Automatic Utterance Segmentation in Instant Messaging Dialogue

Edward Ivanovic
Department of Computer Science and Software Engineering
University of Melbourne
edwardi@csse.unimelb.edu.au

Abstract

Instant Messaging (IM) chat sessions are real-time, text-based conversations which can be analyzed using dialogue-act models. Dialogue acts represent the semantic information of an utterance, however, messages must be segmented into utterances before classification can take place. We describe and compare two statistical methods for automatic utterance segmentation and dialogue-act classification in task-based IM dialogue. It is shown that IM messages can be automatically segmented and classified to a very high accuracy using statistical machine learning.

1 Introduction

Dialogue acts are a useful first level of analysis for describing discourse structure as they represent the illocutionary force of utterances such as assertions and declarations. Early work on speech act theory by Austin (1962) and Searle (1979) has been extended in dialogue acts to model the conversational functions that utterances can perform. Table 1 shows an example dialogue with utterance segments and dialogue acts.

As illustrated in Table 1, some messages contain multiple utterances and thus require segmentation before each utterance can be classified as a dialogue act. Once utterances are classified, the dialogue-acts may then be used for subsequent tasks such as machine translation (Tanaka and Yokoo, 1999), dialogue game detection (Levin et al., 1999), and, in the case of spoken dialogue, speech recognition (Stolcke et al., 2000).

Speaker	Message
Sanders	[Hello Customer]$^{\text{CONVENTIONAL-OPENING}}$, [thank you for contacting MSN Shopping]$^{\text{THANKING}}$. [This is Sanders and I look forward to assisting you today]$^{\text{STATEMENT}}$
Sanders	[How are you doing today?]$^{\text{OPEN-QUESTION}}$
Customer	[good]$^{\text{STATEMENT}}$, [thanks]$^{\text{THANKING}}$
Sanders	[How may I help you today?]$^{\text{OPEN-QUESTION}}$

Table 1: An example of the beginning of a dialogue in our corpus showing utterance boundaries and dialogue-act tags in superscript.

Instant Messaging (IM) consists of two or more people typing messages to each other in real time on a line-by-line basis. Although IM dialogue can take place with a group of people simultaneously writing to each other, for the purposes of this study we assume only two-party dialogue. As described in Ivanovic (2005), sequences of words are grouped into three levels: the first level is a *Turn*, consisting of at least one *Message*, which consists of at least one *Utterance*, defined as follows:

Turn: A dialogue participant normally writes one or more message then waits for the other participant to respond, hence taking *turns* in writing messages.

Message: A message is defined as a group of words that are sent from one dialogue participant to the other participant as a single unit. This is usually achieved by typing a message and pressing the Enter key or a 'Send' button on the client program. A single turn can span multiple messages.

Utterance: An utterance can be thought of as one complete semantic unit with respect to dialogue acts. This can be a complete sentence or as short as an emoticon (e.g. ":-)" to smile). Messages contain one or more utterances. Although it is possible to send a message mid-utterance, resulting in utterances spanning messages, so such instances occur in our corpus, which our model assumes. Example utterances, enclosed within brackets, are shown in Table 1.

Most utterance segmentation research to date has focussed on transcribed speech. The aim of speech segmentation, however, is different to that required by dialogue act classification. That is, large-vocabulary speech recognisers segment speech into *acoustic* segments for more efficient processing, using criteria such as non-speech intervals and turn boundaries in dialogue. These methods are not appropriate for IM utterance segmentation because the acoustic segmentation methods rely on the recorded waveform of speech, which does not exist in IM dialogue.

We show that utterance segmentation for dialogue act classification requires very different criteria to transcribed speech segmentation. Our methods for dialogue act utterance segmentation are based on linguistically and statistically motivated approaches.

The rest of this paper is organised as follows. The data collection and dialogue act tag set are described in Section 2. The methods and language models used in our experiment are explained in Section 3. Evaluation techniques we use are in Section 4. Our experimental results and discussion are in Section 5, with the conclusions and future work in Section 6.

2 Data and Dialogue Act Tag Set

Our data was collected in previous work (Ivanovic, 2005) from an online IM-based support service and consisted of nine chat sessions, totalling 550 utterances, 6,500 words, with a mean message length of 10 words. The chat sessions were manually segmented into utterances by one person and used as a gold standard. These utterances were then annotated by three people

Table 2 shows the dialogue act tag set we use, which was also taken from previous work as described in Ivanovic (2005). The tag set was chosen by manually labelling our corpus using tags that seemed appropriate from the 42 tags used by Stolcke et al. (2000), which in turn were based on the Dialog Act Markup in Several Layers (DAMSL) tag set (Core and Allen, 1997).

