Ranking with Recursive Neural Networks and Its Application to Multi-document Summarization

Ziqiang Cao[1]* **Furu Wei**[2] **Li Dong**[3] **Sujian Li**[1] **Ming Zhou**[2]
[1]Key Laboratory of Computational Linguistics, Peking University, MOE, China
[2]Microsoft Research, Beijing, China
[3]Beihang University, Beijing, China
{ziqiangyeah, lisujian}@pku.edu.cn {furu,v-ldo, mingzhou}@microsoft.com

Abstract

We develop a Ranking framework upon Recursive Neural Networks (R2N2) to rank sentences for multi-document summarization. It formulates the sentence ranking task as a hierarchical regression process, which simultaneously measures the salience of a sentence and its constituents (e.g., phrases) in the parsing tree. This enables us to draw on word-level to sentence-level supervisions derived from reference summaries. In addition, recursive neural networks are used to automatically learn ranking features over the tree, with hand-crafted feature vectors of words as inputs. Hierarchical regressions are then conducted with learned features concatenating raw features. Ranking scores of sentences and words are utilized to effectively select informative and non-redundant sentences to generate summaries. Experiments on the DUC 2001, 2002 and 2004 multi-document summarization datasets show that R2N2 outperforms state-of-the-art extractive summarization approaches.

Introduction

Extractive summarization (Over and Yen 2004) aims to generate a short text summary for a document or a set of documents through selecting salient sentences in the document(s). Generally, there are two major components in extractive summarization systems, namely **sentence ranking** and **sentence selection**. The former gives an informative score to every sentence to measure its importance while the latter, built on both the salience scores and redundancy among sentences, chooses sentences to generate a summary.

Sentence ranking has been extensively investigated in extractive summarization. Different granularities of text (e.g., sentences or n-grams) are scored in the task. Some methods (Ouyang, Li, and Li 2007; Li et al. 2007) directly measure the salience of sentences while others (Gillick and Favre 2009; Li, Qian, and Liu 2013) firstly rank words (or bi-grams) and then combine these scores to rank sentences. Both words and sentences hold supervisions derived from reference summaries. However, it is not well-studied how to take full advantage of them, although more supervisions are expected to build a better model. In addition, regardless of sentence ranking models (Osborne 2002; Galley 2006; Conroy et al. 2004; Li et al. 2007), feature engineering is a necessary but labor-intensive task. It is thus desirable to let the model itself discover explanatory factors from data.

To this end, we develop a ranking framework upon recursive neural networks (R2N2) to rank sentences for multi-document summarization. It concerns the sentence ranking task as a hierarchical regression process, which evaluates the salience of all the non-terminal nodes in the parsing tree. This process is modeled by recursive neural networks (RNN). The learning ability of RNN has been proved in many NLP tasks, such as syntactical parsing (Socher, Manning, and Ng 2010), sentiment analysis (Socher et al. 2013) and discourse parsing (Li, Li, and Hovy 2014). With the guide of supervisions from the word level to the sentence level, RNN is able to learn ranking features automatically.

An example of this ranking framework is shown in Figure 1. In this parsing tree, recursive neural networks measure the importance of all these non-terminal nodes. Word-level and sentence-level ranking scores are obtained at pre-terminal and root nodes respectively. To be specific, R2N2 uses hand-crafted word features as inputs. In the forward propagation step, it recursively computes the representation of a parent node with the combination of its pair of children. Then regression is processed at all the non-terminal nodes, based on the learned features concatenating these hand-crafted ones. In the backward propagation step, weights of the model are updated with the guide of given supervisions over the tree. It is noted the scored parsing tree is similar to the sentiment classification treebank (Socher et al. 2013), where a sentiment label is attached to every non-terminal node. Nevertheless, (Socher et al. 2013) aims to extract the global semantic representation of each node whereas R2N2 tries to learn better local ranking features. We conduct extensive experiments on the DUC 2001, 2002 and 2004 multi-document summarization datasets. The experimental results demonstrate that R2N2 outperforms state-of-the-art extractive summarization approaches.

