The AIIDE-17 Workshop on
Intelligent Narrative Technologies
WS-17-20

Towards End-to-End Natural Language Story Generation Systems

Josep Valls-Vargas,[1] Jichen Zhu,[2] Santiago Ontañón[1]
[1]Computer Science, [2]Digital Media
Drexel University
Philadelphia, PA 19104
josep.vallsvargas@drexel.edu, jichen.zhu@drexel.edu, santi@cs.drexel.edu

Abstract

Storytelling and story generation systems usually require knowledge about the story world to be encoded in some form of knowledge representation formalism, a notoriously time-consuming task requiring expertise in storytelling and knowledge engineering. In order to alleviate this authorial bottleneck, in this paper we propose an *end-to-end computational narrative system* that automatically extracts the necessary domain knowledge from corpus of stories written in natural language and then uses such domain knowledge to generate new stories. Specifically, we employ *narrative information extraction* techniques that can automatically extract structured representations from stories and feed those representations to an *analogy-based story generation* system. We present the structures we used to connect two existing computational narrative systems and report our experiments using a dataset of Russian fairy tales. Specifically we look at the perceived quality of the final natural language being generated and how errors in the pipeline affect the output.

Introduction

Given its significance in human experience, storytelling and narrative analysis have long been of interest to AI researchers (Andersen and Slator 1990). Computational narrative is an emergent field of research at the intersection of traditional narratology, artificial intelligence, natural language processing and cognitive science that focuses on methods to algorithmically analyze, model, and generate narratives.

Within the field of computational narrative a significant body of work focuses on storytelling and story generation. These systems usually require a significant amount of domain knowledge encoded in some form of structured knowledge representation formalism, a notoriously time-consuming task requiring expertise in both storytelling and knowledge engineering. This well-known *authorial bottleneck problem* (Riedl and Sugandh 2008) pushes related approaches from content generation in games to exploit knowledge representation databases (Lenat 1995). Moreover, researchers often develop ad-hoc solutions to suit their particular needs and are domain-dependent (in toy domains) and non-reusable, which leads to multi-year efforts invested in knowledge bases for systems that cannot be reused[1].

[1]As observed by Scott Turner, author of Minstrel: https://grandtextauto.soe.ucsc.edu/2007/10/30/scott-turner-on-minstrel/

In order to alleviate this authorial bottleneck problem, in our previous work (Valls-Vargas, Zhu, and Ontañón 2014; 2015) we proposed the use of natural language processing techniques to automatically extract structured narrative information from text. Additionally, there is growing body of literature that studies narrative information extraction with the purposes of narrative analysis (Elson 2012; Finlayson 2012; Reagan et al. 2016). However, there is little work on connecting these two lines of work (narrative information extraction and story generation) in order to achieve what we call in this paper *end-to-end computational narrative systems*: systems that extract narrative information directly from natural language text and can use it to then generate new narratives.

The idea of end-to-end computational narrative systems would enable authors to provide input to story generation systems using natural language, both easing the development of the required narrative models and enabling them to exploit the existing body of literature. Towards this goal, in this paper we present our work connecting two existing systems that can automatically process input and generate output stories in natural language text. Specifically, we use *Voz* (Valls-Vargas, Zhu, and Ontañón 2014; 2015), a narrative information extraction system, and *Riu* (Ontañón and Zhu 2011), an analogy-based story generation system. We selected these two systems because compatibility between the knowledge structures that *Riu* requires and the structures that *Voz* can extract from text.

The remainder of this paper is organized as follows. In the next section we present some related work on computational narrative. Then we introduce both the story generation and the narrative information extraction systems used. In the next section we describe how we bridged them and finally, we present our experiments on a dataset of Russian fairy tales and report on the performance of our system in terms of the feasibility of the system and the quality of the output. Finally, we discuss the current shortcomings for end-to-end story generation systems and propose future work.

