Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI 2015)

Compressive Document Summarization via Sparse Optimization

Jin-ge Yao, Xiaojun Wan,[*] Jianguo Xiao
Institute of Computer Science and Technology, Peking University, Beijing 100871, China
Key Laboratory of Computational Linguistic (Peking University), MOE, China
{yaojinge, wanxiaojun, xiaojianguo}@pku.edu.cn

Abstract

In this paper, we formulate a sparse optimization framework for extractive document summarization. The proposed framework has a decomposable convex objective function. We derive an efficient ADMM algorithm to solve it. To encourage diversity in the summaries, we explicitly introduce an additional sentence dissimilarity term in the optimization framework. We achieve significant improvement over previous related work under similar data reconstruction framework. We then generalize our formulation to the case of compressive summarization and derive a block coordinate descent algorithm to optimize the objective function. Performance on DUC 2006 and DUC 2007 datasets shows that our compressive summarization results are competitive against the state-of-the-art results while maintaining reasonable readability.

1 Introduction

Automatic document summarization is a seminal task in the field of text mining and information retrieval. Approaches of modern automatic summarization can be roughly divided into two categories: extractive summarization and abstractive summarization. Extractive summarization formulates summaries by selecting sentences from the original documents. Abstractive approaches allow more complex operations on sentences, including deletion, substitution, and reordering. Extractive approaches are rather limited in terms of the final summaries they can produce. However, abstractive summarization is far more difficult than extractive summarization that does not need to ensure structural and semantic coherence within a sentence. Up to now, extractive methods are still the most popular in summarization tasks.

Almost all the existing extractive summarization approaches use ranking models to score and select sentences. To overcome the problem of redundancy that has not been well-addressed in previous approaches, summarization based on data reconstruction has recently been proposed [He *et al.*, 2012]. The intuition is that a good summary should consist of those sentences that can best reconstruct the original document. Unfortunately, although the mathematical idea behind this framework is reasonable, it fails to achieve satisfactory performance. Meanwhile, the accompanying gradient descent algorithm turns out to be slow in practice, which may further limit the generalizability of this paradigm.

In this paper, we formulate document summarization problem as convex sparse optimization with similar idea of data reconstruction but different methodology. The proposed objective function has a decomposable form, which is suitable to be efficiently solved by modern convex optimization algorithms.

With some direct but nontrivial modifications, we can also generalize the proposed formulation to the case of compressive summarization. Under the similar essence of sparse optimization, we explore a framework that jointly optimizes sentence selection and word selection. Grammaticality of compressed sentences is ensured by considering dependency relations during a final generation step. Our methods do not even need full constituent parse trees to generate grammatical compressions.

The major contributions of this work include:

- We formulate document summarization as a decomposable row-sparsity regularized optimization problem, and then present an efficient alternating direction method of multipliers to solve it.
- Inspired by recent research on exemplar-based data representation, we introduce an additional sentence dissimilarity term to encourage diversity in summary sentences.
- We generalize our proposed extractive framework to compressive summarization. The resulting sparse optimization problem is jointly non-convex, so we derive a block coordinate descent algorithm to solve it, followed by a recursive sentence compression phase to impose grammatical constraints. Merely an additional lightweight dependency parser is needed to conduct this step and less efficient constituent parsing is avoided.
- We conduct experiments on DUC 2006 and DUC 2007 datasets to show the improvements of our methods over previous work of unsupervised document summarization based on data reconstruction. Our models achieve fairly competitive results against the state-of-the-art approaches while maintaining reasonable readability.

[*] Corresponding author

2 Our Proposed Framework

2.1 Sentence Selection via $\ell_{2,1}$ Regularization

Assume that the documents are represented as a weighted term-frequency matrix, denoted as $D = [D_1, D_2, ..., D_n] \in \mathbb{R}^{d \times n}$ [1], where d is the size of vocabulary and n is the total number of sentences. Each column $D_i \in \mathbb{R}^{d \times 1}$ stands for a sentence vector.

We are aiming at selecting sentences having both good coverage and low redundancy. Once we have a summary set of sentences taken from the original document, we would like to represent most sentences in the original document well, only with these summary sentences. Therefore a natural goal is to minimize

$$\sum_{j=1}^{n} \|D_j - DA_j\|_2^2,$$

where each $A_j = [a_{1j}, ..., a_{nj}]^T$ is a vector of coefficients used for representing the original sentence vector D_j with other sentence vectors from the document D. This is similar to the intuition behind He et al. [2012]'s formulation from a data reconstruction perspective.

