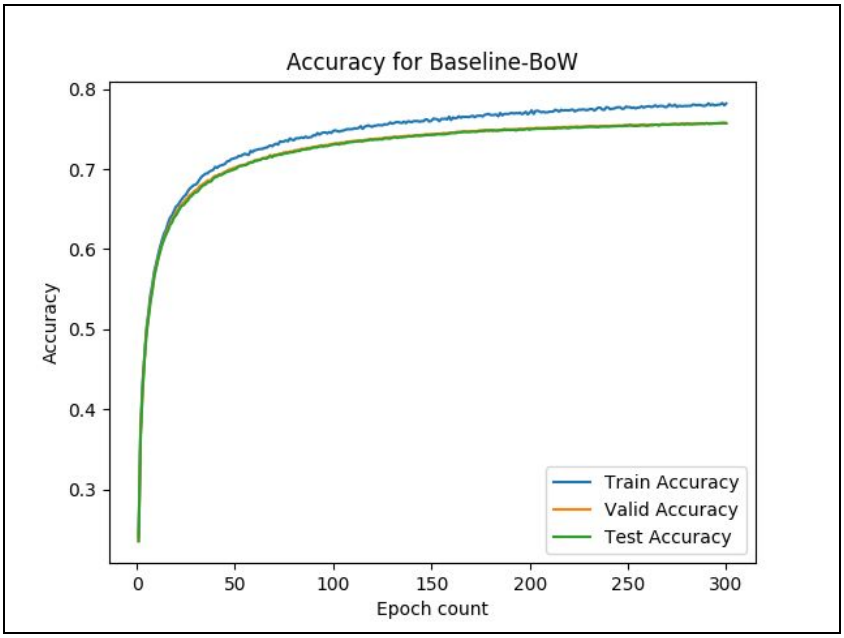
Quantitative Results

	Train Loss	Test Loss	Valid Loss	Train Accuracy	Test Accuracy	Validation Accuracy
Assignment 5 Baseline	0.56	0.66	0.67	0.8	0.76	0.77
Bag of Words	0.69	0.74	0.73	0.78	0.76	0.75
ResNet CNN	0.41	0.71	0.7	0.82	0.79	0.79

Assignment-5 Baseline:

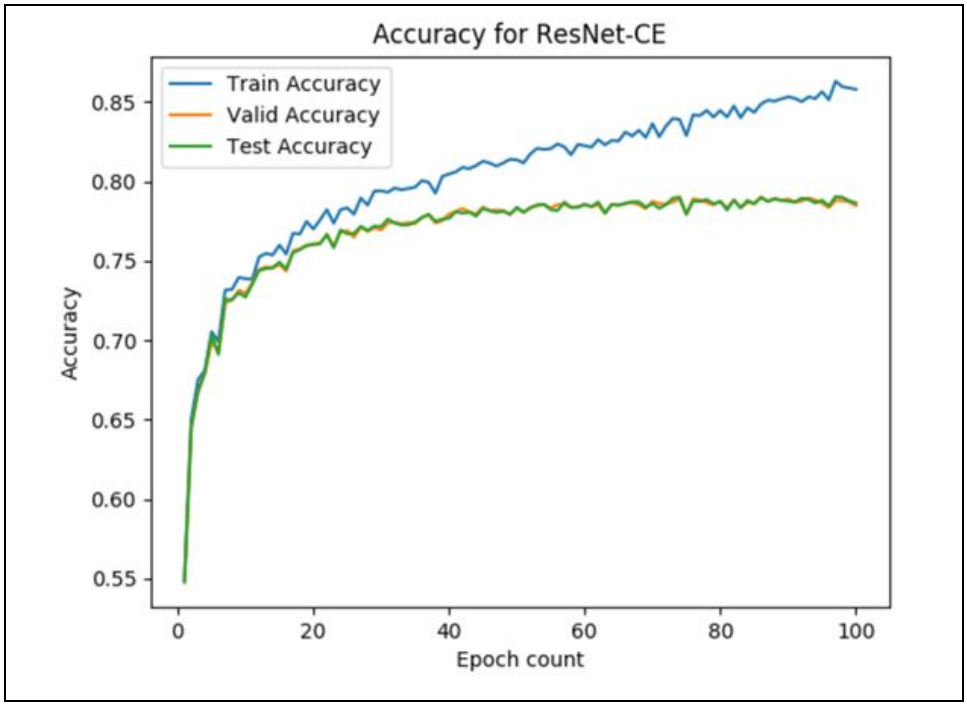


Bag-Of-Word Baseline:



