Technical Report

UCAM-CL-TR-610
ISSN 1476-2986

Number 610

Influence of syntax on prosodic boundary prediction

Tommy Ingulfsen

December 2004

This technical report is based on a dissertation submitted July 2004 by the author for the degree of Master of Philosophy (Computer Speech, Text and Internet Technology) to the University of Cambridge, Churchill College.

ISSN 1476-2986

Abstract

In this thesis we compare the effectiveness of different syntactic features and syntactic representations for prosodic boundary prediction, setting out to clarify which representations are most suitable for this task. The results of a series of experiments show that it is not possible to conclude that a single representation is superior to all others. Three representations give rise to similar experimental results. One of these representations is composed only of shallow features, which were originally thought to have less predictive power than deep features. Conversely, one of the deep representations that seemed to be best suited for our purposes (syntactic chunks) turns out not to be among the three best.

Acknowledgments

I first thank my supervisors at Toshiba Research Europe, Dr. Tina Burrows and Dr. Sabine Buchholz, for supporting and motivating me. It has been a pleasure to work with them. I would also like to express my gratitude to Professor Steve Young, The University of Cambridge Engineering Department for purchasing the Boston University Radio News Corpus so that I could carry out experiments on the most well-suited corpus available. I am also tremendously grateful for the support I have had from the ILK research group at Tilburg University, in particular Dr. Antal van den Bosch, who let me use his iterative deepening script `paramsearch` and Bertjan Busser who was extraordinarily responsive in helping me use the MBSP chunker. In addition, Dan Bikel, University of Pennsylvania, deserves my sincere gratitude for explaining the internal details of his parser to me, so that I could modify it for my experiments. I have also benefited from discussions with Ian Read, University of East Anglia and from Dr. David Temperley, formerly of Carnegie Mellon University, for his comments on feature extraction from link grammar.

Contents

Chapter 1

Introduction

Prosodic, or intonational, phrasing divides utterances up into meaningful "chunks" of information that aid understanding. These prosodic chunks are formed when the speaker pauses at the junctures between words. Such a pause is known as a prosodic phrase break, and the juncture is known as a boundary site. For example, in the sentence

I wanted to go for a drive in the country.

a good place to insert a pause could be at the juncture between "drive" and "in". On the other hand, a break placed between "to" and "go" would sound strange ([9]). For longer, more complex sentences there is usually more than one acceptable arrangement of phrase breaks, often depending upon the meaning of the utterance. Consider two possible intonational phrase arrangements for the sentence "Bill doesn't drink because he's unhappy" (example due to [29]):

1. *Bill doesn't drink because he's unhappy.*
2. *Bill doesn't drink, because he's unhappy.*

Without the comma, which here signifies a prosodic phrase break, the reader may conclude that Bill drinks, but not from unhappiness. The second alternative, which places a pause after the comma, may be understood as if Bill avoids drinking when he is unhappy. Thus, the assignment of appropriate phrase break boundaries is crucial to conveying meaning and is consequently an important part of the generation of prosody in Text-to-Speech (TTS) systems. The resulting prosodic phrase structure is vital in TTS for improving intelligibility and naturalness, in particular with longer texts.

Prosodic boundary prediction is thought to require both shallow information, like the presence of punctuation, and deep information, such as syntactic relationships between constituents, discourse-level knowledge and semantic knowledge. However, extracting all the information that could be potentially useful is currently infeasible for real-time systems. Past research has therefore focused on exploring alternative methods and representations for the placement of breaks. These approaches include manually written rules that employ detailed human knowledge [5, 6, 7], machine learning methods [22, 29] and probabilistic methods [9]. In general we distinguish between two types of phrase breaks, *intonational phrase breaks* which mark strong prosodic boundaries such as the ends of sentences, and *intermediate phrase breaks* which are used for relatively weak boundaries. Some studies do not distinguish between these two types of phrase breaks, and only differentiate between breaks and non-breaks. There is also variation between the methods in which features are used to make the predictions. The simplest features are extracted from information given directly in the text, such as the type of punctuation and the number of words, while the more complex features, such as syntactic and syllable information, require a degree of analysis.

Unfortunately, there is also considerable variation in the data used to evaluate the methods. Two common corpora available for research in universities are the MARSEC database of spoken British English and the Boston University Radio News Corpus (BURNC), but a variety of other data sets, of different sizes, have also been used. Many of these data sets have been prepared for internal use in companies and universities and are not generally available. Making it even more difficult to compare the different methods, researchers have also used dissimilar evaluation metrics. Because of these disparities, it is difficult to determine from the existing literature how effective various feature types and syntactic representations are for prosodic boundary prediction, and how much rich representations contribute to performance. In this project we address these issues with a series of experiments.

1.1 Project aims

The primary goal of this work is to

Compare the effectiveness of different syntactic features and syntactic representations for prosodic boundary prediction.

We design and implement a series of experiments to predict the placement of prosodic phrase boundaries from a selection of syntactic representations. The experiments are based upon previous research, apart from one, which is designed from scratch (see section 4.5). Using the prosodically annotated BURNC corpus we evaluate all the experiments using common measures, which enables a fair comparison between the representations. The experiments are based on machine learning using TiMBL, a machine learning package which is described in section 3.2. The programs for feature extraction have been written by the author of this dissertation. Some of the tools needed to conduct the trials have been written by others, such as parsers and machine learning scripts, as explained in section 3.4.

Chapter 2

Theory and representations

2.1 Brief theory of phrase breaks

The simplest classification of junctures between words is that of a break (i.e. prosodic boundary) versus a non-break. Thus we can imagine a simplified TTS system having a prosody module that makes only two decisions at the junctures between words. This simplified strategy, which we call the *N/B prediction level* in the later experiments, is common in the literature. A finer gradation divides the junctures into three classes rather than just two: non-breaks, and two classes for the breaks (e.g. intermediate and intonational breaks). For example, the corpus utilised in [7] distinguished between "secondary" phrase boundaries, associated with pitch changes, and "primary" boundaries, which were commonly accompanied by a pause.

ToBi (Tones and Break Indices, [19]) is a system for describing aspects of prosody that is frequently employed for intonational annotation[1]. It contains a break tier level which is useful for characterising the structure of an utterance's prosodic phrases, based upon both the pitch curve of an utterance and a listener's subjective opinion. This form of labelling consists of five levels, numbered 0–4. Levels 0 and 1 are by far the most common break levels, signifying words that are closely bound together. Level 2 is assigned to junctures that exhibit some form of contradiction between the pitch pattern and subjective cues like preboundary lengthening that support the subjective sense of boundary strength. Break indices 3 and 4 form a more natural progression from 0 and 1. Index 3 is commonly assigned to junctures that exhibit a relatively weak, but clear break (intermediate phrase break) while 4 signals the strongest breaks (intonational phrase break). In practice, break index 4 occurs at the end of an utterance, between sentences and in the presence of punctuation, while level 3 breaks can occur both with and without punctuation. Usually, level 3 and level 4 junctures are considered phrase breaks, while levels 0–2 denote non-breaks. In the literature a distinction is sometimes made between level 3 and level 4 breaks, while all the other values are classified in the non-break category. We call this prediction level 3/4/N in the experiments. Thus there are two common levels of prediction that a prosodic phrasing module for TTS can employ, N/B and 3/4/N.

2.2 Syntactic phrasing versus prosodic phrasing

Adding more complications to the task of predicting phrase breaks are differences in syntactic and prosodic phrasing. While there is general agreement in the literature that there is some relationship between the two, there is no direct mapping from one to the other, as intonational

[1]See `http://www.ling.ohio-state.edu/research/phonetics/E_ToBI/singer_tobi.html` for annotation guidelines.

phrasing may disregard major syntactic boundaries in order to satisfy constraints on the phrase length ([7]).

[7] lays out a theoretical study of the relationship. First, they present the factors said to contribute to prosodic phrasing. A difference in syntactic category tends to affect phonetic quality, exemplified by the word "live", which is pronounced differently depending upon whether it is used as a verb or an adjective. Knowing which category a word belongs to is usually necessary to produce the correct stress pattern in the sentence. On the phrasing level, [7] notes that misalignments between syntactic and prosodic structure occur frequently. Sentence 1 is an example of this, in which the prosodic boundaries corresponding to the most common phrasing appear as vertical bars in sentence 2[2]:

1. *This is* $[_{NP}$ *the cat that caught* $[_{NP}$ *the rat that stole* $[_{NP}$ *the cheese*]]]
2. *This is the cat* || *that caught* || *the rat that stole* || *the cheese*

Because prosodic structure is normally flatter than syntactic structure, there have been attempts at devising rules to readjust the syntax to better fit the prosodic phrasing. However, [7] claims that such rules are problematic since prosodic structure is heavily influenced by syntactically unrelated issues. Semantics clearly has prosodic effects, but it is usually infeasible to include semantic knowledge in a traditional Text-to-Speech system (although Concept-to-Speech generation may benefit from semantic understanding). Mismatches between syntactic and prosodic structure occur systematically and are often related to non-syntactic factors like prosodic phrase length, which frequently overrides syntactic constituent length to generate a flatter prosodic arrangement. Thus, while syntactic information is certainly an important source of information for the TTS system's prosody generation module, more shallow features are equally useful. It is this apparent dilemma that gives rise to the many different representations that have been employed for predicting prosodic phrase boundaries.

2.3 Predicting prosodic phrasing

There is a great variety in both the syntactic representations used to derive features and the methods for predicting prosodic boundary locations found in the literature. The complexity of representations range from punctuation and Part-of-Speech (POS) [9] via supertags [16] and syntactic chunks [14] to link grammar [17], dependency trees [16] and syntactic constituents [18]. These varying levels of syntactic analysis can be predicted with different degrees of accuracy. Information derived directly from the text, such as punctuation, is easily extracted. POS, the next step up on the ladder of complexity, typically requires a tagger. Taggers nowadays perform with very high accuracy, so POS can be predicted with higher accuracy than chunks, which in turn are both easier and faster to predict than a full syntactic parse. Also, the accuracy of the low-level representation affects the accuracy of the high-level representation since chunking ("shallow parsing") and parsing rely on POS tagging. Thus, the choice of representation for prosodic boundary prediction has consequences for the TTS system's size and speed. The extra time taken to compute a full parse, say, must be traded off with the potential benefits of increased accuracy in the prosodic boundary prediction module. As shown above, the considerable variation in corpora and evaluation methods employed in previous research causes difficulty in determining which representation is best suited for the purposes of TTS generation. Commonly, three different methods are employed with the representations: probabilistic methods [9], machine learning methods [10, 29] and manually written rules [5, 6, 7].

[2]We always consider the end of a sentence to be a phrase break, although it is not marked as such here.

2.3.1 Machine learning methods

In this work, we are primarily concerned with the machine learning approaches, since they allow us to compare a number of different representations more easily than manually written rules, which require a much more time-consuming implementation. In recent years, the machine learning approach has become the most popular. Using such methods, one extracts certain features, which are thought to have a positive effect on the resulting classifier's predictive power, from the text and syntactic representations. Typical features include the identity of punctuation at a juncture and a POS N-gram. One set of features is derived for each juncture. The data is generally divided into a training set and a smaller test set for evaluation. The machine learning approach has the advantage that it is flexible, as new features can easily be added to the existing feature set, allowing for quick experimentation. In this section we review some of the studies into machine learning techniques for prosodic boundary prediction.

Shallow representations

[9] was an influential study that utilised a probabilistic model with a POS N-gram. The first observation they made is that while a model that simply inserts phrase breaks after punctuation is rarely wrong in assignment, it underpredicts and allows overly long phrases in the absence of punctuation. To improve this simple model, the researchers turned to POS information, and focused in particular on the need for estimating the probability of a break using not only the words immediately surrounding each juncture, but also some knowledge of the overall context. This would presumably lead to a classifier that optimises the break assignment across the entire utterance rather than locally. For example, in sentence 1 it is argued that a break between "drive" and "in" would work well, whereas in sentence 2 such an assignment would be unsuitable.