We are trying to represent the whole document by selecting only a few sentences. Then the goal is to establish sentence level sparsity on sentence selection coefficients (the A_is), while retaining low reconstruction error. If we concatenate these column vectors as a matrix $A = \{A_1, ..., A_n\} \in \mathbb{R}^{n \times n}$, the sentence level sparsity will obviously correspond to row sparsity in A. Hence the desired selection sparseness could be induced by performing ℓ_1-regularization on rows, or more compactly, an $\ell_{2,1}$ regularization on matrix A:

$$\|A\|_{2,1} = \sum_{i=1}^{n} \|A_i\|_2$$

The document summarization problem can now be succinctly formulated as:

$$\min_{A} \quad \|D - DA\|^2 + \lambda \|A\|_{2,1} \tag{1}$$

$$s.t. \quad a_{ij} \geq 0, \ \forall i, j \tag{2}$$

$$diag(A) = 0. \tag{3}$$

Here an additional constraint (3) is introduced to avoid the numerically trivial solution ($A \approx I$) in practice, by forcing the diagonal elements to be zeros. This formulation resembles the problem of subspace clustering that has been studied in the community of computer vision [Elhamifar and Vidal, 2013], trying to learn a self-representation matrix under certain constraints [2].

Meanwhile, our convex objective function can be decomposed into two separated convex functions. In recent years, much attention has been paid upon this type of decomposable objectives, especially in the context of sparse optimization. In the next section we present an alternating direction method of multipliers (ADMM) for our optimization problem.

[1]In this paper we follow the convention to use uppercase letters to denote matrices. We place a subscript k when addressing the k-th column. Corresponding lowercase letters are used for elements (scalars) in matrices/vectors.

[2]The major difference between (1) and sparse subspace clustering is in the regularization terms.

2.2 An ADMM Solver

Problem (1) can be equivalently expressed as:

$$\min_{X,Z} \quad \|D - DX\|^2 + \lambda \|Z\|_{2,1} \tag{4}$$

$$s.t. \quad X = Z, diag(X) = 0, x_{ij} \geq 0, \ \forall i, j. \tag{5}$$

By expressing the original A separately as X and Z, the objective function has been factorized into two independently convex parts. ADMM tries to solve such decomposed problems by iteratively updating X and Z, as well as Lagrangian multipliers (denoted as a matrix U) on the constraint $X = Z$. We omit details of our mathematical derivations due to space limit and point interested readers to Section 6.4 of Boyd et al. [2011], where a similar problem is presented.

The entire ADMM algorithm is listed in Algorithm 1.

Algorithm 1 An ADMM solver for Problem (1)

```
1: for $k = 0, 1, 2, ...$ do
2:     // update every column $X_j$ of $X$:
3:     $X_j^{k+1} \leftarrow (D^T D + \rho I)^{-1}(D^T D_j + \rho(Z_j^k - U_j^k))$
4:     $X^{k+1} \leftarrow X^{k+1} - diag(X^{k+1}), x_{ij}^{k+1} \leftarrow \max\{x_{ij}^{k+1}, 0\}$
5:     // update every row $Z_i$ of $Z$:
6:     $Z_i^{k+1} \leftarrow \mathcal{S}_{\lambda/\rho}(X_i^{k+1} + U_i^k)$
7:     $Z^{k+1} \leftarrow Z^{k+1} - diag(Z^{k+1}), z_{ij}^{k+1} \leftarrow \max\{z_{ij}^{k+1}, 0\}$
8:     // update dual variables $U$:
9:     $U^{k+1} \leftarrow U^k + \rho(X^{k+1} - Z^{k+1})$
10:    if $\|X^{k+1} - Z^{k+1}\| < \epsilon$ then return $Z$
11:    end if
12: end for
```

where ρ is a constant parameter and the shrinkage operator $\mathcal{S}$ in Line 6 is defined as:

$$\mathcal{S}_\gamma(x) = \max\{1 - \frac{\gamma}{\|x\|_2}, 0\}x.$$

Note that the implementation of the $X-$update step (Line 3 in Algorithm 1) does not require matrix inversion. It suffices to solve a bunch of related linear systems to get the closed-form solution [3]. In our experiments, this ADMM process converges significantly faster than the gradient descent solution used in He et al. [2012]'s data reconstruction framework.

After solving the optimization problem, summaries can be generated via a typical greedy sentence selection process, according to the magnitude of row vectors in Z that corresponds to the selection coefficients of each sentence.