R2N2 makes three contributions to multi-document summarization:

- It transforms sentence ranking into a hierarchical regression task. Therefore more supervision knowledge can be taken into account;
- It is capable of automatically learning additional ranking

*Contribution during internship at Microsoft Research

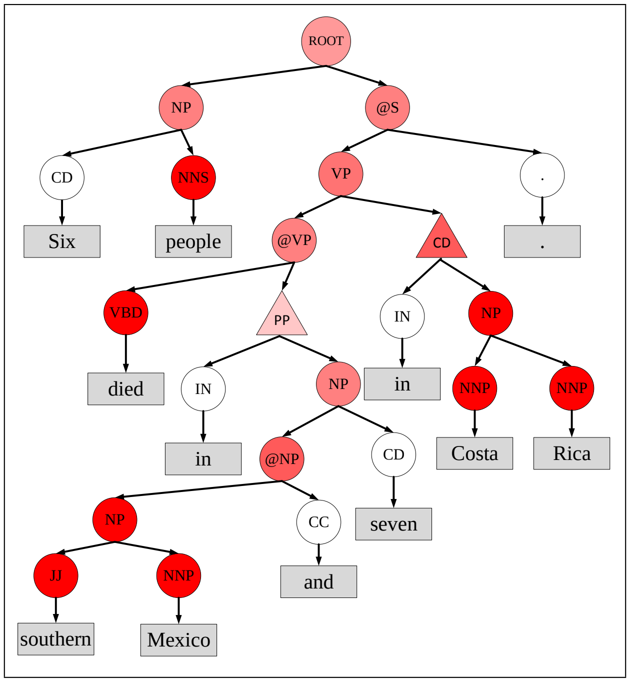

Figure 1: An example of R2N2 for the sentence "Six people *died* in *southern Mexico* and seven in *Costa Rica*". Leaf nodes in the gray rectangles stand for input words and the shade depth of a non-terminal node indicates its salience according to reference summaries. Circle means a nice regression result while Triangle means not.

features for a sentence as well as the constituents over the parsing tree;

- It provides consistent ranking scores from words to sentences, which makes more accurate sentence selection method possible (see Sentence Selection Section).

Related Work

Work on extractive generic summarization spans a large range of approaches. Starting with unsupervised methods, one of the widely known approaches is Maximum Marginal Relevance (MMR) (Carbonell and Goldstein 1998). It uses a greedy approach to select sentences and considers the tradeoff between relevance and redundancy. Good results can be achieved by reformulating this as an integer linear programming (ILP) problem which can find the optimal solution (McDonald 2007). Alternatively, concepts, usually represented by n-grams, can be regarded as textual units and the task changes to select a subset that covers most informative n-grams (Gillick and Favre 2009). Graph-based models play a leading role in the summarization area. LexRank (Erkan and Radev 2004) is a popular stochastic graph-based summarization approach, which computes sentence importance grounded on the concept of eigenvector centrality in a graph of sentence similarities. Graph-based methods have a lot of extensions. For example, (Wan and Yang 2008) pairs graph-based methods with clustering.

In contrast to these unsupervised approaches, there are also numerous efforts on supervised learning for summarization where a model is trained to predict the importance. Different classifiers have been explored for this task, such as maximum entropy (Osborne 2002), conditional random field (CRF) (Galley 2006) and hidden markov model (HMM) (Conroy et al. 2004). Besides these metrics which directly model sentences, many researches (Li, Qian, and Liu 2013; Hong and Nenkova 2014) focus on n-gram regression. Recently, treating multi-document summarization as a submodular maximization problem has attracted a lot of interest (Sipos, Shivaswamy, and Joachims 2012; Dasgupta, Kumar, and Ravi 2013).

To the best of our knowledge, (Hu and Wan 2013) is the only work that ranks both sentences and n-grams. They use support vector regression (SVR) to evaluate sentence importance while a simple unsupervised method is introduced to measure the salience of n-grams. However, our model deals with word regression and sentence regression simultaneously and consistently.

Sentence Ranking with RNN

Recursive Neural Network

The recursive neural network (RNN) has shown its power to model hierarchical concepts, such as syntactic parsing (Socher, Manning, and Ng 2010), discourse parsing (Li, Li, and Hovy 2014), sentiment analysis (Socher et al. 2013) and machine translation (Liu et al. 2014). Briefly, RNN processes structured inputs (usually a binary tree) by repeatedly applying the same neural network at each node. In such a setting, assume we have an input of vectors $\{v_j\}$ each of which has the same dimensionality $v_j \in \mathbb{R}^{1\times k_h}$. Then, suppose the binary tree is given in the form of branching triplets $(n \rightarrow c_1c_2)$. Each triplet denotes that a parent node n has two children and a child can be either an input vector or an internal node in the tree. Then we can compute representation for each node from the bottom up by:

$$l(n) = a_t([l(c_1), l(c_2)] \times W_t) \tag{1}$$

where $W_t \in \mathbb{R}^{2k_h\times k_h}$ is the transition matrix and a_t stands for the activation function. To be concise, we do not write out the bias item. $l(.) \in \mathbb{R}^{1\times k_h}$ is used to represent the vector for a node. Note that in order to replicate the neural network, all the nodes must have the same dimensionality (k_h) as input.

