Abstractive Multi-document Summarization by Partial Tree Extraction, Recombination and Linearization

Litton J Kurisinkel
IIIT-H, Hyderabad
litton.jKurisinkel@research.iiit.ac.in

Yue Zhang
SUTD, Singapore
yue_zhang@sutd.edu.sg

Vasudeva Varma
IIIT-H, Hyderabad
vv@iiit.ac.in

Abstract

Existing work for abstractive multi-document summarization utilise existing phrase structures directly extracted from input documents to generate summary sentences. These methods can suffer from lack of consistence and coherence in merging phrases. We introduce a novel approach for abstractive multi-document summarization through partial dependency tree extraction, recombination and linearization. The method entrusts the summarizer to generate its own topically coherent sequential structures from scratch for effective communication. Results on TAC 2011, DUC- 2004 and 2005 show that our system gives competitive results compared with state-of-the-art abstractive summarization approaches in the literature. We also achieve competitive results in linguistic quality assessed by human evaluators.

1 Introduction

Multi-document summarization generates a textual summary from a corpus of documents dealing with a set of related topics. An optimum generated summary should encompass the most relevant and topically diverse content, which can represent the input corpus in stipulated summary space. Extractive multi-document summarization approaches pick out a subset of sentences to constitute the summary, which can be noisy and incoherent, as all the portions of a sentence may not be relevant for summary generation (Lin and Bilmes, 2011).

An abstractive multi-document summarizer, in contrast, infers the most relevant information and generates summary sentences exhibiting coherence and fluency. There has been relatively little existing work on abstractive multi-document summarization. Bing et al. (2015) merge phrases that are extracted from input documents into a coherent summary. Banerjee et al. (2015) utilize multi-sentence compression for summarization. Both of the above approaches rely on sequential arrangement of phrasal or subsentential structures existing in the input corpus to generate summary. However, an ideal abstractive multi-document summarizer should enjoy the freedom to exhibit its own writing style and to generate the sentences from scratch.

We build a model to this end by leveraging syntactic dependencies. Input for our model is the set of syntactic dependency trees obtained by parsing sentences in the corpus to be summarized. Relevant and noise pruned partial tree structures are extracted from the set of dependency trees and different subsets of maximally relevant partial dependency structures are identified. Partial trees in different subsets are linearized to generate individual summary sentences. In this work, we utilize transition-based syntactic linearization approach proposed by Puduppully et al. (2016) to linearize a combination of partial trees and to generate a noise free summary sentence. The combinability of a set of partial trees to form a full dependency tree of a valid sentence is estimated using a generative model of syntactic dependency trees (Zhang et al., 2016). As a result, the model is allowed to exhibit its own learnt writing style while generating summary sentences.

The summaries generated by our system are evaluated on the DUC 2004, DUC 2007 and TAC 2011 muti-document summarization data-sets. In addition, we relied on human evaluation to evaluate factual accuracy and linguistic quality of generated summary sentences. To our knowledge this is the first work on multi-document abstrac-

Proceedings of the The 8th International Joint Conference on Natural Language Processing, pages 812–821,
Taipei, Taiwan, November 27 – December 1, 2017

tive summarization with syntactic dependency trees, which entrust the summarization model to generate summary sentences without exploiting any kind of subsentential or phrasal sequential structures originally present in the input corpus. Our code is released at https://bitbucket.org/litton_kurisinkel/tree_sum

2 Related Work

Text summarization can be achieved using extractive (Takamura and Okumura, 2009; Lin and Bilmes, 2011; Wang et al., 2008) and abstractive methods (Bing et al., 2015; Li, 2015). Extractive summarization has the advantage of output fluency due to direct use of human-written texts. However, extractive summarization cannot ensure a noise free and coherent summary. It can also result in a wrong inference to the reader due to out of context sentence usage. In contrast, abstractive summarization techniques can generate a noise-free summary out of most relevant information in the input corpus.

A subset of previous extractive summarization approaches utilized parsed sentence structures to execute noise pruning while extracting content for summary (Morita et al., 2013; Berg-Kirkpatrick et al., 2011). As a first step towards abstracting content for summary generation, sentence compression techniques were introduced (Lin, 2003; Zajic et al., 2006; Martins and Smith, 2009; Woodsend and Lapata, 2010; Almeida and Martins, 2013), but these techniques can merely prune noise, and cannot combine related facts from different sentences to generate new ones. (Banerjee et al., 2015) suggests a better way of doing sentence compression without harming linguistic quality.

