VoxML Annotation Tool Review and Suggestions for Improvement

Mark Klement*[1], Alexander Henlein*[2], and Alexander Mehler*[3]

*Text Technology Lab, Goethe-University Frankfurt
[1]s0058073@stud.uni-frankfurt.de
[2]henlein@em.uni-frankfurt.de
[3]mehler@em.uni-frankfurt.de

Abstract

The aim of this paper is to evaluate the annotation tool used in VoxML Track ISA-17 henceforth called VoxML Annotation Environment[1]. We describe our experiences with the modeling language VoxML, present our own annotation tool – called TTLab VoxML Annotator –. and work out qualitative criteria for evaluating both tools. In our view, the most important criteria for efficient annotation are its speed and the quality and standardization of the resulting annotations for downstream tasks. In addition to this evaluation, we make suggestions for improvements to help increase the practical use of VoxML Annotation Environment in the underlying annotation domain. Finally, we describe a set of extensions and additional application areas.

1 Introduction

VoxML (Pustejovsky and Krishnaswamy, 2016) is a versatile modeling language for storing "semantic knowledge about real-world objects" and events and attributes related to them. In the area of Text2Scene generation, VoxML stands out above all for its ability to model so-called habitats and verbs of motion and thus generate not only static but also animated scenes and context-sensitive behavior fully automatically (Krishnaswamy and Pustejovsky, 2016).

In this paper we describe our work in connection with VoxML. We discuss the data types we have chosen and the tools we have created to gain evaluation criteria for VoxML Annotation Environment. This is followed by a description and evaluation of using its interface and annotation guidelines. Thirdly, we propose a number of improvements and extensions to VoxML Annotation Environment and VoxML modeling language itself.

[1]https://github.com/csu-signal/VoxML-Track-Annotation-2021

2 TTLab VoxML Annotator

Our primary field of application for VoxML is Text2Scene generation(Tan et al., 2019). VoxML itself supports the modeling of attributes (like colors), functions (e.g. n:1 mapping), programs (motion sequences of verbs), relations (on, in front of, ...) and the assignment of these to corresponding objects. Based on these VoxML descriptions, VoxSim (Krishnaswamy and Pustejovsky, 2016) offers an "event simulation engine" for the simulation of agents behaviors. Our goal was to create an annotation tool that can be used to directly annotate these descriptions.

Since VoxML documents are XML-compliant and thus have a tag-based structure with repetitive elements, the idea arose to simplify and accelerate the creation process for VoxML documents. For this purpose, we developed a desktop annotation program, called "TTLab VoxML Annotator ", with a graphical user interface using Python / Tkinter. As already indicated, the simplification was achieved by exploiting repetitive tags and components of the documents. In addition, we strive for a high degree of standardization of annotation documents by using spin boxes and option menus with predefined options for input instead of text fields wherever possible. For a screenshot of TTLab VoxML Annotator see Figure 1. A major advantage of simplifying the annotation process is that it allows the annotation tool to be used by non-specialists with little effort and without a long training period.

To further speed up the annotation process we started experimenting with annotations of objects based on machine learning (ML). This was achieved by exploiting the fact that objects can often be assigned to classes of objects that consequently share sets of features. More specifically, we trained a classifier that maps point clouds to the