1. *I wanted to go for a drive in the country.*
2. *I wanted to go for a drive in.*

In order to give the classifier some sense of context, the authors employed a Markov model to predict probabilities of POS sequences for utterances, where each state represented either a break or a non-break and the transitions between states represented the likelihood of particular sequences of breaks and non-breaks occurring. Each state had an observation probability distribution giving how likely that state was to have produced a particular sequence of POS tags. The window around each juncture consisted of the two words before and the word after the juncture, and the model represented the sequence

$$P(c_{k-1}, c_k, c_{k+1} | j_k)$$

where j is the current juncture and c_k is the POS tag immediately preceding it. Using Bayes rule, one can then combine the transition probability ($P(j)$) with the emission probability of each state ($P(C|j)$). In order to take more context into account, [9] then used an N-gram of junctures, giving the probability of a juncture type (break or non-break) given the previous N-1 junctures:

$$P(j_k | j_{k-1}, j_{k-2}, \ldots, j_{k-N+1})$$

Training was accomplished straightforwardly by counting occurrences of junctures in all contexts. The MARSEC corpus of spoken British English was used, consisting of a training set of 31707 words and a test set of 7662 words.

Model	**B correct**	**Overall**	**NB incorrect**
Punctuation	54.27%	90.76%	0.85%
C/F	84.40%	71.29%	31.73%
POS N-gram, full tagset (37)	74.22%	90.26%	6.04%
POS N-gram, best tagset (23)	79.27%	91.60%	5.57%

Table 2.1: Results obtained by [9] for the N/B task. The column labelled "Overall" gives the overall accuracy for breaks and non-breaks combined. See section 3.3 for definitions of performance measures.

In [9] two simple baselines were used, one deterministically inserting a phrase break following all punctuation and the other inserting a break after all content words followed by a function word, as used for example by [20]. The results for the N/B task are shown in table 2.1 (the distinction between different types of breaks received little attention). In summary, the punctuation model was conservative, assigning phrase breaks at positions that are almost always correct, but missing many other breaks. The content/function word model predicted many more breaks correctly at the cost of overinsertion. A great increase in performance was achieved using the POS N-gram. The latest version of the Penn Treebank tagset was employed, which consists of 37 tags. By reducing the size of the tagset, the researchers gained good improvement. They also varied the size of the N-gram, but concluded that the number of words in the window is not very important as long as more than two are used.

In [26], a method for optimising the tagset was investigated. It is computationally infeasible to generate all the possible arrangements of POS tags and then evaluate the performance of each, so "best-first search" was employed instead. Best-first search is a classic artificial intelligence technique that aims to find the best possible solution for a specified problem given a particular heuristic. The heuristic is employed to make the choice between several possible solutions. The search algorithm then tries to find the solution that will optimise the heuristic. In [26], best-first search was tested with a number of heuristics including the number of junctures correct, the number of non-breaks correct and the number of breaks correct. The best heuristic, junctures correct, increased the accuracy (junctures correct) by 2% and reduced the false insertion rate by 1%.

Another improvement over the original POS model was suggested in [10]. This study added two new feature types to the POS window, the CFP value and the expanded tag. The CFP value distinguishes between content words, function words and punctuation. The choice between content word or function word is based upon POS tags. The expanded tag takes the value of the word itself if it is a function word and the POS tag otherwise (see section 4.3 and table 4.5). In addition to these two new features, the word itself was also added to the feature set. All the feature types were employed in a window of the same size as the POS window, and the researchers then experimented with different window sizes, feature combinations and machine learning parameters to find the optimal combination. Their best experiment yielded an increase of 2.5% in F-Score compared to [9]. Despite this result however, the authors were not satisfied, concluding that they had not been able to show that adding more complex information sources to the very general POS N-grams improves accuracy. In view of this conclusion, the main contribution of [10] was probably the application of Memory-Based Learning (MBL) to the phrase break prediction task. MBL is a classification-based, supervised learning approach in which the classifier keeps all training data in memory and only abstracts at classification time by extrapolating a class from the most similar item(s) in memory. Refer to section 3.2.1 for more details on MBL.

[29] employed another commonly used machine learning technique, classification and regression trees (CART). This study, conducted well before [9], took inspiration from [4] and [24]. [4] proposed an algorithm for finding prosodic phrasing and pitch accent location, requiring both discourse-level and syntactic information. [24] presented a method utilising a Hidden Markov Model (HMM) for computing break indices from observed relative durations of phonetic segments. Noting that [4] assumes input not available from parsers, and that [24] made use of a very small corpus for training and testing (35 sentence pairs), [29] carried out a series of experiments using both shallow and deep features. The medium-size ATIS database was employed for training and testing[3]. Numerous feature and information types were employed in their study:

- The length of the utterance, both in words and seconds. Utterance length was thought useful since long utterances may tend to contain more boundaries than short utterances.
- The distance of the juncture from the beginning and end of the utterance, measured both in seconds and in words. The intuition behind this feature was that boundary placement may be affected by the position of the potential boundary site in question.
- The distance from the last phrase break location to the current juncture, which is potentially useful since consecutive prosodic phrases often have approximately equal lengths.
- The sentence type, as it was considered possible that certain sentence classes contain more phrase breaks than others. The sentences that were questions were thus divided into three categories: wh-questions, direct yes/no questions and indirect yes/no questions.
- A POS N-gram, which was assumed useful since breaks were believed to be rare after function words. A four-word window was used.
- The class of the lowest node in the parse tree to dominate the word to the left of the phrase break but not dominating the right word, and vice versa. This constituency information was included to test the hypothesis that some constituents may be more likely to be separated by intonational boundaries than others.
- The pitch accent values of the words surrounding the juncture, as phrase breaks were thought to occur less frequently between two deaccented words than between two accented words. Both predicted accent values, obtained by text analysis, and observed accent values, which were retrievable directly from the annotated corpus, were employed.

With different combinations of these feature types, the study proceeded with a series of experiments. The first trial included all the features listed above. The experiment, using prediction level N/B, was first carried out with the observed accent values and then with the predicted values, and the results were very nearly the same for both trials. The overall accuracy was deemed very high: 90% of junctures were correctly classified. One of the advantages of decision trees is that the tree can be manually inspected after training to find which information sources are most predictive. In this way, it was found that the prosodic phrase length was of great importance. Syntactic constituency, which indicated that noun phrases form a tightly bound unit, and accent information were also given great weight by the trained tree. When the actual accent information was replaced by the predicted accent values, the tree compensated well, although the predictions were sometimes incorrect.

In subsequent experiments, the amount of information presented to the machine learner was gradually reduced. First, the phrase length was removed while keeping the observed accent

[3]See `http://wave.ldc.upenn.edu/Catalog/readme_files/atis/sspcrd/corpus.html`

values, and then the predicted accent values were employed rather than the observed ones. This resulted in only a small decrease in performance, to about 89%. Analysing the decision tree, it became apparent that the learner had adapted to give higher weights to other information sources than those originally weighted highest when all the feature types were in use. The conclusion of the study was therefore that there was considerable redundancy in the features used and that the CART technique was well suited to the task since linguistic insight was gained by examining the tree's nodes.

In a similar study ([15]), the development of the prosodic phrasing module in the Bell Laboratories TTS system is described. Again employing decision trees, the feature set from [29] was extended to include syllable information. While this new information did not appear to be crucial for the classifier, the results provided very useful insights. The performance of the decision tree was measured by comparing its break predictions against those of a manually annotated corpus. Inspecting the errors, it was found that only 101 out of 1072 mispredictions actually produced phrasings that were unacceptable to a listener. Most of the apparent mispredictions turned out to be acceptable, and a few were due to mistakes in the human annotation of the corpus. It was also noted in [15] that the majority of the unacceptable phrasings were due to the lack of deep features. Higher level constituency information was lacking, and this led to predictions that were wrong when there was no disambiguating punctuation but the sentence contained syntactic or semantic ambiguities. The obvious solution was to add syntactic information to the feature set.

Deep representations

The work of [15] was therefore extended by adding features derived from a parse tree to the feature set, as described in [18]. These new features included the sizes of the syntactic phrases surrounding the juncture, in addition to boolean features denoting whether or not the phrases were of a major type (NP, VP, PP, ADVP, ADJP) or labelled SBAR. The enlarged feature set improved the F-Score by almost 2%, again using the N/B prediction level.

Following from ideas set out in [1], [14] explored a different syntactic representation, syntactic chunks. The argument is that a syntactic chunk representation is better suited to prosodic boundary prediction because it corresponds more closely to prosodic structure than standard constituency-based grammars. The chunks are tree fragments where "problematic" segments (for example prepositional phrases) are left unattached, as shown in figure 2.1.

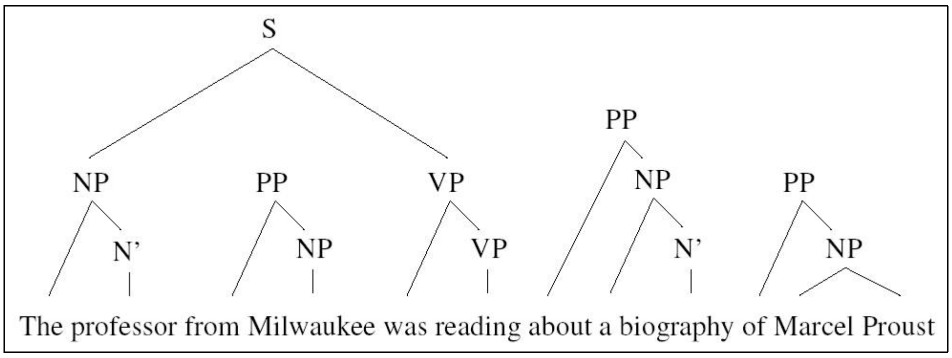

S
PP
NP
PP
VP
NP
PP
N'
NP
VP
N'
NP
The professor from Milwaukee was reading about a biography of Marcel Proust

Figure 2.1: Chunk representation, showing how some phrases are not attached to the parse tree. (figure taken from [3]).

A chunk parser described in [11] was employed, which encoded the chunk tree structure with brackets, for example:

[[*the device*] [*is attached*] [*to a plastic wristband*]]

It can be seen that this structure is close to a possible prosodic structure (with prosodic boundaries between "device" and "is" and between "attached" and "to"), and the study treated the rightmost terminals in each chunk as carriers of phrase breaks. Thus a single boolean feature, true if the word preceding the break is the last in its chunk, was employed to predict phrase breaks, with reasonable results. It was found that performance could be improved by ignoring the one-word chunks, and only treat the final words in chunks containing two words or more as carriers of phrase breaks.

Another alternative to the standard constituent-based syntax, link grammar, was employed in [17]. As shown in figure 4.7 in section 4.6, link grammar is based on the fact that links can be drawn between syntactically related words without the links crossing. The idea underlying the use of link grammar for the purpose of prosodic phrase break prediction is the similarity between link grammar and break indices. A break index reflects the prosodic coupling between words, while link grammar structures reflect the explicit syntactic coupling between words. Assuming that there is a correspondence between these two types of coupling, [17] derived eight features (described in section 4.6) that were extracted from link grammar parses and utilised by a machine learner for predicting phrase breaks. While the results reported by [17] were encouraging, a weakness of this study is that training and testing was done on a small corpus of just 100 sentences. Furthermore, manual intervention was needed to select the correct parse among the possibilities returned by the parser, which suggests that the method may not be suitable for automation, at least with the parser in question[4]. More details on the different deep representations mentioned in this chapter are given in chapter 4.

2.3.2 Rule-based methods

In contrast to machine learning methods, rule-based approaches to prosodic boundary prediction utilise linguistic knowledge and require no training data [5, 6, 7]. The disadvantage of such methods is that they are difficult to maintain and implement, lacking the flexibility of the machine learning algorithms. While we are only concerned with machine learning approaches in this dissertation, it is useful to briefly address one of the rule mechanisms that have been tried in the past.

The authors in [7] has followed up on the theoretical analysis of prosodic phrasing with a set of rules for predicting such phrases. Two types of rules are presented, one for predicting the location of breaks and one focusing on the relative break strength. The location rules first form phonological words from the lexical items in a parse tree and subsequently group the phonological words into phrases. The resulting phonological phrases consist of elements that cohere so strongly that they cannot be broken up with phrase breaks without changing the semantics of the utterance, so phrase breaks can then be placed between the phonological phrases. The phonological phrases are then combined with the parse tree, and the resulting structure is processed by the boundary salience rules, which merge some of the phonological phrases into larger ones (intonational phrases), creating a final phrase hierarchy. In this way, breaks between phrases are diminished and strengthened such as to create a rhythm of phrases of roughly equal lengths. The evaluation of this method was weak, with a corpus of only 35 sentences, so it is difficult to determine how effective the method is.

