Fake News Identification using Stance Detection

Amanul Haque
ahaque2@ncsu.edu
NC State University

Suvodeep Majumder
smajumd3@ncsu.edu
NC State University

Abstract

In this work, we try to identify fake news using stance detection over News articles. We identify the stance of a news article towards a given headline and formulate this as a multivariate classification problem with 4 classes. The stance of a news article towards a headline could be one of the following, 'unrelated', 'discuss', 'agree' and disagree', with each class being exclusive. The dataset used was collected from Fake News Challenge (FNC-1) which has over 75000 labeled data-instances, with 73% belonging to 'unrelated' that skews the dataset. We established the baseline of 83% overall accuracy using Gradient Boosting algorithm. Our Multi-layer Perceptron with Tf-idf beats this baseline by 4% (87% overall accuracy) while word-embedding also achieves comparable performance and with much-balanced precision, recall for all classes compared to baseline. We further improve the results through dimensionality reduction using a text summarizer. Model trained on a smaller dataset with dimensions reduced show significant improvement with overall accuracy at 93% (10% higher than our baseline).

1 Introduction & Background

The New York Times defines "Fake news" as "made-up stories written with the intention to deceive" and published in formats similar to those of traditional "real new" [1]. Fake News is a huge threat towards high-quality journalism and well-informed public discourse. Fake news detection is vast area of active research, in this project we try to explore a subset of that called "Stance Detection" (i.e. identifying click baits fake news). This project, focuses on identifying News article stance with respect to the News headlines, where given a headline and the corresponding News article text, can we determine the relationship (if any) that exists between the two.

The usages of NLP to create solution for news stance detection comes from the recent success of deep neural network's ability to understanding human language as a sequence to sequence modeling. Over the years many researches has been performed for understanding how to understand the relation between two texts, such as target-specific stance prediction has been performed for tweets [2, 4], online debates [9]. There are many different types of strategies which has been employed in these studies, such as exploring structural and linguistic and lexical features, jointly modeling disagreement and collective stance using probabilistic soft logic, neural models with encoding [3], model summarization [7], inclusion of attention networks [6], to list a few.

2 Dataset Overview

The dataset for our project was taken from Fake News Challenge (FNC) [1] organization's repository[1]. The goal of the Fake News Challenge was to explore how artificial intelligence technologies, particularly machine learning and natural language processing, might be leveraged to combat the fake news problem. FNC focuses on a smaller sub-problem of the larger fake news issue by trying

[1] https://github.com/FakeNewsChallenge/fnc-1

32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, Canada.

to assess the veracity of a news story given a corresponding headline. This can be achieved using stance detection techniques wherein the stance to be measured is that of a News article (complete News body as a text document) towards that of a given headline. The given dataset has over 75,000 data instances and each data instance is a stance tuple containing a -

- **Headline:** A news headline
- **News article body text:** Complete news article text corresponding to the given headline.
- **Stance:** The true stance of the article with respect to the headline.

The data set contains 4 different types of stance (which we use as our class labels) -

- **agree:** The article text agrees with the headline (7% of all data instances)
- **disagree:** The article text disagrees with the headline (2% of all data instances)
- **discuss:** The article text is a discussion of the headline, without taking a position on it (18% of all data instances)
- **unrelated:** The article text is unrelated to the headline (i.e. it does not address the same topic) (73% of all data instances)

As shown by the distribution in fig 1, there is a huge imbalance in data distribution across various classes. Just predicting everything into 'unrelated' class (majority) will yield an accuracy of 73%. Hence we have used f1-score for each class individually to evaluate performance of our models instead of just overall accuracy. To build testing and training set we randomly split the data into 67% (roughly 50,000 instances) for the training set and remaining 33% (roughly 25,000) for the testing set. We maintained a similar distribution across classes for both the training and testing set (as in the original dataset).

- **agree:** 7% for both training and testing.
- **disagree:** 2% for training and 3% for testing dataset.
- **discuss:** 18% for both training and testing dataset.
- **unrelated:** 73% and 72% for training and testing respectively.

