Discourse Chunking and its Application to Sentence Compression

Caroline Sporleder and **Mirella Lapata**
School of Informatics
University of Edinburgh
2 Buccleuch Place
Edinburgh EH8 9LW, UK
{csporled,mlap}@inf.ed.ac.uk

Abstract

In this paper we consider the problem of analysing sentence-level discourse structure. We introduce discourse chunking (i.e., the identification of intra-sentential nucleus and satellite spans) as an alternative to full-scale discourse parsing. Our experiments show that the proposed modelling approach yields results comparable to state-of-the-art while exploiting knowledge-lean features and small amounts of discourse annotations. We also demonstrate how discourse chunking can be successfully applied to a sentence compression task.

1 Introduction

The computational treatment of discourse phenomena has recently attracted much attention, partly due to their increasing importance for potential applications. In summarisation, for example, the extraction of sentences to include in a summary crucially depends on their rhetorical status (Marcu, 2000; Teufel and Moens, 2002); one might want to extract contrastive or explanatory statements while omitting sentences that contain background information. In information extraction, discourse-level knowledge can be used to identify co-referring events (Humphreys et al., 1997) and to determine their temporal order. Discourse processing could further enhance question answering systems by interpreting the user's question either in isolation or in the context of preceding questions (Chai and Jing, 2004).

Discourse analysis is often viewed as a parsing task. Rhetorical Structure Theory (RST, Mann and Thomson, 1988), one of the most influential frameworks in discourse processing, represents texts by trees whose leaves correspond to elementary discourse units (*edus*) and whose nodes specify how these and larger units (e.g., multi-sentence segments) are linked to each other by rhetorical relations (e.g., *Contrast*, *Elaboration*). Discourse units are further characterised in terms of their text importance: *nuclei* denote central segments, whereas *satellites* denote peripheral ones.

Recent advances in discourse modelling have greatly benefited from the availability of resources annotated with discourse-level information such as the RST Discourse Treebank (RST-DT, Carlson et al., 2002). Even though discourse parsing at the document-level still poses a significant challenge to data-driven methods, sentence-level discourse models (e.g., Soricut and Marcu, 2003) trained on the RST-DT have attained accuracies comparable to human performance. The availability of discourse annotations is partly responsible for the success of these models. Another important reason is the development of robust syntactic parsers (e.g., Charniak, 2000) that can be used to provide critical structural and lexical information to the discourse parser. Unfortunately, discourse annotated corpora are largely absent for languages other than English. Furthermore, reliance on syntactic parsing renders discourse parsing practically impossible for languages for which state-of-the-art parsers are unavailable.

In this paper we propose discourse chunking as an alternative to discourse parsing. Analogous to sentence chunking, discourse chunking is an intermediate step towards full parsing. Following an RST-style analysis, we focus solely on two subtasks: (a) discourse segmentation, i.e., determining which word sequences form *edus* and (b) inferring whether these *edus* function as nuclei or satellites. The motivation for tackling these subtasks is two-fold. First, they are of crucial importance for full-scale discourse parsing. For example, Soricut and Marcu (2003) show that perfect discourse segmentation delivers an error reduction of 29% in the performance of their discourse parser. Second, some applications may not require full-scale discourse parsing. For example, it has been shown that nuclearity is important

Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP), pages 257–264, Vancouver, October 2005.

for summarisation, i.e., nuclei are more likely to be retained when summarising than satellites (Marcu, 2000). While nuclearity alone may not be sufficient for document summarisation (Marcu, 1998), such knowledge could prove useful at the sentence level, for example for producing sentence compressions.

The algorithms introduced in this paper are purposely knowledge-lean. We abstain from using syntactic parsers or semantic databases such as WordNet (Fellbaum, 1998), thus exploring the portability of our methods to languages for which such resources are not available. We employ lexical and low-level syntactic information (e.g., parts of speech, syntactic chunks) and show that the performance of our discourse chunker on the two subtasks (mentioned above) is comparable to that of a state-of-the-art sentence-level discourse parser (Soricut and Marcu, 2003). We also assess its application potential on a sentence compression task (Knight and Marcu, 2003).

