Argument Generation with Retrieval, Planning, and Realization

Xinyu Hua, Zhe Hu, and **Lu Wang**
Khoury College of Computer Sciences
Northeastern University
Boston, MA 02115
{hua.x, hu.zhe}@husky.neu.edu, luwang@ccs.neu.edu

Abstract

Automatic argument generation is an appealing but challenging task. In this paper, we study the specific problem of counter-argument generation, and present a novel framework, CANDELA. It consists of a powerful retrieval system and a novel two-step generation model, where a **text planning decoder** first decides on the main talking points and a proper language style for each sentence, then a **content realization decoder** reflects the decisions and constructs an informative paragraph-level argument. Furthermore, our generation model is empowered by a retrieval system indexed with 12 million articles collected from Wikipedia and popular English news media, which provides access to high-quality content with diversity. Automatic evaluation on a large-scale dataset collected from Reddit shows that our model yields significantly higher BLEU, ROUGE, and METEOR scores than the state-of-the-art and non-trivial comparisons. Human evaluation further indicates that our system arguments are more appropriate for refutation and richer in content.

1 Introduction

Counter-argument generation aims to produce arguments of a different stance, in order to refute the given proposition on a controversial issue (Toulmin, 1958; Damer, 2012). A system that automatically constructs counter-arguments can effectively present alternative perspectives along with associated evidence and reasoning, and thus facilitate a more comprehensive understanding of complicated problems when controversy arises.

Nevertheless, constructing persuasive arguments is a challenging task, as it requires an appropriate combination of credible evidence, rigorous logical reasoning, and sometimes emotional appeal (Walton et al., 2008; Wachsmuth et al., 2017a; Wang et al., 2017). A sample counter-argument

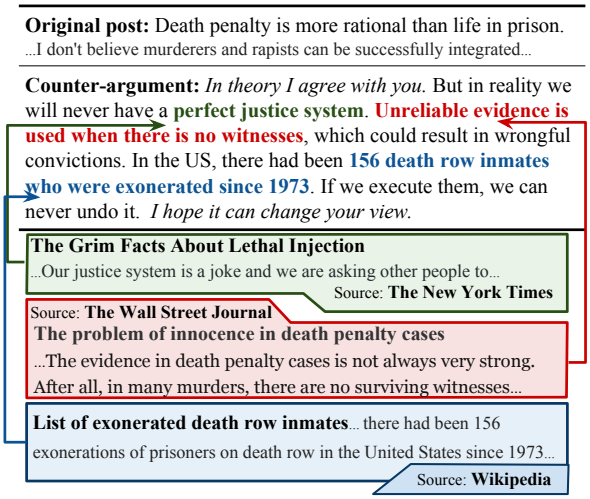

Figure 1: Sample counter-argument for a pro-death penalty statement from Reddit /r/ChangeMyView. The argument consists of a sequence of propositions, by synthesizing opinions and facts from diverse sources. Sentences in italics contain stylistic languages for argumentation purpose.

for a pro-death penalty post is shown in Figure 1. As can be seen, a sequence of talking points on the "imperfect justice system" are presented: it starts with the fundamental concept, then follows up with more specific evaluative claim and supporting fact. Although retrieval-based methods have been investigated to construct counter-arguments (Sato et al., 2015; Reisert et al., 2015), they typically produce a collection of sentences from disparate sources, thus fall short of coherence and conciseness. Moreover, human always deploy *stylistic languages* with specific argumentative functions to promote persuasiveness, such as making a concessive move (e.g., "*In theory I agree with you*"). This further requires the generation system to have better control of the languages style.

Our goal is to design *a counter-argument generation system to address the above challenges and*

produce paragraph-level arguments with rich-yet-coherent content. To this end, we present **CANDELA**—a novel framework to generate Counter-Arguments with two-step Neural Decoders and ExternaL knowledge Augmentation.[1] Concretely, CANDELA has three major distinct features:

First, it is equipped with two decoders: one for **text planning**—selecting talking points to cover for *each sentence* to be generated, the other for **content realization**—producing a fluent argument to reflect decisions made by the text planner. This enables our model to produce longer arguments with richer information.

Furthermore, multiple objectives are designed for our text planning decoder to both handle content selection and ordering, and select a proper argumentative discourse function of a desired language style for each sentence generation.

Lastly, the input to our argument generation model is augmented with keyphrases and passages retrieved from a large-scale search engine, which indexes 12 million articles from Wikipedia and four popular English news media of varying ideological leanings. This ensures access to reliable evidence, high-quality reasoning, and diverse opinions from different sources, as opposed to recent work that mostly considers a single origin, such as Wikipedia (Rinott et al., 2015) or online debate portals (Wachsmuth et al., 2018b).

We experiment with argument and counter-argument pairs collected from the Reddit `/r/ChangeMyView` group. Automatic evaluation shows that the proposed model significantly outperforms our prior argument generation system (Hua and Wang, 2018) and other non-trivial comparisons. Human evaluation further suggests that our model produces more appropriate counter-arguments with richer content than other automatic systems, while maintaining a fluency level comparable to human-constructed arguments.

2 Related Work

To date, the majority of the work on automatic argument generation leads to rule-based models, e.g., designing operators that reflect strategies from argumentation theory (Reed et al., 1996; Carenini and Moore, 2000). Information retrieval systems are recently developed to extract arguments relevant to a given debate motion (Sato et al., 2015). Although content ordering has been investigated (Reisert et al., 2015; Yanase et al., 2015), the output arguments are usually a collection of sentences from heterogeneous information sources, thus lacking coherence and conciseness. Our work aims to close the gap by generating eloquent and coherent arguments, assisted by an argument retrieval system.

Recent progress in sequence-to-sequence (seq2seq) text generation models has delivered both fluent and content rich outputs by explicitly conducting content selection and ordering (Gehrmann et al., 2018; Wiseman et al., 2018), which is a promising avenue for enabling end-to-end counter-argument construction (Le et al., 2018). In particular, our prior work (Hua and Wang, 2018) leverages passages retrieved from Wikipedia to improve the quality of generated arguments, yet Wikipedia itself has the limitation of containing mostly facts. By leveraging Wikipedia and popular news media, our proposed pipeline can enrich the factual evidence with high-quality opinions and reasoning.