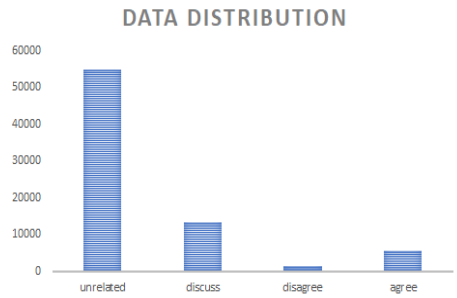


Figure 1: Data distribution across different classes (stance)

3 Methodology and Experimental Setup

3.1 Data Preprocessing:

As our input data is text headline and article body of News articles, it needs some preprocessing before feeding the input to the model. We performed some standard text preprocessing over the text data. Following preprocessing steps were applied: (1) Removed unnecessary character encoding (2) Stop words were removed. (3) Punctuation were removed. (4) URLs replaced with a token <URL>. (5) Lemmatization: Lematized each word in the text to its corresponding root word. (6) Word and sentence tokenization.

To generate word embeddings we used pretrained "Google News" word2vec, which is trained on 3 billion English words from Google News corpus [2]). We used a dimension of 300 for each vector representation.

3.2 Approach 1 (Baseline)

We first tried a few standard classification algorithms on the dataset to establish a baseline performance for this dataset. We used preprocessed text from headline and article body as input features to the model and the predicted values were one of the 4 labels described under section 2. The 2 best performing models were *K-Nearest Neighbours* (with n=9) and *Gradient Boosting classifier*, see results in table 1.

The classifiers performed decently and gave an accuracy of 81% and 83% respectively. However, the classifiers were biased towards majority class ('unrelated' and 'discuss') which constitutes 90% of this dataset. This can be seen in the confusion matrix table 2. The classifier was unable to identify many instances of 'agree' and 'disagree' and needs some more sophisticated model to be able to identify instances of these minority classes.

3.3 Approach 2 (Multi-layer Perceptron)

The baseline model performed well on majority classes but did poorly on minority ones. To further improve the performance of our model for minority classes we adopted a multi-layer perceptron (MLP) approach as described by [5]. The MLP classifier has one hidden layer of 100 units and a softmax on the output of the final linear layer. We use the Rectified Linear Unit (ReLU) activation function as non-linearity for the hidden layer. The model architecture is shown in fig 2.

We used following as features for the model:

- Word vectors of headline text
- Word Vectors of news article body text
- Cosine similarity between news article body and headline
- Hedge word/phrase counts (Words that make a statement less assertive, eg: Seems, possibly, maybe, looks like, etc.)
- Sentiment of headline and News article text

We used 2 approaches for word vectorization:

- Word-embeddings: Used Google News pretrained model - Each vector of dimension 300
- Bag-of-words: Used Tf-idf - Each vector of dimension 5000

We also experimented with different length of news article text

- Considering only first 5 sentences
- Considering top 5 sentences based on similarity with headline.

The model improved the classification for minor classes significantly, see results in table 1.

3.4 Approach 3 (Dimensionality reduction)

Text is high dimensional data as each word forms a dimension, and as the vocabulary size is pretty huge for this dataset, our model could potentially perform better with reduced dimensions. We adopted a text summarizer as described in [8] to reduce text dimensions (we used their pretrained Bert model). It is specially designed for news text as a sentence level summarizer that produces a news headline like summary for each sentence.

Some examples of text summarization using this model.

[2] https://drive.google.com/file/d/0B7XkCwpI5KDYNlNUTTlSS21pQmM/edit?usp=sharing

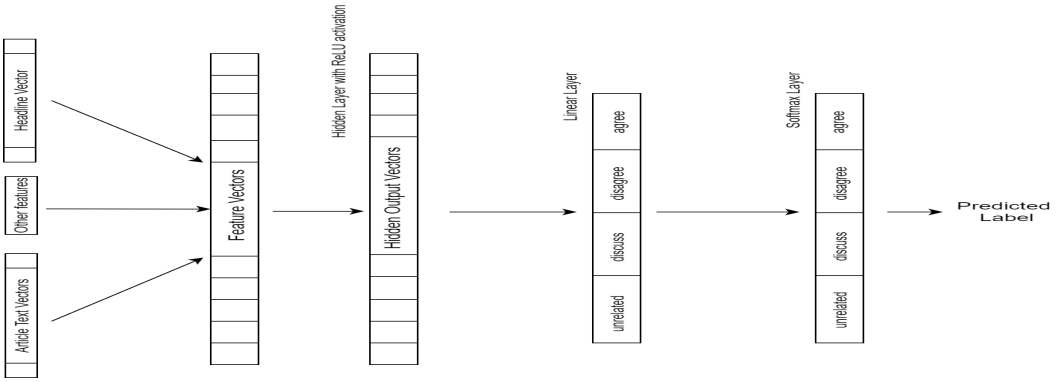


Figure 2: Architecture for Multi-layer Perceptron used in approach 2 and discussed under section 3.3

