Large-Margin Learning of Submodular Summarization Models

Ruben Sipos
Dept. of Computer Science
Cornell University
Ithaca, NY 14853 USA
rs@cs.cornell.edu

Pannaga Shivaswamy
Dept. of Computer Science
Cornell University
Ithaca, NY 14853 USA
pannaga@cs.cornell.edu

Thorsten Joachims
Dept. of Computer Science
Cornell University
Ithaca, NY 14853 USA
tj@cs.cornell.edu

Abstract

In this paper, we present a supervised learning approach to training submodular scoring functions for extractive multi-document summarization. By taking a structured prediction approach, we provide a large-margin method that directly optimizes a convex relaxation of the desired performance measure. The learning method applies to all submodular summarization methods, and we demonstrate its effectiveness for both pairwise as well as coverage-based scoring functions on multiple datasets. Compared to state-of-the-art functions that were tuned manually, our method significantly improves performance and enables high-fidelity models with number of parameters well beyond what could reasonably be tuned by hand.

1 Introduction

Automatic document summarization is the problem of constructing a short text describing the main points in a (set of) document(s). Example applications range from generating short summaries of news articles, to presenting snippets for URLs in web-search. In this paper we focus on extractive multi-document summarization, where the final summary is a subset of the sentences from multiple input documents. In this way, extractive summarization avoids the hard problem of generating well-formed natural-language sentences, since only existing sentences from the input documents are presented as part of the summary.

A current state-of-the-art method for document summarization was recently proposed by Lin and Bilmes (2010), using a submodular scoring function based on inter-sentence similarity. On the one hand, this scoring function rewards summaries that are similar to many sentences in the original documents (i.e. promotes coverage). On the other hand, it penalizes summaries that contain sentences that are similar to each other (i.e. discourages redundancy). While obtaining the exact summary that optimizes the objective is computationally hard, they show that a greedy algorithm is guaranteed to compute a good approximation. However, their work does not address how to select a good inter-sentence similarity measure, leaving this problem as well as selecting an appropriate trade-off between coverage and redundancy to manual tuning.

To overcome this problem, we propose a supervised learning method that can learn both the similarity measure as well as the coverage/reduncancy trade-off from training data. Furthermore, our learning algorithm is not limited to the model of Lin and Bilmes (2010), but applies to all monotone submodular summarization models. Due to the diminishing-returns property of monotone submodular set functions and their computational tractability, this class of functions provides a rich space for designing summarization methods. To illustrate the generality of our approach, we also provide experiments for a coverage-based model originally developed for diversified information retrieval (Swaminathan et al., 2009).

In general, our method learns a parameterized monotone submodular scoring function from supervised training data, and its implementation is available for download.[1] Given a set of documents and their summaries as training examples,

[1] http://www.cs.cornell.edu/~rs/sfour/

we formulate the learning problem as a structured prediction problem and derive a maximum-margin algorithm in the structural support vector machine (SVM) framework. Note that, unlike other learning approaches, our method does not require a heuristic decomposition of the learning task into binary classification problems (Kupiec et al., 1995), but directly optimizes a structured prediction. This enables our algorithm to directly optimize the desired performance measure (e.g. ROUGE) during training. Furthermore, our method is not limited to linear-chain dependencies like (Conroy and O'leary, 2001; Shen et al., 2007), but can learn any monotone submodular scoring function.

This ability to easily train summarization models makes it possible to efficiently tune models to various types of document collections. In particular, we find that our learning method can reliably tune models with hundreds of parameters based on a training set of about 30 examples. This increases the fidelity of models compared to their hand-tuned counterparts, showing significantly improved empirical performance. We provide a detailed investigation into the sources of these improvements, identifying further directions for research.

2 Related work

Work on extractive summarization spans a large range of approaches. Starting with unsupervised methods, one of the widely known approaches is Maximal Marginal Relevance (MMR) (Carbonell and Goldstein, 1998). It uses a greedy approach for selection and considers the trade-off between relevance and redundancy. Later it was extended (Goldstein et al., 2000) to support multi-document settings by incorporating additional information available in this case. Good results can be achieved by reformulating this as a knapsack packing problem and solving it using dynamic programing (McDonald, 2007). Alternatively, we can use annotated phrases as textual units and select a subset that covers most concepts present in the input (Filatova and Hatzivassiloglou, 2004) (which can also be achieved by our coverage scoring function if it is extended with appropriate features).