Recent work attempts to solve the problem of abstractive multi-document summarization (Bing et al., 2015; Li, 2015), claiming that the method has the advantage of generating new sentences. Bing et al. (2015) extracts relevant noun phrases and verb phrases and recombines them to generate new sentences while Li (2015) system make use of semantic link network on basic semantic units (BSUs) to generate summary. Neither of these methods employ a learnt model to generate summary sentences. Instead, they make use sequential structures in the source text itself to construct the summary sentences. Cheng and Lapata (2016) propose a fully data driven approach using neural network for single document summarization by extracting words. They have treated highlighted text in news articles consisting of very short bulleted lines on the web as summary of the corresponding article.

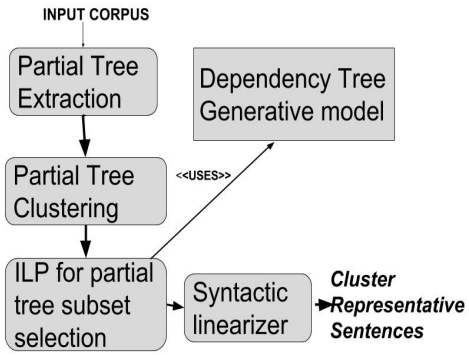

Figure 1: Overall approach

A major challenge of using a fully data driven approach for multi-document abstractive summarization using neural network is the expensive task of creating dataset which can be used to jointly model extraction of relevant content and generation good quality summary sentences. But multi-document abstractive summarization becomes an achievable task, by disintegrating the extraction of knowledge granules from the input corpus and using a separate language generation model trained in a sophisticated dataset which can take a subset of extracted knowledge granules and generate a summary sentence of high linguistic quality. To this end, we leverage partial tree linearization (Zhang, 2013) synthesizing summaries from extracted treelets.

3 Approach

The overall approach (Figure 1) for abstractive multi-document summarization detailed in the current work starts with extracting the most relevant set of partial trees of varying sizes from the set of all syntactic dependency trees in the corpus using maximum density sub-graph cut algorithm (Su et al., 2008). In parallel, generative model for syntactic dependency trees is trained (Zhang et al., 2016) so that it can be leveraged for estimating the combinability of a set of partial trees for constructing a whole or part of a full dependency tree of a valid sentence during summary generation. Also, the transition based syntactic linearization model Puduppully et al. (2016) is trained using

a dataset, consisting of dependency trees of sentences in Penn-tree bank and corresponding sentence.

The set of extracted partial dependency trees are clustered to ensure topical diversity. We identify a subset of partial trees from each cluster, which can be linearized to a single sentence which represents the cluster in the final summary. Integer linear programming is used for locating the most accurate subset of partial trees out of which representative sentence can be generated.

The objective function consists of linear components for maximising *total relevance* and *total combinability* of the subset of partial trees selected. *total combinability* is measured using the generative model Zhang et al. (2016). Here we try to jointly model the human summarizers method of collecting relevant knowledge granules and deciding the combinable set of information. Cluster representative sentence are generated using the linearization model of Puduppully et al. (2016) from the subset of partial trees identified earlier. The following sections explain how the system achieves each one of the tasks listed above in detail.

3.1 Extracting Relevant Partial Trees

For each dependency tree in the input corpus, there is a subtree rooted at every node. Each subtree do not necessarily contain extractable information to generate a summary. The method used to create a noise pruned subset of subtrees containing cognizable information from the set of all subtrees in the corpus can be split into two steps.

- Identify the roots of valid subtrees containing cognizable information in all the syntactic dependency trees in the input corpus.
- Prune the identified subtrees so that it contains nodes relevant for generating summary.