Related Work

There are two main bodies of research within the field of computational narrative: narrative analysis and generation.

Narrative analysis focuses on the analysis of narratives with the purpose of story understanding or to study existing literature or validate narrative theories. Examples include comparing the works of different authors (Elson 2012), validating narrative theories (Finlayson 2012), analyzing trends and recurring patterns (such as dramatic arcs in popular fiction (Reagan et al. 2016), story structure (Finlayson 2012) or character interactions (Chaturvedi et al. 2015; Elson 2012)). Because of limitations in current NLP techniques and ambiguities in the text, fully automatic approaches are limited and in certain cases are replaced or complemented with manual or semi-automatic annotation tools (Finlayson 2012; Elson 2012). There have been efforts to standardize the process of annotating natural language (Malec 2001; Mani 2012) but with no consensus on representation formalisms, researchers develop ad-hoc solutions to suit their particular needs which are non-reusable and expensive to generate.

Narrative generation explores computational creativity and storytelling techniques and ranges from story generation (Meehan 1981; Ontañón and Zhu 2011) to *interactive storytelling* (Zhu and Ontañón 2014) work such as interactive fiction. Two major families of techniques used within story generation are *planning-based* and *analogy-based* (among others, such as simulation). *Planning-based* approaches are the most prevalent in storytelling and story generation applications (Young et al. 2013). These include logic, graph and plan-like representations of a *story space* or the rules defining a simulation or agent behavior (Theune et al. 2003; Riedl and Young 2004; Riedl 2009). Despite some popular representations (such as extensions to PDDL), because of the specialization of the different systems, these tend to be ad-hoc and not reusable. *Analogy-based* approaches and other similarity-based approaches such as story merging or structure mapping are a well-known but underexploited in the current literature (Gervás 2009; Zhu and Ontañón 2014). These approaches use a wide variety of frame-based representations of a story such as description logics or semantic networks but differ from the previous in terms of operation and their expressive affordances (Turner 1993; Gervás et al. 2005; Ontañón and Zhu 2011). In the work presented in this paper, we use the *Riu* story generation system (Ontañón and Zhu 2011) and its *Story Analogies through Mapping* (SAM) algorithm. For the purposes of this paper, the main difference between planning-based and analogy-based approaches is that the former requires domain knowledge in the form of planning operators, and the latter in the form of frame-based structures.

To the best of our knowledge, there is little work on end-to-end computational narrative systems. A couple recent exceptions are the work by Swanson et al. (Swanson and Gordon 2012) and Li et al. (Li et al. 2013). Swanson proposes an open-domain interactive storytelling system where snippets from a massive corpus of text are proposed to continue a given story. The system uses *case-based reasoning* techniques and a surface similarity function to retrieve text snippets. In Li's work, they automatically process a set of crowd-sourced short reports describing a given theme (e.g., bank robbery). They extract Schankian script-like structures (Chambers and Jurafsky 2008) from each and compile them into a *plot graph* that joins common events. These graph-like structures are related to plot points or planning operators used to describe a story space. Despite currently not eliminating human intervention, this approach could be used to exploit existing text as in Swanson's work.

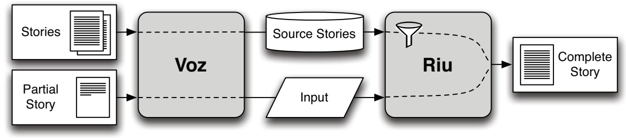

Figure 1: Overview of our end-to-end story generation system. The knowledge structures required by *Riu* are generated automatically by *Voz* from natural language.

Riu

Riu is an interactive story generation system that uses the *Story Analogies through Mapping* (SAM) algorithm (Ontañón and Zhu 2011). Analogy-based story generation emulates the human cognitive process of analogy-making by identifying similarities and a mapping between two domains. This general idea is that if two domains are similar in a certain way, they are likely to be similar in another and we should be able to find a mapping which we can exploit to transfer knowledge from one another.