[4]See `http://www.link.cs.cmu.edu/link/`

Chapter 3

Experimental setup

To achieve the goal of comparing syntactic representations for their effectiveness for prosodic boundary prediction, we reimplement in experiments some of the approaches described in chapter 2 within a single machine learning framework. In order to enable comparison among the results of the experiments, we use a single corpus with the same machine learning software and evaluation methods throughout. This setup is described in detail in the following sections.

3.1 The corpus

In the experiments we employ the Boston University Radio News Corpus (BURNC) [23], which is based on speech from seven radio news announcers. The radio newscasting style is regarded as particularly appropriate for Text-to-Speech applications and prosody modelling for several reasons. The professional radio announcers tend to use clear and consistent indications of prosody, which facilitate modelling and analysis of the style. Also, the task facing the newsreaders is similar to that of a TTS system in that text should be read out in a pleasant and prosodically correct way by choosing a prosodic pattern that reduces the risk of ambiguity.

The BURNC is divided up into two portions, *radio news* and a smaller *lab news* portion. The radio news was recorded in the radio studio during broadcast, while the lab news part was recorded in a laboratory. Since some of the radio news data contain background noise, it is of inferior signal quality to the lab news, but as we do not work directly with the speech signal itself, this is unlikely to have a drastic effect on our results.

3.1.1 Annotation

The two news portions consist of radio news stories forming paragraphs of a few sentences per utterance. In addition to the actual speech, the corpus database contains orthographic transcriptions, POS tags, prosodic labels and phonetic alignments for each utterance, stored in separate files. The orthographic transcriptions were generated by hand, as were the prosodic labels. A tagger was employed to automatically tag the corpus, with the oldest version of the Penn Treebank tagset. For the lab news data, the POS tags were hand-corrected, and the tagger's error rate was found to be 2%. A speech recogniser provided phonetic alignments, and these were employed to assist in prosodic annotation.

The ToBi labelling system was used to annotate the utterances prosodically. In addition to the break index tier described in section 2.1, ToBi also consists of a tone tier for labelling accents and phrase tones. We only utilised the phrase break annotation, however. Unfortunately, using the corpus was not as straightforward as hoped.

3.1.2 Inconsistencies

The BURNC has been widely used since its inception about ten years ago, and it was therefore somewhat surprising to find a number of inconsistencies in the corpus. As a result, some of the data files had to be excluded from the training and test set.

The inconsistencies are made up of quite subtle errors. For our purposes, only the phrase break files, POS files and phonetic alignment files were required, all of which have a single entry per word. Words such as "school-based" are sometimes split and sometimes not, but unfortunately there are cases in which one of the file types contains a split and the others do not, leading to misalignments. In addition, there are examples of phrase break files which have extra entries that cannot correspond to any of the words in the text. Many of the problems were quite easy to repair, and we manually edited some of the files to make them usable. However, a large number of files had to be excluded from use because they suffered from errors that were difficult to correct. We were unable to find any documentation describing these problems, and the inconsistencies led to a smaller training and test set than would otherwise have been employed.

3.1.3 Training and test data

We divided the usable data in the BURNC such that a little over 90% of the words were used for training and a little less than 10% for testing, with 24462 words in the training set and 1895 in the test set. For the purposes of testing, the speech from two speakers, one male and one female (respectively labelled "m3b" and "f3a") was employed, while for training all but speaker "m3b" were included. The choice of speakers for the training and test set was made to ensure that no single speaker should dominate either set, so that the quality of the training and test data should be roughly similar. The training and test files are listed in appendix A. This split was used for training and testing in all of the experiments.

3.2 TiMBL

Two machine learning packages were considered in this work: the decision tree learner C4.5[1] and the memory-based learner TiMBL[2] (Tilburg Memory-Based Learner) [13]. Both have been successfully employed for prosodic phrase break prediction, although decision trees are more frequently used.

3.2.1 Memory-based learning versus decision trees

In memory-based learning, all training data is kept in memory and abstraction is only performed at classification time by extrapolating a class from the most similar items. This is in contrast to decision trees, where rules are extracted from training data and subsequently used for classification. The proponents of memory-based learning argue that the lack of abstraction is an advantage for natural language processing tasks, since the approach remembers exceptional, low-frequency items which are useful to extrapolate from. Decision trees, they claim, forget such useful knowledge because of their frequency-based abstraction-methods and pruning. In [10] and [22], TiMBL was found to perform well for predicting phrase breaks. Nevertheless, decision trees can be manually inspected after training, which is very useful for understanding which information types provide discrimination.

[1]See `http://www2.cs.uregina.ca/~hamilton/courses/831/notes/ml/dtrees/c4.5/tutorial.html`
[2]See `http://pi0657.kub.nl/software.html`

In the end, TiMBL was chosen for practical reasons. One of the supervisors Sabine Buchholz, is a TiMBL expert, and we benefited greatly from consultation with her. We employed the memory-based learner throughout, using the IB1 metric. IB1 classifies on the basis of the distance between a test item and each memory item (neighbour). In TiMBL this measure is implemented by summing the differences between the feature values. Having chosen the algorithm, there remained several settings that could be set to optimise the classifier's performance, such as how neighbours should be weighted and how many neighbours should be used in measuring the distance. There are a great number of potential settings that could be used for any given prediction task, but fortunately an algorithm for finding the best learner settings was made available by the TiMBL development team.

3.2.2 Iterative deepening

To improve the experimental performance, iterative deepening was applied. This process optimises the parameters of the learner given an algorithm (IB1 in our case) and training data, beginning with a large pool of parameter settings. The first step applies each setting to a small amount of training data and tests it on a held-out portion of the training data. Only the best settings are kept, and the others discarded in the next round. This is repeated in a loop, exponentially reducing the number of settings while exponentially increasing the size of the training material, such that the time taken for each step is kept almost constant. The process terminates when only the single best setting is left, or the best score is attained by several settings, in which case a random parameter combination is chosen among the best ones. Iterative deepening was applied in all our experiments.

3.3 Evaluation

To assess the performance of the machine learning methods, appropriate evaluation measures were required. Other than precision and recall, common measures of performance for prosodic phrase break prediction are percentage of junctures correct (accuracy)[3], non-breaks correct and breaks correct. Some authors give results where breaks and non-breaks are grouped together (e.g. [29]). Grouping predicted breaks and non-breaks together artificially increases the performance measures since non-breaks are much more frequent than breaks and easier to predict. For example, in our data 72.6% of the junctures were non-breaks, so a classifier which predicts non-breaks everywhere would yield a 72.6% accuracy.

The variation in evaluation methods causes confusion and makes results difficult to compare [10]. In this study, we therefore employ three well-known, accepted evaluation measures: precision (P), recall (R) and F-Score. We use the following definitions:

$$\text{Precision} = \frac{\text{Number of breaks correctly predicted by classifier}}{\text{Number of breaks predicted by classifier}} \tag{3.1}$$

$$\text{Recall} = \frac{\text{Number of breaks correctly predicted by classifier}}{\text{Total number of breaks in corpus}} \tag{3.2}$$

$$\text{F-Score} = \frac{2 \cdot P \cdot R}{P + R} \tag{3.3}$$

[3]Note that junctures correct $= \frac{\text{Number of breaks and non-breaks predicted correctly by the classifier}}{\text{Total number of breaks and non-breaks in corpus}}$

Since non-breaks are far more common than breaks, the task of predicting non-breaks is relatively easy. We are therefore not concerned with measuring performance on the predicted non-breaks, and all results pertain to the prediction of phrase breaks.

3.4 Implementation

This dissertation focuses on comparing syntactic representations for prosodic phrase break prediction, rather than production of software. Therefore, we do not discuss the details of the implementation. Nevertheless, a significant programming effort has been invested. The major part of the implementation has involved writing scripts and programs for extracting features from the BURNC files and the output of parsers. In total, roughly 1000 lines of shell scripts, 1500 lines of Java and 9000 lines of C++ were written, and are included in the separate code appendix.

While the actual feature extraction programs were written by the author, we employed several programs written by others. The TiMBL package, the script for iterative deepening and the MBSP chunker were all created at Tilburg University. In addition, Collins' parser ([12]) and Dan Bikel's parser ([8]) were employed for the experiments that required deep features, as well as a script written by Sabine Buchholz for converting the output of Collins' parser to dependency grammar. The lines of Java referred to above pertain to additions made by the author of this dissertation to Dan Bikel's parser.

Chapter 4

Machine learning experiments

To compare the various approaches discussed in the previous sections, we carried out experiments with six different representations. For the first set of trials, only punctuation information was employed. The complexity of feature extraction and syntactic representation was then increased steadily via experiments using shallow features such as content/function word distinctions and ultimately using features derived from a full parse tree.

All the experiments were carried out using the same training and test data sets (except for very minor changes, see section 4.4 and 4.6). In addition, the same machine learner, TiMBL, was used throughout, with iterative deepening. Each of the representations was used for both the N/B and the 3/4/N prediction tasks. The six sets of experiments are described in detail in the following sections.

4.1 Punctuation

4.1.1 Feature set and results

The most basic representation we can use to predict prosodic phrasing is punctuation information. When speaking, we naturally pause at the end of sentences, and also often do so within sentences. Since punctuation is used by writers to indicate rhythm and pausing, we intuitively expect features derived from punctuation to provide strong clues for a machine learner on where to place phrase breaks. This experiment could easily have been carried out employing a program that deterministically inserts breaks at every punctuation mark. However, in order to compare the results to the other trials, the punctuation experiment was executed using TiMBL.

This experiment uses only a single feature for each juncture, the punctuation mark itself, as shown in figure 4.1. We defined the characters ".", ",", ":", ";", "!", """, "?" to be punctuation. The performance results for this experiment are given in tables 4.1 and 4.2.

Experiment	**Precision**	**Recall**	**F-Score**	**Prediction level**
Punctuation	0.969	0.406	0.572	N/B
Punctuation	0.934	0.391	0.551	3/4/N

Table 4.1: Results for predicting prosodic phrase breaks using punctuation only. The measures are for breaks only and do not consider non-breaks.

Sentence	But Jay Ash, Deputy Superintendent of the Hampton County Jail in Springfield says the surveillance system is not that sinister.	
Corpus	**Text**	**Break value**
	Ash,	4
	Deputy	3-
	Superintendent	1
	of	2
	the	1
Representation	**Feature value**	**Break value**
	\,	4
	=	3
	=	N
	=	N
	=	N

Figure 4.1: Punctuation features for the TiMBL machine learning software. "=" denotes a word that has no punctuation, while the "\" is necessary to escape TiMBL control characters. Note how the diacritic on the break value for "Deputy" is ignored. The "Corpus" entry shows how the text is represented in the BURNC files. This example is for the 3/4/N prediction level.

Experiment	Precision 3	Precision 4	Recall 3	Recall 4
Punctuation	-	0.934	-	0.598

Table 4.2: The results of the 3/4/N experiments using punctuation only. "-" means that no breaks of level 3 were predicted.

Experiment	**Precision**	**Recall**	**F-Score**	**Prediction level**
Punctuation	0.969	0.406	0.572	N/B
POS	0.695	0.761	0.726	N/B
Punctuation	0.934	0.391	0.551	3/4/N
POS	0.650	0.537	0.588	3/4/N

Table 4.3: The results of the POS experiments. Punctuation results are also given for comparison.

4.1.2 Analysis

We note the high precision and low recall, leading to a low F-Score. Since only a single feature is used for each juncture, the poor results shown in tables 4.1 and 4.2 are to be expected. The classifier predicts a break everywhere it sees a punctuation mark, since in the training data almost every juncture preceded by punctuation is marked as being a break. This conservative strategy leads to underprediction of breaks. The breaks that are not predicted by the classifier occur in contexts with no punctuation. In the training data there are overwhelmingly many more instances of non-breaks in such contexts, so the classifier sets all junctures not preceded by a punctuation mark to be non-breaks (72.6% of all junctures in the corpus are non-breaks). In practice, such underprediction means that the classifier allows overly long phrases when there is no punctuation in the text.