For the ranking problem, every non-terminal node n is associate with a salience score $s(n)$. In this case, the goal of RNN is twofold. First, it computes new representations over the tree according to Eq. 1. Second, it measures the importance of the node through a regression process:

$$\hat{s}(n) = a_r(l(n) \times W_{r1}) \tag{2}$$

where $W_{r1} \in \mathbb{R}^{k_h\times 1}$ is the regression matrix and a_r stands for the regression function. Finally, guided through the error from $\hat{s}(n)$ to $s(n)$, W_{r_1} and W_t can be updated with back propagate algorithm (McClelland et al. 1986).

Ranking Framework upon RNN (R2N2)

Pre-processing The input structure of our model is the parsing tree of a sentence. We use the Stanford CoreNLP (Manning et al. 2014) to parse and convert a sentence into a binary tree, as shown in Figure 1. The parsing tree provides a meaningful representation from words to the sentence. Then, the salience of each non-terminal node ($s(n)$) (word-level regression is conducted on the POS layer) is determined by ROUGE (Lin 2004), a widely-accepted automatic summarization evaluation metric. The variants ROUGE-1 and ROUGE-2 are reported to best emulate human evaluation (Owczarzak et al. 2012). Therefore we define the following rating strategy: For a pre-terminal node, because it is associated with a single word, we just use the ROUGE-1 (R_1) score. For an upper node, the combination of ROUGE-1 (R_1) and ROUGE-2 (R_2) scores is adopted to measure its importance. Let $N = \{n\}$ be the set of whole non-terminal nodes and N_w be the set of pre-terminal nodes. The scoring formula is:

$$s(n) = \begin{cases} R_1(n), n \in N_w \\ \alpha R_1(n) + (1-\alpha)R_2(n), n \in N - N_w \end{cases} \tag{3}$$

Here we empirically set the coefficient $\alpha = 0.5$. As for multiple references, we choose the maximal value. A salient parent node indicates its children are also salient, but not vice versa. Since ROUGE score ranges from 0 to 1, we choose sigmoid (σ) as the regression function ($a_r = \sigma$) and evaluate the result with cross entropy (CE):

$$\begin{aligned} &CE(s(n), \hat{s}(n)) = \\ &-(s(n) \ln \hat{s}(n) + (1 - s(n)) \ln(1 - \hat{s}(n))) \end{aligned} \tag{4}$$

Ranking with RNN We do a series of adjustments in R2N2 to make it fit for summarization.

- The input of R2N2 is the word features $f_w \in \mathbb{R}^{1 \times k_w}$ rather than automatically learned embeddings, unlike many previous applications to RNN (Socher, Manning, and Ng 2010; Socher et al. 2011; Li, Li, and Hovy 2014). For summarization, words hold different importance in different documents, which cannot be represented by a global word embedding.
- A projection layer is added to transform raw f_w into hidden features $f_h \in \mathbb{R}^{1 \times k_h}$.

  $$f_h = HT(f_w \times W_p) \tag{5}$$

  where HT, namely HardTanh, is the activation function:

  $$HT(x) = \begin{cases} 1, x \geq 1 \\ -1, x \leq -1 \\ x, otherwise \end{cases}$$

