News Recommender Systems: Nearest Neighbor and NLP Approach

Brandi Hamming BHAMMIN1@JH.EDU

1. Abstract

Content based recommender systems often require a large amount of information about the items being recommended, but that data can be difficult to obtain. I hypothesized that we can still build a simple recommender system with the help from natural language processing (NLP) tasks. This research is focused on building news recommender systems that only use two or three features to build its recommendations off of. 5 different experiments were performed in this research. All methods used a nearest neighbor based approach to score similarity between historical articles and candidate articles for each user. Three of the experiments expanded the similarity score through NLP tasks like semantic textual similarity and natural language inference and two of the experiments built recommender systems based only on articles categorized as news articles. The results of the experiments demonstrate that incorporating NLP tasks into the similarity scoring resulted in more relevant recommendations. The NLP tasks had higher precision for recommending the top 3 and 5 articles to all users. Overall, the research demonstrates that a simple recommender system can be built even when there are limitations to your data thanks to the help of NLP tasks.

2. Introduction

The purpose of this paper is to explore ways to build a simple content based news recommender system that uses natural language processing to improve it's performance. For many recommender systems, the more data, the better the performance. Specifically for content based news recommender systems, numerous features about the candidate recommendations and user's reading history are often needed to provide accurate recommendations. These features are often text based such as the article's summary, named entities in the article, title of the article, category, sub-category, etc. Sometimes all of that information is not available due to time, access, costs, etc. Or the data could be available, but it would require time consuming pre-processing since transforming plain-text into numeric data is not a simple task. Additionally, users need to have trust when using a system. This means many users want to know why they are being recommended something. These problems are present for various recommender systems: news, movies, music, medical treatment, etc.

When data about the products to be recommended is unavailable, I hypothesize that natural language processing can help fill in gaps in the data. Natural language processing allows for computers to understand words. There are many different types of natural lan-

guage processing tasks like semantic textual similarity and natural language inference. For this research, I will be building nearest neighbor based news recommender systems. The algorithm will compare each user's historical articles to candidate articles using a similarity metric, then provide 5 article recommendations for each user. The system will be built using a data set that contains limited information about each article, which means the initial model is expected to perform poorly. To improve the performance of this model, I hypothesize that using semantic textual similarity and natural language inference scoring will improve the similarity score used in the nearest neighbor based recommender system. While the focus of this research is geared towards news recommendations, I believe this approach can be applied to other areas such as movies, music and book recommenders. This paper will contribute to extending the research on collecting and pre-processing a relevant news dataset, building a simple nearest neighbor based recommender systems, finding additional use cases for NLP tasks and overcoming the problem of having limited data.

2.1 Recommender Systems

Recommender systems can be built using content based filtering, collaborative based filtering or a hybrid approach. For content based filtering, recommendations are based on items that are similar to a user's history. Collaborative based filtering provides recommendations based on other users who are similar to the current user. A hybrid approach is a combination of the two methods. Collaborative and hybrid approach are outside of the scope of this research.

One way to build a content based news recommender system is to score similarity of content using a nearest neighbor (NN) approach. This means that for each feature, a score/distance will be taken between the candidate article and the historical article. Then the sum of all features scores is taken to achieve a final score. This approach has been used for various movie recommenders [Nik+21] and [ASN19]. NN approaches for recommender systems have key advantages such as simplicity, justifiability and efficiency [Nik+21]. NN models are simple because the are easy to implement and intuitive [Nik+21]. They are also have built in explainability because demonstrating to a user how articles are close in proximity to each other provide straightforward justification for a recommendation [Nik+21]. Finally, NN methods are model free, which means there is no resource consuming training phase [Nik+21].

2.2 Semantic Textual Similarity

Semantic textual similarity (STS) is used to compare the relatedness of two phrases. In one study, two different news recommender systems were built using STS between words extracted from the body of the articles [Cap+12]. Their study had relatively high accuracy scores: 78 and 80 percent [Cap+12]. This approach has been used in other disciplines such as a fashion item recommender [Jar+22]. For this research, a STS score will be taken between titles of articles. This approach could help eliminate ties between articles that might share the same score when using only a NN approach.

For example, let's say User A has read the following articles: "Best Recipes Using Sharp Cheddar", "Can Something Be Too Cheesey: A Study On Cheese Filled Recipes", and "The Best 5-Cheese Ziti Recipe". All of these articles would share the category of lifestyle and

the sub-cateogry of food. Now let's say we have two candidate articles both with the same category and sub-category: "Best Vegan Snack Recipes" and "Cheeses You Must Try When Traveling to France". Now the expected NN approach would score both candidate articles the same. But with STS, the article "Cheeses You Must Try When Traveling to France" should score higher because its title sounds the most similar to the historical articles. At least for this example user, the STS model would result in a better recommendation because the cheese related article would be ranked higher than the vegan article.

2.3 Natural Language Inference

Natural language inference (NLI) is used to predict whether two phrases infer each other, contradict each other or have no relationship to each other. The first phrase is usually called the premise and the second phrase is often called the hypothesis. When comparing phrases, the direction is important– specifically for entailment.

For example, let's say the premise is "Florida Republican Senator Joe Smith Always Votes Pro Life". If the the hypothesis is "Florida Senator Will Vote Pro Life In The Next Election", then this hypothesis would be labeled as entailment because this title is inferring the premise. But if we flip this order, then this pairing would actually be labeled as neutral. For the most part with entailment, the premise is often something very specific and the hypothesis will often be a summarization and/or generalization of the premise. Now let's say the hypothesis was "Florida Senator Will Vote Pro Choice". That would make this title a contradiction because it implies the opposite voting decision. Finally, a different neutral example would be if the hypothesis was "The President Is Running For Re-election" because the phrases are unrelated to each other.

