Broad Coverage Paragraph Segmentation across Languages and Domains

CAROLINE SPORLEDER
Tilburg University
and
MIRELLA LAPATA
University of Edinburgh

This paper considers the problem of automatic paragraph segmentation. The task is relevant for speech-to-text applications whose output transcipts do not usually contain punctuation or paragraph indentation and are naturally difficult to read and process. Text-to-text generation applications (e.g., summarisation) could also benefit from an automatic paragaraph segementation mechanism which indicates topic shifts and provides visual targets to the reader. We present a paragraph segmentation model which exploits a variety of knowledge sources (including textual cues, syntactic and discourse related information) and evaluate its performance in different languages and domains. Our experiments demonstrate that the proposed approach significantly outperforms our baselines and in many cases comes to within a few percent of human performance. Finally, we integrate our method with a single document summariser and show that it is useful for structuring the output of automatically generated text.

Categories and Subject Descriptors: I.2.6 [**Artificial Intelligence**]: Learning—*Knowledge Acquisition*; I.2.7 [**Artificial Intelligence**]: Natural Language Processing—*Text analysis*; I.2.7 [**Artificial Intelligence**]: Natural Language Processing—*Discourse*

General Terms: Algorithms; Experimentation; Languages

Additional Key Words and Phrases: machine learning, paragraph breaks, segmentation, summarisation

1. INTRODUCTION

Written texts are usually broken up into sentences, paragraphs, headings and subheadings. Sentence splitting is a necessary pre-processing step for a number of Natural Language Processing (NLP) tasks, including part-of-speech tagging and parsing but also for applications such as text simplification and summarisation. Although humans perform sentence segmentation effortlessly while reading, automatic approaches are faced with the problem of inferring whether punctuation is

Authors' Addresses: Caroline Sporleder, ILK/Language and Information Science, Tilburg University, P.O. Box 90153, 5000 LE Tilburg, The Netherlands, csporled@uvt.nl
Mirella Lapata, School of Informatics, 2 Buccleuch Pl., Edinburgh EH8 9LW, UK, mlap@inf.ed.ac.uk

sentence-final or not (e.g., a period can be used in an abbreviation or a decimal point as well as to mark the end of a sentence). Given the difficulty of identifying sentence boundaries automatically and its importance for NLP, it is not surprising that the task has attracted a lot of attention [Reynar and Ratnaparkhi 1997; Palmer and Hearst 1997; Shriberg et al. 2000].

Beyond sentence segmentation, much research has been devoted to identifying topically coherent blocks of text that span across multiple paragraphs. The automatic segmentation of texts into subtopics has important uses for text understanding [Morris and Hirst 1991], summarisation [Barzilay and Elhadad 1997], hypertext navigation [Hearst 1997; Choi 2000], and information retrieval [Hearst 1997; Yaari 1997]. In some types of texts, such as scientific papers or generally technical documents, headings and subheadings can signal subtopic structure; however, in most cases, texts are not visibly marked with subtopic structure and it is precisely for these texts that subtopical segmentation can be useful. Topic segmentation identifies subtopic boundaries in a linear fashion and makes use primarily of lexical distribution information. It is thus different from discourse segmentation which is often finer-grained and focuses on identifying hierarchical relations across utterances (e.g., Marcu [2000]).

In contrast to sentence and subtopic segmentation, there has been virtually no previous research on inferring paragraph boundaries automatically. One reason for this is that paragraph boundaries are usually marked unambiguously in text by a new line and extra white space. However, a number of applications could benefit from a paragraph segmentation mechanism. Text-to-text generation applications such as single- and multidocument summarisation as well as text simplification usually take naturally occurring texts as input and transform them into new texts satisfying specific constraints (e.g., length, style, language). The output texts do not always preserve the structure and editing conventions of the original text. In summarisation, for example, sentences are typically extracted verbatim and concatenated to form a summary. Insertion of paragraph breaks could improve the readability of the summaries by indicating topic shifts and providing visual targets to the reader [Stark 1988].

Machine translation is another application for which automatic paragraph detection is relevant. Current systems deal with paragraph boundary insertion in the target language simply by preserving the boundaries from the source language. However, there is evidence for cross-linguistic variation in paragraph formation and placement, particularly for language pairs that are not closely related such as English and Chinese [Hinds 1979; Zhu 1999]. So, a paragraph insertion mechanism that is specific to the target language, instead of one that relies solely on the source language, may yield more readable texts.

Paragraph boundary detection is also relevant for speech-to-text applications. The output of automatic speech recognition systems is usually raw text without any punctuation or paragraph breaks. This naturally makes the text very hard to read and may cause difficulties in situations where humans have to process the output text with ease. This is precisely what happens when speech recognition is used to provide deaf students with real-time transcripts of lectures. In this case an automatic paragraph insertion mechanism would improve the transcripts' readability.

Furthermore, sometimes the output of a speech recogniser needs to be processed automatically by applications such as information extraction or summarisation. Most of these applications port techniques developed for written texts to spoken texts (e.g., Christensen et al., [2004]) and therefore require input that is punctuated and broken into paragraphs. While there has been some research on finding sentence boundaries in spoken text [Stevenson and Gaizauskas 2000], relatively little research has examined the automatic insertion of paragraph boundaries. A notable exception are Hauptmann and Smith [1995] who segment spoken texts into "acoustic paragraphs" on the basis of pauses in the flow of speech. However, acoustic paragraphs may not necessarily correspond to paragraphs in written text.

It can be argued that paragraphs are mainly an aesthetic device for visually breaking up long texts into smaller chunks [Longacre 1979], and therefore paragraph boundaries could be easily inserted by splitting a text into several equal-size segments. Psycho-linguistic research, however, indicates that paragraphs are not purely aesthetic. For example, Stark [1988] asked subjects to reinstate paragraph boundaries into fiction texts from which all paragraph breaks had been removed and found that humans are able to do so with an accuracy that is higher than would be expected by chance. Crucially, she also found that (a) individual subjects did not make all their paragraphs the same length and (b) paragraphs in the original text whose length deviated significantly from the average paragraph length were still identified correctly by a large proportion of subjects. These results show that people are often able to identify paragraphs correctly, even if they are exceptionally short or long, without defaulting to a simple template of average paragraph length.

Human agreement on the task suggests that the text itself provides cues for paragraph insertion, even though there is some disagreement over which specific cues are used by humans. Bond and Hayes [1984] found that mainly three devices are used by readers to identify a paragraph: (a) the repetition of content words (nouns, adjectives, verbs, adverbs), (b) pronoun co-reference, and (c) paragraph length as determined by sentence count information. Stark [1988], on the other hand, argued that paragraph length on its own may not be a very useful cue and that other factors, such as theme marking, the presence and absence of coordination and local semantic connectedness may be more predictive. Textual cues for paragraph structure may also vary across languages and text genres. For example, Longacre [1979] presents evidence that some languages have special particles to mark the start of a new paragraph, while other languages indicate paragraph boundaries by increased or decreased use of back-references. Brown and Yule [1983] show that temporal adverbials often indicate the start of a new paragraph in narrative texts, whereas paragraph boundaries in philosophical texts are more often indicated by clauses that signify a change in the direction of the argument (see Hinds [1979] for a detailed discussion on the relation between paragraph structure and genre).

In this paper, we investigate whether it is possible to exploit textual cues together with syntactic and discourse related information to determine paragraph boundaries automatically. We view paragraph segmentation as a classification task (see Section 3). Given a set of sentences making up a document, we label each sentence as either paragraph initial or non-paragraph initial. We perform this labelling using boosting [Shapire and Singer 2000], a machine learning technique that

combines many simple and moderately accurate categorisation rules into a single, highly accurate categorisation rule. We exploit a variety of knowledge sources and evaluate their contribution to the paragraph identification task. We also examine the cross-language and cross-genre portability of our methods, thus empirically assessing whether languages and genres differ in marking paragraph structure. Our machine learning experiments are complemented by a study in which we investigate human performance on the same task and whether it differs across domains and languages (see Section 4). Finally, we integrate our paragraph segmentation method with a single document summariser and evaluate its relevance for this particular text-to-text generation application (see Section 5). We start by giving an overview of related work.

2. RELATED WORK

To our knowledge there has been no previous attempt to automatically infer paragraph boundaries in written text. As mentioned in Section 1 much previous work has focused on topic segmentation and many algorithms have been proposed in the literature for performing this task. These algorithms are in their majority unsupervised and rely on the distribution of words in the text to provide cues for topic segmentation, under the assumption that different subtopics are signalled by different sets of lexical items and therefore when a subtopic changes, a significant proportion of the vocabulary changes as well.

Hearst's [1997] TextTiling algorithm, for example, determines subtopic boundaries on the basis of term overlap in adjacent text blocks, where the term overlap across a boundary is expected to be smaller than the term overlap between blocks which belong to the same subtopic. Hearst's algorithm uses a sliding window to compute block similarity at regular intervals in the text and then determines subtopic boundaries by looking for significant dips in the resulting similarity plot. Variations of this method have been employed by Richmond et al. [1997], Salton et al. [1996] and Boguraev and Neff [2000].

Reynar [1998] extends Hearst's [1997] ideas by representing term overlap in a matrix and then applying an optimisation algorithm to determine boundaries. A similar approach is taken by Choi [2000] who uses a similarity rank matrix in combination with clustering. Brants et al. [2002] provide a different extension and integrate term overlap with Probabilistic Latent Semantic Analysis [Hofmann 2001].

Another strand of work utilises lexical chains (see Morris and Hirst [1991]) rather than term overlap. A lexical chain is a sequence of semantically related words, where semantic relatedness is determined on the basis of a thesaurus or a similar resource, such as WordNet [Miller et al. 1990]. Because a chain corresponds to a theme or topic in the text, subtopic boundaries can often be determined by looking for places where many chains end and many new chains begin. The usefulness of lexical chains to detect subtopic boundaries has been investigated by Hearst [1994].

A third line of research was suggested by Utiyama and Isahara [2001]. They model text segmentation probabilistically and use a graph search algorithm to find the segmentation with the maximum probability (see Section 4.5 for more details).

Finally, Genzel and Charniak [2003] investigated how word entropy rate and syntactic complexity change within texts and found that paragraph initial sentences

are usually syntactically less complex and have a lower word entropy rate than sentences within a paragraph. They comment that these properties could be exploited to determine paragraph boundaries automatically but do not pursue this idea further.

Supervised approaches to text segmentation operate on texts that are explicitly marked with subtopic boundaries. Given the paucity of such data, few researchers have attempted the segmentation task in a supervised manner. Most approaches combine term co-occurrence with other cues, such as expressions which typically indicate topic shifts. As one might expect, these cues vary across domains. For example, in the spoken news domain expressions such as *welcome back*, *good evening*, and *joining us* often indicate a new topic [Reynar 1998]. Beeferman et al. [1999] employ language models to detect topic shifts and combine them with cue word features in a maximum entropy model. Litman and Passonneau [1995] use a decision tree learner for the segmentation task; they employ three sets of cues: prosodic cues, cue phrases and noun phrases (e.g., the presence or absence of anaphora). Kan et al. [1998] rely nearly exclusively on lexical chains but combine these with a weighting scheme, where the weights are set in a supervised training step.

While most approaches produce a linear, non-hierarchical segmentation, there are also methods delivering a hierarchical segmentation. Yaari [1997] discusses an unsupervised method based on term overlap and clustering while Kan [2001] uses supervised decision rule learning to infer a hierarchical segmentation.

Although related to text segmentation, our work differs from these previous approaches in that paragraphs do not always correspond to subtopics. While topic shifts often correspond to paragraph breaks, not all paragraph breaks indicate a topic change. Breaks between paragraphs are often inserted for other (not very well understood) reasons (see e.g., Longacre [1979], Brown and Yule [1983] and Stark [1988]).[1] Therefore, the segment granularity is more fine-grained for paragraphs than for topics. An important advantage for methods developed for paragraph detection (as opposed to those developed for text segmentation) is that training data is readily available, since paragraph boundaries are usually unambiguously marked in texts. Hence, supervised methods are "cheap" for this task.