2.3 Diversity from Sentence Dissimilarity

The data reconstruction and sparse optimization formulations tend to select sentences that can cover the documents. However, there is no explicit tendency to select diverse sentences capturing different but also important information described in the documents. A very recent work [Liu *et al.*, 2015] also address this diversity problem. They explicitly add the correlation coefficients between sentence pairs in the objective

[3]The matrix $D^T D + \rho I$ is unchanged during iterations. Thus we may pre-compute the required Cholesky factorization to avoid redundant computations for solving those linear systems.

function of He et al. [2012]'s data reconstruction formulation. This makes their objective function difficult to optimize.

Inspired by recent work on exemplar-based sparse selection of representative points [Elhamifar *et al.*, 2012], we introduce an additional dissimilarity term

$$tr(\Delta^T X) = \sum_{i=1}^{n} \sum_{j=1}^{n} \delta_{ij} x_{ij},$$

into our optimization problem (4), weighted by a constant parameter μ:

$$\min_{X,Z} \quad \|D - DX\|^2 + \mu\, tr(\Delta^T X) + \lambda \|Z\|_{2,1} \tag{6}$$

$$s.t. \quad X = Z, diag(X) = 0, x_{ij} \geq 0,\ \forall i, j. \tag{7}$$

where the matrix Δ denotes pairwise dissimilarity δ_{ij} between sentence i and sentence j, measured by an information-theory based criterion as described by Frey and Dueck [2007]: For each word in sentence j, if it is also in sentence i, then set the encoding cost for the word to the logarithm of sentence i's length; Otherwise, set the encoding cost for the word to the logarithm of the vocabulary size. The dissimilarity δ_{ij} is just the sum of these costs. Note that this dissimilarity is asymmetric. We would like to explore other dissimilarity measurements in our future work.

The term $tr(\Delta^T X)$ is linear. Involvement of this term is implemented by simply changing Line 3 of Algorithm 1 into:

$$X_j^{k+1} \leftarrow (D^T D + \rho I)^{-1} (D^T D_j + \rho(Z_j^k - U_j^k) - \mu(\Delta^T)_j)$$

The matrix Δ is pre-computed before the ADMM procedure.

Without loss of generality, we temporarily omit the additional dissimilarity term $tr(\Delta^T X)$ and go back to the compact form (1) in the next section for clarity of description.

3 Compressive Document Summarization

The above sparse optimization problems encode a process of sentence selection. Keeping the same spirit of selection sparseness, we can naturally generalize this sentence selection process to word selection and perform compressive summarization. This can be achieved after some nontrivial modifications of (1):

$$\min_{R,A} \quad \|D - RA\|^2 + \lambda_1 \|A\|_{2,1} + \lambda_2 \textstyle\sum_{i=1}^{n} \|R_i\|_1 \tag{8}$$

$$s.t. \quad r_{ij}, a_{ij} \geq 0,\ \forall i, j;\ grammatical(R_i). \tag{9}$$

The matrix $R \in \mathbb{R}^{d \times n}$ can be regarded as a sparse approximation of the original document D: each column R_i is a compressed version of original sentence vector D_i with several unimportant word dimensions shrinkage to zero by a ℓ_1 regularizer. Some additional constraints ($grammatical(R_i)$) on word selection should be added to ensure grammaticality of compressed sentences. For example, if a verb is selected, then the subject of this verb should also be selected. These constraints are typically induced by dependency relations from dependency parse trees.

The problem (8) jointly optimize over sentence selection coefficients A as well as sparse sentence representation vectors R. Since directly involving the grammaticality constraints might be inconvenient, we leave them to a final step of sentence generation after performing joint optimization.

3.1 Joint Sparse Optimization

The product RA in (8) makes the problem nonconvex. Since the objective function decomposes naturally for R and A, it is straightforward to conduct a block coordinate descent process for this joint sparse problem.

If we have R fixed, the resulting problem for A is:

$$\min_{A} \quad \|D - RA\|^2 + \lambda_1 \|A\|_{2,1} \tag{10}$$

$$s.t. \quad a_{ij} \geq 0,\ \forall i, j. \tag{11}$$

This is a problem we are able to solve by calling the same ADMM procedure derived in section 2.2.

If we have A fixed, the resulting problem for R is:

$$\min_{R} \quad \|D - RA\|^2 + \lambda_2 \textstyle\sum_{i=1}^{n} \|R_i\|_1 \tag{12}$$

$$s.t. \quad r_{ij} \geq 0,\ \forall i, j. \tag{13}$$

This is essentially a sparse dictionary learning problem [Donoho and Elad, 2003]. We can get a solution by iteratively solving LASSO problems defined over columns R_i.

After iteratively solving these two problems, we will get sparse sentence vectors as columns in R.

3.2 Generation of Compressed Sentences

Once the values of each reconstructed sentence vector R_i are ready, we can recover compressed sentences accordingly, considering the previously ignored grammaticality constraints.