Like the STS, NLI could be used to break ties when using only a NN approach recommender system. While NLI labels phrases as three different parts, in theory, a user would only be interested in articles that infer (entail) things that they like. Especially for political news articles, users generally do not like reading articles that contradict their views and prefer articles that align with their views. Based on my research, NLI has not directly been used in recommender systems, but it was used in a study to label customer reviews on products [Tan+21]. Items with similar sounding reviews were labeled as entailment [Tan+21]. Reviews that falsified a review were labeled as a contradiction and reviews unrelated to each other were labeled as neutral [Tan+21]. The authors of this study mentioned that these NLI labels could be incorporated into a recommender system, but creating a system was outside the scope of their work [Tan+21].

3. Methods

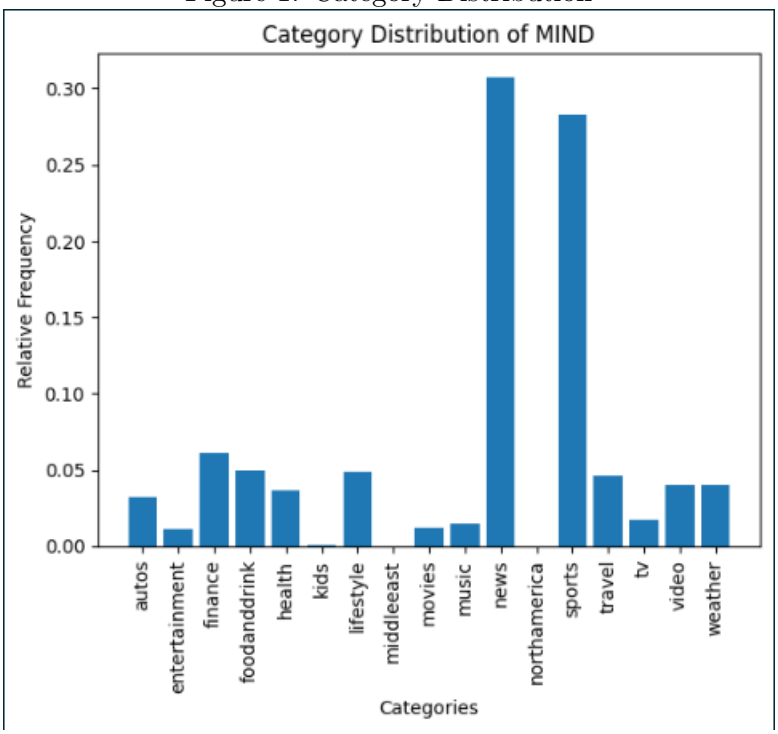
3.1 Data set

The data set used for all experiments is the MIcrosoft News Dataset (MIND) [Wu+20]. This data set contains news and user information. There is a large and small data set. The small data set consists of 50,000 random users of the Microsoft News website. For each user, the user is given a unique id, a list of historical articles read and a list of impressions of shown articles. The impressions are a list of random articles shown to a user at a particular point in time and an indicator representing if a user clicked on the article or not. For the news

information, a news id is given to each article along with its title, category, sub-category, abstract/summary, entities in article and url.

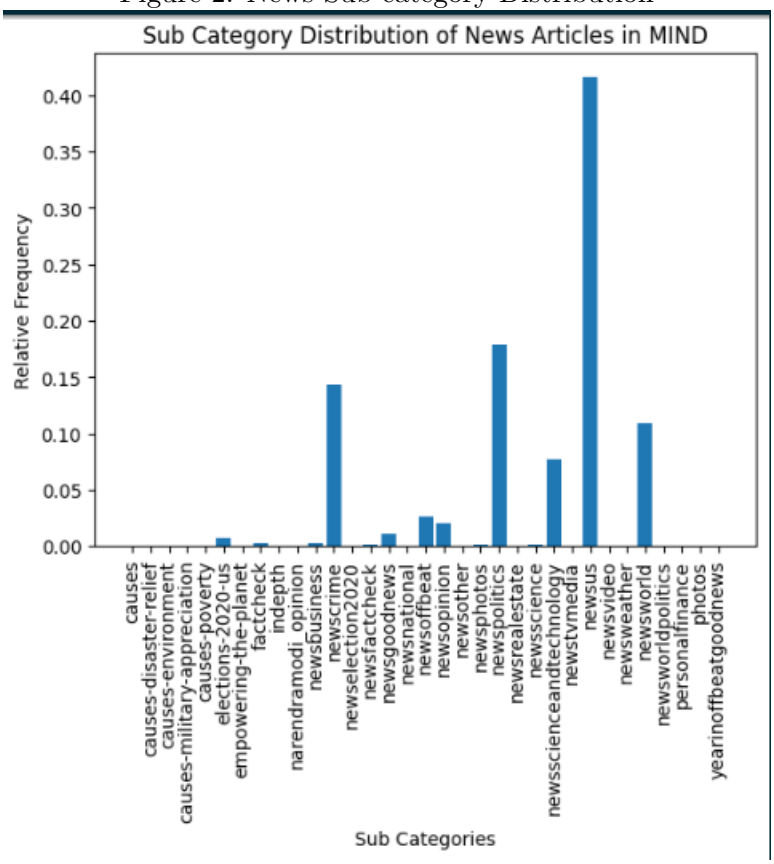
There are 17 unique article categories such as music, news, weather and health. See distribution of category frequency in the image Category Distribution. The majority of the articles are in the news and sports category each representing 30 percent of the news data. The next largest category is finance, which contains 6 percent of the news data. 7 of categories represent around 3 to 4 percent of the news data such as food and drink, lifestyle and auto articles. Then entertainment, movies, tv and music represent 1 percent of the news data. Finally, three categories represent less than .001 of the news data: kids, Middle-East and North America.