2 Related Work

Initial work towards the development of discourse parsers has primarily relied on hand-crafted rules for specifying world knowledge or constraints on tree structures (e.g., Hobbs 1993). Recent work has seen the emergence of treebanks annotated with discourse structure, thus enabling the development of more robust, data-driven models. Marcu (2000) presents a shift-reduce parsing model that segments texts into *edus* and determines how they should be assembled into rhetorical structure trees. Soricut and Marcu (2003) introduce a syntax-based sentence-level discourse parser, which consists of two components: a statistical segmentation model and a parser working on the output of the segmenter. Both components are trained on the RST-DT and exploit lexical features as well as syntactic dominance features (which are taken from syntactic parse trees).

Given that discourse-level information plays an important role in human summarisation (Endres-Niggemeyer, 1998), it is not surprising that models of discourse structure have found use in automatic summarisation. For instance, Marcu (2000) proposes a summarisation algorithm that builds an RST tree for the entire text, and identifies its most important parts according to discourse salience.

Our work differs from previous approaches in two key respects. First, we do not attempt to produce a hierarchical discourse structure. We introduce discourse chunking, a less resource demanding task than full discourse parsing. We show that good chunking performance can be achieved with low-level information. Second, we apply our discourse chunker to sentence compression. Although previous approaches have utilised discourse information for document summarisation, its application to sentence condensation is novel to our knowledge.

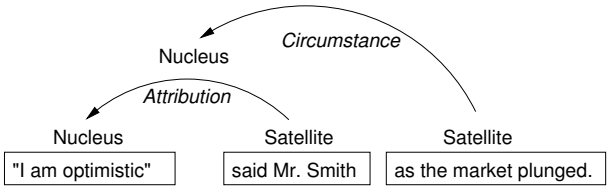

Figure 1: Discourse Tree in RST-DT

3 Discourse Chunking

3.1 Data and Representation

We propose a supervised machine learning approach to discourse chunking. Our data were obtained from the RST-DT (Carlson et al., 2002), which consists of 385 Wall Street Journal articles manually annotated with discourse structures in the framework of Mann and Thompson (1987). An example of an RST-based tree representation is shown in Figure 1; rectangular boxes denote *edus* and arcs indicate which relations (e.g., *Circumstance* or *Attribution*) hold between them. Relations are typically binary with one unit being the nucleus (indicated by arrows in Figure 1) and the other the satellite, but multi-nuclear and non-binary relations are also possible.

We are only interested in the lowest level of the tree, i.e., we aim to identify the *edus* and determine whether they are nuclei or satellites. For example, in the sentence in Figure 1 we want to identify the three *edus "I am optimistic", said Mr. Smith*, and *as the market plunged.* and determine that the first of these functions as a nucleus at the lowest level of the tree whereas the latter two function as satellites. We do not try to determine that the first two *edus* are merged at a higher level and then function as the overall nucleus of the sentence.

The discourse chunking task assumes a non-hierarchical representation. We converted each sentence-level discourse tree into a flat chunk representation by assigning each token (i.e., word or punctuation mark) a tag encoding its nuclearity status at the *edu* level. We adopted the chunk representation proposed by Ramshaw and Marcus (1995) and used four different tags: B-NUC and B-SAT for nucleus and satellite-initial tokens, and I-NUC and I-SAT for non-initial tokens, i.e., tokens inside a nucleus and satellite span. As all tokens belong either

to a nucleus or a satellite span, we do not need a special tag (typically denoted by O in syntactic chunking) to indicate elements outside a chunk. The chunk representation for the sentence in Figure 1 is thus:

"/B-NUC I/I-NUC am/I-NUC optimistic/I-NUC "/I-NUC said/B-SAT Mr./I-SAT Smith/I-SAT as/B-SAT the/I-SAT market/I-SAT plunged/I-SAT ./I-SAT

Discourse and sentence structure do not always correspond, and for 5% of sentences in the RST-DT no discourse tree exists. We excluded these from our data. We also disregarded sentences without internal structure, i.e., those which consist of only one *edu*. The RST-DT is partitioned into a training (342 articles) and test set (43 articles). We preserved this split in all our experiments. 52 articles in the RST-DT are doubly annotated. We used these to compute human agreement on the discourse chunking task (see Section 4.1).