Tag	Example	%
STATEMENT	I am sending you the page now	36.0
THANKING	Thank you for contacting us	14.7
YES-NO-QUESTION	Did you receive the page?	13.9
RESPONSE-ACK	Sure	7.2
REQUEST	Please let me know how I can assist	5.9
OPEN-QUESTION	how do I use the international version?	5.3
YES-ANSWER	yes, yeah	5.1
CONVENTIONAL-CLOSING	Bye Bye	2.9
NO-ANSWER	no, nope	2.5
CONVENTIONAL-OPENING	Hello Customer	2.3
EXPRESSIVE	haha, :-), grr	2.3
DOWNPLAYER	my pleasure	1.9

Table 2: The 12 dialogue act labels with examples and frequencies given as percentages of the total number of utterances in our corpus.

A Kappa analysis used to compare inter-annotator agreement normalised for chance (Siegel and Castellan, 1988), resulted in a value of 0.87 with 89% agreement (Ivanovic, 2005). A Kappa statistic of 0.8 and above is considered a satisfactory indication that our corpus can be labelled reliability using our tag set (Carletta, 1996).

A complete list of the 12 dialogue acts we used is shown in Table 2 along with examples and the frequency of each dialogue act in our corpus.

3 Methods

Our first goal was to determine which features obtained from IM transcripts would be useful in detecting utterance segments within messages. The data available from IM chat transcripts are the *speaker, message text,* and *time stamp* of each message. Unlike regular written prose, IM chats are often very informal—omitting usual punctuation such as commas, periods, question marks, and initial capital letters for proper names and new sentences. Spelling mistakes, acronyms for common phrases, and ungrammatical messages are also quite common.

The observation that utterances in our data do not cross message boundaries allows us to focus on segmenting one message at a time. We use two ap-

A: $[_{INTJ}$ hello $_{UH}]$ $[_{NP}$ customer $_{NN}]$, $_O$ <s> $[_{VP}$ thank $_{VB}]$ $[_{NP}$ you $_{PRP}]$ $[_{PP}$ for $_{IN}]$ $[_{VP}$ contact $_{VBG}]$ $[_{NP}$ Msn $_{NNP}$ Shopping $_{NNP}]$. $_O$ <s> $[_{NP}$ this $_{DT}]$ $[_{VP}$ be $_{VBZ}]$ $[_{NP}$ Sanders $_{NNP}]$ $[_O$ and $_{CC}]$ $[_{NP}$ I $_{PRP}]$ $[_{VP}$ look $_{VBP}]$ $[_{ADVP}$ forward $_{RB}]$ $[_{PP}$ to $_{TO}]$ $[_{VP}$ assist $_{VBG}]$ $[_{NP}$ you $_{PRP}]$ $[_{NP}$ today $_{NN}]$. $_O$
A: $[_{ADVP}$ how $_{WRB}]$ $[_O$ be $_{VBP}]$ $[_{NP}$ you $_{PRP}]$ $[_{VP}$ do $_{VBG}]$ $[_{NP}$ today $_{NN}]$? $_O$
B: $[_{ADJP}$ good $_{JJ}]$, $_O$ <s> $[_{NP}$ thanks $_{NNS}]$

Figure 1: Sample tagged and chunked data.

proaches to segment the messages: Hidden Markov Models (HMMs) and a probabilistic model based on parse trees. We discuss each of these in turn.

3.1 HMM Method

In the absence of reliable punctuation cues, we looked at approaches based on the available lexical information. One such method was to use an HMM to find the most likely segment boundaries. We experimented with three versions of the HMM approach, based on sequences of: (i) lemmas, (ii) part of speech tags, and (iii) head words of chunks.

The rationale behind using chunks is that the number of possible segments is reduced since utterance boundaries do not lie within chunks. The data was assigned POS tags and segmented into chunks via the FNTBL Toolkit (Ngai and Florian, 2001), which is an efficient implementation of Eric Brill's Transformation-based learning algorithm (Brill, 1995). Lemmatisation on our corpus was performed using the morphological tools described in Minnen et al. (2001).

