**N.b.: A graphical user interface for annotating spoken dialogue**

Giovanni Flammia and Victor Zue  
Spoken Language Systems Group  
Laboratory for Computer Science  
Massachusetts Institute of Technology  
Cambridge, Massachusetts 02139 USA

**Abstract**

Corpora of transcribed and annotated dialogues are very useful for developing and evaluating the coverage of algorithms for discourse generation and interpretation and dialogue modelling. On the other hand, there is no agreement on the choice of units and conventions for annotating discourse constituents, and the annotation process can be difficult and prone to inconsistencies.

This paper presents *N.b.*, a graphical user interface for annotating the discourse structure of spoken dialogue. Different annotation instructions and different theories about discourse interpretation and generation can be easily incorporated in the annotation process without the need of changing the graphical user interface. The instructions and the annotated text are displayed in a clear-cut way, and typing is reduced to a minimum. We describe how to use *N.b.* for annotating embedded discourse segments and the system's end-to-end performance in a transcribed dialogue.

**Introduction**

Research and development of human language technology typically relies on the availability of suitably collected corpora. Such corpora can help researchers understand the regularity/variability of the linguistic phenomena under investigation, propose computational models to mimic their behavior, estimate the parameters of the models, and evaluate the effectiveness of either the models or systems that embed these models (Goodine et al. 92, Silverman et al. 92, Hirshman et al. 93). To develop phonetic recognition algorithms, for example, researchers in the US have relied on the TIMIT corpus (Lamel et al. 86) to understand the acoustic realizations of phonemes under varying phonetic environments, thereby developing phonetic models to capture such contextual variations (Lee and Hon 89, Goldenthal and Glass 94).

For a corpus to be truly useful, it must be properly annotated. Corpus annotation involves defining the inventory of constituent units (phonemes, syntactic categories, semantic categories, etc.), together with a set of annotation conventions. For example, at the syntactic level, pronouns might be annotated along with the noun phrases they refer to (Walker 89) and at the semantic level, phrases might be annotated with their meaning according to formal models like logical forms or semantic frames (Allen 94).

Annotation of low-level linguistic phenomena, such as phonetic variants and disfluencies, are relatively straightforward, since agreement on the choices of units and conventions can often be reached (Lee and Hon 89, Silverman et al. 92). As a result, the task of annotation can often be shared across sites, and the aggregate corpora are larger and more useful to a wider community. As we move up the linguistic chain, however, the picture can rapidly deteriorate. In most cases, the controversy stems from the fact that the choice of units and conventions are often tied to linguistic theories that are not universally subscribed. Therefore, corpora annotated by one site may not be useful to researchers from other sites, leading to duplication of effort and inhibiting cross-system comparisons. One approach to dealing with this problem is to provide a set of minimal, theory-neutral annotations. The Penn Treebank (Marcus et al. 93) is an excellent example of linguistic data annotated using this approach. Syntactic structure of sentences is implicitly described by bracketing major constituents without actually attaching labels to them. While there were some initial doubts regarding the ultimate utility of such an annotation scheme, they were largely put to rest once researchers had a chance to make use of the corpus. The Penn Treebank has been instrumental in facilitating the comparison of several general English parsers (Black 93).

