A Query-Focused Summarization Method that Guarantees the Inclusion of Query Words

Norihito Yasuda, Masaaki Nishino, Tsutomu Hirao, Jun Suzuki and Ryoji Kataoka

NTT Corporation

What is the Query-Focused Summarization?

- A variant of automatic text summarization, which reflects the given query.
- used for
  - search result snippet
  - support summaries for answers in question-answering systems
  - and so on
- usually based on sentences’ score and relevance score with query.

Automatic summarization as a optimization problem

Recently (extractive) automatic summarizations are formalized as an optimization problem.

- instead of greedy selecting the highest score sentences.

Automatic summarization as a optimization probolem

sentence ID	score	# of chars.
1	0.8	35
2	0.7	20
3	0.9	17
4	0.6	48
5	0.5	19

sentences that gives max score <= 40 chars

select 2, 3

this can be assumed as 0-1 Knapsack Problem

Problem with score based methods.

score = sentence importance score ＋ relevance score with the query

- A resulting summary may not contain any word in the query.
  - may possible to reduce the probability by the weight of relevance score.
  - Essentially we cannot avoid that.
- Crucial information especially for support summary of question-answering.
  - also import for web snippets.

Adding New Constraint to Objective

vector representing the selected sentences.

$$\mathbf{y}^* = \underset{\mathbf{y}}{\operatorname{argmax}}\, f(\mathbf{y}, q) = \sum_{i=1}^{N} w_i(q) y_i$$

objective function

$$\text{subject to } \sum_{i=1}^{N} l_i y_i \leq L_{\max}$$

length constraint

$$\boxed{\sum_{i=1}^{N} c_q(y_i) \geq 1}$$

proposed constraint that assures inclusion of query terms

number of sentences that includes words in the query

Problem with new formalization

- By adding the constraint we can assure the inclusion of at least one word of the query.
- However, the new form problem is not a 0-1 knapsack problem.

  (reason) the function is not a linear function of **y.**

Introducing Lagrangian Relaxation

Original problem:

$$\mathbf{y}^{*} = \underset{\mathbf{y}}{\text{argmax}}\, f(\mathbf{y}, q) = \overset{N}{\underset{i=1}{\text{å}}} w_i(q) y_i$$

objective function

$$\text{subject to } \overset{N}{\underset{i=1}{\text{å}}} l_i y_i \text{ £ } L_{\max}$$

length constraint

$$\boxed{\overset{N}{\underset{i=1}{\text{å}}} c_q(y_i) \text{ ³ } 1}$$

propoosed constraint that assures inclusion of query terms

It's Lagrangian Relaxation

Lagrange multipliers

$$L(u,\mathbf{y}) = f(\mathbf{y},q) + u\left(\sum_{i=1}^{N} c_q(y_i) - 1\right)$$

Add constraint to the objective function.

