XML Format guidelines for the TUNA Corpus data in the REG Chalenge 2008

Albert Gatt

1 Introduction

This document explains the XML format of the TUNA corpus data in the REG Challenge 2008. Samples of this data are also provided. Each file in the sample data consists of a single *corpus instance*, i.e. a single pairing of a domain (description of entities and their properties) and description which is intended to describe one target in the domain. This data is the result of an experiment, in which participants were asked to describe objects in visual domains. The domains consisted of seven objects shown in a sparse 3 (row)×5 (column) grid.

1.1 Filenames in the TUNA data

Filenames follow the naming convention `sNtM.xml`. Each filename is unique, and functions as a unique identifier for a corpus instance.

1.2 Images

The full set of images used in the TUNA elicitation experiment is also provided. Entities in the XML files have pointers to these images; see §2.2.1 below. The images can be found in the `images/` subdirectory in this distribution.

1.3 Object types

The TUNA data was elicited in two different domain types: There are references to people, and references to furniture items. The data is divided by domain type, with files in correspondingly named sub-directories. The images are also subdivided.

2 XML Format description

The basic format of corpus instances is shown in Figure 1.

```
<TRIAL ID=''sNtM''>

    <DOMAIN>
      representation of entities and their properties
    </DOMAIN>

    <WORD-STRING>
        the string describing the target referent in the domain
    </WORD-STRING>

    <ANNOTATED-WORD-STRING>
        the string in WORD-STRING, where the relevant substrings are annotated
        with attributes in ATTRIBUTE-SET
    </ANNOTATED-WORD-STRING>

    <ATTRIBUTE-SET>
     the set of domain attributes which are true of the referent,
     and which are included in the description
    </ATTRIBUTE-SET>

</TRIAL>
```

Figure 1: Format of corpus instances

2.1 The TRIAL node

The `TRIAL` node pairs domains and descriptions. This node has the following XML attributes:

1. `ID`: the unique corpus instance identifier. This is identical to the filename.

2. `CONDITION`: This takes one of two values: ±`LOC`. The value of `CONDITION` is a reflection of the experimental condition in which the data was elicited. In the +`LOC` condition, participants were told that they could refer to entities using any of their properties, including their location. In the −`LOC` condition, they were *discouraged* from doing so, though not prevented. This means that, while it is more likely that descriptions in the +`LOC` condition contain locative expressions, this is not

always the case. Moreover, there are some instances in the −LOC condition where participants did use locative expressions. The annotation of locatives is explained in §2.5.

2.2 The DOMAIN node

The DOMAIN node represents the entities in a domain. Every domain consists of exactly 7 unique entities, each described in terms of their attributes.

2.2.1 The ENTITY node

Attributes in the furniture/household domain		
Attribute	**Type**	**Possible values**
TYPE	literal	*chair, sofa, desk, fan*
COLOUR	literal	*blue, red, green, grey*
SIZE	literal	*large, small*
ORIENTATION	literal	*left, right, front, back*
X-DIMENSION (column number)	gradable	1, 2, 3, 4, 5
Y-DIMENSION (row number)	gradable	1, 2, 3
Attributes in the people domain		
Attribute	**Type**	**Possible values**
TYPE	literal	*person*
ORIENTATION	literal	*front, left, right*
AGE	literal	*young, old*
HAIRCOLOUR	literal	*dark, light, other*
HASSHIRT	boolean	0 (`false`), 1 (`true`)
HASBEARD	boolean	0 (`false`), 1 (`true`)
HASHAIR	boolean	0 (`false`), 1 (`true`)
HASGLASSES	boolean	0 (`false`), 1 (`true`)
HASTIE	boolean	0 (`false`), 1 (`true`)
HASSUIT	boolean	0 (`false`), 1 (`true`)
X-DIMENSION (column number)	gradable	1, 2, 3, 4, 5
Y-DIMENSION (row number)	gradable	1, 2, 3

Table 1: Attributes and values used in the TUNA corpus

An ENTITY node consists of a set of attributes, corresponding to a description of a domain object. The ENTITY node has the following XML attributes:

1. `ID`: A unique integer identifier.

2. `IMAGE`: The filename containing the picture of this entity. These images are found in the `images/` directory provided with this distribution.

3. `TYPE`: This specifies whether an entity in this specific corpus instance was the *target* or a *distractor*. There is only one target, and six distractors. The target is the intended referent which the description is intended to identify, relative to the domain distractors.

The set of attributes and possible values in each domain (furniture/people) is displayed in Table 1.

2.2.2 ATTRIBUTE nodes

Every `ENTITY` node has a number of child `ATTRIBUTE` nodes, representing properties of the entity in attribute-value notation. Each has the following XML attributes:

1. `NAME`: the name of the attribute e.g. *colour* or *size*

2. `VALUE`: the value of the attribute e.g. *red* or *large*

3. `TYPE`: the value type. This is either `literal` (where the value is an element of a set of possible values), `boolean` (the value is 1 or 0) or `gradable`, meaning the attribute takes a numeric value. There are only 2 gradable attributes, `x-dimension` and `y-dimension`, corresponding to the column (X) and row (Y) of the entity in the domain grid.