Figure 1: Category Distribution



There are 264 sub-categories with 283 category and sub-category combinations. The sub-category with the highest frequency was US news, which represented 1.2 percent of the article data followed by the sub-category of NFL football, which contained 1 percent of the article data. 57 of the sub-categories only had one article in the data set and 16 of the sub-categories only had two articles within the data set. 42 percent of the sub-categories had less than 10 articles within the data set. For articles with news as its category, there were 33 different sub-categories. See distribution of the frequency at the News Sub-category Distribution image. The majority of news articles have the sub-category of US News (41 percent) and next highest sub category is news politics (18 percent). Only 4 sub-categories of news articles have a relative frequency of over 1 percent.

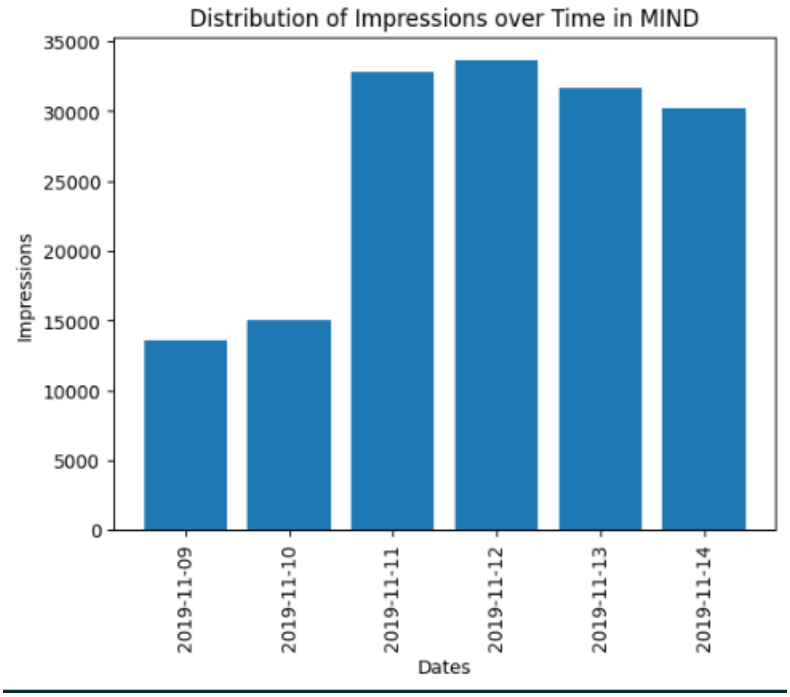
Figure 2: News Sub-category Distribution



For the user data set, users have a huge range of impressions (shown articles). The largest number of impressions shown was 299 articles. Most users were only shown two articles, this represented 20 percent of the users. Two is also the minimum number of impressions for a user to have. 27 percent of the user data had 10 or less impressions. In terms of history, the user with the largest history had 558 articles read and the smallest history size was 1 article read. Around 30 percent of the users had history size of 10 articles or less.

The small data set spans 6 days of data collection from 11-09-2019 until 11-14-2019. All days except for 9th and 10th contain about 20 percent of the data. The 9th and 10th both contain about 10 percent of the data. Due to processing time, this study will only use data from the last day of the MIND experiment (9-14) for the first 3 experiments. See more details about distribution of impressions over time in figure Impressions Over Time.

Figure 3: Impressions Over Time



The reason this data is used is because the impressions will serve as a label field. This will allow for the model to be built using a supervised learning approach. For each user, I will recommend 5 articles. For performance measuring, I will see if each predicted article had a positive (click) impression or not. This data set also contains limited content information about articles, and it provides text data that can be used for the NLP experiments. For the nearest neighbor based approach, only the category and sub-category features will be used for feature scoring. The article title will be used for the NLP experiments.

In order to feed this data into the models, some pre-processing was needed. First, one-hot encoding was performed on the category and sub-category fields. Note, no pre-processing is performed on the title feature until the NLP experiments. Since user history is stored as a list/array for each user, this history list was split into a multiple rows so that each read article had its own row. The same pre-processing was done for the impression field, which was also a list field. Next, the label field was created from splitting the impression indicator from each impression article into its own separate column. Since generating predictions turned out to be a time consuming task, only 1,000 selected users from the last day of the MIND experiment will be scored for the first three experiments. All of the users selected for the experiments will have a small amount of historical articles read and candidate articles shown (10 or less). This was also done to improve the speed of predication generation.