Our work is also in line with argument retrieval research, where prior effort mostly considers single-origin information source (Rinott et al., 2015; Levy et al., 2018; Wachsmuth et al., 2017b, 2018b). Recent work by Stab et al. (2018) indexes all web documents collected in Common Crawl, which inevitably incorporates noisy, low-quality content. Besides, existing work treats individual sentences as arguments, disregarding their crucial discourse structures and logical relations with adjacent sentences. Instead, we use multiple high-quality information sources, and construct paragraph-level passages to retain the context of arguments.

3 Overview of CANDELA

Our counter-argument generation framework, as shown in Figure 2, has two main components: **argument retrieval** model (§ 4) that takes the input statement and a search engine, and outputs relevant passages and keyphrases, which are used as input for our **argument generation** model (§ 5) to produce a fluent and informative argument.

Concretely, the argument retrieval component retrieves a set of candidate passages from Wikipedia and news media (§ 4.1), then further selects passages according to their stances towards

[1] Code and data are available at `https://xinyuhua.github.io/Resources/acl19/`.

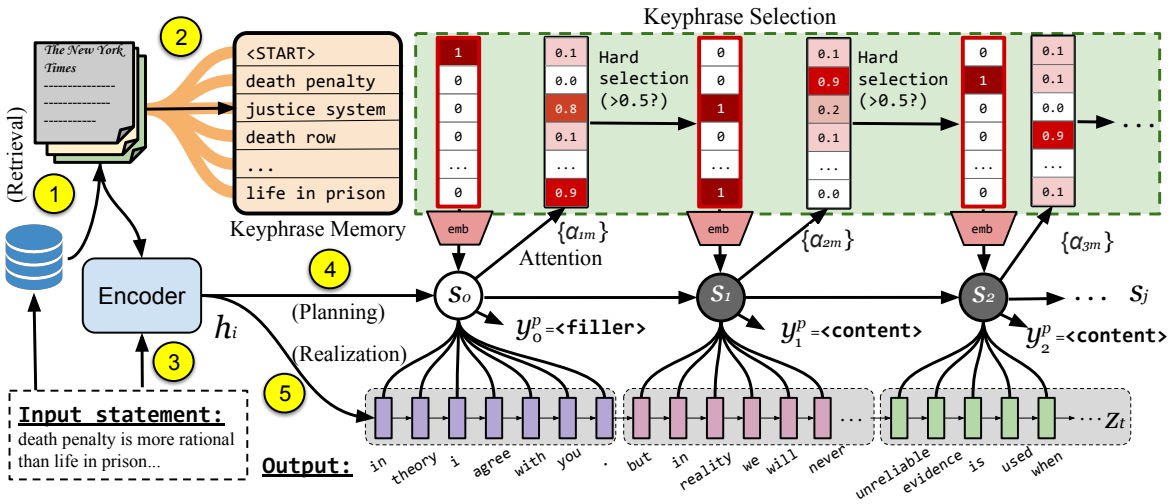

Figure 2: Architecture of CANDELA. ① Argument retrieval (§ 4): a set of passages are retrieved and ranked based on relevance and stance (§ 4.1, 4.3), from which ② a set of keyphrases are extracted (§ 4.2), with both as input for argument generation. ③ The biLSTM encoder consumes the input statement and passages returned from step 1. ④ A text planning decoder outputs a representation per sentence, and simultaneously predicts an argumentative function and selects keyphrases to include for the next sentence to be generated (§ 5.2). ⑤ A content realization decoder produces the counter-argument (§ 5.3).

the input statement (§ 4.3). A **keyphrase extraction** module distills the refined passages into a set of talking points, which comprise the keyphrase memory as additional input for generation (§ 4.2).

The argument generation component first runs the **text planning decoder** (§ 5.2) to produce a sequence of hidden states, each corresponding to a *sentence-level representation* that encodes the selection of keyphrases to cover, as well as the predicted argumentative function for a desired language style. The **content realization decoder** (§ 5.3) then generates the argument conditioned on the sentence representations.

4 Argument Retrieval

4.1 Information Sources and Indexing

We aim to build a search engine from diverse information sources with factual evidence and varied opinions of high quality. To achieve that, we use Common Crawl[2] to collect a large-scale online news dataset covering four major English news media: `The New York Times` (NYT), `The Washington Post` (WaPo), `Reuters`, and `The Wall Street Journal` (WSJ). HTML files are processed using the open-source tool jusText (Pomikálek, 2011) to extract article content. We deduplicate articles and remove the ones with less than 50 words. We also download a Wikipedia dump. About 12 million articles are processed in total, with basic statistics shown in Table 1.

Source	# Articles	# Passages	Date Range
Wikipedia	5,743,901	42,797,543	dump of 12/2016
WaPo	1,109,672	22,564,532	01/1997 - 10/2018
NYT	1,952,446	28,904,549	09/1895 - 09/2018
Reuters	1,052,592	9,913,400	06/2005 - 09/2018
WSJ	2,059,128	16,109,392	01/1996 - 09/2018
Total	11,917,739	120,289,416	-

Table 1: Statistics on information sources for argument retrieval. News media are sorted by ideological leanings from left to right, according to `https://www.adfontesmedia.com/`.

We segment articles into passages with a sliding window of three sentences, with a step size of two. We further constraint the passages to have at least 50 words. For shorter passages, we keep adding subsequent sentences until reaching the length limit. Per Table 1, 120 million passages are preserved and indexed with Elasticsearch (Gormley and Tong, 2015) as done in Stab et al. (2018).

Query Formulation. For an input statement with multiple sentences, one query is constructed per sentence, if it has more than 5 content words (10 for questions), and at least 3 are distinct. For each query, the top 20 passages ranked by BM25 (Robertson et al., 1995) are retained, per medium. All passages retrieved for the input statement are merged and deduplicated, and they will

[2] `http://commoncrawl.org/`

be ranked as discussed in § 4.3.

