Summarization with a Joint Model for Sentence Extraction and Compression

André F. T. Martins[*†] and **Noah A. Smith**[*]
[*]School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213, USA
[†]Instituto de Telecomunicações, Instituto Superior Técnico, Lisboa, Portugal
{afm,nasmith}@cs.cmu.edu

Abstract

Text summarization is one of the oldest problems in natural language processing. Popular approaches rely on extracting relevant sentences from the original documents. As a side effect, sentences that are too long but partly relevant are doomed to either not appear in the final summary, or prevent inclusion of other relevant sentences. Sentence compression is a recent framework that aims to select the shortest subsequence of words that yields an informative and grammatical sentence. This work proposes a one-step approach for document summarization that jointly performs sentence extraction and compression by solving an integer linear program. We report favorable experimental results on newswire data.

1 Introduction

Automatic text summarization dates back to the 1950s and 1960s (Luhn, 1958; Baxendale, 1958; Edmundson, 1969). Today, the proliferation of digital information makes research on summarization technologies more important than ever before. In the last two decades, machine learning techniques have been employed in **extractive summarization** of single documents (Kupiec et al., 1995; Aone et al., 1999; Osborne, 2002) and multiple documents (Radev and McKeown, 1998; Carbonell and Goldstein, 1998; Radev et al., 2000). Most of this work aims only to *extract* relevant sentences from the original documents and present them as the summary; this simplification of the problem yields scalable solutions.

Some attention has been devoted by the NLP community to the related problem of **sentence compression** (Knight and Marcu, 2000): given a long sentence, how to maximally *compress* it into a grammatical sentence that still preserves all the relevant information? While sentence compression is a promising framework with applications, for example, in headline generation (Dorr et al., 2003; Jin, 2003), little work has been done to include it as a module in document summarization systems. Most existing approaches (with some exceptions, like the vine-growth model of Daumé, 2006) use a two-stage architecture, either by first extracting a certain number of salient sentences and then feeding them into a sentence compressor, or by first compressing all sentences and extracting later. However, regardless of which operation is performed first—compression or extraction—two-step “pipeline” approaches may fail to find overall-optimal solutions; often the summaries are not better that the ones produced by extractive summarization. On the other hand, a pilot study carried out by Lin (2003) suggests that summarization systems that perform sentence compression have the potential to beat pure extractive systems if they model cross-sentence effects.

In this work, we address this issue by merging the tasks of sentence extraction and sentence compression into a *global* optimization problem. A careful design of the objective function encourages “sparse solutions,” i.e., solutions that involve only a small number of sentences whose compressions are to be included in the summary. Our contributions are:

- We cast joint sentence extraction and compression as an integer linear program (ILP);
- We provide a new formulation of sentence compression using dependency parsing information that only requires a linear number of variables, and combine it with a bigram model;
- We show how the full model can be trained in a max-margin framework. Since a dataset of summaries comprised of extracted, compressed sentences is unavailable, we present a procedure that trains the compression and extraction models separately and tunes a parameter to interpolate the

two models.

The compression model and the full system are compared with state-of-the-art baselines in standard newswire datasets. This paper is organized as follows: §2–3 provide an overview of our two building blocks, sentence extraction and sentence compression. §4 describes our method to perform one-step sentence compression and extraction. §5 shows experiments in newswire data. Finally, §6 concludes the paper and suggests future work.

2 Extractive summarization

Extractive summarization builds a summary by extracting a few informative sentences from the documents. Let $D \triangleq \{t_1, \ldots, t_M\}$ be a set of sentences, contained in a single or in multiple related documents.[1] The goal is to extract the best sequence of sentences $\langle t_{i_1}, \ldots, t_{i_K}\rangle$ that summarizes D whose total length does not exceed a fixed budget of J words. We describe some well-known approaches that wil serve as our experimental baselines.

Extract the leading sentences (Lead). For single-document summarization, the simplest method consists of greedily extracting the leading sentences while they fit into the summary. A sentence is skipped if its inclusion exceeds the budget, and the next is examined. This performs extremely well in newswire articles, due to the journalistic convention of summarizing the article first.

Rank by relevance (Rel). This method ranks sentences by a relevance score, and then extracts the top ones that can fit into the summary. The score is typically a linear function of feature values:

$$\text{score}_{\text{rel}}(t_i) \triangleq \boldsymbol{\theta}^\top \mathbf{f}(t_i) = \textstyle\sum_{d=1}^{D} \theta_d f_d(t_i), \qquad (1)$$

Here, each $f_d(t_i)$ is a feature extracted from sentence t_i, and θ_d is the corresponding weight. In our experiments, relevance features include (i) the reciprocal position in the document, (ii) a binary feature indicating whether the sentence is the first one, and (iii) the 1-gram and 2-gram cosine similarity with the headline and with the full document.