- Original Sentence: A small meteorite crashed into a wooded area in Nicaragua's capital of Managua overnight, the government said Sunday.
  - Summarized sentence: *Small meteorite crashes in nicaragua.*
- Original Sentence: HBO plans to launch its stand-alone Net video service next month on Apple TV and other devices, according to the International Business Times.
  - Summarized sentence: *HBO to launch stand-alone Net video service.*

As the examples above demonstrate the text summary contains all the relevant details from the original sentence in the same order and grammatical structure but in highly condensed form.

Unfortunately we could not try this approach on the whole dataset as the model has a long running time and runs into memory Error even with small batch size. We randomly sampled a smaller subset of the dataset to test performance.

Data distribution for sample dataset (We maintained a similar distribution for training and test set):

- Training set: Total 4319 training instances (Distribution: Unrelated: 4063, Discuss: 147, Agree: 103, Disagree: 6)
- Testing set: Total 1733 testing instances (Distribution: Unrelated: 1568, Discuss: 113, Agree: 45, Disagree: 7)

3.5 Approach 4 (LSTM with bidirectional conditional encoding)

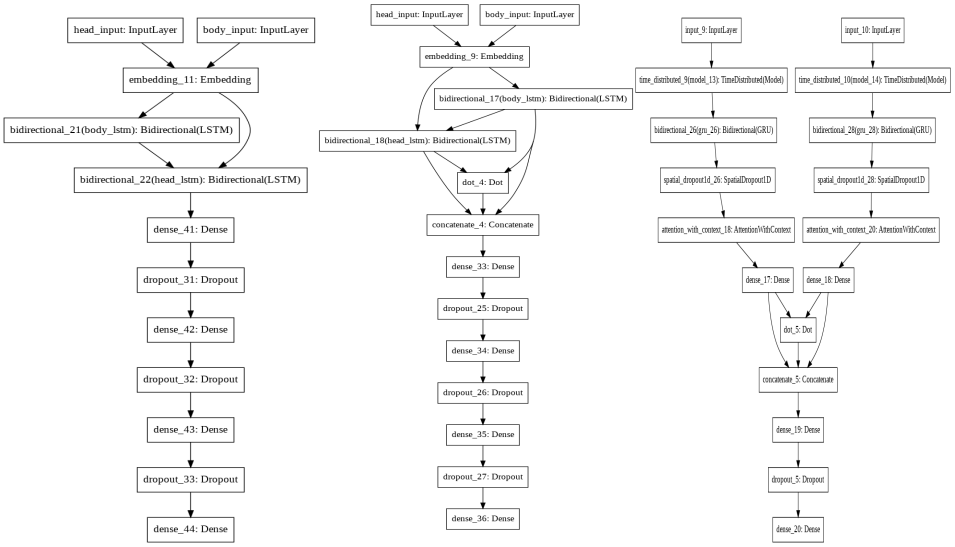
In this approach, we have explored the use of multiple recurrent network layers to encode headlines and news bodies before classifying the resulting state vectors with a softmax transformation. Our approach was to use conditional encoding, which involves recurrent cells placed in sequence rather than in parallel. Here we use a shared embedding layer (non-trainable) to get the vector representation of each sentence in headline and body for each stance, then we first sent the headline text through an LSTM layer and then initialized another LSTM layer for the body of the news articles with the final hidden state vector from the first LSTM. From here we perceived two different approaches, in one approach we directly use the output of the second LSTM to initialize three dense layer, and then dropout layers in between (to regularize and prevent over fitting) . Figure 3a show the model architecture for the bidirectional conditional encoding model.

In the second approach, we did the same conditional encoding, but instead of sending the LSTM encoded data from the second LSTM directly to the dense layer, we perform a dot product between the headline encoded output from first LSTM and news article body's encoded output from the second LSTM. The result of the dot product is concatenated with the encoding form the 2 LSTM layers and that is used as the input as the first dense layer. Figure 3b shows this summarization model, with bidirectional conditional ending.

