A Functional Approach to Speech Recognition Evaluation

Ben Hutchinson

Syrinx Speech Systems
ben.hutchinson@syrinx.com.au

Abstract
The paper describes a new evaluation measure for speech recognition in spoken language dialogue systems. The measure is based on the usefulness of the recognition for the system, and the usefulness is measured at the level of meaning representation. It is argued that the new measure is more useful than word error rate, and is more accurate than simpler functional measures.

1. Introduction

The most common evaluation measure of speech recognition used in the literature has been Word Error Rate (WER). While this may be a good measure of the usefulness of the recogniser in voice-enabled language applications, it does not directly measure the usefulness of the recogniser for a spoken language dialogue system (SLDS). When the speech recogniser is coupled with an analysis component to build a speech understanding system, a more meaningful measure of recogniser usefulness is how often the correct meaning is returned by the speech understanding system. Taking attribute value matrices (AVM) as a standard representation of meaning, this paper proposes an evaluation of speech recognition based on the comparison of meanings. In section 2, a range of simple evaluation methods are discussed. Functional approaches to parser evaluation are discussed in section 3, and their suitability for speech recognition evaluation is considered in section 4. The SPARKLE evaluation measure is modified in section 5 to make it suitable for speech recognition evaluation. In section 6 an example application is discussed.

2. Discussion of some evaluation measures for speech recognition accuracy

In this section I discuss some simple measures that have been, or could be, used to evaluate speech recognition performance. It is useful to first consider what the role of speech recognition in a spoken dialogue system is. The speech recogniser receives input from the user and returns results which influence the actions of the system. Misrecognitions are harmful because they may either cause the system to do something the user did not want, or lengthen the dialogue because the user has to correct the misrecognition in later dialogue turns. It therefore makes sense to speak of the usefulness of a recognition result being the degree to which it facilitates correct system actions and fewer dialogue turns. The usefulness of a recognition result is intrinsically connected with whether the system understands the user. I shall use the term meaning in this paper in a very narrow, application dependent sense. Basically, two utterances, or words, will be said to have the same meaning if and only if they have identical effects on the system.

2.1. Word Error Rate (WER)

The easiest way to evaluate the performance of a recogniser is to measure how many words were correctly recognised. However in a natural language system there are several reasons why WER is an inaccurate measure of the usefulness of a recognition result. Firstly, it is not always clear what counts as a word. Speech recognition tokens do not always correspond in a one-to-one fashion with orthographic words, and punctuation and whitespace may or may not correspond to word boundaries [10]. Depending on how the recogniser treats word boundaries, recognition may be more accurate if certain word sequences, for example I'd like, are treated as single recognition words/tokens. Secondly, inflections of may not affect the meaning of a word (for example, tense or aspect may not be included in the meaning representation for a particular application), and neither may clitics (the Olympic system [3] treats women's and women as synonymous). Thirdly, different words have different information content, as can clearly be seen by the common use of word or phrase spotting, or of partial parsing techniques in spoken language dialogue systems. However, for certain voice-enabled applications, where the user does not use natural language, WER may be an accurate measure of the usefulness of a recognition result.

2.2. WER of Information Words

When the language understanding component does not perform a full structural analysis of the utterance, many, if not most, words of the utterance may be ignored. It is therefore possible to just measure the WER of words which contain information that the system uses. While I am not aware of this measure actually being used to evaluate SLDSs, it is worth considering as an alternative to standard WER since it addresses the issue of different information content.

2.3. Word Meaning Error Rate

The problem of whether inflections and clitics should be ignored or not can be overcome by grouping words into equivalence classes based on their information content. Two words are in the same class if they are treated identically by the system. This marks an important step in moving from an evaluation based on words to one based on (system) meaning. However since meaning is being evaluated on a word-by-word basis, problems of tokenisation are not overcome. Nevertheless, for systems where tokenisation is clear cut and unchanging during the lifetime of the system, this may be a useful evaluation measure.

1 This measure has however been used to evaluate machine interpretation systems [6].


2.4. Language understanding

For speech recognition systems handling natural language, a comparison of final semantic analyses is a better way to measure the usefulness of a word sequence returned by the recogniser. Under this approach, the recognition results are syntactically and semantically analysed before the evaluation is done. This approach has been described as evaluating speech understanding [1]. If we can expect the syntactic-semantic analysis to have a 0% error rate, then any errors in the meaning can be attributed to misrecognitions. What is being measured, therefore, is how well the recogniser is performing its function of producing utterances for analysis.

