Designing Annotation Tools based on Properties of Annotation Problems.

Dennis Reidsma Nataša Jovanović
Dennis Hofs
University of Twente, Dept. of Computer Science, HMI Group
P.O. Box 217, 7500 AE Enschede, the Netherlands
{dennisr,natasa,hofs}@ewi.utwente.nl

Abstract

The creation of richly annotated, extendable and reusable corpora of multimodal interactions is an expensive and time-consuming task. Support from tools to create annotations is indispensable. This paper argues that annotation tools should be focused on specific classes of annotation problems to make the annotation process more efficient. The central part of the paper discusses how the properties of an annotation problem influence the design of the specialized tools used to manually create the annotations. Two existing tools, developed at the University of Twente, are used as examples.

1 Introduction

Research into multimodal human-human interactions has become more and more important during the last years. As a consequence, the need for annotated corpora which contains different aspects of natural human interactions has increased as well. A good corpus should be reusable, in a sense that it should be used in more than one research project and by more than one research teams, and extendable i.e. capable for further augmentation [10]. The creation of richly annotated, extendable and reusable corpora is an expensive and time-consuming task. A number of examples of the immense amount of effort needed for diverse types of annotations is presented in [13]. Corpus creation is also very costly because there is little effective technological support for annotation. Therefore many researchers agree that investment in tools to support creation and exploitation of annotated corpora is very important [4, 5, 10, 2].

Requirements for annotation tools come from a variety of users in different research areas. Section 2 gives a structured overview of user requirements which are collected from selected publications in this area. Since those publications are all written from a certain perspective, each brings its own valuable insights that are not covered by the other papers.

There are many requirements that cannot all be covered in one tool. Therefore, when one sets out to annotate a new phenomenon, it often turns out that there is no tool that quite fits this task. Creating a new specific tool carries the risk of having something that can, again, not be reused. Creating a very general tool that should cover a broader range of functionalities may lead to a tool that does not take advantage of the properties of one specific problem that could improve the efficiency of the annotation process.

The solution, as it is explained in [5], is neither to develop unrelated special-purpose annotation tools nor to build one integrated tool which supports all user requirements. The solution is to build a set of "focused tools" that all make use of a common corpus exchange format and API. A focused tool should support adding annotations to a data source, displaying them, and manipulating and extracting annotated data that match specified conditions [5].

This paper presents design guidelines for focused annotation tools[1]. The

[1]The work described in this paper was partly supported by the European Union 6th FWP

guidelines consider the general-specific trade-off, namely: they are general enough to be used in different domains and specific enough to take advantage of the properties of a particular annotation problem in order to make the annotation process more efficient.

The properties of annotation problems are analysed in section 3 regarding their influence on the design of a tool. Sections 4 and 5 illustrates the influence of properties of annotation problems on the design of two focused annotation tools: DACoder and the CVL (Continuous Video Labelling) tool.

2 Requirements

Developers of annotation tools are faced with various user needs from users in various research areas (e.g. linguistics, psycholinguistics, psychology, speech and language engineering, etc.). For obtaining user requirements for annotation tools we use several existing reviews of the available annotation tools [8, 9, 14, 7]. These reviews together outline most of the criteria used to rate existing annotation tools or to design new tools.

The chosen evaluations are performed from different perspectives due to different evaluation goals. The aim of the ISLE Natural Interactivity and Multimodality Working Group report [8] is to provide a survey of world-wide tools which support annotations of natural interactivity and multimodal data. As a result it outlines the most important overall user needs reflected in the tools and projects which created them. The aim of the evaluations presented in [9], [14] and [7] is to select a tool or set of tools based on analysis of research project needs. The reviews follow the same evaluation procedure which consists of two steps. First, based on the analysis of the project needs a list of requirements for annotation tools is defined (e.g. simplicity, quality assurance, compatibility with other tools, customization of the annotation scheme, etc.). After that, from the given annotation requirements the "evaluation criteria" are derived.

Table 2 lists all of the collected criteria. We classified those criteria according to user types, quality aspects and whether the criteria are related to creating annotations or to browsing and analyzing annotated data.