Maximal Marginal Relevance (MMR). For long documents or large collections, it becomes important to penalize the *redundancy* among the extracted sentences. Carbonell and Goldstein (1998) proposed greedily adding sentences to the summary S to maximize, at each step, a score of the form

$$\lambda \cdot \text{score}_{\text{rel}}(t_i) - (1-\lambda) \cdot \text{score}_{\text{red}}(t_i, S), \qquad (2)$$

where $\text{score}_{\text{rel}}(t_i)$ is as in Eq. 1 and $\text{score}_{\text{red}}(t_i, S)$ accounts for the *redundancy* between t_i and the current summary S. In our experiments, redundancy is the 1-gram cosine similarity between the sentence t_i and the current summary S. The trade-off between relevance and redundancy is controlled by $\lambda \in [0, 1]$, which is tuned on development data.

McDonald (2007) proposed a non-greedy variant of MMR that takes into account the redundancy between each pair of candidate sentences. This is cast as a global optimization problem:

$$\begin{aligned} \hat{S} \;\; = \;\; & \arg\max_{S} \lambda \cdot \textstyle\sum_{t_i \in S} \text{score}_{\text{rel}}(t_i) - \\ & (1-\lambda) \cdot \textstyle\sum_{t_i, t_j \in S} \text{score}_{\text{red}}(t_i, t_j), \quad (3) \end{aligned}$$

where $\text{score}_{\text{rel}}(t_i) \triangleq \boldsymbol{\theta}_{\text{rel}}^\top \mathbf{f}_{\text{rel}}(t_i)$, $\text{score}_{\text{red}}(t_i, t_j) \triangleq \boldsymbol{\theta}_{\text{red}}^\top \mathbf{f}_{\text{red}}(t_i, t_j)$, and $\mathbf{f}_{\text{rel}}(t_i)$ and $\mathbf{f}_{\text{red}}(t_i, t_j)$ are feature vectors with corresponding learned weight vectors $\boldsymbol{\theta}_{\text{rel}}$ and $\boldsymbol{\theta}_{\text{red}}$. He has shown how the relevance-based method and the MMR framework (in the non-greedy form of Eq. 3) can be cast as an ILP. By introducing indicator variables $\langle \mu_i \rangle_{i=1,\ldots,M}$ and $\langle \mu_{ij} \rangle_{i,j=1,\ldots,M}$ with the meanings

$$\begin{aligned} \mu_i \;\; &= \;\; \begin{cases} 1 & \text{if } t_i \text{ is to be extracted} \\ 0 & \text{otherwise} \end{cases} \\ \mu_{ij} \;\; &= \;\; \begin{cases} 1 & \text{if } t_i \text{ and } t_j \text{ are both to be extracted} \\ 0 & \text{otherwise} \end{cases} \end{aligned} \qquad (4)$$

one can reformulate Eq. 3 as an ILP with $O(M^2)$ variables and constraints:

$$\begin{aligned} \max_{\langle \mu_i \rangle, \langle \mu_{ij} \rangle} \quad & \lambda \cdot \textstyle\sum_{i=1}^{M} \mu_i \text{score}_{\text{rel}}(t_i) - \qquad (5) \\ & (1-\lambda) \cdot \textstyle\sum_{i=1}^{M} \sum_{j=1}^{M} \mu_{ij} \text{score}_{\text{red}}(t_i, t_j), \end{aligned}$$

subject to binary constraints $\mu_i, \mu_{ij} \in \{0, 1\}$, the length constraint $\sum_{i=1}^{M} \mu_i N_i \leq J$ (where N_i is the number of words of the ith sentence), and the following "agreement constraints" for $i, j = 1, \ldots, M$

[1] For simplicity, we describe a unified framework for single and multi-document summarization, although they may require specialized strategies. Here we experiment only with single-document summarization and assume $t_1, ..., t_M$ are ordered.

(that impose the logical relation $\mu_{ij} = \mu_i \wedge \mu_j$):

$$\mu_{ij} \leq \mu_i, \quad \mu_{ij} \leq \mu_j, \quad \mu_{ij} \geq \mu_i + \mu_j - 1 \quad (6)$$

Let us provide a compact representation of the program in Eq. 5 that will be used later. Define our vector of parameters as $\boldsymbol{\theta} \triangleq [\lambda\boldsymbol{\theta}_{\text{rel}}, -(1-\lambda)\boldsymbol{\theta}_{\text{red}}]$. Packing all the feature vectors (one for each sentence, and one for each pair of sentences) into a matrix $\mathbf{F}$,

$$\mathbf{F} \triangleq \begin{bmatrix} \mathbf{F}_{\text{rel}} & 0 \\ 0 & \mathbf{F}_{\text{red}} \end{bmatrix}, \quad (7)$$

with $\mathbf{F}_{\text{rel}} \triangleq [\mathbf{f}_{\text{rel}}(t_i)]_{1 \leq i \leq M}$ and $\mathbf{F}_{\text{red}} \triangleq [\mathbf{f}_{\text{red}}(t_i, t_j)]_{1 \leq i < j \leq M}$, and packing all the variables μ_i and μ_{ij} into a vector $\boldsymbol{\mu}$, the program in Eq. 5 can be compactly written as

$$\max_{\boldsymbol{\mu}} \boldsymbol{\theta}^\top \mathbf{F} \boldsymbol{\mu}, \quad (8)$$

subject to binary and linear constraints on $\boldsymbol{\mu}$. This formulation requires $O(M^2)$ variables and constraints. If we do not penalize sentence redundancy, the redundancy term may be dropped; in this simpler case, $\mathbf{F} = \mathbf{F}_{\text{rel}}$, the vector $\boldsymbol{\mu}$ only contains the variables $\langle \mu_i \rangle$, and the program in Eq. 8 only requires $O(M)$ variables and constraints. Our method (to be presented in §4) will build on this latter formulation.