Specifically, *Riu* operates over a repository of complete stories. The story generation process starts with a given story fragment as input *target story*. Riu then identifies which of the complete stories in the repository has a stronger analogy with the provided story fragment (*the source story*). *Riu* then calculates an analogical mapping between the target and the source stories, and completes the target story by analogy with the source. *Riu* requires all the input stories to be encoded using a frame-based symbolic knowledge representation formalism. In all the existing literature on the *Riu* system, each of the stories in the repository and the given story fragment have been manually authored. In this work, we will study the performance of *Riu* when the representation of such stories is automatically generated using the *Voz* system, as illustrated in Figure 1.

Riu's Story Representation

Riu uses the *Story Analogies through Mapping* (SAM) algorithm which internally uses the *Structure Mapping Engine* (SME) algorithm described by Falkenhainer et al. (Falkenhainer, Forbus, and Gentner 1989). Thus, given SEM is a symbolic analogical mapping algorithm, it requires a symbolic representation of the stories.

Riu represents stories as *scenes* and *phases*. A *scene* is a small encapsulated segment of a story involving a limited set of characters, actions and locations. Furthermore, each *scene* is broken into several *phases* (which represent specific story states the relations between the characters and props in the story at that particular state, and the actions the characters are performing). For the purposes of this paper, we will see a story as a scene which is in turn represented as a sequence of phases (more complex stories are represented in *Riu* as collections of scenes, but we will focus on single-scene stories in this paper). Each of these phases contains two representations: a *Computer-Understandable Description* (CUD) and a *Human-Understandable Description* (HUD). The CUD is a symbolic, frame-based representation of the phase that includes the different entities present in the phase and a graph-like structure that defines links between them using *expressions*. The original authors of *Riu* reported experiments using different representations in the CUD and have shown that the representation formalism has a substantial impact on computational analogy (Zhu and Ontañón 2014).

On the other hand, the HUD is a collection of annotated natural language phrases and sentences. The CUD and HUD are linked, so that during the analogical reasoning process, SAM can manipulate the CUD and use the HUD to realize the final text of the story. An example of such representation is shown in Figure 2. For more detail on these representations, we refer the reader to the previously published work on *Riu* (Ontañón and Zhu 2011).

Analogy-based Story Generation

Given an incomplete input target story, *Riu* compares it with the stories in its repository and selects the most similar story (the source story). In order to evaluate similarity, *Riu* uses both the *Computer-Understandable Description* (CUD) and the *Human-Understandable Description* (HUD) to compute structural and surface similarities (Zhu and Ontañón 2014). Then *Riu* invokes the SAM algorithm for finding an analogical mapping between source and target stories and generating the output story by completing the partial target story[2].

SAM takes two input parameters: a source story S from the story repository and a target T in place from the given story segment. Note that both S and T are story representations encoded as a sequence of phases, each with a CUD and HUD. It generates a set of all possible consistent injective mappings M between S and T. Then, SAM finds a mapping $m* \in M$ that maximizes a numerical similarity between the entities and relationships defined in the CUD for the phases in S and T using the mapping $m*$. With this mapping $m*$, SAM can construct a new story R by applying the mapping $m*$ to the phases of the source S, and then bringing them to T. For each element in the CUD that is brought from S to T, the corresponding elements from the HUD are also brought to T (applying the appropriate transformations given the analogical mapping) in order to realize the output in natural language.