$$\text{subject to } \sum_{i=1}^{N} l_i y_i \le L_{\max}$$

Now *L()* is the linear function of **y**

▷ can be maximized as a knapsack problem

Lagrangian Dual Problem

Lagrangian

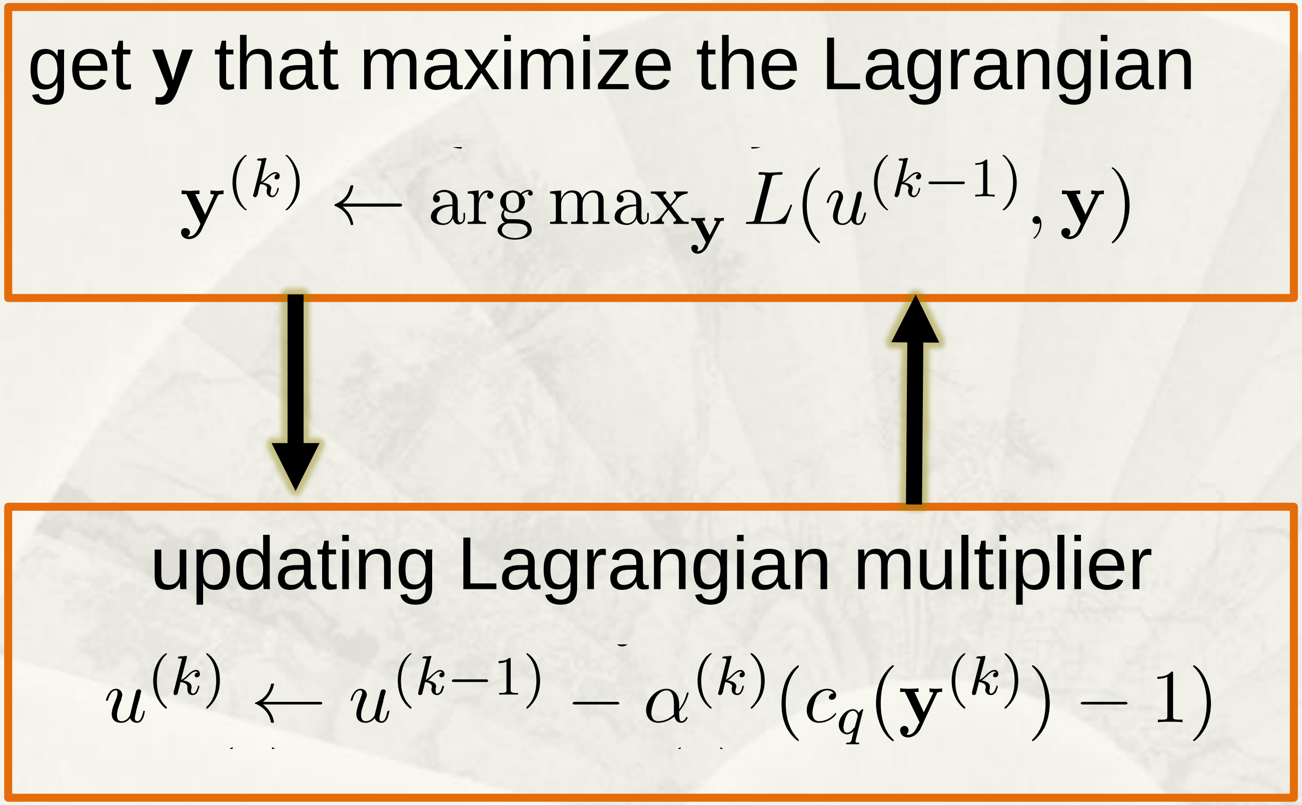
$$L(u) = \max_{\mathbf{y}} L(u, \mathbf{y})$$

Lagrangian dual problem

$$\min_{u} L(u)$$

by using subgradient method we can get the tightest upper bound of the exact solution of the original problem.

Solving process



get **y** that maximize the Lagrangian

$$\mathbf{y}^{(k)} \leftarrow \arg\max_{\mathbf{y}} L(u^{(k-1)}, \mathbf{y})$$

can solve efficiently

updating Lagrangian multiplier

$$u^{(k)} \leftarrow u^{(k-1)} - \alpha^{(k)}(c_q(\mathbf{y}^{(k)}) - 1)$$

using subgradient method

Summaries so far

- introduced a new constraint to summarization
  - at least one word of the query must be contained.
- by exploiting Lagrangian relaxation, the problem can be solved by iteration of knapsack problem.

One word → n word

- For longer queries, we want summaries containing more keywords than one.
- extend the constraint to contain at least any n (content) words in the query.

Naïve Formulation

``Who made the first airplane that could fly?”

content words

{make, first, airplane, fly}

straight-forward write down of the condition:

$$s(c_{\text{make}}(\mathbf{y}))+s(c_{\text{first}}(\mathbf{y}))+s(c_{\text{airplane}}(\mathbf{y}))+s(c_{\text{fly}}(\mathbf{y}))\geq 2$$

$c_{\text{make}}(\mathbf{y})$: number of sentences that includes ``make”

$$s(x)=\begin{cases}1 & (x>0)\\ 0 & (x=0)\end{cases}$$

$\mathbf{y}$: vector representing the selected sentences.

This cannot be solved as a knapsack problem ☹

Formalize by Linear Function

Contain n words from a set of Q words.

can be expressed by $_{Q}C_{Q-n+1}$ constraints of linear function

It’s practical in case m is small.