Work on text segmentation has nevertheless inspired our choice of features for the paragraph identification task. We will make use of cue phrases, language models and term overlap as indicators of topic shifts; but we will also investigate the contribution of relatively knowledge-intensive features that are based on syntactic structure. Our contributions are three-fold: an automatic method for paragraph segmentation which we show can be easily ported across languages and text genres; an empirical validation of claims regarding paragraph placement in the literature; and an application of our paragraph segmenter to single-document summarisation.

[1] In Section 4.5 we investigate empirically whether existing subtopic segmentation models can be used to infer paragraph breaks. Our results confirm that paragraph structure is indeed different from subtopic structure and existing topic segmentation algorithms do not lead to very good results on the paragraph segmentation task.

3. MODELING PARAGRAPH SEGMENTATION

In this paper we adopt a supervised approach to paragraph segmentation. The availability of text marked with paragraph breaks makes such an approach feasible and enables us to empirically investigate which cues are important for the paragraph insertion task and whether they generalise across languages and text genres.

We concentrated on three languages: English, German, and Greek. These languages vary in morphological and syntactic complexity. English has relatively impoverished morphology whereas German and Greek are highly inflected languages. More importantly the three languages differ in terms of word order which is relatively fixed for English, semi-free for German and fairly flexible for Greek. Additionally, Greek has a non-Latin writing system. The focus on different languages allowed us to study whether languages differ in the linguistic devices they employ in indicating the start of paragraphs [Longacre 1979].

Furthermore, we wanted to know whether paragraph conventions vary across domains as has been previously noted in the literature [Brown and Yule 1983]. If this is indeed the case, then one may have to re-train a paragraph segmenter for each new domain. We concentrated on texts representative of three domains: fiction, news, and parliamentary proceedings. Previous experimental work on the role of paragraph markings [Stark 1988] has focused exclusively on fiction texts and has shown that humans can identify paragraph boundaries in this domain reliably. It therefore seemed natural to test our automatic method on a domain for which the task has been shown to be feasible. We selected news texts since most summarisation methods today focus on this domain and we can therefore explore the relevance of our approach for this application. Finally, parliamentary proceedings are transcripts of speech, and we can examine whether a method that relies solely on textual cues is also useful for transcribed spoken texts.

Prior work on topic segmentation has exploited several different cues about where topic boundaries lie. Our machine learning experiments use many cues from the topic segmentation literature as well as novel ones that are particularly tailored to the paragraph detection task. Our features fall broadly into three different areas: surface features, language modelling features and syntactic features (see the following sections for a detailed discussion). The latter were only applied to English and German as we did not have access to a suitable parser for Greek. The surface features take into account the distribution of paragraph breaks, lexical information (e.g., paragraph starting words, word overlap), as well as punctuation. The syntactic features are read off from parse trees and are used to record primarily syntactic complexity. We deliberately did not include anaphora-based features. While anaphors can help determine paragraph boundaries (paragraph initial sentences tend to contain few or no anaphors), anaphora structure is dependent on paragraph structure rather than the other way round. Hence, in applications which manipulate texts and thereby potentially "mess up" the anaphora structure (e.g., multi-document summarisation), anaphors are not a reliable cue for paragraph identification.[2]

[2]This is also true for some of the other features we use (e.g., sentence length) but not quite to the same extent.

In the following sections we give a brief overview of the machine learner we used for our experiments (Section 3.1) and describe in more detail our features and the motivation behind their selection (Section 3.2 to Section 3.4). Section 3.6 presents our evaluation measures.

3.1 BoosTexter

We used BoosTexter [Shapire and Singer 2000] as our machine learning system. BoosTexter is a member of a family of boosting algorithms described in detail in Schapire and Singer [1999; 2000]; for completeness, we give a brief description in this section.

The main idea behind boosting is to find a highly accurate classification rule by combining many *weak hypotheses*, each of which may be only moderately accurate. Boosting presupposes access to a *weak* or *base learner* for computing the weak hypotheses. Let $\mathcal{X}$ denote the set of instances and let $\mathcal{Y}$ be a finite set of labels. Then $S = \langle (x_1, Y_1), \ldots, (x_m, Y_m) \rangle$ is a sequence of training examples where each pair (x_i, Y_i) consists of an instance $x_i \in \mathcal{X}$ and a set of labels $Y_i \subseteq \mathcal{Y}$. A weak learner h is a triple $(p, \vec{\alpha}, \vec{\beta})$, which tests a predicate p of the input x and assigns a weight $\alpha_i (i = 1, \ldots, n)$ for each member y of $\mathcal{Y}$ if p is true in x and assigns a weight $(\vec{\beta_i})$ otherwise.

From the pool of weak learners $H = \{h\}$, a combined weak learner is selected iteratively. At each iteration t, a weak learner h_t is selected that minimises a prediction error loss function on the training corpus. The output of the weak learner is a hypothesis $h : \mathcal{X} \times \mathcal{Y} \rightarrow R$. The sign of $h_t(x, y)$ (i.e., -1 or $+1$) is interpreted as a prediction and its magnitude $|h_t(x, y)|$ as a measure of confidence in the prediction [Shapire and Singer 2000]. At each iteration the weight α_t is updated for each example-label pair. For instance, the weight is increased for example-label pairs which are misclassified by h_t. The iterative algorithm stops after a prespecified number of iterations or when the accuracy in the test set remains stable.

Boosting is a general purpose method and can be combined with any classifier. So far it has been used with decision trees [Drucker and Cortes 1996], neural nets [Drucker et al. 1992], and decision stumps [Shapire and Singer 2000]. The latter are BoosTexter's default classifier and have been used for all our experiments. More specifically, the weak hypotheses have the same basic form as a one-level decision tree. The test at the root of this tree can be a check for the presence or absence of a word or a sequence of words in a given sentence; a check for the value of a particular attribute (discrete attributes); or a check for the attribute value being above or below some threshold (continuous attributes).

For all domains and languages our training examples were sentences.[3] Class labels encoded for each sentence whether it was starting a paragraph or not. The values of our features are numeric, boolean or "text". Text-valued features can, for example, encode the words or part-of-speech tags of a sentence. BoosTexter applies n-gram models when forming classification hypotheses for features with "text" values (i.e., it tries to detect n-grams in the text which are particularly good predictors for a class label).

[3]Though some of our features did refer back to the previous sentence.

3.2 Surface Features

Our surface features can be easily estimated from the raw text without recourse to elaborate semantic or syntactic knowledge. They are applicable across languages provided that word and sentence boundaries are marked or can be identified using automatic means.

Distance (D_s, D_w). Distance features capture how paragraphs are distributed in a given text. More specifically, we encode the distance of the current sentence from its previous paragraph break. We measure distance in terms of the number of intervening sentences (D_s) as well as in terms of the number of intervening words (D_w). These features should work well if paragraph breaks are driven purely by aesthetics.[4] One would only need to know how the paragraphs are distributed in a given domain or language in order to reinstate the paragraph breaks for unseen data.

Sentence Length ($Length$). This feature encodes the number of words in the current sentence. Average sentence length is known to vary with relative text position [Genzel and Charniak 2003; Keller 2004] and it is possible that it also varies with paragraph position. For example, one could hypothesise that paragraph initial sentences are often relatively short.

Relative Position (Pos). Previous work on text segmentation uses the relative position of a sentence in a text as an indicator of text layout [Kan 2001]. We additionally hypothesise that paragraph length may vary with text position. For example, it is possible that paragraphs at the beginning and end of a text are shorter than paragraphs in the middle and hence a paragraph break is more likely at the two former text positions. We calculate the relative position of a sentence in the text by dividing the current sentence number by the number of sentences in the text.

Quotes ($Quote_p$, $Quote_c$, $Quote_i$). We explicitly encode the presence of quotation marks, in an attempt to represent, albeit in a shallow manner, the presence or absence of direct speech. More specifically, we test for pairs of quotation marks in the previous ($Quote_p$) and current ($Quote_c$) sentence. The third feature ($Quote_i$) tests whether one of previous sentences contained opening quotation marks which sofar have not been closed. This is taken as evidence that the current sentence continues a stretch of direct speech. Direct speech should make a good cue for paragraph boundaries in some domains (e.g., fiction) because speaker turns are often signalled by a paragraph break. Furthermore, sentences that continue direct speech are relatively unlikely to be paragraph initial.

Final Punctuation ($FinPun$). This feature keeps track of the final punctuation mark of the previous sentence. Some punctuation marks may provide hints as to whether a break should be introduced [Kan 2001]. For example, a question and answer pair is unlikely to be interrupted by a paragraph break, unless both are direct speech and there was a speaker turn. Consequently, in domains which

[4]One could also use the history of class labels assigned to previous sentences as a feature (as in part-of-speech tagging); however, we leave this to future research.

do not contain much direct speech (e.g., news), the fact that the previous sentence ended in a question mark should decrease the probability that the current sentence is paragraph initial. Other punctuation marks may be similarly useful cues.

Words (W_1, W_2, W_3, W_{all}). Many topic segmentation algorithms rely on the frequency of individual words [Hearst 1997; Reynar 1998; Beeferman et al. 1999] or multi-word phrases [Reynar 1998] to indicate topic change. Although paragraph breaks are not necessarily correlated with topic and consequently vocabulary change, there are certain words that may occur frequently at the start of a paragraph (e.g., *Yes*, *Oh*, *If*). This is in line with the idea that languages employ specific phrases (e.g., temporal adverbials) to mark paragraph boundaries [Longacre 1979; Brown and Yule 1983]. Furthermore, by taking the words of a sentence into account, we implicitly capture the presence or absence of cue words. The use of cue words has been widespread in discourse segmentation [Litman and Passonneau 1995] as well as topic segmentation [Kan 2001; Beeferman et al. 1999]. W_{all} takes the complete sentence as its value; W_1 encodes the first word in a sentence, W_2 the first two words, and W_3 the first three words.

Word Overlap (W_{over}). We use word overlap as a superficial way of capturing theme changes in the text. In particular, we hypothesise that paragraph starting sentences are more likely to exhibit relatively low word overlap with their preceding sentences. For each adjacent pair of sentences X and Y, we use the Dice coefficient to measure word overlap. This is defined as follows:

$$W_{over}(X,Y) = \frac{2|X \cap Y|}{|X| + |Y|} \tag{1}$$

where $|X \cap Y|$ is the number of words that occur in both sentences and $|X|$ ($|Y|$) is the number of words in sentence X (Y). Notice that our use of word overlap is relatively localised; we do not keep track of how individual words distribute within an entire document as is customary in topic segmentation (e.g., Hearst [1997]).

3.3 Language Modelling Features

Our motivation for including language modelling features stems from Genzel and Charniak's [2003] work where they show that the word entropy rate is lower for paragraph initial sentences than for non-paragraph initial ones. We therefore decided to examine whether per-word entropy is a useful feature for the paragraph prediction task. Following Genzel and Charniak [2003] we computed per-word entropy as follows:

$$\hat{H}(X) = -\frac{1}{|X|} \sum_{x_i \in X} \log P(x_i | x_{i-(n-1)} \dots x_{i-1}) \tag{2}$$

Here, $\hat{H}(X)$ is the estimate of the per-word entropy of sentence X, consisting of words x_i, and n is the size of the n-gram. In addition to per-word entropy, we also estimated the probability of a sentence according to a language model of size n:

$$P_{LM}(x_1 \dots x_n) = \prod_{i=1}^{n} P(x_i | x_{i-1}) \tag{3}$$

Using the training set for each language and domain, we created language models with the CMU language modelling toolkit [Clarkson and Rosenfeld 1997]. We experimented with language models of variable length (i.e., 1–5); the models were smoothed using Witten-Bell discounting. We additionally experimented with character level n-gram models. Such models are defined over a relatively small vocabulary and can be easily constructed for any language without pre-processing. Character level n-gram models have been applied to the problem of authorship attribution and obtained state-of-the art results [Peng et al. 2003]. If some characters are more often attested in paragraph starting sentences (e.g., "A" or "T"), then we expect these sentences to have a higher probability compared to non-paragraph starting ones. Again, we used the CMU toolkit for building the character level n-gram models (see (3)). We experimented with models whose length varied from 2 to 8 and estimated the probability assigned to a sentence according to the character level model (P_{CM}).