Voz

Voz is an information extraction pipeline that exploits off-the-shelf natural language processing (NLP) tools, machine learning and domain knowledge in order to automatically process stories (Valls-Vargas, Zhu, and Ontañón 2014; 2015). *Voz* uses machine learning and Vladimir Propp's narrative theory (Propp 1973) to process and perform several tasks that extract different layers of narrative information from text. *Voz* implements a feedback loop where extracted narrative information is fed-back and used to improve the performance of earlier tasks in the pipeline. Specifically, *Voz* handles the following narrative information extraction tasks: mention extraction (where it identifies mentions to entities in the text), coreference resolution (where several mentions to the same entity are grouped), character identification (where mentions to characters are separated from other entities), entity classification (where the rest of the entities are classified in several classes such as locations, objects, etc.), verb argument identification (where entities mentioned in a verb's subject and object arguments are identified), role identification (where the narrative role of each character is identified; e.g. hero, villain, etc.), dialog participant identification (where speakers and intended listeners of quoted text are identified), and, narrative function identification (where Propp's narrative functions are identified in segments of text). For more detail on these, we refer the reader to the previously published work on *Voz* (Valls-Vargas, Zhu, and Ontañón 2016; 2014; 2015).

[2]Available online: https://sites.google.com/site/santiagoontanonvillar/software

Connecting Voz and Riu

In this section we describe our contribution on how to map the output of *Voz* to the dual representations used by *Riu*.

There are three distinctive parts to consider: segmentation (breaking a story in phases), entities (from the text's mentions) and expressions (that link entities and provide structure for the SME algorithm to use). We generate these by automatically transforming and annotating the output of *Voz* described in the previous section.

Segmentation. *Riu* uses two levels of segmentation; a story is divided in scenes and each scene is divided in phases. Given the significant amount of subjectivity involved in determining the scenes and phases of a story, we used the notion of *narrative functions* as defined by Propp (1973) for this purpose. Thus, we use these functions to segment each story in 5 phases: phase 1) the introductory and setup functions (alpha – lambda in Propp's work); phase 2) villainy/misfortune, mediation, counteraction and departure (A/a, B, C, ↑ in Propp's work); phase 3) the main development of the story (including functions D – K); phase 4) return (including ↓, Pr, Rs, o, L – T, Ex); and phase 5) conclusions (including U, W and any narrator's remarks after that). This segmentation will be used for both the CUD and the HUD representations. The first two phases make up the introductory scene and in the experiments reported in this paper we use only these first two phases mainly because of performance concerns. Note that currently, *Voz* can only identify narrative functions in stories that have been previously segmented into chunks of text by hand (since it basically predicts which function is expressed in each chunk of text). In our experimental evaluation we report results with the functions automatically identified by *Voz* and with manual annotations but both use the previous manual segmentation of the text in chunks. The overall accuracy for predicting narrative function in *Voz* is 0.287 but the accuracy for the first and second phases used to segment our experiments is 0.534 and 0.487 respectively.

Entities. One of the tasks *Voz*'s development has focused on is the extraction of *referring expressions* or *mentions*

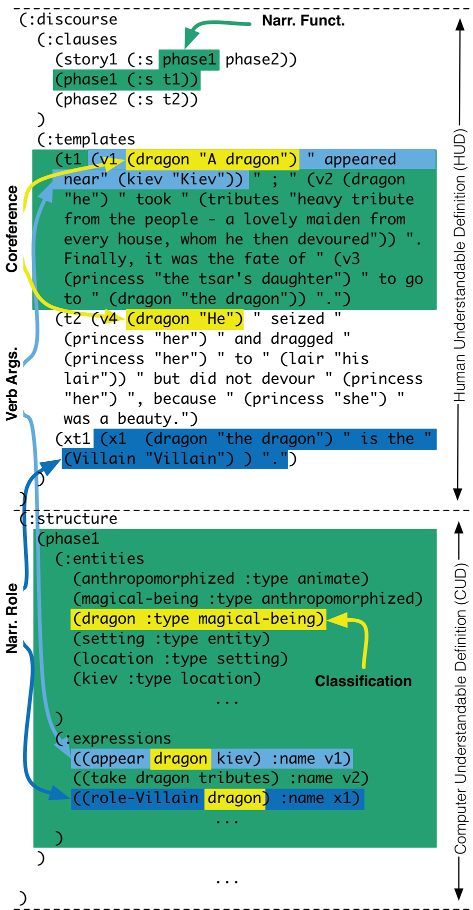

Figure 2: Example input used by *Riu*. Bold labels identify information from *Voz*. Different colors illustrate linked parts.

