Identifying and inferring objects from textual descriptions of scenes from books

Andrew Cropper

Department of Computing, Imperial College London
a.cropper13@imperial.ac.uk

Abstract

Fiction authors rarely provide detailed descriptions of scenes, preferring the reader to fill in the details using their imagination. Therefore, to perform detailed text-to-scene conversion from books, we need to not only identify explicit objects but also infer implicit objects. In this paper, we describe an approach to inferring objects using Wikipedia and WordNet. In our experiments, we are able to infer implicit objects such as *monitor* and *computer* by identifying explicit objects such as *keyboard*.

1998 ACM Subject Classification I.2.10 Vision and Scene Understanding, I.2.6 Learning, I.2.7 Natural Language Processing

Keywords and phrases Text-to-Scene Conversion, Natural Language Processing, Artificial Intelligence

Digital Object Identifier 10.4230/OASIcs.ICCSW.2014.1

1 Introduction

Since the release of Toy Story in 1995, animation films have become increasingly popular [21]. As animators strive for photorealistic animation, budgets are spiralling [16]. Similarly, with next-generation consoles pushing the boundaries of computer graphics, expectations for visually aesthetic video games are increasing. Consequently, to generate more detailed content, games companies have to recruit more game artists, increasing the cost of development [17].

A text-to-scene conversion system (TTSCS) generates a 2D or 3D scene from a textual description provided by the user. For example, figure 1 shows the scene generated by Wordseye [7], a TTSCS described in section 2, for the text “The lawn mower is 5 feet tall. John pushes the lawn mower. The cat is 5 feet behind John. The cat is 10 feet tall”. TTSCSs have the potential to reduce budgets and production times for animation films and video games by automatically generating scenes from user-provided scripts.

In this paper, we focus on textual descriptions of scenes from fiction books, where authors rarely provide detailed descriptions, preferring the reader to fill in the details using their imagination. Therefore, to perform detailed text-to-scene conversion from fiction books, we need to not only identify explicit objects but also infer implicit objects. For example, for the extract “I was going to email Van and Jolu to tell them about the hassles with the cops, but as I put my fingers to the keyboard, I stopped again.”, a TTSCS needs to identify the keyboard and then infer that a computer is a likely accompaniment. In the following sections, we describe an approach using Wikipedia and WordNet.

1.1 Contributions

To our knowledge, this is the first paper to investigate inferring objects in a TTSCS using Wikipedia as a source of world-knowledge. Our main results are as follows:

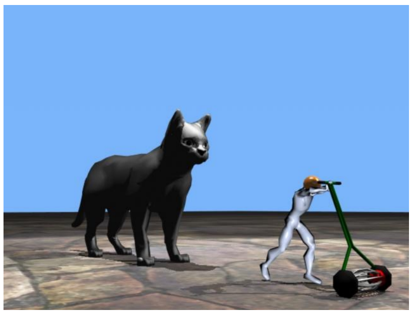

Figure 1 3D scene generated by WordsEye [7] for the text "The lawn mower is 5 feet tall. John pushes the lawn mower. The cat is 5 feet behind John. The cat is 10 feet tall."

- we demonstrate a system able to identify explicit objects using WordNet
- we demonstrate a system able to infer implicit objects using Wikipedia

1.2 Paper outline

This paper is organised as follows. We detail related work in section 2. In section 3, we discuss the limitations of the existing work and propose a solution that uses Wikipedia and WordNet. This is followed in section 4 by a description of the Python implementation used in the experiments. In the experiments in section 5 we show that explicit objects can be identified and implicit objects can be inferred using our technique. Finally, we conclude the paper and propose future work in section 6.

2 Related work

Often cited as the first TTSCS, SHRDLU [22] allows users to enter natural language commands, which are reissued to a virtual robot arm which moves blocks around in a small 'blocks world'. SHRDLU has a basic vocabulary of objects and properties, and basic semantics for interpreting them and the environment to which they apply [20].

Following SHRDLU came two microworld systems (highly restricted sets of objects or ideas that operate in accordance with a limited set of rules). The Clowns of Microworld [18] processes user-commands concerning spatial descriptions of clowns and their actions to generate a simple diagram of a scene. Similarly, NALIG (Natural Language Image Generation) [1] takes sequences of descriptions regarding the spatial relations between objects to generate a scene.

WordsEye [7] generates 3D scenes from user descriptions. WordsEye works by converting a parse tree to a dependency representation, which is then converted into a semantic representation. Depiction rules convert the semantic representations into a set of low-level depictors (representing 3D objects, poses, spatial relations, colour attributes, etc). Conflicting and illegal depictors are removed, and a final 3D scene is generated.