Most of modern sentence compression techniques require global constrained optimization, such as integer linear programming [Clarke and Lapata, 2008] and first-order Markov Logic Networks [Huang *et al.*, 2012]. The inference process is far less efficient for our purpose of compressive summarization. Moreover, currently available sentence compression datasets are typically in small scale. Therefore, we only consider unsupervised method with simplest but effective form in this final step. We generate a compressed sentence by selecting a subtree from the full dependency tree of the original sentence.

Suppose for all sentences we have dependency parses labeled with grammatical relations. As the grammatical constraints are usually defined on word-to-word dependency relations, we can treat the dependency arcs locally and perform local inclusion-deletion decisions on dependency trees. We use an efficient recursive algorithm that utilizes the word-level weights and subtree structural constraints. The goal of this algorithm is to extract a subtree with maximum score while satisfying grammatical constraints.

The constraints are expressed in the form of two sets of grammatical relations. The first one is denoted as KEEP_HEAD. It includes dependency relations for cases when the inclusion of a modifier should always imply the inclusion of its direct head. An example relation in this set is noun modifiers (NMOD): if we select the word *nice* in *nice book*, then the head noun *book* should also be selected. The second one is denoted as SIMUL_DEL. It includes dependency relations that should force simultaneous inclusion or deletion of the head and the modifier. For example, subject and object relations (SBJ and OBJ) should be included in this set since dropping any word in a subject-verb-object path will often result in an ungrammatical sentence. We trivially derive the complete list of the two sets from Clarke and Lapata [2008], where these constraints were described in the form of linear constraints in integer linear programs.

The whole process for sentence compression along with subtree scoring is listed in Algorithm 2 in a recursive style. Scores and costs (number of words actually kept) of a subtree will be accumulated from bottom up at each node. The score at each node (subtree) is initialized to be the value of the corresponding word dimension in vector R_i, described in the last section, for the i-th sentence. The cost at each node is initialized to be 1, for the word at this node only. The algorithm decide whether to delete a subtree at Line 5, where we also consider bigram scores (e.g. document frequency of bigrams) to locally ensure grammatical smoothness. At Line 11 the algorithm picks the subtree with maximum score by comparing with the current best subtree $cmax$ (short for current maximum). Note that this comparison is only made when the root is an indicator verb of a grammatical partial sentence [4].

Algorithm 2 A recursive procedure for sentence compression

```
 1: Init: node scores ← $R_i$, node costs ← 1, $cmax \leftarrow NULL$
 2: function GET_MAXSCORE_SUBTREE($V$)
 3:     for each child $C$ in $V$.children do
 4:         $T_C$ ← GET_MAXSCORE_SUBTREE($C$)
 5:         if $C$.label $\notin$ SIMUL_DEL and $T_C$.score/$T_C$.cost $< \epsilon$
            and deletion of $T_C$ does not decrease local bigram score then
 6:             Delete $T_C$ from $C$
 7:         end if
 8:         $V$.score $+=$ $T_C$.score
 9:         $V$.cost $+=$ $T_C$.cost
10:     end for
11:     if $V$ is an indicator verb and $V$.label $\notin$ KEEP_HEAD and
        $V$.score $>$ $cmax$.score then
12:         $cmax \leftarrow V$
13:     end if
14:     if $V$.score $>$ $cmax$.score then return $V$
15:     else return $cmax$
16:     end if
17: end function
```

Here we illustrate the algorithm with an example from bottom up after initialization, with special attention on cases that make Line 5 or Line 11 active. The example is also depicted in Figuire 1, where on each node we use a shorthand notation l, s and c to represent the *label* of grammatical relation, the *score* and the *cost* at current node respectively. Underscored values are initial values at each node. Shadowed nodes will be dropped along with all their subnodes, while bold nodes will be kept. The original sentence is: "*He said the Corps has toughened regulations since 1996, requiring the damage as small as possible.*" and $\epsilon = 0.02$. At the very bottom, dropping the leaf *possible* from *as* will decrease the bigram score, hence it should be kept. Two subnodes of *small* have score-cost ratio $< \epsilon$ and deleting them will not hurt local smoothness in terms of the bigram score, then they will be dropped along with their subnodes. Moving above, the phrase