from the text and their coreference information (a.k.a. coreference resolution) which groups mentions into unique entities (Valls-Vargas, Zhu, and Ontañón 2015). Furthermore, the SME algorithm uses a taxonomy for the entities in the CUD for which we use the entity classification labels *Voz* provides. *Voz* classifies each entity into different classes using a taxonomy inspired on Chatman's existents (Chatman 1980) in order to provide a deeper structure for the SME. These include existents (e.g., characters: male, magical being, etc.; settings: locations and time anchors) and happenings (e.g., events such as rain). This information is linked in the HUD where the original text for each mention is used and annotated with their coreference group information. *Voz* has perfect recall of mentions but some false positives yield an accuracy of 0.954. The accuracy for classifying into the 15 classes of our taxonomy is 0.535. In terms of coreference, on our corpus of Russian fairytales, *Voz* achieves the following performance measured using 3 common metrics; F1 score for MUC: 0.932; CEAF$_e$: 0.208; B^3: 0.538.

Expressions. The SME algorithm within *Riu* favors deep, structural similarity during the analogy process. Besides the aforementioned taxonomy, this structure is represented in the CUD using logical predicates. Specifically, we use:

1) *Verb and verb argument information* encoded as triplets representing an interaction between two entities, the subject or actor and the object or receiver. This results in an expression for each verb in the CUD and an annotation over the span of text where the verb and mentions appear in the HUD. Note that we define a *null* entity placeholder for intransitive verbs where there is no object. Additionally, to provide a closed language for the SME, we automatically abstract the verbs and group them in 191 semantic classes based on the Levin verb classification (Levin 1993) used for the expression's predicates. *Voz* has an accuracy of 0.807 for verb identification, 0.229 when considering the verb and arguments.

2) *Narrative role information* identified for the extracted characters is used to add additional expressions in the CUD for characters (a subset of the entities). For the HUD we add the template "{character} is the {narrative role}" so it can be used by SAM if necessary during text realization. The accuracy of *Voz* for identifying characters is 0.931 and then 0.394 for identifying their narrative roles; but that accuracy is 0.540 and 0.622 for the first two phases.

3) *Narrative function information* is used to add a layer of structure on top of verb expressions that span complete sentences within a phase. For each narrative function within a phase; and for each sentence within the span of the function from which a verb expression has been extracted; an expression is added in the CUD linking the narrative function and the expression of the root verb of the sentence. The narrative function is also annotated in the HUD for each sentence.

Figure 2 shows an example story representation that highlights the aforementioned parts, the links between the CUD and the HUD, and illustrates how these map to the output from *Voz*. Note that for clarity, this example uses human-readable symbols such as *dragon* and *take* whereas *Voz* uses equivalent symbols such as *E1* or *Levin-02-1*.

Experimental Evaluation

In this section we present our experimental evaluation. In this work we seek to answer the following questions:

1. Can we build a completely automated end-to-end computational narrative system?
2. How is the quality of the generated stories affected by the knowledge structures automatically generated by *Voz* with respect to using the ground truth annotations on the stories to generate these structures?

Dataset

For our experimental evaluation we assembled a corpus of 20 Russian folk tales translated to English. We have annotations for each of the 20 stories and each of the tasks in the information extraction pipeline (Valls-Vargas, Zhu, and Ontañón 2016). In the following experimental section we report the results in two scenarios: 1) generating *Riu*'s story representation formalism using our complete automated narrative information extraction pipeline (*Voz*), and, 2) instead of using *Voz*, generating *Riu*'s knowledge structures directly from the ground truth annotations on the text (*GT*).