Carsim [8, 2, 9] takes textual police traffic accident reports and generates 3D animations from them. It uses a number of natural language techniques such tokenising, part-of-speech tagging, noun group detection, and domain-specific multi-words to parse a report's text into a formal representation outlining what happened. Using temporal relations between the events, Carsim is able to generate and then animate the scene.

Other TTSCSs include: [6] which also focuses on spatial relationships and uses restricted keywords such as *in*, *on*, and *at* to manipulate spatial arrangements of existing objects within a scene; [10] who propose an automatic text illustration system which automatically extracts keywords from the text and matches these keywords to a database of pictures; [14] and [23] who both present systems to generate pictures from a natural language text; and [4] who focus on automatically translating natural language patient instructions into pictures.

Sproat [19] looks into inferring the environment in a TTSCS. For example, for the sentence "John was eating breakfast", Sproat tries to infer that the scene is taking place in a kitchen. The system is able to match events such as 'wash clothes' to 'laundry room' and 'wash hands' to 'bathroom'. However, the system does not attempt to populate matched rooms with objects you would expect to find within them.

To our knowledge, the only other work that focuses on inferring implicit relations is [5], who use an existing 3D scene database to create a spatial knowledge-base with priors on the hierarchy of objects in scenes. Specifically, they parse the input text into a scene template, which places constraints on what objects must be present and the relationships between them. They then use the priors from the spatial knowledge-base to expand the scene template by inferring additional implicit constraints. These implicit constraints are then used to select objects from an object dataset.

3 Limitations and proposed solutions

Existing TTSCS tend to focus on extracting specific information from text, then matching this information to a pre-determined and often basic vocabulary of objects and properties. Once matched, most systems emphasise the spatial relationships of objects to position them within a scene. To our knowledge, only one other paper [5] considers objects not explicitly mentioned within the text. In addition, the majority of existing systems, including [5], ignore the dynamic nature of natural language, and often the only dynamic content within these systems are the heavily constrained, and in some cases, simple narratives supplied by the user.

We propose that by first identifying explicit objects, we can then infer implicit objects. For example, for the sentence "She placed the pen on the desk", we suggest that we can infer a chair by identifying the desk. However, exploring the semantic properties of a desk suggests nothing concerning its relationship to a chair. A desk has draws, legs, and is a type of table, but there is nothing in its semantic properties referring to its relationship to a chair. Humans make this inference using world-knowledge. For a machine to perform this inference, we need a machine-readable source of world-knowledge. In the following section, we describe an approach to this problem using Wikipedia. This approach differs from [5] since we use a dynamic source of world-knowledge which is adaptable to changes in natural language, whereas [5] use a static database of objects collected from 133 small indoor scenes.

4 Implementation

We now describe the Python implementation of our system. We use the Natural Language Toolkit [3] for the natural language processing tasks.

Note that in this paper, we ignore the problem of scene detection, i.e. we take a book already parsed into scenes as input. See [12] for work on scene detection.

4.1 Object identification

The input to the system is a collection of scenes from a book, where each scene is a string. For each scene, we identify nouns by tokenising and part-of-speech (POS) tagging [11] the text. Plurals are singularised and word frequencies aggregated, so that each scene is reduced to a set of nouns and associated frequencies.

Only extracting nouns is insufficient to identify objects. For example, in the sentence "I went to the shop on Wednesday", the word *Wednesday* is a noun, but not a physical object. We use WordNet [15], a lexical semantic dictionary, to identify physical objects. WordNet places words into one or more logical categories, of which there are 45, including the following:

- noun.artifact: denoting man-made objects
- noun.communication: denoting communicative processes and contents
- noun.location: denoting spatial position
- noun.person: denoting people

We use the *noun.artifact* category to decide whether a word refers to a physical object.

A reader might question the need to POS tag the text if we use WordNet to identify objects. POS tagging is necessary because words of the *noun.artifact* category can appear in a scene but not as nouns. For example, in the sentence, "The politician wishes to table an amendment to the proposal", the word *table* is of the type *noun.artifact* but is used as a verb, not an object.

As mentioned, WordNet places words into one or more logical categories. If a word is in multiple categories, then WordNet orders the categories by estimated frequency of use[1]. For example, the word *mouth* has eight entries in WordNet (entries 3-7 are omitted for brevity):

1. noun.body: mouth, oral cavity, oral fissure, rima oris (the opening through which food is taken in and vocalizations emerge)
2. noun.body: mouth (the externally visible part of the oral cavity on the face and the system of organs surrounding the opening)
8. noun.artifact: mouth (the opening of a jar or bottle)

Here, the *noun.artifact* category is the least likely interpretation. But how do we know which interpretation corresponds to the sense of the word in the scene? To investigate this, we checked WordNet for the 20 most frequent nouns in a chapter of a book (described in section 5.1). We found that the correct interpretation of the word was in the first three results returned by WordNet for 19 of the 20 words. Therefore, we only use the first three results to decide whether a word refers to a object. Developing a more sophisticated approach, such as using word-sense disambiguation, remains a topic for future work.