{make, first, airplane, fly}

$$c_{\text{make}}(\mathbf{y}) + c_{\text{first}}(\mathbf{y})\ ^{3}\ 1$$

$$c_{\text{make}}(\mathbf{y}) + c_{\text{airplane}}(\mathbf{y})\ ^{3}\ 1$$

$$c_{\text{make}}(\mathbf{y}) + c_{\text{fly}}(\mathbf{y})\ ^{3}\ 1$$

$$c_{\text{first}}(\mathbf{y}) + c_{\text{airplane}}(\mathbf{y})\ ^{3}\ 1$$

$$c_{\text{first}}(\mathbf{y}) + c_{\text{fly}}(\mathbf{y})\ ^{3}\ 1$$

$$c_{\text{airplane}}(\mathbf{y}) + c_{\text{fly}}(\mathbf{y})\ ^{3}\ 1$$

(Additinonal usage)
Constraint by NE type

In case the query is a question and we can determine the question type.

▼

the summary should contain a named entity (NE) that matches the request type.

NE type constraint example

question	question type	words that matches the NE type
``Who made the first airplane that could fly?”	**WHO**	{The president, Charles Lindbergh, Scott Lindbergh, Raymond Orteig, …}
``When was George Foreman born?”	**WHEN**	{July, Sunday, Friday, August 1949, 1951, 1970, …}

▼

add constraint that contain at least one of this set.

Evaluation

Dataset

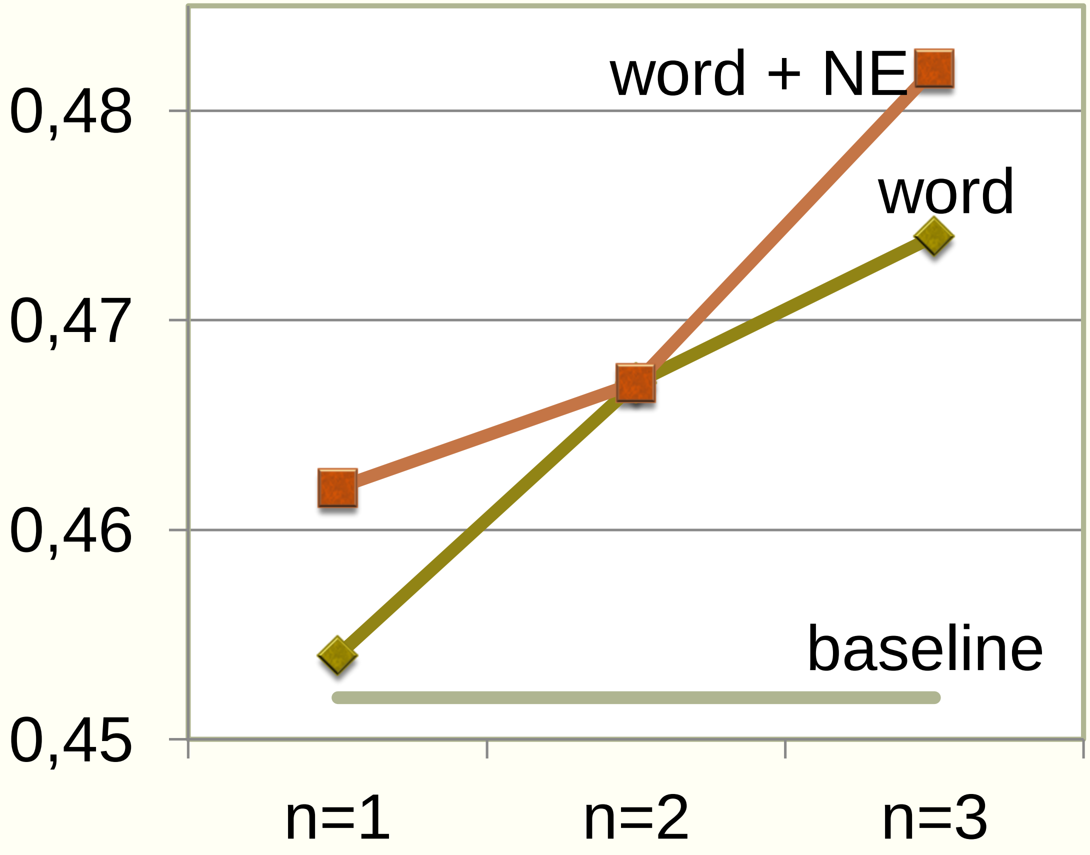
- Text Summarization Challenge 3 (TSC3)
  - A dataset for query-focused multi-document summarization on Japanese news-wire.
  - consists of documents, questions and reference summaries produced by humans.
  - References are made so as to supply the answer to the given question.
  - 30 topics.

Evaluation Settings

- evaluated using average ROUGE socres over the 30 topics.
  - ROUGE: a standard method to evaluate automatic summarization.
- Baseline: no constraints on inclusion of query terms.
- Constraints in our method: at least n (=1,2,3) content words of question.