3.2 Experiment 1: KNN Approach

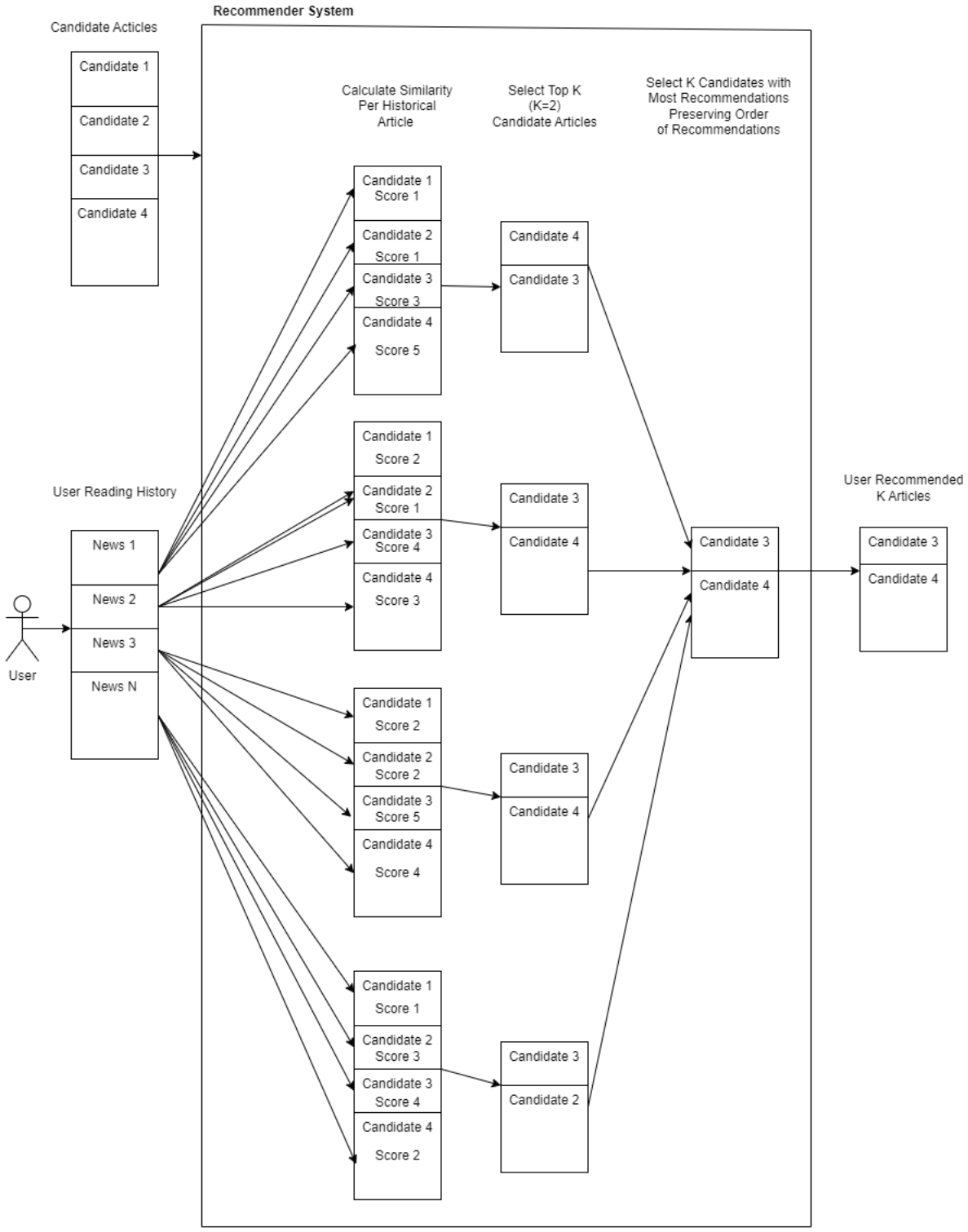
For the first experiment, a nearest neighbor based recommender system is created. This method was chosen because the model is simple and can easily provide explainability as

discussed in the introduction section. The overall architecture is shown in Figure Recommender System Architecture.

For each user, every historical articles' features are scored based on distance between all candidate articles' features. This will be referred to as the feature score. For example, if a user has 3 historical articles and there are 4 candidate articles, there will be 12 scores generated. This score is generated using only the category and sub-category features. For the feature scoring step, the model will have the similarity measure as a hyper-parameter. The similarity measures will be the cosine distance and Pearson Correlation (PC) similarity. These two distance metrics were chosen because they are common measures used in recommender systems according to [Nik+21] and PC was used in the KNN movie recommender system [ASN19]. Since a low cosine distance means items are more similar to each other and a high PC score means the same thing, the cosine distance was negated during the scoring phase to keep the score meaning consistent within the model.

Then for each historical article's score, the top K scores will be taken. For example, if K is 2 then, 6 candidate articles will remain in the running for being recommended. This will be called the neighbor history step. The final step is to select the final top K recommendations based on the articles most recommended by all of the candidates from the neighbor history step. If there are multiple candidate articles with the same number of neighbor history recommendations, then the tie will be broken based on which article that had the highest score followed by title order. For all experiments, each user will be given 5 recommendations. Hyper-parameter tuning will be performed on the distance measure and the number for K. The optimal parameter distance measure found in experiment 1 will be repeated for all of the remaining experiments.

Figure 4: Recommender System Architecture


Recommender System
Candidate Acticles
Candidate 1
Candidate 2
Candidate 3
Candidate 4
Calculate Similarity Per Historical Article
Select Top K (K=2) Candidate Articles
Select K Candidates with Most Recommendations Preserving Order of Recommendations
User Reading History
User
News 1
News 2
News 3
News N
Candidate 1 Score 1
Candidate 2 Score 1
Candidate 3 Score 3
Candidate 4 Score 5
Candidate 4
Candidate 3
Candidate 1 Score 2
Candidate 2 Score 1
Candidate 3 Score 4
Candidate 4 Score 3
Candidate 3
Candidate 4
Candidate 1 Score 2
Candidate 2 Score 2
Candidate 3 Score 5
Candidate 4 Score 4
Candidate 3
Candidate 4
Candidate 1 Score 1
Candidate 2 Score 3
Candidate 3 Score 4
Candidate 4 Score 2
Candidate 3
Candidate 2
Candidate 3
Candidate 4
User Recommended K Articles
Candidate 3
Candidate 4