As a final step, we aggregate synonyms (identified using WordNet) to end with a set of words and frequencies that potentially refer to objects in a scene.

4.2 Object inference

Having identified potential objects in a scene, the task is to infer implicit objects. To do this, we look at the corresponding Wikipedia page for each potential object in a scene. For example, if we identify the word *chair* as being a potential object, we look at the

[1] https://wordnet.princeton.edu/wordnet/man/wn.1WN.html

corresponding Wikipedia page for the word *chair*[2]. We extract the contents of the page using the Wikipedia export pages[3] and repeat the object identification steps described in section 4.1. Following this step, we have a set of words and frequencies that potentially relate to objects in a scene.

4.3 Object ranking

We assign each object a score to reflect how important it is in a scene. This ensures that less common items are represented in the scene. For example, we might assign the word *clarinet* a higher score than the word *chair* because a *clarinet* is a less common object. The tfidf (term-frequency inverse-document-frequency) weighting scheme [13] is used in information retrieval and text mining to rank documents based on their similarity to a query. This scheme assigns each word a value which increases proportionally to the number of times the word appears in a document, but is offset by the frequency of the word in the corpus, which helps to discriminate against words that generally appear more often than others. Specifically, in the tfidf scheme, the term frequency component *tf* is calculated by summing the instances of a term t in a document d. The document frequency component *df* is computed by dividing the total number of documents n by the number of documents that term t occurs in d, then taking the log of this value. These two values are multiplied to give the weighting for a term. We use this weighting scheme to rank the objects in a scene. Specifically, we say that a scene and its inferred objects form a document, and that the collection of all scenes forms the corpus. The term frequency component is the frequency of an object in a document (scene), and the document frequency component is the number of documents (scenes) in which each object appears. Using this scheme, we run the object identification and object inference steps for all scenes in our dataset and rank the objects using their tfidf score.

5 Experiments

The experiments investigate whether our system is able to identify and infer objects in a scene.

5.1 Materials

In this paper, we work with unlabelled data, i.e. we are given a book and the task is to identify and infer objects without any supervision. Therefore, the results must be evaluated using intuition. Experimentation using labelled data is left for future work.

For the choice of textual material, we experimented with several texts, all in the public domain. However, we found that the system often inferred anachronisms. Therefore, for the following experiments, we use Corey Doctorow's 'Little Brother'[4], chosen specifically due to its modern content, e.g. it includes objects such as *radio* and *Xbox*.

5.2 Methods

For the experiments, the input is chapter 7 from Corey Doctorow's 'Little Brother' manually parsed into scenes. The output is a list of potential objects for each scene ordered by their

[2] http://en.wikipedia.org/wiki/Chair
[3] http://en.wikipedia.org/wiki/Special:Export
[4] http://craphound.com/littlebrother/download/

tfidf score.

5.3 Results

We now describe the results for one scene from chapter 7 of Corey Doctorow's 'Little Brother'. Results for other scenes are omitted for brevity.

5.3.1 Object identification

For the scene, we identified the following objects:

bed, computer, jail, camp, picture, telephone, room, ceiling, projector, wall, filter, screen, microscope, bag, radar, keyboard

Most objects are present in the scene, but there are exceptions. For the sentence "If anyone knew how to keep our butts out of jail, it would be him", we incorrectly identified that the word *jail* is an object in the scene. This example highlights a problem with using the tfidf weighting scheme because the word *jail* was not in any other scene, and was thus given a high weighting. In addition, we missed several objects. For example, the scene starts with the sentence "I hooked up my Xbox as soon as I got to my room". However, we did not detect the word *Xbox* an object because this word does not exist in WordNet. Neither do many commonplace technological objects such as *iPhone*, *iPod*, *Wii*, *Mac*, etc. This is a limitation of the work, i.e. we are restricted to objects in WordNet.

5.3.2 Object inference

Having identified explicit objects, the next task is to infer implicit objects. Table 1 shows a sample of the results for this step where the column header is an explicit object identified in the scene and row items are inferred objects, displayed in descending order by their tfidf score. The results for *keyboard* are particularly good, with *computer*, *laptop*, and *joystick* all inferred. In comparison, the results for *telephone* are less impressive, with objects such as *coil*, *telegraph*, and *candlestick* inferred.

keyboard	**telephone**	**computer**	**screen**	**bed**
key	microphone	machine	computer	mattress
computer	receiver	microprocessor	panel	box
typewriter	coil	telephone	tube	frame
screen	handset	transistor	cathode	bedding
keypad	bell	keyboard	stand	mortise
laptop	telegraph	disc	keyboard	cot
joystick	candlestick	webcam	telephone	bedpost

Table 1 Words and inferred related words ordered in descending order by tfidf

6 Conclusions and future work

In this paper, we have described a technique which uses Wikipedia and WordNet to identify explicit objects and infer implicit objects from textual descriptions of a scenes from a book. No known existing literature has attempted this. Our results show potential, and we are able to infer implicit objects such as *keyboard* and *screen* by identifying explicit objects such as *computer*.