Figure 1 illustrates some characteristics of the data. Utterance boundaries are marked by <s> tags, chunk boundaries are enclosed within brackets, and words' POS tags are shown in subscript after the word. The actual chunks in the data use IOB tags similar to that described in Ramshaw and Marcus (1995).

We first trained an n-gram statistical language model with add-one smoothing and Katz backoff (Katz, 1987) to hypothesize the most probable locations of utterance boundaries for each individual message. The resulting segmentations were then evaluated using the WindowDiff metric as described in Section 4.

Elements used to represent the segments were lemmas, POS tags, and chunks. Segment beginnings in our training data were marked with a <s> tag. This allowed each element to be in one of two states: S or NO-S depending on whether it had a <s> tag before it. We build two probability distributions P_S and P_{NO-S} representing the probability that token t_k is at the beginning of a segment or not, respectively. Using this state information permits us to use an HMM with the following forward computation for the likelihoods of the states at each position k as described by Stolcke and Shriberg (1996):

$$\begin{aligned}
P_{NO-S}(t_1...t_k) &= P_{NO-S}(t_1...t_{k-1}) \times \\
&\quad p(t_k|t_{k-2}t_{k-1}) \\
&\quad +P_S(t_1...t_{k-1}) \times \\
&\quad p(t_k|\texttt{<s>}t_{k-1}) \\
P_S(t_1...t_k) &= P_{NO-S}(t_1...t_{k-1}) \times \\
&\quad p(\texttt{<s>}|t_{k-2}t_{k-1})p(t_k|\texttt{<s>}) \\
&\quad +P_S(t_1...t_{k-1}) \times \\
&\quad p(\texttt{<s>}|\texttt{<s>}t_{k-1})p(t_k|\texttt{<s>})
\end{aligned}$$

where t is a lemma, POS or chunk token. A corresponding Viterbi algorithm is then used to find the most likely sequence of S and NO-S states given the lemmatised words. Note that this model treats segment marks, <s>, as tokens.

3.2 Parse Tree Method

Parse trees generally contain nodes of clauses as illustrated in Figure 2. We assume that utterance boundaries only occur at major syntactic boundaries. This is similar in principle to the use of chunks as described in Section 3.1, where we hypothesise that a segment boundary exists before each token. The notion of a token, however, changes from representing chunks to sub-trees within a parse tree. Since a token in this context represents multiple words, and utterance segments may only occur in between tokens, this method significantly reduces the possibility of obtaining false-positive segment boundaries

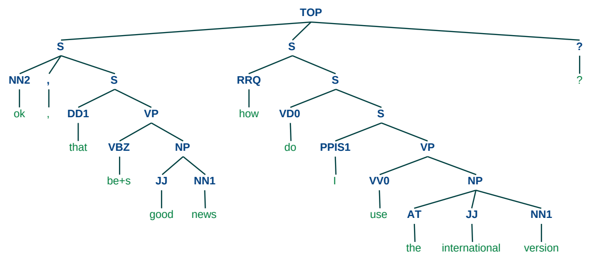

Figure 2: Parse tree of a message showing utterances separated into sub-trees as generated by RASP (Briscoe and Carroll, 2002).

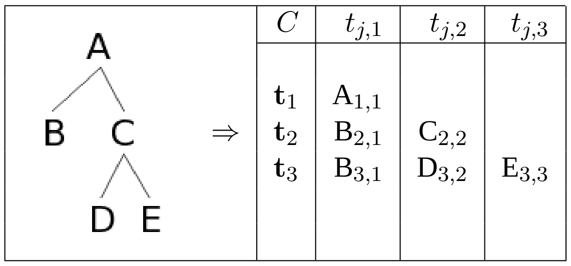

Figure 3: Proper analyses, C_1^3 from a parse tree.

when compared with using word or chunk tokens assuming correct parse trees. If the parse trees are not correct, however, this technique will have the opposite effect. This is discussed in more detail in Section 5.3.

To produce the parse trees, we use the RASP (Robust Accurate Statistical Parsing) parser described in Briscoe and Carroll (2002). RASP is designed to be domain-independent in order to handle text from different genres. Given that our data comes from instant messaging, which exhibits less predictable prose than that typically found in newspapers, we chose RASP over other parsers such as Collins (1999) and Charniak (2000) that are optimised on the Wall Street Journal treebank.