3.1.1 Identify Subtree Roots

The level of granularity at which syntactic elements chosen to generate summary sentence considerably decides factual accuracy of a generated sentence. As extracted set of subtrees are basic building units for summary sentence generation, the structure of subtrees extracted should be suitable for generating factually correct summary sentences with respect to the corpus to be summarized. We observe that a node containing *subject*

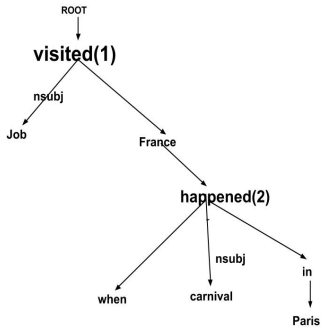

Figure 2: Marking relevant subtree roots

relationship with its child node is a valid candidate. Consequently any node in a dependency tree sharing dependency relations such as *nsubj, csubj, nsubjpass* or *xsubj* with any of its child node is treated as the root node of a subtree which contains extractable and cognizable information. In Figure 2, for example, subtrees rooted at words *visited* and *happened* are valid subtrees to contribute for summary generation as per the linguistic criteria discussed above.

3.1.2 Pruning Subtrees

Identified subtrees are dissected out from their original dependency trees after pruning away their noisy portions which are irrelevant for summary generation. Su et al. (2008) introduced a dynamic programming approach to find a length-constrained maximum-density subtree containing the root r, from a tree rooted at r for which weight and length is preset for all edges. The density of a tree T is defined as follows.

$$density = \frac{\sum_{e \epsilon E} W(e)}{\sum_{e \epsilon E} Len(e)} \quad (1)$$

where *E* is the set of all edges in *T*, *W(e)* is the weight of edge *e* and *Len(e)* is the length of edge *e*. The constraint on length implies that total length of all edges in the resultant maximum density subtree should be between lower bound L and upper bound U which are taken arguments by the algorithm. We leverage multi density subtree extraction algorithm for dissecting out a noise pruned subtree from an identified valid subtree in a dependency tree after setting values for edge weights, edge lengths, *L* and *U*.

The weight of an edge in a syntactic dependency in the input corpus should represent its topical relevance for summary generation. Consequently the

weight depends on the frequency of *bigram* constituted by words on either side of the dependency edge(D_{bigram}). We set the weight of edge *e* as,

$$W(e) = \frac{\log(1 + fd_{bigram})}{depth(e)^2}, \qquad (2)$$

where fd_{bigram} is the frequency of the D_{bigram} of *e* in the corpus and *depth(e)* is the depth of dependent node of e in the tree. Summary should prefer general information over context specific information. The specificity of the information contained increases with depth in dependency tree and the denominator term in Equation 2 penalizes deeper edges.

The length of the edge is set as the size of the word in the dependent node in bytes so that length of all edges in the tree equals the total size of all words in the tree. Length constraint *U* is the total length of all edges in the tree and *L* is calculated as follows

$$L = \alpha * \tanh(\beta * twe - \sigma) * TL,$$

where *twe* is the total weight of all edges in the subtree, TL is the total length of all edges in the subtree and σ, α and β are constants optimized empirically. σ sets the universal upper bound for all subtrees minimum length while α and β set the slope and position of tanh curve. The value of L ensures that subtrees containing more relevant information are pruned lesser. We enforce a rule to retain the tree nodes to maintain grammaticality while pruning. If the grammatical relation pointing to a node from its parent is nsubj, csubj, nsubjpass, xsubj, aux, xcomp, pobj, acomp, dobj, case, det, poss, possessive, auxpass, ccomp, neg, expl, cop, prt, mwe, pcomp, iobj, number, quantmod, predet, dep or mark, then the node cannot be removed without also removing its parent to maintain grammatical and factual correctness.

For the rest of this paper, we refer to a noise-pruned subtree extracted out of a dependency tree as a *partial tree*. The partial trees pruned out of subtrees sharing anscestor-descendant relationship may contain overlapping information (eg: subtrees rooted at 'visited' and 'happened' in Figure 2). So ancestor-descendant relationship between corresponding partial trees is recorded to avoid redundant information during summary sentence generation.