But how reasonable is it to expect an analysis error rate of 0%? According to [1], for a SLDS the use of robust analysis techniques make this goal possible. The range of possible utterances that must be understood is very restricted, and so grammars for analysis can be much smaller and more robust than those used for parsing texts from an open domain such as newspapers. Furthermore, it can be ensured that the grammars for analysis will work on all utterances returned by the recogniser by using the same grammars for recognition and analysis (e.g. the Nuance grammar format), or deriving recognition and analysis grammars from a common source [4]. It has been noted that when the accuracy of the understanding component is so high: “it does not make sense to evaluate it on its own, but only in connection with a recogniser” [1]. My claim here is that in such a situation the evaluation of the combined recogniser and analyser is fact equivalent to evaluating the usefulness of the recogniser.

We take two propositions as given for the remainder of this paper: 1) we want to evaluate the understanding of the system by comparing representations of meaning; and 2) crucial to the understanding is the relation between concepts.

3. Functional evaluation of parsing

In the last few years several parser evaluation measures have been proposed based on functional or grammatical relations (GR) ([9], FAME [2], SPARKLE [7] and [8]). These measures were proposed to overcome the inaccuracies of previous measures which attempted to compare grammatical tree structures. The function based approaches reduce a parse to a set of GRs, which are treated as the essential and fundamental components of a grammatical analysis. (Not all parsers return sets of grammatical relations, and [9] addresses this by providing an algorithm for producing a set of dependency relations from an arbitrary parse tree.) The accuracy of a parse is determined by comparing the set of relations it induces with the correct set. Where these approaches have differed from each other is primarily in the types of GRs they use. [9] and FAME use sets of relations motivated by dependency grammar and LFG, respectively. SPARKLE modularizes the representation of analyses by having a smaller set of functional relations (FAME) represented separately from morpho-syntactic features. This modularization is motivated by an aim of theory neutrality (a goal the authors concede is “almost unattainable”).

A result of reducing a grammatical description to a set of simple grammatical relations is that partially correct analyses are penalized in proportion to the amount of correct information they contain. For example, if a PP is internally correct, but is incorrectly attached, then it still receives a positive score for its internal correctness. For example, consider the following sentence and two possible AVMs for representing the grammatical relations.

1) Jane saw the man with black hair
2) [PRED see ] [SUBJ [PRED Jane ]] [OBJ [PRED man ]] [WITH [PRED hair ]] [MOD [PRED black]]

A comparison of the above two AVMs from the outermost matrix in would fail to capture the fact that the modification of hair by black is identical in both cases. This difficulty is overcome by reducing each of (2) and (3) AVM to a set of GRs between heads and dependents, shown in (4) and (5), respectively.

4) Subj(see,jane) Obj(see,man) Mod(man,hair,with)
5) Subj(see,jane) Obj(see,man) Mod(see,hair,with)

Comparing (4) and (5) shows that hair has been modified identically in both parses, and the evaluation assigns both precision and recall scores equal to 75%.

4. Assessing functional evaluation for speech recognition

[2] and [11] identify two uses for parser evaluation: monitoring the development of a particular grammar and parsing system; and comparing different grammars or parsing systems. I now consider the usefulness of functional evaluation for a different task than those for which it has previously been proposed: evaluating the performance of a speech recogniser which feeds utterances to the analysis component. The analyser returns a representation of the utterance’s meaning, which we assume to be in the form of an AVM. These AVMs can be compared using the functional evaluation techniques introduced above. Note that the AVMs representing meaning are indistinguishable from the grammatical function AVMs introduced in the previous section. That is, the semantics of utterances is directly reflected in the grammatical functions.

The first aspect of functional evaluation to note is the lack of a GR expressing the fact that a value is the head of the outermost AVM. (SPARKLE and FAME do not express

\[ \text{A comparison of the above two AVMs from the outermost matrix in would fail to capture the fact that the modification of hair by black is identical in both cases. This difficulty is overcome by reducing each of (2) and (3) AVM to a set of GRs between heads and dependents}, \]

\[ \text{shown in (4) and (5), respectively.} \]

\[ \text{Comparing (4) and (5) shows that hair has been modified identically in both parses, and the evaluation assigns both precision and recall scores equal to 75%.} \]

\[ \text{4. Assessing functional evaluation for speech recognition} \]

\[ \text{[2] and [11] identify two uses for parser evaluation: monitoring the development of a particular grammar and parsing system; and comparing different grammars or parsing systems. I now consider the usefulness of functional evaluation for a different task than those for which it has previously been proposed: evaluating the performance of a speech recogniser which feeds utterances to the analysis component. The analyser returns a representation of the utterance’s meaning, which we assume to be in the form of an AVM. These AVMs can be compared using the functional evaluation techniques introduced above. Note that the AVMs representing meaning are indistinguishable from the grammatical function AVMs introduced in the previous section. That is, the semantics of utterances is directly reflected in the grammatical functions.} \]