  , and $W_p \in \mathbb{R}^{k_w \times k_h}$ is the projection matrix. HT has the advantage of being cheap to compute while leaving good generalization performance (Collobert et al. 2011). This step makes our model exactly match the parsing tree structure. More importantly, it provides the chance to automatically learn summarization-specific features. Then f_h is taken as input of RNN and the state of a upper node ($l(n)$) is computed as Eq. 1. We also set $a_t = HT$.
- Initial word features f_w and sentence features $f_s \in \mathbb{R}^{1 \times k_s}$ are added into their own regression processes. In practice, we find raw features greatly improve the performance of learned features. This phenomenon is also found in the machine translation task (Lu, Chen, and Xu 2014). Maybe deep learning features strongly stand for high order correlations between the activities of the original features. As a result, we introduce extra two regression matrices $W_{r2} \in \mathbb{R}^{k_w \times 1}$ and $W_{r3} \in \mathbb{R}^{k_s \times 1}$ for word-level and sentence-level regression respectively. Then Eq. 2 splits into three cases:

  $$\hat{s}(n) = \begin{cases} \sigma(l(n) \times W_{r1} + f_w \times W_{r2}), n \in N_w \\ \sigma(l(n) \times W_{r1}), n \in N_i \\ \sigma(l(n) \times W_{r1} + f_s \times W_{r3}), n \in N_s \end{cases} \tag{6}$$

  where N_w is the set of pre-terminal (linked to words) nodes, N_s represents the set of root nodes (linked to sentences) and N_i the set of the rest internal nodes.

The whole model is shown in Fig 2. In the forward propagation step, word-level features f_w are first projected into hidden features f_h (Eq. 5). These hidden features are then used as inputs of RNN to compute all the upper node representation $l(n)$ (Eq. 1). Finally, different kinds of regression are conducted over the tree (Eq. 6). In the backward propagation step, weights W_p,W_t and $W_{r1\sim3}$ are updated with the guide of all the supervisions. Using CE to measure errors and HT as the activation function, this step can be computed as fast as linear models.

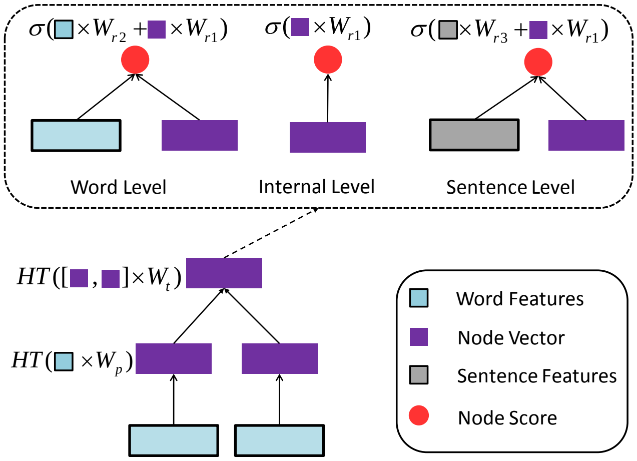

Figure 2: RNN for summarization model. A circle means a value while a rectangle stands for a vector. Different colors are used to discriminate feature types, and a black border indicates this type of features is hand-crafted. We use the dashed rectangle to denote the regression difference on three text levels.

Features for Sentence Ranking

The input of our model consists of two parts: word-level features f_w and sentence-level features f_s. Details of these features we use are listed in Table 1.

Feature	Description
TF	Term frequency over the cluster.
IDF	Total document number in the datasets, divided by the frequency of documents which contains this word.
CF	The frequency of documents which contains this word in the current cluster.
POS	A 4-dimension binary vector indicates whether the word is a noun, a verb, an adjective or an adverb. If the word has another part-of-speech, the vector is all-zero.
Named entity	A binary value equals one iff the output of the named entity classifier from CoreNLP is not empty.
Number	A binary value denotes whether the word is a number.
Sentence length	The maximal length of sentences owning the word.
Sentence TF	The maximal TF score of sentences owning the word.
Sentence CF	The maximal CF score of sentences owning the word.
Sentence IDF	The maximal IDF score of sentences owning the word.
Sentence sub-sentences	The maximal sub-sentence count of sentences owning the word. A sub-sentence means the node label is “S” or “@S” in the parsing tree.
Sentence depth	The maximal parsing tree depth of sentences owning the word.
Position	The position of the sentence. Supposing there are M sentences in the document, for the i_{th} sentence, the position feature is computed as $1-(i-1)/(M-1)$.
Length	The number of words in the sentence.
Sub-sentences	Sub-sentence count of the parsing tree.
Depth	The root depth of the parsing tree.
Averaged TF	The mean TF values of words in the sentence, divided by the sentence length.
Averaged IDF	The mean word IDF values in the sentence, divided by the sentence length.
Averaged CF	The mean word CF values in the sentence, divided by the sentence length.
POS ratio	The numbers of nouns, verbs, adjectives and adverbs in the sentence, divided by the sentence length respectively.
Named entity ratio	The number of named entities, divided by the sentence length.
Number ratio	The number of digits, divided by the sentence length.
Stopword ratio	The number of stopwords, divided by the sentence length. We just use the stopword list of ROUGE.