Over the past few years, our group has been involved with research and development of interactive conversational systems, i.e., systems that can understand and respond to verbal input. We have become increasingly aware of the importance of discourse and dialogue modelling, and the need for properly annotated data to aid research in this area. To uncover the discourse structure of a dialogue, utterances might be classified into speech acts. Sequences of utterances might be bracketed into a multi-level structure of embedded seg-
<table>
<thead>
<tr>
<th>Segment</th>
<th>Action</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Open</td>
<td>System: okay.</td>
</tr>
<tr>
<td>2</td>
<td>User:</td>
<td>I’d like to fly from Boston to Pittsburgh.</td>
</tr>
<tr>
<td>3</td>
<td>System:</td>
<td>What date will you be travelling on?</td>
</tr>
<tr>
<td>4</td>
<td>User:</td>
<td>October twelfth.</td>
</tr>
<tr>
<td>5</td>
<td>System:</td>
<td>What time will you be leaving.</td>
</tr>
<tr>
<td>6</td>
<td>User:</td>
<td>After 5:00 p.m.</td>
</tr>
<tr>
<td>7</td>
<td>System:</td>
<td>The earliest flight leaving after 5:00 P.M. is US Air Flight 128. It leaves at 5:10 P.M. and arrives at 7:00 P.M.</td>
</tr>
<tr>
<td>8</td>
<td>System:</td>
<td>The earliest flight leaving after 5:00 P.M. is US Air Flight 127. It leaves at 5:10 P.M. and arrives at 7:15 P.M.</td>
</tr>
<tr>
<td>9</td>
<td>User:</td>
<td>How about after 6 o’clock.</td>
</tr>
<tr>
<td>10</td>
<td>System:</td>
<td>United flight 581 and 6177 leaves at 6:20 A.M., connects in Washington and arrives at 10:00 A.M.</td>
</tr>
<tr>
<td>11</td>
<td>User:</td>
<td>I mean six p.m.</td>
</tr>
<tr>
<td>12</td>
<td>System:</td>
<td>American Flight 463 and 1104 leaves at 6:00 P.M., connects in Chicago and arrives at 10:18 P.M.</td>
</tr>
<tr>
<td>13</td>
<td>User:</td>
<td>Are there any other flights leaving after six p.m.</td>
</tr>
</tbody>
</table>

Figure 1: *N.b.* main window at the beginning of a discourse annotation session. The top left area displays annotation instructions and choices for the current sentence (sentence 7) and the main text panel (bottom right) displays a window of the text to be annotated. The current sentence is displayed in a different color in the center of the text panel.

...ments, and each segment might be annotated with its purpose with respect to the entire discourse (Grosz and Sidner 86, Mann and Thompson 88, Cohen et al. 90, Moore and Pollack 92, Hovy 93, Allen 94). Uncovering the discourse structure is a difficult task to accomplish and to evaluate (Rotondo 84). Recent work on human discourse annotation reliability (Grosz and Hirshberg 92, Passoneau and Litman 93) and discourse segmentation algorithms (Morris and Hirst 91, Hearst 94) has inspired us to investigate general discourse annotation methods and evaluation tools.

Following the example set forth by the Penn Treebank project, we have decided to develop a minimal and theory neutral annotation method. We believe that the development of efficient annotation tools constitutes an important aspect of this research. In a typical annotation session, a subject is presented with some instructions and with the text to be annotated, and is expected to produce an annotated text according to a specified format. This process can be difficult, time consuming and prone to inconsistencies, unless the subject is provided with an appropriate interface for entering the annotation. A good set of tools can greatly facilitate the annotation process, both in throughput, accuracy, and consistency, thereby leading to useful data that can serve the needs of the research community.

We think that the ideal interface should be ergonomic and portable. By ergonomical we mean that it should present the annotation instructions and the annotated text in a clear-cut way, so that subjects will be able to produce a consistent annotation without difficulty. By portable we mean that the interface should be general enough to accommodate a large variety of annotating instructions and linguistic theories, so that changes in the instruction formats and different experimental conditions can be incorporated easily, possibly without modifying the internal structure of the interface software.

In the following sections, we describe *N.b.* (*Nota Bene*¹), a dialogue annotation tool that we are developing with the goal of meeting the two requirements of ergonomicity and portability. We will describe its functions, provide some examples of how it can be used, and summarize future development plans.

---

¹ *N.b.* stands for the Latin *Nota Bene* which means *to annotate well.*
Overview of \textit{N.b.}

\textit{N.b.} is a graphical user interface for editing annotations of text and transcribed speech according to instruction sequences specified in program arguments. In this section we give an overview of the user interface and in the following sections we describe how the program works and how to specify annotation instructions by way of two simple examples.

Figure 1 shows the main window in the current implementation after loading a log file with the transcription of an actual human-computer travel planning dialogue using the PEGASUS spoken language system (Zue \textit{et al.} 94). The window presents to the user the information necessary for annotating the discourse segments of the dialogue. The main window is divided into two different areas. The annotation area for each sentence is located at the top left and the main text panel displaying the annotated text is located in the bottom right.