Utterance segments in our data always occur within a maximum depth of 2 nodes from the root of the parse tree. Using this depth limit, we first build a table of possible "cuts" through the tree. These cuts, or proper analyses as described in Chomsky (1965), contain every combination of sub-trees, as illustrated in Figure 3, resulting in a sequence C of nodes:

$$C = \mathbf{t}_1, \mathbf{t}_2, \mathbf{t}_3, \ldots, \mathbf{t}_h \qquad (1)$$

where each combination $\mathbf{t}_i$ is a sequence of tree nodes such that:

$$\mathbf{t}_i = t_1, t_2, t_3, \ldots, t_n \qquad (2)$$

where the leaves of each tree node t_i represent a possible utterance.

We then calculate the most likely dialogue act for the leaves (words) within each node, independently in the combination table. The result and its corresponding dialogue-act are stored with the node t_i. Next, we calculate the probability of a correct sequence of utterances based on the product of the dialogue-act classification probabilities, using the following formulae:

$$\langle \mathbf{t}^*, \mathbf{d}^* \rangle = \underset{\langle \mathbf{t} \in C, \mathbf{d} \rangle}{\arg\max} \prod_{t_i \in \mathbf{t}} \hat{P}(d_i | t_i, d_{i-1})$$

$$\hat{P}(d_i | t_i, d_{i-1}) = P(d_i | d_{i-1}) \prod_{v \in \text{leaves}(t_i)} P(v | d_i)$$

where $\mathbf{t}^*$ is the best node combination (or segments), C is the set of proper analyses, $P(d_i|t_i)$ is the probability of node $t_i \in \mathbf{t}$ being dialogue-act d based on its leaves (words), d_{i-1} is the previously assigned dialogue-act (using bigrams), and v is a word in node t_i.

Using this method has the effect of evaluating the classification and segmentation tasks at the same time, taking the most probable combination. Algorithm 1 shows the process used to find the best proper analysis in C. The `classify` method returns the highest probability of all dialogue acts given the words in node n using the naive Bayes method. It also returns the corresponding dialogue act which is then stored with the respective node n.

Importantly, the naive Bayes algorithm uses a bag-of-words as its features, taking the product of each word's probability of being in any given dialogue act. This allows the product in line 6 of Algorithm 1 to be used as a ranking score amongst the proper analyses even though the number of nodes n in $\mathbf{t}$ may vary within C. If a different classification algorithm were used, then line 6 may have to be modified to preserve mathematical tractability.

Algorithm 1 Find best utterance segmentations $\mathbf{t}_i \in C$. The `classify` method also returns the best dialogue acts and probabilities which are stored with their nodes n.

```
1:  $max_p \leftarrow 0$ {stores the best probability}
2:  $max_t \leftarrow$ None {best tree node}
3:  for all $\mathbf{t}$ in $C$ do
4:      $p \leftarrow 1$
5:      for all $n$ in $\mathbf{t}$ do
6:          $p \leftarrow p \times$classify(leaves($n$))
7:      end for
8:      if $p > max_p$ then
9:          $max_p \leftarrow p$
10:         $max_t \leftarrow \mathbf{t}$
11:     end if
12: end for
```

4 Evaluation

The segmentation algorithms described in Section 3 were evaluated via 9-fold cross-validation where eight of the chat sessions in our corpus were used for training and one for testing. This process is repeated for all dialogues and the mean result is presented.

In this section, we first discuss why the standard information retrieval evaluation metrics of recall and precision are not appropriate for this type of segmentation, and then discuss the WindowDiff metric, which is used instead.

4.1 Using the Recall and Precision Metrics for Segmentation

The standard information retrieval metrics of recall and precision are not well-suited to evaluating segmentation tasks. Recall is the ratio of correctly hypothesised segment boundaries to the total number of actual boundaries. Precision is the ratio of correct boundaries detected to all hypothesised boundaries.

There are two main problems with using these metrics for segmentation tasks: the first is related to the inherently subjective nature of segmentation. An example is with the message "ok - that's great, thanks" in which "ok - that's great" could be segmented and tagged as a single ACKNOWLEDGEMENT or as the two utterances: "[ok]$^{\text{ACKNOWLEDGEMENT}}$ - [that's great]$^{\text{STATEMENT}}$". Deciding which segmentation should be considered correct depends largely on how the utterances will be used, that is, the downstream task. The traditional recall and precision metrics will regard the alternative segmentation as an error.