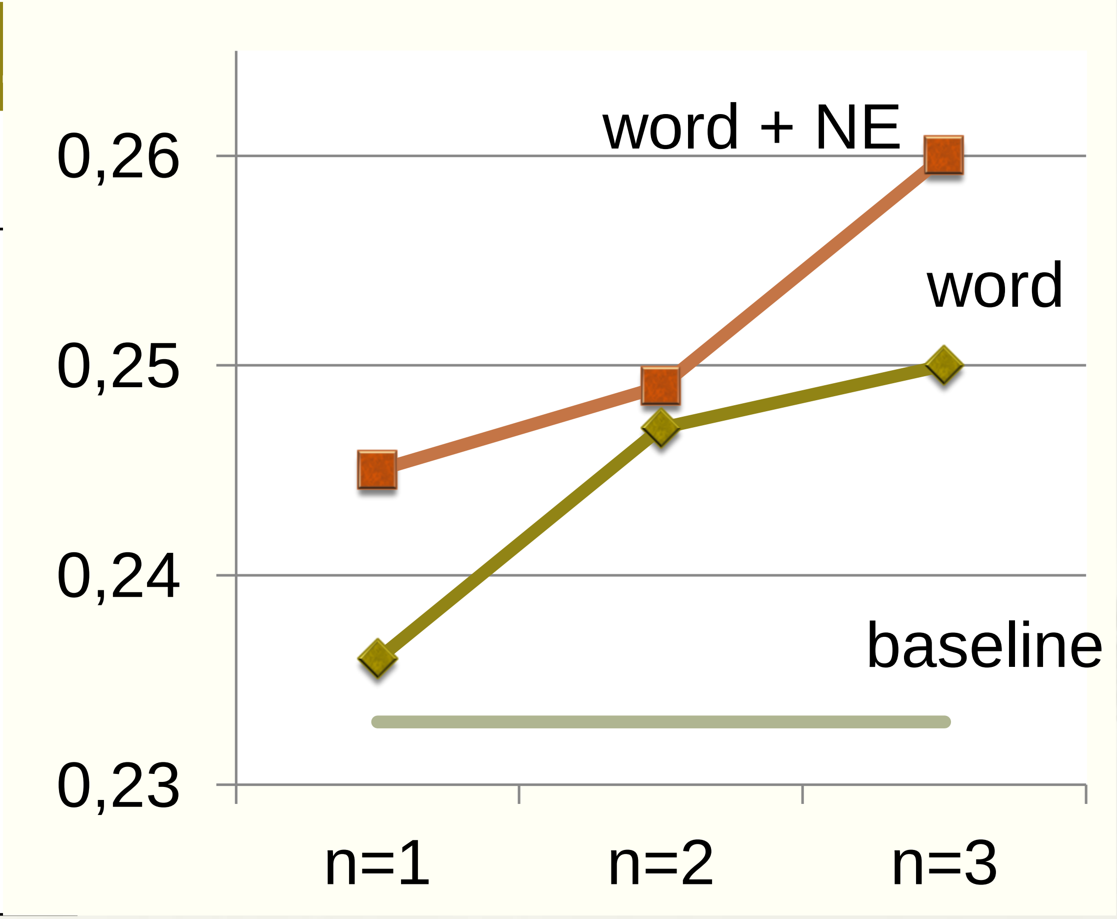
Evaluation Result (ROUGE-1)

Method	ROUGE-1
baseline	0.452
n=1	0.454
n=2	0.467
n=3	0.474
n=1+NE	0.462
n=2+NE	0.467
n=3+NE	0.482



Evaluation Result (ROUGE-2)

Method	ROUGE-
baseline	0.233
n=1	0.236
n=2	0.247
n=3	0.250
n=1+NE	0.245
n=2+NE	0.249
n=3+NE	0.260



Discussions

- All proposed settings significantly improve ROUGE score.
  - The reference summary is intended to support answer and tend to contain many words in the question.
- Score increases with n.
  - (open) How to know the optimal n?
- By adding NE constraint, the scores are further imporeved
  - But the difference is not significant.

Summary

- Inroduced a new constraint into query biased summarization that
- Lagrangian relaxation brings us fast solve
  - using DP + updating parameter
- Easily expandable to handle NE type

Thank you!
Arigato.