\[ \text{The first aspect of functional evaluation to note is the lack of a GR expressing the fact that a value is the head of the outermost AVM. (SPARKLE and FAME do not express} \]

\[ \text{The following SPARKLE relations are used in this paper: Subj(head,dependent), Obj(head, dependent) and Mod(head,dependent), representing subject, object and modifier relations. The modifier relation can optionally contain a feature specifying a preposition which introduces the modification.} \]

\[ \text{2 The following SPARKLE relations are used in this paper: Subj(head,dependent), Obj(head, dependent) and Mod(head,dependent), representing subject, object and modifier relations. The modifier relation can optionally contain a feature specifying a preposition which introduces the modification.} \]
this, however the measure in [9] does.) For example, (8) is the only GR produced from (6). When the recognition is partially correct, the GR evaluation will therefore underestimate the result. For example, suppose the speaker says (6) with correct AVM (7) and GR relation (8), but the recogniser returns (9), producing AVM (10) and the single GR relation (11).

6) A supreme with olives
7) [PRED supreme ]
   [WITH olives ]
8) Mod(supreme, olives,<intro=with>)
9) A supreme with anchovies
10) [PRED supreme ]
    [WITH anchovies]
11) Mod(supreme, anchovies,<intro=with>)

Now (11) is different from (8), so the precision and recall scores would be 0. However, the AVMs have one identical AV, so a score of 0 seems unfairly harsh. We can see that (8) has some utility to the dialogue controller if we consider the imaginary dialogue fragment (12).

12) System: Welcome to the pizzeria. What kind of pizza would you like?
   User: A supreme with olives.
   System: Would you like a supreme with anchovies?
   User: No, a supreme.
   System: You’d like a supreme with olives, right?
   User: Yes.

In (12) the user has only had to correct one piece of information, and so in spite of the misrecognition the turn has proved to have some use. Compare this with the hypothetical dialogue in (13), where a complete misrecognition forces the user to supply more information, with all the encumbent risks of further misrecognitions.

13) System: Welcome to the pizzeria. What kind of pizza would you like?
   User: A supreme with olives.
   System: Would you like a hawaiian with anchovies?
   User: No, a supreme with olives.
   System: You’d like a supreme with anchovies, right?
   User: No, with olives.

The second shortcoming of the GR evaluation is that it does not reward partially correct analyses when the dependent is correct but the head is not. For example, if a supreme with olives is misrecognised as a marinara with olives, then only relation (14) is produced, which is correct except for the head.

14) Mod(marinara, olives,<intro=with>)

The usefulness of the information in the incorrect recognition result is demonstrated by dialogue (5).

15) System: Welcome to the pizzeria. What kind of pizza would you like?
   User: A supreme with olives.
   System: Would you like a marinara with olives?
   User: No, a supreme.
   System: So you’d like a supreme with olives, right?
   User: That’s right.

It is also interesting to analyse the usefulness of a recognition result when dependents are incorrectly misrecognised, for example if only the preposition of a PP is correct. Consider the misrecognition (12) and compare its GR (in SPARKLE notation) (13) with that of the correct utterance (6).

16) A supreme without olives
17) Mod(without, supreme, olives)

The usefulness of the misrecognition is demonstrated by the fact that it is possible to correct the object of a preposition without restating the preposition.

18) System: Welcome to the pizzeria. What kind of pizza would you like?
   User: A supreme with olives.
   System: Would you like a supreme with anchovies?
   User: No, olives.

It is useful to compare this with the case where the preposition’s object is correctly recognised, but the preposition itself is not. While one may be able to just restate the preposition, it is more natural to restate the whole PP.

19) System: Welcome to the pizzeria. What kind of pizza would you like?
   User: A supreme with olives.
   System: Would you like a supreme without olives?
   User: No, with olives.  

The difference between (18) and (19) is no doubt related to the fact that a preposition cannot form a phrase on its own, while a noun can. Since it is more difficult to recognise a PP than a noun, it is more useful if only the noun is misrecognised than if only the preposition is. So if we are trying to recognise a supreme with olives, then (20) shows the usefulness of some possible recognition results.

20) A supreme with olives > A supreme with anchovies > A supreme without olives > A supreme without anchovies

The grammatical function evaluation used for parser evaluation fails to uphold this ranking of misrecognitions: it ranks all three of the misrecognitions in (20) as equally bad.