3 Sentence Compression

Despite its simplicity, extractive summarization has a few shortcomings: for example, if the original sentences are too long or embed several clauses, there is no way of preventing lengthy sentences from appearing in the final summary. The **sentence compression** framework (Knight and Marcu, 2000) aims to select the best *subsequence* of words that still yields a short, informative and grammatical sentence. Such a sentence compressor is given a sentence $t \triangleq \langle w_1, \ldots, w_N \rangle$ as input and outputs a subsequence of length L, $c \triangleq \langle w_{j_1}, \ldots, w_{j_L} \rangle$, with $1 \leq j_1 < \ldots < j_L \leq N$. We may represent this output as a binary vector $\mathbf{s}$ of length N, where $s_j = 1$ iff word w_j is included in the compression. Note that there are $O(2^N)$ possible subsequences.

3.1 Related Work

Past approaches to sentence compression include a noisy channel formulation (Knight and Marcu, 2000; Daumé and Marcu, 2002), heuristic methods that parse the sentence and then trim constituents according to linguistic criteria (Dorr et al., 2003; Zajic et al., 2006), a pure discriminative model (McDonald, 2006), and an ILP formulation (Clarke and Lapata, 2008). We next give an overview of the two latter approaches.

McDonald (2006) uses the outputs of two parsers (a phrase-based and a dependency parser) as features in a discriminative model that decomposes over pairs of consecutive words. Formally, given a sentence $t = \langle w_1, \ldots, w_N \rangle$, the score of a compression $c = \langle w_{j_1}, \ldots, w_{j_L} \rangle$ decomposes as:

$$\text{score}(c; t) = \textstyle\sum_{l=2}^{L} \boldsymbol{\theta}^\top \mathbf{f}(t, j_{l-1}, j_l) \quad (9)$$

where $\mathbf{f}(t, j_{l-1}, j_l)$ are feature vectors that depend on the original sentence t and consecutive positions j_{l-1} and j_l, and $\boldsymbol{\theta}$ is a learned weight vector. The factorization in Eq. 9 allows exact decoding with dynamic programming.

Clarke and Lapata (2008) cast the problem as an ILP. In their formulation, Eq. 9 may be expressed as:

$$\begin{aligned} \text{score}(c; t) \;=\; & \sum_{i=1}^{N} \alpha_i \boldsymbol{\theta}^\top \mathbf{f}(t, 0, i) + \\ & \sum_{i=1}^{N} \beta_i \boldsymbol{\theta}^\top \mathbf{f}(t, i, n+1) + \\ & \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \gamma_{ij} \boldsymbol{\theta}^\top \mathbf{f}(t, i, j), \quad (10) \end{aligned}$$

where α_i, β_i, and γ_{ij} are additional binary variables with the following meanings:

- $\alpha_i = 1$ iff word w_i starts the compression;
- $\beta_i = 1$ iff word w_i ends the compression;
- $\gamma_{ij} = 1$ iff words w_i and w_j appear consecutively in the compression;

and subject to the following agreement constraints:

$$\begin{aligned} \textstyle\sum_{i=1}^{N} \alpha_i &= 1 \\ \textstyle\sum_{i=1}^{N} \beta_i &= 1 \\ s_j &= \alpha_j + \textstyle\sum_{i=1}^{j-1} \gamma_{ij} \\ s_i &= \beta_i + \textstyle\sum_{j=i+1}^{N} \gamma_{ij}. \quad (11) \end{aligned}$$

This framework also allows the inclusion of constraints to enforce grammaticality.

To compress a sentence, one needs to maximize the score in Eq. 10 subject to the constraints in Eq. 11. Representing the variables through

$$\boldsymbol{\nu} \triangleq \langle \alpha_1, \ldots, \alpha_N, \beta_1, \ldots, \beta_N, \gamma_{11}, \ldots, \gamma_{NN} \rangle \tag{12}$$

and packing the feature vectors into a matrix $\mathbf{F}$, we obtain the ILP

$$\max_{\mathbf{s}, \boldsymbol{\nu}} \boldsymbol{\theta}^\top \mathbf{F} \boldsymbol{\nu} \tag{13}$$

subject to linear and integer constraints on the variables $\mathbf{s}$ and $\boldsymbol{\nu}$. This particular formulation requires $O(N^2)$ variables and constraints.

3.2 Proposed Method

We propose an alternative model for sentence compression that may be formulated as an ILP, as in Eq. 13, but with only $O(N)$ variables and constraints. This formulation is based on the output of a dependency parser.