3.4 Syntactic Features

For the English and German data we also used several features encoding syntactic complexity. Genzel and Charniak [2003] show that the syntactic complexity of sentences varies with their position in a paragraph. Their findings suggest that paragraph starting sentences are less complex than non-paragraph starting ones. Hence, measures of syntactic complexity may be good indicators of paragraph boundaries. To estimate complexity, we parsed the English texts with Charniak's [2000] parser. The latter was trained and tested on the Penn Treebank with a lexicalised Markov grammar parsing model that achieved an average precision/recall of 89.5% on sentences of length < 100. German texts were parsed using Dubey's [2004] parser, a bottom-up CYK parser that employs an unlexicalised Markov grammar model. The German parser was trained and tested on the Negra corpus,[5] achieving an average precision/recall of 79%. The following complexity features were extracted from English and German parse trees.

Parsed. A few sentences (typically less than 1%) in each data set could not be parsed. Whether a sentence can be parsed or not is probably correlated with its syntactic complexity. Hence, we introduced a boolean feature to record this information.

Number of phrases (num_s, num_{vp}, num_{np}, num_{pp}). One way to measure syntactic complexity is by recording the number of S, VP, NP, and PP constituents in the parse tree.

Signature ($Sign$, $Sign_p$). The sequence of part-of-speech tags in a sentence can also be viewed as a way of encoding syntactic complexity. $Sign$ only encodes the sequence of word tags, while $Sign_p$ also includes punctuation tags.

Children of Top-Level Nodes ($Childr_{s1}$, $Childr_s$). These features encode the top-level complexity of a parse tree: $Childr_{s1}$ takes as its value the sequence of syntactic labels of the children of the S1-node (i.e., the root of the parse tree), while $Childr_s$ encodes the syntactic labels of the children of the highest S-node(s). For

[5]See `http://www.coli.uni-sb.de/sfb378/negra-corpus/`.

example, $Childr_{s1}$ may encode that the sentence consists of one clause and $Childr_s$ may encode that this clause consists of an NP, a VP, and a PP.

Branching Factor ($Branch_s$, $Branch_{vp}$, $Branch_{np}$, $Branch_{pp}$). These features measure the complexity of S, VP, NP, and PP constituents in the sentence by recording the average number of their children (cf. Genzel and Charniak [2003]).

Tree Depth. We define tree depth as the average length of a path (from root to leaf node) in the parse tree.

Discourse Cues (Cue_s, Cue_m, Cue_e). Our word-based features simply record information about words present in a given sentence while being agnostic of their particular function. Here, we focus explicitly on discourse cues in order to assess their individual contribution. Cue_s, Cue_m, and Cue_e are boolean features and encode whether there are any cues (such as *because*) at the start, in the middle and at the end of the sentence, respectively. We define "start" as the first word, "end" as the last one, and everything else as "middle". We keep track of all cue word occurrences, without attempting to distinguish between their syntactic and discourse usages. For English, there are extensive lists of discourse cues (we employed Knott's [1996]), but such lists are not widely available for German and Greek. Hence, we only used this feature on the English data.

3.5 Data

Since our approach is supervised, we require training examples (i.e., sentences) whose class labels indicate whether they are paragraph starting or not. As mentioned previously, we concentrated on three languages (English, German, Greek) and three domains (fiction, news, parliamentary proceedings). We therefore compiled corpora representative of these domains and languages for training and testing purposes.

For English, we used the whole Hansard section of the BNC as our corpus of parliamentary proceedings. We then created a fiction corpus of similar size by randomly selecting prose files from the fiction part of the BNC. A news corpus was created by randomly selecting files from the Penn Treebank.

For German, we used the prose part of Project Gutenberg's e-book collection[6] as our fiction corpus. The news corpus was created from the complete Frankfurter Rundschau part of the ECI corpus[7] and the corpus of parliamentary proceedings was obtained by randomly selecting a subset of the German section from the Europarl corpus [Koehn 2002].

For Greek, a fiction corpus was compiled from the ECI corpus by selecting all prose files that contained paragraph markings. Our news corpus was downloaded from the WWW site of the Modern Greek weekly newspaper Eleftherotypia and consists of financial news from the period of 2001–2002. A corpus of parliamentary proceedings was again created from the Europarl corpus [Koehn 2002] by randomly selecting a subset of the Greek section.

[6]See `http://www.gutenberg.net/`. For copyright reasons, this web site mainly contains books published before 1923.
[7]See `http://www.elsnet.org/eci.html`.

	fiction	news	parliament
English	1,140,000	1,156,000	1,156,000
German	2,500,000	4,100,000	3,400,000
Greek	563,000	1,500,000	1,500,000

Table I. Number of words per corpus

Parts of the data were further pre-processed to insert sentence boundaries. We trained a publicly available sentence splitter [Reynar and Ratnaparkhi 1997] on a small manually annotated sample (1,000 sentences per domain per language) and applied it to our corpora. Table I shows the corpus sizes. We tried to use corpora of similar size for each domain of a given language. However, due to limitations in the availability of data this was not always possible. Our corpus sizes also vary between languages. In general, we tried to make the data sets for German and Greek larger than for English because the corpora for these languages were less carefully constructed and annotated and we wanted to counteract the effect of noise contained in the corpora or introduced by the automatic sentence splitting. All corpora were split into training (72%), development (24%) and test set (4%).

3.6 Evaluation Measures

Table II shows the evaluation measures we used to assess the performance of BoosTexter on the paragraph segmentation task. The first four measure performance in terms of the number of true positives (*tp*, the number of paragraph starting sentences classified correctly), true negatives (*tn*, the number of non-paragraph starting sentences classified correctly), false positives (*fp*, the number of non-paragraph starting sentences classified as paragraph starting ones) and false negatives (*fn*, the number of paragraph starting sentences classified as non-paragraph starting ones).

Accuracy is defined as the number of correctly assigned labels divided by the number of all classification decisions. As the distribution of class labels in the paragraph prediction task is typically strongly skewed towards the majority class (i.e., there are many more sentences which are not paragraph initial than sentences which are paragraph initial), the accuracy will be relatively high for a classifier which always predicts the majority class. However, in most applications a classifier which never inserts a paragraph break is clearly not very useful. Instead one would prefer a classifier which optimises the number of true positives (i.e., correctly predicted paragraph breaks). The F-measure takes into account true positives, false negatives and false positives but not true negatives and is therefore better suited to assess how well a classifier identifies correct breaks. Hence we report both accuracy and F-measure in our experiments.

While accuracy and F-measure are standard evaluation methods in machine learning, they have the disadvantage of not being sensitive to near misses. For example, a model will not be given partial credit if it is slightly off, e.g., if it misplaces a paragraph boundary by just one sentence. This also means that accuracy and F-measure do not discriminate very well between models which are slightly off and models which place boundaries in completely the wrong places. To remedy this, Beeferman et al. [1999] proposed an alternative evaluation metric for text segmentation, which encodes how likely it is that a segmenter wrongly places two

Accuracy	$\frac{tp+tn}{tp+fp+tn+fn}$
Precision	$P = \frac{tp}{tp+fp}$
Recall	$R = \frac{tp}{tp+fn}$
F-measure	$F = \frac{2PR}{P+R}$
WindowDiff	$WD(ref, hyp) = \frac{1}{N-k} \sum_{i=1}^{N-k} (\lvert b(ref_i, ref_{i+k}) - b(hyp_i, hyp_{i+k}) \rvert > 0)$

Table II. Evaluation measures

adjacent sentences into different segments.

Pevzner and Hearst [2002] point out some problems with Beeferman et al's. [1999] metric (e.g., it penalizes false negatives more heavily than false positives and over-penalizes near-misses) and provide an extended version, called *WindowDiff*. WindowDiff uses a sliding window of length k (typically set to half of the true segment size). At each position, the number of boundaries within the window is determined for both the automatically derived segmentation and the gold standard segmentation. If the number of boundaries is not the same, a penalty is assigned. Penalties are summed over the whole text and then normalised so that the metric returns a value between 0 (segmentations are identical) and 1 (segmentations are maximally different).

The precise formulation of WindowDiff is given in Table II, where $b(i, j)$ represents the number of boundaries between positions i and j in the text, N represents the number of sentences in the text, *ref* are the topic boundaries in the gold standard and *hyp* are the boundaries hypothesised by the automatic segmenter. Like accuracy, WindowDiff is also slightly biased towards classifiers which default to the majority class. A further disadvantage is that WindowDiff values on their own are slightly more difficult to interpret than accuracy or F-scores, however they can still be used to compare different segmentations.[8] Because WindowDiff is a popular evaluation measure for text segmentation we included it in our evaluation.

4. EXPERIMENTS

In this section, we describe and analyse the experiments we performed using BoosTexter. In all our experiments, we trained and tested BoosTexter as follows: first we optimised the number of iterations on the development set by training BoosTexter for 500 iterations; then we re-trained BoosTexter with the number of iterations that had led to the lowest error rate on the development set and tested on the test set. Throughout this paper we report the performance of the optimised models on the test set.

We conducted five experiments in total. First, we assessed whether humans agree on identifying paragraph boundaries thus providing an upper bound for the task. Second, we investigated whether it is possible predict paragraph breaks based solely on non-syntactic features (i.e., surface and language modelling features) are

[8]Pevzner and Hearst [2002] mention that they found evidence that the WindowDiff metric grows approximately linearly with the difference between two segmentations.

	Kappa	Acc	F-score	WDiff
EngFiction	.72	88.58	80.00	.238
EngNews	.47	77.45	62.72	.228
EngParl	.76	88.50	86.54	.115
GerFiction	.76	88.67	85.77	.113
GerNews	.70	85.67	82.35	.143
GerParl	.49	76.00	68.05	.240
GreFiction	.57	88.00	64.71	.120
GreNews	.61	82.94	74.26	.170
GreParl	.61	90.83	66.41	.092

Table III. Human agreement on the paragraph identification task

used (Section 4.2). In a third experiment (Section 4.3), we examined the role of syntactic information in the paragraph segmentation task. In a fourth experiment (Section 4.4) we looked at the effect of the training set size on the performance of our models. Finally, we assessed whether an unsupervised text segmentation method can be used for the paragraph detection task instead of our approach (Section 4.5).

Before we give the details of our experiments, we first describe the corpora we used (Section 3.5) and report on a study that assessed how well humans can predict paragraph breaks (Section 4.1), thus establishing an upper bound for the task.

4.1 Upper Bound

We established an upper bound against which our automatic methods could be compared by conducting an experiment that assessed how well humans agree on identifying paragraph boundaries. For each language and domain, 3 to 5 participants were given an extract from the original test set (consisting of approximately 200 sentences) and asked to insert paragraph breaks as they deemed appropriate. No other instructions were given, as we wanted to see whether they could independently perform the task without any specific knowledge regarding the domains and their paragraphing conventions.

We measured the agreement of the judges using the Kappa coefficient [Siegel and Castellan 1988] but also report accuracy (Acc), F-measure (F-score) and WindowDiff (WDiff) to facilitate comparison with our models. All measures are computed in a pairwise fashion and their mean is reported. Our results are shown in Table III.

