End-to-end Argument Generation System in Debating

Misa Sato Kohsuke Yanai Toshihiko Yanase
Toshinori Miyoshi Makoto Iwayama Qinghua Sun Yoshiki Niwa
Hitachi Ltd. Research & Development Group
1-280, Higashi-koigakubo, Kokubunji-shi, Tokyo 185-8601 Japan
{misa.sato.mw, kohsuke.yanai.cs, toshihiko.yanase.gm,
toshinori.miyoshi.pd, makoto.iwayama.nw,
qinghua.sun.ap, yoshiki.niwa.tx}@hitachi.com

Abstract

We introduce an argument generation system in debating, one that is based on sentence retrieval. Users can specify a motion such as *This house should ban gambling*, and a stance on whether the system *agrees* or *disagrees* with the motion. Then the system outputs three argument paragraphs based on "values" automatically decided by the system. The "value" indicates a topic that is considered as a positive or negative for people or communities, such as *health* and *education*. Each paragraph is related to one value and composed of about seven sentences. An evaluation over 50 motions from a popular debate website showed that the generated arguments are understandable in 64 paragraphs out of 150.

1 Introduction

This paper describes our end-to-end argument generation system, developed to participate in English debating games as an AI debater. When users give a "motion" like *This house should ban gambling* and a "stance" on whether the system should *agree* or *disagree* with the motion, the system generates argument scripts in the first constructive round of a debate.

Among NLP communities, interest is growing in argumentation, such as argumentation mining and claim detection (Levy et al., 2014; Mizuno et al., 2012; Park et al., 2014). However, argument generation is still as hard a task as other text generation tasks; no standard methods or systems exist, as far as we know.

We assume that argument generation systems are helpful in a variety of decision-making situations such as business, law, politics and medical care. This is because people usually investigate existing arguments on the Internet, newspapers, or research papers before reaching conclusions. In this research, we focus on debating game style because there is similarity in argument construction between debating games and actual decision-making.

The difficulty in argument generation is that argument scripts have to be *persuasive*. We explain this need by comparing argument generation with multi-document summarization. In the two tasks, one practical approach is combining partial texts retrieved from multiple documents. Generated scripts in both tasks should be natural and have sufficient content. Because the summarization task is to generate summary scripts of multiple documents, the essential basis of its evaluation is coverage, that is, how much content in the original documents is included in the generated scripts. However, as the role of argument scripts is to persuade people, *persuasiveness* is more important than coverage.

We believe that the following three points are required to generate persuasive argument scripts:

1. Consistency with a given stance
2. Cause and effect relationships
3. Relevance to people's values

For example, when debaters focus on an *agree* stance with a motion of *This house should ban gambling*, one persuasive argument would discuss the negative effects of *gambling*. To reach a discussion about the negative effects under this condition, we need to consider the three points.

1. Consistency means that the stance of argument scripts must be equal to the given stance and consistent in the overall arguments. For example, because the gambling motion implies the claim that *gambling is negative*, the generated argument should include only negative aspects of gambling.

2. Causality makes argumentation persuasive. To capture causality, we focus on promoting/suppressing relationships. Hashimoto et

Proceedings of ACL-IJCNLP 2015 System Demonstrations, pages 109–114,
Beijing, China, July 26-31, 2015.