A popular stochastic graph-based summarization method is LexRank (Erkan and Radev, 2004). It computes sentence importance based on the concept of eigenvector centrality in a graph of sentence similarities. Similarly, TextRank (Mihalcea and Tarau, 2004) is also graph based ranking system for identification of important sentences in a document by using sentence similarity and PageRank (Brin and Page, 1998). Sentence extraction can also be implemented using other graph based scoring approaches (Mihalcea, 2004) such as HITS (Kleinberg, 1999) and positional power functions. Graph based methods can also be paired with clustering such as in CollabSum (Wan et al., 2007). This approach first uses clustering to obtain document clusters and then uses graph based algorithm for sentence selection which includes inter and intra-document sentence similarities. Another clustering-based algorithm (Nomoto and Matsumoto, 2001) is a diversity-based extension of MMR that finds diversity by clustering and then proceeds to reduce redundancy by selecting a representative for each cluster.

The manually tuned sentence pairwise model (Lin and Bilmes, 2010; Lin and Bilmes, 2011) we took inspiration from is based on budgeted submodular optimization. A summary is produced by maximizing an objective function that includes coverage and redundancy terms. Coverage is defined as the sum of sentence similarities between the selected summary and the rest of the sentences, while redundancy is the sum of pairwise intra-summary sentence similarities. Another approach based on submodularity (Qazvinian et al., 2010) relies on extracting important keyphrases from citation sentences for a given paper and using them to build the summary.

In the supervised setting, several early methods (Kupiec et al., 1995) made independent binary decisions whether to include a particular sentence in the summary or not. This ignores dependencies between sentences and can result in high redundancy. The same problem arises when using learning-to-rank approaches such as ranking support vector machines, support vector regression and gradient boosted decision trees to select the most relevant sentences for the summary (Metzler and Kanungo, 2008).

Introducing some dependencies can improve the performance. One limited way of introducing dependencies between sentences is by using a linear-chain HMM. The HMM is assumed to produce the summary by having a chain transitioning

between summarization and non-summarization states (Conroy and O'leary, 2001) while traversing the sentences in a document. A more expressive approach is using a CRF for sequence labeling (Shen et al., 2007) which can utilize larger and not necessarily independent feature spaces. The disadvantage of using linear chain models, however, is that they represent the summary as a sequence of sentences. Dependencies between sentences that are far away from each other cannot be modeled efficiently. In contrast to such linear chain models, our approach on submodular scoring functions can model long-range dependencies. In this way our method can use properties of the whole summary when deciding which sentences to include in it.

More closely related to our work is that of Li et al. (2009). They use the diversified retrieval method proposed in Yue and Joachims (2008) for document summarization. Moreover, they assume that subtopic labels are available so that additional constraints for diversity, coverage and balance can be added to the structural SVM learning problem. In contrast, our approach does not require the knowledge of subtopics (thus allowing us to apply it to a wider range of tasks) and avoids adding additional constraints (simplifying the algorithm). Furthermore, it can use different submodular objective functions, for example word coverage and sentence pairwise models described later in this paper.

Another closely related work also takes a max-margin discriminative learning approach in the structural SVM framework (Berg-Kirkpatrick et al., 2011) or by using MIRA (Martins and Smith, 2009) to learn the parameters for summarizing a set of documents. However, they do not consider submodular functions, but instead solve an Integer Linear Program (ILP) or an approximation thereof. The ILP encodes a compression model where arbitrary parts of the parse trees of sentences in the summary can be cut and removed. This allows them to select parts of sentences and yet preserve some gramatical structure. Their work focuses on learning a particular compression model based on ILP inference, while our work explores learning a general and large class of sentence selection models using submodular optimization. The third notable approach uses SEARN (Daumé, 2006) to learn parameters for joint summarization and compression model, however it uses vine-growth model and employs search to to find the best policy which is then used to generate a summary.

A specific subclass of submodular (but not monotone) functions are defined by Determinantal Point Processes (DPPs) (Kulesza and Taskar, 2011). While they provide an elegant probabilistic interpretation of the resulting summarization models, the lack of monotonicity means that no efficient approximation algorithms are known for computing the highest-scoring summary.

3 Submodular document summarization

In this section, we illustrate how document summarization can be addressed using submodular set functions. The set of documents to be summarized is split into a set of individual sentences $x = \{s_1, ..., s_n\}$. The summarization method then selects a subset $\hat{y} \subseteq x$ of sentences that maximizes a given scoring function $F_x : 2^x \rightarrow \mathbb{R}$ subject to a budget constraint (e.g. less than B characters).

$$\hat{y} = \arg\max_{y \subseteq x} F_x(y) \qquad s.t. \; |y| \leq B \qquad (1)$$

In the following we restrict the admissible scoring functions F to be submodular.

Definition 1. *Given a set x, a function $F : 2^x \rightarrow \mathbb{R}$ is submodular iff for all $u \in U$ and all sets s and t such that $s \subseteq t \subseteq x$, we have,*

$$F(s \cup \{u\}) - F(s) \geq F(t \cup \{u\}) - F(t).$$

Intuitively, this definition says that adding u to a subset s of t increases f at least as much as adding it to t. Using two specific submodular functions as examples, the following sections illustrate how this diminishing returns property naturally reflects the trade-off between maximizing coverage while minimizing redundancy.