Table 1: Input Ranking Features

Double lines are used to distinguish f_w and f_s. The last 7 word-level features are borrowed from sentences where the word appears. Meanwhile, the last 10 sentence-level features are related to words in the sentence. In total, the dimensions of f_w (k_w) and f_s (k_s) are 16 and 14 respectively. All these features are scaled to $[0,1]$.

Sentence Selection

Above-mentioned, R2N2 provides ranking scores for both words (at pre-terminal nodes) and sentences (at root nodes). Since a summary is obliged to offer both informative and non-redundant content, we employ two widely-used sentence selection methods in this paper. One is the greedy algorithm (GA) (Li and Li 2014) which selects the most salient sentences with a similarity threshold. This metric just adopts regression results at root nodes. The other is integer linear programming (ILP) (Hu and Wan 2013) based sentence selection. ILP is intended to find the global optimum and combine the regression results of words and sentences.

Greedy based Sentence Selection In each step of selection, the sentence with maximal salience is added into the summary, unless its similarity with a sentence already in the summary exceeds a threshold. Here we use tf-idf cosine similarity and set the threshold $T_{sim} = 0.6$.

ILP based Sentence Selection Similar to previous work (McDonald 2007; Gillick and Favre 2009; Galanis, Lampouras, and Androutsopoulos 2012), we also consider the way of ILP to select sentences. Since sentences and words are scored simultaneously in our model, we define an objective function which combines the weights of sentences (WS) and words (WC). Redundancy is implicitly measured by benefiting from including each word only once.

$$\begin{aligned}
\max: \quad & \lambda \sum_{i \in N_s} l_i WS(i) x_i + (1-\lambda) \sum_{j \in N_w} WC(j) c_j \\
s.t.: \quad & \sum_i l_i x_i \le L \\
& x_i O_{i,j} \le c_i, \quad \forall i,j \\
& \sum_i x_i O_{i,j} \ge c_j, \quad \forall j \\
& x_i \in \{0,1\}, \quad \forall i \\
& c_j \in \{0,1\}, \quad \forall j
\end{aligned} \tag{7}$$

where x_i indicates whether sentence i is in the summary and c_j stands for the appearance of word j. $O_{i,j}$ points out the occurrence of the word j in sentence i. We use l_i to represent the length of sentence i. In the objective function, l_i penalizes short sentences. L means the length limitation while the parameter λ indicates the trade-off between sentence and word weights. Because words and sentences are regressed consistently in R2N2, we set $\lambda = 0.5$. Note if $\lambda = 0$, this function is similar to the system of (Gillick and Favre 2009) except that our model is supervised and we use words rather than bi-grams to represent concepts. Although ILP is NPhard in general, considerable optimization research has produced software for solving instances efficiently, and we use IBM CPLEX Optimizer [1] in this paper.

Experiments

Datasets

The most commonly used evaluation corpora for summarization are the ones published by the Document Understanding Conferences (DUC[2]) and Text Analytics Conferences (TAC[3]). Here we focus on the generic multi-document summarization task, which was carried out in DUC 2001, 2002 and 2004. The documents are all from the news domain and are grouped into various thematic clusters. Table 2 shows the size of the three datasets and the maximum length of summaries for each task. We train the model on two years' data and test it on the other year.

Year	Clusters	Sentences	Length Limit
2001	30	11295	100 words
2002	59	15878	100 words
2004	50	13070	665 bytes

Table 2: Statistics on the DUC datasets.

Evaluation Metric

For the evaluation, we use ROUGE[4] (Lin 2004). It has grown up to be a standard automatic evaluation metric for DUC since 2004. The parameter of length constraint is "-l 100" for DUC 2001/2002, and "-b 665" for DUC 2004. We take ROUGE-2 recall as the main metric for comparison due to its high capability of evaluating automatic summarization systems (Owczarzak et al. 2012).