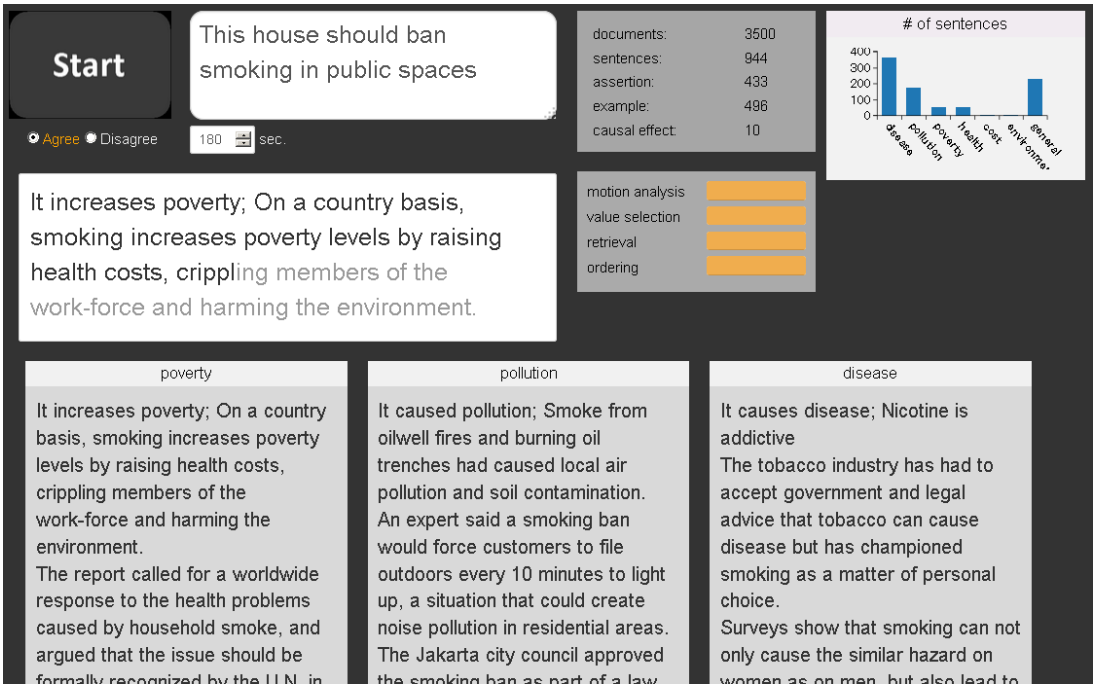

Figure 1: Screenshot and Sample Input & Output Script

al. (2012) also showed that the relationships are useful for causality extractions. The claim *gambling promotes negative issues* would be persuasive in an argumentation that agrees with a ban on gambling.

3. Values There are topics obviously considered to be positive or negative and highly relevant to people's values. For instance, *health*, *education* and *natural environment* are considered to be positive values, while *crime*, *pollution* and *high cost* are considered to be negative. It is possible to generate scripts about negative effects by collecting partial texts describing negative values linked to *gambling*, such as *crime*.

2 Overview

2.1 Demo Description

Visitors will have the opportunity to select a motion and a stance and to run the system to generate argument scripts automatically.

Each argument script generated by the system consists of three topics corresponding to values, such as *health*, *education* and *revenue*. This approach comes from our observations that persuasive arguments would be related to multiple values. Figure 1 shows the interface of the system and an example of generated argument scripts. First, users give text scripts about the "motion" and select the "stance" whether *agree* or *disagree*. In the figure, the given motion is *This house should ban smoking in public spaces*, and the given stance is an *agree* side. When users click the start button, the system begins processing. Users can see how many sentences or documents are processed and how many sentences belong to each value in the graphs in the upper right corner. Finally, the system provides three generated paragraphs with their value titles such as *poverty*, *pollution*, and *disease* while the generated argument scripts are read aloud by our text-to-speech system.

2.2 System Overview

Figure 2 shows the overview of the system.

As discussed above, the key of constructing arguments is to find positive/negative effects of a target in the motion. In this paper, we call the target "a motion keyphrase".

Positive/negative effects appear in the form of *affect* relationships like *something affects something*. Main elements of arguments are sentences that contain *affect* relationships whose subject is a motion keyphrase and whose object represents a value.

We have two types of affect predicates: *affect+* and *affect−*. *Affect+* means a promoting predicate such as *create, enhance, expand, improve, increase*. On the other hand, *affect−* means a sup-

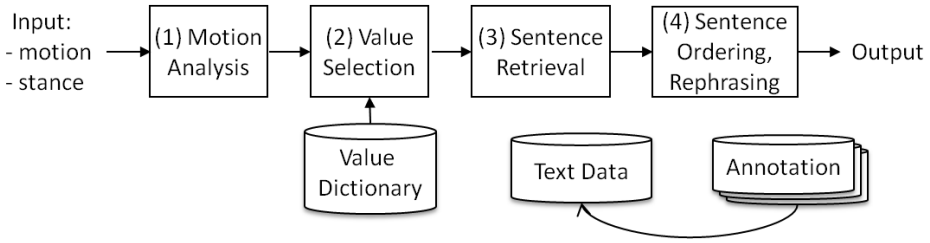

Figure 2: System Overview

pressing predicate such as *decrease, discourage, eliminate, limit, threaten*. The system stored the affect relationships in text data as automatically added annotations described in Section 4.

Though the system consists of 21 algorithms, we describe four main components in this paper:

(1) A motion analysis component decides a motion keyphrase and a polarity of arguments to be generated.

(2) A value selection component decides multiple values relevant to the given motion by retrieving sentences that contain *affect* relationships.

(3) A sentence retrieval component retrieves sentences relevant to each value from the stored text data.

(4) A sentence ordering and rephrasing component combines and arranges the retrieved sentences to construct natural argument scripts.