During training, almost all of the level 3 breaks observed by the machine learner are associated with junctures that have no punctuation. But nearly all of the punctuation-free junctures are non-breaks, so the classifier decides that all of the junctures without punctuation are non-breaks. Thus, as expected, predicting three classes rather than just break/non-break is a harder task.

4.2 Part-of-Speech

4.2.1 Feature set and results

POS information is another readily available source of features, since the text in the BURNC corpus is tagged. However, even if we employed untagged data, many fast and accurate POS taggers exist. Another advantage of POS as a feature source is that both tagging and feature extraction is fast. Thus, following the ideas in [9], we implemented a machine learning experiment using a POS trigram as the only information source. The trigram was of the familiar form

$$POS_{n-2}, POS_{n-1}, POS_{n+1}$$

where POS_{n-2} and POS_{n-1} are Parts-of-Speech for the two words preceding the juncture and POS_{n+1} is the Part-of-Speech for the following word. Punctuation is included explicitly in the POS trigram, as shown in figure 4.2. The results of the two POS experiments are shown in tables 4.3 and 4.4.

4.2.2 Analysis

When using POS trigrams as training data, the F-Score and recall are clearly better than when only punctuation is employed. The improvements displayed in tables 4.3 and 4.4 are to be expected, since the classifier is now trained with three times as many features. For both

Sentence	But Jay Ash, Deputy Superintendent of the Hampton County Jail in Springfield says the surveillance system is not that sinister.			
Corpus	**Word**	**Tag**	**Break value**	
	Ash,	NP	4	
	Deputy	NP	3-	
	Superintendent	NP	1	
	of	IN	2	
	the	DT	1	
Representation	POS_{n-2}	POS_{n-1}	POS_{n+1}	**Break value**
	\,	NP	NP	B
	NP	NP	NP	B
	NP	NP	IN	N
	NP	IN	DT	N
	IN	DT	NP	N

Figure 4.2: Example POS trigram for input to TiMBL. Since "Ash" has a comma after it, POS_{n-2} becomes the comma rather than the POS tag of the previous word. The POS_{n+1} value for "the" comes from the tag of the following word "Hampton". This example illustrates the break/non-break prediction level.

Experiment	**Precision 3**	**Precision 4**	**Recall 3**	**Recall 4**
Punctuation	-	0.934	-	0.598
POS	-	0.650	-	0.821

Table 4.4: The results of the 3/4/N POS experiments.

prediction levels, precision is lower than for the previous experiments, but recall is much higher, yielding a better F-Score.

As opposed to the previous experiment with breaks/non-breaks, there is now more over-prediction, since precision has decreased. With three times as many features per record fed to the machine learner, many more breaks can easily be distinguished from the non-break cases in the training data, and this results in many more breaks being predicted. Consequently, recall is much higher than for the punctuation-based experiment, but precision falls. Punctuation is still, however, a very important feature in predicting breaks. Manual inspection of the classifier's output shows that every instance of sentence-final punctuation (".") is predicted to be a break.

There are still problems when predicting break levels 3 and 4 rather than just grouping these two breaks into one class. Despite tripling the number of features, none of the breaks are classified as 3. This is essentially due to the same reasons as noted in the punctuation-based experiment. Most of the breaks in the training data are associated with punctuation. Of these, almost all the breaks are level 4, so the classifier decides that all junctures with punctuation should be breaks of level 4. In contrast, the "softer" breaks of level 3 often occur in circumstances where there is no punctuation. As mentioned in section 4.1.2, there is such an overwhelming majority of non-breaks in these contexts that the trained classifier predicts non-breaks in every such instance.

No attempt was made to optimise the tagset. All 37 tags in the BURNC corpus were used, which is most likely not ideal [9], [26]. For consistency, the same tagset was employed in all the other experiments that utilised POS features. Although we did not investigate the effects of varying the window and tagset size, it is obvious that there is a limit to how well a model based on Parts-of-Speech alone can perform. This shallow model can be extended in two ways, either by adding further shallow features such as content/function word distinctions or taking a deeper approach using a parser. Both approaches were implemented, as described in the following sections.

4.3 CXPOS

4.3.1 Feature set and results

The next set of experiments utilised two additional types of shallow information in addition to Parts-of-Speech. Based on [10], we augmented the POS trigram with a *CFP trigram* and an *expanded tag trigram*. The CFP-value distinguishes between content words (C), function words (F) and punctuation (P), taking on one of the three values per word. Table 4.5 lists the POS tags that together make up the F-tag. The expanded tag trigram contains the word itself if it is a function word and the POS tag otherwise.

The idea behind adding these features is to change the granularity of the POS tags. The CFP value brings down the number of tags to three, while the expanded tag removes all open class words, leaving behind a fixed set of POS tags and closed-class words as values. An example representation is shown in figure 4.3. The results of the two CXPOS experiments are shown in table 4.6 and in table 4.7.

POS tag	**Function**
CC	Conjunction
DT	Determiner
EX	Existential “there”
IN	Preposition
MD	Modal
POS	Possessive
RP	Particle
TO	“to”
WDT	Wh-determiner
WP	Wh-pronoun
WRB	Wh-adverb
CD	Cardinal number
PP	Personal pronoun
PP$	Possessive pronoun
UH	Interjection
WP$	Possessive wh-pronoun

Table 4.5: The POS-tags making up the function words (F). This table is based on [10].

Sentence	But Jay Ash, Deputy Superintendent of the Hampton County Jail in Springfield says the surveillance system is not that sinister.

Corpus

Word	**Tag**	**Break value**
Ash,	NP	4
Deputy	NP	3-
Superintendent	NP	1
of	IN	2
the	DT	1

Representation

CFP			**XTAG**			**POS**			**Break value**
C	P	C	\,	NP	NP	\,	NP	NP	B
P	C	C	NP	NP	NP	NP	NP	NP	B
C	C	F	NP	NP	of	NP	NP	IN	N
C	F	F	NP	of	the	NP	IN	DT	N
F	F	C	of	the	NP	IN	DT	NP	N

Figure 4.3: Example CXPOS representation, as presented to the TiMBL software. The expanded tag also includes the identity of punctuation, in the same way as the POS trigram.

4.3.2 Analysis

It is clear that the CXPOS representation is superior to using punctuation or POS alone. There is a large increase in all performance measures. The most significant improvement is that we are now able to predict level 3 phrase breaks, although the precision and recall shown in table 4.7 is very low for this type of prosodic break. Of course, the improvement is not unexpected, since

Experiment	**Precision**	**Recall**	**F-Score**	**Prediction level**
Punctuation	0.969	0.406	0.572	N/B
POS	0.695	0.761	0.726	N/B
CXPOS	0.747	0.804	0.774	N/B
Punctuation	0.934	0.391	0.551	3/4/N
POS	0.650	0.537	0.588	3/4/N
CXPOS	0.631	0.621	0.626	3/4/N

Table 4.6: The results of the CXPOS experiments. Punctuation and POS results are also given for comparison. XTAG stands for expanded tag.

Experiment	**Precision 3**	**Precision 4**	**Recall 3**	**Recall 4**
Punctuation	-	0.934	-	0.598
POS	-	0.650	-	0.821
CXPOS	0.307	0.694	0.142	0.873

Table 4.7: The results of the 3/4/N CXPOS experiments.

we are now using nine features rather than just one or three as in the preceding experiments.

For the break/non-break experiment, the analysis is largely the same as for the corresponding trials with POS trigrams. There is now more information available. Punctuation is still an important clue for the classifier, particularly since the expanded tag also contains punctuation information. The extra features give rise to roughly 4% increase in all three performance measures.

It is only at this stage that the trained classifier is able to predict level 3 breaks. There are now so many features that the level 3 breaks can be distinguished from non-breaks, although with very low recall and precision. A close inspection of the classifier's output reveals that the feature most important for predicting the level 3 breaks is the expanded tag. As mentioned, the expanded tag is the word itself if it is a function word, and the POS tag otherwise. This feature is alone responsible for prediction of almost all the level 3 breaks. For the N/B-experiment, it appears that both new features are equally effective in generating improvement over the POS experiment. Using both new features together gives an extra boost to the results.

The CFP and expanded tag features that were added to the POS trigram are lexical. The alternative, which may seem more promising, is to add deeper syntactic features instead, in order to complement the POS features which themselves are properties of individual words. This route was explored in the rest of the experiments.

4.4 Parsing

4.4.1 Feature set and results

In order to explore how close the correspondence between standard constituent-based syntax structures and prosodic structure is, we first carried out an experiment inspired by [18], which reports improvement when adding information derived from parse trees to shallow features. Our linguistic intuition is that prosodic phrase breaks are more likely to occur between large syntactic phrases and after major phrases (NP, VP, PP, ADVP, ADJP). Augmenting Dan Bikel's

Experiment	**Precision**	**Recall**	**F-Score**	**Prediction level**
Punctuation	0.969	0.406	0.572	N/B
POS	0.695	0.761	0.726	N/B
CXPOS	0.747	0.804	0.774	N/B
Parser without POS	0.725	0.748	0.737	N/B
Parser and POS	0.717	0.774	0.745	N/B
Punctuation	0.934	0.391	0.551	3/4/N
POS	0.650	0.537	0.588	3/4/N
CXPOS	0.631	0.621	0.626	3/4/N
Parser without POS	0.757	0.499	0.602	3/4/N
Parser and POS	0.688	0.593	0.637	3/4/N

Table 4.8: The results of the experiments with Dan Bikel's parser, in addition to previous results. "Parser without POS" and "Parser and POS" denotes that the four parser features were employed without and with a POS trigram.

Experiment	**Precision 3**	**Precision 4**	**Recall 3**	**Recall 4**
Punctuation	-	0.934	-	0.598
POS	-	0.650	-	0.821
CXPOS	0.307	0.694	0.142	0.873
Parser without POS	-	0.757	-	0.762
Parser and POS	0.333	0.739	0.105	0.850

Table 4.9: The results of the 3/4/N parser experiments.

probabilistic parser [8][1] with functions and classes for extracting features from the parse tree, we implemented experiments using the following features:

F1 A boolean feature indicating whether or not the largest phrase ending with the current word is a major phrase.