3.3 Experiment 2: NN Approach with Semantic Textual Similarity

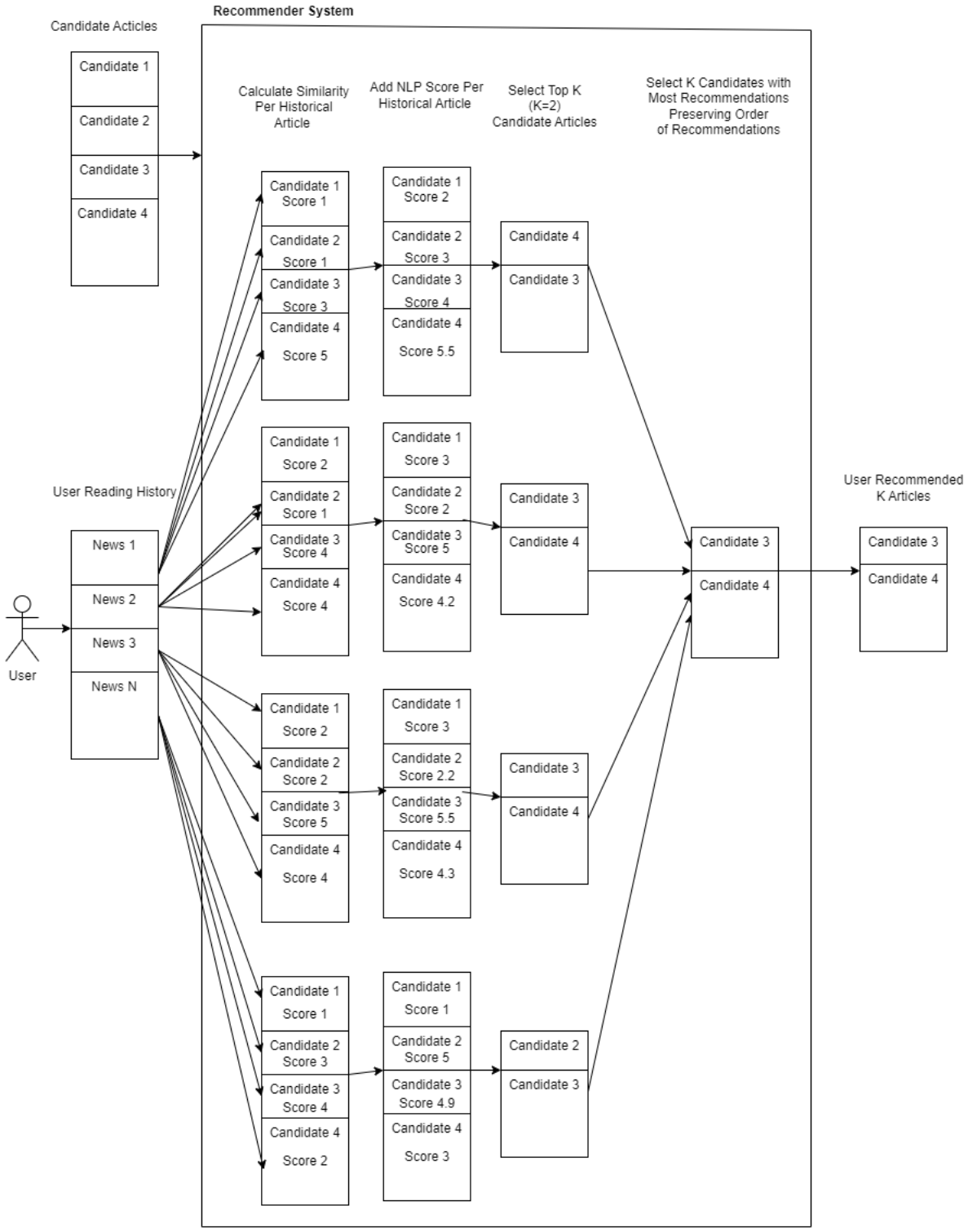
For this experiment, the same nearest neighbor based model from experiment 1 is reused, but an additional step will be done after the feature score is generated. See figure Recommender System NLP Architecture below for an updated architecture model. A semantic textual similarity (STS) score will be generated, which will be added to the feature score. For each user, a semantic textual similarity score will be generated between each candidate article's title and historical article's title. STS model will help improve performance by breaking ties when the feature similarity score from category and sub-category alone provide multiple articles with the score like mentioned in the STS introduction section. STS could also recommend articles whose titles imply they are about a related subject from a user's history yet the category is different. For example, the article "College gymnast dies following training accident in Connecticut" is labeled as a news category, but this article could be interesting to a user that reads lots of sports categorized articles about gymnastics. For this case, the STS model would score this article higher than the NN model would.

The STS score will be generated using Sentence-BERT (sBERT), which is a pre-trained BERT based transformer [RG19] and [RG20]. BERT and RoBERTa are transformer networks meant for sentence pair regression tasks such as semantic textual similarity and natural language inference. Only two sentences are required as input into the networks [RG19]. Sentence-BERT (sBERT) is an expansion of BERT because BERT was not meant for large processing [RG19]. For example, scoring 10,000 sentences would take about 65 hours to compute with BERT [RG19]. With sBert, the time it takes to make the sentence embeddings is 5 seconds and then .01 seconds to get the cosine similarity [RG19]. sBERT uses siamese and triplet network structures to create sentence embeddings that can be used to calculate cosine-similarity between pairs, which serve as the STS score for the experiment [RG19].

Implementing sBERT is very straightforward because all that is required is to download the pre-trained sentence-transformer, feed two strings, the article titles for this experiment, into the model embedding method and then call the cosine-similarity method. No additional fine-tuning on the model will be performed due to time limitations with the project. sBert was chosen for this step because of its speed, ease of use and is considered a state of art NLP model.

Hyper-parameter tuning will be performed on the K value. No other tuning will be performed. This tuning will be performed on only 100 users. The optimal distance measure from experiment 1 will be used in the nearest neighbor step for this experiment.