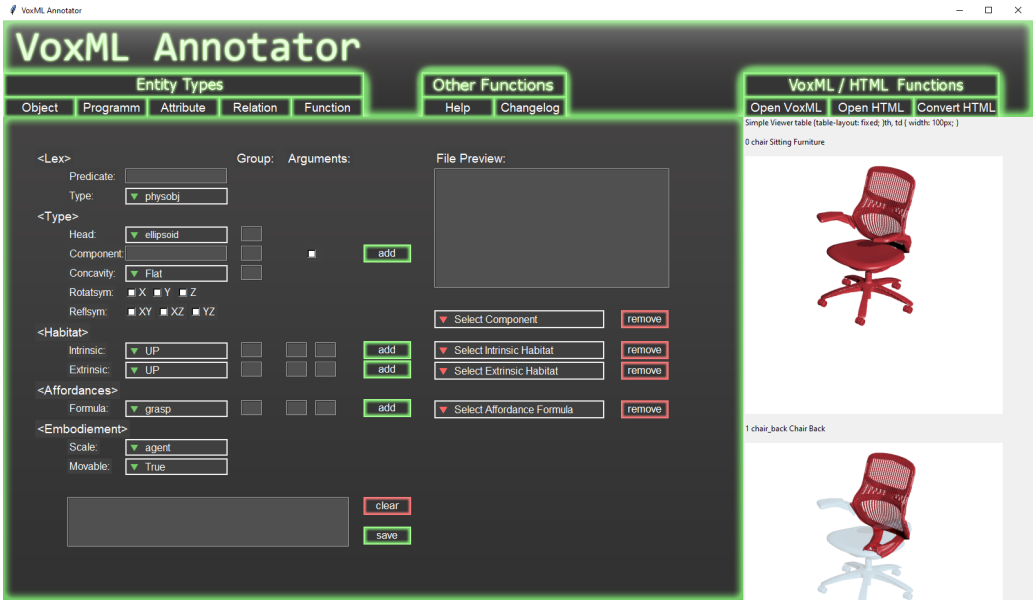

Figure 1: The interface of TTLab VoxML Annotator for simplifying VoxML-related annotations. In the upper left corner, the different VoxML types can be selected and then annotated in the main window. On the right, you can open the preview images of ShapeNet (Chang et al., 2015) and PartNet (Mo et al., 2019) and thus align the object annotation with these models.

10 basic classes of ModelNet10 (Wu et al., 2015). The neural network itself is based on PointNet (Qi et al., 2017) and DGCNN (Wang et al., 2019). For all 10 classes we created predefined VoxML documents, which then only have to be adapted for the corresponding objects. ModelNet also provides a dataset with 40 object categories. For these we have already prepared corresponding VoxML documents, the classifier only needs to be trained on this dataset. In the future, as more data becomes available, this classification will be applied directly to individual attributes instead of giving pre-annotated suggestions.

Based on these adaptations and extensions, the simplification and acceleration or automation of annotation as well as the standardization of its results are the most important factors guaranteeing efficient annotation processes. Accordingly, we utilize these criteria to evaluate VoxML Annotation Environment.

3 Experiencing VoxML Annotation Environment

VoxML Annotation Environment, provided for the ISA17 Visualization Track, gives the annotator an image as annotation source. The annotator is then asked to generate a caption for the image and describe the key activity depicted in it. This includes a list of involved objects and the activities associated with them. Furthermore, it must be answered which conditions must be fulfilled for these activities and whether they are also represented in the scene.

3.1 Advantages

VoxML Annotation Environment is easy to use and clearly structured with respect to the addressed task. One gets into the annotation process very quickly without having to familiarize oneself extensively with the tool. Its browser interface makes its mobile, ubiquitous use just as easy. In contrast to this, the annotation of VoxML entities, programms, etc. appears comparatively abstract and requires a basic knowledge of the underlying model and its predefined syntax. However, by means of pictorial examples and the replacement of the XML-based VoxML syntax by textual annotations, the annotation process is simplified. The setup itself is also straightforward.

The annotation manual is short and compact, using an example as a guide. For most annotations this is sufficient, but a few points remained unclear (see Section 3.2)).

3.2 Disadvantages

Given the three criteria of efficient annotation processes from Section 2, there are several reference points for improving VoxML Annotation Environ-

ment.

Currently, user input is done exclusively via text fields. When naming scenes (headings) and objects, this is a legitimate approach due to the amount of possible scenes and configurations. However, we see potential for optimization in the modeling of interactions and spatial relations:

(1) In the area of entity interactions, redundant annotations occur due to the use of the passive voice (e.g., "The woman holds the cup" vs. "The cup is held by the woman"). By representing interactions as tuples, annotations of passive constructions can be generated automatically. The task of this simplification is to represent the verb denoting the interaction in the infinitive together with the arguments involved in order to generate the active or passive form. This can not only speed up the annotation, but also increase its quality in cases where the annotator is not a native speaker or is not fully familiar with the passive voice of, for example, rare verbs.

(2) In the case of annotating object relations, a related problem concerns their reference points. Thus, spatial configurations and relations can be simplified by analogy to interactions. For example, reverse relations can be automatically generated by tuples with reversed argument order. This also concerns the automatic generation of transitive relations: if it is annotated that there is a phone on a table in the current scene and the ceiling fan above the phone, it can be inferred that the fan is above the table. Since the number of spatial relations is relatively small, it is better for the user to select them from predefined option menus or spin boxes rather than typing them into freely editable text fields. A side effect of predefined inputs is the avoidance of spelling errors. At the same time, the spatial relations can be adapted to the IsoSpace standard (Pustejovsky et al., 2011; ISO, 2020). However, this can again significantly increase the complexity of the annotation.

(3) The last note refers to the fact that completed annotations can no longer be modified. If, after annotating multiple images, the annotator finds that a particular activity should be named consistently instead of using multiple synonyms, there is no option for post-correction. In contrast, standardizing the annotation of interactions and relations in the manner described above would significantly reduce the need for this feature.

3.3 Summary

The annotation process is clear and it is quite easy to get familiar with the system. The annotation guide itself is compact, but sufficient in most cases, and the few ambiguities should be easy to add to. The simplicity and accessibility of the annotation is partly due to the many free-text fields, but we would still like to see more standardization and thus less room for error. Many annotations could also be automated and thus accelerated in the long run if the same information did not have to be entered for the umpteenth time (if, for example, several cups had already been annotated before), but could be taken from (one's own) previous annotations.