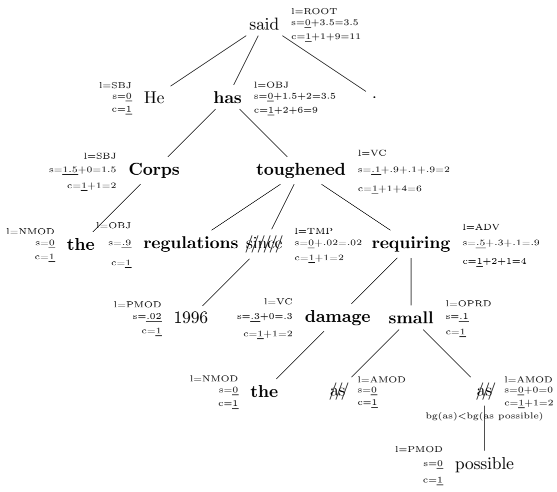

Figure 1: An example of sentence compression

the damage should be kept as a whole to avoid bigram score decrease. Meanwhile, the score of subtree *the damage* (0.3) plus the score of *small* (0.1) will be added onto their head node *requiring* (Line 8) and so are the costs. On the other hand, the subtree *since 1996* will be dropped since its score-cost ratio is $0.02/2 = 0.01 < \epsilon$. Keep going upwards like this, when we reach the node *has*, we have a verb suggesting a grammatical subtree so Line 11 will assign this subtree to $cmax$. The algorithm will ultimately find that the full tree rooted at *said* will have a score not larger than $cmax$.score. Thus $cmax$ will not be updated and the final compressed sentence we get is "*the Corps has toughened regulations, requiring the damage small.*"

This is a linear time algorithm since every node in the dependency tree will be visited for only once.

4 Experiments

4.1 Data Preparation

There exist several benchmark datasets for the task of multi-document summarization. To form direct comparisons with original formulations of data reconstruction based summarization [He *et al.*, 2012], we run our experiments on exactly the same DUC 2006 and DUC 2007 datasets. The main task requires each system to generate a summary less than 250 words for each document cluster, given a short description of topical query. The ϵ for sentence compression is set as $0.01\|R_i\|$ for sentence i. All other parameters involved in the optimization problems are tuned on a fraction of earlier DUC 2005 dataset for convenience.

The dependency relations needed by the compressive summarization part are generated using the MATE tool [5], a fast and accurate dependency parser with the state-of-the-art performance [Bohnet, 2010].

[4]We simply treat non-VBG verbs as such indicator verbs.

[5]http://code.google.com/p/mate-tools/

4.2 Compared Methods

As our framework is fully unsupervised, we do not compare it with supervised methods. Document summarization based on data reconstruction, along with the gradient descent algorithm in He et al. [2012]'s work, naturally becomes the direct baseline for comparison, denoted as DSDR. We denote our sparse optimization formulation of $\ell_{2,1}$ regularization (1) as SpOpt-$\ell_{2,1}$ (the version with diversity term denoted as SpOpt-Δ), and the compressive solution as SpOpt-comp. We also report experimental results on the same datasets (if reported in their paper) given by several recent state-of-the-art unsupervised systems, including matrix-factorization [Wang *et al.*, 2008], the document-sensitive graph model DsR-Q [Wei *et al.*, 2010], the bi-mixture probabilistic latent semantic analysis method (BI-PLSA) [Shen *et al.*, 2011], graph based multimodality learning [Wan and Xiao, 2009], and a very recent work on two-level sparse representation [Liu *et al.*, 2015].

We denote a weeker baseline that extract the leading sentence from each document as LEAD. This baseline is included in the official evaluation. PEER 24 and PEER 15 are the DUC 2006/2007 participants with highest ROUGE performance respectively. For DUC 2007 main task, there is an extractive baseline named CLASSY04 that ignores topic narrative but achieved the highest performance in general multi-document summarization task of DUC 2004.

4.3 Evaluation and Results

We run the commonly used ROUGE (Recall-Oriented Understudy for Gisting Evaluation) [6] metrics for summarization tasks [Lin and Hovy, 2003; Lin, 2004]. The ROUGE metric automatically scores a candidate summary by measuring the amount of ngram overlap between this candidate and manually-created reference summaries. ROUGE-k uses k-gram overlap.

We report ROUGE recall of summaries generated by all systems in comparison. These results are listed in Table 1.

SpOpt-$\ell_{2,1}$ is still superior to DSDR, even though the motivations are similar and the optimization problems are both convex. One probable reason is that the $\ell_{2,1}$ form is directly optimizing coefficient matrix with row sparsity, while the original formulation tries to guide this sparsity indirectly with another group of variables.

Our unsupervised compressive framework outperforms all other unsupervised systems in comparison and achieves very competitive results against the best peer system in DUC 2006/2007. We also observe that adding the sentence dissimilarity term (-Δ) can indeed improve performance.