Running Riu with the Output of Voz

In order to answer the first question, using the output of *Voz*, we are specially interested in whether the current performance of *Voz* suffices to generate *Riu*'s required input.

We used a *leave-one-story-out* evaluation procedure, where we provided the first phase of one story to *Riu* as the target story, and the remaining 19 stories as the repository of stories to use as source stories. The expected output of *Riu* is to complete the target story with a second phase that continues the given input story.

Considering the two scenarios described in the previous section (*Voz* and *GT*), we launched a total of 40 experiments (one per story and scenario). Out of the 40, 22 completed successfully within a few minutes, 7 exhibited errors that prevented Riu from being executed and 11 timed out within the time allocated (48 hours). Inspecting the experiments with errors, we observed that 2 stories had segmentation issues that yield a phase without contents and therefore, 4 experiments cannot be completed in either the fully automated (*Voz*) scenario or the annotated (*GT*) scenario. Additionally, there are 3 experiments that cannot be completed because the automated scenario is unable to extract any expressions for the initial phase (that is, there are no roles identified nor verbs with arguments). Inspecting the input files for the experiments that didn't finish we found that in the *GT* scenario, the segmentation was causing some stories to have unexpectedly longer phases with a great number of entities and expressions. Since SME uses a systematic search approach for computing the analogy mappings, these phases cause a combinatorial explosion that cannot be properly handled. In the *Voz* scenario we observed a similar problem but additionally, because of some coreference errors, a large number of different entities are present in the output. As part of our future work, we plan to modify *Riu* to use a more efficient computational analogy algorithm.

Evaluating Riu's Output

We manually inspected the output of the 22 experiments that completed successfully. In this section we report the major trends we observed. Note, minor spacing and punctuation corrections were made to the output reproduced in this section and square brackets and ellipsis are used to provide interpretation remarks and shorten some story fragments. In the following examples, the span in *italics* corresponds to *Riu*'s generated continuation of the given introduction.

Successful Output. We found several of the experimental runs where *Riu* found a plausible mapping and generated output that could be considered successful. With some considerations mentioned in the following sections, we consider 6 out of 10 stories in the *Voz* scenario and 6 out of 12 stories in the *GT* scenario to have a plausible output using the top ranked mapping or any of the 3 top ranked mappings when tied. 5 out of these 6 stories are the same and share characteristics that make them suitable for being combined with other stories (i.e., they have a generic introduction). This is one example (from the *Voz* scenario):

> A dragon appeared near Kiev; he took heavy tribute from the people - a lovely maiden from every house, whom he then devoured. Finally, it was the fate of the tsar's daughter to go to the dragon. *He [the king] just didn't know; there was no one suitable for the job the job. Then he remembered the seven Simeons [seven brothers who offered their services to the king]. Summoning them, he commissioned them as soldiers, with the assignment to bring him that princess.*

Natural Language Generation Issues. Currently, the natural language generation (NLG) component of *Riu* relies on SAM's mapping and performs the identified replacements directly on the *Human-Understandable Description* (HUD). These replacements lack the variety and use of pronouns found in the original text (e.g., Ivan, he, me). Consider this original text fragment:

> Well, Ivan undertook to kill that Snake, so he said to his father, "Father make me a mace five poods in weight."

The following fragment from the *Voz* scenario illustrates the aforementioned problem:

> *Well, A fox undertook to kill that Snake, so A fox said to his his father, "Father make A fox a mace five poods in weight."*

Notice a repeated "his his". This is an error caused by the mention identification not properly capturing the full span of the mention and replacing "father" with "his father".

Nonsensical Output. Some of the original stories use strange rhetoric figures and constructs that may be surprising for a western audience not familiar with Russian and Slavic folktales. Still, in the output of the system we found several instances that do not have continuity and with nonsensical mappings, despite these mappings exhibiting a high score by the SME. Inspecting these mappings we found that besides errors in the narrative information extraction the lack of depth of our *Computer-Understandable Description* (CUD) structure was assigning analogies between stories that a human would not consider.