In both the architecture, we have used LSTM with 100 units, recurrent dropout with a value of 0.2. For the Dense layers we have used relu activation function with 100 units, except for the final output layer with softmax activation function.

3.6 Approach 5 (LSTM with hierarchical attention network)

In this approach we use hierarchical attention network, to classify the news stance. A HAN model exhibits a hierarchical structure very similar to that of documents, wherein two levels of attention mechanisms (word-level and sentence-level) are applied, which enables it to identify the key representative and dictating features for document classification. It consists of a word sequence encoder, a word-level attention layer, a sentence encoder and a sentence-level attention layer. First, we use an embedding layer which maps the words into a 100-dimensional embedding space. These embedded vectors are then fed to a bidirectional LSTM Layer (word sequence encoder) which comprises of 100 units. In this layer, the words are annotated to incorporate contextual information. Then we add an attention layer on top of this to extract only those words that are important to the meaning of the sentence and we aggregate the representation of those informative words to form a sentence vector. Now, a similar set of layers is added to extract document vectors. Here a bidirectional LSTM Layer with 100 units serves as the sentence encoder and an attention layer is added on top of this to extract only those sentences that are meaningful representatives of the context, sentiment conveyed or the category of the document. This document vector encompasses key information that helps in document classification. For this study we use 2 parallel HAN models, one to encode the news headers and another to encode the news article body. The result of these 2 attention models are send to a dot product layer to get a combined representation and then used that along with the output of the 2 attention models to concatenate. The output concatenation of the 2 attention layers and dot layer are fed into a fully connected layer which flattens the output into a class-dimensional vector with probabilities indicating the likelihood of the document belonging to a particular class. The loss is computed using categorical cross entropy loss for multi-class classification task and binary cross entropy loss for binary classification. Similarly, the activation for the output is softmax for multi-class and binary classification tasks along with Adam optimizer.



(a) LSTM with bidirectional conditional encoding

(b) LSTM with bidirectional conditional encoding with summarization

(c) Hierarchical Attention Network with text summarization

Figure 3: Deep Neural Network Architecture for Stance Detection. This figure shows the 3 different network architecture used in our study.

3.7 Choosing appropriate hyperparameters:

Though our models included up to a dozen hyperparameters to support experiments with capabilities like learning rate decay and gradient clipping, we focused on a few for this experiment -

- **Learning Rate:** The learning rate of the model decides the amount changes in the neuron to neuron weights, A high weight means for each item the changes will be large, this might result in model not converging. On the other hand, a small weight may result in local optima or very slow convergence. So choosing appropriate learning rate is essential. For this experiment, we varied the learning rate from 0.001 to 0.1 and we finally set for 0.001 as this gave us best results.
- **Batch Size:** batch size decides number of samples per gradient to be update. With large batch sizes, the updates will be too large, and may result in non-convergence, while with very small the convergence rate would be very slow. we varied the number from 32 to 128 and found 50 to give us the best result.
- **Token length for headlines and articles:** Both the header and article body contained varied number of sentences and words, the ranges for number of sentence from 1 to 4, and in article body the number of sentence varied from 4 to 42. As the model takes fixed length inputs, We selected 2 different approaches to handle this. First one is to fix the number of sentence (also the number of words in each sentence) to be selected (we fixed it to the median) and on the other approach, we used padding to make all of them similar.
- **Number and types of layers:** In Deep neural network, each layer represents the encoding of the model, as the number of layers increases the model's complexity increases. Also for there are different type of recurrent neural layers, that can be used. We tried with GRU, LSTM as RNN layers, also we tried different dropout layer like spatial dropout, normal dropout . In the final architecture we used LSTMs and mix between spatial dropout, normal dropout depending on the inputs. We also varied the number of layers in the model to see the test accuracy and we found as we increase the number of layers the model's complexity increases and it over fits.