Annotation Area

The top left rectangular area displays the annotation instructions and annotation choices available for each sentence. A message displays brief instructions for annotating the current sentence. A text panel displays the available choices for annotating the current sentence. The annotator can grab text from this text panel to build annotations. In the figure, the current sentence is number 71, the current field is named \textit{Now}, and the annotator has the options of opening a new segment, closing a segment, or linking two segments. Annotation instructions, fields, and messages are all program arguments specified in a configuration file.

Annotate Buttons

After the annotator has entered a value, she or he has the options of deleting the last annotation step (Undo), clearing the annotation field (Clear), accepting the annotation and proceed according to instructions (Accept), or scroll to the next sentence (\text{->}).

Check Buttons

When clicked, these buttons call functions that help in evaluating the consistency of the annotation process. The functions being computed depend on the annotation instructions and are specified as external programs. In the case of discourse segmentation, the Check All button shows a popup window with the annotated text \textit{pretty-printed} with each segment assigned...
to a different color and an appropriate level of indentation. The Check One button highlights one discourse segment at the time, and the Check Entry button displays all the annotations entered for the current sentence and the list of segments.

**Scroll Buttons**

Nine scroll buttons can center the main text panel to the first sentence (Begin), to the last one (End), to the currently annotated sentence (Here), to the preceding (<-- or next sentence (-->), to the preceding page (<---) or the next page (--->) or to specific annotated end-points (|| < - and - > ||)), The annotator can scroll back and forth through the entire text and does not need to annotate the sentences in any particular order.

**Main Text Panel**

The main text panel displays a window of the annotated text. The currently annotated sentence (sentence 7 in Figure 1) is highlighted in a different color in the center of the text panel. The annotator can grab words and phrases from this text panel with the mouse to build annotation strings.

**File Buttons**

These buttons are used for loading text files and annotated text files, saving annotated text files, and quitting the program.

**Input Buttons**

Input Text shows a large popup window with the entire input text file. Input Signal runs an external program for viewing the speech waveform corresponding to the current sentence, and Input Play plays back the speech waveform corresponding to the current sentence. Playback and editing commands are specified as external programs in a configuration file.

**Sentence Buttons**

These buttons can be used for editing the text. Sentence Break is used for breaking the sentence into two or more separate units for annotating them separately, and Sentence Merge is used for merging two sentences into one single unit to be annotated.

**Using \textit{N.b.}**

In the following, we illustrate \textit{N.b.} works and how to write annotating instruction sequences by way of two simple examples. We annotate an actual human-machine dialogue using \textsc{Pegasus}. This is a mixed initiative system in which the machine may prompt the user for specific missing information in order to specify the user’s request, and misunderstandings may occur because of speech recognition errors or semantic analysis failures. In the first example, the task is to annotate the discourse segments. In the second example, the task is to evaluate the system’s performance in the dialogue, proceeding utterance by utterance.

**Annotating Discourse Segments**

For each sentence, the initial choices are either \texttt{Link}, \texttt{Open_Segment}, \texttt{Close_Segment}, or \texttt{Evaluate Dialogue}. This last field is used for answering some questions about global properties of the dialogue, such as what are the customer and the agent goals and whether or not the dialogue has been successful. The annotator can bracket the text with embedded segments (see Figure 2). Segment boundaries are annotated in the text with the words \texttt{(Open_Segment Name)} and \texttt{(Close_Segment Name)}. Names can specify discourse segment purposes or intentions and provide easily memorized identifiers for each segment. They can either correspond to specific keywords displayed in the text panels, or they may be typed in. The entire text is indented automatically according to the proposed segmentation. Warning messages are issued when crossing brackets violate the balanced structure of the segment sequence. If the annotator wishes to model the segments with a graph structure rather than with a tree structure, she or he can link two or more non-overlapping segments using the command \texttt{Link}, at which point \textit{N.b.} prompts for selecting the names of the segments to be linked. The commands \texttt{Open_Segment} and \texttt{Close_Segment} can be used to move existing segment end-points in the annotated text. The Undo button can be used at each sentence to delete annotations and backtrack to a previous annotation step. For example, undoing \texttt{(Open_Segment Name)} effectively deletes a segment from the annotated text.