We also ask three annotators (who are not among the authors of this paper and are fluent in English) to carry out human evaluation for the generated summarization in terms of different aspects of quality, including Grammaticality (GR), Non-Redundancy (NR), Referential Clarity (RC), Topic Focus (TF) and Structural Coherence (SC), similar to the evaluation in DUC 2006 [Dang, 2006]. Each aspect is rated with scores from 1 (poor) to 5 (good). This evaluation is performed on the same random sample of 10 topics in DUC 2006.

[6]Parameter options of ROUGE are set to be consistent with official evaluation of corresponding tasks at DUC 2006 and DUC 2007.

Table 1: Results of ROUGE evaluation on DUC 2006/2007

DUC 2006	**ROUGE-1**	**ROUGE-2**	**ROUGE-SU4**
LEAD	0.30217	0.04947	0.09788
MatrixFacto.	0.39551	0.08549	N/A
DsR-Q	0.39550	0.08990	N/A
BI-PLSA	0.39384	0.08497	N/A
MultiModal.	0.40503	0.08545	N/A
[Liu *et al.*, 2015]	0.34034	0.05233	0.10730
DSDR	0.37695	0.07312	0.11678
SpOpt-$\ell_{2,1}$	0.39069	0.08336	0.13791
SpOpt-Δ	0.39962	0.08682	0.14227
SpOpt-comp	0.41331	0.09136	0.15046
SpOpt-comp-Δ	**0.41534**	0.09455	0.15310
PEER 24	0.41095	**0.09551**	**0.15523**

DUC 2007	**ROUGE-1**	**ROUGE-2**	**ROUGE-SU4**
LEAD	0.31250	0.06039	0.10507
CLASSY04	0.40562	0.09382	0.14641
DsR-Q	0.42190	0.11230	N/A
MultiModal.	0.42609	0.10438	N/A
[Liu *et al.*, 2015]	0.35399	0.06448	0.11669
DSDR	0.39765	0.08679	0.13732
SpOpt-$\ell_{2,1}$	0.41833	0.10627	0.16304
SpOpt-Δ	0.42360	0.11109	0.16474
SpOpt-comp	0.44517	0.12025	0.17072
SpOpt-comp-Δ	**0.44607**	**0.12454**	0.17429
PEER 15	0.44515	0.12448	**0.17715**

The evaluated summaries include the summaries produced by the compressive framework and those from their extractive counterpart, i.e. SpOpt-$\ell_{2,1}$ that only involves sentence selection and extraction. Also one of the official reference summaries generated by human is also in comparison and can be treated as an upper bound for all aspects. The average score and standard deviation for each metric are displayed in Table 2.

From the comparison between compressive summarization and the extractive version, there exist slight improvements of non-redundancy. This exactly matches what we can expect from sentence compression that keeps only important part and drop redundancy. We also observe certain amount of improvements on structural coherence. This may be a result of iterative joint optimization of sentence and word selection that simultaneously considers more global and local coherence.

There may be multiple reasons behind the loss of grammaticality, such as errors of dependency outputs given by the dependency parser, the incompleteness of constraint sets, etc.

Table 3 shows an example summary produced by our compressive system. Words in grey are not selected in the final compressed summaries. In most cases, the removed phrases do not hurt the overall readability of the summary.

In the experiments, the time consumption of our methods is significantly less than the original reconstruction formulation with gradient descent algorithm. Even for the compressive case, the acceleration ratio achieves more than 60 under the same single machine computing environment.

Table 2: Human evaluation results

Written	GR	NR	RC	TF	SC
SpOpt-Δ	4.20±0.70	3.53±0.62	3.93±0.93	3.80±0.87	3.40±0.66
SpOpt-comp-Δ	3.43±0.84	3.97±0.60	4.00±0.86	3.90±0.79	3.73±0.68
Human	4.97±0.18	4.93±0.25	5.00±0.00	4.93±0.25	4.93±0.25