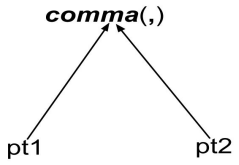

Figure 3: Combine(pt_1, pt_2)

3.2 Estimation of Combinability of Partial trees using Generative Model of Dependency Trees

During the final stage of summary sentence generation, each of the summary sentence is generated from a precisely identified subset of relevant partial trees. To arrive at such a precise subset, along with information regarding topical relevance, due consideration should be given to the combinability of partial trees in the subset to form a full dependency tree of a valid sentence w.r.t to the syntactic and semantic information contained in them. As a primitive measure upon which total combinability of a set of partial trees can be built upon, we estimate the combinability of any two partial trees to form whole or a part of a valid dependency tree as follows

$$DepTree \rightarrow Combine(pt_1, pt_2) \qquad (3)$$

$$C(pt_1, pt_2) \rightarrow P_{depgen}(DepTree), \qquad (4)$$

where *Combine*(pt_1, pt_2) combines two partial trees pt_1 and pt_2 as represented in Figure 3 by adding a dummy root containing ',' with 'comma' as the dependency relation of edges and pt_1 comes before pt_2 in breadth first order. $C(pt_1, pt_2)$ is the *Combinability* of any two partial trees and P_{depgen} represents a generative distribution of syntactic dependency trees trained with a large set of dependency trees. *Combine*(pt_1, pt_2) need not exactly represent a substep in the final process of combination and linearization of a set of selected partial trees to generate a summary sentence. But $P_{depgen}(DepTree)$ acts as an indicative measure on how much pt_1 and pt_2 can together participate in the construction of a full-dependency tree of a valid sentence with respect to the syntactic and semantic information contained in them.

3.2.1 Learning P_{depgen}

Zhang et al. (2016) introduced Tree Long Short-

Term Memory (TreeLSTM), a neural network model based on LSTM, which is designed to predict a tree rather than a linear sequence. They define probability of a sentence as the generation probability of its dependency tree. Under the assumption that each word *w* in a dependency tree is only conditioned on its dependency path, the probability of a sentence *S* given its dependency tree *T* is:

$$P(S|T) = \prod_{w \epsilon BFS(T) \setminus ROOT} P(w|D(w)) \quad (5)$$

where D(w) is the dependency path of w and how dependency path for each node is identified is detailed in the paper Zhang et al. (2016) and each word *w* is visited according to its breadth-first search order (BFS(T)). $P(S|T)$ can be restated as a generative probability $P_{depgen}(T^{'})$ where $T^{'}$ is a restricted syntactic dependency tree structure in which, for all nodes, left and right dependants and breadth first order of its children are fixed. As the above mentioned structural restrictions can be fixed in parameter for partial trees for *Combine* in Equation 3, we can directly use TreeLSTM network to estimate P_{depgen} in Equation 4.

The sentence dataset shared by Zhang et al. (2016) is parsed using Standford parser for training a TreeLSTM network. The negative log probability values produced by the network is normalized for partial tree pairs in the corpus.

3.3 Syntactic Linearization

We make use of the syntactic linearization model proposed by Puduppully et al. (2016) to linearize the input set of partial trees. Puduppully et al. (2016) and Liu et al. (2015) propose a transition-based word ordering model, which takes a bag of words, together with optional POS and dependency arcs on a subset of input words, yields a sentence together with its dependency parse tree that conforms to input syntactic constraints (Zhang, 2013). The system is flexible with respect to input constraints, performing abstract word ordering when no constraints are given, but gives increasingly confined outputs when more POS and dependency relations are specified. We retrain their model[1] using autoparsed data obtained using Stanford Dependency Parser.

[1]https://github.com/SUTDNLP/ZGen

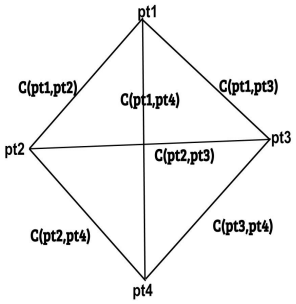

Figure 4: Cluster graph of partial trees

3.4 Clustering for Topical Diversity

To ensure topical diversity we apply K-means clustering in the set of partial trees with an aim of generating a sentence from each of the cluster as a topical representative of the respective cluster. Cosine similarity between D_{bigram} frequency vectors of partial trees is treated as the similarity metric during clustering. Number of clusters (K) is decided using the relation,

$$K = \lfloor q * ShannonEnt \rfloor \quad (6)$$

where *ShannonEnt* is the Shannon entropy of unigram distribution of the corpus and *q* is a constant.