Similarly, if our corpus has a message that is manually segmented into two or more adjacent utterances with the same dialogue-act, the system should not necessarily be penalised for regarding the span of text as one segment. For example, "[Goodbye]$^{\text{CONVENTIONAL-CLOSING}}$ and [take care]$^{\text{CONVENTIONAL-CLOSING}}$" could just be marked as one utterance.

The second problem with using recall and precision to evaluate segmentation tasks is the question of how to handle near-boundary misses, that is, a false-positive that occurs near a true boundary. Using recall and precision in the way described will penalise a system equally regardless of whether a hypothesised segment boundary is off by one or ten words.

4.2 The WindowDiff Metric

The manually segmented data is used as a gold standard with which to compare hypothesised segmentations using the WindowDiff metric. The WindowDiff metric, proposed by Pevzner and Hearst (2002), aims to improve segmentation evaluation by rewarding near-misses.

WindowDiff works by choosing a window size k that is typically equal to half of the average segment length in a corpus. This k-sized window then slides over the hypothesised segmentation data and compares segment and non-segment marks with the reference data. If the number of hypothesised and reference segments within the window size differ, a counter is incremented and the window continues to the next position. The final score is then divided by the number of scans performed. A perfect system would therefore receive a zero score.

In most segmentation tasks, segment lengths are uniformly distributed, so using a fixed value for k is appropriate. However, because utterance lengths in our data vary considerably, as shown in Figure 4, we evaluate for different values of k. We adjust k from 1 to 20 for each message, taking the mean result for each value of k. The maximum allowable value of k is the message length on a per-message basis. This technique provides a fair evaluation given the varied utterance lengths.

Another question for our experiment is whether allowing any deviation from our reference segmented data is acceptable, such as inserting a boundary somewhere near an actual boundary. Depending on where a boundary is inserted, this may result in two incomplete utterances as in example (1) below:

(1) a. Ref: [thanks] [you've been very helpful]
    b. Hyp: [thanks you've] [been very helpful]

The segments in (1) differ only by one word, but the resulting utterances in (1-b) are confusing, especially when taken in isolation. In this case, we would not want to allow any deviation from the reference data. However, there are cases where a near-miss is acceptable, such as in (2):

(2) a. Ref: [Thank you for waiting], Customer. [I have found a page that lists a wide variety of Rock Climbing Shoes]
    b. Hyp: [Thank you for waiting, Customer]. [I have found a page that lists a wide variety of Rock Climbing Shoes]

Here, the hypothesised segmentation (2-b) is just as acceptable as the reference (2-a).

The difference between examples (1) and (2) is that (2-b) has maintained the clause boundaries. The word *Customer* in (2-a) is not part of either segment, so including it in the utterance does not affect the adjacent utterance. Since an utterance is a complete phrase, this is the only way a near-miss may still be considered correct. Some other exceptions exist involving single-word utterances which will not be considered here.

5 Experimental Results and Discussion

The WindowDiff results for the various models and window sizes are shown in Figure 5 along with the baseline WindowDiff scores. A lower score indicates higher accuracy. The best result was achieved by the parse tree method. The worst result was given by the HMM POS tag model, but it still exceeded the baseline.

The relative difference between the models varies little as the window size changes. The WindowDiff score begins to taper off as k increases past 20 words, which is at approximately the 90th percentile

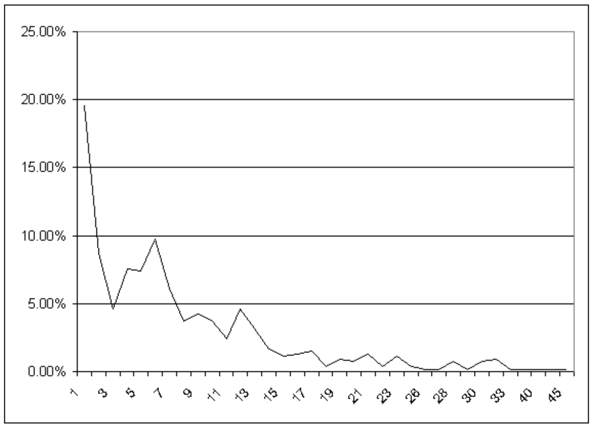

Figure 4: Frequency distribution of utterance length in words. The mean length is 7.6 words and the median is 6 words.

of utterance lengths in our corpus. This plateau is due to window lengths having no effect on shorter messages as a result of the adjustment we make to k when k is greater than the message length.