As can be seen, participants tend to agree with each other on the task. But no clear trends emerge regarding which domains are most difficult. For English the lowest agreement is achieved for the news domain. For German, agreement is relatively high for the news domain, whereas least agreement is achieved for the parliamentary proceedings; for Greek the fiction domain seems to be most difficult. These differences are probably due to the individual corpora we used. For example, the English news corpus consists of Wall Street Journal texts, which are written in a particular style and have somewhat idiosyncratic paragraphing conventions (with an average paragraph length of just two sentences). Determining paragraph boundaries in such texts may be difficult for non-experts, i.e., people unfamiliar with that particular writing style. The German news corpus, on the other hand, consists of texts from the Frankfurter Rundschau which include passages such as

weather forecasts and cinema listings and are therefore probably easier to segment. Furthermore, while the fiction domain did not normally cause our subjects too many problems, the Greek fiction text proved fairly difficult, possibly because it was an extract from an epistolary novel. Overall it seems that the ease with which non-experts can re-insert paragraph breaks depends not so much on the genre of a text but more on the properties of the text itself; i.e., texts which are difficult to paragraph can probably be found in all genres.

4.2 The Influence of Non-syntactic Features

In our first set of experiments, we investigated whether it is possible to exploit relatively shallow textual cues to predict paragraph breaks automatically. We ran BoosTexter on all nine corpora but only used the surface and language modelling features, which we will collectively refer to as *non-syntactic features*. To assess the contribution of individual features to the classification task, we built a set of one-feature classifiers in addition to a combined classifier which was based on all features. Tables IV to VI show our results for the one-feature classifiers and for the combined classifier (all_{nonsyn}) for all languages and domains. As explained in Section 3.6 we report accuracy (Acc), F-measure (F-score), and WindowDiff (WDiff). But we mainly base our discussion on the F-measure results, as the other two evaluation measures tend to be biased towards classifiers which default to the majority class. We use NA (short for non-applicable) in cases where the F-measure cannot be calculated. This arises when our models assign a single label to all instances in the test data. We use boldface to highlight the results of the combined classifier as well as the results of the three best performing one-feature classifiers for each evaluation measure.

The length of the language and character models was optimised on the development set. A single best model was then applied on the test set. These models are indicated as P_{LM} (best language model), P_{CM} (best character model), and H (entropy rate estimated with best language model).[9]

BoosTexter's performance was further compared against two baselines: B_d and B_m. The former is distance-based and was obtained by hypothesising a paragraph break after every d sentences. We estimated d in the training data by counting the average number of sentences between two paragraphs. The second baseline, B_m, defaults to the majority class, i.e., assumes that the text does not have paragraph breaks.

For all languages and domains, the combined models perform better than the baselines (under all evaluation measures). In order to determine whether this difference is significant we compared the models' precision and recall against the distance baseline B_d using a χ^2 test. The diacritic $*$ ($\not*$) in Tables IV–VI indicates whether a given model is (not) significantly different (both in terms of precision and recall) from the baseline.[10] As can be seen, significant results are achieved across the board

[9]Which language and character models perform best varies slightly across corpora but no clear trends emerge.

[10]Throughout this paper we apply significance tests separately on precision and recall. Unless otherwise stated, when we mention that a model is significantly different from another model, we imply both in terms of precision and recall.

	EngFiction			EngNews			EngParl		
Feature	Acc	F-score	WDiff	Acc	F-score	WDiff	Acc	F-score	WDiff
B_m	71.04	NA	.485	51.44	NA	.486	69.38	NA	.536
B_d	60.16	26.16	.566	51.73	41.08	.509	59.50	32.01	.537
$Dist_s$	71.07	0.20	.585	57.74	57.47	.423	54.02	24.98	.437
$Dist_w$	71.04	0.70	.485	**63.08**	**64.31**	**.369**	65.64	46.94	.525
$Length$	72.08	15.36	.470	56.11	37.34	.439	68.45	15.61	.529
$Position$	71.04	NA	.485	49.18	47.91	.508	38.71	40.26	.842
$Quote_p$	**80.84**	**63.52**	**.309**	56.25	34.63	.438	30.62	46.88	.924
$Quote_c$	**80.64**	**64.45**	**.313**	54.95	29.71	.450	31.00	46.93	.921
$Quote_i$	71.04	NA	.485	51.44	NA	.486	30.62	46.88	.924
$FinPun$	72.08	0.78	.469	54.18	10.85	.458	71.75	15.21	.512
W_1	72.96	22.17	.454	57.74	59.81	.423	**82.05**	59.35	**.328**
W_2	73.42	30.45	.433	58.51	60.10	.415	**80.62**	**66.73**	**.328**
W_3	73.68	30.27	.433	**59.90**	**64.30**	**.401**	**80.73**	**66.73**	**.324**
W_{all}	**73.97**	32.19	**.429**	**61.78**	57.56	**.382**	75.06	**61.97**	.413
W_{over}	71.04	NA	.485	54.95	41.69	.448	68.12	22.90	.536
P_{LM}	72.41	14.57	.461	50.48	**60.75**	.466	56.22	35.73	.538
H	59.77	**34.84**	.464	52.21	58.51	.478	67.95	15.65	.527
P_{CM}	72.10	15.49	.458	53.70	56.91	.426	69.33	19.35	.513
all_{nonsyn}	**82.45**	**66.66***	**.307**	**70.91**	**68.92***	**.291**	**82.82**	**69.23***	**.310**
UpperBound	88.58	80.00	.238	77.45	62.72	.228	88.50	86.54	.115

Table IV. The contribution of non-syntactic features for English. The diacritic $*$ ($\not*$) indicates whether a model is (not) significantly different from the distance-based baseline B_d (χ^2 test).

with the exception of Greek parliamentary proceedings. On this data set, BoosTexter significantly outperforms the baseline in terms of precision but not recall. Also notice that in most cases the combined classifier yields performances lower than the upper bound. This observation holds for all three evaluation measures. The sole exception is the English news domain where BoosTexter outperforms the upper bound by 6.2% when the F-measure is taken into account. We attribute this to the unfamiliarity of our subjects with the Wall Street Journal style of writing (see previous section).

In general, the best performing features vary across domains but not languages. For fiction, quotes ($Quote_p$, $Quote_c$) and punctuation ($FinPun$) seem to be useful for predicting paragraph breaks. This reflects the fact that fiction texts often contain a significant proportion of dialogue and speaker turns are typically signalled by paragraph breaks. Word features (W_1–W_3, W_{all}) are also important for the fiction domain but to a lesser extent. For the news and parliamentary domains, on the other hand, the word features yield the best individual feature performances. Table VII shows word combinations that are good predictors of paragraph starting sentences (i.e., BoosTexter associates high confidence in their predictions). The table focuses on English but similar word combinations were selected for the other two languages. Most word combinations shown in Table VII seem plausible. For example, in the fiction domain, BoosTexter chose sequences such as *Yes*, *No* and *Of course*, which are likely to appear at the beginning of a speaker turn and are therefore good predictors of a paragraph break. In the news domain, sequences such as *In early trading* and *In addition to* were chosen. Again these are likely to occur

	GerFiction			GerNews			GerParl		
Feature	Acc	F-score	Wdiff	Acc	F-score	WDiff	Acc	F-score	Wdiff
B_m	75.75	NA	.417	68.24	NA	.541	66.17	NA	.338
B_d	65.44	21.93	.525	59.03	27.82	.549	58.26	29.06	.544
$Dist_s$	75.80	0.40	**.417**	68.25	0.07	.541	66.23	0.34	.338
$Dist_w$	75.80	0.40	**.417**	67.70	16.14	.487	67.09	27.42	.329
$Length$	75.75	NA	**.417**	72.55	35.40	.464	67.08	12.26	.329
$Position$	75.68	0.12	**.417**	68.05	0.95	.543	66.35	1.48	.336
$Quote_p$	72.97	**26.43**	.430	68.24	NA	.541	66.23	0.40	.338
$Quote_c$	72.35	**32.75**	.447	68.24	NA	.541	66.17	NA	.338
$Quote_i$	75.75	NA	**.417**	68.24	NA	.541	66.17	NA	.338
$FinPun$	73.15	**25.12**	.455	**76.36**	**41.12**	**.432**	69.53	18.07	.305
W_1	75.43	5.39	.425	73.84	32.53	.463	75.27	48.85	.247
W_2	75.80	8.86	.418	**74.89**	**40.39**	**.439**	**76.76**	**54.74**	**.232**
W_3	75.60	9.64	.420	**74.58**	38.52	**.445**	**76.81**	**55.70**	**.232**
W_{all}	75.50	14.80	.419	73.01	36.29	.464	**76.25**	**55.45**	**.238**
W_{over}	75.50	NA	**.417**	68.32	2.36	.540	66.27	0.79	.337
P_{LM}	**75.93**	1.84	**.417**	70.49	37.37	.478	67.40	11.55	.331
H	**75.90**	1.84	**.417**	40.94	**44.14**	.519	33.83	50.56	.328
P_{CM}	**75.98**	1.84	**.417**	73.37	40.19	.446	67.33	11.61	.327
all_{nonsyn}	**75.90**	**48.80***	**.400**	**79.61**	**57.45***	**.371**	**79.43**	**64.22***	**.206**
UpperBound	88.67	85.77	.113	85.67	82.35	.143	76.00	68.05	.240

Table V. The contribution of non-syntactic features for German. The diacritic $*$ ($\not*$) indicates whether a model is (not) significantly different from the distance-based baseline B_d (χ^2 test).

at the beginning of paragraphs in the news domain. Finally, for the parliamentary domain the chosen sequences include *Mr. Speaker*, *The hon. Gentleman* and *Order* which also seem to be plausible predictors for paragraph breaks in this domain.

The language models behave similarly across domains and languages. Their predictive power seems to be larger for the news domain than for the other two domains (i.e., the F-scores are higher for the news domain). This may reflect the fact that the structure of news stories is more rigid than that of fiction texts or parliamentary speeches. The word entropy rate yields relatively better results in German whereas sentence probability yields higher F-scores in English and Greek. The character models perform poorly in all languages and domains with the exception of news where we observe that they yield performances comparable to the language models. We attribute this to the formulaic nature of news texts. In general, our results show that language modelling features are not particularly useful for the paragraph segmentation task.

While some clear trends emerge regarding which features are good predictors in a given domain, there are also some noticeable differences. For example, the distance in words from the previous paragraph boundary ($Dist_w$) is a good indicator for paragraph breaks in the news domain for English but less useful for the other two languages. An explanation might be that the English news corpus is very homogeneous (i.e., it contains articles that not only have similar content but are also structurally alike). The Greek news corpus is slightly less homogeneous; while it mainly contains financial news articles there are also some interviews. Hence, there is greater variation in paragraph length, which means that the distance feature is

	GreFiction			GreNews			GreParl		
Feature	Acc	F-score	WDiff	Acc	F-score	WDiff	Acc	F-score	WDiff
B_m	67.57	NA	.559	53.99	NA	.460	76.25	NA	.428
B_d	59.00	28.60	.558	52.41	40.02	.509	66.48	23.46	.501
$Dist_s$	67.68	0.64	.559	57.94	15.78	.421	76.30	0.43	.428
$Dist_w$	68.31	4.42	.555	59.76	23.56	.402	76.30	0.43	.428
$Length$	67.57	NA	.559	60.84	44.93	.392	76.55	2.54	.424
$Position$	67.57	NA	.559	56.52	17.09	.435	76.25	NA	.428
$Quote_p$	**72.80**	**48.72**	**.430**	58.00	16.15	.420	76.30	0.43	.428
$Quote_c$	**71.03**	**37.19**	**.468**	53.99	NA	.460	76.25	NA	.428
$Quote_i$	67.57	NA	.559	53.99	NA	.460	76.25	NA	.428
$FinPun$	**73.33**	30.71	**.474**	59.86	22.6	.401	76.55	2.54	.423
W_1	67.05	28.89	.521	67.41	**61.73**	.326	76.81	10.94	.414
W_2	66.37	29.88	.514	**68.22**	**63.86**	**.318**	**78.48**	**23.51**	**.391**
W_3	67.63	**33.65**	.493	**67.88**	**62.29**	**.321**	**78.43**	**21.48**	**.391**
W_{all}	67.78	33.33	.507	**67.88**	60.16	**.321**	**77.26**	**17.98**	**.401**
W_{over}	67.57	NA	.559	55.00	18.84	.450	76.35	0.85	.428
P_{LM}	67.83	1.91	.559	56.29	59.71	.386	75.69	16.72	.423
H	66.89	17.90	.537	56.12	11.43	.439	76.40	2.52	.425
P_{CM}	67.68	0.96	.557	60.64	50.84	.383	76.14	5.25	.424
all_{nonsyn}	**77.67**	**59.37***	**.388**	**76.31**	**72.97***	**.237**	**79.86**	**38.89⁄***	**.357**
UpperBound	88.00	64.71	.120	82.94	74.26	.170	90.83	66.41	.092

Table VI. The contribution of non-syntactic features for Greek. The diacritic $*$ ($\not*$) indicates whether a model is (not) significantly different from the distance-based baseline B_d (χ^2 test).