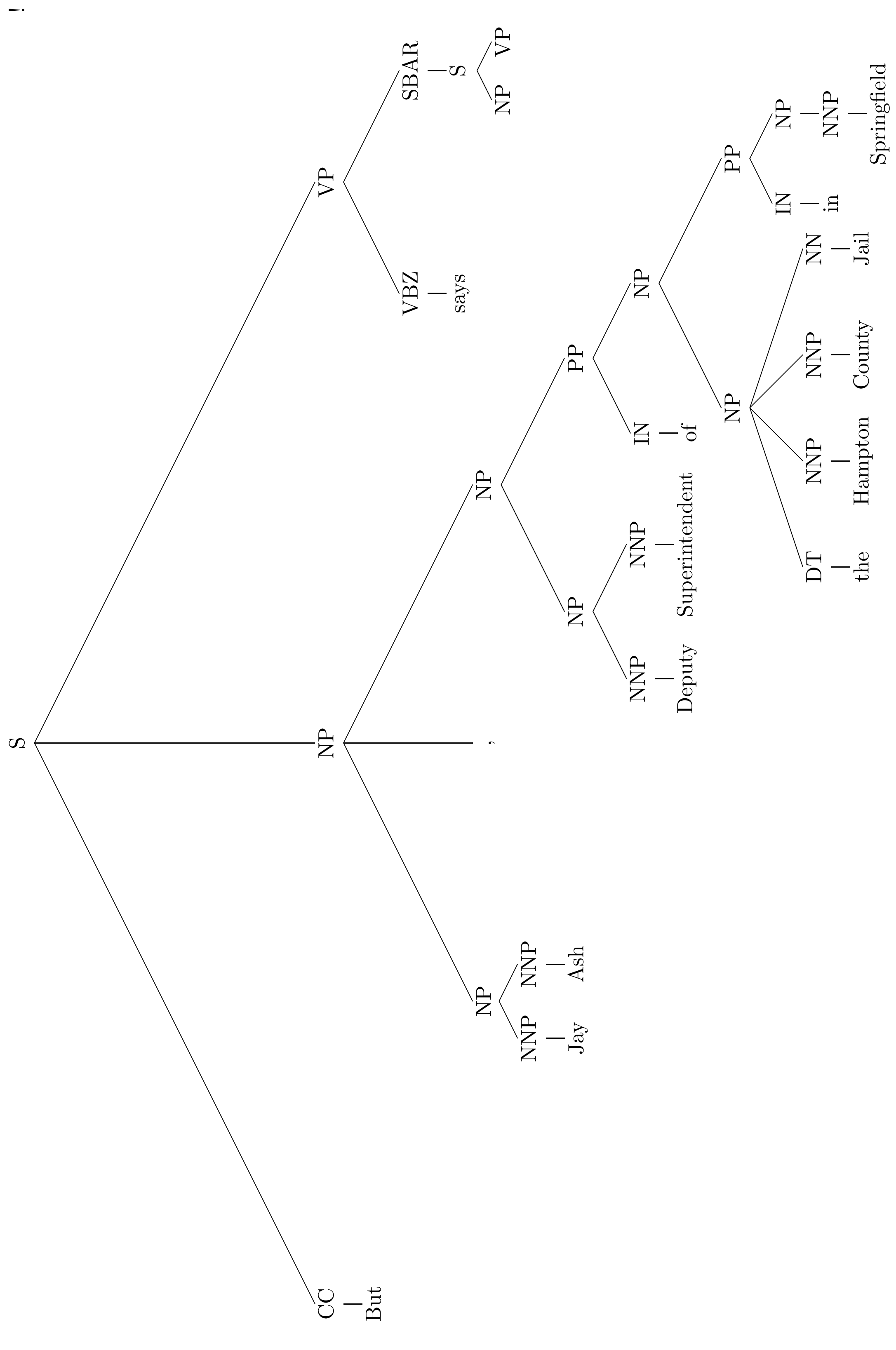
F2 The size of the largest phrase ending with this word.

F3 A binary flag indicating if the next phrase is an SBAR.

F4 The size of the next phrase which is on the same level in the parse tree as the largest phrase ending with the current word.

If the following phrase is just one word, it is collapsed with the next phrase with regard to the last feature. The final feature was added since intonational phrase breaks frequently occur before sub-clauses. The parse tree for the example sentence is displayed in figure 4.4 and overleaf, and example feature values are shown in figure 4.5. In addition to these features, we also added a POS trigram (as in the previous experiments) in another set of experiments. Very minor changes to the training and test set were required, since a few of the sentences did not parse. One sentence was removed from the corpus, and five sentences needed trivial changes to the POS tags in order to parse. The results are shown in tables 4.8 and 4.9.

[1]See `http://www.cis.upenn.edu/~dbikel/`


S
CC
But
NP
NP
NNP
Jay
NNP
Ash
,
NP
NP
NNP
Deputy
NNP
Superintendent
PP
IN
of
NP
NP
DT
the
NNP
Hampton
NNP
County
NN
Jail
PP
IN
in
NP
NNP
Springfield
VP
VBZ
says
SBAR
S
NP
VP
.

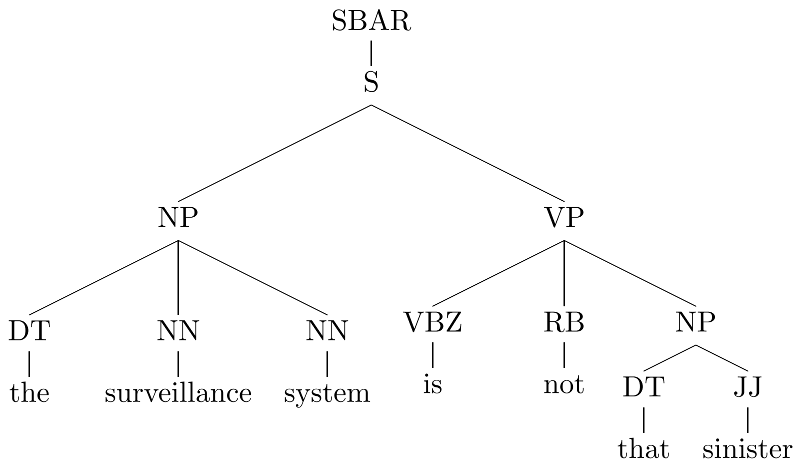


Figure 4.4: The SBAR branch of the example parse tree.

Sentence	But Jay Ash, Deputy Superintendent of the Hampton County Jail in Springfield says the surveillance system is not that sinister.				
Corpus	**Text**	**Break value**			
	Ash,	4			
	Deputy	3-			
	Superintendent	1			
	of	2			
	the	1			
Representation	**F1**	**F2**	**F3**	**F4**	**Break value**
	Y	2	N	9	B
	N	1	N	2	B
	Y	2	N	7	N
	N	1	N	6	N
	N	1	N	2	N

Figure 4.5: Example representation using Dan Bikel's parser. The parser uses the latest version of the Penn Treebank tagset, and since the BURNC is tagged with the old tags, we had to map from the old to the new tagset manually.

Experiment	Feature	Precision	Recall	F-Score
Parser and POS	1	0.633	0.552	0.590
Parser and POS	2	0.687	0.565	0.620
Parser and POS	3	0.638	0.561	0.597
Parser and POS	4	0.656	0.542	0.594

Table 4.10: Using only one of the syntactic features in addition to the POS window for the 3/4/N experiments.

4.4.2 Analysis

Tables 4.8 and 4.9 show that only one of the experiments, "Parser and POS" on the 3/4/N task, improves upon the results from the CXPOS experiment. That "Parser and POS" experiment also trades off recall for precision, while recall and precision using the CXPOS representation are better balanced. It is somewhat surprising that this is the only experiment that yields improvement over CXPOS, since we are now using a combination of shallow and deep features rather than only shallow features.

Without the POS trigrams, the "coarse" break/non-break experiment yields a slightly better F-Score than when using POS on its own. When predicting level three and level four phrase breaks in addition to non-breaks, the parser-derived features also improve upon the F-Score of the POS trigram. Since the difference between recall and precision is large, with relatively high precision and low recall, the parser derived features are more conservative than POS alone. This suggests that the syntactic information could work as a replacement for POS trigrams. However, due to the possible trade-offs in memory requirements and computational cost, such a substitution may not be practical.

Comparing the results of the best parser experiment against the CXPOS experiment, it is evident that the CXPOS approach performs better at predicting level 3 breaks, but worse when it comes to predicting level 4 breaks. It is difficult to determine why this is the case, but the difference is so small that it may not be significant, as the absolute number of level 3 predictions is low (about 70).

Tables 4.10 and 4.11 show the results when using only one of the syntactic features at a time, in addition to the POS window. Feature 2 is the most effective at predicting level 4 breaks, and since these breaks are the most numerous, the overall result is improved. Features 3 and 4 are the only ones that enable the classifier to predict level 3 breaks, but are still not very effective on their own. The number of predicted level 3 breaks is much higher when all features are used together.

In summary, the idea that phrase breaks most often occur between large syntactic phrases and after major phrases is confirmed, as the syntactic representation used in this experiment outperforms the POS trigram on its own. However, the improvement over the simpler experiments is not great, which was also observed by [18]. It is somewhat surprising that the parse tree features are only superior to those used in the CXPOS experiments for prediction level 3/4/N with POS. A possible explanation is that there were more features used in the CXPOS experiments, which may have outweighed any advantages that the parse tree features have in the case of the break/non-break experiments. In addition, punctuation information was included in two of the features in the CXPOS experiments. The next experiments employ a larger number of deep features, with different syntactic representations, and thus give us an opportunity to decide whether deep feature types, with more information per data point, improve the prediction of phrase breaks.

Experiment	Feature	Precision 3	Precision 4	Recall 3	Recall 4
Parser and POS	1	-	0.636	-	0.8436
Parser and POS	2	-	0.687	-	0.863
Parser and POS	3	0.25	0.654	0.025	0.844
Parser and POS	4	0.125	0.679	0.012	0.821

Table 4.11: Using only one of the syntactic features in addition to the POS window. Feature 1 is a boolean value indicating whether or not the phrase ending with the word is a major one (NP, VP, ADJP, ADVP, PP). Feature 2 is the number of words in the biggest phrase ending with the word. Feature 3 is a boolean value indicating whether or not the next phrase has the label SBAR. Feature 4 is the number of words in the next phrase.

4.5 Syntactic chunks

4.5.1 Feature set and results

Shallow parsing, or chunking, is a simpler syntactic representation than a full parse tree. When reading, we naturally split the text into chunks. Superficially, the chunks are similar to prosodic phrases, since boundaries usually fall between chunks [3]. Thus an advantage of the chunk representation, is that it might correspond more directly to prosodic phrasing than the other syntactic representations investigated here. From a practical perspective, an additional benefit of syntactic chunks is that they require less computation time than a full parse. Despite these advantages, the chunk representation has not been extensively employed for predicting prosodic phrasing. The author of [5] used chunks to obtain ϕ-phrases, a kind of phonological phrase categorised as a low-level prosodic phrase. [22] utilised the identity of chunks together with other, non-chunk related features, in experiments on prosodic boundaries and pitch accents in Dutch. Taking inspiration from some of these earlier studies, we designed experiments to ascertain the effectiveness of features derived from syntactic chunks.

For this purpose, we used MBSP (Memory-Based Shallow Parser)[2]. The MBSP chunker does not return a full parse, but rather identifies phrases (NP, VP and so on). The output is a partial parse, an example of which is depicted in figure 4.6. The following features were extracted:

F1 The identity of the chunk holding each word (NP, PP ...), extracted in a trigram window in order to take account of context. For words outside of chunks, the value "O" is used.

F2 Distance to end of sentence, measured in words. This feature is not specific to the syntactic chunks representation, and was only employed in the experiments labelled "FullChunks" and "FullChunks and POS".

F3 Length of the current chunk. For words outside chunks this feature is 0.

F4 Length of the previous chunk. If the word before the start of the current chunk is outside all chunks, this feature is 0.

F5 Boolean feature indicating whether or not a word is followed by punctuation. This feature is not specific to the syntactic chunks representation, and was only employed in the experiments labelled "FullChunks" and "FullChunks and POS".

F6 Boolean feature indicating whether or not a word is the last in its chunk.

[2]See `http://ilk.kub.nl/cgi-bin/tstchunk/demo.pl`

F7 A symbolic feature, extracted in a trigram window in order to take context into account:

B - The word marks the beginning of a chunk.

E - The word marks the end of a chunk.

I - The word is inside a chunk, but neither first nor last.

O - The word is outside all chunks.

The first round of experiments was completed with a version of MBSP that accepts POS tags from an external tagger, so that we could use the BURNC tags. However, we discovered subtle errors in the chunker's output, when compared to a recent version of MBSP that uses its own internal tagger. For example, some of the chunks would be very long and obviously erroneous. This might be because the version of MBSP that accepts tagged input is old and not well trained. The chunked data from this version was not adequate for our purposes, and we used the current MBSP implementation, which performs its own POS tagging. We were therefore not able to employ the POS tags from the BURNC for this experiment, as used in the other experiments, so we cannot directly compare the results of the chunk-based experiments with the others, without bearing in mind that their performance might be affected by the accuracy of the MBSP tagger. Nevertheless, if we assume that the accuracy of the MBSP tagger is high, the results should still be good enough to let us conclude if the chunk representation is well suited to prosodic phrase prediction or not.

