The TaleMaker database of mixed-initiative co-created stories

Mijael R. Bueno Perez
M.R.BuenoPerez@tudelft.nl
Delft University of Technology
The Netherlands

Rafael Bidarra
R.Bidarra@tudelft.nl
Delft University of Technology
The Netherlands

ABSTRACT

Several datasets of stories and text have been proven useful for a variety of research fields. Yet, many of these datasets have suffered from the burden of being manually authored and/or annotated, affecting their size and potential to grow. To overcome this problem, we propose a novel database of stories collected from TaleMaker, an online multiplayer game that facilitates the playful co-creation of a story in order to eliminate the tedious task of authoring and annotating a dataset of stories. TaleMaker's database relational schema provides a simple story representation, in which stories are named and clearly annotated. A story is composed of a sequence of plot points, each with several slots (e.g. action, character, location) filled with sense-annotated tokens (words) associated with a WordNet synset. In this paper, we describe in detail the database schema of TaleMaker's stories repository. In addition, we suggest some of the potential applications of this repository of stories, including fostering research in fields such as story generation, narrative world generation, and word sense disambiguation.

CCS CONCEPTS

• Information systems → Relational database model; **• Human-centered computing → Collaborative and social computing theory, concepts and paradigms**.

KEYWORDS

story co-creation, storytelling game, collaborative storytelling, story database, stories repository

ACM Reference Format:
Mijael R. Bueno Perez and Rafael Bidarra. 2022. The TaleMaker database of mixed-initiative co-created stories. In *FDG '22: Proceedings of the 17th International Conference on the Foundations of Digital Games (FDG '22), September 5–8, 2022, Athens, Greece.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3555858.3555910

1 INTRODUCTION

Several datasets of stories have been proposed to foster research in various fields, including story generation, and natural language processing (NLP). For example, *ROCStories* and *VIST* datasets have been extensively used for *story reasoning* and *story generation* [5, 9]. While other datasets, such as *SemCor* and *One Million Sense-Tagged Instances* (OMSTI) with sense-annotated words, have been shown to be useful for *word sense disambiguation* (WSD) [8, 12]. Despite the usefulness of these datasets, they have suffered from the burden of being manually authored and annotated, therefore limiting their size and their potential to grow. In particular, the lack of sense-annotated data has hindered many state-of-the-art WSD systems based on supervised learning [13].

As a viable alternative to the tedious task of putting together a dataset of stories, we propose the mixed-initiative co-creation process of TaleMaker, an online multiplayer game that facilitates collaborative story authoring through play [2]. Moreover, the collected output produced by TaleMaker consists of original and clearly annotated stories. The stories are named and stored in a simple story representation consisting of a sequence of plot points with several slots (e.g. action, character, location) filled with sense-annotated tokens (words) associated to a WordNet synset. In this paper, we describe the relational database schema that facilitates easy storage and retrieval of the stories created with TaleMaker. Furthermore, we discuss some of the potential applications that would benefit from an ever-growing collection of stories produced by the creative efforts of TaleMaker's player-base.

2 DESIGN

In this section, we briefly describe the story co-creation process of TaleMaker and the design details of our database of stories.

2.1 TaleMaker Story Co-Creation Overview

TaleMaker is an online multiplayer game designed to facilitate playful co-creation of a story through collaborative and competitive gameplay. The game begins by first providing a story prompt, where a player is randomly selected to add two custom characters and set a location for the story. This allows players to iteratively form creative associations among these elements, to unfold the story. Later on, when a majority of players decide to change location of

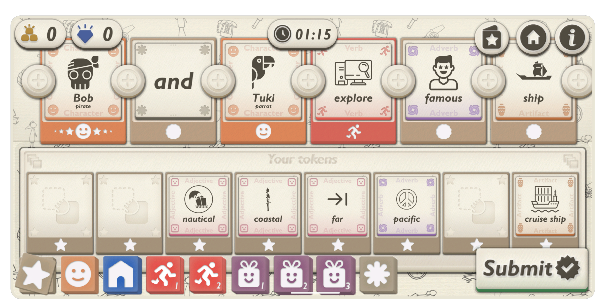

Figure 1: TaleMaker sentence composition: players create a plot point by dragging and dropping tokens from their collection (below) into the appropriate slots (above).

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

FDG '22, September 5–8, 2022, Athens, Greece

ACM ISBN 978-1-4503-9795-7/22/09.
https://doi.org/10.1145/3555858.3555910

Figure 2: Visual story retrieved from TaleMaker's database of stories.

the story, the process is repeated by another player, who introduces two new characters and a new location.