Figure 5: Recommender System NLP Architecture


Recommender System
Candidate Acticles
Candidate 1
Candidate 2
Candidate 3
Candidate 4
Calculate Similarity Per Historical Article
Add NLP Score Per Historical Article
Select Top K (K=2) Candidate Articles
Select K Candidates with Most Recommendations Preserving Order of Recommendations
User Reading History
News 1
News 2
News 3
News N
User
Candidate 1 Score 1
Candidate 2 Score 1
Candidate 3 Score 3
Candidate 4 Score 5
Candidate 1 Score 2
Candidate 2 Score 3
Candidate 3 Score 4
Candidate 4 Score 5.5
Candidate 4
Candidate 3
Candidate 1 Score 2
Candidate 2 Score 1
Candidate 3 Score 4
Candidate 4 Score 4
Candidate 1 Score 3
Candidate 2 Score 2
Candidate 3 Score 5
Candidate 4 Score 4.2
Candidate 3
Candidate 4
Candidate 1 Score 2
Candidate 2 Score 2
Candidate 3 Score 5
Candidate 4 Score 4
Candidate 1 Score 3
Candidate 2 Score 2.2
Candidate 3 Score 5.5
Candidate 4 Score 4.3
Candidate 3
Candidate 4
Candidate 1 Score 1
Candidate 2 Score 3
Candidate 3 Score 4
Candidate 4 Score 2
Candidate 1 Score 1
Candidate 2 Score 5
Candidate 3 Score 4.9
Candidate 4 Score 3
Candidate 2
Candidate 3
Candidate 3
Candidate 4
User Recommended K Articles
Candidate 3
Candidate 4

3.4 Experiment 3: NN Approach with Natural Language Inference

For experiment 3, all steps will be the same as in experiment 2 except instead of semantic textual similarity score being generated between candidate and historical article titles an entailment score will be created. This entailment score is taken from using the sBERT model [RG19]. sBERT was trained on SNLI [Bow+15] and Multi-Genre NLI [WNB18] data sets. SNLI contains 570,000 sentences pairs with labels of contradiction, neutral and entailment and MulitNLI contains 430,000 sentence pairs. To use sBERT as an entailment score, the cross encoder model from the sentence transformer must be populated and then the prediction method should be called. This will return 3 scores relating to classifying the phrases as a contradiction, entailment and neutral. Only the entailment score will be used. This is because titles unrelated (neutral) to each other will not provide insight into a recommendation and contradicting titles will likely result in unfavorable predictions. For example, if a user is interested in political articles, they likely will not be interested in articles that contradict their reading history. This NLP based approach will help fill in the gaps that the NN model provides.

This model will be built with an indicator to have the entailment score calculated once or twice. If the indicator is set to twice, then the first score will be generated by having the candidate article labeled as the premise and the historical article labeled as the hypothesis. Then the second score is generated from the opposite (historical article as the premise and candidate article as hypothesis). If the indicator is set to be called once, then the candidate article is the premise and the historical article is the hypothesis. This one direction is chosen because the premise usually is the more specific phrase and the hypothesis is often more general. This direction was chosen because I believe that if a user likes an article, they would prefer something more specific about the topic rather than a more general article. The number of times the entailment score is generated will be the only hyper-parameter tuning for this model. This tuning will be performed on 100 users. All other parameters will be the same as experiment 2.

3.5 Experiment 4: News Only NN Approach

For the 4th experiment, if a user has read mostly articles from the news category, then that user was selected for this experiment. The threshold for the amount of news categorized articles read was set at 75 percent. The reason for a threshold is because there were not enough users that only read news categorized articles in the data set. Only around 200 users read only news categorized articles. The amount of users for this experiment is just over 500 users. This experiment was chosen to determine if our recommender system will have better performance if we have a specific target audience rather than a diverse audience. Based on the targeted audience, we can determine the type of repository for the recommender system. If the target audience is specific, then the repository of articles should all be of the same category. Otherwise, the repository of articles should include various categories. All other steps for this experiments will be the same as in experiment 1. No hyper-parameter tuning was performed for this experiment. The optimal distance and K value from experiment 1 will be reused for this experiment. The only difference between experiment 1 and experiment 4 is the users selected for this experiment.

3.6 Experiment 5: News Only NN Approach with STS

For the final experiment, the same users from experiment 4 will be selected for this experiment. The model will be the exactly the same as experiment 2. Again, the only difference will be users selected for the experiment. No hyper-parameter tuning was performed on this model. The same optimal distance and K value from experiment 2 will be used for this experiment.

3.7 Metrics

For all experiments, there were two different metrics used to rate the performance of the models: accuracy and information retrieval's Mean Average Precision @ K (MAP@K). For both metrics, the higher the score, the better are recommendations are.

Accuracy is calculated using each user's number one recommendation.

$$Accuracy = \frac{\sum ClickedRecommendation}{TotalRecommendations}$$

For this calculation, the total number of recommendations is the same as the number of users since only the highest recommendation for each user is used. This score is also equivalent to MAP@1. This score informs us of how well our model is doing when recommending only a single article to a user.

MAP@K is calculated from using the top 5 recommendations provided from each experiment and then scoring only K (3 and 5) of those recommendation. This formula requires a few different metrics to produce the score.

$$MAP@K = \frac{\sum AP@K}{N}$$

where N is the number of users.

The Average Precision @ K (AP@K) is calculated per user with the following algorithm

$$AP@K = \frac{\sum_{i=0}^{k} P@K}{K}$$

The Precision @K (P@K) is from the following

$$P@K = \frac{NumberofRecommendationClickedOn}{K}$$

K represents the number of recommendations per user. Precision at K is the fraction of top K selected articles that are of interest to a user.