Sentence	But Jay Ash, Deputy Superintendent of the Hampton County Jail in Springfield says the surveillance system is not that sinister.											
Corpus	**Word**	**Tag**	**Break value**									
	Ash,	NP	4									
	Deputy	NP	3-									
	Superintendent	NP	1									
	of	IN	2									
	the	DT	1									
Chunks	But/CC [NP Jay/NNP Ash//NNP NP] ,/, [NP Deputy/NNP Superin-tendent//JJ NP] {PNP [PP of/IN PP] [NP the/DT Hampton//NNP County/NNP jail/NN NP] PNP} {PNP [PP in/IN PP] [NP Spring-field/NNP NP] PNP} ,/, [VP says/VBZ VP] [NP the/DT surveillance/NN system/NN NP] [VP is/VBZ VP] not/RB [ADJP that/RB sinister/JJ ADJP] ./.											
Representation	**F1 trigram**			**F2**	**F3**	**F4**	**F5**	**F6**	**F7 trigram**			**Break value**
	NP	NP	NP	17	2	0	Y	Y	B	E	B	B
	NP	NP	NP	16	2	2	N	N	E	B	E	B
	NP	NP	PP	15	2	2	N	N	B	E	B	N
	NP	PP	NP	14	4	2	N	Y	E	B	B	N
	PP	NP	NP	13	4	2	N	Y	B	B	I	N

Figure 4.6: Example showing chunks as produced by MBSP and features presented to TiMBL. The chunks do not overlap. The chunk label "PNP" groups together a preposition and one or more NPs that together form a prepositional chunk. The PNP label was ignored in our experiments, since it sometimes creates overly long phrases and is therefore unlikely to be useful for prosodic boundary prediction. The chunks inside PNPs were treated as ordinary PPs and NPs on their own.

Experiment	Precision	Recall	F-Score	Prediction level
Punctuation	0.969	0.406	0.572	N/B
POS	0.695	0.761	0.726	N/B
CXPOS	0.747	0.804	0.774	N/B
Parser without POS	0.737	0.680	0.707	N/B
Parser and POS	0.702	0.785	0.741	N/B
ChunksWindow	0.653	0.418	0.510	N/B
ChunksWindowPOS	0.706	0.778	0.740	N/B
AdvChunksWin	0.629	0.708	0.666	N/B
AdvChunksWinPOS	0.710	0.787	0.746	N/B
Punctuation	0.934	0.391	0.551	3/4/N
POS	0.650	0.537	0.588	3/4/N
CXPOS	0.631	0.621	0.626	3/4/N
Parser without POS	0.741	0.531	0.619	3/4/N
Parser and POS	0.686	0.586	0.632	3/4/N
ChunksWindow	0.491	0.471	0.481	3/4/N
ChunksWindowPOS	0.636	0.552	0.591	3/4/N
AdvChunksWin	0.493	0.482	0.488	3/4/N
AdvChunksWinPOS	0.637	0.557	0.594	3/4/N

Table 4.12: The results of the chunk-based experiments. The ChunksWindow experiment only utilises the trigram of feature 1, the identity of the chunk containing the two preceding words and the succeeding word. AdvChunksWin employs all the features except 2 and 5.

In all, four sets of experiments were conducted. The first experiment (ChunksWindow) used the trigram of feature 1 only, while the second experiment added a POS trigram (ChunksWindowPOS). The third experiment used all of the chunk-related features above (features 1, 3, 4, 6 and 7) without POS (AdvChunksWin), while the fourth added a POS trigram (AdvChunksWinPOS). The results of this series of experiments are shown in tables 4.12 and 4.13.

4.5.2 Analysis

It is natural to compare the ChunksWindow experiment with the POS experiment, since both employ a basic feature type. The difference between the two is that the features in ChunksWindow are the names of the chunks that hold the preceding and succeeding words, rather than the individual words' POS tags. Since the chunks can be regarded as a more natural syntactic unit to use for prosodic phrase prediction than single words, one might expect the results of the chunk-based experiments to be similar to that of the POS-based experiments. However, it is evident from table 4.12 that the ChunksWindow representation is by far inferior to the POS representation.

In fact, for the break/non-break experiment, the F-Score is more than twenty per cent lower when using just the chunk identity trigram instead of the POS trigram. Precision is only four per cent lower, but the recall is so poor that the F-Score is drastically reduced. This is probably because the POS trigram makes punctuation information explicit, while there is no punctuation information present in the chunk trigram. When the POS information is added, the trained classifier only predicts a break if there is punctuation. Although the chunk trigram is different from the POS trigram in that chunk names are employed instead of POS tags, we conclude

Experiment	**Precision 3**	**Precision 4**	**Recall 3**	**Recall 4**
Punctuation	-	0.934	-	0.598
POS	-	0.650	-	0.821
CXPOS	0.307	0.694	0.142	0.873
Parser without POS	-	0.741	-	0.811
Parser and POS	0.194	0.734	0.043	0.873
ChunksWindow	-	0.494	-	0.736
ChunksWindowPOS	0.207	0.669	0.037	0.824
AdvChunksWin	-	0.491	-	0.720
AdvChunksWinPOS	0.195	0.686	0.049	0.824

Table 4.13: The results of the 3/4/N chunk experiments.

that giving the classifier context information on tags or phrase/chunk names without providing punctuation information is a poor representation.

Of all the syntactic representations we tested, the chunk representation initially appeared to be the most promising, because it is seemingly closer to the prosodic structure than any of the other representations. However, tables 4.12 and 4.13 show that adding the other chunk-related features to the chunk identity trigram does not in general yield good improvements over the ChunksWindow experiments. When adding more chunk-related features, it is only the N/B AdvChunksWin experiment that brings about great improvement over ChunksWindow (15.6% higher F-Score). Therefore, employing all the chunk features on the N/B task is preferable to using just the chunk identity trigram, but the improvement is much lower than expected for the 3/4/N experiment. Furthermore, the AdvChunksWinPOS representation does not give greatly improved results compared to ChunksWindowPOS. It is natural to compare the AdvChunksWin experiments to the parser experiments, since AdvChunksWin represents our best attempt at utilising syntactic chunks for the purposes of prosodic boundary prediction. The only AdvChunksWin experiment with better results than the corresponding parser experiment is the AdvChunksWinPOS N/B experiment, yielding a 0.5% higher F-Score than the Parser and POS experiment. Given that the AdvChunksWin experiments involve more features than the parser experiments and the apparent similarity between the syntactic chunks and prosodic structure, these results are surprising.

In order to further investigate the chunk representation, we executed several more experiments, summarised in tables 4.14 and 4.15 (for clarity, the results for the more detailed prediction level experiments are omitted). The AdvChunks1 and AdvChunks2 experiments were carried out to investigate the effect of employing features 6 and 7, since the information in feature 6 is repeated in feature 7. It can be observed that the AdvChunks1 and AdvChunks2 experiments perform similarly, which is not surprising given that a duplicated piece of information is the only difference between them. The FullChunks experiments were carried out to determine whether the two additional, non-chunk features 2 and 5 would improve performance. For the N/B experiment, this idea is confirmed, since FullChunks gives a 3.8% increase in F-Score over the corresponding AdvChunksWin experiment. However, comparing FullChunksWin, AdvChunks1 and AdvChunks2, it is also evident that the trigrams of features 1 and 7 designed to provide context information are superfluous, since AdvChunks1 and AdvChunks2 performs better than both FullChunksWin and AdvChunksWin (without the POS trigram).

The most interesting result is that ChunksWindowPOS performs so well compared to the other chunk-based experiments. Although these experiments were being carried out with different POS tags than those found in the corpus, if we assume that the MBSP tagger is accurate, we

Experiment	Description
AdvChunks1	Uses all the features apart from POS individually (not employing trigrams for features one and seven).
AdvChunks2	Same as AdvChunks1, but without feature six.
FullChunks	Using all the features apart from POS, with a trigram for features one and seven.
FullChunks and POS	As FullChunks, but including the POS trigram.

Table 4.14: Describing the further chunks experiments.

Experiment	Precision	Recall	F-Score	Prediction level
AdvChunks1	0.723	0.695	0.709	N/B
AdvChunks2	0.722	0.691	0.706	N/B
FullChunks	0.714	0.693	0.704	N/B
FullChunks and POS	0.711	0.770	0.739	N/B

Table 4.15: The results of the further chunk-based experiments.

can conclude that the chunk representation is not as well suited to prosodic boundary prediction as expected. Despite adding many features designed to provide fine-grained information on how the words are distributed among the chunks, we were generally unable to produce improved results in comparison with the parser and CXPOS representations (although AdvChunksWinPOS performs well on the N/B task). Nevertheless, we cannot completely dismiss the syntactic chunks on the basis of these experiments. It may be that the features we have derived are not particularly good, and that other features would have yielded a better outcome. Rather than conducting further investigations into chunks, however, we tested a different syntactic representation.

4.6 Dependency grammar

4.6.1 Feature set and results

In dependency grammars, phrase-structure rules and constituents do not play any fundamental role. The structure of a sentence is instead described in terms of words and binary syntactic or semantic relations between these words. Often, these relations can be depicted in a diagram consisting of arcs connecting words. There are several implementations of the dependency grammar formalism. One of these computational grammars is Link Grammar [27][3].

The basis of link grammar is the observation that for sentences in most languages we can draw non-crossing undirected arcs between syntactically related words. Link parsers produce a link diagram, showing the syntactic links between words, each link labelled with a type [27]. We considered using the link grammar formalism as an alternative to standard constituent-based grammars, since the diagrams show explicit syntactic coupling between words. Because the break indices in the BURNC reflect the prosodic coupling between words, there is a similarity between the two representations on the abstract level. The initial goal was therefore to reimplement an experiment from [17]. Unfortunately, it turned out that the Link Grammar parser employed by

[3]See `http://www.link.cs.cmu.edu/link/`

[17] is not robust, as it is only able to parse a little less than half of the sentences of the BURNC. In addition, it usually returned more than one parse per sentence, such that the correct parse for each sentence would have to be manually selected. Therefore, utilising Link Grammar for predicting prosodic phrasing was deemed infeasible for these experiments. Instead, we carried out a series of experiments with the same features as suggested in [17], using a dependency grammar structure extracted from the textual output of Collins' parser [12][4]. Collins' parser is a robust, statistical parser and parses all (but one) of the sentences in the corpus. The dependencies produced by Collins' parser are not unlike those found by the Link Grammar parser, but differ in that the arcs are directed.

The features employed in [17] were

F1 (label) - The label on the most immediate link crossing a word juncture.

F2 (cover) - The number of links above the most immediate link.

F3 (to_left) - The number of words from the word to the left of the juncture to the left end of the most immediate link.

F4 (to_right) - The number of words from the word to the right of the juncture to the right end of the most immediate link.

F5 (prev_nlc) - The number of links from the word on the left of the juncture to its left.

F6 (prev_nrc) - The number of links from the word on the left of the juncture to its right.

F7 (next_nlc) - The number of links from the word on the right of the juncture to its left.

F8 (next_nrc) - The number of links from the word on the right of the juncture to its right.

The assumptions behind these features are that links between words couple them more tightly prosodically than if no links exist between the words. Also, shorter links will provide tighter coupling than long links, and if there is an increase in the coupling of a word in one direction there will be a corresponding decrease in the other direction. Thus, as the value of *cover* rises, the coupling across the juncture should increase and the more likely it will be that there is no break between the words. Likewise, as the value of *to_left* increases, we expect the strength of coupling over the current juncture to decrease, thus making the juncture more likely to be a break.

Figure 4.7 shows the conversion from the textual output of Collins' parser through to the features presented to the machine learner. One of the training sentences had to be removed from the data set because it could not be parsed. The results of the experiments we carried out with the features F1–F8 are shown in table 4.16 and table 4.17.

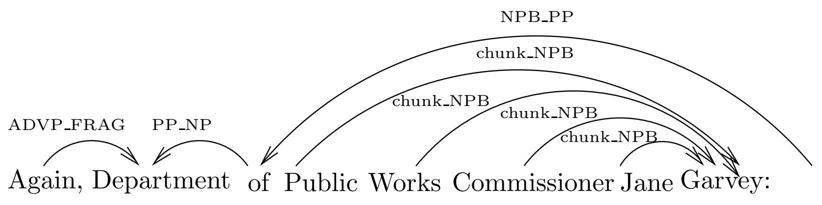
[4]The dependencies were extracted employing a script written by Sabine Buchholz of Toshiba Research Europe. See `http://www.ai.mit.edu/people/mcollins/` for more information on the parser.