Pranav Agnihotri | Daniel Mogilny

Character-Embedding ResNet:



We also present a confusion matrix for our ResNet, as well as the precision for each subreddit:

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
0	2319	9	3	5	9	3	10	8	5	0	0	77	11	4	27	10	0	5	17	14
1	2	1999	0	1	2	1	0	1	5	0	6	2	4	9	1	31	0	374	60	1
2	8	0	1927	253	7	134	52	15	1	0	6	14	48	0	44	17	0	0	21	4
3	15	0	199	2046	1	27	14	13	2	0	0	11	173	6	9	13	0	0	12	4
4	2	1	2	2	1471	9	171	43	140	3	29	16	75	374	8	107	1	2	40	12
5	5	2	115	25	17	2036	127	83	6	3	3	4	35	11	9	29	0	2	24	8
6	10	2	36	5	99	84	1822	80	58	1	13	6	50	94	21	98	9	3	54	15
7	3	0	6	3	35	42	40	2181	16	0	6	10	15	38	4	47	1	2	36	3
8	7	3	1	0	103	9	61	23	1775	2	14	6	88	231	3	60	1	7	38	10
9	0	0	1	0	0	0	0	0	0	2499	1	0	1	0	0	0	0	0	0	1
10	6	10	4	1	70	8	30	5	22	3	2235	7	8	32	6	14	2	3	56	5
11	66	3	4	4	17	3	13	9	7	0	3	2245	21	9	14	16	0	6	10	13
12	10	8	73	257	62	18	64	34	68	1	3	20	1574	156	14	70	0	8	42	5
13	4	2	2	2	180	8	73	70	140	1	8	8	93	1723	3	172	0	4	16	8
14	18	1	23	2	9	2	23	3	7	0	1	27	7	2	2237	6	1	0	12	16
15	4	24	2	14	31	9	40	55	38	6	4	7	30	117	2	2033	0	14	19	4
16	0	0	0	0	1	0	2	0	1	0	1	0	2	0	0	0	2504	0	0	0
17	5	1935	0	3	12	3	0	0	7	0	7	5	6	12	0	40	0	367	54	2
18	14	16	12	9	54	14	52	33	31	0	31	23	13	20	15	24	2	23	2066	14
19	20	3	3	1	22	4	25	2	6	0	16	13	18	17	33	9	1	1	18	2381

Worst Performing subreddit marked in Green, Best marked in Blue, Runner-ups marked in Orange

Subreddit	Precision
"todayilearned"	0.998401918
"EarthPorn"	0.997212266
"soccer"	0.933249896
"mma"	0.918241419
"nfl"	0.914432177
"nba"	0.911490053
"anime"	0.884447962
"showerthoughts"	0.876607717
"movies"	0.837793998
"jokes"	0.828781084
"politics"	0.803929273
"science"	0.800314465
"AskReddit"	0.799919968
"worldnews"	0.755390043
"freefolk"	0.726863227
"interestingasfuck"	0.71171875
"teenagers"	0.684545093
"The_Donald"	0.632891033
"gaming"	0.586523126
"askmen"	0.149308381

Precision for each subreddit ranked best-to-worst

Subreddit Name	Subreddit Number
"nfl"	0
"AskReddit"	1
"worldnews"	2
"politics"	3
"gaming"	4
"science"	5
"interestingasfuck"	6
"showerthoughts"	7
"freefolk"	8
"todayilearned"	9
"anime"	10
"nba"	11
"The_Donald"	12
"teenagers"	13
"soccer"	14
"jokes"	15
"EarthPorn"	16
"askMen"	17
"movies"	18
"mma"	19

Subreddit corresponding to numbering in confusion matrix

The worst performing subreddit is the AskMen subreddit (focused on asking questions to men) that confuses a lot of its posts with the AskReddit subreddit (focused on asking questions to people in general). This may be because the title itself is not a sufficient indicator to distinguish between questions geared towards men versus the general populous, as the post's text-body would likely contain such details.