3.2 Modelling

Using a chunk-based representation effectively renders discourse processing a sequence labelling task. Two modelling approaches are possible. The simplest model performs segmentation and labelling simultaneously. In our case this involves training a classifier that labels each token with one of our four tags (i.e., B-NUC, I-NUC, B-SAT, I-SAT). Alternatively, we could treat discourse chunking as two distinct subtasks involving two binary classifiers: a segmenter, which determines the chunk boundaries and assigns each token a chunk-initial (B) or non-chunk-initial tag (I), and a labeller, which classifies each chunk identified by the segmenter as either nucleus (NUC) or satellite (SAT).[1]

The second approach has a number of advantages. First, abstracting away from a token-based representation in the second step makes it easier to model sentence-level distributional properties of nuclei and satellites, e.g., the fact that every sentence has at least one nucleus. This can be achieved by incorporating additional features into the labeller, such as the number of chunks in the sentence or the length of the current chunk. A two-step approach also avoids the creation of illegal chunk sequences, such as "B-SAT I-NUC". However, a potential drawback is that the number of training examples for the labeller is reduced as the instances to be classified are chunks rather than tokens. We explore the performance of the one-step and the two-step methods in Sections 4.2 and 4.3, respectively.

[1] A similar approach has been proposed for syntactic chunking, e.g., Tjong Kim Sang (2000).

A variety of learning schemes can be employed for the discourse chunking task. We have experimented with Boosting (Schapire and Singer, 2000), Conditional Random Fields (Lafferty et al., 2001), and Support Vector Machines (Vapnik, 1998). Discussion of our results focuses exclusively on boosting, since it had a slight advantage over the other methods. Boosting combines many simple, moderately accurate categorisation rules into a single, highly accurate rule. We used BoosTexter's (Schapire and Singer, 2000) implementation, which combines boosting with simple decision rules. The system permits three different types of features: numeric, nominal and "text". Text-valued features can, for example, encode sequences of words or parts of speech. BoosTexter applies *n*-gram models when forming classification hypotheses for text-valued features.

3.3 Features for the Token-Based Models

While we use similar features for all our classifiers, their concrete implementation depends on whether the classifier is token-based (i.e., the one-step model and the segmenter in the two-step method) or span-based (i.e., the labeller in the two-step method). We first describe the features for the former.

Each token is represented as a feature vector encoding information about the token itself and its context. We intentionally limited our features to a basic set representing grammatical, syntactic, and lexical information.

Tokens This feature simply encodes the identity of the current token; we used raw tokens, without lemmatisation or stemming.

Part-of-Speech Tags Tokens were also annotated with parts of speech using a publicly available state-of-the-art tagger (Mikheev, 1997).

Syntactic Chunks Chunk information is a valuable cue for determining discourse segments; it is unlikely that a segment boundary occurs within a syntactic chunk. We applied a chunker (Mikheev, 1997) to our data to discover noun and verb phrase chunks. The chunker assigned one of five labels to each token, encoding the first element of a noun or verb chunk (B-NP and B-VP, respectively), a non-initial element in a chunk (I-NP and I-VP), and an element outside a chunk (O). We used these chunk labels directly as features and also encoded generalisations over chunk and boundary types (i.e., VP vs. NP and B vs. I, respectively).

Clause Information Knowing where clause boundaries lie is important for segmentation, since

discourse segments often correspond to clauses. We used a rule-based algorithm (Leffa, 1998) to identify clauses from the syntactic chunker's output and recorded for every token whether it is clause-initial (S) or not (X).

Discourse Connectives Discourse connectives such as *but* often indicate which rhetorical relation holds between two spans. While we do not aim to infer the relation proper, knowing the type of relation holding between spans often helps in determining whether they should be labelled as nucleus or satellite. For example, *Contrast* relations (e.g., signalled by *but*) hold between two nuclei whereas *Cause* relations (e.g., signalled by *because*) hold between a nucleus and a satellite. Hence, we recorded the presence of discourse connectives in a sentence to capture, albeit in a shallow manner, the interdependency between rhetorical relations and nuclearity.