In each turn, players collect tokens (words) and use them to compose a sentence involving their available characters. From a players' perspective, they compose a sentence, but internally we represent it as a *plot point*, a sentence-like structure of an important event of the story. Figure 1 shows a game screen of a sentence being composed.

When all players have completed their sentence, they submit it for a voting round in which they chose the winning sentence, which is then appended to the story. Players compose sentences and prompt the story several times until the majority decides to end the story. At the end, all players propose a title for their story, and a final voting round decides the definitive story title.

2.2 Story Representation

The output stories in TaleMaker have a simple representation consisting of a title and a sequence of plot points (see Figure 2). A plot point describes an action undertaken by a character in a particular location of the story. It is composed of a sequence of slots each holding a token. In Figure 3, we illustrate the constituents of a plot point as retrieved from TaleMaker's database of stories.

A slot is a space that holds a token and describes the role that the token plays in the plot point. TaleMaker uses five slot types: *action*, *actor*, *character*, *location* and *free*. An *action* slot only holds tokens associated with a verb synset (e.g., *walk*, *play*, *read*), which stands for the action in the plot point. The *actor* slot is for the character which undertakes the action of the plot point. The *character* slot is any additional character participating in the plot point, for

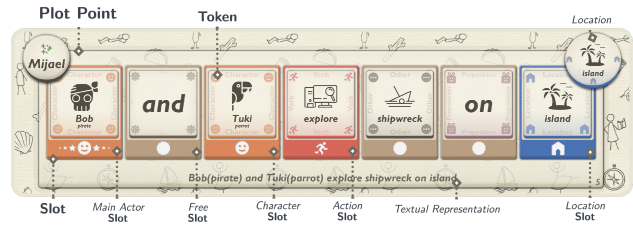

Figure 3: Visual representation of a plot point retrieved from TaleMaker's database of stories

example, a character affected by an action (e.g. Sally expels Bob from the classroom). In addition, the *location* slot holds the location of the plot point, to be used in case a player wishes to express a sentence with the location (e.g. Bob arrives to the castle). Internally, the location is always embedded in the representation of the plot point. Lastly, a *free* slot is reserved for all other tokens other than characters, actions or locations.

A token is a specific entity involved in the plot point; this could be a word associated to a physical entity such as a person, object or location, or to other categories, such as abstract nouns, verbs, adjectives and adverbs. To avoid any ambiguity in their meaning, tokens are sense-annotated with WordNet synsets. A synset is a well defined set of words (synonyms) which expresses the same meaning. We use the synsets from WordNet, a lexical database that provides the meaning of English words [7]. TaleMaker facilitates that players use the proper meaning of a token that corresponds to what they are trying to convey in their sentences. We make sure that players understand this by explicitly showing the type of word (e.g. artifact, plant, animal, verb, etc) and a definition directly retrieved from WordNet. Moreover, tokens of characters and locations are chosen by players when they are selected to create a story prompt. These tokens also contain a custom name written by the players (e.g. a *pirate* named *Bob*).

2.3 Database Schema

The relational schema used for TaleMaker's database of stories is shown in Figure 4. This schema consists of four core tables: *Stories*, *PlotPoints*, *Slots* and *Tokens*. Each row in the *Stories* table contains the *name* and the *story ID*. We provide additional information such as the number of plot points and tokens within the story.

Plot points of all stories are stored within the *PlotPoints* table. In each row, we associate a plot point with a *story ID* and a *token ID* of the location. In addition, the plot point holds an *index* of its position in the sequence of plot points of the story, as well as the *turn* of the game when it was created. Only winning plot points have an *index*, yet, the *turn* can be used to extract other competing plot points that were created at the same time. Furthermore, the *winner* field stores a boolean flag that indicates whether the plot point won the turn. Additionally, the *votes* field contains the number of votes that the plot point received during the voting round and the *text* field stores a textual representation of the plot point.