4.2 Keyphrase Extraction

Here we describe a keyphrase extraction procedure for both input statements and retrieved passages, which will be utilized for passage ranking as detailed in the next section.

For *input statement*, our goal is to identify a set of phrases representing the issues under discussion, such as "death penalty" in Figure 1. We thus first extract the topic signature words (Lin and Hovy, 2000) for input representation, and expand them into phrases that better capture semantic meanings.

Concretely, topic signature words of an input statement are calculated against all input statements in our training set with log-likelihood ratio test. In order to cover phrases with related terms, we further expand this set with their synonyms, hyponyms, hypernyms, and antonyms based on WordNet (Miller, 1994). The statements are first parsed with Stanford part-of-speech tagger (Manning et al., 2014). Then regular expressions are applied to extract candidate noun phrases and verb phrases (details in Appendix A.1). A keyphrase is selected if it contains: (1) at least one content word, (2) no more than 10 tokens, and (3) at least one topic signature word or a Wikipedia article title.

For *retrieved passages*, their keyphrases are extracted using the same procedure as above, except that the input statement's topic signature words are used as references again.

4.3 Passage Ranking and Filtering

We merge the retrieved passages from all media and rank them based on the number of words in overlapping keyphrases with the input statement. To break a tie, with the input as the reference, we further consider the number of its topic signature words that are covered by the passage, then the coverage of non-stopword bigrams and unigrams. In order to encourage diversity, we discard a passage if more than 50% of its content words are already included by a higher ranked passage. In the final step, we filter out passages if they have the same stance as the input statement for given topics. We determine the stances of passages by adopting the **stance scoring model** proposed by Bar-Haim et al. (2017). More details can be found in Appendix A.2.

5 Argument Generation

5.1 Task Formulation

Given an input statement $\boldsymbol{X} = \{x_i\}$, a set of passages, and a keyphrase memory $\mathcal{M}$, our goal is to generate a counter-argument $\boldsymbol{Y} = \{y_t\}$ of a different stance as $\boldsymbol{X}$, x_i and y_t are tokens at timestamps i and t. Built upon the sequence-to-sequence (seq2seq) framework with input attention (Sutskever et al., 2014; Bahdanau et al., 2015), the input statement and the passages selected in § 4 are encoded by a bidirectional LSTM (biLSTM) encoder into a sequence of hidden states $\boldsymbol{h}_i$. The last hidden state of the encoder is used as the first hidden state of both text planning decoder and content realization decoder.

As depicted in Figure 2, the counter-argument is generated as follows. A text planning decoder (§ 5.2) first calculates a sequence of sentence representations $\boldsymbol{s}_j$ (for the j-th sentence) by encoding the keyphrases selected from the previous timestamp $j-1$. During this step, an *argumentative function* label is predicted to indicate a desired language style for each sentence, and a subset of the keyphrases are selected from $\mathcal{M}$ (*content selection*) for the next sentence. In the second step, a content realization decoder (§ 5.3) generates the final counter-argument conditioned on previously generated tokens and the corresponding sentence representation $\boldsymbol{s}_j$.

5.2 Text Planning Decoder

Text planning is an important component for natural language generation systems to decide on content structure for the target generation (Lavoie and Rambow, 1997; Reiter and Dale, 2000). We propose a text planner with two objectives: selecting talking points from the keyphrase memory $\mathcal{M}$, and choosing a proper argumentative function per sentence. Concretely, we train a sentence-level LSTM that learns to generate a sequence of sentence representations $\{\boldsymbol{s}_j\}$ given the selected keyphrase set $\mathbb{C}(j)$ as input for the j-th sentence:

$$\boldsymbol{s}_j = f(\boldsymbol{s}_{j-1}, \sum_{e_k \in \mathbb{C}(j)} \boldsymbol{e}_k) \quad (1)$$

where f is an LSTM network, $\boldsymbol{e}_k$ is the embedding for a selected phrase, represented by summing up all its words' Glove embeddings (Pennington et al., 2014) in our experiments.

Content Selection $\mathbb{C}(j)$. We propose an attention mechanism to conduct content selection and yield

$\mathbb{C}(j)$ from the representation of the previous sentence $\boldsymbol{s}_{j-1}$ to encourage topical coherence. To allow the selection of multiple keyphrases, we use the sigmoid function to calculate the score:

$$\alpha_{jm} = \text{sigmoid}(\boldsymbol{e}_m \boldsymbol{W}^{pa} \boldsymbol{s}_{j-1}) \tag{2}$$

where $\boldsymbol{W}^{pa}$ are trainable parameters, keyphrases with $\alpha_{jm} > 0.5$ are included in $\mathbb{C}(j)$, and the keyphrase with top attention value is always selected. We further prohibit a keyphrase from being chosen for more than once in multiple sentences. For the first sentence $\boldsymbol{s}_0$, $\mathbb{C}(0)$ only contains <start>, whose embedding is randomly initialized. During training, the *true labels* of $\mathbb{C}(j)$ are constructed as follows: a keyphrase in $\mathcal{M}$ is selected for the j-th gold-standard argument sentence if they overlap with any content word.

Argumentative Function Prediction y_j^p. As shown in Figure 1, humans often deploy stylistic languages to achieve better persuasiveness, e.g. agreement as a concessive move. We aim to inform the realization decoder about the choice of style, and thus distinguish between two types of argumentative functions: **argumentative content sentence** which delivers the critical ideas, e.g. "*unreliable evidence is used when there is no witness*", and **argumentative filler sentence** which contains stylistic languages or general statements (e.g., "*you can't bring dead people back to life*").

Since we do not have argumentative function labels, during training, we use the following rules to automatically label each sentence as *content sentence* if it has at least 10 words (20 for questions) and satisfy the following conditions: (1) it has at least two topic signature words of the input statement or a gold-standard counter-argument[3], or (2) at least one topic signature word with a discourse marker at the beginning of the sentence. If the first three words in a *content sentence* contain a pronoun, the previous sentence is labeled as such too. Discourse markers are selected from PDTB discourse connectives (e.g., *as a result*, *eventually*, or *in contrast*). The full list is included in Appendix A.3. All other sentences become *filler sentences*. In the future work, we will consider utilizing learning-based methods, e.g., Hidey et al. (2017), to predict richer argumentative functions.