They are processed in a pipeline and some of the algorithms are processed in parallel on a cluster of 10 machines. We describe key functions of the components in Section 3.

The system uses large text data from Gigaword corpus (Napoles et al., 2012) and the annotations to the text data. The annotations are added automatically in a preprocessing step. Section 4 describes what kinds of annotations exploited in the system. The text and annotation data are stored using Cassandra[1], which is an open-source database management system designed for handling large amounts of data across commodity servers. They are indexed into Solr[2], open source enterprise search platform, that enables full-text search.

2.3 Evaluation

We evaluated the generated argument scripts on the basis of subjective evaluations.

[1]Cassandra: http://cassandra.apache.org

[2]Apache Solr: http://lucene.apache.org/solr

Table 1: Evaluation Results

Evaluation Score	Num of paragraphs
0: make-no-sense	86
1: understandable	38
2: +clear	16
3: +persuasive	10
4: +natural	0

We used 50 motions from a popular debate website *Debatabase* [3] as inputs to the system. The system outputs three paragraphs per motion, and each paragraph is composed of seven sentences, totaling 150 paragraphs for 50 motions. The paragraphs are rated by authors on a five point scale ranging from 0 to 4. Each evaluator judges 30 paragraphs in 10 motions. The paragraphs that do not include any claims or supporting evidence related to the motion, are given a rating of 0 points. A 1 point rating is given when the argument is understood by the evaluator, despite a number of irrelevant sentences. If more than four of the seven sentences in the paragraph are relevant to the given motion and consistent to the stance, it is given a 2 point rating. If the evaluator feels it is persuasive, it is given a 3 point rating. When it satisfies the above conditions and is presented as a natural argument, it is given a 4 point rating.

Table 1 shows the results. We found that the argumentations are understandable in 64 paragraphs (= 38+16+10+0) out of 150.

3 Pipeline Components

3.1 Motion Analysis Component

In the beginning of processing, the system analyzes the given motion text, and extracts a keyphrase, a motion polarity, a predicate, an attitude, and contexts. A predicate is a phrase which gives positive/negative sign to a keyphrase, and an

[3]Debatabase: http://idebate.org/debatabase

Table 2: Motion Analysis Results

motion	keyphrase	pol.	predicate	attitude	contexts
This house believes that casino is harmful for the city	*casino*	-1	*harmful*	*believe*	*the city*
This house would create a single EU army	*a single EU army*	$+1$	–	*create*	–
This house should ban gambling	*gambling*	-1	–	*ban*	–
This house believes that assisted suicide should be legalized	*assisted suicide*	$+1$	–	*legalize*	–

Table 3: Motion Analysis Rules. K = motion keyphrase, C = contexts.

priority	rule	predicate instances
1	K be **modify**-ed for C	**modify**: *good*$(+1)$, *honor*$(+1)$, *popular*$(+1)$, *harmful*(-1), *negative*(-1), *weak*(-1)
2	**affect** K	**affect**: *create*$(+1)$, *enhance*$(+1)$, *increase*$(+1)$, *cut*(-1), *discourage*(-1), *eliminate*(-1)
3	**believe** K	**believe**: *allow*$(+1)$, *legalize*$(+1)$, *permit*$(+1)$, *support*$(+1)$, *ban*(-1), *oppose*(-1)
4	K be **believe**-ed	**believe**: *allow*$(+1)$, *legalize*$(+1)$, *permit*$(+1)$, *support*$(+1)$, *ban*(-1), *oppose*(-1)
...	...	...

attitude is a predicate of *this house*. Table 2 shows results of motion analysis.

To analyze a motion, the system has 22 rules with their priority. Table 3 shows a part of the rules. The rules are applied in the order by their priority, until a motion keyphrase is extracted.

Suppose that the given motion is *This house believes that casino is harmful for the city* and the given stance is *agree*$(+1)$ (corresponding to the first line of Table 2). The first rule "K be **modify**-ed for C" in Table 3 matches the motion. As *harmful* is a *modifying* predicate, *casinos* is K and *the city* is C. An attitude of *this house* is *believe*. A motion polarity is -1 because of the negative predicate *harmful*(-1). In the same way, from the second to the fourth rules in Table 3 can analyze the other three motion examples in Table 2.

The system calculates an argument polarity by multiplying the sign of the given stance and the motion polarity. The system constructs arguments that discuss the motion keyphrase, in accordance with the argument polarity. For example, if the given stance is *agree*$(+1)$ and the motion polarity is *negative*(-1), then the system decides an argument polarity is -1 and constructs arguments that claim "the motion keyphrase is negative(-1)".