3.5 Generating Cluster Representative Sentence

Our system generates a single sentence from each of the partial tree clusters identified as described in section 3.4 to represent the cluster in final summary. Here we search for a subset of partial trees from each cluster which maximise the *total relevance* and *total combinability* of the partial trees in the selected subset. Relevance of a partial tree is defined as the total weight of all dependency edges calculated using the Equation 2.

A data structure which searchably organizes the relevance and combination probabilities of partial trees belonging to a cluster is essential for formulating the partial tree subset selection as an integer linear programming problem. For this purpose we visualize the entire partial tree information organized in a *cluster graph (**CG**)* as shown in Figure 4 in which *nodes* represent partial trees, and *edge weights* represent the combination probability of partial trees represented by the edge nodes calculated using the Equation 4. An *edge exists* between two nodes if the partial trees at the nodes contain mention about same named entity.

From cluster graph we try to extract a *connected subgraph (**SG**)* which maximizes the

objective function which takes a binary indicator vector representing a sub-graph of cluster graph as argument.

$$F(x_1, \ldots, x_n, e_{1,1}, \ldots, e_{n-1,n}) \rightarrow \sum_{i \epsilon Nodes} R(x_i) * x_i + \lambda * \sum_{\{i,j\} \epsilon Edges} W(\{i,j\}) * e_{ij} \quad (7)$$

subject to constraints,

$$x_i + x_j - 2 * e_{ij} >= 0 \rightarrow C_1$$

$$x_i + x_j - 2 * e_{ij} <= 1 \rightarrow C_2$$

$$e_{i,j} <= I(x_i, x_j) \rightarrow C_3$$

$$\sum_i x_i * size(i) <= MaxLen \rightarrow C_4$$

$$x_i + ancestor(i) < 1 \rightarrow C_5$$

where λ is a constant, *R(i)* is the relevance of partial tree at node i, *W({i,j})* is the weight of the edge {i,j} in CG, *Nodes* is the set of all nodes in CG and *Edges* is the set of all edges in CG.

$$x_i = \begin{cases} 1 & \text{if node i is present in input SG} \\ 0 & \text{otherwise} \end{cases}$$

$$e_{ij} = \begin{cases} 1 & \text{if edge \{i,j\} is present in input SG} \\ 0 & \text{otherwise} \end{cases}$$

$$I(x_i, x_j) = \begin{cases} 1 & \text{if edge \{i,j\} is present in CG} \\ 0 & \text{otherwise} \end{cases}$$

ancestor(i) indicator variable that represents a partial tree that is extracted from a ancestor subtree of the subtree from which i is extracted (Explained in section 3.1). *size(i)* is the total size of all words in subtree i and *MaxLen* is the maximum size of a sentence in the input corpus.

The constraints C_1 and C_2 ensure that an edge will be present if and only if the corresponding edge nodes in CG are present in the input vector, while C_3 ensures that the input vector represents a subgraph of ***CG***. The constraint C_4 keeps an upper bound on the size of sentence that is generated from a single cluster, while C_5 ensures that partial trees with overlapping information are not present together in the selected subset of partial trees in a cluster.

The set of partial trees represented by the nodes of the subgraph that maximizes the objective function in Equation 7 functions as the syntactic ingredients to generate the cluster's representative sentence. A selected subset of partial trees are linearized using the transition based syntactic linearization model detailed in Section 3.3 to generate cluster representative sentence.

4 Experiments

We evaluate our method using the test sets of DUC 2004, DUC-2007 and TAC-2011. In particular, the set of attributes of the summary including content coverage of summaries and linguistic quality and factuality of newly generated summary sentences are evaluated. The content coverage is evaluated using ROUGE (Lin, 2004) and we relied on human evaluation for evaluating linguistic quality and factuality.

4.1 Data

DUC 2004, DUC-2007 and TAC-2011 consist of several corpora, each of them consisting of 10 documents and four model summaries for those 10 documents. We have tuned our development parameters using DUC 2003 dataset.