Directed arcs in a dependency tree link pairs of words, namely a *head* to its *modifier*. A dependency parse tree is characterized by a set of labeled arcs of the form (head, modifier, label); see Fig.1 for an example. Given a sentence $t = \langle w_1, \ldots, w_N \rangle$, we write $i = \pi(j)$ to denote that the ith word is the head (the "parent") of the jth word; if j is the root, we write $\pi(j) = 0$. Let $\mathbf{s}$ be the binary vector describing a possible compression c for the sentence t. For each word j, we consider four possible cases, accounting for the inclusion or not of j and $\pi(j)$ in the compression. We introduce (mutually exclusive) binary variables ν_{j11}, ν_{j10}, ν_{j01}, and ν_{j00} to indicate each of these cases, i.e., for $a, b \in \{0, 1\}$,

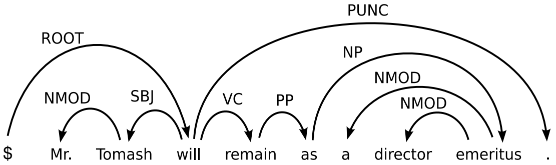

Figure 1: A dependency parse for an English sentence; example from McDonald and Satta (2007).

$$\nu_{jab} \;\triangleq\; s_j = a \wedge s_{\pi(j)} = b. \tag{14}$$

Consider feature vectors $\mathbf{f}_{11}(t, j)$, $\mathbf{f}_{10}(t, j)$, $\mathbf{f}_{01}(t, j)$, and $\mathbf{f}_{00}(t, j)$, that look at the surface sentence and at the status of the word j and its head $\pi(j)$; these features have corresponding weight vectors $\boldsymbol{\theta}_{11}$, $\boldsymbol{\theta}_{10}$, $\boldsymbol{\theta}_{01}$, and $\boldsymbol{\theta}_{00}$. The score of c is written as:

$$\begin{aligned} \text{score}(c; t) &= \textstyle\sum_{j=1}^{N} \sum_{a,b \in \{0,1\}} \nu_{jab} \boldsymbol{\theta}_{ab}^\top \mathbf{f}_{ab}(t, j) \\ &= \textstyle\sum_{a,b \in \{0,1\}} \boldsymbol{\theta}_{ab}^\top \mathbf{F}_{ab} \boldsymbol{\nu}_{ab} \\ &= \boldsymbol{\theta}^\top \mathbf{F} \boldsymbol{\nu}, \end{aligned} \tag{15}$$

where $\mathbf{F}_{ab} \triangleq [\mathbf{f}_{ab}(t, 1), \ldots, \mathbf{f}_{ab}(t, N)]$, $\boldsymbol{\nu}_{ab} \triangleq (\nu_{jab})_{j=1,\ldots,N}$, $\boldsymbol{\theta} \triangleq (\boldsymbol{\theta}_{11}, \boldsymbol{\theta}_{10}, \boldsymbol{\theta}_{01}, \boldsymbol{\theta}_{00})$, and $\mathbf{F} \triangleq \text{Diag}(\mathbf{F}_{11}, \mathbf{F}_{10}, \mathbf{F}_{01}, \mathbf{F}_{00})$ (a block-diagonal matrix).

We have reached in Eq. 15 an ILP isomorphic to the one in Eq. 13, but only with $O(N)$ variables. There are some agreement constraints between the variables $\boldsymbol{\nu}$ and $\mathbf{s}$ that reflect the logical relations in Eq. 14; these may be written as linear inequalities (cf. Eq. 6), yielding $O(N)$ constraints.

Given this proposal and §3.1, it is also straightforward to extend this model to include bigram features as in Eq. 10; the combination of dependency relation features and bigram features yields a model that is more powerful than both models in Eq. 15 and Eq. 10. Such a model is expressible as an ILP with $O(N^2)$ variables and constraints, making use of the variables $\mathbf{s}$, $\boldsymbol{\nu}$, $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$. In §5, we compare the performance of this model (called "Bigram") and the model in Eq. 15 (called "NoBigram").[2]

4 Joint Compression and Extraction

We next describe our joint model for sentence compression and extraction. Let $D \triangleq \{t_1, \ldots, t_M\}$ be a set of sentences as in §2, each expressed as a sequence of words, $t_i \triangleq \langle w_{i1}, \ldots, w_{iN_i} \rangle$. Following §3, we represent a *compression* of t_i as a binary vector $\mathbf{s}_i = \langle s_{i1}, \ldots, s_{iN_i} \rangle$, where $s_{ij} = 1$ iff word w_{ij}

[2] It should be noted that more efficient decoders are possible that do not require solving an ILP. In particular, inference in the NoBigram variant can performed in polynomial time with dynamic programming algorithms that propagate messages along the dependency parse tree; for the Bigram variant, dynamic programming can still be employed with some additional storage. Our ILP formulation, however, is more suited to the final goal of performing document summarization (of which our sentence compression model will be a component); furthermore, it also allows the straightforward inclusion of global linguistic constraints, which, as shown by Clarke and Lapata (2008), can greatly improve the grammaticality of the compressions.

is included in the compression. Now, define a *summary* of D as a set of sentences obtained by *extracting* and *compressing* sentences from D. More precisely, let $\mu_1, \ldots, \mu_M$ be binary variables, one for each sentence t_i in D; define $\mu_i = 1$ iff a compression of sentence t_i is used in the summary. A summary of D is then represented by the binary variables $\langle \mu_1, \ldots, \mu_M, \mathbf{s}_1, \ldots, \mathbf{s}_M \rangle$. Notice that these variables are redundant:

$$\mu_i = 0 \quad \Leftrightarrow \quad \forall j \in \{1, \ldots, N_i\} \quad s_{ij} = 0, \quad (16)$$

i.e., an empty compression means that the sentence is not to be extracted. In the sequel, it will become clear why this redundancy is convenient.