[3]When calculating topic signatures for gold-standard arguments, all replies in the training set are used as background.

The argumentative function label y_j^p for the j-th sentence is calculated as follows:

$$P(y_j^p | y_{<j}^p, \boldsymbol{X}) = \text{softmax}(\boldsymbol{w}_p^T (\tanh(\boldsymbol{W}^{po}[\boldsymbol{c}_j; \boldsymbol{s}_j])) + \boldsymbol{b}_p) \tag{3}$$

$$\boldsymbol{c}_j = \sum_{e_m \in \mathcal{M}} \alpha_{jm} \boldsymbol{e}_m \tag{4}$$

where α_{jm} is the alignment score computed as in Eq. 2, $\boldsymbol{c}_j$ is the attention weighted context vector, $\boldsymbol{w}_p$, $\boldsymbol{W}^{po}$, and $\boldsymbol{b}_p$ are trainable parameters.

5.3 Content Realization Decoder

The content realization decoder generates the counter-argument word by word, with another LSTM network f^w. We denote the sentence id of the t-th word in the argument as $J(t)$, then the sentence representation $\boldsymbol{s}_{J(t)}$ from the text planning decoder, together with the embedding of the previous generated token $\boldsymbol{y}_{t-1}$, are fed as input to calculate the hidden state $\boldsymbol{z}_t$:

$$\boldsymbol{z}_t = f^w(\boldsymbol{z}_{t-1}, \tanh(\boldsymbol{W}^{wp} \boldsymbol{s}_{J(t)} + \boldsymbol{W}^{ww} \boldsymbol{y}_{t-1} + \boldsymbol{b}^w)) \tag{5}$$

The conditional probability of the next token y_t is then computed over a standard softmax, with an attention mechanism applied on the encoder hidden states $\boldsymbol{h}_i$ to obtain the context vector $\boldsymbol{c}_t^w$:

$$P(y_t | y_{<t}, \boldsymbol{X}, \boldsymbol{s}_{J(t)}) = \text{softmax}(\boldsymbol{w}_w^T (\tanh(\boldsymbol{W}^{wo}[\boldsymbol{c}_t^w; \boldsymbol{z}_t])) + \boldsymbol{b}^o) \tag{6}$$

$$\boldsymbol{c}_t^w = \sum_{i=1}^{|\boldsymbol{X}|} \beta_{ti} \boldsymbol{h}_i \tag{7}$$

$$\beta_{ti} = \text{softmax}(\boldsymbol{h}_i \boldsymbol{W}^{wa} \boldsymbol{z}_t) \tag{8}$$

where β_{ti} is the input attention, $\boldsymbol{W}^{wp}$, $\boldsymbol{W}^{ww}$, $\boldsymbol{W}^{wo}$, $\boldsymbol{W}^{wa}$, $\boldsymbol{b}^o$, $\boldsymbol{w}_w$, and $\boldsymbol{b}^w$ are learnable.

Reranking-based Beam Search. Our content realization decoder utilizes beam search enhanced with a reranking mechanism, where we sort the beams at the end of each sentence by the number of selected keyphrases that are generated. We also discard beams with n-gram repetition for $n \geq 4$.

5.4 Training Objective

Given all model parameters θ, our mixed objective considers the target argument ($\mathcal{L}_{\text{arg}}(\theta)$), the argumentative function type ($\mathcal{L}_{\text{func}}(\theta)$), and the next sentence keyphrase selection ($\mathcal{L}_{\text{sel}}(\theta)$):

$$\mathcal{L}(\theta) = \mathcal{L}_{\text{arg}}(\theta) + \gamma \cdot \mathcal{L}_{\text{func}}(\theta) + \eta \cdot \mathcal{L}_{\text{sel}}(\theta) \quad (9)$$

$$\mathcal{L}_{\text{arg}}(\theta) = -\sum_{(\boldsymbol{X},\boldsymbol{Y})\in D} \log P(\boldsymbol{Y}|\boldsymbol{X};\theta) \quad (10)$$

$$\mathcal{L}_{\text{func}}(\theta) = -\sum_{(\boldsymbol{X},\boldsymbol{Y}^p)} \log P(\boldsymbol{Y}^p|\boldsymbol{X};\theta) \quad (11)$$

$$\mathcal{L}_{\text{sel}}(\theta) = -\sum_{\boldsymbol{Y}^p}\sum_{j=1}^{|\boldsymbol{Y}^p|}\Big(\sum_{e_m\in\mathbb{C}(j)}\log(\alpha_{jm}) + \sum_{e_m\notin\mathbb{C}(j)}\log(1-\alpha_{jm})\Big) \quad (12)$$

where D is the training corpus, $(\boldsymbol{X}, \boldsymbol{Y})$ are input statement and counter-argument pairs, and $\boldsymbol{Y}^p$ are the sentence function labels. α_{jm} are keyphrase selection labels as computed in Eq. 2. For simplicity, we set γ and η as 1.0 in our experiments, while they can be further tuned as hyper-parameters.

6 Experimental Setups

6.1 Data Collection and Preprocessing

We use the same methodology as in our prior work (Hua and Wang, 2018) to collect an argument generation dataset from Reddit `/r/ChangeMyView`.[4] To construct input statement and counter-argument pairs, we treat the original poster (OP) of each thread as the input. We then consider the high quality root replies, defined as the ones awarded with Δs or with more upvotes than downvotes (i.e., `karma` > 0). It is observed that each paragraph often makes a coherent argument. Therefore, these replies are broken down into paragraphs, and a paragraph is retained as a target argument to the OP if it has more than 10 words and at least one argumentative content sentence.