The *Slots* table contains rows describing the slots of all plot points. Each slot has a *plot point ID*, as well as the ID of the token it holds. In addition, we use an *index* to store the position of the slot

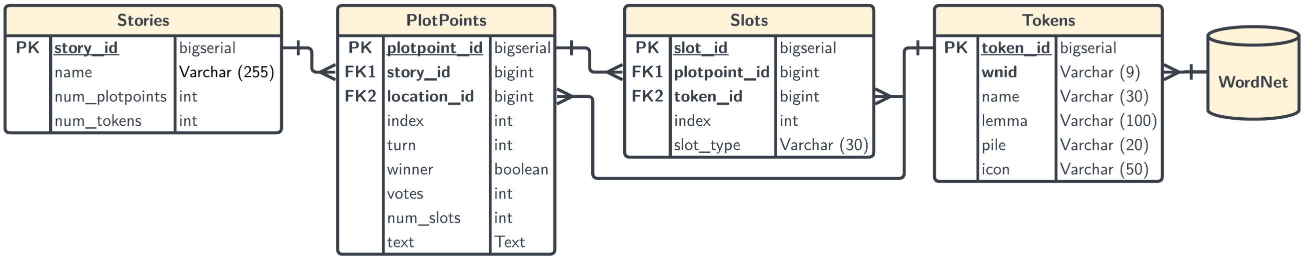

Figure 4: TaleMaker's Database Schema

Table 1: Plot points of an exemplary story in the database. The rows in blue correspond to the winning plot points that have been added to the story, the rows in red correspond to the submitted plot points which were not selected for the story.

story_id	location_id	index	turn	winner	votes	num_slots	text
5	11129 (Island)	0	0	True	2	5	Bart(pirate) find Tuki(parrot) in island
5	11129 (Island)	.	0	False	1	8	Bart(pirate) and Tuki(parrot) go to island by sailboat
5	11129 (Island)	1	1	True	2	4	Tuki(parrot) have nasty behavior
5	11129 (Island)	2	2	True	2	5	Tuki(parrot) threaten Bart(pirate) at island
5	11129 (Island)	.	3	False	1	7	Bart(pirate) adhere to superior behavior of Tuki(parrot)
5	11129 (Island)	3	3	True	2	5	Tuki(parrot) nest on Bart(pirate) sailboat
5	11324 (Church)	.	4	False	1	4	Luis Alberto(astronaut) awaken at Padresito(catholic church)
5	11324 (Church)	4	4	True	2	8	Bart(pirate) take Tuki(parrot) to Padresito(catholic church) because of aggression
5	11324 (Church)	5	5	True	2	7	noisy Tuki(parrot) plan Bart(pirate) death at Padresito(catholic church)
5	11324 (Church)	.	7	False	1	8	Romulo(cook) see purposeful aggression of Tuki(parrot) at Padresito(catholic church)
5	11324 (Church)	6	7	True	2	5	Romulo(cook) make eye contact with Tuki(parrot)
5	11324 (Church)	.	8	False	1	7	Romulo(cook) threaten pride of Tuki(parrot) at Padresito(catholic church)
5	11324 (Church)	7	8	True	2	4	Tuki(parrot) proclaim historical Padresito(catholic church)
5	11567 (House)	.	9	False	0	7	Tuki(parrot) and Bella(parrot) want absolution at Priest house(house)
5	11567 (House)	8	9	True	2	8	Tuki(parrot) threaten Theodorus(priest) to crack Priest house(house) with base hit
5	11567 (House)	.	9	False	1	4	Bella(parrot) nest at Priest house(house)
5	11567 (House)	9	10	True	2	7	Tuki(parrot) and Bella(parrot) ruin Priest house(house) with collaboration
5	11567 (House)	10	11	True	2	10	Theodorus(priest) give absolution for Tuki(parrot) and Bella(parrot) aggression at Priest house(house)
5	11567 (House)	.	11	False	1	9	panicky Theodorus(priest) burn evil Tuki(parrot) and Bella(parrot) with enchilada
5	11567 (House)	11	12	True	2	7	panicky Bart(pirate) and Romulo(cook) go to Priest house(house)
5	11567 (House)	.	13	False	0	9	Bart(pirate) panic and hyperventilation because Tuki(parrot) hand drill at Priest house(house)
5	11567 (House)	12	13	True	2	10	Bart(pirate) and Romulo(cook) threaten evil Tuki(parrot) and Bella(parrot) with pipe wrench
5	11567 (House)	.	13	False	1	11	Bart(pirate) and Romulo(cook) see Theodorus(priest) kidnapping Tuki(parrot) and Bella(parrot) at Priest house(house)
5	11567 (House)	13	14	True	2	10	Tuki(parrot) and Bella(parrot) go away from Bart(pirate) Romulo(cook) and Theodorus(priest)
5	11567 (House)	.	14	False	0	7	Bart(pirate) want terrorization of Bella(parrot) at Priest house(house)

within the sequence of slots of a plot point. A *slot type* is a string that describes the role of the token it holds (e.g. a character).