3.2 Value Selection Component

The value selection component decides multiple values relevant to the given motion by using a value dictionary. The value dictionary formulates a set of values that represents what is important in human's life, what is harmful in communities, etc. Each value is regarded as a viewpoint of the generated argument.

Table 4 describes an example of the value dictionary. As shown in Table 4, each value (e.g., disease, investment) belongs to a field (e.g., health, economy, respectively), and has three attributes: a value polarity, representative phrases, and context phrases. The value polarity $+1(-1)$ means that the value is something good(bad). The representative phrases are linguistic expressions of the values, and the numbers are their weights calculated by IDF in Gigaword corpus. The context phrases are phrases that are likely to appear in neighbor of the value in text documents. They are used to solve ambiguity of the linguistic expressions. The value dictionary of the current system contains 16 fields and 61 values.

The procedure of the value selection is below:

Step 1 Retrieves sentences that contain affect relationships between the motion keyphrase and one of representative phrases in the value dictionary. For instance, it retrieves *Gambling increases the number of crimes*.

Step 2 Calculates a polarity to the keyphrase in each sentence, and filters out the sentences where the polarity is not equivalent to the argument polarity. For instance, the polarity for *Gambling increases the number of crimes* is -1 by multiplying $+1$ (*increase* in the affect dictionary) and -1 (*crime* in the value dictionary), which equals to the argument polarity.

Step 3 Sums weights of found values and selects the top five values.

The value dictionary was created manually. First, fields of the dictionary were determined by the authors in reference to the roles of government agencies, and then value entries related to each field were chosen manually from Debatabase. Second, a rule-based extractor that extracts values discussed in a document was constructed using the dictionary, and the extractor applied to each docu-

Table 4: Value Dictionary

field	value	polarity	representative phrases	context phrases
economy	investment	$+1$	investment:27.8, development_aid:48.2	asset, bank, capital, fund, profit, stock, ...
finance	cost	-1	expense:35.9, expenditure:55.7	budget, dollar, fuel, lower, price, share, ...
finance	income	$+1$	revenue:35.4, wage:39.8	budget, company, earnings, higher, gain, ...
health	disease	-1	disease:36.6, complication:40.1	AIDS, Alzheimer, blood, cancer, death, ...
safety	crime	-1	crime:31.5, prostitution:56.2	arrest, gun, jail, kidnapping, victim, ...
...	...	...	...	...

ment in Debatabase. Third, we manually added new entries to the dictionary. If a value is extracted from a document, we extracted representative/context phrases corresponding to the value from the document. If no value is extracted, we extracted new values that were contained in the document. We continued these steps of classifying documents and adding entries to the dictionary like a Bootstrapping method.

3.3 Sentence Retrieval Component

This component retrieves sentences relevant to each value from the stored text data.

It first retrieves documents using a query composed of weighted phrases. The retrieved documents should contain both the motion keyphrase and more than one representative phrases of the decided values. While the motion keyphrases can be replaced with their synonyms or hyponyms, their weights are smaller than the original keyphrases; those of synonyms are 0.75 and those of hyponyms are 0.5. The synonyms and hyponyms are acquired by WordNet (Miller, 1995). Because short documents don't usually contains informative scripts, the length of retrieved documents are limited to more than 200 words.

For example, when the motion keyphrase is *gambling*, a search query for a *health* value is

```
(gambling#49.53 OR gaming#22.87)
AND (health#27.48 OR disease#36.60
  OR addiction#52.39
  OR hospital#29.76)
AND (length:[200 TO *]).
```

The real numbers following sharp signs are weights of the former phrases, calculated by multiplying the IDF of the phrase and a synonym or hyponym reduction rate.

The retrieval step prefers sentences that contain promote/suppress relationships. The polarities of the retrieved sentences must be equal to the argument polarity. The polarity of each sentence is calculated by the product of the signs of related phrases, such as the predicate of the keyphrase, the promote/suppress verb, and the representative value phrase. In the example of *gambling ban*(-1) *decrease*(-1) *the number of crimes*(-1), the polarity of the sentence is -1.

The system uses about 10,000,000 newswire documents and retrieves 500 per value in this step.

3.4 Sentence Ordering and Rephrasing Component

This component processes the sentence set of each value separately.

The sentence ordering step orders the retrieved sentences in the natural order in debating by the method reported in (Yanase et al., 2015). The method employs an assumpsion that a constructive speech item in debating consists of a claim sentence and one or more supporting sentences, and that the claim sentence lies in the first position of the paragraph. The assumption is implemented as two machine learning problems: claim sentence selection and supporting sentence ordering. The claim sentence selection problem is formulated as a binary-classification problem where a given sentence is a claim or not. In the supporting sentence ordering problem, the method orders the other sentences on the basis of connectivity of pairs of sentences. This problem is formulated as a ranking problem, similarly to (Tan et al., 2013). Features for machine learning models in both problems are extracted not only from the sentences to be ordered but also from the motion text.