4.2 Settings

The values of α, β, σ, q and λ are tuned on the development set for optimum content coverage and sentence quality. In order to objectively evaluate a summary sentence generated by linearizing a set of partial trees, we need a human written reference sentence of high linguistic quality which is written after carefully understanding the information contained in selected partial trees. As it is timeconsuming to create such reference sentences for each of the combination constituted by the possible values of α, β, σ, q and λ, we choose to separate parameter tuning for optimal content coverage and sentence quality. Optimal values of α, β, σ and q contributes prominently for better content coverage while that of λ contributes for better sentence quality as it weights combinability of partial trees. In Subsections 4.2.1 and 4.2.2 we explain how we tune different development parameters for optimal content coverage and sentence quality.

4.2.1 Tuning α, β, σ and q for maximum content coverage

The values α, β, σ and q are optimised for maximum content coverage where pruned partial trees are extracted to fill the allotted summary space without any combination to generate summary sentences. Content coverage is measured as the sum of ROUGE-1 and ROUGE-2 scores with reference summaries on the DUC-2003 dataset. The values of α, β, σ and q optimized using grid search to give maximum average ROUGE score for corpora in DUC-2003 are 0.5, 0.15, 0.5 and 1 respectively.

System	DUC 2004			DUC 2007			TAC 2011	
Metric	R-1	R-2	R-SU4	R-1	R-2	R-SU4	R-2	R-SU4
CompAbsum (Banerjee et al., 2015)	-	0.120	**0.148**	-	-	-	-	-
PhraseAbSum (Bing et al., 2015)	-	-	-	-	-	-	**0.117**	**0.147**
Semantic (Li, 2015)	-	-	-	0.421	**0.110**	**0.150**	-	-
WordCoverage(D_{bigram})	0.382	0.096	0.113	-	-	-	-	-
PartTreeAbSum(ours)	**0.439**	**0.120**	0.140	**0.431**	0.109	**0.150**	0.113	0.141

Table 1: Comparison with state of the art

λ	ROUGE	BLEU	NCS
0	0.431	0.41	17%
9	0.429	0.41	27%
27	0.410	0.48	61%
42	0.403	0.54	70%
48	**0.401**	**0.571**	70%
75	0.373	0.579	77%
105	0.361	0.570	79%

Table 2: Optimizing λ for better topical coverage sentence quality

4.2.2 Tuning λ for optimum content coverage and sentence quality

The values of α, β, σ and q are preset to the optimum values identified in the section above. The value of λ is varied from 0 to 100 with an increment of 3 and set partial trees to form each of the summary sentences for all corpora in DUC 2003 is identified. We asked a human annotator to write a linear sentence out of each of the selected partial trees sets without using new words which are not present in partial tree nodes in the set and the BLEU score of generated summary sentences with respect to the human written sentences is estimated. For each value of λ we estimate the average ROUGE-1 and BLEU for the generated summaries and summary sentences respectively. The value at which total value of average BLEU and ROUGE-1 scores is maximum is set as the value of λ during testing. Table 2 reports BLEU and ROUGE-1 for different values of λ. Column NCS in the Table 2 represents the percentage of complex sentences generated by linearizing more than one partial tree.

4.3 Final Results

4.3.1 Content coverage

While evaluating the relevant content coverage of abstractive summarization system, we also have to evidently substantiate the effectiveness of noise pruning done using multi density partial tree extraction algorithm. For this purpose we have created extractive summarizer using maximum weighted word coverage algorithm Takamura and Okumura (2009), which tries to extract sentences containing maximum weighted D_{bigram}s in their dependency trees.

Table 1 shows the results on DUC-2004, DUC-2007, TAC-2011 along with previous approaches. In the table R-1,R-2 and R-SU4 represents ROUGE-1, ROUGE-2 and ROUGE-SU4 (skip-bigrams with unigrams), respectively. *WordCoverage*(D_{bigram}) summarizer using maximum weighted word coverage algorithm represents the word coverage algorithm using D_{bigram} weights while *PartTreeAbSum* abstractive summarization approach detailed in the paper. Results on DUC-2004 shows that noise-pruning using maximum weighted partial tree extraction was effective in terms of better content coverage. Despite rephrasing content in many contexts, *PartTreeAbSum* shows results comparable with previous approaches. CompAbsum (Banerjee et al., 2015) demands high syntactic overlap between source sentences to recombine and generate new sentences, otherwise resembles an extractive summarization system. Our system shows better ROUGE-1 and equal ROUGE-2 values in DUC-2004 test set. PhraseAbSum (Bing et al., 2015) which extracts phrases from the corpus and phrases can enjoy lower granularity in terms of information content when compared to partial trees by compromising topical coherence in summary sentence generation. Still our results are comparable with that of PhraseAbSum in TAC 2011. Our results show competitiveness with Semantic (Li, 2015) which generate summary content from a corpus level semantic network utilizing linear structures smaller than a partial tree and does not employ explicit means to ensure gramaticality.