The best performing subreddits are TodayILearnt and EarthPorn. Every post in TodayILearnt begins with "TIL" and almost every post in EarthPorn ends with "OC" (Original Capture) followed by the image dimensions in the form [1234x1234]. Since character-embeddings don't drop words (like in our word-embedding pipeline) or ignore words not in the vocabulary (as in pre-trained embeddings like GloVe) these important tags were taken advantage of and posts were identified with high precision.

The sports subreddits for NFL, Soccer, MMA, and NBA perform well also. This result is unsurprising since sports have very consistent terminology across posts.

The more general the subreddit, the worse it performs, as when topics overlap across subreddits so do the model predictions. For example, 20% of posts belonging in the Donald Trump subreddit were found in "Politics", "Teenagers", and "World News".

Qualitative Results

From Subreddit:	Classified As:	Post
TodayILearned	**TodayILearned**	TIL that the entire surface of Saturn's moon, Titan, can move several kilometers per month, suggesting that there is an ocean underneath the surface.
teenagers	**gaming**	Uh oh guys don't do it

Notice that "TodayILearned" classifies very accurately, which is in-line with our analysis of that subreddit above. "Teenagers" classifies poorly due to the core of the post being in the image meme rather than the title; the title is low on information. The main conclusion derived from qualitative analysis is that perhaps for poor performing subreddits image data may be necessary and high performing subreddits typically have clear identifiers within the title. Overall model seems to produce expected results.

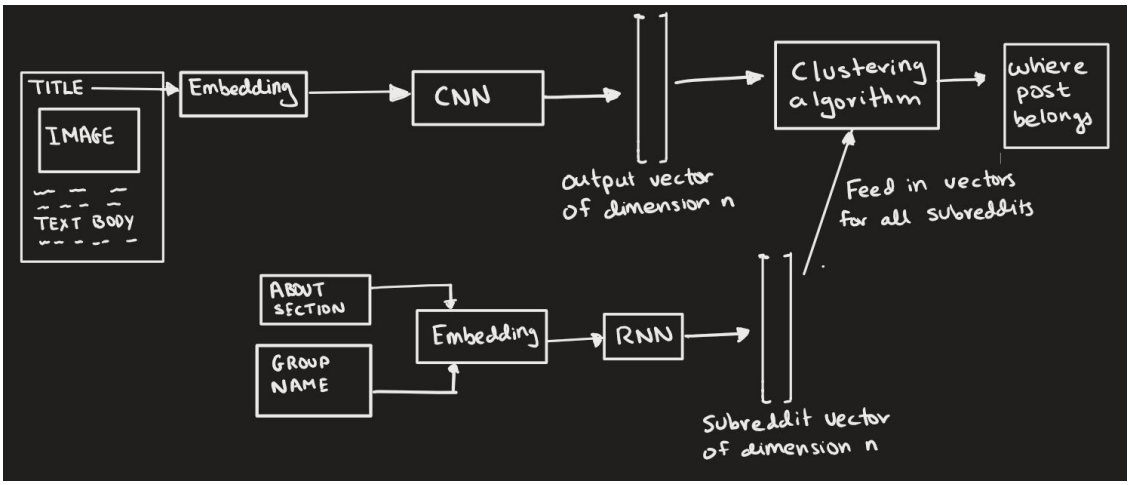
Discussion and Learnings

There are several key issues to address.