EngFiction	EngNews	EngParl
Yes	In New York	To ask the
No	For the nine	The Prime Minister
What do you	In composite trading	My hon Friend
Oh	In early trading	Mr Speaker
What are you	In addition to	The hon Gentleman
Of course	At the same	Order
Ah	One of the	Interruption
What's the	The White House	Does my right hon
Good	In addition the	I am grateful
Don't you	In an interview	Will my right hon

Table VII. Word combinations associated with paragraph starting sentences

overtaken by the word-based features. Finally, the German news corpus is highly heterogeneous, containing not only news stories but also weather forecasts, sports results, and cinema listings. This leads to a large variation in paragraph length, which in turn means that the distance feature performs worse than the best baseline on accuracy and F-score.

The heterogeneity of the German news corpus may also explain another difference: while the final punctuation of the previous sentence ($FinPun$) is among the less useful features for English and Greek, it is the best performing feature for German. The German news corpus contains many "sentences" that end in atypical end-of-sentence markers such as semi-colons (which are found often in cinema

	Parl without Punctuation			Parl with Punctuation		
	Acc	F-score	WDiff	Acc	F-score	WDiff
English	79.41	67.87	.353	82.82	69.23	.310
German	78.83	63.01	.212	79.43	64.22	.206
Greek	78.69	29.34	.380	79.86	38.89	.357

Table VIII. The effect of removing punctuation features (*Quote_p*, *Quote_c*, *Quote_i*, and *FinPun*) from the models.

listings). These markers usually only occur within a paragraph, unlike normal end-of-sentence markers (such as a full-stop), which can occur before paragraph boundaries as well. This fact renders final punctuation a better predictor of paragraph breaks in the German corpus than in the other two corpora.

Our results show that punctuation often provides useful cues for automatic paragraph prediction. However, punctuation is typically not available in one important application for automatic paragraph boundary detection, namely in the output of speech-to-text systems. Although an investigation of how paragraph boundary detection could be interfaced with speech-to-text applications is beyond the scope of this paper, it is still worthwhile to get some idea of the influence of punctuation on paragraph break recognition. We thus conducted a further experiment in which we removed the main punctuation features (*Quote_p*, *Quote_c*, *Quote_i*, and *FinPun*) from our models before training on the parliamentary proceedings data. The latter is closest to spoken text and can provide a more realistic estimate of our models' performance when punctuation information is not taken into account. Table VIII shows the results. Although there is some expected drop in performance, the degradation is rather modest. This is especially the case for English and German where the F-measure decreases by 1.36% for the former and 1.21 for the latter.[11]

Besides punctuation, our models crucially require information about sentence boundaries. For most of our data, sentence boundaries were inserted automatically using a publicly available sentence splitter [Reynar and Ratnaparkhi 1997] (see Section 3.5). We conducted a brief error analysis to evaluate the effect of wrong sentence boundaries on our models. For this, we manually checked the sentence boundaries in the test data from the German fiction domain. This data set is challenging for automatic sentence splitters as it contains a fairly heterogeneous collection of non-contemporary texts with inconsistent spelling and punctuation. We found that one in ten of the predicted sentence boundaries are false positives (i.e., not genuine sentence boundaries) in this data set. We next examined whether BoosTexter inserted paragraph boundaries to coincide with wrong sentence boundaries. In a random sample of 120 false positives we did not find a single instance where this was the case. Hence, it seems that our models are relatively robust with respect to errors in the sentence splitting component.

[11] Note that here we rely on perfect input at the word level, which will not be the case if the text is the output of a speech recognition system.

	EngFiction			EngNews			EngParl		
Feature	Acc	F-score	WDiff	Acc	F-score	WDiff	Acc	F-score	WDiff
B_m	71.04	NA	.485	51.44	NA	.486	69.38	NA	.536
B_d	60.16	26.16	.566	51.73	41.08	.509	59.50	32.01	.537
Cue_s	71.48	4.33	.479	51.49	**63.30**	.485	40.64	**49.44**	.821
Cue_m	70.97	NA	.485	54.28	42.33	.457	59.03	30.47	.565
Cue_e	71.04	NA	.485	51.78	1.57	.482	31.61	46.92	.909
$Parse$	71.04	NA	.485	51.88	4.03	.481	30.62	46.88	.924
Num_s	71.04	NA	.485	53.56	22.60	.464	69.05	17.35	.519
Num_{vp}	71.04	NA	.485	54.18	28.29	.458	70.59	16.04	.517
Num_{np}	71.77	16.00	.470	**56.11**	51.46	**.439**	68.94	9.61	.533
Num_{pp}	71.04	NA	.485	53.61	51.19	.464	64.98	25.18	.534
Num_{adjp}	71.04	NA	.485	51.11	37.34	.489	42.62	42.68	.711
Num_{advp}	71.04	NA	.485	52.40	56.39	.476	47.96	44.90	.683
$Sign$	**75.39**	**43.06**	**.397**	**57.02**	56.47	**.430**	67.95	39.12	**.489**
$Sign_p$	**75.49**	**43.03**	**.398**	**59.18**	**59.32**	**.408**	**70.76**	24.68	.499
$Childr_{s1}$	71.69	7.12	.475	55.87	25.61	.441	**79.35**	**49.93**	**.389**
$Childr_s$	**75.34**	**42.46**	**.408**	55.53	29.76	.445	**79.52**	**51.31**	**.386**
$Branch_s$	71.35	19.83	.458	55.82	46.35	.442	69.11	17.38	.519
$Branch_{vp}$	71.33	15.29	.469	53.46	13.11	.465	70.48	18.04	.515
$Branch_{np}$	71.77	16.00	.470	**56.11**	58.74	**.439**	33.09	46.83	.899
$Branch_{pp}$	71.04	NA	.485	49.09	**61.73**	.509	64.92	25.15	.534
$TreeDepth$	72.57	19.92	.458	54.04	52.25	.460	69.00	16.35	.523
all_{syn}	77.56	50.63	.384	61.92	61.33	.381	74.45	53.69	.412
all_{nonsyn}	82.45	66.66$\not\dagger$	.307	70.91	68.92$\not\dagger$	.291	82.82	69.23†	.310
all	**82.99**	**67.33**$*\not\ddagger$	**.299**	**71.97**	**70.57**$*\not\ddagger$	**.280**	**83.53**	**70.25**$*\not\ddagger$	**.297**
UpperBound	88.58	80.00	.238	77.45	62.72	.228	88.50	86.54	.115

Table IX. Syntactic features for English. We used a χ^2 test to compute statistical significance: $*$ ($\not*$) indicates whether models which use all features (all) are (not) significantly different from the distance-based baseline B_d; † ($\not\dagger$) indicates whether models which only use syntactic features (all_{syn}) are (not) significantly different from those which only use non-syntactic features (all_{nonsyn}); ‡ ($\not\ddagger$) indicates whether models which use all features (all) are (not) significantly different from those which only use non-syntactic features (all_{nonsyn}).

4.3 The Influence of Syntactic Features

Our third set of experiments investigated the usefulness of syntactic features for the English (see Table IX) and German data (see Table X). Again, we created one-feature classifiers and a classifier that combines all syntactic features (all_{syn}), including discourse cues for English. Tables IX and X repeat the performances of the two baselines (B_d and B_m) and the combined non-syntactic models (all_{nonsyn}). Finally, the results of a classifier (all) that combines all features (i.e., syntactic and non-syntactic) are also presented. The three best results for a given evaluation measure and the results of the combined model (all) are again shown in boldface.

As can be seen, classifiers solely trained on syntactic features (all_{syn}) generally perform worse than classifiers trained on non-syntactic features (all_{nonsyn}). A χ^2 test revealed that the difference between all_{syn} and all_{nonsyn} is statistically significant for the parliamentary proceedings domain (indicated by † in Table IX) but not for fiction or news (indicated by $\not\dagger$). When syntactic and non-syntactic features are combined (all), the F-score increases by at most 1–2% compared to

	GerFiction			GerNews			GerParl		
Feature	Acc	F-score	WDiff	Acc	F-score	WDiff	Acc	F-score	WDiff
B_m	75.75	NA	.417	68.24	NA	.541	66.17	NA	.338
B_d	65.44	21.93	.525	59.03	27.82	.549	58.26	29.06	.544
$Parse$	**75.75**	NA	**.417**	68.24	NA	.541	66.17	NA	.338
Num_s	**75.75**	NA	**.417**	68.24	NA	.541	66.17	NA	.338
Num_{vp}	**75.75**	NA	**.417**	68.24	NA	.541	66.17	NA	.338
Num_{np}	**75.75**	NA	**.417**	68.24	0.14	.540	**66.22**	1.48	.338
Num_{pp}	**75.75**	NA	**.417**	68.24	NA	.541	66.05	2.14	.340
Num_{adjp}	**75.75**	NA	**.417**	68.22	0.62	.541	**66.22**	**2.82**	.338
Num_{advp}	**75.75**	NA	**.417**	68.24	NA	.541	66.17	NA	.338
$Sign$	75.57	**2.20**	.420	**69.87**	**11.68**	**.519**	**67.92**	**25.27**	**.321**
$Sign_p$	75.57	**2.20**	.420	**70.09**	**12.89**	**.517**	**67.92**	**25.27**	**.321**
$Childr_{s1}$	75.72	**4.53**	.419	**69.28**	7.21	**.525**	66.17	NA	.329
$Childr_s$	75.00	**9.26**	.422	69.20	6.58	.526	66.17	NA	**.320**
$Branch_s$	**75.75**	NA	**.417**	68.24	NA	.541	66.17	NA	.338
$Branch_{vp}$	**75.75**	NA	**.417**	68.24	NA	.541	66.17	NA	.337
$Branch_{np}$	**75.75**	0.20	**.417**	68.25	0.07	.541	66.17	NA	.338
$Branch_{pp}$	**75.75**	NA	**.417**	68.24	NA	.541	66.17	NA	.338
$TreeDepth$	**75.75**	NA	**.417**	69.10	**11.26**	**.525**	66.03	1.76	.340
all_{syn}	74.59	8.48	.436	69.97	12.23	.519	70.05	32.20	.299
all_{nonsyn}	75.90	48.80†	.400	79.61	57.45†	.371	79.43	64.22†	.206
all	**75.95**	**47.51**$*\not\ddagger$	**.403**	**79.93**	**59.31**$*\not\ddagger$	**.367**	**79.31**	**63.96**$*\not\ddagger$	**.207**
UpperBound	88.67	85.77	.113	85.67	82.35	.143	76.00	68.05	.240

Table X. Syntactic features for German. We used a χ^2 test to compute statistical significance: $*$ ($\not*$) indicates whether models which use all features (all) are (not) significantly different from the distance-based baseline B_d; † (/†) indicates whether models which only use syntactic features (all_{syn}) are (not) significantly different from those which only use non-syntactic features (all_{nonsyn}); ‡ (/‡) indicates whether models which use all features (all) are (not) significantly different from those which only use non-syntactic features (all_{nonsyn}).

using non-syntactic features alone (all_{nonsyn}). This difference is not statistically significant (indicated by $\not\ddagger$ in Table IX). The largest improvement is achieved in the news domain. This is due to the fact that Charniak's [2000] parser was trained on the same corpus (i.e., the Penn Treebank) and therefore the parse trees (and by extension the syntactic features derived from these parse trees) are more accurate than for the other two domains.