Most approaches up to now are concerned with either *extraction* or *compression*, not both at the same time. We will combine the extraction scores in Eq. 8 and the compression scores in Eq. 15 to obtain a single, global optimization problem;[3] we rename the extraction features and parameters to $\mathbf{F}_e$ and $\boldsymbol{\theta}_e$ and the compression features and parameters to $\mathbf{F}_c$ and $\boldsymbol{\theta}_c$:

$$\max_{\boldsymbol{\mu},\boldsymbol{\nu},\mathbf{s}} \boldsymbol{\theta}_e^T \mathbf{F}_e \boldsymbol{\mu} + \textstyle\sum_{i=1}^M \boldsymbol{\theta}_c^T \mathbf{F}_{ci} \boldsymbol{\nu}_i, \quad (17)$$

subject to agreement constraints on the variables $\boldsymbol{\nu}_i$ and $\mathbf{s}_i$ (see Eqs. 11 and 14), and new agreement constraints on the variables $\boldsymbol{\mu}$ and $\mathbf{s}_1, \ldots, \mathbf{s}_M$ to enforce the relation in Eq. 16:

$$\begin{array}{rcl} s_{ij} & \leq & \mu_i, \quad \forall i = 1, \ldots, M, \forall j = 1, \ldots, N_i \\ \mu_i & \leq & \sum_{j=1}^{N_i} s_{ij}, \quad \forall i = 1, \ldots, M \end{array} \quad (18)$$

The constraint that the length of the summary cannot exceed J words is encoded as:

$$\textstyle\sum_{i=1}^M \sum_{j=1}^{N_i} s_{ij} \leq J. \quad (19)$$

All variables are further restricted to be binary. We also want to avoid picking just a few words from many sentences, which typically leads to ungrammatical summaries. Hence it is desirable to obtain "sparse" solutions with only a few sentences extracted and compressed (and most components of $\boldsymbol{\mu}$ are zero) To do so, we add the constraint

$$\textstyle\sum_{j=1}^{N_i} s_{ij} \geq \mu_i \rho N_i, \quad i = 1, \ldots, M, \quad (20)$$

which states, for each sentence t_i, that t_i should be ignored or have at least ρN_i words extracted. We fix $\rho = 0.8$, enforcing compression rates below 80%.[4]

To learn the model parameters $\boldsymbol{\theta} = \langle \boldsymbol{\theta}_e, \boldsymbol{\theta}_c \rangle$, we can use a max-margin discriminative learning algorithm like MIRA (Crammer and Singer, 2003), which is quite effective and scalable. However, there is not (to our knowledge) a single dataset of extracted *and* compressed sentences. Instead, as will be described in Sec. 5.1, there are separate datasets of extracted sentences, and datasets of compressed sentences. Therefore, instead of globally learning the model parameters, $\boldsymbol{\theta} = \langle \boldsymbol{\theta}_e, \boldsymbol{\theta}_c \rangle$, we propose the following strategy to learn them separately:

- Learn $\boldsymbol{\theta}'_e$ using a corpus of extracted sentences,
- Learn $\boldsymbol{\theta}'_c$ using a corpus of compressed sentences,
- Tune η so that $\boldsymbol{\theta} = \langle \boldsymbol{\theta}'_e, \eta \boldsymbol{\theta}'_c \rangle$ has good performance on development data. (This is necessary since each set of weights is learned up to scaling.)

5 Experiments

5.1 Datasets, Evaluation and Environment

For our experiments, two datasets were used:

The DUC 2002 dataset. This is a collection of newswire articles, comprised of 59 document clusters. Each document within the collections (out of a total of 567 documents) has one or two manually created abstracts with approximately 100 words.[5]

Clarke's dataset for sentence compression. This is the dataset used by Clarke and Lapata (2008). It contains manually created compressions of 82 newspaper articles (1,433 sentences) from the British National Corpus and the American News Text corpus.[6]

To evaluate the sentence compressor alone, we measured the compression rate and the precision, recall, and F_1-measure (both macro and micro-averaged) with respect to the "gold" compressed

[3] In what follows, we use the formulation in Eq. 8 *without* the redundancy terms; however these can be included in a straightforward way, naturally increasing the number of variables/constraints.

[4] There are alternative ways to achieve "sparseness," either in a soft way, by adding a term $-\lambda \sum_i \mu_i$ to the objective, or using a different hard constraint, like $\sum_i \mu_i \leq K$, to limit the number of sentences from which to pick words.