We used Knott's (1996) inventory of discourse connectives and encoded two types of information for each token: (a) whether the token is a connective (C) or not (X) and (b) the identity of the connective if the token is a connective (zero otherwise).[2]

Token Position For each token we calculated its relative position in the sentence (defined as the token position divided by the number of tokens). This information is useful to capture potential positional differences between nuclei and satellites, i.e., it may be that nuclei are more likely at the beginning of a sentence than at the end.

Context In addition to the nine features above, which encode information about the token itself, we also implemented 16 contextual features to encode information about its neighbouring tokens. Syntactic chunking approaches typically capture contextual information by defining a small window of a few tokens to the left and right of the current token (see Veenstra, 1998). However, we used the whole sentence as context, since BoosTexter is fairly good at determining automatically relevant *n*-grams within a longer string of tokens. We included this contextual information for all nominal features; that is, we encoded not only the string of preceding and following tokens but also the string of preceding and following part-of-speech tags, syntactic chunk labels, clause labels, and connectives. For example, we had three token features, one encoding the current token itself, and two contextual features (one encoding the string of preceding tokens, and one encoding the string of following tokens); similarly we had three part-of-speech features, nine syntactic chunk features three using the complete chunk tags, three using only the chunk type, and three using the boundary type), and so on.

[2] Some words can have syntactic as well as discourse marking functions (e.g., *but* sometimes functions as a synonym for *except* rather than as a *Contrast* marker). We do not disambiguate between these two usages.

3.4 Features for the Span-Based Model

For the labeller we encoded information about spans rather than tokens. This gave rise to six non-contextual, text-valued features: the string of tokens in the current span, their parts of speech, syntactic chunk tags, clause tags, and the presence and identity of connectives. The positional feature was redefined in terms of relative span position, i.e., the position of the current span divided by the number of spans in the sentence. We restricted contextual features to information about immediately preceding and following spans (within a sentence). We did not include information about non-adjacent spans because only a minority of sentences in our data contained more than three spans. Again, we included contextual information for all nominal features. Finally, to capture intra-sentential span-structure, we added the following features:

Span Length Span length was measured in terms of the number of tokens in it and was represented by three features: the length of the current span, and the lengths of its adjacent spans. Span length information captures differences in the average length of nuclei and satellite spans.

Number of Spans We encoded the number of spans in the sentence overall and the number of spans preceding and following the current span.

4 Experiments

In this section we describe the experiments that assess the merits of the discourse chunking framework introduced above. We also give details regarding parameter estimation and training for our models and introduce the baseline and state-of-the-art methods used for comparison with our approach.

4.1 Upper Bound

Before presenting the results of our modelling experiments, it is worth considering how well humans agree on discourse chunk segmentation and labelling in order to establish an upper bound for the task. We measured both unlabelled and labelled agreement on the 52 doubly annotated RST-DT texts. The former measures whether humans agree in placing chunk boundaries, whereas the latter additionally measures

whether humans agree in assigning chunk labels. To facilitate comparison with our models we report inter-annotator agreement in terms of accuracy and F-score.[3] For the unlabelled case we also report *Window Difference* (*WDiff*), a commonly used evaluation measure for segmentation tasks (Pevzner and Hearst, 2002). It returns values between 0 (identical segmentations) and 1 (maximally different segmentations) and differs from accuracy in that predicted boundaries which are only slightly off are penalised less than those which are completely wrong.

Human agreement is relatively high[4] on both segmentation and span labelling (see Table 1), which can be explained by the fact that (i) the RST-DT annotators were given very detailed and precise instructions and (ii) assigning boundaries and labels is an easier task than creating full-scale discourse trees.

4.2 One-Step Chunking

For the one-step chunking method, our training set consists of approximately 130,000 instances (i.e., tokens). We set aside 10% as a development set for optimising BoosTexter's parameters (i.e., the number of training iterations and the maximal length of the *n*-grams considered for text-valued features). We then re-trained BoosTexter with the optimal setting (700 iterations, $n = 2$) and applied it to the test set, which contained around 15,500 instances.