To explain the metric with an example, let's say the model produces 5 recommendations representing 1 0 1 0 1 1. A 1 means that the recommendation was actually clicked (viewed) by the user. A 0 means the recommendation was a miss. This recommendation list is in order of scoring, as in recommendation at index 0 had the highest recommendation score.

If we want to find AP@3 we would do the following

$$AP@3 = \frac{P@3 + P@2 + P@1}{3}$$

where the P@K would be the following

$$P@3 = \frac{2}{3}, P@2 = \frac{1}{2}, P@2 = \frac{1}{1}$$

This would make AP@3 =1/12. Then to get MAP@3, we would take the average of each users AP@3. With only one user in this example, AP@3 is the same as MAP@3. AP@K and P@K are precision measures for a single user. While MAP@K is used to evaluate the precision of the entire system (all users).

4. Results

4.1 Hyper-parameter Tuning Results

Below are the results of hyper parameter tuning. Only accuracy was used for tuning. Tuning was only done for experiments 1,2 and 3. All time is recorded in minutes.

Experiment	Distance	K Value	Times Scored	Accuracy	Time
NN	cosine	3	NA	.219	2
NN	pearson	3	NA	.185	10
NN	cosine	5	NA	.203	1.5
NN	pearson	5	NA	.188	11
NN with STS	cosine	5	NA	.32	2
NN with STS	cosine	3	NA	.33	2
NN with NLI	cosine	3	1	.36	5
NN with NLI	cosine	3	2	.42	10.5

Table 1: Hyper Parameter Tuning Experiment 1,2 and 3

Table 1 includes all of the results for tuning the value of K, the distance/similarity function and the times scored in experiments one (NN), two (STS) and three (NLI). The time recorded is in minutes. The value NA means the parameter was not applicable to the given model.

4.2 Final Experiment Results

Below are the final results using the optimal parameters from hyper-parameter tuning. The results are for all 5 experiments using accuracy, MAP@3 and MAP@5.

Model	Accuracy	MAP@3	MAP@5
NN	.25	.20	.18
NN with STS	.24	.22	.20
NN with NLI	.22	.21	.19
News NN	.15	.12	.11
News NN with STS	.17	.17	.14

Table 2: All Experiment Results

Table 2 includes all the final scores for all experiments. Experiment 1 is NN, experiment 2 is NN with STS, experiment 3 is NN with NLI, experiment 4 is News NN and experiment 5 is News NN with STS.

User	Clicked Article	Category	Sub-Category	NN	STS	NLI
U27158	Georgia executes man convicted of killing convenience store clerk	News	Crime	1	5	1
U18205	Powell's Warning to Congress About the Next Recession	Finance	Markets	0	5	2
U18205	This Arctic blast is in its final day. But the cold isn't over quite yet	Weather	Top Stories	0	4	0

Table 3: Ranking Examples for Experiments 1,2 and 3

Table 3 includes the ranking (order the article was recommended) of some example clicked articles for a given user. If the ranking is zero, then the article was not recommended by the model. NN is experiment 1, STS is experiment 2 and NLI is experiment 3.

User	Clicked Article	Category	Sub-Category	News NN	News STS
U7166	Bannon calls Pelosi's impeachment strategy "actually quite brilliant"	News	Politics	4	3
U65050	PGE is offering 13.5 billion in compensation to wildfire victims	Finance	Companies	0	0

Table 4: Ranking Examples for Experiments 4 and 5

Table 4 includes the ranking (order the article was recommended) of some example clicked articles for a given user. If the ranking is zero, then the article was not recommended by the model. News NN is experiment 4 and News STS is experiment 5.

5. Discussion

During the hyper-parameter tuning phase for experiment one (NN), the cosine distance with K as 3 had the best overall performance. For experiment 2, the K value of 3 also had the best performance. For the third experiment, computing the entailment score twice had the optimal performance. All of these parameters were used for the final experiments.

The model with the highest accuracy score was from experiment one, NN. It had an accuracy score of .25 and the second experiment, NN with STS, was only .01 lower. When only news categorized data is used, the NN model performed poorly. This is reflected in the fact that experiment 4 (News only NN) had the worst accuracy score, which was .15. The

NN model performed better with a more diverse repository (experiment 1). If a user would only want to see one recommendation at a time, then the NN based approach would be the best suited model to use since accuracy is based on the top ranking recommendation. But overall, all models had low accuracy scores.

While in a perfect world, we would want a user to like all of the recommenders provided; having a user only like a few of the recommendations is more reasonable. This is why the MAP scores were used. For precision, the second experiment (NN with STS) had the best scores for both MAP@3 and MAP@5, which were .22 and .20. The NN with NLI model (experiment 3) had similar yet slightly lower scores than the NN with STS model. The precision scores from the first two NLP models show that NLP can help improve performance of our model when showing a user 3 or 5 recommendations. The scores indicate that the users are more likely to view more of the articles shown from the NLP models (experiment 2 and 3) than just the NN model (experiment 1). Even if the system is designed to only recommend a specific type of article (such as news or sports categorized articles), the NLP model provided better recommendations than the NN model as shown in the News NN and News NN with STS MAP scores.

When looking at recommendation results from the first three experiments (Table 3), we can see that the NLP models were able to recommend more of the clicked articles than the NN model. There where many cases were the NN model didn't recommend a clicked article, but the NLP models were able to, which is demonstrated from user U18205. While sometimes the NN model could predict its top ranking article to be a clicked article, like user U27158, the NLP models were still able to rank the clicked article within its top 5 recommendations. Both of these examples are common situations for the first 3 models.