3.1 Pairwise scoring function

The first submodular scoring function we consider was proposed by Lin and Bilmes (2010) and is based on a model of pairwise sentence similarities. It scores a summary y using the following function, which Lin and Bilmes (2010) show is submodular:

$$F_x(y) = \sum_{i \in x \setminus y, j \in y} \sigma(i,j) - \lambda \sum_{i,j \in y : i \neq j} \sigma(i,j). \qquad (2)$$

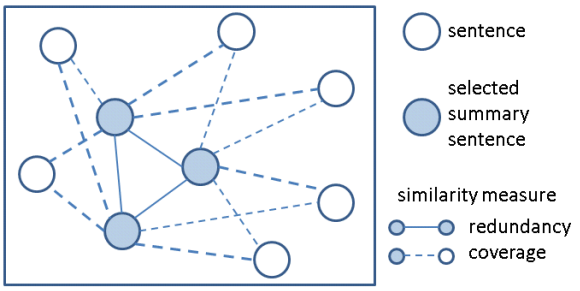

Figure 1: Illustration of the pairwise model. Not all edges are shown for clarity purposes. Edge thickness denotes the similarity score.

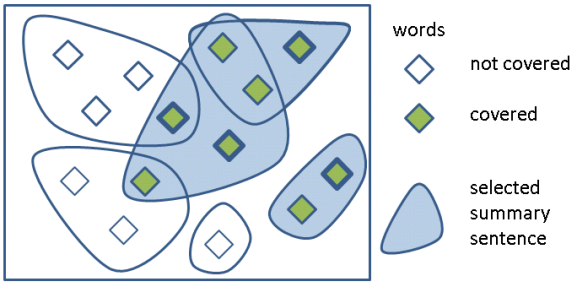

Figure 2: Illustration of the coverage model. Word border thickness represents importance.

In the above equation, $\sigma(i,j) \geq 0$ denotes a measure of similarity between pairs of sentences i and j. The first term in Eq. 2 is a measure of how similar the sentences included in summary y are to the other sentences in x. The second term penalizes y by how similar its sentences are to each other. $\lambda > 0$ is a scalar parameter that trades off between the two terms. Maximizing $F_x(y)$ amounts to increasing the similarity of the summary to excluded sentences while minimizing repetitions in the summary. An example is illustrated in Figure 1. In the simplest case, $\sigma(i,j)$ may be the TFIDF (Salton and Buckley, 1988) cosine similarity, but we will show later how to learn sophisticated similarity functions.

3.2 Coverage scoring function

A second scoring function we consider was first proposed for diversified document retrieval (Swaminathan et al., 2009; Yue and Joachims, 2008), but it naturally applies to document summarization as well (Li et al., 2009). It is based on a notion of word coverage, where each word v has some importance weight $\omega(v) \geq 0$. A summary y covers a word if at least one of its sentences contains the word. The score of a summary is then simply the sum of the word weights its covers (though we could also include a concave discount function that rewards covering a word multiple times (Raman et al., 2011)):

$$F_x(y) = \sum_{v \in V(y)} \omega(v). \quad (3)$$

In the above equation, $V(y)$ denotes the union of all words in y. This function is analogous to a maximum coverage problem, which is known to be submodular (Khuller et al., 1999).

An example of how a summary is scored is illustrated in the Figure 2. Analogous to the definition of similarity $\sigma(i,j)$ in the pairwise model, the choice of the word importance function $\omega(v)$ is crucial in the coverage model. A simple heuristic is to weigh words highly that occur in many sentences of x, but in few other documents (Swaminathan et al., 2009). However, we will show in the following how to learn $\omega(v)$ from training data.

Algorithm 1 Greedy algorithm for finding the best summary $\hat{y}$ given a scoring function $F_x(y)$.

Parameter: $r > 0$.
$\hat{y} \leftarrow \emptyset$
$A \leftarrow x$
while $A \neq \emptyset$ **do**
  $k \leftarrow \arg\max_{l \in A} \frac{F_x(\hat{y} \cup \{l\}) - F_x(\hat{y})}{(c_l)^r}$
  if $c_k + \sum_{i \in \hat{y}} c_i \leq B$ **and** $F_x(\hat{y} \cup \{k\}) - F_x(\hat{y}) \geq 0$ **then**
    $\hat{y} \leftarrow \hat{y} \cup \{k\}$
  end if
  $A \leftarrow A \backslash \{k\}$
end while

3.3 Computing a Summary

Computing the summary that maximizes either of the two scoring functions from above (i.e. Eqns. (2) and (3)) is NP-hard (McDonald, 2007). However, it is known that the greedy algorithm 1 can achieve a $1 - 1/e$ approximation to the optimum solution for any linear budget constraint (Lin and Bilmes, 2010; Khuller et al., 1999). Even further, this algorithm provides a $1 - 1/e$ approximation for any monotone submodular scoring function.