We then identify threads in the domains of politics and policy, and remove posts with offensive languages. Most recent threads are used as test set. As a result, we have $11,356$ threads or OPs ($217,057$ arguments) for training, $1,774$ ($33,318$ arguments) for validation, and $1,703$ ($36,777$ arguments) for test. They are split into sentences and then tokenized by the Stanford CoreNLP toolkit (Manning et al., 2014).

Training Data Construction for Passages and Keyphrase Memory. Since no gold-standard annotation is available for the input passages and keyphrases, we acquire training labels by constructing queries from the gold-standard arguments as described in § 4.1, and reranking retrieved passages based on the following criteria in order: (1) coverage of topic signature words in the input statement; (2) a weighted summation of the coverage of n-grams in the argument[5]; (3) the magnitude of stance score, where we keep the passages of the same polarity as the argument; (4) content word overlap with the argument; and (5) coverage of topic signature words in the argument.

6.2 System and Oracle Retrieved Passages

For evaluation, we employ both *system* retrieved passages (i.e., constructing queries from OP) and KM (§ 4), and *oracle* retrieved passages (i.e., constructing queries from target argument) and KM as described in training data construction. Statistics on the final dataset are listed in Table 2.

	Training	*System*	*Oracle*
Avg. # words per OP	383.7	373.0	373.0
Avg. # words per argument	66.0	65.1	65.1
Avg. # passage	4.3	9.6	4.2
Avg. # keyphrase	57.1	128.6	56.6

Table 2: Statistics on the datasets for experiments.

6.3 Comparisons

In addition to a **Retrieval** model, where the top ranked passage is used as counter-argument, we further consider four systems for comparison. (1) A standard **Seq2seq** model with attention, where we feed the OP as input and train the model to generate counter-arguments. Regular beam search with the same beam size as our model is used for decoding. (2) A **Seq2seqAug** model with additional input of the keyphrase memory and ranked passages, both concatenated with OP to serve as the encoder input. The reranking-based decoder in our model is also implemented for SEQ2SEQAUG to enhance the coverage of input keyphrases. (3) An ablated SEQ2SEQAUG model where the passages are removed from the input. (4) We also reimplement the argument generation model in our prior work (Hua and Wang, 2018) (**H&W**) with PyTorch (Paszke et al., 2017), which is used for CANDELA implementation. H&W takes as input the OP and ranked passages, and then uses two

[4] We further crawled $42,649$ threads from July 2017 to December 2018, compared to the previously collected dataset.

[5] We choose $0.5, 0.3, 0.2$ as weights for 4-grams, trigrams, and bigrams, respectively.

separate decoders to first generate all keyphrases and then the counter-argument. For our model, we also implement a variant where the input only contains the OP and the keyphrase memory.

6.4 Training Details

For all models, we use a two-layer LSTM for all encoders and decoders with a dropout probability of 0.2 between layers (Gal and Ghahramani, 2016). All layers have 512-dimensional hidden states. We limit the input statement to 500 tokens, the ranked passages to 400 tokens, and the target counter-argument to 120 tokens. Our vocabulary has $50K$ words for both input and output, with 300-dimensional word embeddings initialized with GloVe (Pennington et al., 2014) and fine-tuned during model training. We use AdaGrad (Duchi et al., 2011) with a learning rate of 0.15 and an initial accumulator of 0.1 as the optimizer, with the gradient norm clipped to 2.0. Early stopping is implemented according to the perplexity on validation set. For all our models the training takes approximately 30 hours (40 epochs) on a Quadro P5000 GPU card, with a batch size of 64. For beam search, we use a beam size of 5, tuned from $\{5, 10, 15\}$ on validation.

We also pre-train a biLSTM for encoder based on all OPs from the training set, and an LSTM for content realization decoder based on two sources of data: 353K counter-arguments that are high quality root reply paragraphs extended with posts of non-negative karma, and 2.4 million retrieved passages randomly sampled from the training set. Both are trained as done in Bengio et al. (2003). We then use the first layer's parameters to initialize all models, including our comparisons.

7 Results and Analysis

7.1 Automatic Evaluation

We employ ROUGE (Lin, 2004), a recall-oriented metric, BLEU (Papineni et al., 2002), based on n-gram precision, and METEOR (Denkowski and Lavie, 2014), measuring unigram precision and recall by considering synonyms, paraphrases, and stemming. BLEU-2, BLEU-4, ROUGE-2 recall, and METEOR are reported in Table 3 for both setups.

Under system setup, our model CANDELA statistically significantly outperforms all comparisons and the retrieval model in all metrics, based on a randomization test (Noreen, 1989) ($p <$ 0.0005). Furthermore, our model generates longer sentences whose lengths are comparable with human arguments, both with about 22 words per sentence. This also results in longer arguments. Under oracle setup, all models are notably improved due to the higher quality of reranked passages, and our model achieves statistically significantly better BLEU scores. Interestingly, we observe a decrease of ROUGE and METEOR, but a marginal increase of BLEU-2 by removing passages from our model input. This could be because the passages introduce divergent content, albeit probably on-topic, that cannot be captured by BLEU.

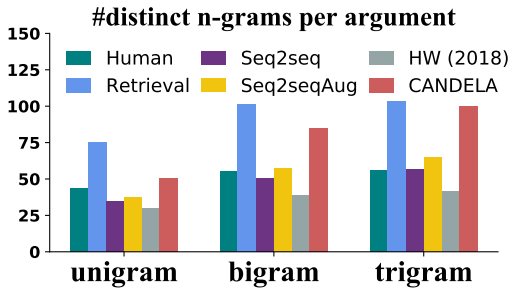

Figure 3: Average number of distinct n-grams per argument.

	K			
	100	500	1000	2000
HUMAN	44.1	25.8	18.5	12.0
RETRIEVAL	50.6	33.3	26.0	18.6
SEQ2SEQ	25.0	7.5	3.2	1.2
SEQ2SEQAUG	28.2	9.2	4.6	1.8
H&W (2018)	38.6	24.0	19.5	16.2
CANDELA	30.0	10.5	5.3	2.3

Figure 4: Percentage of words in arguments that are not in the top-K ($K = 100, 500, 1000, 2000$) frequent words seen in training. Darker color indicates higher portion of uncommon words found in the arguments.