For the example records shown in Table 4, we see that the both news based models were able to predict the clicked article if the category was news like from user U7166. But if the category was not news, both models were unable to predict the viewed article as shown with user U65050. Both of these situations were common for all users in the 4th and 5th experiments.

Overall, the NLP models demonstrated that they had better performance than the NN based model alone as reflected in the STS and NLI MAP scores and the News based STS experiment (experiment 5). While the NLP models did perform better, the scores still are on the low side. There are handful of reasons for this poor performance. For one, the range of user impressions was wide, which made the metrics not fully capture the quality of our recommendations. While all users were recommended their top 5 ranked articles, not all users viewed that many of the candidate articles. For many of the users, they only viewed one or two articles. So if a user only viewed two articles yet we predicted those 2 articles with 3 additional irrelevant articles, then our recommendations would have resulted in a lower score than if we only recommended 2 articles total. If users were selected that had at least 5 clicks/views of candidate articles, then the performance might have been better, but there would have been less data for the experiments.

Another explanation is that many users want a diverse selection of recommendations. Both NLP and the NN models are trying to find articles similar to a user's history. Frequently, users can get bored with repetitive like recommendations and instead want to explore new topics. This issue is often referred to as over-specialization [Nik+21], which means the system assumes users interests will be static and narrow. This issue also relates

the low performance in the news categorized experiments (experiment 4 and 5), which had poorer performance than the first 3 experiments.

Another reason the news categorized experiments had poor performance is that this data set was not meant for this type of experiment. None of the users were shown only a specific category of articles such as news or sports. Instead, all users were shown a diverse range of candidate articles. This experiment might have had better results if there was a data set with only news categorized data as its candidate articles.

The final reason for the large number of irrelevant recommendations is that preferences can be time sensitive. For example, a user might only be interested in political articles during election season. If additional features such as publication date was used in the model, then performance might have been improved. But one of the intentions of this research was to work with a limited data set. Given that all the models only used 2 or 3 features, the performance is as expected.

6. Conclusion

From this research, it is clear that NLP tasks such as semantic textual similarity and natural language inference can be used to boost a recommender systems performance. If a system can only handle using one NLP task, then the semantic textual similarity model should be implemented since the STS experiments had the best overall performance. Although the data set had a limited amount of features, all models were still able to provide relevant predictions to the users. The nearest neighbor based model demonstrated that a system could predict the article that a user is mostly likely to view, but it was unable to handle recommending multiple relevant articles. The NLP tasks of semantic textual similarity and natural language inference were able to fill in the gaps that the nearest neighbor based model had. While the experiments supported my hypothesis that NLP tasks can improve recommender systems output, the models with NLP tasks still had relatively low performance. The data set for the experiments fit the bill for using a dataset with limited features and its impressions made it straighforward to measure the performances of the models. But due the range of impressions per user and the diverse categories of impressions, the quality of the metrics was diluted. Although the models didn't have the best performance, the experiments showed that adding NLP tasks could improve the performance of a recommender system and that relevant recommendations can be made even when the data set is limited. For future work, additional experiments using more NLP tasks like named entity recognition and text summarization could be tested. Additionally, all of the NLP tasks could be combined into one model instead of separating them into separate models.

References

[ASN19] Rishabh Ahuja, Arun Solanki, and Anand Nayyar. "Movie Recommender System Using K-Means Clustering AND K-Nearest Neighbor". In: *2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence)* (2019), pp. 263–268.

[Bow+15] Samuel R. Bowman et al. "A large annotated corpus for learning natural language inference". In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 2015.

[Cap+12] Michel Capelle et al. "Semantics-Based News Recommendation". In: *Proceedings of the 2nd International Conference on Web Intelligence, Mining and Semantics*. WIMS '12. Craiova, Romania: Association for Computing Machinery, 2012. ISBN: 9781450309158. DOI: 10.1145/2254129.2254163. URL: https://doi.org/10.1145/2254129.2254163.

[Jar+22] Shatha Jaradat et al. "Fashion Recommender Systems". In: *Recommender Systems Handbook*. Ed. by Francesco Ricci, Lior Rokach, and Bracha Shapira. Springer US, 2022, pp. 1015–1055. DOI: 10.1007/978-1-0716-2197-4_26. URL: https://doi.org/10.1007/978-1-0716-2197-4%5C_26.

[Nik+21] Athanasios N. Nikolakopoulos et al. "Trust your neighbors: A comprehensive survey of neighborhood-based methods for recommender systems". In: *CoRR* abs/2109.04584 (2021). arXiv: 2109.04584. URL: https://arxiv.org/abs/2109.04584.

[RG19] Nils Reimers and Iryna Gurevych. "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Nov. 2019. URL: https://arxiv.org/abs/1908.10084.

[RG20] Nils Reimers and Iryna Gurevych. "Making Monolingual Sentence Embeddings Multilingual using Knowledge Distillation". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Nov. 2020. URL: https://arxiv.org/abs/2004.09813.

[Tan+21] Chahat Tandon et al. "Use of Natural Language Inference in Optimizing Reviews and Providing Insights to end Consumers". In: *2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS)*. Vol. 1. 2021, pp. 60–65. DOI: 10.1109/ICACCS51430.2021.9442026.

[WNB18] Adina Williams, Nikita Nangia, and Samuel Bowman. "A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, 2018, pp. 1112–1122. URL: http://aclweb.org/anthology/N18-1101.

[Wu+20] Fangzhao Wu et al. “MIND: A Large-scale Dataset for News Recommendation”. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, July 2020, pp. 3597–3606. DOI: 10.18653/v1/2020.acl-main.331. URL: https://aclanthology.org/2020.acl-main.331.