The users of annotation tools may be divided into three groups [4, 1]:

- **Annotators**: users who need a tool for their annotation task without bothering about data representations, internal design, working of the tool.
- **Annotation Consumers**: users who want to use annotated data from different reasons (e.g. theory testing, models evaluation and training, finding relations between data etc.). They have needs for querying and browsing annotated data.
- **Developers**
  Corpus developers: users responsible for corpus design (e.g. design of

IST Integrated Project AMI (Augmented Multi-party Interaction, FP6-506811, publication AMI-30). For more information see http://www.amiproject.org/

new annotation schema's or altering existing ones, understanding of data representation supported by the tool and mapping of their data to the existing structures)
System developers: users with the programming skills who are willing to add new functionalities and new components to the tool.

Regarding to quality aspects the evaluation criteria may be classified into: criteria of functionality and criteria of usability [4]. The functionality concerns the presence or absence of functions relevant for a specific task. It is about the relation tool-task. The aspect of usability concerns the relation tool-user.

The requirements for statistical data analysis and display are supported by software packages that a new tool would hardly displace. Furthermore, the requirements for input/output flexibility, flexibility in coding schemes and querying annotated data are covered by using a stand-off XML data format with a good API such as AGTK, NITE XML Toolkit (NXT) or ATLAS [11]. In this paper we focus only on the annotators as target group and the requirements related to efficiency of creating annotations such as: easy-to-use interface, marking, audio/ video interfaces, annotation process and visualization.

For developing focused tools we use the NITE XML Toolkit (NXT) [3]. It provides library support for building specialized interfaces, displays and analysis of highly structured and cross-annotated multimodal corpora. NXT provides routines for loading, accessing, manipulating and saving data as well as a query language (NiteQL) for exploitation of the annotated data. It uses a stand-off XML data format which consists of several interrelated XML files. The stand-off XML format enables capturing and efficient manipulation of the complex structures.

3 Characterizing Annotation Problems

Different annotation problems, such as transcription, video labelling or text markup, each have their own properties. This section gives an overview of those properties and discusses how they can influence the design of efficient tools for annotation.

3.1 Observation vs interpretation

A specific layer of annotation in a corpus may pertain to *direct observations* of events in the physical world such as certain movements, speech or gaze directions or to *interpretations* of those observations such as emotional states, dialogue acts or complex semantic annotations. The interpretations involve deducing information about the internal mental state of the persons involved in the observation, about their beliefs, desires or attitudes [13].

Interpretation takes a lot more time than simple observations. When coding observations, aiming for a realtime coding process may be sensible. When coding interpretations this may be less feasible. If the annotation is part observation and part interpretation, it may be a good idea to split it up.

Criteria	1	2	3	4	User type	Aspect	Creation
Portability		**X**	**X**	**X**	A, AC, SD		-
Can the tool be used on different platforms? Does it require any additional packages? (2) Is it easy to install? (4)							
Source code	**X**	**X**			AC, AD		-
Does the tool come with the source code?							
Flexible Architecture	**X**				SD		-
allows extension of the tool by adding new components							
Three layered structure	**X**				SD		-
Is the user interface separated from the application logic layer and from data representation layer so that each can be changed independently without influencing one another?							
I/O Flexibility	**X**	**X**		**X**	AC, SD	Functionality	-
What are the tool's input formats? Does the input data need any preprocessing? Is the output format compatible with other tools? Are there converters from/to other formats provided? Can annotation scheme be imported/exported and in which format?(4)							
Robustness and stability	**X**		**X**		AC, SD		-
Is the tool robust, stable and does it work in a real time?							
Audio/Video Interface		**X**			A	Usability	+
Does the tool offer an easy to use method for playing audio and/or video sections and segmenting sections? Does the tool support handling large media files? Does the tool support playing back the media file aligned with an annotation element?							
Flexibility in coding scheme	**X**	**X**	**X**	**X**	CD	Usability	+
Does the tool support easy addition of a new coding scheme or altering of the existing one? (1)(2)(3) Does the tool allow user to restrict format and/or the content of annotation data? (4) Can annotation levels be defined as obligatory or optional?(4) Can tag sets be specified? Can tag sets be structured?(4) Are annotation levels and tag sets defined within the tool or by external files?(4)							
Easy to use interface	**X**	**X**	**X**	**X**	A	Usability	+
The interface should support user as much as it is possible, to be intuitive and based on standard interfaces conventions							
Learnability				**X**	A	Usability	+
Is the tool easy to learn?							
Attractiveness				**X**	A	Usability	+
Does the user enjoy working with tool?							
Transcription Support	**X**			**X**	A	Functionality	+
Can the tool be used for speech transcription?							
Marking	**X**	**X**	**X**	**X**	A, CD	Functionality	+
Does the tool support annotations at different levels, of different modalities and annotations across levels and modalities? How much can the tool mark (e.g. just words or group of words; entire sentences or segments of sentences)? Does it allow the marking of discontinuous fragments? (2) Does the tool support simultaneous annotation for several persons? (3)							
Meta-data		**X**		**X**	A, CD	Functionality	+
Does the tool support meta-data such as annotators comments and notes referring to annotations or relating to the entire document?							
Annotation Process				**X**	A, CD	Usability	+
Does the tool support some kind of (semi) automatic annotation? Does the tool support selection-based annotation where only appropriate the tags are presented to the user?							
Visualization	**X**	**X**	**X**	**X**	A, CD	Usability	+
Scope: Is the annotated information visible for all annotation elements or only the currently active element? Style: Are the annotated element presented in form of text, menu/or radio button, etc? Does the tool provide further means to visualize the annotated information (colour, font size, brackets etc.)? (4) Can the user change visualization dynamically? Can the user define visualization? (1),(4) Does the tool support synchronized view of different annotation layers and of different modalities? (1) Does the tool have a large display to show the current works and corresponding data in a clear manner? (2)							
Documentation		**X**		**X**	A	Usability	+
Availability and quality of user manual; on-line help							
Querying, Extraction	**X**			**X**	AC	Functionality	-
Does the tool support (simple or powerful) search mechanisms and an interface to the search tool? Are the results presented in intuitive and easy-to use way?							
Data Analysis	**X**				AC	Functionality	-
Does the tool support (statistical) analysis of annotated data?							