Table 3: Example compressive summary from DUC 2006

SYSTEM OUTPUT For many parts of the Americas and Asia, the occasional warming and cooling cycles in the tropical Pacific Ocean known as El Nino and La Nina are unwelcome visitors barging in, usually with little warning, then staying for months or sometimes a year or two, bringing all kinds of baggage in the form of distorted patterns of storms and droughts, heat and cold. Although La Nina is already affecting rainfall in Southeast Asia, it hasn't noticeably influenced North America's weather yet, Leetmaa said Instead, it may be that warm water left over from El Nino is contributing to the formation of more tropical storms than normal in the eastern Pacific. Scientists cautioned that like its warm counterpart, El Nino, a La Nina condition will influence global climate and weather until it has completely subsided. La Nina, the assertive sister of last year's hellacious El Nino, has already contributed to freakish weather around the globe and will continue to bring colder than normal temperatures to the West Coast well into spring, according to government forecasters. With no El Nino or La Nina, there is no sheep dog driving the sheep, no strong organizing force giving shape to the otherwise chaotic flow of weather across the seas and continents. La Nina and El Nino form the opposite ends of the same climatic cycle, with El Nino fueled by unusually warm water in the eastern Pacific and La Nina driven by cold. The pool of cold Pacific water that is the heart of La Nina has shrunk to one-fourth the size it was in May, even as warm water left over from El Nino remains, said Bill Patzert, a researcher at the National Aeronautics and Space Administration laboratory in suburban Pasadena.
REFERENCE SUMMARY El Nino is a disruptive weather phenomenon that usually occurs every three to four years on the average. It involves a surface warming of the eastern and central Pacific Ocean around the equator when trade winds weaken. It can disrupt climate around the world, producing extra rain in the southeastern U.S., Peru and Ecuador during the winter, while causing drought in the western Pacific as well as the slowing of trade winds and changes in sea levels. Basically, it reverses normal weather patterns, resulting in drought in usually wet locations and flooding in arid areas. It also helps trigger some man-made disasters, such as forest fires, and enhances conditions that cause viral disease among humans and livestock. La Nina is the phenomenon of rapidly cooling equatorial waters in the central Pacific. It develops every several years and works in reverse of El Nino. It can last for one year causing cooler conditions in central North America and dry warm conditions in the southern states that can be just as disruptive as El Nino. On the positive side, El Nino can be credited with saving lives that would have been lost in normal winter and hurricane seasons. It may also help cut global warming by temporarily stemming release of carbon dioxide from the Pacific Ocean. Computer module studies and satellite systems allow for a better understanding of how El Nino and La Nina form but, unfortunately, when they will develop or what they hold for the future still cannot be predicted.

5 Related Work

Our sparse optimization formulations are closely related to data reconstruction for document summarization. The data reconstruction paradigm for document summarization, originally proposed by He et al. [2012], was inspired by latent semantic analysis (LSA) that utilizes singular value decomposition (SVD) to select highly ranked sentences [Gong and Liu, 2001]. Nonnegative matrix factorization has also been introduced to group sentences into clusters [Wang *et al.*, 2008]. Recently Liu et al. [2015] propose a two-level sparse representation model. Their optimization problem is NP-hard so heuristic methods such as simulated annealing has been used to solve it approximately.

In recent years some research has made much progress beyond extractive summarization, especially in the context of compressive summarization. An earlier attempt made by Zajic et al. [2006] tried a pipeline strategy with heuristics to generate multiple candidate compressions and extract from this compressed sentences. Berg-Kirkpatrik et al. [2011] created linear models of weights learned by structural SVMs for different components and tried to jointly do sentence selection and syntax tree trimming in integer linear programs. Woodsend and Lapata [2012] designed quasi tree substitution grammars for multiple rewriting operations. All these methods involve integer linear programming solvers to generate the final compressed summary, which is time-consuming for multi-document summarization tasks.

Almeida and Martins [2013] formed the compressive summarization problem in a more efficient dual decomposition framework. Models for sentence compression and extractive summarization are trained by multi-task learning techniques. Wang et al. [2013] explored different types of compression on constituent parse trees for query-focused summarization. In these works, the best-performing systems require supervised learning for different subtasks.

Our mathematical formulations are closely related to modern sparse optimization problems. Subspace clustering techniques [Elhamifar and Vidal, 2013] try to learn proper coefficients, aiming at a self-representation. The difference between general sparse subspace clustering problems and our formulations will make key impact on the choice and design of solving algorithms. The difference comes mainly from different motivations. The former expect for sparsity in sentence selection, while the latter typically requires low-rankness in matrix representations.

6 Conclusion and Future Work

In this paper we propose a new formulation for document summarization via sparse optimization with decomposable convex objective function and derive an efficient ADMM algorithm to solve it. We also introduce a sentence dissimilarity term to encourage diversity in summaries. Then we generalize the proposed method to compressive summarization and derive a block coordinate descent procedure along with recursive dependency tree compression to generate the final sentences. Experimental study shows that our compressive summarization framework significantly improves results from the original extractive methods based on data reconstruction.

Structured sparsity has been studied for a while in machine learning community. However, its adaptation in natural language processing and text mining is still at its beginning [Martins *et al.*, 2011; Yogatama and Smith, 2014]. We would like to explore if structured sparsity can become useful for compressive summarization tasks.

Our proposed methods are fully unsupervised. We would like to extend it to supervised case for different optimization problems described in this paper. It is reasonable to expect for even better performance.