4 Minor Points

We list minor issues regarding VoxML Annotation Environment and its annotation guidelines. The first note is about two bugs, one of which is almost negligible and one of which can be at least partially bypassed during annotation. The smaller one is the fact that the first image displayed after starting the interface changes as soon as the annotation is started. The more problematic bug comes into play when new annotations need to be added or removed. Then not only one entry is changed, but all subsequent entries are moved and must be corrected.

A final comment concerns the example given in the annotation guideline, which is obviously too simple also due to lack of ambiguity. Thus, it remains unclear how to deal with duplicates of objects in a scene. One image of the dataset contains a scene with several plates on a table (Figure 28), two of which are covered with food. This example does not clarify whether the plates that do not contain food should be annotated more than once, nor does it explain how to handle the seemingly more important plates that contain food.

5 Points of Interest

Simplicity is of utmost interest: as much information as possible should be extracted from images, with as little effort as possible. However, the more information that needs to be annotated, the longer this process takes and the more exhausting it is for the annotator. In the following enumeration, we provide suggestions as to what information is still of interest to annotate[2].

[2]For illustration purposes, all following examples are related to the example from the Annota-

Timing of the activities: Instead of simply listing the activities associated with an object, they should be arranged chronologically. This eliminates the need to enumerate circumstances that the activity requires in order to be executable, while at the same time better structuring the annotation for possible post-processing.

Entity to image mapping: Since all entities are to be enumerated in writing, we could also mark them directly in the image. However, we are not sure how interesting such a dataset could be, since there are already huge amounts of them (e.g. COCO (Lin et al., 2014)). But it could help with the disambiguation of object terms (table = dining table, bedroom table, ...).

Hyponymy & Hypernymy Annotation of such relations could also reduce annotation overhead. For example, in the example *glass* and *cup* can be grouped together as "drinking vessel" (or per WordNet (Miller, 1995) "container"). "Be drunk from" and "be held" can then be assigned to "drinking vessel" with the annotation that *glass* and *cup* are each a drinking vessel. Ideally, such relations can be extended and completed across all annotations.

Holonymy & Meronymy In addition to the enumeration of entities, part-of-relations are also interesting. A wine glass, for example, is not held "as a whole" but usually by its stem. However, in contrast to hyponymy relations, there are hardly any meronymy datasets so far, and if there are, they are very limited (e.g. PartNet (Mo et al., 2019) or WordNet).

6 Conclusion

We see great potential in the versatility and ubiquitous usability of VoxML Annotation Environment. It is also worth mentioning that this environment allows very precise descriptions of scenes. Nevertheless, we also see considerable potential in simplifying and accelerating the corresponding annotation process in order to relieve the user and increase the quality of annotation results.

References

Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. 2015. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR].

ISO. 2020. Language resource management — Semantic annotation framework (SemAF) — Part 7: Spatial information (ISO-Space). Standard ISO/IEC TR 24617-7:2020.

Nikhil Krishnaswamy and James Pustejovsky. 2016. VoxSim: A visual platform for modeling motion language. In *In Proc. of COLING'16*, pages 54–58, Osaka, Japan. The COLING 2016 Organizing Committee.

Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Kaichun Mo, Shilin Zhu, Angel X. Chang, Li Yi, Subarna Tripathi, Leonidas J. Guibas, and Hao Su. 2019. PartNet: A large-scale benchmark for fine-grained and hierarchical part-level 3D object understanding. In *In Proc. of IEEE/CVPR'19*.

James Pustejovsky and Nikhil Krishnaswamy. 2016. VoxML: A visualization modeling language. In *In Proc. LREC'16*, pages 4606–4613, Portorož, Slovenia. European Language Resources Association (ELRA).

James Pustejovsky, Jessica L Moszkowicz, and Marc Verhagen. 2011. ISO-Space: The annotation of spatial information in language. In *Proc. of the Sixth Joint ISO-ACL SIGSEM Workshop on ISA*, pages 1–9.

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. 2017. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *In Proc. of IEEE'17*, pages 652–660.

Fuwen Tan, Song Feng, and Vicente Ordonez. 2019. Text2scene: Generating compositional scenes from textual descriptions. In *In Proc. of IEEE/CVF'19*, pages 6710–6719.

Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. 2019. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (TOG)*.

Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 2015. 3D Shapenets: A deep representation for volumetric shapes. In *In Proc. of IEEE'15*, pages 1912–1920.

tion Guideline (https://github.com/csu-signal/VoxML-Track-Annotation-2021/blob/main/ISA-17-guideline.pdf).