1: Dybkjaer et al. [8]
2: Garg et al. [9]
3: Rydeman [14]
4: Dipper et al. [7]

A: Annotators
CD: Corpus Developers
AC: Annotation Consumers
SD: Software Developers

Table 1: Collected requirements for annotation tools

3.2 Input layers

Every annotation layer is based on certain sources of input. The most basic layers are based only on the audio and/or video (e.g. labelling of head nodding, transcription, hand tracking). More complex layers may be based on other layers as well (e.g. dialogue acts based on transcriptions, interpretation of gestures for their communicative function). Sometimes the reference from annotation elements to elements in input layers is made explicit, such as dialogue acts referring to text fragments. Sometimes this relation is implicit, such as for example the relation between dialogue acts and video or audio: though the explicit input is the speech of the participant, the video and audio offer valuable input for determining the exact dialogue act (facial expression, intonation, etc).

Explicit and implicit input layers determine what should be displayed in the tool. An annotation tool should preferably display only the explicit and implicit input layers and the created annotations, anything else would be a distraction. The explicit input layer should be displayed in a way that clearly shows its relation to the created annotation elements. The explicit input layers also influence the *selection mechanisms* of the tool.

3.3 Segmentation

The segmentation properties of an annotation have a large impact on the design of the GUI. The segmentation determines what fragments of the explicit input layer(s) an annotation element can refer to. A list of possible characteristics of the segmentation is given below.

- Segments may or may not relate to overlapping parts of the explicit input layers.
- Segments may or may not interleave with each other.
- Segments may or may not be discontinuous.
- Each segment may be annotated with one, or more than one element.
- The segmentation may or may not fully cover the input layer.
- The size of segments may differ per problem: single words, sentences, arbitrary time fragments, etc.

These properties determine how the selection mechanism should be designed, but also whether semi automatic support is possible for segmentation and selection. If for instance a tool is being developed for manual coding of part-of-speech, the segmentation properties suggest that the tool might perform segmentation automatically and present the segments (word) one by one for labelling. For dialogue acts the segmentation is not obvious, so it should be done by the annotator.

3.4 Labelling or complex information

Some annotation layers contain annotation elements that are just labels from a (possibly very complex) set or ontology. Other annotation layers have more complex structures as their constituent elements, such as for example the gestures as labelled in HamnoSys [12] or the multiple labels in MRDA [6].

When the information per annotation element consists only of a label, one can decide to map possible labels on the keyboard or a set of GUI buttons or to use a popup list when a new element is created. If the information is more complex, a separate panel for modifying annotation elements is probably more suitable.