Content Diversity. We further measure whether our model is able to generate diverse content. First, borrowing the diversity measurement from dialogue generation research (Li et al., 2016), we report the average number of distinct n-grams per argument under system setup in Figure 3. Our system generates more unique unigrams and bigrams than other automatic systems, underscoring its capability of generating diverse content. Our model also maintains a comparable type-token ratio (TTR) compared to systems that generate shorter arguments, e.g., a 0.79 for bigram TTR of our model versus 0.83 and 0.84 for SEQ2SEQAUG and SEQ2SEQ. RETRIEVAL, con-

	w/ System Retrieval						*w/ Oracle Retrieval*					
	B-2	**B-4**	**R-2**	**MTR**	**#Word**	**#Sent**	**B-2**	**B-4**	**R-2**	**MTR**	**#Word**	**#Sent**
HUMAN	-	-	-	-	66	22	-	-	-	-	66	22
RETRIEVAL	7.55	1.11	8.64	14.38	123	23	10.97	3.05	23.49	20.08	140	21
Comparisons												
SEQ2SEQ	6.92	2.13	13.02	15.08	68	15	6.92	2.13	13.02	15.08	68	15
SEQ2SEQAUG	8.26	2.24	13.79	15.75	78	14	10.98	4.41	22.97	19.62	71	14
w/o psg	7.94	2.28	10.13	15.71	75	12	9.89	3.34	14.20	18.40	66	12
H&W (2018)	3.64	0.92	8.83	11.78	51	12	8.51	2.86	18.89	17.18	58	12
Our Models												
CANDELA	12.02*	**2.99***	**14.93***	**16.92***	119	22	15.80*	**5.00***	**23.75**	**20.18**	116	22
w/o psg	**12.33***	2.86*	14.53*	16.60*	123	23	**16.33***	4.98*	23.65	19.94	123	23

Table 3: Main results on argument generation. We report BLEU-2 (B-2), BLEU-4 (B-4), ROUGE-2 (R-2) recall, METEOR (MTR), and average number of words per argument and per sentence. Best scores are in bold. $*$: statistically significantly better than all comparisons (randomization approximation test (Noreen, 1989), $p < 0.0005$). Input is the same for SEQ2SEQ for both system and oracle setups.

taining top ranked passages of human-edited content, produces the most distinct words.

Next, we compare how each system generates content beyond the common words. As shown in Figure 4, human-edited text, including gold-standard arguments (HUMAN) and retrieved passages, tends to have higher usage of uncommon words than automatic systems, suggesting the gap between human vs. system arguments. Among the four automatic systems, our prior model (Hua and Wang, 2018) generates a significantly higher portion of uncommon words, yet further inspection shows that the output often includes more off-topic information.

7.2 Human Evaluation

Human judges are asked to rate arguments on a Likert scale of 1 (worst) to 5 (best) on the following three aspects: **grammaticality**—denotes language fluency; **appropriateness**—indicates if the output is on-topic and on the opposing stance; **content richness**—measures the amount of distinct talking points. In order to promote consistency of annotation, we provide descriptions and sample arguments for each scale. For example, an appropriateness score of 3 means the counter-argument contains relevant words and is likely to be on a different stance. The judges are then asked to rank all arguments for the same input based on their overall quality.

We randomly sampled 43 threads from the test set, and hired three native or proficient English speakers to evaluate arguments generated by SEQ2SEQAUG, our prior argument generation model (H&W), and the new model CANDELA, along with gold-standard HUMAN arguments and the top passage by RETRIEVAL.

	Gram.	**Appr.**	**Cont.**	**Top-1**	**Top-2**
HUMAN	4.95	4.23	4.39	75.8%	85.8%
RETRIEVAL	4.85	3.04	3.68	17.5%	55.8%
SEQ2SEQAUG	**4.83**	2.67	2.47	1.7%	22.5%
H&W (2018)	3.86	2.27	2.10	1.7%	7.5%
CANDELA	4.59	**2.97**	**2.93***	3.3%	28.3%

Table 4: Human evaluation on grammaticality (Gram), appropriateness (Appr), and content richness (Cont.), on a scale of 1 to 5 (best). The best result among automatic systems is highlighted in bold, with statistical significance marked with $*$ (approximation randomization test, $p < 0.0005$). The highest standard deviation among all is 1.0. Top-1/2: % of evaluations a system being ranked in top 1 or 2 for overall quality.

Results. The first 3 examples are used only for calibration, and the remaining 40 are used to report results in Table 4. Inter-annotator agreement scores (Krippendorff's α) of 0.44, 0.58, 0.49 are achieved for the three aspects, implying general consensus to intermediate agreement.

Our system obtains the highest appropriateness and content richness among all automatic systems. This confirms the previous observation that our model produces more informative argument than other neural models. SEQ2SEQAUG has a marginally better grammaticality score, likely due to the fact that our arguments are longer, and tend to contain less fluent generation towards the end.

Furthermore, we see that human arguments are

ranked as the best in about 76% of the evaluation, followed by RETRIEVAL. Our model is more likely to be ranked top than any other automatic models. Especially, our model is rated better than either HUMAN or RETRIEVAL, i.e., human-edited text, in 39.2% of the evaluations, compared to 34.2% for SEQ2SEQAUG and 13.3% for our prior model.

7.3 Sample Arguments and Discussions

We show sample outputs of different systems alongside human constructed counter-argument in Figure 5. As can be seen, our system arguments cover many relevant talking points, including the idea of "*taking care of children*" that is also used by human. It further illustrates the effectiveness of our retrieval system and the usage of keyphrase selection as part of text planning to guide argument generation. Moreover, we also observe that our model generation is able to switch between argumentative content sentence and filler sentence, though better control is needed to improve coherence. Meanwhile, SEQ2SEQ frequently echos words from OP, and both SEQ2SEQ and SEQ2SEQAUG suffer from the problems of "hallucination" (e.g., the first sentence in SEQ2SEQAUG) and repetition (e.g., the second and third sentences in SEQ2SEQ).