By default, the one-step method treats every token in isolation, i.e., it assigns each token a tag without taking its neighbouring tags into account. This is not an entirely adequate model, since the likelihood of a tag is influenced by its surrounding tags. For example, the probability of a token being tagged as I-NUC should increase if the preceding token was tagged as B-NUC. One way to take information about surrounding tags into account is by stacking classifiers, i.e., adding the output of one classifier to the input of another. Stacking is frequently used in chunking tasks (e.g., Veenstra, 1998). We stack two BoosTexter classifiers, by adding the string of all preceding and following tags (within a given sentence) to each token's feature vector for the second classifier.

It would be possible to generate training material for the second classifier directly from the original training set by using the gold standard output tags in the augmented feature vector. However, we found that this leads BoosTexter to rely too much on these tags, largely ignoring other features. This causes problems when the model is applied to the test set where the class tags are predicted and may contain errors. Hence, we applied the original model (BT-1-Step) to obtain predicted output tags for the training data and then used these, rather than the gold standard tags, to train the second classifier. Similarly, during testing, we first applied BT-1-Step and used its output tags to augmented the feature vectors of the second classifier.

For comparison, we also applied two baseline models to our data. The first (BaseMaj) is obtained by always assigning the tag that is most common in the training data (I-NUC). This strategy makes no attempt at guessing span boundaries. The second (BaseClMaj) indirectly assesses the importance of clause boundary detection. It implements a strategy which assumes that span boundaries always coincide with clause boundaries. To obtain clause boundaries, we used the gold standard annotation of our data in the Penn Treebank. We then labelled all clause-initial tokens as B-NUC and all other tokens as I-NUC. Note, that the use of gold standard clause boundaries makes this a relatively high baseline. We also applied Spade[5], Soricut and Marcu's (2003) sentence-level discourse parser (see Section 2) to our test set. For evaluation purposes, Spade's output was converted to our chunk representation. It is important to note that Spade is a much more sophisticated model than the ones presented in this paper. We therefore do not expect to be able to obtain a better performance. It is nevertheless interesting to see how far one can go with a modest feature space and considerably less structural information.

Table 1 shows the results. A set of diacritics is used to indicate significance (on accuracy) throughout this paper, see Table 2. On the segmentation task (unlabelled) BT-1-Step and its stacked variant significantly outperform the majority baseline (BaseMaj) but are significantly less accurate than BaseClMaj, which uses gold standard clause boundaries. The two BoosTexter models also perform significantly worse than Spade on segmentation. However, the higher WDiff for Spade on the segmentation task suggests that the boundaries predicted by our models contain more "near misses" than those predicted by Spade. When segmentation and span labelling are taken into account (labelled), our one-step models significantly outperform both baselines but are significantly less accurate than Spade. Classifier stack-

[3] For the unlabelled case, we report the F-score on boundaries; for the labelled case, we report the average F-score over all class labels weighted by class frequency in the training set.

[4] Using the Kappa statistic agreement on segmentation is $K = .97$ and on span labelling $K = .81$.

[5] The software is publicly available from `http://www.isi.edu/licensed-sw/spade/`.

Models	unlabelled			labelled	
	Acc %	F-score	WDiff	Acc %	F-score
BaseMaj	88.50	–	.4021	53.87	38.77
BaseClMaj	93.51	70.06	.2008	56.64	43.62
BT-1-Step	90.07$*$†‡\$	64.64	.2148	74.40$*$†‡\$	74.13
BT-1-Step, stacked	91.86$*$ $\not\dagger$‡\$	68.95	.1795	75.55$*$†‡\$	75.37
BT-2-Step	97.37$*$†‡\$	88.28	.0733	78.27$*$† $\not\ddagger$\$	78.38
BT-2-Step, stacked	97.41$*$†‡\$	88.40	.0727	76.31$*$†‡\$	76.34
Spade	93.49$*$ $\not\dagger$\$	87.06	.5071	79.21$*$†\$	80.91
Humans	99.05	97.96	.0012	89.10	89.03

Table 1: Results on discourse segmentation and span labelling

Symbols	Meaning
$*$ $\not*$	(not) sig different from BaseMaj
† $\not\dagger$	(not) sig different from BaseClMaj
‡ $\not\ddagger$	(not) sig different from Spade
\$ $\not\$$	(not) sig different from Humans

Table 2: Meaning of diacritics indicating statistical significance (χ^2 tests, $p < 0.05$)

ing leads to slight improvements over the simple BoosTexter model, but the difference is not statistically significant.