2.3 The WORD-STRING node

This is the description of the target entity, as written by the human participant, provided without any annotation. Some of these descriptions may contain punctuation. Participants should note, however, that *punctuation will not be counted in the evaluation for the REG Challenge.*

2.4 The ATTRIBUTE-SET node

The `ATTRIBUTE-SET` is the set of attributes that the human description contains. Human descriptions were annotated by identifying substrings which 'realise' domain attributes and annotating them with the corresponding attribute information. For example, the description *a bearded man*

consists of the attributes `hasBeard: 1` and `type: person`. Therefore, `ATTRIBUTE-SET` is a set of attribute nodes representing the semantic content.

Each `ATTRIBUTE` node in the `ATTRIBUTE-SET` has a unique `ID`, as well as a `NAME` and `VALUE` exactly as per the representation of attributes in the `DOMAIN`.

Occasionally, an `ATTRIBUTE-SET` may contain the same `ATTRIBUTE` twice (i.e. two `ATTRIBUTE` nodes with the same `NAME` and `VALUE`, but different `ID`s. This is because a description may contain repeated information. For example, in *the white-haired man with a white beard*, there are two instances of *hairColour*, one for the man's hair, and one for the colour of his beard.

The semantic representation in the `ATTRIBUTE-SET` is 'flat', in the sense that it does not reflect the syntactic structure of a description, or the way attributes are realised in relation to eachother; for information about these aspects, we have provided an `ANNOTATED-WORD-STRING` representation, described next.

NB: The `ATTRIBUTE-SET` representation is exactly the same as the `DESCRIPTION` representation used in the ASGRE Challenge 2007, except that it has been re-named.

2.5 The ANNOTATED-WORD-STRING node

The `ANNOTATED-WORD-STRING` representation is provided for training and development purposes only. It consists of the string in the `WORD-STRING` node, where each substring is annotated with the attribute it represents. The idea is to map substrings to the relevant parts of the target entity's domain representation. This representation is of interest especially to participants in Task 2 (mapping from attribute sets to word strings), because it indicates how individual attributes map to substrings.

The attributes in the `ANNOTATED-WORD-STRING` are exactly those found in the `ATTRIBUTE-SET` (except for an additional `META-ATTRIBUTE`, one which see directly below. Two further issues are worth noting:

1. `META-ATTRIBUTE`: For locative expressions, the `ANNOTATED-WORD-STRING` uses a `META-ATTRIBUTE`. This encloses the relevant expression (e.g. the string *in the middle*), and has, as its child nodes, either an *x-dimension* attribute, a *y-dimension* attribute or both. It has both just in case the locative expression is a combination of both the vertical (Y) and the horizontal (X) position of the target referent. This is also the

reason why the META-ATTRIBUTE was included, since a single locative expression does not correspond to a single location (X or Y) value in one-to-one fashion. For example, the expression *in the centre of the screen*, where the centre is a function of both row and column, is annotated as follows:

```
<META-ATTRIBUTE NAME="location">
    <ATTRIBUTE NAME="x-dimension" VALUE="3" />
    <ATTRIBUTE NAME="y-dimension" VALUE="2" />
        in the centre of the screen
</META-ATTIRIBUTE>
```

NB: The META-ATTRIBUTE *nodes are only included in the* ANNOTATED-WORD-STRING. They are not part of ATTRIBUTE-SET. Nor does any of the REG Challenge tasks require the inclusion of META-ATTRIBUTEs in the output.

2. **Nesting**: A further case of nesting is to be found with *hairColour*, which is always an ATTRIBUTE node that is the daughter of another ATTRIBUTE node. This is because the use of *hairColour* always entails that *hasHair* has the value 1. For example, the expression *white-haired* is annotated as follows:

```
<ATTRIBUTE NAME="hasHair" VALUE="1">
    <ATTRIBUTE NAME="hairColour" VALUE="light" />
        white-
    </ATTRIBUTE>
    haired
</ATTIRIBUTE>
```

3 Inputs and outputs in the REG tasks

There are three tasks in the REG Challenge involving the TUNA corpus. They are listed below in terms of their input/output specifications relative to the corpus data.

1. **TUNA Attribute Selection (AS)**: Takes a DOMAIN as input, and returns an ATTRIBUTE-SET as output. Participants will only have the DOMAIN available in the test set for this task.

2. **TUNA Realisation (R)**: Takes an `ATTRIBUTE-SET` as input, and returns a `WORD-STRING` as output. Participants will have only the `ATTRIBUTE-SET` available in the test set for this task.

3. **TUNA REG**: Takes a `DOMAIN` as input, and returns a `WORD-STRING` as output. Participants will have only the `DOMAIN` in the test set for this task. `WORD-STRING`s do not need to contain punctuation. In comparing human and automatically-generated strings, punctuation will not be taken into account.

NB: The `ANNOTATED-WORD-STRING` node described in §2.5 is available for training and development purposes only. *It will not be available in the test set inputs.*