The algorithm starts with an empty summarization. In each step, a sentence is added to the summary that results in the maximum relative increase

of the objective. The increase is relative to the amount of budget that is used by the added sentence. The algorithm terminates when the budget B is reached.

Note that the algorithm has a parameter r in the denominator of the selection rule, which Lin and Bilmes (2010) report to have some impact on performance. In the algorithm, c_i represents the cost of the sentence (i.e., length). Thus, the algorithm actually selects sentences with large marginal unity relative to their length (trade-off controlled by the parameter r). Selecting r to be less than 1 gives more importance to "information density" (i.e. sentences that have a higher ratio of score increase per length). The $1 - \frac{1}{e}$ greedy approximation guarantee holds despite this additional parameter (Lin and Bilmes, 2010). More details on our choice of r and its effects are provided in the experiments section.

4 Learning algorithm

In this section, we propose a supervised learning method for training a submodular scoring function to produce desirable summaries. In particular, for the pairwise and the coverage model, we show how to learn the similarity function $\sigma(i, j)$ and the word importance weights $\omega(v)$ respectively. In particular, we parameterize $\sigma(i, j)$ and $\omega(v)$ using a linear model, allowing that each depends on the full set of input sentences x:

$$\sigma_x(i, j) = \mathbf{w}^T \phi_x^p(i, j) \quad \omega_x(v) = \mathbf{w}^T \phi_x^c(v). \quad (4)$$

In the above equations, $\mathbf{w}$ is a weight vector that is learned, and $\phi_x^p(i, j)$ and $\phi_x^c(v)$ are feature vectors. In the pairwise model, $\phi_x^p(i, j)$ may include feature like the TFIDF cosine between i and j or the number of words from the document titles that i and j share. In the coverage model, $\phi_x^c(v)$ may include features like a binary indicator of whether v occurs in more than 10% of the sentences in x or whether v occurs in the document title.

We propose to learn the weights following a large-margin framework using structural SVMs (Tsochantaridis et al., 2005). Structural SVMs learn a discriminant function

$$h(x) = \arg\max_{y \in \mathcal{Y}} \mathbf{w}^\top \Psi(x, y) \quad (5)$$

that predicts a structured output y given a (possibly also structured) input x. $\Psi(x, y) \in \mathbb{R}^N$ is called the joint feature-map between input x and output y. Note that both submodular scoring function in Eqns. (2) and (3) can be brought into the form $\mathbf{w}^T \Psi(x, y)$ for the linear parametrization in Eq. (6) and (7):

$$\Psi^p(x, y) = \sum_{i \in x \setminus y, j \in y} \phi_x^p(i, j) - \lambda \sum_{i,j \in y : i \neq j} \phi_x^p(i, j), \quad (6)$$

$$\Psi^c(x, y) = \sum_{v \in V(y)} \phi_x^c(v). \quad (7)$$

After this transformation, it is easy to see that computing the maximizing summary in Eq. (1) and the structural SVM prediction rule in Eq. (5) are equivalent.

To learn the weight vector $\mathbf{w}$, structural SVMs require training examples $(x^1, y^1), ..., (x^n, y^n)$ of input/output pairs. In document summarization, however, the "correct" extractive summary is typically not known. Instead, training documents x^i are typically annotated with multiple manual (non-extractive) summaries (denoted by Y^i). To determine a single extractive target summary y^i for training, we find the extractive summary that (approximately) optimizes ROUGE score – or some other loss function $\Delta(Y^i, y)$ – with respect to Y^i.

$$y^i = \operatorname*{argmin}_{y \in \mathcal{Y}} \Delta(Y^i, y) \quad (8)$$

We call the y^i determined in this way the "target" summary for x^i. Note that y^i is a greedily constructed approximate target summary based on its proximity to Y^i via Δ. Because of this, we will learn a model that can predict approximately good summaries y^i from x_i. However, we believe that most of the score difference between manual summaries and y^i (as explored in the experiments section) is due to it being an extractive summary and not due to greedy construction.

Following the structural SVM approach, we can now formulate the problem of learning $\mathbf{w}$ as the following quadratic program (QP):

$$\min_{\mathbf{w}, \xi \geq 0} \frac{1}{2} \|\mathbf{w}\|^2 + \frac{C}{n} \sum_{i=1}^{n} \xi_i \quad (9)$$

$$\text{s.t. } \mathbf{w}^\top \Psi(x^i, y^i) - \mathbf{w}^\top \Psi(x^i, \hat{y}^i) \geq \Delta(\hat{y}^i, Y^i) - \xi_i, \ \ \forall \hat{y}^i \neq y^i, \ \ \forall 1 \leq i \leq n.$$

The above formulation ensures that the scoring function with the target summary (i.e. $\mathbf{w}^\top \Psi(x^i, y^i)$) is larger than the scoring function

Algorithm 2 Cutting-plane algorithm for solving the learning optimization problem.