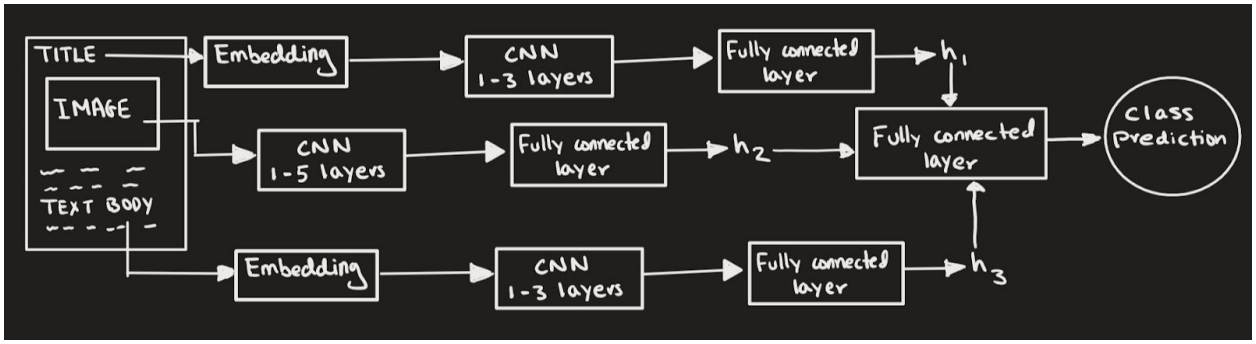
1. Why did character-embeddings outperform word-embeddings?
    a. We believe a major reason is the difficulty in accounting for all possible forms of a word with word-embeddings. Different tenses, it being used as an adjective, a verb, or a noun, or it having a small spelling mistake can mean a less accurate view of that word's presence within text. However, a character embedding approach is better at detecting the root presence of words.
    b. Pre-trained word embeddings such as GloVe simply ignore words that aren't in its vocabulary, and our word-embedding pipeline chose to ignore words that occur less than 100 times in the corpus (to have a reasonably sized vocab with the more common words). This is inherently a loss of information, but character embeddings do not suffer from these drawbacks. Instead, they piece together words as a sum of characters, and so they don't miss out on any possible key-words. Moreover, they also have the benefit of being able to recognize special characters.
2. Why did CNNs outperform RNNs in experimenting?
    a. In our experimentation RNNs failed to significantly outperform our baselines. Since our baselines only focus on what words are present, we believe this implies that the presence of key-words is more important than their sequential relationship. Hence, CNNs' kernel mechanism was better at detecting patterns and extracting key information across the title. This was helped by the titles all

being under 300 characters. A large varying in length would require large padding which introduces a lot of disbalance that would make the model perform poorly.

3. Is supervised classification inherently limiting in applicability?
    a. Our model is inapplicable to classification of posts from outside the subreddits we chose. Incorporating unsupervised methods might increase its applicability. As an example, instead of a network outputting a class prediction, it could output a vector. The labels for subreddits, instead of being numbers, are also vectors generated by an RNN that takes in the name of the subreddit and the “about” section. The output of the model and the label vectors can be grouped via a clustering algorithm. Thus, even if the model is trained on a few subreddits, it might learn to output vectors that generalize better beyond the subreddits trained on.



4. Did our data collection decisions limit performance?
    a. Even though our final model outperformed our baseline models, the margin wasn’t very high. As was seen in the case where AskMen posts were confused with AskReddit posts, the titles only go so far in providing the context of a post. They still play a major role, given that a 20-class classifier achieved a 79% accuracy from purely titles, while a random guesser does no better than 5%. However, a lot of information about the post is missed out on by not including the post’s text-body and image if present. A possible approach would be to include the first 300-500 characters of the text-body, and the image as well, and train three parallel networks to get a more nuanced understanding of the post. A possible architecture is visualized below.



Ethical Framework

The key stakeholders affected by this project are moderators, site users, and Reddit.

Maleficence, Autonomy, and Justice:

- Users may begin to conform their post format and language to not be flagged by spam if the algorithm is adopted but not very accurate, thus infringing on their autonomy to post with free expression. If the algorithm is in the hands of all, people looking to post unrelated spam can tweak how they write their posts, especially since the model just looks for the presence of key words. If the algorithm is reserved by the site, then Justice is increased in the sense that the penalty on the autonomy of “trolls” also affects the autonomy of regular users.
- New ideas or evolving subreddits might get flagged. An example is the r/Canada subreddit which turns political only during election season. This actively impedes on the ability for communities to organically grow and change, thus reducing non-maleficence.

Beneficence and Justice:

- Moderators are less burdened
- Users are directed to other subreddits that may better suit their needs
- Community members don’t end up interacting with off-topic posts
- Reddit is able to offer advertisers an increased ability to tailor their ads to feel like it is part of a subreddit. This benefits Reddit as ad engagement increases, and it benefits users as they receive advertisements related to their interests. Thus, justice is maximized for users and company.
- Reddit has a history of subreddits swarming other subreddits with their content (the r/politics subreddit posting anti-republican content in the r/DonaldTrump subreddit during election time). This *brigading* will be more easily dealt with by flagging such off-topic posts.