4.3 Two-Step Chunking

In the two-step model, chunking consists of two separate subtasks: segmentation and labelling. To generate training material for the segmenter, we replaced the four chunk labels in the original data set by their corresponding boundary labels (B, I). For the labeller, training instances are spans rather than tokens. We used the gold standard span boundaries to convert the original training set to a span-based representation. This new training set contained around 15,000 instances (compared to 130,000 instances in the token-based set). For both the segmenter and labeller, we set aside 10% of the material as development data to optimise BoosTexter's parameters (900 iterations, $n = 3$ for segmentation, and 600 iterations, $n = 2$ for labelling).

For testing, we first applied the segmenter to obtain discourse chunk boundaries. We then used the predicted boundaries to convert the test data into a span-based representation, which we then used as input for the labeller. For evaluation, the output of the labeller was converted back to a token-based representation. As with one-step chunking, we also implemented a stacked variant, stacking both the segmentation and the labelling models.

It can be seen in Table 1 that the two-step models outperform the one-step models. This difference is significant except for the stacked model on the labelling task (labelled). Both two-step models significantly outperform both baselines on segmentation (unlabelled) and labelling (labelled). They also significantly outperform Spade on the boundary prediction task, which is in itself an important subtask for discourse parsing. The unstacked two-step BoosTexter model performs comparably to Spade with respect to labelled accuracy; the difference between the two models is not statistically significant. Hence, we achieve results similar to Spade but with much simpler and knowledge-leaner features. As with the one-step method, the stacked model performs (insignificantly) better than its unstacked counterpart on the segmentation task. However, on the labelling task, the stacked variant performs significantly worse. We conjecture that the reduced training set size for the labeller causes the stacked model (which is effectively trained twice) to overfit. Expectedly, all models perform significantly worse than humans on both tasks.

To assess whether our discourse chunker could be ported to languages for which discourse treebanks are not yet available, we investigated how much annotated data is required to achieve satisfactory results. Assuming that annotators proceed sentence-by-sentence, we varied the amount of sentences in our training data and determined its effect on the learner's (BT-2-Step) performance. Figure 2 shows that satisfactory labelled and unlabelled performance (86.52% and 74.64% F-score, respectively) can be achieved with approximately half the training data (i.e., around 2,000 sentences). In fact, using the entire data set yields a moderate increase of 1.78% for the unlabelled task and 3.68% for the labelled task. Hence, it seems that our knowledge-lean method is suitable even for relatively small training sets. We next examine whether the two-step chunking model can be usefully employed in a practical application such as sentence compression.

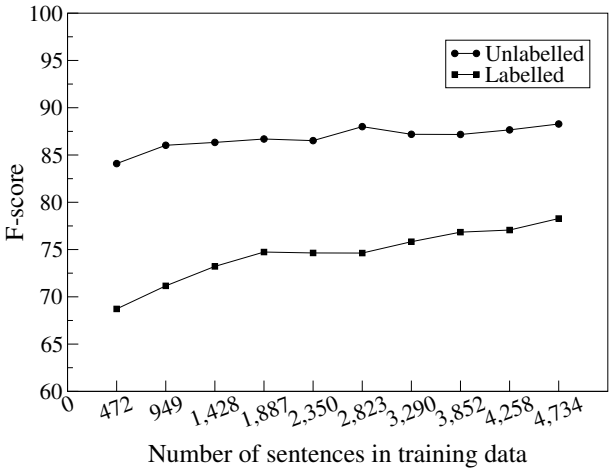

Figure 2: Learning curve for discourse segmentation (unlabelled) and span labelling (labelled)

4.4 Sentence Compression

Sentence compression can be likened to summarisation at the sentence level. The task has an immediate impact on several applications ranging from summarisation to audio scanning devices for the blind and caption generation (see Knight and Marcu, 2002 and the references therein). Previous data-driven approaches (Knight and Marcu, 2003; Riezler et al., 2003) relied on parallel corpora to determine what is important in a sentence. The models learned correspondences between long sentences and their shorter counterparts, typically employing a rich feature space induced from parse trees. The task is challenging since the compressed sentences should retain essential information and convey it grammatically.