Nonetheless, there is a huge space for improvement. First, our model tends to overuse negation, such as *"this is not to say that it is unfair..."*. It is likely due to its overfitting on specific stylistic languages, e.g., negation is often observed for refutation in debates (Wang et al., 2017). Second, human arguments have significantly better organization and often deploy complicated argumentation strategies (Wachsmuth et al., 2018a), which so far is not well captured by any automatic system. Both points inspire future work on (1) controlling of the language styles and corresponding content, and (2) mining argumentation structures for use in guiding generation with better planning.

OP: The wage gap isn't about gender. (...) So one factor definitely the fact that there are some government jobs that are seriously dominated by women and happen to pay really low (...)

Human: Children are the first factor. I work in a traditionally male-dominated field, and questions are asked to determine familial status, even though those are illegal. Employers know single moms have to stay home if their child is sick. They know single moms can only work during daycare hours. They know single moms are unavailable for weekend and late night work. They know single moms cannot travel. The workplace is very family-unfriendly.

Seq2seq: the problem with this is that there are a lot of people who do n't want to hire women . there are plenty of women who do n't have the money to pay for it . the problem is that women do n't need to pay for their work . they do n't have to worry about it . it 's a matter of money .

Seq2seqAug: men and women are not 39.5 % of the pay gap . the problem is not that women are more likely to be victims of sexism , but rather that they are more natural good-looking/attractive action . this is not the case .

CANDELA: the problem with this argument is that the **wage gap** does not have to do with the **gender pay gap** . it is a fact that women are more likely to be able to **take care of their children** than their **male counterparts** . this is not to say that it is unfair to assume that women are being paid less than men , but that does not mean that it is not the case that **women are discriminated against** . *it is not a matter of the wage gap , it is a matter of opinion* . it is the job of the employer to make sure that the job is not the same as the other

Keyphrase Memory: **wage gap**; discrimination; **gender pay gaps**; **raise the child**; **male colleagues**; paid maternity leave; **underlying gender discrimination** ...

Figure 5: Sample arguments generated by different systems along with a sample human argument. For our model CANDELA, additionally shown are the keyphrase memory with selected phrases in color, and argumentative filler sentence in italics.

8 Conclusion

We present a novel counter-argument generation framework, CANDELA. Given an input statement, it first retrieves arguments of different perspectives from millions of high-quality articles collected from diverse sources. An argument generation component then employs a text planning decoder to conduct content selection and specify a suitable language style at sentence-level, followed by a content realization decoder to produce the final argument. Automatic evaluation and human evaluation indicate that our model generates more proper arguments with richer content than non-trivial comparisons, with comparable fluency to human-edited content.

Acknowledgements

This research is supported in part by National Science Foundation through Grants IIS-1566382 and IIS-1813341. We thank Varun Raval for helping with data processing and search engine indexing. We are grateful to the three anonymous reviewers for their constructive suggestions.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Roy Bar-Haim, Indrajit Bhattacharya, Francesco Dinuzzo, Amrita Saha, and Noam Slonim. 2017. Stance classification of context-dependent claims. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 251–261. Association for Computational Linguistics.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155.

Giuseppe Carenini and Johanna Moore. 2000. A strategy for generating evaluative arguments. In *INLG'2000 Proceedings of the First International Conference on Natural Language Generation*, pages 47–54, Mitzpe Ramon, Israel. Association for Computational Linguistics.

T Edward Damer. 2012. *Attacking faulty reasoning*. Cengage Learning.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 376–380, Baltimore, Maryland, USA. Association for Computational Linguistics.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159.

Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 1019–1027. Curran Associates, Inc.

Sebastian Gehrmann, Yuntian Deng, and Alexander Rush. 2018. Bottom-up abstractive summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4098–4109, Brussels, Belgium. Association for Computational Linguistics.

Clinton Gormley and Zachary Tong. 2015. *Elasticsearch: The definitive guide: A distributed real-time search and analytics engine*. " O'Reilly Media, Inc.".

Christopher Hidey, Elena Musi, Alyssa Hwang, Smaranda Muresan, and Kathy McKeown. 2017. Analyzing the semantic types of claims and premises in an online persuasive forum. In *Proceedings of the 4th Workshop on Argument Mining*, pages 11–21. Association for Computational Linguistics.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177. ACM.

Xinyu Hua and Lu Wang. 2018. Neural argument generation augmented with externally retrieved evidence. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 219–230. Association for Computational Linguistics.

Benoit Lavoie and Owen Rambow. 1997. A fast and portable realizer for text generation systems. In *Fifth Conference on Applied Natural Language Processing*.

Dieu-Thu Le, Cam Tu Nguyen, and Kim Anh Nguyen. 2018. Dave the debater: a retrieval-based and generative argumentative dialogue agent. In *Proceedings of the 5th Workshop on Argument Mining*, pages 121–130. Association for Computational Linguistics.

Ran Levy, Ben Bogin, Shai Gretz, Ranit Aharonov, and Noam Slonim. 2018. Towards an argumentative content search engine using weak supervision. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2066–2081. Association for Computational Linguistics.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, San Diego, California. Association for Computational Linguistics.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*.

Chin-Yew Lin and Eduard Hovy. 2000. The automated acquisition of topic signatures for text summarization. In *COLING 2000 Volume 1: The 18th International Conference on Computational Linguistics*.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland. Association for Computational Linguistics.

George A. Miller. 1994. Wordnet: A lexical database for english. In *HUMAN LANGUAGE TECHNOLOGY: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

Eric W Noreen. 1989. *Computer-intensive methods for testing hypotheses*. Wiley New York.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Soumith Chintala, and Gregory Chanan. 2017. Pytorch: Tensors and dynamic neural networks in python with strong gpu acceleration. *PyTorch: Tensors and dynamic neural networks in Python with strong GPU acceleration*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Jan Pomikálek. 2011. *Removing boilerplate and duplicate content from web corpora*. Ph.D. thesis, Masaryk university, Faculty of informatics, Brno, Czech Republic.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind K Joshi, and Bonnie L Webber. 2008. The penn discourse treebank 2.0. In *LREC*. Citeseer.