Parameter: desired tolerance $\epsilon > 0$.
$\forall i : \mathcal{W}_i \leftarrow \emptyset$
repeat
  for $\forall i$ **do**
    $\hat{y} \leftarrow \arg\max_y w^T \Psi(x^i, y) + \Delta(Y^i, y)$
    if $w^T \Psi(x^i, y^i) + \epsilon \leq w^T \Psi(x^i, \hat{y}) + \Delta(Y^i, \hat{y}) - \xi_i$ **then**
      $\mathcal{W}_i \leftarrow \mathcal{W}_i \cup \{\hat{y}\}$
      $w \leftarrow$ solve QP (9) using constraints $\mathcal{W}_i$
    end if
  end for
until no $\mathcal{W}_i$ has changed during iteration

for any other summary $\hat{y}^i$ (i.e., $\mathbf{w}^\top \Psi(x^i, \hat{y}^i)$). The objective function learns a large-margin weight vector $\mathbf{w}$ while trading it off with an upper bound on the empirical loss. The two quantities are traded off with a parameter $C > 0$.

Even though the QP has exponentially many constraints in the number of sentences in the input documents, it can be solved approximately in polynomial time via a cutting plane algorithm (Tsochantaridis et al., 2005). The steps of the cutting-plane algorithm are shown in Algorithm 2. In each iteration of the algorithm, for each training document x^i, a summary $\hat{y}^i$ which most violates the constraint in (9) is found. This is done by finding

$$\hat{y} \leftarrow \arg\max_{y \in \mathcal{Y}} w^T \Psi(x^i, y) + \Delta(Y^i, y),$$

for which we use a variant of the greedy algorithm in Figure 1. After a violating constraint for each training example is added, the resulting quadratic program is solved. These steps are repeated until all the constraints are satisfied to a required precision ϵ.

Finally, special care has to be taken to appropriately define the loss function Δ given the disparity of Y^i and y^i. Therefore, we first define an intermediate loss function

$$\Delta_R(Y, \hat{y}) = \max(0, 1 - ROUGE1_F(Y, \hat{y})),$$

based on the ROUGE-1 F score. To ensure that the loss function is zero for the target label as defined in (8), we normalized the above loss as below:

$$\Delta(Y^i, \hat{y}) = \max(0, \Delta_R(Y^i, \hat{y}) - \Delta_R(Y^i, y^i)),$$

The loss Δ was used in our experiments. Thus training a structural SVM with this loss aims to maximize the ROUGE-1 F score with the manual summaries provided in the training examples, while trading it off with margin. Note that we could also use a different loss function (as the method is not tied to this particular choice), if we had a different target evaluation metric. Finally, once a $\mathbf{w}$ is obtained from structural SVM training, a predicted summary for a test document x can be obtained from (5).

5 Experiments

In this section, we empirically evaluate the approach proposed in this paper. Following Lin and Bilmes (2010), experiments were conducted on two different datasets (DUC '03 and '04). These datasets contain document sets with four manual summaries for each set. For each document set, we concatenated all the articles and split them into sentences using the tool provided with the '03 dataset. For the supervised setting we used 10 resamplings with a random 20/5/5 ('03) and 40/5/5 ('04) train/test/validation split. We determined the best C value in (9) using the performance on each validation set and then report average performence over the corresponding test sets. Baseline performance (the approach of Lin and Bilmes (2010)) was computed using all 10 test sets as a single test set. For all experiments and datasets, we used $r = 0.3$ in the greedy algorithm as recommended in Lin and Bilmes (2010) for the '03 dataset. We find that changing r has only a small influence on performance.[2]

The construction of features for learning is organized by word groups. The most trivial group is simply all words (*basic*). Considering the properties of the words themselves, we constructed several features from properties such as capitalized words, non-stop words and words of certain length (*cap+stop+len*). We obtained another set of features from the most frequently occuring words in all the articles (*minmax*). We also considered the position of a sentence (containing

[2] Setting r to 1 and thus eliminating the non-linearity does lower the score (e.g. to 0.38466 for the pairwise model on DUC '03 compared with the results on Figure 3).

the word) in the article as another feature (*location*). All those word groups can then be further refined by selecting different thresholds, weighting schemes (e.g. TFIDF) and forming binned variants of these features.

For the pairwise model we use cosine similarity between sentences using only words in a given word group during computation. For the word coverage model we create separate features for covering words in different groups. This gives us fairly comparable feature strength in both models. The only further addition is the use of different word coverage levels in the coverage model. First we consider how well does a sentence cover a word (e.g. a sentence with five instances of the same word might cover it better than another with only a single instance). And secondly we look at how important it is to cover a word (e.g. if a word appears in a large fraction of sentences we might want to be sure to cover it). Combining those two criteria using different thresholds we get a set of features for each word. Our coverage features are motivated from the approach of Yue and Joachims (2008). In contrast, the hand-tuned pairwise baseline uses only TFIDF weighted cosine similarity between sentences using all words, following the approach in Lin and Bilmes (2010).