Baseline Methods

We consider three support machine regression[5] baselines: unigram regression with GA selection (Ur), sentence regression with GA selection (Sr) and the ILP method where WS and WC are measured by related regressions (U+Sr). All these regression baselines adopt the same features and selection parameters as R2N2. These three baselines are used to examine whether our model is superior to the traditional way of regression. Meanwhile, LexRank (Erkan and Radev 2004), a common graph-based summarization model, is introduced as an extra baseline. We adopt this baseline to display the performance of regression approaches based on these features.

In addition, we list the best participates and results published for these corpora. The best systems we find are ClusterHITS (Wan and Yang 2008) for DUC 2001, ClusterCMRW (Wan and Yang 2008) for 2002 and REGSUM[6] (Hong and Nenkova 2014) for DUC 2004. Brief introduction of these methods can be found in Related Work.

Model Configuration

We set the dimension of RNN (k_h) to 8, which is a half of the input layer dimension. We use mini-batch gradient descent with L2-norm regularization to update weights. The learning rate is 0.005 and regularization factor is 0.1. We set the batch size to 100. The training process with 100 iterations spends about 30 minutes.

Results and Discussion

In this section, we present the results of our models and compare them with other systems, mainly measured by ROUGE-2. In Table 3, each row represents one summarization system and each column describes one evaluation metric. We cite the scores of others from their papers, indicated with the sign "*". Original DUC results are named Peer plus their system ID. R2N2_ILP and R2N2_GA differ in their sentence selection metrics.

Year	System	ROUGE-1	ROUGE-2
2001	Peer T	33.06	**8.06**
	ClusterHITS*	**37.42**	6.81
	LexRank	33.22	5.76
	Ur	34.28	6.66
	Sr	34.06	6.65
	U+Sr	33.98	6.54
	R2N2_GA	35.88	7.64
	R2N2_ILP	36.91	7.87
2002	Peer 26	35.15	7.64
	ClusterCMRW*	**38.55**	8.65
	LexRank	35.09	7.51
	Ur	34.16	7.66
	Sr	34.23	7.81
	U+Sr	35.13	8.02
	R2N2_GA	36.84	8.52
	R2N2_ILP	37.96	**8.88**
2004	Peer 65	37.88	9.18
	REGSUM*	38.57	9.75
	LexRank	37.92	8.78
	Ur	37.22	9.15
	Sr	36.72	9.10
	U+Sr	37.62	9.31
	R2N2_GA	38.16	9.52
	R2N2_ILP	**38.78**	**9.86**

Table 3: Comparison results (%) on DUC datasets.

Seen from this table, R2N2 significantly[7] outperforms the traditional regression methods all the time. However, there is no significant difference between the sentence selection methods ILP and GA. For R2N2, ILP is always superior to

[1] http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/

[2] http://duc.nist.gov/ from 2001 ∼ 2007

[3] http://www.nist.gov/tac/ from 2007 ∼ now

[4] ROUGE-1.5.5 with options: -n 2 -m -u -c 95 -x -r 1000 -f A -p 0.5 -t 0

[5] We use LIBLINEAR. http://www.csie.ntu.edu.tw/~cjlin/liblinear/

[6] REGSUM truncates a summary to 100 words.

[7] T-test with p-value=0.05

GA, while jointing the regression results of words and sentences sometimes even produces a worse solution. Maybe it is because ranking of Ur and Sr can seriously conflict. Compared with other systems, R2N2 achieves the best ROUGE-2 scores on DUC 2002 and 2004. As for DUC 2001, the result of Peer T is somewhat strange. Its ROUGE-1 value is the lowest, but ROUGE-2 just the opposite. Except Peer T, R2N2 greatly promotes ROUGE-2. Models of (Wan and Yang 2008) carry out a bit larger ROUGE-1 score than R2N2. However, no proposed model in their paper can always outperform the others on different datasets. For example, ClusterCMRW works well on DUC 2002 but it only gets 35.71 (ROUGE-1 value) and 6.55 (ROUGE-2 value) on DUC 2001. In contrast, R2N2 works quite stably.

Note REGSUM is a kind of unigram regression. It achieves good performance through using larger training data and importing complex features. However, R2N2 only adopts basic sentence and word features. R2N2 exceeds REGSUM on both ROUGE-1 and ROUGE-2 metrics, indicating its ability to automatically learn ranking features.