For German, the syntax-based classifiers perform significantly worse than the classifiers trained on non-syntactic features (see Table X). In fact, the addition of syntactic features harms the performance of the combined classifier (all) for the fiction and parliamentary domains (see Table X). For the news domain, adding syntactic features increases the F-score marginally (by 1.86%), which is, however, not statistically significant. Again the difference between the news domain and the other two domains can be explained by the fact that the German parser, too, was trained on news texts.

The syntactic features seem to be less domain dependent than the non-syntactic ones. In general, the part-of-speech signature features ($Sign$, $Sign_p$) tend to be fairly good predictors (i.e., they lead to relatively high F-scores). Examples of part-of-speech signatures with high confidence for paragraph starting sentences are

EngFiction	EngNews	EngParl
UH	NNP NNP	NNP NNP
NNP VBD	NNP NNP NNP	NNP NNP NNP
NNP	NNP	DT NNP NNP
NNP VBD PRP$ NN	NNP NNPS	VB
UH UH	NNP NNP VBD CD TO CD	NN
NNP NNP	NNP NNP JJ NNS NN CD	NNP NNP NNP NNP
UH VBD NNP	NNP NNP CD CD	UH
NNP VBD RB	NNP NNP CD TO CD	NNP VB
UH NNP	JJ NNP	DT JJ

Table XI. Part-of-speech signatures associated with paragraph starting sentences (UH: interjection, NNP: proper noun singular, VBD: verb past tense, PRP$: possessive pronoun, NN: noun singular or mass, RB: adverb, NNPS: proper noun plural, CD: cardinal number, TO: *to*, JJ: adjective, DT: determiner, VB: verb base form)

shown in Table XI. As can be seen noun phrases (e.g., NNP NNP, NNP NNPS) are frequently found in paragraph initial sentences, especially for the news and parliamentary domains. Interjections (e.g., UH, UH NNP) are good predictors for the fiction domain, presumably because they commonly indicate speaker turns. For the English parliamentary texts, however, part-of-speech signatures are outperformed by features that encode the top level complexity of the parse tree ($Childr_{s1}$, $Childr_s$).

For the news domain, features dealing with the NPs and PPs of the sentence ($Branch_{np}$, $Branch_{pp}$, Num_{np}, Num_{pp}) also lead to a relatively good performance for English. This is plausible since paragraph initial sentences in the Wall Street Journal often contain named entities, such as company names, which are typically fairly complex and thus have a high branching factor. However, for German news texts, the tree depth ($TreeDepth$) is a better predictor.

Discourse cues were only used for the English data, where their presence sentence initially (Cue_s) proved a useful predictor in both parliamentary and news texts but not in fiction texts.

4.4 The Effect of Training Size

Our experiments in the previous sections have shown that BoosTexter significantly outperforms our baselines for most domains. This good performance may however be due to the large amounts of training data available. While training data with paragraph boundaries is generally easy to obtain for written text, it may not be readily available in large amounts for speech transcripts. In this case manual annotation might be necessary. In the experiments reported in this section, we investigated how much manual annotation would be necessary for BoosTexter to achieve good performance. We assessed this indirectly by examining the effect of the training set size on the learner's performance.

We conducted our experiments on all three domains and languages. From each training set we created ten progressively smaller data sets, the first being identical to the original set, the second containing 9/10 of sentences in the original training set, the third containing 8/10, etc. BoosTexter was trained on each of these sets

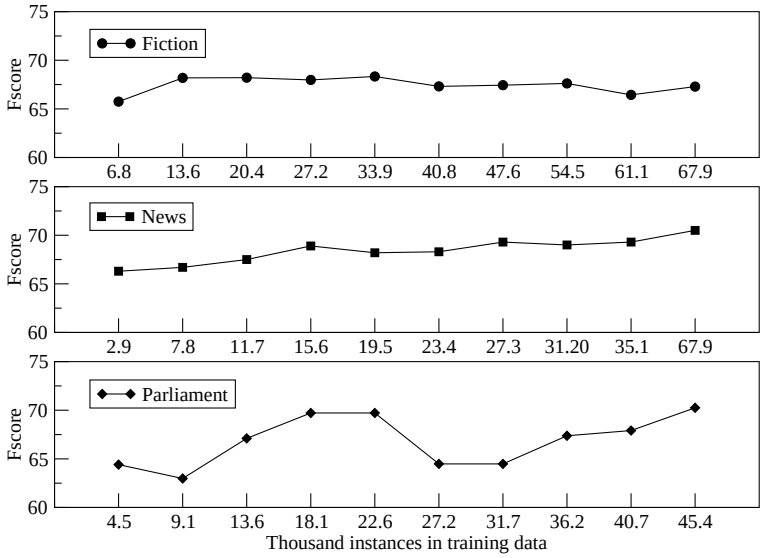

Fig. 1. Learning Curves for English

(using the complete feature set available for each language),[12] and tested on the test set.

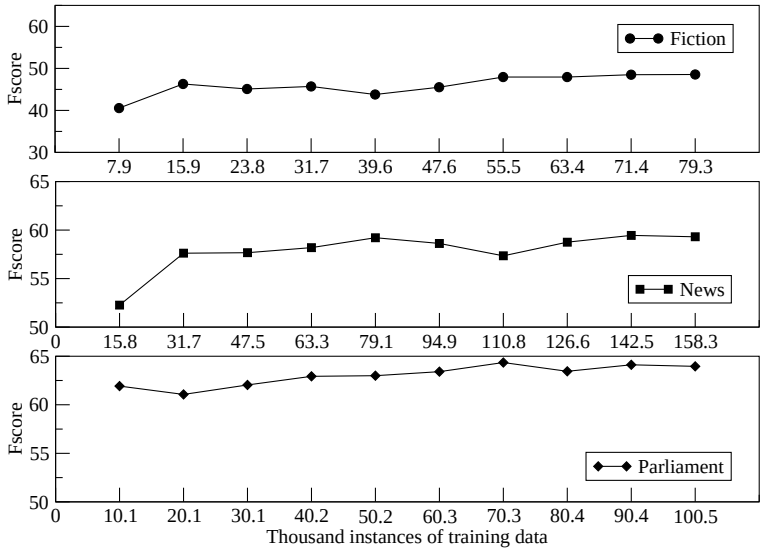

Fig. 2. Learning Curves for German

[12] As in the previous experiments, we optimised the parameter settings for each BoosTexter model separately on the development set.

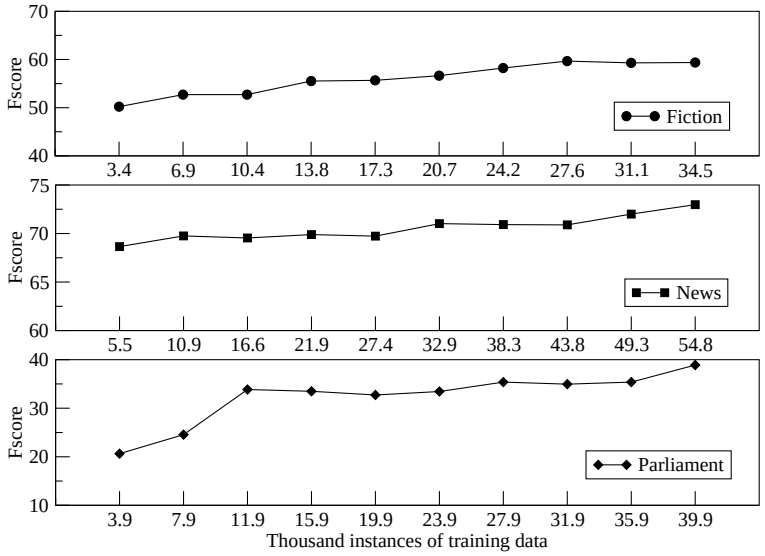

Fig. 3. Learning Curves for Greek

Figures 1 to 3 show the learning curves obtained this way. The curves reveal that increasing the amount of training data yields improved performance. The effect is starkest for the Greek parliament domain, where F-score increases by 20% when the size of the training data is ten times larger. However, even the smallest of our training sets (containing around 3,000 sentences) is big enough to outperform the best baseline. Hence, it is possible to do well on this task even with a fairly small amount of training data. This means that in situations where no training data is available, only a relatively modest annotation effort is required.

4.5 Comparison with Unsupervised Methods

Our results from the previous section show that only a relatively small training set is required to outperform the best baseline. Furthermore, corpora annotated with paragraph boundaries are usually easy to obtain. However, our results from the experiments in Section 4.2 also suggest that there are domain specific differences regarding which features are good predictors of paragraph breaks. This means that paragraph segmentation models would have to be retrained for every new domain.

This is in marked contrast with unsupervised methods developed for text segmentation [Hearst 1997; Choi 2000; Utiyama and Isahara 2001], which do not require training data and therefore are, at least in theory, applicable across several text types without requiring additional tuning (see Section 2 for details). Of course paragraph boundary identification is different from text segmentation in that paragraph boundaries do not always correspond to topic boundaries. However, given that at least some paragraph boundaries will coincide with topic boundaries, it is possible that existing unsupervised text segmentation methods still lead to relatively good results on the paragraph segmentation task.

To determine whether this is a promising route we compared our supervised

method against Utiyama and Isahara's [2001] topic segmentation algorithm which is fully unsupervised and based on a statistical model that finds the maximum-probability segmentation of a given text. The most likely segmentation ($\hat{S}$) for a text is given by:

$$\hat{S} = \underset{s}{\operatorname{argmax}} P(W|S)P(S) \tag{4}$$

where W is the text and S is a segmentation of the text. By assuming that different topics have different word distributions and that the words within the scope of a topic are statistically independent of each other given the topic, $P(W|S)$ can be defined:

$$P(W|S) = P(W_1, W_2 \ldots W_m|S) = \prod_{i=1}^{m} \prod_{j=1}^{n_i} P(w_j^i|S_i) \tag{5}$$

where n_i is the number of words in segment S_i, w_j^i the j-th word in S_i, and $\sum_{i=1}^{m} = n_i$. The definition of $P(S)$ can vary depending on prior information about the possibility of segmentation S. For example, if information about the average segment length is known, it can be easily incorporated into $P(S)$. In cases where no prior information is available, Utiyama and Isahara [2001] make use of a description length prior:

$$P(S) = 2^{-l(S)} \tag{6}$$

where $l(S) = m \log n$ bits.

To find the maximum probability segmentation $\hat{S}$, Utiyama and Isahara [2001] define $C(S)$, the cost of segmentation S and then accordingly minimise it using a graph-based search algorithm:

$$\hat{S} = \underset{s}{\operatorname{argmin}}\, C(S) = \underset{s}{\operatorname{argmin}} - \log P(W|S)P(S) \tag{7}$$

Utiyama and Isahara's [2001] algorithm outperformed Choi's [2000] method as well as TextTiling [Hearst 1997], DotPlot [Reynar 1998], and Segmenter [Kan et al. 1998]. The algorithm was tested on an artificial corpus consisting of a concatenation of segments where each segment corresponded to the first n sentences of a randomly selected document from the Brown corpus [Choi 2000].

We used Utiyama and Isahara's [2001]'s probabilistic model[13] to segment our test data in all three domains and languages. We then compared the output of the automatic text segmenter to our gold standard. We experimented with the default description length prior $P(S)$ (see Equation 6) but also with a prior that takes into account the average paragraph length, which we estimated from our training data. The results are presented in Table XII.