In the *Tokens* table, the *wnid* field stores the WordNet ID of a synset entry from the WordNet lexical database. This ID is a string composed of one character for part of speech followed by 8 digits (e.g., *n02958343* for the synset of *car*). Also, the *name* field stores a custom name specifically set for characters and locations and the *lemma* field represents the word displayed in the token (e.g. *doctor*, *cat*, *ball*). The *pile* field stores the category of the token (e.g. *adjective*, *artifact*, *plant*, etc.) and the *icon* field represents the URL of the image that illustrates the token.

2.4 Implementation

We designed and implemented this schema in PostgreSQL, a free and open-source relational database. SQL commands can be used to extract content from the database. For example, in Table 1, we show the plot points of a story extracted with the following command:

```
SELECT story_id, location_id, index, turn,
       winner, votes, num_slots, text
FROM talemaker_plotpoints
WHERE story_id = 5
ORDER BY turn;
```

Results are sorted by the *turn* field to retrieve the sequence of plot points as they were created. The plot points with an index (blue rows) correspond to those that form part of the story, and the plot points without an index (red rows) are those that were not selected to be part of the story. The textual representation shows the *lemma* and *name* of each token placed in a sequence.

3 APPLICATIONS

In this section, we discuss a few potential applications that would benefit from the use of TaleMaker's stories extracted from the database. We facilitate access to this data through a repository [3].

3.1 Story Generation

Story generation is the challenging problem of selecting or completing a sequence of events that can be interpreted as a story. Stories in the TaleMaker's database can be accessed by story generators, such as *Say Anything* and *Creative Help*, since these systems require a large collection of stories to propose the next plot point (or sentence) for a story [10, 11]. Each story in our database contains multiple plot points per turn, both the selected one and the ones not selected, along with the number of votes assigned to each as relevance information. This information can be used to provide authors with alternative sentences for their stories.

Moreover, datasets such as *ROCStories* and *VIST* have been used as training and validation set for story generators evaluated by the *Story Cloze Test* [9]. In this test, a 4-sentence story is presented to the generator, and it must pick the 'right' ending among two options. The stories produced by TaleMaker are useful as additional training and validation data. In TaleMaker's stories, the 'right' sentence is the winner sentence of a turn, and the 'wrong' option is any sentence that was not selected to become part of the story.

Additionally, our story representations and annotations provided by TaleMaker's stories are compatible with the event representation used in *event2event* and *event2sentence* systems [6]. Similarly, our representation uses WordNet synsets for tokens involved in the events, as well as slots that further describe each token as an action, character, or location. The main difference is that this does not need to be automatically annotated, as the players themselves curate a story, through the creation of characters, locations and the plot points of the story.

3.2 Narrative World Generation

A *narrative world* (NW) is an environment which supports enacting a given story [1]. For a system to be capable of generating NWs, it must be able to derive objects from associations to actions, characters, locations, and objects involved in a story. In order to form such associations, a wide range of contextual knowledge is needed. As an example, *TaleForge* has been designed as an interactive prototype of a mixed initiative approach that assists designers in populating an NW with relevant entities for a given story [4]. Currently, *TaleForge* derives contextual associations from an embedding of synset vectors learned from a large dataset of photo captions with sense-annotated tokens. Therefore, the narrative contexts in TaleMaker's database of stories would be valuable to such a tool. In particular, collected stories contain clearly annotated plot points, which provide contextual associations between characters, actions, locations, and the object(s) involved.

3.3 Word Sense Disambiguation

WSD is an ongoing and long-standing problem in NLP. This problem consists of determining the correct meaning of a word within a given sentence. A WSD system determines the correct meaning of a polysemous word by examining its context. For example, the meaning of the word 'cell' differs when used in the sentences 'Bob goes to jail and is locked up in a cell' and 'Bob studies the cell in a biology class'. In order to derive the correct meaning, WSD systems need to model a large number of examples of sense-annotated data of contextual knowledge. SemCor and OMSTI datasets have been used extensively as sources of sense-annotated exemplars identified with WordNet synsets. SemCor consists of approximately 200,000 sense-annotated tokens, while OMSTI is composed of almost 1 million. TaleMaker's database of stories can be utilized as an additional and valuable resource in this regard. Stories contain sense-annotated tokens which can be utilized to improve the sense identification capabilities of WSD systems.