The better evaluation scores for small values of k are simply due to the way the WindowDiff algorithm compares segments within a window. An equal penalty is applied regardless of whether there are five or two segments within a window that should only contain one. Therefore, a window length spanning the entire message will at most return only one penalty if the hypothesised segments differ at all from the reference segments. Since the window spans the entire message, only one comparison is performed which results in the equivalent of a 100% error rate. Conversely, when k is small, the number of unequal windows between the reference and hypothesised segmentations will also be small since we have so few false positives. At the same time, the number of comparisons will be high, leading to a low WindowDiff score.

A perfect score of 0 is never achieved since there are always some misaligned segments. We never see a score of 1 since many of the single-utterance messages are accurately detected, as discussed in Section 5.1 below. Likewise, none of the models approach the baseline as the window size increases, which indicates that some of the multi-utterance messages are also accurately detected.

Although no individual value of k can be used to judge performance because of the varying segment

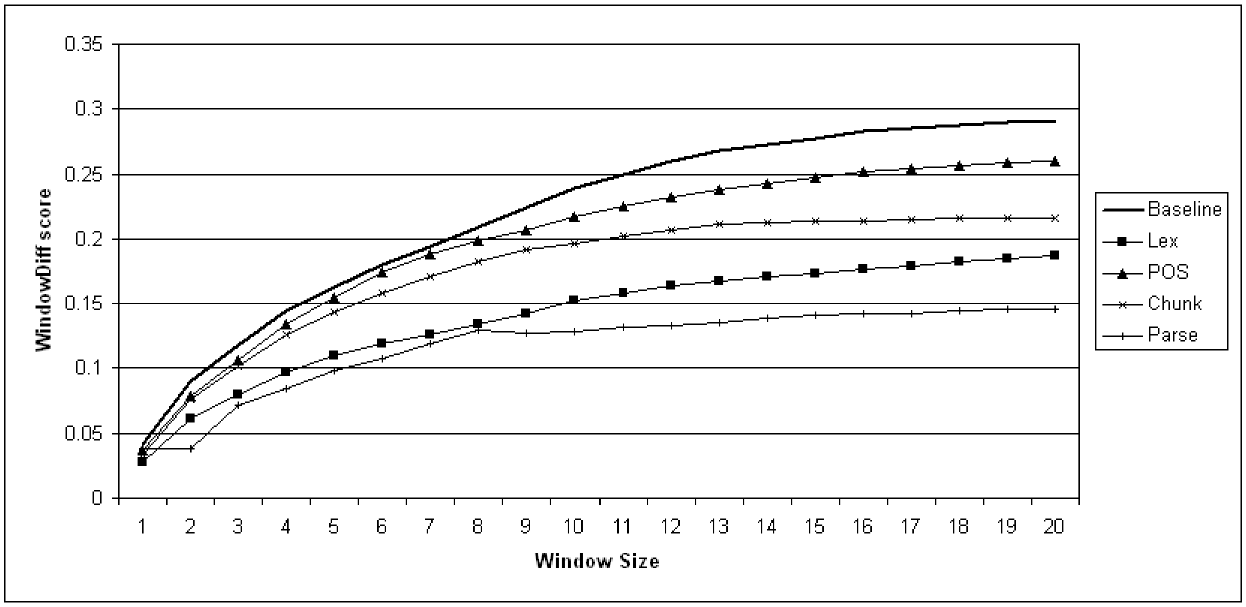

Figure 5: WindowDiff results of various models used and varying window size k from 1 to 20. A lower score indicates better accuracy.

(utterance) lengths in our corpus, we can confidently gauge the performance of each method relative to each other method since their respective rankings remain constant for all values of k.

5.1 Baseline

An analysis of our data revealed that messages contain up to three utterances. Of these messages, 60% contain only one utterance, 20% contain two utterances, and the remaining 20% consist of three utterances.

The baseline is calculated by assuming that each message contains only one utterance since this is the majority class.

5.2 HMM Results

We used three types of features with the HMM: lemmas, POS tags, and the head word of chunked data. The POS tag model performs the worst, whereas the lemma model is the best of the HMM models. This indicates that cue words play a major part in determining utterance segment boundaries. Replacing the words with their respective POS tags loses this information.