3.5 Relations

Some annotation elements may define relations between/to other annotation elements. As far as the annotator is concerned, there are two views on a relation. One of the related elements may be considered an attribute of the other element, or their relation may be seen as an annotation element in its own right, stored in a separate layer.

3.6 Constraints

There may be constraints to element contents and relations (e.g. an answer belongs to a question, certain combinations of tags are not allowed, etc.). The tool may help keeping integrity by enforcing those constraints, limiting the choices of the annotator.

3.7 Default values

A special type of 'constraint' is a default value. If a default value for a certain attribute can be defined, the tool can support faster coding by pre-filling the attribute. Syrdal et al. show that in some cases default suggestions can speed up manual labelling without introducing too much bias in the annotation [15].

4 DA coder

The DA coder, or dialogue act annotation tool, currently supports the annotation of dialogue acts and adjacency pairs (relations between two dialogue acts). For each of the dimensions listed in the previous section, this section describes the properties of the dialogue act annotation problem and it discusses how these properties influenced the design of the DA coder.

Observation vs interpretation The annotation of dialogue acts and adjacency pairs involves the interpretation of the transcription and the media recordings to determine the intention of a speaker. Because this takes more time than making observations, it is not possible to annotate in real time. The

annotator should therefore be able to browse through the observation, pause the media and replay fragments. In the DA coder the user can move to any point in the observation and play the media from that point. The presence of the transcription providing a textual overview of the entire observation facilitates the annotation task. The media and the transcription are synchronized. While playing the video, words in the transcription are highlighted. It is also possible to play a media fragment directly from a selection in the transcription.

Input layers For the dialogue act annotations, the explicit layer is the transcription. This is reflected in the DA coder as the dialogue act annotations are displayed within the transcription text and they are created from selections in the transcription.

The media are an implicit input layer. In the DA coder this relation is visualized through the synchronization of the transcription with the media and vice versa.

The explicit input layer of the adjacency pairs are the dialogue acts. Upon creation of an adjacency pair, the user needs to select the source and target dialogue acts – as visualized in the transcription. The DA coder has a separate view that lists all adjacency pairs with the types of their source and target dialogue acts. When an adjacency pair is selected, the dialogue acts are highlighted to visualize the relation between dialogue acts and adjacency pairs.

Segmentation The dialogue acts can refer to any sequence of words pronounced by one agent. From the perspective of the annotations for one agent, this means that the segments must be continuous and consequently they cannot interleave with each other. From the perspective of the entire observation however, there may be utterances of other agents interleaved with the dialogue act of one agent. In that case the dialogue act segments may be discontinuous. The boundaries of a segment coincide with word boundaries, but it is not defined at what word boundaries dialogue acts may start or end. Any word may belong to at most one dialogue act, so overlap is not allowed and the segmentation need not cover the entire input layer.

All together this has consequences for the selection mechanism. Selections are made in the transcription that is displayed for the entire observation. The DA coder will always select whole words and it will only select word sequences from one agent, even if there are utterances from other agents in between. In that sense discontinuous selections are allowed. Selection of words that already belong to a dialogue act is not allowed and it is not possible to select a word sequence from one agent if that sequence contains another dialogue act. In that sense discontinuous selections are not allowed.

Labelling or complex information The annotation of dialogue acts can be considered a labelling task as the annotation elements consist of a label taken from an ontology of dialogue act types. However the annotator can optionally specify a set of addressees for a dialogue act as well. If it is chosen to do so,

the annotation is not a labelling task anymore. The DA coder provides for both options. If the addressees are not annotated, the user can make a new annotation by selecting the appropriate segment and pressing a key that makes a list pop up displaying the ontology of dialogue act types. On the other hand the DA coder also includes a separate panel for the creation or modification of dialogue acts. In this panel the user can select a dialogue act type as well as the addressees of the dialogue act.

Adjacency pairs involve complex information consisting of two dialogue acts and a type that specifies how the dialogue acts are related. In another sense the annotation of adjacency pairs can be considered a labelling task in which each annotation element has three labels (the available dialogue acts can be considered a set of possible labels). Therefore the annotation of adjacency pairs takes a similar approach as the dialogue acts. The user can press a key to get a list displaying the ontology of adjacency pair types. Then the source and target dialogue acts should be selected. Because the dialogue act annotations are displayed in the transcription, the user can simply click them. A separate panel is provided as an alternative to assign the three elements of an adjacency pair.