4.3.2 Linguistic quality and factual accuracy

In order to fully evaluate the effectiveness of an abstractive summarization approach it is also useful to evaluate the linguistic quality and factual accuracy of generated sentences. Here linguistic quality refers to the quality of sentences in terms

Input Corpus Sentences
Hun Sen's Cambodian People's Party won 64 of the 122 parliamentary seats in July's elections [1]
Sam Rainsy and a number of opposition figures have been under court investigation for a grenade attack on Hun Sen's Phnom Penh residence on Sep. [2]
Hun Sen was not home at the time of the attack. [3]
Ranariddh and Sam Rainsy have charged that Hun Sen's victory in the elections was achieved through widespread fraud. [4]
Sentence Generated by *PhraseAbSum* **(Bing et al., 2015)**
Sam Rainsy and a number of opposition figures, have been under court investigation for a grenade attack on Hun Sen's Phnom Penh residence on Sep, charged that Hun Sen's victory in the elections was achieved through widespread fraud (*source sentences [2] and [4]*)
Sentences Generated by *PartTreeSum* **(Current Work)**
Sam Rainsy and a number of opposition figures have been under court investigation for attack on Hun Sen residence, at the time of the attack Hun Sen (He) was not home (*source sentences [2] and [3]*)
Hun Sen's party won 64 of the 122 parliamentary seats in elections, victory in the elections was achieved through widespread fraud, Ranariddh and Sam Rainsy have charged (*source sentences [1] and [4]*)

Table 3: Tree Combination vs Phrase Combination

	LQ	FA
Human Summary	4.5	4.3
PartTreeSum(WC)	2.1	2.32
PartTreeSum	**3.15**	**3.09**

Table 4: Human Evaluation on sentence quality

of grammaticality and readability, and factual accuracy refers to how much the information conveyed by the generated summary sentences are true with respect to what is contained in the input corpus. For this purpose we employed 4 manual evaluators who are post-graduate students in English literature. 10 random corpora were chosen for manual evaluation and we asked the evaluators to read the documents in each corpus and rate corresponding summary sentences for their linguistic quality and factual accuracy.

For each corpus the summaries participated in manual evaluation include a randomly chosen human summary for corpus, current approach for abstractive summarization (PartTreeSum), the current approach without combinability measure by setting λ to 0 (PartTreeSum(WC)). Human evaluation results shown in Table 4 proves that linguistic quality and factual accuracy have considerably increased with the introduction of combinability measure.

4.4 Discussions

Tree combination vs phrase combination: Table 3 contains the fours input corpus sentences in one test example and sentences generated by *PhraseAbSum* (Bing et al., 2015) and the current work (*PartTreeSum*), *PhraseAbSum* could generate only one sentence respectively, due to the hard constraint for verb phrases to coincidentally share same noun phrase in source sentences and the sentence exhibit poor topical coherence. In contrast, *PartTreeSum* is flexible to generate more sentences and the *Combinablity* component in Equation 7 ensures that generated summary sentence contain topically related content. The original content is rephrased when required as observed in the second half of generated sentences. In Table 3, the summary sentences generated by *PartTreeSum* are more topically coherent compared to those generated by *PhraseAbSum*.

Error analysis : We have analysed the low-rated sentences from human evaluators with their corresponding set of partial trees. Though the partial trees contained information which can be combined in a single complex sentence, text aggregation during linearization should be more effective to improve the quality of sentences. For future work we plan to construct a neural generation model, which can aggregate and generate a sentence from a set of partial trees while maintaining factual accuracy with respect to the input documents. Also there should be a means to treat quotes separately apart from normal sentences.