Sentence	Again, Department of Public Works Commissioner Jane Garvey:
Parser output	(TOP Department 1 1 (FRAG Department 2 2 (ADVP Again 1 1 Again/RB ,/PUNC,) (NP Department 2 1 (NPB Department 1 1 Department/NNP) (PP of 2 1 of/IN (NPB Garvey 5 5 Public/NNP Works/NNP Commissioner/NNP Jane/NNP Garvey/NNP :/PUNC:)))))

Extracted

Word	**Tag**	**Punc.**	**Dependency**	**Distance**
Again	RB	,/PUNC,	ADVP_FRAG	1
Department	NNP	-	TOP_ROOT	0
of	IN	-	PP_NP	-1
Public	NNP	-	chunk_NPB	4
Works	NNP	-	chunk_NPB	3
Commissioner	NNP	-	chunk_NPB	2
Jane	NNP	-	chunk_NPB	1
Garvey	NNP	:/PUNC:	NPB_PP	-5

Dependency diagram



Features

F1	**F2**	**F3**	**F4**	**F5**	**F6**	**F7**	**F8**	**Break value**
ADVP_FRAG	0	0	0	0	1	0	0	3
PP_NP	0	0	0	0	0	1	0	1
NPB_PP	0	0	4	1	0	0	1	1
chunk_NPB	1	0	3	0	1	0	1	1
chunk_NPB	2	0	2	0	1	0	1	3
chunk_NPB	3	0	1	0	1	0	1	4
chunk_NPB	4	0	0	0	1	1	0	1
chunk_NPB	0	5	0	1	0	=	=	4

Figure 4.7: Deriving features for the dependency grammar experiment from the output of Collins' parser. The values of **next_nlc** and **next_nrc** (features 7 and 8) are set to "=" for the last word, to denote end of sentence.

4.6.2 Analysis

Tables 4.16 and 4.17 show that the dependency grammar representation is the most promising of the deeper representations. Both with and without the POS trigram it yields the best results for both prediction levels compared to the parser and chunk representations.

Based on the tables, it is natural to compare the dependency representation to CXPOS, since they have similar F-Scores. While the dependency grammar features are clearly better than the CXPOS features for the detailed prediction level (comparing the F-Scores of Dependency and POS experiment with the CXPOS experiment), the difference between recall and precision is much higher when using dependency grammar. The dependency features allow the classifer to retain a high precision for predicting breaks at the cost of recall. In practice, this means that the breaks predicted by the classifer are often correct, but that too few of the phrase breaks are

Experiment	**Precision**	**Recall**	**F-Score**	**Prediction level**
Punctuation	0.969	0.406	0.572	N/B
POS	0.695	0.761	0.726	N/B
CXPOS	0.747	0.804	0.774	N/B
Parser without POS	0.737	0.680	0.707	N/B
Parser and POS	0.702	0.785	0.741	N/B
ChunksWindow	0.653	0.418	0.510	N/B
ChunksWindowPOS	0.706	0.778	0.740	N/B
AdvChunksWin	0.629	0.708	0.666	N/B
AdvChunksWinPOS	0.710	0.787	0.746	N/B
Dependency	0.779	0.667	0.719	N/B
Dependency and POS	0.740	0.821	0.779	N/B
Punctuation	0.934	0.391	0.551	3/4/N
POS	0.650	0.537	0.588	3/4/N
CXPOS	0.631	0.621	0.626	3/4/N
Parser without POS	0.741	0.531	0.619	3/4/N
Parser and POS	0.686	0.586	0.632	3/4/N
ChunksWindow	0.491	0.471	0.481	3/4/N
ChunksWindowPOS	0.636	0.552	0.591	3/4/N
AdvChunksWin	0.493	0.482	0.488	3/4/N
AdvChunksWinPOS	0.637	0.557	0.594	3/4/N
Dependency	0.729	0.529	0.613	3/4/N
Dependency and POS	0.732	0.571	0.642	3/4/N

Table 4.16: Results of the dependency grammar experiments, including previous experimental results for comparison.

Experiment	**Precision 3**	**Precision 4**	**Recall 3**	**Recall 4**
Punctuation	-	0.934	-	0.598
POS	-	0.650	-	0.821
CXPOS	0.307	0.694	0.142	0.873
Parser without POS	-	0.741	-	0.811
Parser and POS	0.194	0.734	0.043	0.873
ChunksWindow	-	0.494	-	0.736
ChunksWindowPOS	0.207	0.669	0.037	0.824
AdvChunksWin	-	0.491	-	0.720
AdvChunksWinPOS	0.195	0.686	0.049	0.824
Dependency	0.429	0.749	0.056	0.779
Dependency and POS	0.375	0.749	0.037	0.853

Table 4.17: The results of the 3/4/N dependency grammar experiments.

predicted. The F-Score is a means of weighing the precision and recall into an average measure, and thus the dependency grammar representation scores highly. It is not clear if in practice either precision or recall is most important, or whether they should be roughly the same. The only way to answer this question would be to carry out perceptual experiments in which subjects listen to speech produced using CXPOS features and dependency grammar features.

Chapter 5

Conclusions

5.1 Summary and conclusion

5.1.1 Overall results

We have implemented and evaluated six of the most important syntactic representations for predicting prosodic phrase breaks. Three of these, CXPOS, constituency-based parsing and dependency grammar, stand out as superior to the rest. The differences in performance between these three representations are not so great, however, that we can safely pick a "winner". While both the parser and dependency grammar experiments show improvements in F-Score over CX-POS when predicting level 3 and level 4 phrase breaks rather than breaks/non-breaks, there is still a large difference between precision and recall. As mentioned, we do not know which of precision and recall to prioritise, so without perceptual evaluation it is not feasible to claim that either the parser or the dependency grammar representation is more suited to break prediction than the other. It is quite surprising that the shallow CXPOS representation performs so well compared to the deeper chunk-based, parser-based and dependency grammar-based representations. This indicates that it may not be worth the extra time taken to extract features from the more advanced syntactic representations. While there are certainly fast, robust parsers around, none can match the speed of the extraction of CXPOS features, which is based only on the words and POS tags of the text.

The most surprising result, however, is the relatively poor performance of the syntactic chunks, as produced by a shallow parser. On a superficial level, it seems that the syntactic chunks structure is much closer to the flat structure of prosody, so we expected the chunk-based features to yield very good performance. Given the use of external tagging, we cannot rule out that the results would have been better if we had employed a more suitable shallow parser. It would have been useful to employ the CASS chunker[1] for comparison, or even vary the features, if time had permitted [2].

5.1.2 Predicting level 3 and level 4 phrase breaks

In general, the task of predicting a level 3 phrase break is much more difficult than predicting a level 4 break. From inspection of the training and test data sets, we would intuitively expect this result, since there are more than twice as many level 4 breaks as level 3 breaks. Moreover, the level 3 breaks often occur in contexts without punctuation. As shown, punctuation is often taken by the machine learner as a strong indicator of a level 4 phrase break. But where there is no punctuation, a non-break is much more likely than a break. The non-break instances are so

[1]See http://www.vinartus.net/spa/

numerous in these cases that the classifier almost always chooses non-break.

Scrutinising the classifier's output for the various experiments in detail, it is clear that using punctuation alone does not provide enough information for the classifier to discriminate the two break levels. Almost all the level 3 breaks occur in contexts without punctuation, and where there is no punctuation, the classifier most often decides non-break. The level 4 breaks are more numerous than the level 3 breaks where punctuation exists, so no breaks of type 3 are predicted here either. It appears that the classifier is able to predict that there should be a break but not which type the break should be. Many of the breaks correctly predicted in the N/B-experiments are level 3, but the classifier often gets the same level 3 breaks wrong when carrying out the 3/4/N task. There are also confusions between level 3 and 4 breaks, usually in cases where there are several examples in the training data of the same feature values for both levels. Since break level 4 is more common than 3, 4 is often predicted in these cases.

The experiments that give the best results when we predict break/non-break also yield the best performance for the fine-grained experiments. In particular, the CXPOS experiment is far superior to all the other experiments when it comes to deciding on level 3 phrase breaks, as displayed in table 4.17. However, the actual precision and recall for the level 3 breaks is still low, so that even for the CXPOS experiment we cannot be satisfied with the result.

5.1.3 Summary

In summary, we have compared different syntactic representations for their effectiveness in predicting prosodic phrase breaks. The results are useful in that they suggest that shallow features like CXPOS can perform just as well as deep features. Especially in the areas of perceptual evaluation and the significance of the distinction between different phrase breaks more research is clearly required if we are to choose a particular representation. Some suggestions for such work are outlined in the next section.

5.2 Further work

Although we have compared six important syntactic representations, there are still many holes left to be filled in our knowledge of prosodic phrase break prediction for TTS. One of the most valuable contributions would be a thorough perceptual study of how the accuracy of phrase break prediction affects the listener's experience. Two of the few studies completed on this topic are [28] and [21]. These two papers concentrate mainly on how to take into account the fact that there may be more than a single prosodic phrase structure for any given sentence, which affects performance evaluation. Both publications showed that disagreement between human evaluators is common when assessing the phrasing produced by prosodic modules in TTS systems. Such variability in evaluation is so commonplace that it was found that a phrasing could only be rated as unacceptable if more than half of the evaluators rejected it. This highlights a weakness in our, and most other authors', work, implying that simply relying on the corpus annotations is unlikely to produce results that are useful in practice. It was also observed in the two papers that the number of sentences that are fully compliant with the prosodic phrasing assigned by annotators is important. A single misplaced break (or non-break) can render a sentence unacceptable to human listeners. Hence, if this project were extended, we would recommend that more attention be paid to human evaluation and sentence-level assessment.

None of the studies reported so far concentrate on the distinction between level three and level four phrase breaks. Most studies focus on the distinction between breaks and the absence of breaks rather than examining the finer difference between breaks. But when listening to the radio speech samples in the BURNC, it is obvious that such a distinction is valid. Nevertheless, it has not been proved that distinguishing between the different types of prosodic phrase breaks improves the performance of a practical TTS system. A study on the use of different types of phrase breaks as compared with a single type of break would therefore be worthwhile.

Another issue that has not been fully resolved is how one should trade off precision against recall. [28] goes some way toward answering this when discussing evaluation at the sentence level. But since no prosodic phrasing module can be expected to correctly identify all the breaks, it would be of practical value to find out if one should give priority to either precision or recall, focus on maximising F-Score, or instead try to balance precision and recall as much as possible.

The corpora available for use in universities are not large enough for the purposes of training machine learners for prosodic boundary prediction. The BURNC contains just over 32000 words but is full of errors and inconsistencies that limit the number of useful utterances. It would therefore be of helpful to research if these BURNC problems were corrected. An alternative corpus (for English) is the MARSEC database of spoken British English [25]. Consisting of recorded radio stories and totalling about 40000 words, this corpus is somewhat larger than the BURNC, but compared to the 68000-word corpus employed in [18], it is still not particularly large. Further work on corpora would therefore be of benefit to university research, but probably not until more is known about how best to differentiate between different types of breaks.

Bibliography

[1] Steven Abney. Chunks and Dependencies: Bringing Processing Evidence to Bear on Syntax. In *Computational Linguistics and the Foundations of Linguistic Theory.* CSLI, 1995.

[2] Steven Abney. *The SCOL Manual Version 0.1b*, 1997.

[3] Steven P. Abney. Parsing by chunks, 1991. In: Robert C. Berwick, Steven P. Abney, and Carol Tenny, editors, Principle-Based Parsing: Computation and Psycholinguistics, pages 257–278. Kluwer Academic Publishers, Boston, 1991.

[4] Bengt Altenberg. Prosodic Patterns in Spoken English: Studies in the Correlation Between Prosody and Grammar for Text-to-Speech Conversion. In *Lund Studies in English*, volume 76. Lund University Press, 1987.

[5] Michaela Atterer. Assigning prosodic structure for speech synthesis: a rule-based approach. In *Proceedings of the First International Conference on Speech Prosody SP2002*, Aix-en-Provence, France, 2002.

[6] Michaela Atterer and Ewan Klein. Integrating Linguistic and Performance-Based Constraints for Assigning Phrase Breaks. In *Proceedings of COLING*, Taipei, Taiwan, 2002.

[7] J. Bachenko and E. Fitzpatrick. A Computational Grammar of Discourse-Neutral Prosodic Phrasing in English. *Computational Linguistics*, 16(3), September 1999.