5. Comparing AVMs for speech recognition evaluation

It was optimistic to suppose that an evaluation method could be used for a new task without modification. Whereas parser evaluation is performed on a fixed text, speech recognition evaluation necessarily involves variation in the words in each utterance. If some words are recognised correctly, then even though the relations between them may be wrong their recognition may still have some value for the system. The evaluation measure needs to reflect partial success. This section presents a modification of the SPARKLE notation and evaluation that achieves this purpose. The modification consists of two changes.

A. The head of an utterance phrase should be denoted as such in a separate relation

As discussed above in relation to (6-11), if an AVM contains the correct head but incorrect dependent, it gets a score of zero under SPARKLE, even though some useful information is present. We therefore follow [9] in including the fact that a concept is not dependent on any other in the evaluation, using the notation:

21) Dep(NULL,dependent)

For example, utterances (6) and (9), reproduced as (22) and (24), would now have the sets of dependencies shown in (23) and (25).

22) A supreme with olives
23) Dep(NULL, supreme)

Mod(supreme, olives,<intro=with>)

24) A supreme with anchovies
25) Dep(NULL, supreme)

Mod(supreme, anchovies,<intro=with>)
6. Results

The new evaluation measure has been implemented and used to evaluate the performance of the speech recogniser when changes were made so that it catered for the possibility of filled pauses (e.g. um and ah) [5]. In the experiments, filled pauses are included in the dictionary despite not being words, which could cause problems for a simple implementation of the WER measure. Table 1 shows relations assigned to some misrecognitions by the new measure. Table 2 shows the evaluation scores assigned by word error rate (filled pauses are not included here), SPARKLE, and the new measure.

Table 1: Misrecognitions of I want a supreme with olives and the relations they produce

<table>
<thead>
<tr>
<th>Utterance</th>
<th>Relations</th>
</tr>
</thead>
<tbody>
<tr>
<td>We want the supreme with the olives.</td>
<td>Dep(NULL, supreme) Mod(supreme, olives.&lt;intro=with&gt;)</td>
</tr>
<tr>
<td>I want a supreme with ham.</td>
<td>Dep(NULL, supreme) Mod(supreme, ham.&lt;intro=with&gt;)</td>
</tr>
<tr>
<td>We want a supreme without the olives.</td>
<td>Dep(NULL, supreme) Mod(supreme, olives.&lt;intro=without&gt;)</td>
</tr>
<tr>
<td>I want a marinara with um olives</td>
<td>Dep(NULL, marinara) Mod(marinara, olives.&lt;intro=with&gt;)</td>
</tr>
<tr>
<td>I want a marinara without olives</td>
<td>Dep(NULL, marinara) Mod(marinara, olives.&lt;intro=without&gt;)</td>
</tr>
<tr>
<td>I’d like a marinara with ham</td>
<td>Dep(NULL, marinara) Mod(marinara, ham.&lt;intro=with&gt;)</td>
</tr>
</tbody>
</table>

Table 2: (Precision, Recall) scores for the utterances in Table 1, with which rows correspond respectively.

<table>
<thead>
<tr>
<th>Utterance</th>
<th>Relations</th>
</tr>
</thead>
<tbody>
<tr>
<td>We want the supreme with the olives.</td>
<td>Dep(NULL, supreme) Mod(supreme, olives.&lt;intro=with&gt;)</td>
</tr>
<tr>
<td>I want a supreme with ham.</td>
<td>Dep(NULL, supreme) Mod(supreme, ham.&lt;intro=with&gt;)</td>
</tr>
<tr>
<td>We want a supreme without the olives.</td>
<td>Dep(NULL, supreme) Mod(supreme, olives.&lt;intro=without&gt;)</td>
</tr>
<tr>
<td>I want a marinara with um olives</td>
<td>Dep(NULL, marinara) Mod(marinara, olives.&lt;intro=with&gt;)</td>
</tr>
<tr>
<td>I want a marinara without olives</td>
<td>Dep(NULL, marinara) Mod(marinara, olives.&lt;intro=without&gt;)</td>
</tr>
<tr>
<td>I’d like a marinara with ham</td>
<td>Dep(NULL, marinara) Mod(marinara, ham.&lt;intro=with&gt;)</td>
</tr>
</tbody>
</table>

7. Conclusion

A new evaluation measure has been presented based on the usefulness of meaning representations for the rest of the system. This usefulness is measured by a metric on meaning representations that is similar to existing metrics for evaluating parsers but is more finely grained, allowing a more accurate measure of usefulness. The evaluation measure can be used to either compare the performance of different spoken language systems, or to monitor the development of a single system.

8. Acknowledgments

I would like to thank my colleagues Dominique Estival and Cécile Pereira for their assistance and valuable discussion.

9. References