Using POS tags can sometimes help overcome data sparseness problems as it has the effect of generalising words. However, in this case it overgeneralises, resulting in poorer performance.

The rationale behind using chunks is that it reduces the number of possible boundaries as we hypothesise boundaries between chunks rather than words. Since utterance boundaries do not lie within chunks, this may have increased the probability of correct segment boundary detection. However, the results show that the HMM benefits from using all words rather than only the chunks' head words.

The main types of errors produced by the HMM are false positives based on words that commonly occur at the start of an utterance, such as "what", occurring mid-sentences as in (3):

(3) but I'm not sure what to get her

The reference data has this as one utterance, but the HMM detects a false positive starting at "what".

5.3 Parse Tree Results

The parse tree method gives the best results. A qualitative evaluation of the dialogue act classifications assigned to detected utterances gave an accuracy of 84%. The baseline for the dialogue act classification task was 36%, which was the majority class being STATEMENT.

The most common type of error the parse tree method makes is to separate words near the root of

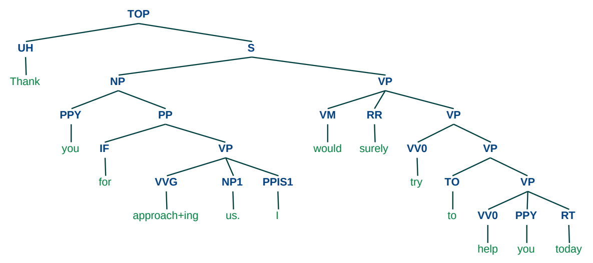

Figure 6: Erroneous parse tree of sentence (4) as produced by RASP.

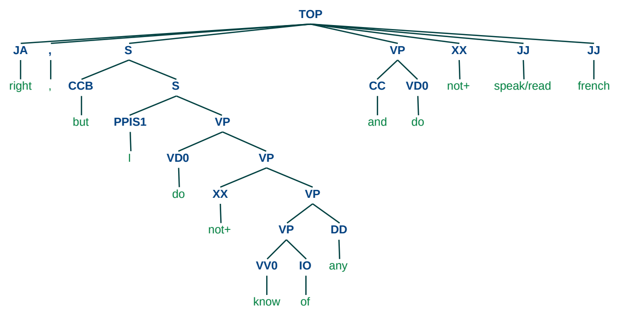

Figure 7: Parse tree of sentence (5) as produced by RASP.

a parse tree away from a deeper right node. Figure 6 shows a parse tree produced by RASP for (4) below:

(4) Thank you for approaching us. I would surely try to help you today

The parse tree for (4) is problematic. The first word, "thank", is detached from the S node that contains the rest of the sentence. Our model treats (4) as a sequence of word tokens $W = w_1, w_2, w_3, ..., w_{13}$ and finds that $P(\text{THANKING}|w_1) \times P(\text{STATEMENT}|W_2^{13}) > P(d|W)$, where d is any dialogue act. In this instance, RASP failed to segment the two sentences in this message, which prevented our model from evaluating the correct utterances. This illustrates the high dependency our model has on the quality of the generated parse trees.

Another type of error is that the model does not detect any segmentations within a message where there ought to be. An instance of this is in (5) below:

(5) right, but I do not know of any and do not speak/read french

The reference data has the word "right" segmented and tagged as RESPONSE-ACK and the rest of the message as one STATEMENT. However, our model does not evaluate that possibility as the corresponding parse tree in Figure 7 does not combine the words as would be expected.

6 Conclusion and Future Work

Finding utterance boundaries in IM dialogue is a critical step for aiding utterance classification and downstream language processing modules such as dialogue response planning. We have shown that the parse trees model obtains the best results. Of the HMM models, the HMM over lemmas in messages performs better than using chunked data and POS-tags, which lose too much information and impede accuracy.

The parse tree method performed best overall and has the advantage of combining both segmentation and classification tasks in one step to give the optimal combined result. It is based on the linguistic intuition that utterances are complete constituents, which are modelled well by parse trees. However, this heavy reliance on the quality of the parse trees is also a weakness. Most of the errors obtained using the parse tree method may be attributed to poor, or at least unexpected, parse trees being produced. That notwithstanding, the preliminary results using the RASP parser are very encouraging.