Impact of Raw Feature Supplement Here we verify the model performance of adding raw features. Take DUC 2004 as an example. The result is listed in Table 4. Note the model performance heavily depends on these raw features, especially on the sentence level. Meanwhile, the previous section shows that R2N2 is prior to direct unigram or sentence regression with identical features. Consequently, learned features seem to serve as favorable supplements of raw ones.

Features	ROUGE-1	ROUGE-2
None	35.44	6.97
+U	36.08	7.22
+S	38.05	9.36
+Both	38.78	9.86

Table 4: DUC 2004 Performance (%) w/o additional raw features. T-test shows all the changes are significant.

Impact of Lambda In previous experiments, λ, the ILP parameter, is fixed to 0.5 since we want the model to be biased toward neither word regression nor sentence regression. In this section, we test how λ affects the performance. Let λ increase from 0.1 to 0.9. The corresponding changes of ROUGE-2 are shown in Figure 3. As can be seen, the performance of DUC 2001 decreases with λ while for DUC 2004 it is somewhat positively correlated. On the whole, the model performs well when treating word level and sentence level regressions equally. Although it looks a bit better to set $\lambda = 0.4$, for generalization we choose 0.5.

Regression Error Analysis This section aims to check the error distribution of different text granularities. Since the range of scores changes along with the tree depth, here we adopt relative standard deviation (rsd) to measure errors:

$$rsd = \frac{\sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(s(i)-\hat{s}(i)\right)^{2}}}{\frac{1}{N}\sum_{i=1}^{N}s(i)}$$

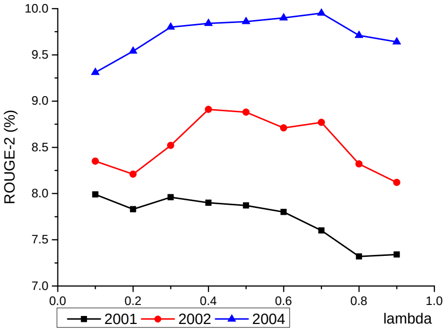

Figure 3: ROUGE-2 change with λ.

Ur and Sr errors are likewise computed as reference. The result is listed in Table 5. In general, the error shows a negative correlation with the text length. But the word level error nearly equals the internal level error. One reason is that the supplement of raw features works. Compared with Ur and Sr, R2N2 makes much more accurate prediction.

Year	Method	Word	Intra	Sentence
2001	R2N2	1.07	1.00	0.57
	Ur	1.30	-	-
	Sr	-	-	1.19
2002	R2N2	1.10	1.05	0.60
	Ur	1.39	-	-
	Sr	-	-	1.65
2004	R2N2	0.88	0.85	0.49
	Ur	1.04	-	-
	Sr	-	-	0.88

Table 5: Error distribution of different tree levels

Conclusion and Future Work

This paper presents a Ranking framework upon Recursive Neural Networks (R2N2) for the sentence ranking task of multi-document summarization. It transforms the ranking task into a hierarchical regression process which is modeled by recursive neural networks. Learned features are capable of supplementing hand-crafted features to rank sentences. In addition, based on the ranking scores of both words and sentences, we design a optimized sentence selection method. We conduct experiments on DUC benchmarks. Our model achieves higher ROUGE scores than state-of-the-art summarization approaches. Meanwhile, the error analysis shows that our model makes much more accurate predication than traditional support vector regression.

We believe R2N2 can be advanced and extended from many different perspectives. First, we are interested in applying R2N2 for query-focused summarization, which can be achieved by introducing query-related features to the input of the neural networks. Second, as the by-product of R2N2, we can get the scores for the internal nodes (i.e., clauses and phrases) in the parsing trees. This information will be very useful for many other summary-related tasks, such as sentence compression and keyphrase extraction.

Acknowledgments

We thank all the anonymous reviewers for their helpful comments. This work was partially supported by National High Technology Research and Development Program of China (No. 2012AA011101), National Key Basic Research Program of China (No. 2014CB340504), National Natural Science Foundation of China (No. 61273278), and National Key Technology R&D Program (No: 2011BAH10B04-03). The correspondence author of this paper is Sujian Li.

References

Carbonell, J., and Goldstein, J. 1998. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of SIGIR*, 335–336.

Collobert, R.; Weston, J.; Bottou, L.; Karlen, M.; Kavukcuoglu, K.; and Kuksa, P. 2011. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research* 12:2493–2537.