Chris Reed, Derek Long, and Maria Fox. 1996. An architecture for argumentative dialogue planning. In *International Conference on Formal and Applied Practical Reasoning*, pages 555–566. Springer.

Paul Reisert, Naoya Inoue, Naoaki Okazaki, and Kentaro Inui. 2015. A computational approach for generating toulmin model argumentation. In *Proceedings of the 2nd Workshop on Argumentation Mining*, pages 45–55, Denver, CO. Association for Computational Linguistics.

Ehud Reiter and Robert Dale. 2000. *Building natural language generation systems*. Cambridge university press.

Ruty Rinott, Lena Dankin, Carlos Alzate Perez, Mitesh M. Khapra, Ehud Aharoni, and Noam Slonim. 2015. Show me your evidence - an automatic method for context dependent evidence detection. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 440–450, Lisbon, Portugal. Association for Computational Linguistics.

Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. 1995. Okapi at trec-3. *Nist Special Publication Sp*, 109:109.

Misa Sato, Kohsuke Yanai, Toshinori Miyoshi, Toshihiko Yanase, Makoto Iwayama, Qinghua Sun, and Yoshiki Niwa. 2015. End-to-end argument generation system in debating. In *Proceedings of ACL-IJCNLP 2015 System Demonstrations*, pages 109–114, Beijing, China. Association for Computational Linguistics and The Asian Federation of Natural Language Processing.

Christian Stab, Johannes Daxenberger, Chris Stahlhut, Tristan Miller, Benjamin Schiller, Christopher Tauchmann, Steffen Eger, and Iryna Gurevych. 2018. Argumentext: Searching for arguments in heterogeneous sources. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 21–25. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3104–3112. Curran Associates, Inc.

Stephen Edelston Toulmin. 1958. *The use of argument*. Cambridge University Press.

Henning Wachsmuth, Nona Naderi, Ivan Habernal, Yufang Hou, Graeme Hirst, Iryna Gurevych, and Benno Stein. 2017a. Argumentation quality assessment: Theory vs. practice. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 250–255. Association for Computational Linguistics.

Henning Wachsmuth, Martin Potthast, Khalid Al Khatib, Yamen Ajjour, Jana Puschmann, Jiani Qu, Jonas Dorsch, Viorel Morari, Janek Bevendorff, and Benno Stein. 2017b. Building an argument search engine for the web. In *Proceedings of the 4th Workshop on Argument Mining*, pages 49–59. Association for Computational Linguistics.

Henning Wachsmuth, Manfred Stede, Roxanne El Baff, Khalid Al Khatib, Maria Skeppstedt, and Benno Stein. 2018a. Argumentation synthesis following rhetorical strategies. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3753–3765. Association for Computational Linguistics.

Henning Wachsmuth, Shahbaz Syed, and Benno Stein. 2018b. Retrieval of the best counterargument without prior topic knowledge. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 241–251. Association for Computational Linguistics.

Douglas Walton, Christopher Reed, and Fabrizio Macagno. 2008. *Argumentation schemes*. Cambridge University Press.

Lu Wang, Nick Beauchamp, Sarah Shugars, and Kechen Qin. 2017. Winning on the merits: The joint effects of content and style on debate outcomes. *Transactions of the Association for Computational Linguistics*, 5:219–232.

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2018. Learning neural templates for text generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3174–3187, Brussels, Belgium. Association for Computational Linguistics.

Toshihiko Yanase, Toshinori Miyoshi, Kohsuke Yanai, Misa Sato, Makoto Iwayama, Yoshiki Niwa, Paul Reisert, and Kentaro Inui. 2015. Learning sentence ordering for opinion generation of debate. In *Proceedings of the 2nd Workshop on Argumentation Mining*, pages 94–103, Denver, CO. Association for Computational Linguistics.

A Appendices

A.1 Chunking Grammar for Keyhrase Extraction

In order to construct keyphrase candidates, we compile a set of regular expressions based on the following grammar rules, and extract all matched `NP` and `VP` patterns as candidates.

```
NP: {<DT|PP$>?<JJ|JJR>*<NN.*|CD|JJ>+}
PP: {<IN><NP>}
VP: {<MD>?<VB.*><NP|PP>}
```

A.2 Stance Scoring Model

Our stance scoring model calculates the score by aggregating the sentiment words surrounding the opinion targets. Here we choose the keyphrases of input statement as opinion targets, denoted as $\mathbb{T}$. We then tally sentiment words, collected from Hu and Liu (2004), towards targets in $\mathbb{T}$, with positive words counted as $+1$ and negative words as -1. Each score is discounted by $d_{\tau,l}^{-5}$, with $d_{\tau,l}$ being the distance between the sentiment word l and the target $\tau \in \mathbb{T}$. The stance score of a text psg (an input statement or a retrieved passage) towards opinion targets $\mathbb{T}$ is calculated as:

$$Q(psg, \mathbb{T}) = \sum_{\tau \in \mathbb{T}} \sum_{l \in psg} \text{sgn}(l) \cdot d_{\tau,l}^{-5} \qquad (13)$$

In our experiments, we only keep passages with a stance score of the opposite sign to that of the input statement, and with a magnitude greater than 5, i.e. $|Q(psg, \mathbb{T})| > 5$ (determined by manual inspection on training set).

A.3 List of Discourse Markers

As described in §5.2 in the main paper, we use a list of discourse markers together with topic signature words to label argumentative content sentences. The following list of discourse markers are manually selected from the Appendix B in Prasad et al. (2008).

- **Contrast:** although, though, even though, by comparison, by contrast, in contrast, however, nevertheless, nonetheless, on the contrary, regardless, whereas
- **Restatement/Equivalence/Generalization:** eventually, in short, in sum, on the whole, overall
- **Result:** accordingly, as a result, as it turns out, consequently, finally, furthermore, hence, in fact, in other words, in short, in the end, in turn, therefore, thus, ultimately