The main issue complicating the work is the ambiguity of natural language, which is a common problem across all areas of natural language processing [11]. For example, when identifying explicit objects from text, we did not disambiguate if an object was in the scene or if it was just being discussed. Future work should integrate more sophisticated natural language processing techniques to resolve such ambiguity, such as using word-sense disambiguation techniques. In addition, we described an unsupervised learning technique. In future work, it would be interesting to investigate whether a semi-supervised machine learning approach would work.

References

1 Giovanni Adorni, Mauro Di Manzo, and Fausto Giunchiglia. Natural language driven image generation. In *Proceedings of the 10th international conference on Computational linguistics*, pages 495–500. Association for Computational Linguistics, 1984.

2 Ola Åkerberg, Hans Svensson, Bastian Schulz, and Pierre Nugues. Carsim: an automatic 3d text-to-scene conversion system applied to road accident reports. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 2*, pages 191–194. Association for Computational Linguistics, 2003.

3 Steven Bird, Ewan Klein, and Edward Loper. *Natural language processing with Python*. " O'Reilly Media, Inc.", 2009.

4 Duy Bui, Carlos Nakamura, Bruce E Bray, and Qing Zeng-Treitler. Automated illustration of patients instructions. In *AMIA Annual Symposium Proceedings*, volume 2012, page 1158. American Medical Informatics Association, 2012.

5 Angel X Chang, Manolis Savva, and Christopher D Manning. Semantic parsing for text to 3d scene generation. *ACL 2014*, page 17, 2014.

6 Sharon Rose Clay and Jane Wilhelms. Put: Language-based interactive manipulation of objects. *Computer Graphics and Applications, IEEE*, 16(2):31–39, 1996.

7 Bob Coyne and Richard Sproat. Wordseye: an automatic text-to-scene conversion system. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 487–496. ACM, 2001.

8 Sylvain Dupuy, Arjan Egges, Vincent Legendre, and Pierre Nugues. Generating a 3d simulation of a car accident from a written description in natural language: The carsim system. In *Proceedings of the workshop on Temporal and spatial information processing-Volume 13*, page 1. Association for Computational Linguistics, 2001.

9 Richard Johansson, David Williams, Anders Berglund, and Pierre Nugues. Carsim: a system to visualize written road accident reports as animated 3d scenes. In *Proceedings of the 2nd Workshop on Text Meaning and Interpretation*, pages 57–64. Association for Computational Linguistics, 2004.

10 Dhiraj Joshi, James Z Wang, and Jia Li. The story picturing engine—a system for automatic text illustration. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP)*, 2(1):68–89, 2006.

11 Dan Jurafsky and James H Martin. *Speech & language processing*. Pearson Education India, 2000.

12 Marie Louise Lingaya. Automatic scene extraction from natural language text. Master's thesis, Nottingham Trent University School of Science and Technology, UK, 2008.

13 Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge, 2008.

14 Rada Mihalcea and Chee Wee Leong. Toward communicating simple sentences using pictorial representations. *Machine Translation*, 22(3):153–173, 2008.

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

16 K Onstad. Pixar gambles on a robot in love. `http://www.nytimes.com/2008/06/22/movies/22onst.html`, 2008. Accessed: 25-06-2014.

17 J Reimer. Cross-platform game development and the next generation of consoles. `http://arstechnica.com/old/content/2005/11/crossplatform.ars/7`, 2005. Accessed: 25-06-2014.

18 Robert F Simmons. The clowns microworld. In *Proceedings of the 1975 workshop on Theoretical issues in natural language processing*, pages 17–19. Association for Computational Linguistics, 1975.

19 Richard Sproat. Inferring the environment in a text-to-scene conversion system. In *Proceedings of the 1st international conference on Knowledge capture*, pages 147–154. ACM, 2001.

20 Daniel Allen Tappan. *Knowledge-based spatial reasoning for automated scene generation from text descriptions.* PhD thesis, New Mexico State University, 2004.

21 Meng Wang. Research on the relationship between story and the popularity of animated movies. Master's thesis, Purdue University, United States, 2012.

22 Terry Winograd. Procedures as a representation for data in a computer program for understanding natural language. Technical report, DTIC Document, 1971.

23 Xiaojin Zhu, Andrew B Goldberg, Mohamed Eldawy, Charles R Dyer, and Bradley Strock. A text-to-picture synthesis system for augmenting communication. In *AAAI*, volume 7, pages 1590–1595, 2007.