When a non-informative prior (noPr) is used, the unsupervised method inserts very few paragraph breaks. While this leads to relatively high accuracy, it results in a very low F-score. Better F-scores are obtained with the average paragraph length prior (avglenPr). With the exception of the Greek parliament domain, Utiyama and Isahara's [2001] algorithm outperforms the distance-based baseline (B_d) when the

[13] Available at `http://www.crl.go.jp/jt/a132/members/mutiyama/softwares.html`.

	EngFiction			EngNews			EngParl		
method	Acc	F-score	Wdiff	Acc	F-score	WDiff	Acc	F-score	WDiff
B_m	71.04	NA	.485	51.44	NA	.486	69.38	NA	.536
B_d	60.16	26.16	.566	51.73	41.08	.509	59.50	32.01	.537
U&I (noPr)	70.99	0.88	.484	51.83	1.99	.482	69.66	2.48	.533
U&I (avglenPr)	28.95	44.91#	.906	49.52	48.02#	.505	44.27	39.62#	.791
BoosTexter	82.99	67.33$	.299	71.97	70.57$	.280	83.53	70.25$	.297
UpperBound	88.58	80.00	.238	77.45	62.72	.228	88.50	86.54	.115

	GerFiction			GerNews			GerParl		
method	Acc	F-score	Wdiff	Acc	F-score	Wdiff	Acc	F-score	Wdiff
B_m	75.75	NA	.417	68.24	NA	.541	66.17	NA	.338
B_d	65.44	21.93	.525	59.03	27.82	.549	58.26	29.06	.544
U&I (noPr)	75.76	0.62	.417	68.27	0.35	.540	66.25	0.50	.338
U&I (avglenPr)	63.61	24.97#	.538	56.81	32.00#	.655	55.61	34.40#	.444
BoosTexter	75.95	47.51$	.403	79.93	59.31$	.367	79.31$	63.96$	.207
UpperBound	88.67	85.77	.113	85.67	82.35	.143	76.00	68.05	.240

	GreFiction			GreNews			GreParl		
method	Acc	F-score	WDiff	Acc	F-score	WDiff	Acc	F-score	WDiff
B_m	67.57	NA	.559	53.99	NA	.460	76.25	NA	.428
B_d	59.00	28.60	.558	52.41	40.02	.509	66.48	23.46	.501
U&I (noPr)	67.68	0.64	.559	54.03	0.29	.460	76.30	0.85	.427
U&I (avglenPr)	58.37	35.81#	.651	46.81	42.20#	.520	61.75	19.49#	.574
BoosTexter	77.67	59.37$	.388	76.31$	72.97$	.237	79.86	38.89$	.357
UpperBound	88.00	64.71	.120	82.94	74.26	.170	90.83	66.41	.092

Table XII. Comparison of BoosTexter with Utiyama and Isahara's [2001] unsupervised text segmentation algorithm. The diacritic # (/#) indicates whether the unsupervised method is (not) significantly different from the distance-based baseline B_d; $ (/$) indicates whether the best BoosTexter model is (not) significantly different from the unsupervised method (using a χ^2 test).

informative prior is used. However, in most cases the difference is not statistically significant. The diacritic # (/#) indicates whether their model is (not) significantly better than the baseline in Table XII. The results of the unsupervised method are significantly worse than BoosTexter for all domains and languages (indicated in Table XII by the diacritic $).

We therefore conclude that unsupervised text segmentation methods cannot be readily used for the paragraph insertion task without further modification. One potential caveat is that most of the unsupervised methods rely on vocabulary changes to determine where a boundary should be placed. While topic shifts are typically indicated by changes in word distribution, paragraph boundaries are probably influenced by additional factors (see the discussion in Section 1) and therefore text segmentation methods may not be very well suited for the paragraph identification task. It is a matter of future research to examine whether it is possible to devise a special purpose unsupervised method for this task.

5. RELEVANCE FOR SUMMARISATION

In our introductory section we argued that a mechanism for automatic paragraph insertion can be useful for text-to-text generation applications such as summarisation and machine translation. To substantiate this claim we ported our paragraph segmentation method into a summarisation system and performed a user study in order to assess whether humans prefer texts with or without automatically inserted paragraph breaks. In the following we describe our method for assembling the set of experimental materials and eliciting judgements from human participants.

Materials and Design. Our evaluation focused on single-document summaries produced by the English version of SweSum [Dalianis et al. 2003], a publicly available summariser[14] that performs sentence extraction by identifying the most relevant sentences in a text. SweSum employs a state-of-the-art architecture: first tokenisation and keyword extraction take place; secondly sentences are ranked using a combination of heuristics such as typesetting (e.g., boldface), the presence of numerical data and keywords, and term frequency; thirdly the summary is generated by preserving the order of the sentences in the original document. It is important to note that SweSum does not have a dedicated paragraph insertion module; instead each sentence forms its own paragraph. When compared against gold standard summaries, SweSum achieved accuracies of 52%, 68%, and 84% for compression rates of 10%, 30%, and 40%, respectively.

We randomly selected nine newspaper articles from the portion of the Penn Treebank that formed our test data set. Average sentence length was 65.2. The articles were summarised using SweSum at a compression rate of 30%; we used a version of BoosTexter that employed our full feature set to automatically insert paragraph breaks. In addition to the summaries that contained automatic paragraph breaks, we created two baseline versions. Our first baseline preserved SweSum's default strategy of inserting a paragraph break after every sentence. Our second baseline randomly determined the number and placement of paragraph breaks. For any summary of length n, there are $n-1$ potential breaks. We determined the number of breaks per summary by randomly selecting a number m ranging from 1 to $n-1$. Then we randomly chose m from n summary sentences and assumed that they are paragraph initial. Our materials therefore contained contained $3 \times 9 = 27$ summaries.

Procedure and Subjects. We partitioned our materials into three subsets, each corresponding to the automatically inserted paragraphs (AutPar), SweSum's default summary output (Default), and the randomly generated paragraph breaks (RandPar). Each participant saw nine summaries, three from each subset. We made sure that all nine summaries corresponded to different documents, i.e., no subject saw more than one version of the same summary. Examples of the summaries our participants saw are given in Appendix A.

Participants were asked to use a seven point scale to rate the structure of the summaries on the basis of their paragraph breaks. It was explained that the summaries had been produced automatically by a computer program.

[14]The summariser is available at `http://swesum.nada.kth.se/index-eng.html`.

Version	Rating
Default	2.37
RandPar	3.61
AutPar	4.66

Table XIII. Mean Ratings for automatic summaries

The study was conducted remotely over the Internet. Participants first saw a set of instructions that explained the task, and had to fill in a short questionnaire including basic demographic information. Then the summaries were presented; a new random order was generated for each participant. The experiment was completed by 30 unpaid volunteers, all self-reported native speakers of English. Participants were recruited via postings to local emailing lists.

Results. We carried out an Analysis of Variance (ANOVA) to examine the effect of different types of paragraph breaks (default, random and automatic) on the summaries. Statistical tests were done using the mean of the ratings shown in Table XIII. The ANOVA revealed a reliable effect of paragraph structure: $F1(3;27) = 16.30$, $p < 0.01$. Post-hoc Tukey tests indicated that summaries with automatically inserted paragraph breaks are perceived as significantly better than the default summaries ($\alpha = 0.01$) and the random baseline summaries ($\alpha = 0.05$). The latter are significantly better than the default summaries ($\alpha = 0.01$).

Our results indicate that an automatic paragraph insertion mechanism could be useful for structuring the output of automatic summarisers. Such a mechanism is preferable to the simple strategy of listing the summary sentences, especially when the compression rate is relatively high. It can be incorporated in a summarisation system and can be easily obtained for different types of texts and domains.

6. CONCLUSIONS

In this paper, we investigated whether it is possible to predict paragraph boundaries automatically using a supervised approach which exploits textual, syntactic and discourse cues. We achieved F-scores of up to 73%, for each data set, our results were significantly better than the best baseline, and in many cases came to within a few percent of human performance on the task. Our results were also significantly better than those obtained by applying an existing unsupervised text segmentation method [Utiyama and Isahara 2001] to the paragraph segmentation task.

To get some insight into whether there are cross-domain or cross-language differences, we conducted our study in three different domains and languages. We found that the most useful cues vary between domains but not languages. For the fiction domain, punctuation and quotation marks are highly predictive of paragraph boundaries, whereas for the news and parliamentary proceedings domains, word based features work better. We also found that including syntactic and discourse cues does not improve the performance very much and may even harm it in cases where the accuracy of the parser is sub-optimal. Since low-level features lead to a good performance, predicting paragraph boundaries automatically is feasible even for languages for which parsers or cue word lists are not readily available.

However, one potential drawback of our method is that it is supervised and re-

lies on the availability of training data which is annotated with paragraph breaks. This type of training data is usually easy to come by, but some manual annotation effort may be required when dealing with spoken texts, e.g., if our method is used to determine paragraph boundaries automatically in the output of a text-to-speech system. To assess how much training data is required, we experimented with training sets of different sizes and found that significant performance gains over the baseline can be obtained even with a training set of relatively modest size.

Finally, we assessed whether our method can be used to improve the output of a text summarisation system. For this we conducted a user study to determine how humans rate summaries with automatically inserted paragraph breaks. The results showed that the paragraph structure produced by our method is not only preferred over randomly inserted paragraph breaks but also over the default output of the text summariser (which places a break after each sentence). This suggests that our method is indeed a useful add-on for a text summarisation system.

The work presented here can be extended in several directions. While we experimented with languages that displayed structural, grammatical and morphological dissimilarities, it should be interesting to evaluate our paragraph segmentation algorithms on languages with more complex writing systems or languages without delimiters between words (e.g., Hindi, Chinese, Japanese). More cross-linguistic studies would further allow us evaluate the extent to which languages differ in their paragraph structure [Hinds 1979; Zhu 1999].

While our results show that unsupervised methods which are based on word distributions only achieve moderate results for the paragraph prediction task, it may be possible to devise novel unsupervised strategies which can help determine paragraph boundaries. Our results indicate that some surface features (e.g., sentence initial word n-grams, punctuation) are relatively good predictors of paragraph structure. We plan to take advantage of these cues and their distributions in developing an unsupervised approach, thus avoiding the potential cost of manual annotation in cases where paragraph boundaries are not explicitly marked. Another possibility would be to combine the predictions made by an unsupervised topic segmentation algorithm with our supervised paragraph identification method. This might improve the performance in situations where a paragraph break corresponds to a topic boundary.

Finally, we aim to investigate the usefulness of our method for speech-to-text applications. Such applications are particularly interesting, because it is not entirely clear what the paragraph structure of transcribed text should be. Furthermore, speech transcripts are typically unedited texts, and any method that segments texts into meaningful units could potentially benefit applications such as summarisation and information extraction. While our methods work well for transcripts of parliamentary speeches, these are typically carefully drafted and therefore relatively close to written texts. The situation may be different for domains which involve spoken language that is more spontaneous, e.g., transcripts of lectures or business meetings. When dealing with spoken language, it is not only the transcripts that provide useful information but also the original speech signal. We therefore aim to take advantage of the latter and explore the benefit of prosodic cues, such as pauses or intonation, since they may indicate places where a paragraph boundary could be

placed in a transcript (see the related work of Litman and Passonneau [1995] and Hauptmann and Smith [1995]).

ACKNOWLEDGMENTS

The authors acknowledge the support of EPSRC (Sporleder, grant GR/R40036/01; Lapata, grant GR/T04540/01). We are grateful to Chris Callison-Burch, Stephen Clark, Frank Keller, Alex Lascarides, and Simone Teufel for valuable comments and suggestions. This paper is a revised and extended version of Sporleder and Lapata [2004]; we thank the anonymous reviewers of that paper for their comments. Special thanks to our annotators Beatrice Alex, Colin Banard, Markus Becker, Carsten Brockmann, Roi Georgila, Ben Hachey, Ben Hutchinson, Vasilis Karaiskos, and David Talbot.

REFERENCES

Barzilay, R. and Elhadad, M. 1997. Using lexical chains for text summarization. In *Proceedings of the ACL Workshop on Intelligent Scalable Text Summarization*. Madrid, Spain, 10–17.