4 CONCLUSION

In this paper, we described the story representation and the relational database schema of the repository of stories created with TaleMaker. Additionally, we explained some potential applications that would benefit from this collection. At the time of writing, the player base is still small, and there are not yet many stories created with TaleMaker. However, as the player base expands, we anticipate that TaleMaker's database will gather increasingly more and more original stories created on a daily basis. TaleMaker's collected stories are available at the game's website [3], from where the current version of the game can also be downloaded. We therefore warmly welcome the research community both to use the stories for any (non-commercial) application, and to play the game, thus contributing to extend the database with new stories.

REFERENCES

[1] J. Timothy Balint and Rafael Bidarra. 2022. Procedural generation of narrative worlds. submitted for publication.

[2] Mijael R. Bueno Pérez and Rafael Bidarra. 2022. Mixed-initiative story co-creation with TaleMaker. In *17th International Conference on the Foundations of Digital Games*. ACM, Athens, Greece, 13 pages.

[3] Mijael R. Bueno Pérez and Rafael Bidarra. 2022. TaleMaker Game and Data. https://mijaelb.github.io/talemaker/.

[4] Mijael R. Bueno Pérez, Elmar Eisemann, and Rafael Bidarra. 2021. A synset-based recommender method for mixed-initiative narrative world creation. In *Interactive Storytelling - 14th International Conference on Interactive Digital Storytelling (ICIDS 2021) (Lecture Notes in Computer Science, Vol. 13138)*, Alex Mitchell and Mirjam Vosmeer (Eds.). Springer, Tallinn, Estonia, 13–28. https://doi.org/10.1007/978-3-030-92300-6_2

[5] Ting-Hao Kenneth Huang, Francis Ferraro, Nasrin Mostafazadeh, Ishan Misra, Aishwarya Agrawal, Jacob Devlin, Ross Girshick, Xiaodong He, Pushmeet Kohli, Dhruv Batra, C. Lawrence Zitnick, Devi Parikh, Lucy Vanderwende, Michel Galley, and Margaret Mitchell. 2016. Visual Storytelling. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, San Diego, California, 1233–1239. https://doi.org/10.18653/v1/N16-1147

[6] Lara J. Martin, Prithviraj Ammanabrolu, Xinyu Wang, William Hancock, Shruti Singh, Brent Harrison, and Mark O. Riedl. 2018. Event Representations for Automated Story Generation with Deep Neural Nets. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence (AAAI'18/IAAI'18/EAAI'18)*. AAAI Press, New Orleans, Louisiana, USA, Article 106, 8 pages.

[7] George A Miller. 1995. WordNet: a lexical database for English. *Commun. ACM* 38, 11 (1995), 39–41.

[8] George A. Miller, Martin Chodorow, Shari Landes, Claudia Leacock, and Robert G. Thomas. 1994. Using a semantic concordance for sense identification. In *Proceedings of the Workshop on Human Language Technology (HLT '94)*. Association for Computational Linguistics, Plainsboro, New Jersey, USA, 240–243. https://doi.org/10.3115/1075812.1075866

[9] Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A Corpus and Evaluation Framework for Deeper Understanding of Commonsense Stories. https://doi.org/10.48550/ARXIV.1604.01696

[10] Melissa Roemmele and Andrew S Gordon. 2015. Creative Help: A story writing assistant. In *Interactive Storytelling - 8th International Conference on Interactive Digital Storytelling (ICIDS 2015) (Lecture Notes in Computer Science, Vol. 9445)*, Henrik Schoenau-Fog, Luis Emilio Bruni, Sandy Louchart, and Sarune Baceviciute (Eds.). Springer, Copenhagen, Denmark, 81–92. https://doi.org/10.1007/978-3-319-27036-4_8

[11] Reid Swanson and Andrew S. Gordon. 2008. Say Anything: A massively collaborative open domain story writing companion. In *Interactive Storytelling*, Ulrike Spierling and Nicolas Szilas (Eds.). Springer, Berlin, Heidelberg, 32–40.

[12] Kaveh Taghipour and Hwee Tou Ng. 2015. One million sense-tagged instances for word sense disambiguation and induction. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*. Association for Computational Linguistics, Beijing, China, 338–344. https://doi.org/10.18653/v1/K15-1037

[13] Zhi Zhong and Hwee Tou Ng. 2009. Word sense disambiguation for all words without hard labor. In *Twenty-First International Joint Conference on Artificial Intelligence*. IJCAI, Pasadena, CA, USA, 1616–1621.