In Figure 2, the annotator has chosen \texttt{Close_Segment} for sentence 5. Then, the program displays a list of existing segments in the Available Choices text panel. When the annotator grabs the words \textit{Request Info from boston to pittsburgh} with the mouse, the choice is displayed in the annotation text field, the segment is closed, and the annotated text is automatically indented and displayed again.

The annotation instructions for opening and closing segments are specified in a configuration text file. The instructions are illustrated at the top of Figure 4 at the end of this paper. The instructions specify the name and the possible values for each field to be annotated and in which order the fields should be filled out. For each field, we specify its name, what to display in the Available Choices text panel, what field has to be annotated after each one of the choices has been made, a default next field, and the instruction message to be displayed. The statement \texttt{Display Field} specify to display all the values that have been entered for \texttt{Field} up to the current sentence. The statement \texttt{Display Open_Segment} specifies to display all the segments annotated between the first sentence and the current sentence.
Evaluating A Spoken Language System

In this example, we use N.b. for annotating each sentence separately with specific sequences of key words. The task is to evaluate the end-to-end performance of the PEGASUS system for the transcribed dialogue. The main window at the end of the annotating session is displayed in Figure 3, and the bottom of figure 4 displays the annotation instructions for evaluating the system’s performance.

When evaluating the system’s utterance, first the annotator has to classify it according to some types such as **Question** or **Answer** and then she or he has to evaluate the sentence by grabbing one of the available choices, at which point instead of annotating another field, **Go_To_Next_Entry** specifies to scroll the main text panel to the next sentence. Different evaluation protocols can be easily integrated in the annotation process by changing the specifications in the configuration file.

Conclusions

In this paper we have presented N.b., a graphical user interface for annotating spoken dialogue in a consistent fashion, with typing reduced to the minimum, according to instruction sequences that can be easily specified and modified.

Currently, we are testing and improving the various parts of the program in order to ensure ergonomicity and portability with respect to different annotation instructions, and to allow for maximum flexibility for input and output file formats. We are using this interface for developing a minimal and theory neutral annotation method of corpora of naturally occurring task-oriented human-human telephone conversations. We successfully completed an experiment for evaluating the inter-subject annotation agreement with a corpus of 12 recorded telephone conversations between customer and agent, each conversation being annotated by 5 different subjects. We are also investigating the definition of evaluation criteria such as precision, recall, and accuracy for the problem of comparing embedded non-linear bracketings of text. The evaluation of the annotated data will serve as benchmark study for assessing the human performance in this annotation task and the basis for developing and evaluating an algorithm for the discovery of the discourse constituents of spoken dialogue.

We hope that the annotating instruction specifications of N.b. are general enough to accommodate a large variety of linguistic theories, and even different annotation purposes than the ones we are currently interested
in.

While *N.b.* is intended for annotating discourse in spoken dialogue, we found that it can be used for other tasks, such as evaluating spoken language system performance. We think it is also possible to use *N.b.* for annotating words and phrases with their syntactic or semantic categories.

Currently, the program has been implemented in C on a Sun workstation under the X windows graphic environment. When the program will reach a deliverable state, *N.b.* will be available to all interested researchers.

**Acknowledgments**

We wish to thank Joe Polifroni, Christine Pao, Michael McCandless, Rob Kassel and Christine Nakatani for evaluating early versions of the program and providing very useful suggestions for improving the user interface.

This research was supported by ARPA under Contract N6601-94-C-6040, monitored through the Office of Naval Command, Control and Ocean Surveillance Center.

The corpora of telephone conversations used for the discourse annotation experiments is being contributed by American Airlines SABRE Travel Information Network and by Bell South Intelliventures.

**World Wide Web Reference**

Up-to-date information about *N.b.* can be found in the World Wide Web resource:


**References**


Hovy E.H., “Automated discourse generation using discourse structure relations”, *Artificial Intelligence*


Silverman K. et al. “TOBI: A standard for labeling En-