The resulting summaries are evaluated using ROUGE version 1.5.5 (Lin and Hovy, 2003). We selected the ROUGE-1 F measure because it was used by Lin and Bilmes (2010) and because it is one of the commonly used performance scores in recent work. However, our learning method applies to other performance measures as well. Note that we use the ROUGE-1 F measure both for the loss function during learning, as well as for the evaluation of the predicted summaries.

5.1 How does learning compare to manual tuning?

In our first experiment, we compare our supervised learning approach to the hand-tuned approach. The results from this experiment are summarized in Figure 3. First, supervised training of the pairwise model (Lin and Bilmes, 2010) resulted in a statistically significant ($p \leq 0.05$ using paired t-test) increase in performance on both datasets compared to our reimplementation of the manually tuned pairwise model. Note that our reimplementation of the approach of Lin and Bilmes (2010) resulted in slightly different performance numbers than those reported in Lin and Bilmes (2010) – better on DUC '03 and somewhat lower on DUC '04, if evaluated on the same selection of test examples as theirs. We conjecture that this is due to small differences in implementation and/or preprocessing of the dataset. Furthermore, as authors of Lin and Bilmes (2010) note in their paper, the '03 and '04 datasets behave quite differently.

model	dataset	ROUGE-1 F (stderr)
pairwise	DUC '03	**0.3929** (0.0074)
coverage		**0.3784** (0.0059)
hand-tuned		0.3571 (0.0063)
pairwise	DUC '04	**0.4066** (0.0061)
coverage		0.3992 (0.0054)
hand-tuned		0.3935 (0.0052)

Figure 3: Results obtained on DUC '03 and '04 datasets using the supervised models. Increase in performance over the hand-tuned is statistically significant ($p \leq 0.05$) for the pairwise model on the both datasets, but only on DUC '03 for the coverage model.

Figure 3 also reports the performance for the coverage model as trained by our algorithm. These results can be compared against those for the pairwise model. Since we are using features of comparable strength in both approaches, as well as the same greedy algorithm and structural SVM learning method, this comparison largely reflects the quality of models themselves. On the '04 dataset both models achieve the same performance while on '03 the pairwise model performs significantly ($p \leq 0.05$) better than the coverage model.

Overall, the pairwise model appears to perform slightly better than the coverage model with the datasets and features we used. Therefore, we focus on the pairwise model in the following.

5.2 How fast does the algorithm learn?

Hand-tuned approaches have limited flexibility. Whenever we move to a significantly different collection of documents we have to reinvest time to retune it. Learning can make this adaptation to a new collection more automatic and faster – especially since training data has to be collected even for manual tuning.

Figure 4 evaluates how effectively the learning algorithm can make use of a given amount of training data. In particular, the figure shows the

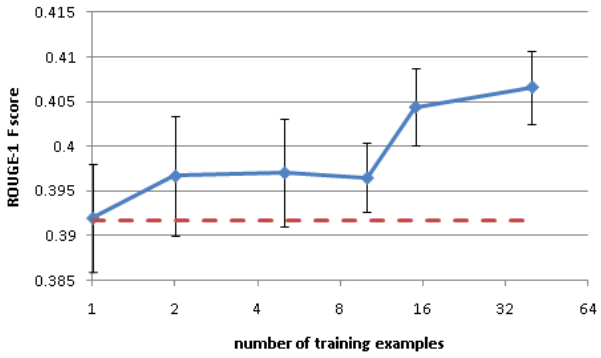

Figure 4: Learning curve for the pairwise model on DUC '04 dataset showing ROUGE-1 F scores for different numbers of learning examples (logarithmic scale). The dashed line represents the preformance of the hand-tuned model.

learning curve for our approach. Even with very few training examples, the learning approach already outperforms the baseline. Furthermore, at the maximum number of training examples available to us the curve still increases. We therefore conjecture that more data would further improve performance.

5.3 Where is room for improvement?

To get a rough estimate of what is actually achievable in terms of the final ROUGE-1 F score, we looked at different "upper bounds" under various scenarios (Figure 5). First, ROUGE score is computed by using four manual summaries from different assessors, so that we can estimate inter-subject disagreement. If one computes the ROUGE score of a held-out summary against the remaining three summaries, the resulting performance is given in the row labeled *human* of Figure 5. It provides a reasonable estimate of human performance.

Second, in extractive summarization we restrict summaries to sentences from the documents themselves, which is likely to lead to a reduction in ROUGE. To estimate this drop, we use the greedy algorithm to select the extractive summary that maximizes ROUGE on the test documents. The resulting performance is given in the row *extractive* of Figure 5. On both dataset, the drop in performance for this (approximately[3]) optimal extractive summary is about 10 points of ROUGE.