Beeferman, D., Berger, A., and Lafferty, J. 1999. Statistical models for text segmentation. *Machine Learning 34,* 1/3, 177–210.

Boguraev, B. K. and Neff, M. S. 2000. Discourse segmentation in aid of document summarization. In *Proceedings of the 33rd Hawaii International Conference on System Sciences*.

Bond, S. and Hayes, J. 1984. Cues people use to paragraph text. *Research in the Teaching of English 18*, 147–167.

Brants, T., Chen, F., and Tsochantaridis, I. 2002. Topic-based document segmentation with probabilistic latent semantic analysis. In *Proceedings of the 11th International Conference on Information and Knowledge Management*. 211–218.

Brown, G. and Yule, G. 1983. *Discourse Analysis*. Cambridge University Press.

Charniak, E. 2000. A maximum-entropy-inspired parser. In *Proceedings of the 1st North American Annual Meeting of the Association for Computational Linguistics*. 132–139.

Choi, F. 2000. Advances in domain independent linear text segmentation. In *Proceedings of the 1st North American Annual Meeting of the Association for Computational Linguistics*. Seattle, WA, 26–33.

Christensen, H., Kolluru, B., Gotoh, Y., and Renals, S. 2004. From text summarisation to style-specific summarisation for broadcast news. In *Proceedings of the European Conference on Information Retrieval*. Sunderland.

Clarkson, P. and Rosenfeld, R. 1997. Statistical language modeling. In *Proceedings of ESCA EuroSpeech'97*. Rhodes, 2707–2710.

Dalianis, H., Hassel, M., Wedekind, J., Haltrup, D., and Lech, C. 2003. Automatic text summarization for the scandinavian languages. In *Nordisk Sprogteknologi 2002: Årbog for Nordisk Språkteknologisk Forskningsprogram 2000-2004*, H. Holmboe, Ed. Museum Tusculanums Forlag, 153–163.

Drucker, H. and Cortes, C. 1996. Boosting decision trees. *Advances in Neural Information Processing Systems 8*, 479–485.

Drucker, H., Schapire, R., and Simard, P. 1992. Boosting performance in neural networks. *International Journal of Pattern Recognition and Artificial Intelligence 7,* 4, 705–719.

Dubey, A. 2004. The limits of lexicalization in probabilistic parsing. Ph.D. thesis, Universität des Saarlandes, Saarbrücken.

Genzel, D. and Charniak, E. 2003. Variation of entropy and parse trees of sentences as a function of the sentence number. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*. Sapporo, 65–72.

Hauptmann, A. G. and Smith, M. A. 1995. Text, speech and vision for video segmentation: The informedia project. In *Proceedings of the AAAI Fall Symposium, Computational Models for Integrating Language and Vision*. Cambridge, MA.

Hearst, M. A. 1994. Multi-paragraph segmentation of expository text. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*. 9–16.

Hearst, M. A. 1997. TextTiling: Segmenting text into multi-paragraph subtopic passages. *Computational Linguistics 23,* 1, 33–64.

Hinds, J. 1979. Organizational pattern in discourse. *Syntax and Semantics 12*, 135–157.

Hofmann, T. 2001. Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning 42,* 1, 177–196.

Kan, M.-Y. 2001. Combining visual layout and lexical cohesion features for text segmentation. Tech. rep., Columbia University, Computer Science Technical Report, CUCS-002-01.

Kan, M.-Y., Klavans, J. L., and McKeown, K. R. 1998. Linear segmentation and segment significance. In *Proceedings of the 6th International Workshop of Very Large Corpora*. Montréal, Canada, 197–205.

Keller, F. 2004. The entropy rate principle as a predictor of processing effort: An evaluation against eye-tracking data. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*. 317–324.

Knott, A. 1996. A data-driven methodology for motivating a set of coherence relations. Ph.D. thesis, Department of Artificial Intelligence, University of Edinburgh.

Koehn, P. 2002. Europarl: A multilingual corpus for evaluation of machine translation. Unpublished Draft, `http://www.isi.edu/~koehn/publications/europarl.ps`.

Litman, D. J. and Passonneau, R. J. 1995. Combining multiple knowledge sources for discourse segmentation. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*. 108–115.

Longacre, R. E. 1979. The paragraph as a grammatical unit. *Syntax and Semantics 12*, 115–134.

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

Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D., and Miller, K. J. 1990. Introduction to WordNet: An on-line lexical database. *International Journal of Lexicography 3,* 4, 235–312.

Morris, J. and Hirst, G. 1991. Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational Linguistics 17,* 1, 21–48.

Palmer, D. D. and Hearst, M. A. 1997. Adaptive multilingual sentence boundary disambiguation. *Computational Linguistics 23,* 2, 241–267.

Peng, F., Schuurmans, D., Keselj, V., and Wang, S. 2003. Language independent authorship attribution using character level language models. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*. Budapest, 267–274.

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

Reynar, J. C. 1998. Topic segmentation: Algorithms and applications. Ph.D. thesis, Computer and Information Science, University of Pennsylvania.

Reynar, J. C. and Ratnaparkhi, A. 1997. A maximum entropy approach to identifying sentence boundaries. In *Proceedings of the 5th Conference on Applied Natural Language Processing*. Washington, DC, 16–19.

Richmond, K., Smith, A., and Amitay, E. 1997. Detecting subject boundaries within text: a language independent statistical approach. In *Proceedings of the 2nd Conference on Empirical Methods in Natural Language Processing*. 47–54.

Salton, G., Singhal, A., Buckley, C., and Mitra, M. 1996. Automatic text decomposition using text segments and text themes. In *Proceedings of the 7th ACM Conference on Hypertext*. 53 – 65.

Shapire, R. E. and Singer, Y. 1999. Improved boosting algorithms using confidence-rated predictions. *Machine Learning 37,* 3, 297–336.

Shapire, R. E. and Singer, Y. 2000. Boostexter: A boosting-based system for text categorization. *Machine Learning 39,* 2/3, 135–168.

Shriberg, E., Stolcke, A., Hakkani-Tür, D., and Tur, G. 2000. Prosody-based automatic segmentation of speech into sentences and topics. *Speech Communication 32,* 1-2, 127–154.

Siegel, S. and Castellan, N. J. 1988. *Non Parametric Statistics for the Behavioral Sciences.* McGraw-Hill, New York.

Sporleder, C. and Lapata, M. 2004. Automatic paragraph identification: A study across languages and domains. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing.* 72–79.

Stark, H. A. 1988. What do paragraph markings do? *Discourse Processes 11*, 275–303.

Stevenson, M. and Gaizauskas, R. 2000. Experiments on sentence boundary detection. In *Proceedings of the 6th Applied Natural Language Processing Conference.* Seattle, WA, 84–89.

Utiyama, M. and Isahara, H. 2001. A statistical model for domain-independent text segmentation. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics.* Toulouse, 491–498.

Yaari, Y. 1997. Segmentation of expository texts by hierarchical agglomerative clustering. In *Proceedings of the 2nd International Conference on Recent Advance in Natural Language Processing.* 59–65.

Zhu, C. 1999. UT once more: The sentence as the key functional unit of translation. *Meta 44,* 3, 429–447.

Appendix A. Example Summaries

Here, we give an example of the summaries our participants saw in the three experimental conditions: summaries with a paragraph break after every sentence (Default), summaries with randomly inserted breaks (RandPar), and automatically inserted breaks using BoosTexter (AutPar).

Default Summary
Assuming the stock market doesn't crash again and completely discredit yuppies and trading rooms, American television audiences in a few months may be seeing Britain's concept of both. "Capital City" is a weekly series that premiered here three weeks ago amid unprecedented hype by its producer, Thames Television. The early episodes make you long for a rerun of the crash of 1987. According to the program's publicity prospectus, "Capital City", set at Shane Longman, a fictional mid-sized securities firm with $500 million capital, follows the fortunes of a close-knit team of young, high-flying dealers, hired for their particular blend of style, genius and energy. Turned loose in Shane Longman's trading room, the yuppie dealers do little right. Judging by the money lost and mistakes made in the early episodes, Shane Longman's capital should be just about exhausted by the final 13th week. In the opening episode we learn that Michelle, a junior bond trader, has indeed pulled off another million before lunch. Little chance that Shane Longman is going to recoup today. And a large slice of the first episode is devoted to efforts to get rid of some nearly worthless Japanese bonds (since when is anything Japanese nearly worthless nowadays?). Surprisingly, Shane Longman survives the week, only to have a senior executive innocently bumble his way into becoming the target of a criminal insider trading investigation. After all, this isn't old money, but new money, and in many cases, young money. In producing and promoting "Capital City", Thames has spent about as much as Shane Longman loses on a good day. Perhaps without realizing it, Mr Taffner simultaneously has put his finger on the problem and an ideal solution: "Capital City" should have been a comedy, a worthy sequel to the screwball British "Carry On" movies of the 1960s.

RandPar Summary
Assuming the stock market doesn't crash again and completely discredit yuppies and trading rooms, American television audiences in a few months may be seeing Britain's concept of both. "Capital City" is a weekly series that premiered here three weeks ago amid unprecedented hype by its producer, Thames Television. The early episodes make you long for a rerun of the crash of 1987. According to the program's publicity prospectus, "Capital City", set at Shane Longman, a fictional mid-sized securities firm with $500 million capital, follows the fortunes of a close-knit team of young, high-flying dealers, hired for their particular blend of style, genius and energy. Turned loose in Shane Longman's trading room, the yuppie dealers do little right. Judging by the money lost and mistakes made in the early episodes, Shane Longman's capital should be just about exhausted by the final 13th week. In the opening episode we learn that Michelle, a junior bond trader, has indeed pulled off another million before lunch. Little chance that Shane Longman is going to recoup today. And a large slice of the first episode is devoted to efforts to get rid of some nearly worthless Japanese bonds (since when is anything Japanese nearly worthless nowadays?). Surprisingly, Shane Longman survives the week, only to have a senior executive innocently bumble his way into becoming the target of a criminal insider trading investigation. After all, this isn't old money, but new money, and in many cases, young money. In producing and promoting "Capital City", Thames has spent about as much as Shane Longman loses on a good day. Perhaps without realizing it, Mr Taffner simultaneously has put his finger on the problem and an ideal solution: "Capital City" should have been a comedy, a worthy sequel to the screwball British "Carry On" movies of the 1960s.

AutPar Summary

Assuming the stock market doesn't crash again and completely discredit yuppies and trading rooms, American television audiences in a few months may be seeing Britain's concept of both.

"Capital City" is a weekly series that premiered here three weeks ago amid unprecedented hype by its producer, Thames Television. The early episodes make you long for a rerun of the crash of 1987.

According to the program's publicity prospectus, "Capital City", set at Shane Longman, a fictional mid-sized securities firm with $500 million capital, follows the fortunes of a close-knit team of young, high-flying dealers, hired for their particular blend of style, genius and energy.

Turned loose in Shane Longman's trading room, the yuppie dealers do little right. Judging by the money lost and mistakes made in the early episodes, Shane Longman's capital should be just about exhausted by the final 13th week. In the opening episode we learn that Michelle, a junior bond trader, has indeed pulled off another million before lunch. Little chance that Shane Longman is going to recoup today. And a large slice of the first episode is devoted to efforts to get rid of some nearly worthless Japanese bonds (since when is anything Japanese nearly worthless nowadays?). Surprisingly, Shane Longman survives the week, only to have a senior executive innocently bumble his way into becoming the target of a criminal insider trading investigation. After all, this isn't old money, but new money, and in many cases, young money.

In producing and promoting "Capital City", Thames has spent about as much as Shane Longman loses on a good day.

Perhaps without realizing it, Mr Taffner simultaneously has put his finger on the problem and an ideal solution: "Capital City" should have been a comedy, a worthy sequel to the screwball British "Carry On" movies of the 1960s.