Relations Dialogue acts can be related to other dialogue acts in the form of an adjacency pair. There are two possible views on this relation. One dialogue act could point to another dialogue act and define an adjacency pair type, or we could introduce adjacency pair elements that point to two dialogue acts and define their relation. In the DA coder, as already explained, we chose for the second view, which is more appropriate for the mutual character of the adjacency pair relation. For the annotator this means that an adjacency pair relation between two dialogue acts is defined by creating an adjacency pair element and let it point to two dialogue acts.

Constraints The only constraint enforced in the DA coder is that a dialogue act can be addressed to any subset of the participating agents, but that subset may never contain the speaker of the dialogue act. For the assignment of addressees, the DA coder shows check boxes for the participating agents but the check box for the speaker is disabled.

One can conceive several other useful constraints, in particular with respect to pairs of dialogue acts that can form an adjacency pair, but they are not considered in the current version of the DA coder.

Default values We did not yet experiment with default values. One of the attributes that might have default values is the addressee attribute. Defaults might be the previous speaker, or all participants. We do not know however if this would improve the annotation process.

Display All input layers are displayed in the DA coder, but the visualization of the annotation elements are most clearly linked to their explicit input layer. For

the dialogue acts this is the transcription. As explained before, the dialogue acts are visualized in the transcription view. Because of the segmentation properties, which say that the dialogue act segments are continuous and they do not overlap (for one agent), it is possible to mark the start and end of each dialogue act. This is done in a different colour (blue) than the colour of the transcription text (black). The start marker includes the type of the dialogue act. The addressees attribute is not visualized in the transcription in order not to complicate the view unnecessarily. Selection of the annotation elements is easy as the user can simply click a blue dialogue act marker.

Because of the more complicated segmentation properties for adjacency pairs, the dialogue act markers in the transcription are not an appropriate place to visualize the adjacency pairs. The DA coder has a separate view that lists all available adjacency pairs along with their type and the types of the source and target dialogue acts (transcription fragments are not included as yet). When an adjacency pair is selected in this list, the markers of the source and target dialogue acts in the transcription are highlighted, each with a different colour.

The link with the implicit input layer, the media, is made through the synchronization of the media player with the transcription as explained before.

Several visualization improvements could be made, in particular user customization. This may concern the colour and displayed attributes of an annotation element, but also the option to show or hide annotations of a particular layer. In anticipation of such improvements, the transcription view has been designed to facilitate configurable display of *any* type of annotation that is defined on the transcriptions.

5 CVL tool

The CVL (Continuous Video Labelling) tool supports labelling of time-aligned annotation layers directly related to the signal files. This section describes the properties of the annotation problems supported by the CVL tool and discusses how these properties influence its design. In contrast to the DA coder tool, the CVL tool targets a *class* of annotation problems rather than one specific problem. Any annotation layer that consists of simple labelling of non-overlapping segments of the time line can be coded using this tool. Examples are gaze direction, postures, target of pointing gestures, emotions, etc.

Observation vs interpretation The annotations supported by the CVL tools can be both observation and interpretation. For instance, labelling gaze direction or postures is an observation problem while labelling emotions involves the interpretation of the observed events. Therefore the CVL tool supports real-time as well as off-line annotations. The user can create annotations by clicking the label buttons while the media file is playing or by pausing the media file at the certain point before clicking the label button.

Input layers The explicit input layers of time-aligned annotation layers are video and/or audio files. An observation may have more than one associated signal files. Each of these files can be useful for labelling different types of information as well as different aspects of the same annotation layers. For instance, for labelling gaze direction of a person, the video file that contains that person should be used as an input. The CVL tool enables users to select the appropriate input in the media player.

Segmentation A segment of time-aligned annotations is an arbitrary time fragment. From the perspective of the annotation of one person, the segments are continuous, non-overlapping and they fully cover the whole input layer. Each segment may be annotated with only one annotation element within the layer.

These properties influence the selection mechanisms in the CVL tool. When a user click on a new label button, the end time of previous segment is set with the current video time and a new segment is automatically started at this point. Hence the selection in the CVL tool coincides with the creation of annotation elements.