Here, we propose a complementary approach which utilises discourse chunking. A compressed sentence can be obtained from the output of the chunker simply by removing satellites. We thus capitalise on RST's (Mann and Thompson, 1987) notion of nuclearity and the widely held assumption that spans functioning as satellites can often be deleted without disrupting coherence. To evaluate the compressions produced by our chunking model, we elicited judgements from human subjects. We describe our elicitation study and results as follows.

Data We randomly selected 40 sentences from the test portion of the RST-DT. Average sentence length was 38.75. The sentences were compressed by chunking them with our (unstacked) two-step model (BT-2-Step) and then dropping satellites. We applied the same strategy to derive compressed sentences from the output of Spade (Soricut and Marcu, 2003), and also produced human compressions. Finally, we added a simple baseline compression algorithm proposed by Jing and McKeown (2000) which removed all prepositional phrases, clauses, to-infinitives, and gerunds. Both the baseline and Spade operate on parse trees which were obtained from Charniak's (2000) parser. Our set of experimental materials contained $4 \times 40 = 160$ compressions.

Original
Administration officials traveling with President Bush in Costa Rica interpreted Mr. Ortega's wavering as a sign that he isn't responding to the military attacks so much as he is searching for ways to strengthen his hand prior to the elections.
Baseline
Administration officials interpreted Mr. Ortega's wavering.
BT-2-Step
Administration officials interpreted Mr. Ortega's wavering as a sign that he isn't responding to the military attacks so much as he is searching for ways.
Spade
Administration officials traveling with President Bush in Costa Rica interpreted Mr. Ortega's wavering as a sign.
Human
Administration officials interpreted Mr. Ortega's wavering as a sign that he is searching for ways to strengthen his hand prior to the elections.

Table 3: Example compressions

Compression	AvgLen	Rating
Baseline	9.70	1.93
BT-2-Step	22.06	3.21
Spade	19.09	3.10
Humans	20.07	3.83

Table 4: Mean ratings for automatic compressions

Procedure and Subjects We obtained compression ratings during an elicitation study completed by 45 unpaid volunteers, all native speaker of English. The study was conducted remotely over the Internet. Participants first saw a set of instructions that explained the task, and defined sentence compression using multiple examples. The materials consisted of the original sentences together with their compressed versions. They were randomised in lists following a Latin square design ensuring that no two compressions in a list were generated from the same sentence. As in Knight and Marcu's (2003) study, participants were asked to use a five point scale to rate the systems' compressions (taking into account the felicity of the compression as well as its grammaticality); they were told that all outputs were generated automatically. Examples of the compressions our participants saw are given in Table 3.

Results We carried out an Analysis of Variance (ANOVA) to examine the effect of different types of compressions (Baseline, BT-2-Step, Spade, and Human). Statistical tests were done using the mean

of the ratings shown in Table 4. The ANOVA revealed a reliable effect of compression type by subjects ($F_1(3,90) = 149.50$, $p < 0.001$) and by items ($F_2(3;117) = 40.23$, $p < 0.001$). Post-hoc Tukey tests indicated that human compressions are perceived as significantly better than the compressions produced by the baseline, BT-2-Step, and Spade ($\alpha = 0.01$). The discourse chunker and Spade are significantly better than the baseline ($\alpha = 0.01$). The Tukey test revealed no statistically significant difference between these two algorithms ($\alpha = 0.01$). To summarise, both BoosTexter and Spade perform closer to human performance than the baseline; yet, humans perform significantly better than our compression algorithms.

5 Conclusions

In this paper we proposed discourse chunking as an alternative to full-scale parsing. Central in our approach is the use of low-level syntactic and grammatical information which we argue holds promise for the development of discourse processing models across languages and domains. We showed that a knowledge-lean feature space achieves good performance both on segmentation and span labelling. Furthermore, we assessed the application potential of our chunker and showed that it can be successfully employed to generate sentence compressions, thus confirming one of RST's main claims regarding the nuclearity of discourse spans (at least on the sentence-level).