4 Results

Table 1: Results for the first 3 approaches discussed in sections 3.2, 3.3 and 3.4

Sn	Approach	Word Vectorization	Classifier	F1-score				Accuracy
				unrelated	discuss	agree	disagree	
1	Approach 1	Word2Vec	Gradient Boosting	0.93	0.64	0.0	0.01	0.83
2	Approach 1	Word2Vec	K-nearest Neighbour	0.93	0.57	0.01	0.16	0.80
3	Approach 2	Word2Vec	Multi-layer Perceptron	0.92	0.57	0.00	0.41	0.81
4	Approach 2	tf-idf	Multi-layer Perceptron	0.97	0.73	0.09	0.42	0.87
5	Approach 3	Word2Vec	Multi-layer Perceptron	0.97	0.57	0.25	0.44	0.93

Table 2: Confusion Matrix for approaches discussed in sections 3.2 , 3.3 and 3.4

Confusion Matrix for baseline model (section 3.2) using gradient boosting algorithm				
	agree	disagree	discuss	unrelated
agree	6	3	1390	504
disagree	1	1	379	316
discuss	23	5	3146	1290
unrelated	3	2	474	17870

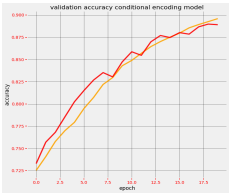
Confusion Matrix for approach 2 (section 3.3) using word2Vec and Multi-layer Perceptron classifier				
	agree	disagree	discuss	unrelated
agree	767	4	492	640
disagree	147	0	136	414
discuss	659	0	2218	1587
unrelated	223	2	408	17716

Confusion Matrix for approach 2 (3.3) using Tf-idf and Multi-layer Perceptron classifier				
	agree	disagree	discuss	unrelated
agree	624	21	1156	102
disagree	91	35	477	94
discuss	329	32	3899	204
unrelated	48	1	622	17678

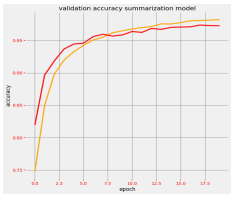
Confusion Matrix (reduced dataset) for approach 3 (3.4) using Word2Vec and Multi-layer Perceptron classifier				
	agree	disagree	discuss	unrelated
agree	23	0	3	19
disagree	1	1	2	3
discuss	16	0	53	44
unrelated	20	0	15	1533

5 Discussion

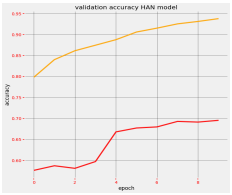
Our baseline model as discussed under section 3.2 uses word2vec for word vectorization and Gradient Boosting classifier. The model achieved an overall accuracy of 83% and F1-score of 0.93, 0.64, 0.0 and 0.01 for the classes *unerelated, discuss, agree* and *disagree* respectively (rows 1 and 2 in table‘1). The model works well for the majority class but fails to identify instance from minority class.



(a) LSTM with bidirectional conditional encoding

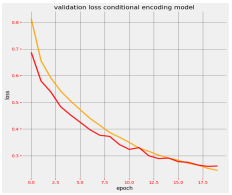


(b) LSTM with bidirectional conditional encoding with summarization

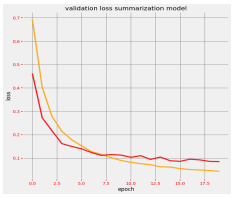


(c) Hierarchical Attention Network with text summarization

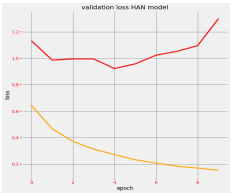
Figure 4: Validation accuracy for the deep neural network models showed in figure 3.



(a) LSTM with bidirectional conditional encoding



(b) LSTM with bidirectional conditional encoding with summarization



(c) Hierarchical Attention Network with text summarization

Figure 5: Validation loss for the deep neural network models showed in figure 3.

Table 3: Confusion Matrix for model LSTM with bidirectional conditional encoding as described in section 3.5 and Hierarchical attention network as described in

Confusion Matrix for bidirectional conditional encoding with without Summarization				
	agree	disagree	discuss	unrelated
agree	596	5	434	789
disagree	193	5	168	331
discuss	719	26	2144	1575
unrelated	845	15	1265	16224

Confusion Matrix for bidirectional conditional encoding with Summarization				
	agree	disagree	discuss	unrelated
agree	512	20	293	1078
disagree	171	4	67	455
discuss	758	12	1840	1854
unrelated	1224	34	1169	15922

Confusion Matrix for Hierarchical attention network				
	agree	disagree	discuss	unrelated
agree	572	27	293	968
disagree	164	7	67	515
discuss	139	62	1734	1870
unrelated	1032	37	1361	15962

Approach 2 using Multi-layer Perceptron (MLP) improves the performance for the minority classes (rows 3 and 4 in table 1) without deteriorating the performance for the majority class. Tf-idf gave better results than word2vec for this model achieving an overall accuracy of 87% (4% higher than our baseline model) and better F1-score across all classes compared to the baseline. The F1-score for the classes *agree* and *disagree* increased to 16% (15% higher than baseline) and 9% (9% higher than baseline). The results in table 1 show that approach 2 consistently improved the results across all classes and reduced the majority class bias that approach 1 was victim to.