Third, we expect some drop in performance, since our model may not be able to fit the optimal extractive summaries due to a lack of expressiveness. This can be estimated by looking at training set performance, as reported in row *model fit* of Figure 5. On both datasets, we see a drop of about 5 points of ROUGE performance. Adding more and better features might help the model fit the data better.

Finally, a last drop in performance may come from overfitting. The test set ROUGE scores are given in the row *prediction* of Figure 5. Note that the drop between training and test performance is rather small, so overfitting is not an issue and is well controlled in our algorithm. We therefore conclude that increasing model fidelity seems like a promising direction for further improvements.

bound	dataset	ROUGE-1 F
human	DUC '03	0.56235
extractive		0.45497
model fit		0.40873
prediction		0.39294
human	DUC '04	0.55221
extractive		0.45199
model fit		0.40963
prediction		0.40662

Figure 5: Upper bounds on ROUGE-1 F scores: agreement between manual summaries, greedily computed best extractive summaries, best model fit on the train set (using the best C value) and the test scores of the pairwise model.

5.4 Which features are most useful?

To understand which features affected the final performance of our approach, we assessed the strength of each set of our features. In particular, we looked at how the final test score changes when we removed certain features groups (described in the beginning of Section 5) as shown in Figure 6.

The most important group of features are the *basic* features (pure cosine similarity between sentences) since removing them results in the largest drop in performance. However, other features play a significant role too (i.e. only the basic ones are not enough to achieve good perfor-

[3]We compared the greedy algorithm with exhaustive search for up to three selected sentences (more than that would take too long). In about half the cases we got the same solution, in other cases the soultion was on average about 1% below optimal confirming that greedy selection works quite well.

mance). This confirms that performance can be improved by adding richer fatures instead of using only a single similarity score as in Lin and Bilmes (2010). Using learning for these complex model is essential, since hand-tuning is likely to be intractable.

The second most important group of features considering the drop in performance (i.e. *location*) looks at positions of sentences in the articles. This makes intuitive sense because the first sentences in news articles are usually packed with information. The other three groups do not have a significant impact on their own.

removed group	ROUGE-1 F
none	0.40662
basic	**0.38681**
all except basic	**0.39723**
location	**0.39782**
sent+doc	0.39901
cap+stop+len	0.40273
minmax	0.40721

Figure 6: Effects of removing different feature groups on the DUC '04 dataset. Bold font marks significant difference ($p \leq 0.05$) when compared to the full pariwise model. The most important are basic similarity features including all words (similar to (Lin and Bilmes, 2010)). The last feature group actually lowered the score but is included in the model because we only found this out later on DUC '04 dataset.

5.5 How important is it to train with multiple summaries?

While having four manual summaries may be important for computing a reliable ROUGE score for evaluation, it is not clear whether such an approach is the most efficient use of annotator resources for training. In our final experiment, we trained our method using only a single manual summary for each set of documents. When using only a single manual summary, we arbitrarily took the first one out of the provided four reference summaries and used only it to compute the target label for training (instead of using average loss towards all four of them). Otherwise, the experimental setup was the same as in the previous subsections, using the pairwise model.

For DUC '04, the ROUGE-1 F score obtained using only a single summary per document set was 0.4010, which is slightly but not significantly lower than the 0.4066 obtained with four summaries (as shown on Figure 3). Similarly, on DUC '03 the performance drop from 0.3929 to 0.3838 was not significant as well.

Based on those results, we conjecture that having more documents sets with only a single manual summary is more useful for training than fewer training examples with better labels (i.e. multiple summaries). In both cases, we spend approximately the same amount of effort (as the summaries are the most expensive component of the training data), however having more training examples helps (according to the learning curve presented before) while spending effort on multiple summaries appears to have only minor benefit for training.

6 Conclusions

This paper presented a supervised learning approach to extractive document summarization based on structual SVMs. The learning method applies to all submodular scoring functions, ranging from pairwise-similarity models to coverage-based approaches. The learning problem is formulated into a convex quadratic program and was then solved approximately using a cutting-plane method. In an empirical evaluation, the structural SVM approach significantly outperforms conventional hand-tuned models on the DUC '03 and '04 datasets. A key advantage of the learning approach is its ability to handle large numbers of features, providing substantial flexibility for building high-fidelity summarization models. Furthermore, it shows good control of overfitting, making it possible to train models even with only a few training examples.

Acknowledgments

We thank Claire Cardie and the members of the Cornell NLP Seminar for their valuable feedback. This research was funded in part through NSF Awards IIS-0812091 and IIS-0905467.