5 Conclusion

We built a model for abstractive multi-document summarization by extracting partial dependency

trees to represent knowledge granules, and generating summary sentences using combinable granules utilizing syntactic linearization. Compared to existing methods for the task, our method has the advantages of generating new sequential sentential structures by rephrasing information if required as decided by the linearization model. On standard evaluation of using ROUGE metric and human evaluation for qualitative aspects of summary, this method showed competitive accuracies to the state-of-the-art methods for multi-document summarization.

Acknowledgements

We thank Ratish Puduppally, Zhiyang Teng and the three anonymous reviewers for conversations and feedback on earlier drafts. We thank Raghuram Vadapally and Faraaz Nadeem for their creative suggestions.

References

Miguel B Almeida and Andre FT Martins. 2013. Fast and robust compressive summarization with dual decomposition and multi-task learning. In *ACL (1)*. pages 196–206.

Siddhartha Banerjee, Prasenjit Mitra, and Kazunari Sugiyama. 2015. Multi-document abstractive summarization using ilp based multi-sentence compression. In *Proceedings of the 24th International Conference on Artificial Intelligence*. AAAI Press, pages 1208–1214.

Taylor Berg-Kirkpatrick, Dan Gillick, and Dan Klein. 2011. Jointly learning to extract and compress. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. Association for Computational Linguistics, pages 481–490.

Lidong Bing, Piji Li, Yi Liao, Wai Lam, Weiwei Guo, and Rebecca J Passonneau. 2015. Abstractive multi-document summarization via phrase selection and merging. *arXiv preprint arXiv:1506.01597* .

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. *arXiv preprint arXiv:1603.07252* .

Wei Li. 2015. Abstractive multi-document summarization with semantic information extraction. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. pages 1908–1913.

Chin-Yew Lin. 2003. Improving summarization performance by sentence compression: a pilot study. In *Proceedings of the sixth international workshop on Information retrieval with Asian languages-Volume 11*. Association for Computational Linguistics, pages 1–8.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out: Proceedings of the ACL-04 workshop*. Barcelona, Spain, volume 8.

Hui Lin and Jeff Bilmes. 2011. A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. Association for Computational Linguistics, pages 510–520.

Yijia Liu, Yue Zhang, Wanxiang Che, and Bing Qin. 2015. Transition-based syntactic linearization. In *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015*. pages 113–122. http://aclweb.org/anthology/N/N15/N15-1012.pdf.

André FT Martins and Noah A Smith. 2009. Summarization with a joint model for sentence extraction and compression. In *Proceedings of the Workshop on Integer Linear Programming for Natural Langauge Processing*. Association for Computational Linguistics, pages 1–9.

Hajime Morita, Ryohei Sasano, Hiroya Takamura, and Manabu Okumura. 2013. Subtree extractive summarization via submodular maximization. In *ACL (1)*. Citeseer, pages 1023–1032.

Ratish Puduppully, Yue Zhang, and Manish Shrivastava. 2016. Transition-based syntactic linearization with lookahead features. In *Proceedings of NAACL-HLT*. pages 488–493.

Hsin-Hao Su, Chin Lung Lu, and Chuan Yi Tang. 2008. An improved algorithm for finding a length-constrained maximum-density subtree in a tree. *Information Processing Letters* 109(2):161–164.

Hiroya Takamura and Manabu Okumura. 2009. Text summarization model based on maximum coverage problem and its variant. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 781–789.

Dingding Wang, Tao Li, Shenghuo Zhu, and Chris Ding. 2008. Multi-document summarization via sentence-level semantic analysis and symmetric matrix factorization. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, pages 307–314.

Kristian Woodsend and Mirella Lapata. 2010. Automatic generation of story highlights. In *Proceedings of the 48th Annual Meeting of the Association for*

Computational Linguistics. Association for Computational Linguistics, pages 565–574.

David M Zajic, Bonnie Dorr, Jimmy Lin, and Richard Schwartz. 2006. Sentence compression as a component of a multi-document summarization system. In *Proceedings of the 2006 Document Understanding Workshop, New York*.

Xingxing Zhang, Liang Lu, and Mirella Lapata. 2016. Top-down tree long short-term memory networks. *arXiv preprint arXiv:1511.00060* pages 0–5.

Yue Zhang. 2013. Partial-tree linearization: Generalized word ordering for text synthesis.