Conroy, J. M.; Schlesinger, J. D.; Goldstein, J.; and Oleary, D. P. 2004. Left-brain/right-brain multi-document summarization. In *Proceedings of DUC*.

Dasgupta, A.; Kumar, R.; and Ravi, S. 2013. Summarization through submodularity and dispersion. In *Proceedings of ACL*, 1014–1022.

Erkan, G., and Radev, D. R. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *J. Artif. Intell. Res.(JAIR)* 22(1):457–479.

Galanis, D.; Lampouras, G.; and Androutsopoulos, I. 2012. Extractive multi-document summarization with integer linear programming and support vector regression. In *Proceedings of COLING*, 911–926.

Galley, M. 2006. A skip-chain conditional random field for ranking meeting utterances by importance. In *Proceedings of EMNLP*, 364–372.

Gillick, D., and Favre, B. 2009. A scalable global model for summarization. In *Proceedings of the Workshop on ILP for NLP*, 10–18.

Hong, K., and Nenkova, A. 2014. Improving the estimation of word importance for news multi-document summarization. In *Proceedings of EACL*.

Hu, Y., and Wan, X. 2013. Ppsgen: learning to generate presentation slides for academic papers. In *Proceedings of IJCAI*, 2099–2105.

Li, Y., and Li, S. 2014. Query-focused multi-document summarization: Combining a topic model with graph-based semi-supervised learning. In *Proceedings of COLING*, 1197–1207.

Li, S.; Ouyang, Y.; Wang, W.; and Sun, B. 2007. Multi-document summarization using support vector regression. In *Proceedings of DUC*.

Li, J.; Li, R.; and Hovy, E. 2014. Recursive deep models for discourse parsing. In *Proceedings of EMNLP*.

Li, C.; Qian, X.; and Liu, Y. 2013. Using supervised bigram-based ilp for extractive summarization. In *Proceedings of ACL*, 1004–1013.

Lin, C.-Y. 2004. Rouge: A package for automatic evaluation of summaries. In *Proceedings of the ACL Workshop*, 74–81.

Liu, S.; Yang, N.; Li, M.; and Zhou, M. 2014. A recursive recurrent neural network for statistical machine translation. In *Proceedings of ACL*, 1491–1500.

Lu, S.; Chen, Z.; and Xu, B. 2014. Learning new semi-supervised deep auto-encoder features for statistical machine translation. In *Proceedings of ACL*, 122–132.

Manning, C. D.; Surdeanu, M.; Bauer, J.; Finkel, J.; Bethard, S. J.; and McClosky, D. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of ACL*, 55–60.

McClelland, J. L.; Rumelhart, D. E.; Group, P. R.; et al. 1986. Parallel distributed processing. *Explorations in the microstructure of cognition* 2.

McDonald, R. 2007. *A study of global inference algorithms in multi-document summarization*. Springer.

Osborne, M. 2002. Using maximum entropy for sentence extraction. In *Proceedings of ACL Workshop on Automatic Summarization*, 1–8.

Ouyang, Y.; Li, S.; and Li, W. 2007. Developing learning strategies for topic-based summarization. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, 79–86.

Over, P., and Yen, J. 2004. Introduction to duc-2001: an intrinsic evaluation of generic news text summarization systems. In *Proceedings of DUC*.

Owczarzak, K.; Conroy, J. M.; Dang, H. T.; and Nenkova, A. 2012. An assessment of the accuracy of automatic evaluation in summarization. In *Proceedings of Workshop on Evaluation Metrics and System Comparison for Automatic Summarization*, 1–9.

Sipos, R.; Shivaswamy, P.; and Joachims, T. 2012. Large-margin learning of submodular summarization models. In *Proceedings of the 13th conference of the European chapter of ACL*, 224–233.

Socher, R.; Lin, C. C.; Manning, C.; and Ng, A. Y. 2011. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of ICML*, 129–136.

Socher, R.; Perelygin, A.; Wu, J. Y.; Chuang, J.; Manning, C. D.; Ng, A. Y.; and Potts, C. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of EMNLP*, 1631–1642.

Socher, R.; Manning, C. D.; and Ng, A. Y. 2010. Learning continuous phrase representations and syntactic parsing with recursive neural networks. In *Proceedings of NIPS Workshop*, 1–9.

Wan, X., and Yang, J. 2008. Multi-document summarization using cluster-based link analysis. In *Proceedings of SIGIR*, 299–306.