[5] `http://duc.nist.gov`

[6] `http://homepages.inf.ed.ac.uk/s0460084/data`

	Compression	Micro-Av.			Macro-Av.		
	Ratio	P	R	F_1	P	R	F_1
HedgeTrimmer	57.64%	0.7099	0.5925	0.6459	0.7195	0.6547	0.6367
McDonald (2006)	71.40%	0.7444	0.7697	0.7568	0.7711	0.7852	**0.7696**
NoBigram	71.20%	0.7399	0.7626	0.7510	0.7645	0.7730	**0.7604**
Bigram	71.35%	0.7472	0.7720	0.7594	0.7737	0.7848	**0.7710**

Table 1: Results for sentence compression in the Clarke's test dataset (441 sentences) for our implementation of the baseline systems (*HedgeTrimmer* and the system described in McDonald, 2006), and the two variants of our model, NoBigram and Bigram. The compression ratio associated with the reference compressed sentences in this dataset is 69.06%. In the rightmost column, the statistically indistinguishable best results are emboldened, based on a paired t-test applied to the sequence of F_1 measures ($p < 0.01$).

sentences, calculated on unigrams.[7]

To evaluate the full system, we used Rouge-N (Lin and Hovy, 2002), a popular n-gram recall-based automatic evaluation measure. This score compares the summary produced by a system with one or more valid reference summaries.

All our experiments were conducted on a PC with a Intel dual-core processor with 2.66 GHz and 2 Gb RAM memory. We used ILOG CPLEX, a commercial integer programming solver. The interface with CPLEX was coded in Java.

5.2 Sentence Compression

We split Clarke's dataset into two partitions, one used for training (1,188 sentences) and the other for testing (441 sentences). This dataset includes one manual compression for each sentence, that we use as reference for evaluation purposes. Compression ratio, i.e., the fraction of words included in the compressed sentences, is 69.32% (micro-averaged over the training partition).

For comparison, two baselines were implemented: a simple compressor based on Hedge Trimmer, the headline generation system of Dorr et al. (2003) and Zajic et al. (2006),[8] and the discriminative model described by McDonald (2006), which captures "soft syntactic evidence" (we reproduced the same set of features). Both systems require a phrase-structure parser; we used Collins' parser (Collins, 1999);[9] the latter system also derives features from a dependency parser; we used the MST-Parser (McDonald et al., 2005).[10]

We implemented the two variants of our compressor described in §3.2.

NoBigram. This variant factors the compression score as a sum over individual scores, each depending on the inclusion or not of each word and its head in the compression (see Eq. 15). An upper bound of 70% was placed on the compression ratio. As stated in §3.2, inference amounts to solving an ILP with $O(N)$ variables and constraints, N being the sentence length. We also used MSTParser to obtain the dependency parse trees.

Bigram. This variant includes an extra term standing for a *bigram score*, which factors as a sum over pairs of consecutive words. As in McDonald (2006), we include features that depend on the "in-between" words in the original sentence that are to be omitted in the compression.[11] As stated in §3.2, inference through this model can be done by solving an ILP with $O(N^2)$ variables and constraints.

[7] Notice that this evaluation score is not able to properly capture the grammaticality of the compression; this is a known issue that typically is addressed by requiring human judgments.

[8] Hedge Trimmer applies a deterministic compression procedure whose first step is to identify the lowest leftmost S node in the parse tree that contains a NP and a VP; this node is taken as the root of the compressed sentence (i.e., all words that are not spanned by this node are discarded). Further steps described by Dorr et al. (2003) include removal of low content units, and an "iterative shortening" loop that keeps removing constituents until a desired compression ratio is achieved. The best results were obtained without iterative shortening, which is explained by the fact that the selection of the lowest leftmost S node (first step of the algorithm) already provides significant compression, as illustrated in Table 1.

[9] http://people.csail.mit.edu/mcollins/code.html

[10] http://sourceforge.net/projects/mstparser

[11] The major difference between this variant and model of McDonald (2006) is that the latter employs "soft syntactic evidence" as *input* features, while we make the dependency relations part of the *output* features. All the non-syntactic features are the same. Apart from this, notice that our variant does not employ a phrase-structure parser.

For both variants, we used MSTParser to obtain the dependency parse trees. The model parameters are learned in a pure discriminative way through a max-margin approach. We used the 1-best MIRA algorithm (Crammer and Singer, 2003; McDonald et al., 2005) for training; this is a fast online algorithm that requires solving the inference problem at each step. Although inference amounts to solving an ILP, which in the worst case scales exponentially with the size of the sentence, training the model is in practice very fast for the NoBigram model (a few minutes in the environment described in §5.1) and fast enough for the Bigram model (a couple of hours using the same equipment). This is explained by the fact that sentences don't usually exceed a few tens of words, and because of the structure of the ILPs, whose constraint matrices are very sparse.

Table 1 depicts the micro- and macro-averaged precision, recall and F_1-measure. We can see that both variants outperform the Hedge Trimmer baseline by a great margin, and are in line with the system of McDonald (2006); however, none of our variants employ a phrase-structure parser. We also observe that our simpler NoBigram variant, which uses a linear-sized ILP, achieves results similar to these two systems.