An important future direction lies in extending our model to the document-level and the assignment of rhetorical relations, thus going beyond the basic nucleus-satellite distinction. Our results indicate that a modular approach to discourse processing (i.e., treating segmentation as separate from labelling) could increase performance. In the future, we plan to investigate how to combine our chunker with models like Spade for improved prediction on both local and global levels.

Acknowledgments

The authors acknowledge the support of EPSRC (Sporleder, grant GR/R40036/01; Lapata, grant GR/T04540/01). Thanks to Amit Dubey, Ben Hutchinson, Alex Lascarides, Simone Teufel, and three anonymous reviewers for helpful comments and suggestions.

References

L. Carlson, D. Marcu, M. E. Okurowski. 2002. RST Discourse Treebank. Linguistic Data Consortium, 2002.

J. Chai, R. Jing. 2004. Discourse structure for context question answering. In *Proceedings of the Workshop on Pragmatics of Question Answering at HLT-NAACL 2004*, 23–30.

E. Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of the 1st NAACL*, 132–139.

B. Endres-Niggemeyer. 1998. *Summarising Information*. Springer, Berlin.

C. Fellbaum, ed. 1998. *WordNet: An Electronic Database*. MIT Press, Cambridge, MA.

J. R. Hobbs, M. Stickel, D. Appelt, P. Martin. 1993. Interpretation as abduction. *Journal of Artificial Intelligence*, 63(1–2):69–142.

K. Humphreys, R. Gaizauskas, S. Azzam. 1997. Event coreference for information extraction. In *Proceedings of the ACL Workshop on Operational Factors in Practical Robust Anaphora Resolution for Unrestricted Texts*, 75–81.

H. Jing, K. McKeown. 2000. Cut and paste summarization. In *Proceedings of the 1st NAACL*, 178–185.

K. Knight, D. Marcu. 2003. Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artificial Intelligence*, 139(1):91–107.

A. Knott. 1996. *A Data-Driven Methodology for Motivating a Set of Coherence Relations*. Ph.D. thesis, Department of Artificial Intelligence, University of Edinburgh.

J. Lafferty, A. McCallum, F. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th ICML*, 282–289.

V. J. Leffa. 1998. Clause processing in complex sentences. In *Proceedings of the 1st LREC*, 937–943.

W. C. Mann, S. A. Thompson. 1987. Rhetorical structure theory: A theory of text organization. Technical Report ISI/RS-87-190, ISI, Los Angeles, CA, 1987.

D. Marcu. 1998. To build text summaries of high quality, nuclearity is not sufficient. In *Working Notes of the AAAI-98 Spring Symposium on Intelligent Text Summarization*, 1–8.

D. Marcu. 2000. *The Theory and Practice of Discourse Parsing and Summarization*. The MIT Press, Cambridge, MA.

A. Mikheev. 1997. The LTG part of speech tagger. Technical report, University of Edinburgh, 1997.

L. Pevzner, M. Hearst. 2002. A critique and improvement of an evaluation metric for text segmentation. *Computational Linguistics*, 28(1):19–36.

L. A. Ramshaw, M. P. Marcus. 1995. Text chunking using transformation-based learning. In *Proceedings of the 3rd ACL Workshop on Very Large Corpora*, 82–94.

S. Riezler, T. H. King, R. Crouch, A. Zaenen. 2003. Statistical sentence condensation using ambiguity packing and stochastic disambiguation methods for lexical-functional grammar. In *Proceedings of HLT/NAACL 2003*, 118–125.

R. E. Schapire, Y. Singer. 2000. BoosTexter: A boosting-based system for text categorization. *Machine Learning*, 39(2/3):135–168.

R. Soricut, D. Marcu. 2003. Sentence level discourse parsing using syntactic and lexical information. In *Proceedings of HLT/NAACL 2003*.

S. Teufel, M. Moens. 2002. Summarizing scientific articles – experiments with relevance and rhetorical status. *Computational Linguistics*, 28(4):409–446.

E. F. Tjong Kim Sang. 2000. Text chunking by system combination. In *Proceedings of CoNLL-00*, 151–153.

V. Vapnik. 1998. *Statistical Learning Theory*. Wiley-Interscience, New York.

J. Veenstra. 1998. Fast NP chunking using memory-based learning techniques. In *Proceedings of BENELEARN*, 71–79.