Labelling or complex information The CVL tool specifically targets those annotations that involve labelling tasks. The labels are mapped into GUI buttons. In the case where a label set contains the agents involved in interaction, the buttons are ordered so that they match the positions of the speakers in played video files. This facilitates the annotation process in a sense that the annotator doesn't have to bother about the position of agents that are not captured in the video file. From the positions of the buttons it should be clear where the agents are seated. We expect that mapping the labels onto keystrokes will increase the efficiency of annotation process. Therefore in the next version of the CVL tool we will enable users to configure those mappings.

Relations and Constraints The time-aligned annotation layers supported in the CVL tool do not define any relation between/to other annotation elements. The CVL coder also does not include constraint support

Display One of the most convenient displays for this type of annotation problem would be the time aligned annotation board as provided by programs like TASX and Anvil. Since development of this type of this display is not yet finished it has not yet been included in the current version of the CVL tool. For now the labelled elements are shown as a list of text representations, with highlighting synchronized to the time line of the video.

6 Conclusions

This paper argues that annotation tools should be focused on specific classes of annotation problem to make the annotation process more efficient. The central

part of the paper discusses how the properties of an annotation problem influence the design of the specialized tools used to manually create the annotations. Two tools developed at the University of Twente are used as an example. Initial experiences with the tools show that their efficiency is acceptable, but more experiments have to be done to compare them properly with existing tools and approaches. The tools will soon be available for free. More information can be requested from the authors through email.

References

[1] N. O. Bernsen, L. Dybkjr, and M. Kolodnytsky. The nite workbench - a tool for annotation of natural interactivity and multimodal data. In *Proc. of the Third International Conference on Language Resources and Evaluation*, 2002.

[2] S. Bird and M. Liberman. A formal framework for linguistic annotation. *Speech Communication*, 2000.

[3] J. Carletta, S. Evert, U. Heid, J. Kilgour, J. Robertson, and H. Voormann. The NITE XML toolkit: flexible annotation for multi-modal language data. *Behavior Research Methods, Instruments, and Computers*, 35(3):353–363, 2003.

[4] J. Carletta, A. Isard, M. Klein, A. Mengel, and M.B. Moller. The mate annotation workbench: User requirements. In *Proc of ACL-99 Workshop Towards Standards and Tools for Discourse*, 1999.

[5] J. Carletta, D. McKelvie, A. Isard, A. Mengel, M. Klein, and M. B. Mller. *A generic approach to software support for linguistic annotation using XML.* Continuum International., 2002.

[6] R. Dhillon, S. Bhagat, H Carvey, and E. Shriberg. Meeting recorder project: Dialogue act labeling guide. Technical report, ICSI Speech Group, Berkeley, USA, 2003.

[7] S. Dipper, M. Goetze, and M. Stede. Simple annotation tools for complex annotation tasks: an evaluation. In *Proceedings of the LREC Workshop on XML-based Richly Annotated Corpora*, 2004.

[8] L. Dybkjaer, S. Berman, M. Kipp, M.W. Olsen, V. Pirelli, N. Reithinger, and C. Soria. Survey of existing tools, standards and user needs for annotation of natural interaction and multimodal data. Technical report, January 2001.

[9] S. Garg, B. Martinovski, S. Robinson, J. Stephan, J. Tetreault, and D.R. Traum. Evaluation of transcription and annotation tools for a multi-modal, multi-party dialogue corpus. In *Proceedings of the LREC*, 2004.

[10] N. Ide and C. Brew. Requirements, tools and architectures for annotated corpora. In *Proc. of Data Architectures and Software Support for Large Corpora*, 2000.

[11] C. Laprun, J.G. Fiscus, J. Garofolo, and S. Pajot. A practical introduction to atlas. In *Proceedings of the LREC*, 2002.

[12] S. Prillwitz, R. Leven, H. Zienert, T. Hanke, and J. Henning. Hamnosys. version 2.0; hamburg notation system for sign languages. an introductory guide. Technical report, 1989.

[13] D. Reidsma, R. Rienks, and N. Jovanovic. Meeting modelling in the context of multimodal research. In *Proc. of the Workshop on Machine Learning and Multimodal Interaction*, 2004.

[14] B. Rydeman. Using multimodal annotatino tools in the study of multimodal communication involving non-speaking persons. 2003.

[15] Ann K. Syrdal, Julia Hirschberg, Julie McGory, and Mary Beckman. Automatic tobi prediction and alignment to speed manual labelling of prosody. *Speech communication*, 33:135–151, 2001.