Text input quickly grows in dimension with increased vocabulary size, and as inputs in this case is complete news article text, the dimension of the input grows quickly. This can adversely affect the performance of the model. We first tried addressing this issue by experimenting with different input sizes as discussed in section 3.3. However, selecting the top 5 sentences based on cosine similarity with headlines and the initial few lines, weren't the best approach to reduce dimension as these couldn't efficiently capture the complete news article text. We used text summarization as discussed in section 3.4 to reduce the dimensions of the input. This greatly reduced the size of each news article without losing much information that could be crucial in determining the stance. The results show significant improvement over all our previous approaches (row 5 in table 1). With MLP as classifier and word-embedding the model achieved an overall accuracy of 93% (10% higher than our baseline)

and F1-score of minority classes (agree and disagree) increased by 25% and 43% over the baseline respectively.

6 Conclusion and Future Work

As we have already seen the dataset is highly skewed (fig 1) which makes training models to detect minority classes (*agree* and *disagree* in this case) considerably difficult. Our models show promising results that consistently improved performance with more and more sophisticated approaches. Specifically, dimensionality reduction using text summarization shows very good prospects. The model was able to reduce dimension without loosing much information and showed improved performance over previous models. The model was trained only on a limited dataset with about 4500 training instances, with increases training data the performance of this model could further be improved. We keep this as one of the future works for this project, to investigate how more training for this approach can improve the model performance further.

We also experimented with different text preprocessing. Text preprocessing alone improved the accuracy by 10%. The baseline model without preprocessing gives an accuracy of 72% (which is worse than guessing everything belonging to majority class in this case). Most indicative features as determined by our models were, headline vectors, article text vectors and similarity measure (headline and article text similarity). Adding headline and news article sentiments, hedge word counts and different set of sentences (based on either most similar to headline or initial 5 sentences) didn't show any significant improvement over a simpler subset of features.

7 Project Code

The project code can be found at https://github.com/ahaque2/News_Stance_Detection.

References

[1] http://www.fakenewschallenge.org/.

[2] Isabelle Augenstein, Tim Rocktäschel, Andreas Vlachos, and Kalina Bontcheva. Stance detection with bidirectional conditional encoding. *arXiv preprint arXiv:1606.05464*, 2016.

[3] Isabelle Augenstein, Tim Rocktäschel, Andreas Vlachos, and Kalina Bontcheva. Stance detection with bidirectional conditional encoding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 876–885, Austin, Texas, November 2016. Association for Computational Linguistics.

[4] Saif Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. Semeval-2016 task 6: Detecting stance in tweets. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 31–41, 2016.

[5] Benjamin Riedel, Isabelle Augenstein, Georgios P. Spithourakis, and Sebastian Riedel. A simple but tough-to-beat baseline for the Fake News Challenge stance detection task. *CoRR*, abs/1707.03264, 2017.

[6] Sneha Singhania, Nigel Fernandez, and Shrisha Rao. 3han: A deep neural network for fake news detection. In *International Conference on Neural Information Processing*, pages 572–581. Springer, 2017.

[7] Kaiqiang Song, Bingqing Wang, Zhe Feng, Liu Ren, and Fei Liu. Controlling the amount of verbatim copying in abstractive summarization. *arXiv preprint arXiv:1911.10390*, 2019.

[8] Kaiqiang Song, Bingqing Wang, Zhe Feng, Liu Ren, and Fei Liu. Controlling the amount of verbatim copying in abstractive summarization. 11 2019.

[9] Marilyn A Walker, Pranav Anand, Robert Abbott, and Ricky Grant. Stance classification using dialogic properties of persuasion. In *Proceedings of the 2012 conference of the North American chapter of the association for computational linguistics: Human language technologies*, pages 592–596. Association for Computational Linguistics, 2012.