References

T. Berg-Kirkpatrick, D. Gillick and D. Klein. *Jointly Learning to Extract and Compress.* In Proceedings of ACL, 2011.

S. Brin and L. Page. *The Anatomy of a Large-Scale*

Hypertextual Web Search Engine. In Proceedings of WWW, 1998.

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

J. M. Conroy and D. P. O'leary. *Text summarization via hidden markov models.* In Proceedings of SIGIR, 2001.

H. Daumé III. *Practical Structured Learning Techniques for Natural Language Processing.* Ph.D. Thesis, 2006.

G. Erkan and D. R. Radev. *LexRank: Graph-based Lexical Centrality as Salience in Text Summarization.* In Journal of Artificial Intelligence Research, Vol. 22, 2004, pp. 457–479.

E. Filatova and V. Hatzivassiloglou. *Event-Based Extractive Summarization.* In Proceedings of ACL Workshop on Summarization, 2004.

T. Finley and T. Joachims. *Training structural SVMs when exact inference is intractable.* In Proceedings of ICML, 2008.

D. Gillick and Y. Liu. *A scalable global model for summarization.* In Proceedings of ACL Workshop on Integer Linear Programming for Natural Language Processing, 2009.

J. Goldstein, V. Mittal, J. Carbonell, and M. Kantrowitz. *Multi-document summarization by sentence extraction.* In Proceedings of NAACL-ANLP, 2000.

S. Khuller, A. Moss and J. Naor. *The budgeted maximum coverage problem.* In Information Processing Letters, Vol. 70, Issue 1, 1999, pp. 39–45.

J. M. Kleinberg. *Authoritative sources in a hyperlinked environment.* In Journal of the ACM, Vol. 46, Issue 5, 1999, pp. 604-632.

A. Kulesza and B. Taskar. *Learning Determinantal Point Processes.* In Proceedings of UAI, 2011.

J. Kupiec, J. Pedersen, and F. Chen. *A trainable document summarizer.* In Proceedings of SIGIR, 1995.

L. Li, Ke Zhou, G. Xue, H. Zha, and Y. Yu. *Enhancing Diversity, Coverage and Balance for Summarization through Structure Learning.* In Proceedings of WWW, 2009.

H. Lin and J. Bilmes. 2010. *Multi-document summarization via budgeted maximization of submodular functions.* In Proceedings of NAACL-HLT, 2010.

H. Lin and J. Bilmes. 2011. *A Class of Submodular Functions for Document Summarization.* In Proceedings of ACL-HLT, 2011.

C. Y. Lin and E. Hovy. *Automatic evaluation of summaries using N-gram co-occurrence statistics.* In Proceedings of NAACL, 2003.

F. T. Martins and N. A. Smith. *Summarization with a joint model for sentence extraction and compression.* In Proceedings of ACL Workshop on Integer Linear Programming for Natural Language Processing, 2009.

R. McDonald. 2007. *A Study of Global Inference Algorithms in Multi-document Summarization.* In Advances in Information Retrieval, Lecture Notes in Computer Science, 2007, pp. 557–564.

D. Metzler and T. Kanungo. *Machine learned sentence selection strategies for query-biased summarization.* In Proceedings of SIGIR, 2008.

R. Mihalcea. 2004. *Graph-based ranking algorithms for sentence extraction, applied to text summarization.* In Proceedings of the ACL on Interactive poster and demonstration sessions, 2004.

R. Mihalcea and P. Tarau. *Textrank: Bringing order into texts.* In Proceedings of EMNLP, 2004.

T. Nomoto and Y. Matsumoto. *A new approach to unsupervised text summarization.* In Proceedings of SIGIR, 2001.

V. Qazvinian, D. R. Radev, and A. Özgür. 2010. *Citation Summarization Through Keyphrase Extraction.* In Proceedings of COLING, 2010.

K. Raman, T. Joachims and P. Shivaswamy. *Structured Learning of Two-Level Dynamic Rankings.* In Proceedings of CIKM, 2011.

G. Salton and C. Buckley. *Term-weighting approaches in automatic text retrieval.* In Information processing and management, 1988, pp. 513–523.

D. Shen, J. T. Sun, H. Li, Q. Yang, and Z. Chen. *Document summarization using conditional random fields.* In Proceedings of IJCAI, 2007.

A. Swaminathan, C. V. Mathew and D. Kirovski. *Essential Pages.* In Proceedings of WI-IAT, IEEE Computer Society, 2009.

I. Tsochantaridis, T. Hofmann, T. Joachims and Y. Altun. *Large margin methods for structured and interdependent output variables.* In Journal of Machine Learning Research, Vol. 6, 2005, pp. 1453-1484.

X. Wan, J. Yang, and J. Xiao. *Collabsum: Exploiting multiple document clustering for collaborative single document summarizations.* In Proceedings of SIGIR, 2007.

Y. Yue and T. Joachims. *Predicting diverse subsets using structural svms.* In Proceedings of ICML, 2008.