Acknowledgments

We thank all the anonymous reviewers for helpful comments and suggestions. This work was supported by National Hi-

Tech Research and Development Program (863 Program) of China (2015AA015403, 2014AA015102) and National Natural Science Foundation of China (61170166, 61331011).

References

[Almeida and Martins, 2013] Miguel Almeida and Andre Martins. Fast and robust compressive summarization with dual decomposition and multi-task learning. In *ACL*, pages 196–206, August 2013.

[Berg-Kirkpatrick *et al.*, 2011] Taylor Berg-Kirkpatrick, Dan Gillick, and Dan Klein. Jointly learning to extract and compress. In *ACL-HLT*, pages 481–490, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

[Bohnet, 2010] Bernd Bohnet. Top accuracy and fast dependency parsing is not a contradiction. In *COLING 2010*, pages 89–97, August 2010.

[Boyd *et al.*, 2011] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

[Clarke and Lapata, 2008] James Clarke and Mirella Lapata. Global inference for sentence compression: An integer linear programming approach. *Journal of Artificial Intelligence Research*, 31:273–381, 2008.

[Dang, 2006] Hoa Trang Dang. Overview of duc 2006. In *Proceedings of the DUC*, 2006.

[Donoho and Elad, 2003] David L Donoho and Michael Elad. Optimally sparse representation in general (nonorthogonal) dictionaries via $\ell 1$ minimization. *PNAS*, 100(5):2197–2202, 2003.

[Elhamifar and Vidal, 2013] Ehsan Elhamifar and Rene Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(11):2765–2781, 2013.

[Elhamifar *et al.*, 2012] Ehsan Elhamifar, Guillermo Sapiro, and René Vidal. Finding exemplars from pairwise dissimilarities via simultaneous sparse recovery. In *NIPS*, 2012.

[Frey and Dueck, 2007] Brendan J Frey and Delbert Dueck. Clustering by passing messages between data points. *science*, 315(5814):972–976, 2007.

[Gong and Liu, 2001] Yihong Gong and Xin Liu. Generic text summarization using relevance measure and latent semantic analysis. In *SIGIR*, 2001.

[He *et al.*, 2012] Zhanying He, Chun Chen, Jiajun Bu, Can Wang, Lijun Zhang, Deng Cai, and Xiaofei He. Document summarization based on data reconstruction. In *AAAI*, 2012.

[Huang *et al.*, 2012] Minlie Huang, Xing Shi, Feng Jin, and Xiaoyan Zhu. Using first-order logic to compress sentences. In *AAAI*, 2012.

[Lin and Hovy, 2003] Chin-Yew Lin and Eduard Hovy. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 71–78. Association for Computational Linguistics, 2003.

[Lin, 2004] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 74–81, 2004.

[Liu *et al.*, 2015] He Liu, Hongliang Yu, and Zhi-Hong Deng. Multi-document summarization based on two-level sparse representation model. In *AAAI*, 2015.

[Martins *et al.*, 2011] Andre Martins, Noah Smith, Mario Figueiredo, and Pedro Aguiar. Structured sparsity in structured prediction. In *EMNLP*, pages 1500–1511, July 2011.

[Shen *et al.*, 2011] Chao Shen, Tao Li, and Chris HQ Ding. Integrating clustering and multi-document summarization by bi-mixture probabilistic latent semantic analysis (plsa) with sentence bases. In *AAAI*, 2011.

[Wan and Xiao, 2009] Xiaojun Wan and Jianguo Xiao. Graph-based multi-modality learning for topic-focused multi-document summarization. In *IJCAI*, pages 1586–1591, 2009.

[Wang *et al.*, 2008] Dingding Wang, Tao Li, Shenghuo Zhu, and Chris Ding. 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*, pages 307–314. ACM, 2008.

[Wang *et al.*, 2013] Lu Wang, Hema Raghavan, Vittorio Castelli, Radu Florian, and Claire Cardie. A sentence compression based framework to query-focused multi-document summarization. In *ACL*, pages 1384–1394, August 2013.

[Wei *et al.*, 2010] Furu Wei, Wenjie Li, Qin Lu, and Yanxiang He. A document-sensitive graph model for multi-document summarization. *Knowledge and information systems*, 22(2):245–259, 2010.

[Woodsend and Lapata, 2012] Kristian Woodsend and Mirella Lapata. Multiple aspect summarization using integer linear programming. In *EMNLP-CoNLL*, pages 233–243, 2012.

[Yogatama and Smith, 2014] Dani Yogatama and Noah A. Smith. Linguistic structured sparsity in text categorization. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 786–796, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

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