5.3 Joint Compression and Extraction

For the summarization task, we split the DUC 2002 dataset into a training partition (427 documents) and a testing partition (140 documents). The training partition was further split into a training and a development set. We evaluated the performance of Lead, Rel, and MMR as baselines (all are described in §2). Weights for Rel were learned via the SVM-Rank algorithm;[12] to create a gold-standard ranking, we sorted the sentences by Rouge-2 score[13] (with respect to the human created summaries). We include a Pipeline baseline as well, which ranks all sentences by relevance, then includes their compressions (using the Bigram variant) while they fit into the summary.

We tested two variants of our joint model, combining the Rel extraction model with (i) the NoBigram compression model (§3.2) and (ii) the Bigram variant. Each variant was trained with the procedure described in §4. To keep tractability, the inference ILP problem was relaxed (the binary constraints were relaxed to unit interval constraints) and non-integer solution values were rounded to produce a valid summary, both for training and testing.[14] Whenever this procedure yielded a summary longer than 100 words, we truncated it to fit the word limit.

	Rouge-1	Rouge-2
Lead	0.384 ± 0.080	**0.177 ± 0.083**
Rel	0.389 ± 0.074	**0.178 ± 0.080**
MMR $\lambda = 0.25$	0.392 ± 0.071	**0.178 ± 0.077**
Pipeline	0.380 ± 0.073	0.173 ± 0.073
Rel + NoBigr $\eta = 1.5$	**0.403 ± 0.080**	**0.180 ± 0.082**
Rel + Bigr $\eta = 4.0$	**0.403 ± 0.076**	**0.180 ± 0.076**

Table 2: Results for sentence extraction in the DUC2002 dataset (140 documents). Bold indicates the best results with statistical significance, according to a paired t-test ($p < 0.01$); Rouge-2 scores of all systems except Pipeline are indistinguishable according to the same test, with $p > 0.05$.

Table 2 depicts the results of each of the above systems in terms of Rouge-1 and Rouge-2 scores. We can see that both variants of our system are able to achieve the best results in terms of Rouge-1 and Rouge-2 scores. The suboptimality of extracting and compressing in separate stages is clear from the table, as Pipeline performs worse than the pure extractive systems. We also note that the configuration Rel + Bigram is not able to outperform Rel + NoBigram, despite being computationally more expensive (about 25 minutes to process the whole test set, against the 7 minutes taken by the Rel + NoBigram variant). Fig. 2 exemplifies the summaries produced by our system. We see that both variants were able to include new pieces of information in the summary without sacrificing grammaticality.

These results suggest that our system, being capable of performing joint sentence extraction and compression to summarize a document, offers a powerful alternative to pure extractive systems. Finally, we note that no labeled datasets currently exist on which our full model could have been trained with supervision; therefore, although inference is performed

[12] SVMRank is implemented in the SVM^{light} toolkit (Joachims, 1999), `http://svmlight.joachims.org`.

[13] A similar system was implemented that optimizes the Rouge-1 score instead, but it led to inferior performance.

[14] See Martins et al. (2009) for a study concerning the impact of LP relaxations in the learning problem.

MMR baseline:

Australian novelist Peter Carey was awarded the coveted Booker Prize for fiction Tuesday night for his love story, “Oscar and Lucinda”.

A panel of five judges unanimously announced the award of the $26,250 prize after an 80-minute deliberation during a banquet at London’s ancient Guildhall.

Carey, who lives in Sydney with his wife and son, said in a brief speech that like the other five finalists he had been asked to attend with a short speech in his pocket in case he won.

Rel + NoBigram:

Australian novelist Peter Carey was awarded the *coveted* Booker Prize for fiction *Tuesday night* for his love story, “Oscar and Lucinda”.

A panel of five judges *unanimously* announced the award of the $26,250 prize after an *80-minute* deliberation during a banquet at London’s ancient Guildhall.

The judges made their selection from 102 books published in Britain in the past 12 months *and which they read in their homes.*

Carey, who lives in Sydney with his wife and son, said *in a brief speech that* like the other five finalists he had been asked to attend with a short speech in his pocket in case he won.

Rel + Bigram:

Australian novelist Peter Carey was awarded the *coveted* Booker Prize for fiction *Tuesday night* for his *love* story, “Oscar and Lucinda”.

A panel of *five* judges *unanimously* announced the award of the $26,250 prize after an *80-minute* deliberation during a banquet at London’s ancient Guildhall.

He was unsuccessful in the prize competition in 1985 when his novel, “Illywhacker,” was among the final six.

Carey called the award a “*great* honor” and he thanked the prize sponsors for “provoking *so much* passionate discussion *about literature _ perhaps* there will be *more* tomorrow”.

Carey was the *only* non-Briton in the final six*.*

Figure 2: Summaries produced by the strongest baseline (MMR) and the two variants of our system. Deleted words are *marked as such*.

jointly, our training procedure had to learn separately the extraction and the compression models, and to tune a scalar parameter to trade off the two models. We conjecture that a better model could have been learned if a labeled dataset with extracted compressed sentences existed.