Finally, the rephrasing step trims or replaces surplus phrases referring to too many details for argument scripts, such as dates and people's names. Several simple rules are used.

4 Data Preprocessing: Annotations

The system adds annotations automatically in preprocessing into the stored text data by using dictionaries and syntax information by Stanford Core NLP (Manning et al., 2014). In the current system, about 250 million annotations are stored. Users

can add the annotations manually. A list of main semantic annotations is below:

affect: promoting/suppressing relationships. For example, it adds an annotation of "affect+: *casino → the number of crimes*" into a text *casino increases the number of crimes*. The affect dictionary, which is manually created, contains 608 positive phrases and 371 negative phrases.

modify: phrases which gives positive/negative sign to words governed by them. For example, it adds an annotation of "modify−: *environment*" into a text *harmful environment*. The modification dictionary contains 79 positive phrases and 134 negative phrases.

believe: relationships which represents attitudes of a subject to its object. For example, it adds an annotation of "believe−: *smoking*" into a text *The government bans smoking in public spaces* because of a negative believe phrase *ban*. The believe dictionary contains 30 positive phrases, 47 negative phrases and 15 neutral phrases.

5 Error Analysis

We describe three major problems of the system here: (1) identification errors, (2) polarity errors, (3) motion format limitation.

(1) Identification errors occur on recognizing a motion keyphrase in text data on sentence retrieval step. The system can incorrectly retrieve sentences including mentions whose expressions are the same as or similar to the motion keyphrase but different in their meanings. In the screenshot of Figure 1, for example, although "smoking" in the motion refers to "tobacco smoking," the first sentence in the *pollution* paragraph argues about "smoking caused by a fire."

The identification problem is especially obvious in the case a motion keyphrase forms a compound noun. For instance, on the motion of *This House should ban cosmetic surgery*, it is not clear if *surgery* in some text is equal to *cosmetic surgery* or not. The errors would show requirements of more precise word sense disambiguation or coreference resolution among multiple documents.

(2) Polarity errors are not so rare. Regarding the *disease* paragraph in Figure 1, the second sentence would contain an argument on the opposite stance in error.

(3) Motion format limitation is that the system can process only motions in formats which ask people if its motion keyphrase should be banned or permitted. Representative examples of unacceptable motions are comparison like *This house believes that capitalism is better than socialism* and questions of an adequate degree like *This House should lower the drinking age.*

6 Conclusion

We described a demonstration of our argument generation system. Our system can generate understandable arguments on a given motion for a given stance. Our next work is to generate counterarguments, which argue against the opponents.

Acknowledgments

We would like to thank Prof. Kentaro Inui from Tohoku University for valuable discussion.

References

Chikara Hashimoto, Kentaro Torisawa and Stijn De Saeger. 2012. *Excitatory or Inhibitory: A New Semantic Orientation Extracts Contradiction and Causality from the Web*, In *Proceedings of EMNLP-CoNLL 2012*, pages 619–630.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard and David McClosky. 2014. *The Stanford CoreNLP Natural Language Processing Toolkit*, In *Proceedings of ACL 2014 System Demonstrations*, pages 55–60.

George A. Miller. 1995. *WordNet: A Lexical Database for English* In *Communications of the ACM*, 38(11): pages 39–41.

Joonsuk Park and Claire Cardie. 2014. *Identifying Appropriate Support for Propositions in Online User Comments* In *Proceedings of the First Workshop on Argumentation Mining*, pages 29–38.

Junta Mizuno, Eric Nichols, Yotaro Watanabe and Kentaro Inui. 2012. *Organizing Information on the Web through Agreement-Conflict Relation Classification* In *Proceedings of the Eighth Asia Information Retrieval Societies Conference* , pages 126–137.

Napoles, Courtney, Matthew Gormley and Benjamin Van Durme. 2012. *Annotated English Gigaword LDC2012T21*. Web Download, Philadelphia: Linguistic Data Consortium.

Ran Levy, Yonatan Bilu, Daniel Hershcovich, Ehud Aharoni and Noam Slonim. 2014. *Context Dependent Claim Detection* In *Proceedings of COLING 2014: Technical Papers*, pages 1489–1500.

Jiwei Tan, XiaojunWan and Jianguo Xiao 2013. *Learning to order natural language texts* In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 87-91.

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.