[8] Daniel M. Bikel. Design of a Multi-lingual, Parallel-processing Statistical Parsing Engine. In *Proceedings of Human Language Technology Conference*, San Diego, 2002.

[9] Alan W. Black and Paul Taylor. Assigning Phrase Breaks from Part-of-Speech Sequences. In *Proceedings of Eurospeech'97*, pages 995–998, Rhodes, Crete, 1997.

[10] Bertjan Busser, Walter Daelemans, and Antal van den Bosch. Predicting phrase breaks with memory-based learning. In *Proceedings 4th ISCA Tutorial and Research Workshop on Speech Synthesis*, pages 29–34, Perthshire, Scotland, August/September 2001.

[11] Glenn Carroll and Mats Rooth. Valence Induction with a Head-Lexicalized PCFG. In *Proceedings of the 3rd Conference on Empirical Methods in NLP Workshop (EMNLP3)*, Granada, 1998.

[12] Michael Collins. *Head-Driven Statistical Models for Natural Language Parsing.* PhD thesis, University of Pennsylvania, 1999.

[13] Walter Daelemans, Jakub Zavrel, Ko van der Sloot, and Antal van den Bosch. TiMBL: Tilburg Memory-Based Learner version 5.0 Reference Guide. Technical Report ILK 03-10, Tilburg University, November 2003.

[14] Marcus Fach. A Comparison Between Syntactic and Prosodic Phrasing. In *Proceedings of Eurospeech'99*, volume 1, pages 527–530, Budapest, 1999.

[15] J. Hirschberg and P.Prieto. Training Intonational Phrasing Rules Automatically for English and Spanish Text-to-Speech. In *Proceedings of the Second ESCA/IEEE Workshop on Speech Synthesis*, pages 64–68, 1994.

[16] Julia Hirschberg and Owen Rambow. Learning Prosodic Features using a Tree Representation. In *Proceedings of Eurospeech'01*, Aalborg, 2001.

[17] Andrew J. Hunt. Improving Speech Understanding Through Integration of Prosody and Syntax. In *Proc. 7th Aust. Joint Conf. on Artificial Intelligence*, pages 442–449, Armidale, Australia, 1994.

[18] P. Koehn, S. Abney, J. Hirschberg, and M. Collins. Improving Intonational Phrasing with Syntactic Information. In *Proceedings of ICASSP-00*, Istanbul, 2000.

[19] K.Silverman, M. Beckman, J. Pitrelli, M. Ostendorf, C. Wightman, P. Price, J. Pierrehumbert, and J. Hirschberg. TOBI: A Standard for Labeling English Prosody. In *Proceedings of the International Conference on Spoken Language Systems*, volume 2, pages 867–870, Banff, Canada, October 1992.

[20] Mark Y. Liberman and Kenneth W. Church. Text Analysis and Word Pronunciation in Text-to-Speech Synthesis. In S. Furui and M. Sondhi, editors, *Advances in Speech Signal Processing*, pages 791–831. Marcel Dekker, Inc., 1992.

[21] Erwin Marsi. Optionality in evaluating prosody prediction. In *Proceedings of 5th ISCA Speech Synthesis Research Workshop*, pages 13–18, Pittsburgh, USA, June 2004.

[22] Erwin Marsi, Martin Reynaert, Antal van den Bosch, Walter Daelemans, and Veronique Hoste. Learning to predict pitch accents and prosodic boundaries in Dutch. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 489–496, Sapporo, Japan, 2003.

[23] M. Ostendorf, P.J. Price, and S. Shattuck-Hufnagel. The Boston University Radio News Corpus. Technical Report ECS-95-001, Electrical, Computer and Systems Engineering Department, Boston University, Boston, MA, 1995.

[24] Mari Ostendorf, Patti Price, Colin Wightman, and John Bear. The Use of Relative Duration in Syntactic Disambiguation. In *Proceedings of DARPA Speech and Natural Language Workshop*, pages 26–31, 1990.

[25] P. Roach and G. Knowles and T. Varadi and S. Arnfield. MARSEC: a Machine-Readable Spoken English Corpus. *Journal of the International Phonetic Association*, 24(1), May 1994.

[26] Ian. Read and Stephen Cox. Using Part-Of-Speech For Predicting Phrase Breaks. In *Proceedings of 8th International Conference on Spoken Language Processing*, volume 1, 2004.

[27] Daniel Sleator and Davy Temperley. Parsing English with a Link Grammar. Technical Report CMU-CS-91-196, Carnegie Mellon University Computer Science, October 1991.

[28] M. Céu Vianal, Luis C. Oliveira, and Ana I. Mata. Machine and Human Evaluation, International Journal of Speech Technology. *International Journal of Speech Technology*, 6(1):83–94, January 2003. Kluwer Academic Publishers.

[29] Michelle Q. Wang and Julia Hirschberg. Automatic Classification of Intonational Phrase Boundaries. *Computer Speech and Language*, 6:175–196, 1992.

Appendix A

TRAINING SET:

f1ajrlp1	f1ajrlp2	f1ajrlp3	f1ajrlp4	f1ajrlp5
f1ajrlp6	f1aprlp2	f1aprlp3	f1aprlp4	f1arrlp1
f1arrlp2	f1arrlp3	f1arrlp4	f1arrlp5	f1arrlp6
f1arrlp7	f1atrlp5	f1atrlp7	f1as01p1	f1as01p2
f1as01p3	f1as01p4	f1as02p1	f1as02p2	f1as02p3
f1as03p1	f1as03p2	f1as03p3	f1as03p4	f1as03p5
f1as04p1	f1as04p2	f1as04p3	f1as04p4	f1as04p5
f1as04p6	f1as05p1	f1as05p2	f1as05p3	f1as05p4
f1as05p5	f1as06p1	f1as06p2	f1as06p3	f1as06p4
f1as06p5	f1as06p6	f1as06p7	f1as06p8	f1as07p2
f1as07p3	f1as07p4	f1as07p5	f1as08p1	f1as08p2
f1as08p3	f1as08p4	f1as08p5	f1as09p1	f1as09p2
f1as09p3	f1as09p4	f1as09p6	f1as10p2	m1bjrlp1
m1bjrlp2	m1bjrlp3	m1bjrlp4	m1bjrlp5	m1bjrlp6
m1bprlp1	m1bprlp2	m1bprlp3	m1bprlp4	m1brrlp1
m1brrlp2	m1brrlp3	m1brrlp4	m1brrlp5	m1brrlp6
m1brrlp7	m1btrlp1	m1btrlp2	m1btrlp3	m1btrlp4
m1btrlp5	m1btrlp6	m1btrlp7	m1bs01p2	m1bs01p3
m1bs01p4	m1bs01p5	m1bs02p1	m1bs02p2	m1bs02p3
m1bs02p4	m1bs02p5	m1bs02p7	m1bs03p3	m1bs03p4
m1bs03p6	m1bs04p3	m1bs04p4	m1bs04p6	m1bs04p8
m1bs05p2	m1bs05p3	m1bs05p4	m1bs05p5	m1bs07p7
m1bs07p9	m1bs07pb	m1bs08p5	m1bs08p6	m1bs08p7
m1bs08p9	m1bs09p1	m1bs10p1	m1bs10p2	m1bs10p3
m1bs10p4	m1bs10p5	m1bs10p6	m1bs11p1	m1bs11p2
m1bs11p5	m1bs11p6	m1bs11p7	m1bs12p1	m1bs12p2
m1bs12p3	m1bs12p4	m1bs12p5	m1bs13p1	m1bs13p2

TRAINING SET CONTINUED:

m1bs13p3	m1bs14p1	m1bs14p2	m1bs15p1	f2bjrop1
f2bjrop2	f2bjrop3	f2bjrop4	f2bjrop5	f2bjrop6
f2bjrlp1	f2bjrlp2	f2bjrlp3	f2bjrlp4	f2bjrlp5
f2bjrlp6	f2bprop1	f2bprop2	f2bprop3	f2bprop4
f2bprlp1	f2bprlp2	f2bprlp3	f2bprlp4	f2brrlp1
f2brrlp2	f2brrlp3	f2brrlp4	f2brrlp5	f2brrlp6
f2brrlp7	f2btrop1	f2btrop2	f2btrop3	f2btrop4
f2btrop5	f2btrop6	f2btrop7	f2btrlp1	f2btrlp2
f2btrlp3	f2btrlp4	f2btrlp5	f2btrlp6	f2btrlp7
f2bs01p1	f2bs01p2	f2bs01p3	f2bs02p2	f2bs02p3
f2bs02p4	f2bs02p5	f2bs02p6	f2bs03p1	f2bs03p2
f2bs03p3	f2bs03p4	f2bs03p5	f2bs04p2	f2bs04p3
f2bs04p4	f2bs04p5	f2bs05p1	f2bs05p2	f2bs05p3
f2bs05p4	f2bs06p1	f2bs06p2	f2bs06p3	f2bs06p4
f2bs07p4	f2bs07p5	f2bs08p1	f2bs08p2	f2bs08p3
f2bs08p6	f2bs09p1	f2bs09p2	f2bs09p3	f2bs09p5
f2bs10p1	f2bs10p2	f2bs10p3	f2bs10p4	f2bs10p5
f2bs10p6	f2bs11p1	f2bs11p2	f2bs11p4	f2bs11p5
f2bs12p1	f2bs12p4	f2bs12p5	f2bs12p6	f2bs12p7
f2bs12p8	f2bs12p9	f2bs13p1	f2bs13p2	f2bs13p3
f2bs13p4	f2bs13p5	f2bs14p1	f2bs14p2	f2bs14p3
f2bs14p4	f2bs14p5	f2bs14p6	f2bs14p7	f2bs16p1
f2bs16p2	f2bs16p3	f2bs16p4	f2bs16p5	f2bs17p2
f2bs17p3	f2bs17p4	f2bs17p6	f2bs17p7	f2bs17p8
f2bs18p2	f2bs18p3	f2bs18p4	f2bs18p5	f2bs18p6
f2bs18p8	f2bs19p1	f2bs19p2	f2bs19p3	f2bs19p4
f2bs20p1	f2bs20p4	f2bs21p1	f2bs21p3	f2bs21p5
f2bs23p1	f2bs24p1	f2bs24p2	f2bs25p1	f2bs25p2
f2bs26p1	f2bs27p1	f2bs27p2	f2bs28p1	f2bs28p2
f2bs29p1	f2bs29p2	f2bs29p3	f2bs29p4	f2bs30p1
f2bs31p1	f2bs31p2	f2bs31p3	f2bs31p4	f2bs31p5
f2bs31p6	f2bs32p1	f2bs32p2	f2bs32p3	f2bs32p4
f2bs32p5	f2bs33p1	f2bs33p2	f2bs33p3	f2bs33p4
f2bs34p1	f2bs34p2	f2bs34p3	f2bs34p4	m2bjrlp6
m2bprlp1	m2bprlp3	m2bprlp4	m2brrlp2	m2brrlp3
m2brrlp4	m2bs01p3	m2bs01p4	m2bs01p7	m2bs01p8
m2bs01p9	m2bs02p3	m2bs02p4	m2bs02p5	m2bs02p6
m2bs02p7	m2bs02p8	m2bs02p9	m2bs02pa	m2bs03p1
m2bs03p2	m2bs03p3	m2bs03p4	m2bs03p5	m2bs03p6
m2bs03p7	m2bs03p8	m2bs03p9	m2bs03pa	m2bs04p2
m2bs04p3	m2bs04p4	m2bs04p5	f3ajrlp1	f3ajrlp4
f3ajrlp5	f3ajrlp6	f3aprlp2	f3aprlp3	f3arrlp1
f3arrlp4	f3arrlp6	f3atrlp2	f3atrlp3	

TEST SET:

f3atrlp4	f3atrlp5	f3atrlp7	f3as02p1	f3as03p1
f3as04p1	f3as07p1	f3as08p1	f3as09p1	f3as09p2
m3bjrlp1	m3bjrlp3	m3bjrlp4	m3bprlp2	m3bprlp3
m3brrlp1	m3brrlp2	m3brrlp3	m3brrlp4	m3brrlp5
m3brrlp6	m3btrlp1	m3btrlp2	m3btrlp4	m3btrlp5
m3btrlp6	m3btrlp7			