6 Conclusion and Future Work

We have presented a summarization system that performs sentence extraction and compression in a single step, by casting the problem as an ILP. The summary optimizes an objective function that includes both extraction and compression scores. Our model encourages “sparse” summaries that involve only a few sentences. Experiments in newswire data suggest that our system is a valid alternative to existing extraction-based systems. However, it is worth noting that further evaluation (e.g., human judgments) needs to be carried out to assert the quality of our summaries, e.g., their grammaticality, something that the Rouge scores cannot fully capture.

Future work will address the possibility of including linguistic features and constraints to further improve the grammaticality of the produced summaries.

Another straightforward extension is the inclusion of a redundancy term and a query relevance term in the objective function. For redundancy, a similar idea of that of McDonald (2007) can be applied, yielding a ILP with $O(M^2 + N)$ variables and constraints (M being the number of sentences and N the total number of words). However, such model will take into account the redundancy among the original sentences and not their compressions; to model the redundancy accross compressions, a possibility is to consider a linear redundancy score (similar to cosine similarity, but without the normalization), which would result in an ILP with $O(N + \sum_i P_i^2)$ variables and constraints, where $P_i \leq M$ is the number of sentences in which word w_i occurs; this is no worse than $O(M^2 N)$.

We also intend to model discourse, which, as shown by Daumé and Marcu (2002), plays an important role in document summarization. Another future direction is to extend our ILP formulations to more sophisticated models that go beyond word deletion, like the ones proposed by Cohn and Lapata (2008).

Acknowledgments

The authors thank the anonymous reviewers for helpful comments, Yiming Yang for interesting discussions, and Dipanjan Das and Sourish Chaudhuri for providing their code. This research was supported by a grant from FCT through the CMU-Portugal Program and the Information and Communications Technologies Institute (ICTI) at CMU, and also by Priberam Informática.

References

C. Aone, M. E. Okurowski, J. Gorlinsky, and B. Larsen. 1999. A trainable summarizer with knowledge acquired from robust nlp techniques. In *Advances in Automatic Text Summarization*. MIT Press.

P. B. Baxendale. 1958. Machine-made index for technical literature—an experiment. *IBM Journal of Research Development*, 2(4):354–361.

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

J. Clarke and M. Lapata. 2008. Global inference for sentence compression an integer linear programming approach. *JAIR*, 31:399–429.

T. Cohn and M. Lapata. 2008. Sentence compression beyond word deletion. In *Proc. COLING*.

M. Collins. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania.

K. Crammer and Yoram Singer. 2003. Ultraconservative online algorithms for multiclass problems. *J. Mach. Learn. Res.*, 3:951–991.

H. Daumé and D. Marcu. 2002. A noisy-channel model for document compression. In *Proc. of ACL*.

H. Daumé. 2006. *Practical Structured Learning Techniques for Natural Language Processing*. Ph.D. thesis, University of Southern California.

B. Dorr, D. Zajic, and R. Schwartz. 2003. Hedge trimmer: A parse-and-trim approach to headline generation. In *Proc. of HLT-NAACL Text Summarization Workshop and DUC*.

H. P. Edmundson. 1969. New methods in automatic extracting. *Journal of the ACM*, 16(2):264–285.

R. Jin. 2003. *Statistical Approaches Toward Title Generation*. Ph.D. thesis, Carnegie Mellon University.

T. Joachims. 1999. Making large-scale SVM learning practical. In *Advances in Kernel Methods - Support Vector Learning*. MIT Press.

K. Knight and D. Marcu. 2000. Statistics-based summarization—step one: Sentence compression. In *Proc. of AAAI/IAAI*.

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

C.-Y. Lin and E. Hovy. 2002. Manual and automatic evaluation of summaries. In *Proc. of the ACL Workshop on Automatic Summarization*.

C.-Y. Lin. 2003. Improving summarization performance by sentence compression-a pilot study. In *Proc. of the Int. Workshop on Inf. Ret. with Asian Languages*.

H. P. Luhn. 1958. The automatic creation of literature abstracts. *IBM Journal of Research Development*, 2(2):159–165.

A. F. T. Martins, N. A. Smith, and E. P. Xing. 2009. Polyhedral outer approximations with application to natural language parsing. In *Proc. of ICML*.

R. T. McDonald, F. Pereira, K. Ribarov, and J. Hajič. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proc. of HLT-EMNLP*.

R. McDonald. 2006. Discriminative sentence compression with soft syntactic constraints. In *Proc. of EACL*.

R. McDonald. 2007. A study of global inference algorithms in multi-document summarization. In *Proc. of ECIR*.

M. Osborne. 2002. Using maximum entropy for sentence extraction. In *Proc. of the ACL Workshop on Automatic Summarization*.

D. R. Radev and K. McKeown. 1998. Generating natural language summaries from multiple on-line sources. *Computational Linguistics*, 24(3):469–500.

D. R. Radev, H. Jing, and M. Budzikowska. 2000. Centroid-based summarization of multiple documents: sentence extraction, utility-based evaluation, and user studies. In *Proc. of the NAACL-ANLP Workshop on Automatic Summarization*.

D. Zajic, B. Dorr, J. Lin, and R. Schwartz. 2006. Sentence compression as a component of a multi-document summarization system. In *Proc. of the ACL DUC Workshop*.