In future work, we intend to focus more on parsing IM messages, taking into account some of its distinct characteristics. Some obvious steps to produce better parse trees are to perform spelling corrections and expand acronyms, such as "idk" for "I don't know". Existing parsers will thus be able to produce more accurate parse trees, which will in turn result in higher segmentation accuracy.

We will also investigate the subjectiveness of utterance segmentation by performing Kappa (Siegel and Castellan, 1988) analysis on our segmentation boundaries. The Kappa analysis will give an indication as to the meaningful upper bounds of the performance of our system.

Acknowledgments

Thanks to Steven Bird, Timothy Baldwin, and the Language Technology Group at Melbourne University for their constructive comments. Thanks also to Trevor Cohn, Phil Blunsom, and the anonymous reviewers for their very helpful feedback. The data used in this study was POS tagged, chunked, and parsed by Timothy Baldwin using the tools described with some modifications.

References

John L. Austin. 1962. *How to do Things with Words*. Clarendon Press, Oxford.

Eric Brill. 1995. Transformation-based error-driven learning and natural language processing: a case study in part-of-speech tagging. *Computational Linguistics*, 21(4):543–565.

Ted Briscoe and John Carroll. 2002. Robust accurate statistical annotation of general text. In *Proceedings of the Third International Conference on Language Resources and Evaluation*, pages 1499–1504, Las Palmas, Gran Canaria.

Jean Carletta. 1996. Assessing agreement on classification tasks: the kappa statistic. *Computational Linguistics*, 22(2):249–254.

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of the first conference on North American chapter of the Association for Computational Linguistics*, pages 132–139, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Noam Chomsky. 1965. *Aspects of the Theory of Syntax*. MIT Press, Cambridge, MA.

Michael John Collins. 1999. *Head-driven statistical models for natural language parsing*. Ph.D. thesis, University of Pennsylvania, Philadelphia. Supervisor-Mitchell P. Marcus.

Mark Core and James Allen. 1997. Coding dialogs with the DAMSL annotation scheme. *Working Notes of the AAAI Fall Symposium on Communicative Action in Humans and Machines*, pages 28–35.

Edward Ivanovic. 2005. Dialogue act tagging for instant messaging chat sessions. In *Proceedings of the ACL Student Research Workshop*, pages 79–84, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Slava M. Katz. 1987. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 35(3):400–401.

Lori Levin, Klaus Ries, Ann Thyme-Gobbel, and Alon Lavie. 1999. Tagging of speech acts and dialogue games in Spanish call home. *Towards Standards and Tools for Discourse Tagging (Proceedings of the ACL Workshop at ACL'99)*, pages 42–47.

Guido Minnen, John Carroll, and Darren Pearce. 2001. Applied morphological processing of english. *Natural Language Engineering*, 7(3):207–223.

Grace Ngai and Radu Florian. 2001. Transformation-based learning in the fast lane. In *Proceedings of NAACL-2001*, pages 40–47.

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

Lance Ramshaw and Mitch Marcus. 1995. Text chunking using transformation-based learning. In David Yarovsky and Kenneth Church, editors, *Proceedings of the Third Workshop on Very Large Corpora*, pages 82–94, Somerset, New Jersey. Association for Computational Linguistics.

John R. Searle. 1979. *Expression and Meaning: Studies in the Theory of Speech Acts*. Cambridge University Press, Cambridge, UK.

Sidney Siegel and N. John Castellan, Jr. 1988. *Nonparametric statistics for the behavioral sciences*. McGraw-Hill, second edition.

Andreas Stolcke and Elizabeth Shriberg. 1996. Automatic linguistic segmentation of conversational speech. In *Proceedings, ICSLP 96. Fourth International Conference on Spoken Language*, volume 2, pages 1005–1008, Philadelphia, PA, Oct. ICSLP.

Andreas Stolcke, Noah Coccaro, Rebecca Bates, Paul Taylor, Carol Van Ess-Dykema, Klaus Ries, Elizabeth Shriberg, Daniel Jurafsky, Rachel Martin, and Marie Meteer. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational Linguistics*, 26(3):339–373.

Hideki Tanaka and Akio Yokoo. 1999. An efficient statistical speech act type tagging system for speech translation systems. In *Proceedings of the 37th conference on Association for Computational Linguistics*, pages 381–388. Association for Computational Linguistics.