> An old man lived with his his old wife; they had a daughter and a little son. "Daughter, daughter," said the mother, "we are going to work. [...] Be careful , watch over your your little brother, do not leave the house." The parents went away and the daughter forgot what they had told her her; she put her her brother on the grass beneath the window, ran out into the street, and became absorbed in games. *The prince flew straight into the royal palace, turned into a goodly youth, and asked the palace guards: "Will your your king take me into his service?" "Why should he not take such a goodly youth?" they answered. Thus he entered the service of that king and lived in his palace for one week, then a second, then a third . [...]*

Scoring, Ranking and Bias Issues. Sometimes the input given to *Riu*, that is, the first phase in some stories, is either too short or *Voz* is not able to extract sufficient information to compile a rich CUD. This situation cascades into problems retrieving candidate sources for analogy and the analogical reasoning itself. For example, consider the following two continuations from the *GT* scenario that got tied scores for one target partial story.

This soldier got permission to go on leave. *The tsar's word was law. This soldier was banished and This soldier left home not knowing where to go. This soldier walked for a long time; at last This soldier arrived in another kingdom, presented This soldier to the king, and asked to be taken into his service.*

This soldier got permission to go on leave. *Well, This soldier undertook to kill that Snake, so This soldier said to his his father, "Father make This soldier a mace five poods in weight".*

We observe this problem in the same 5 stories in both the *Voz* and *GT* scenarios. On the other hand, some stories in the database feature longer/richer structures which bias the algorithm towards using them more often and scoring them higher. For example, in the following case, the dragon (the villain) is intended to be "punished", so *Riu* identifies the act of trying to cook the dragon on the stove as a punishment:

A dragon appeared near Kiev; he took heavy tribute from the people - a lovely maiden from every house, whom he then devoured. Finally, it was the fate of the tsar's daughter to go to the dragon. *So Alenka heated the stove hot, ever so hot, and said to A dragon, "Come here and sit on this shovel!"*

In this second case, Prince Ivan or Ivanshko, in the original story, is being (unfairly) punished for his action and again, the same continuation is used despite there being more plausible punishments available:

For many years a certain tsar had kept under lock and key a little peasant [who was a cunning wizard ...] Prince Ivan, the tsar's son who was still a little boy [... set the prisoner free by accident ...] *So Alenka heated the stove hot, ever so hot, and said to Ivashko, "Come here and sit on this shovel!".*

This Alenka continuation is used (in some instances tied) in 6 stories in the *GT* scenario and 4 in the *Voz* scenario.

Identifying Current Shortcomings

Throughout this section we showed examples from the two scenarios evaluated in our experiments: executing *Riu* with input automatically generated with *Voz*, and executing *Riu* with input generated from the ground truth annotations in the stories. When generating short story snippets (as done in our experiments), we don't observe relevant differences between these two scenarios. We attribute this to the fact that the structures in the CUD are quite similar in terms of depth (and contain the same kinds of expressions). However, with longer examples or trying to generate full stories (not reported in this paper), several shortcomings arise.

Coreference. In the experiments reported in this paper, the second phase is usually short enough that the coreference graph in the second phase is relatively small and the substitutions made by *Riu* make sense. In some longer instances although none selected in the final output, we noticed that missing links in the coreference graph cause *Riu* to miss replacements. This was more evident if we try completing introductory segments using full stories (all 5 phases).

Segmentation. Errors in the function identification task used for segmentation cause phases to lack key information or include out-of-place information that belongs in other phases. Moreover, 2 of the stories in our dataset do not seem to conform to the expected Propp structure (and hence they have some empty phases), so these issues may indicate that there might have been an issue with the segmentation scheme used in our experiments.

Shallow Structure. A recurrent problem we observe is due to the lack of structure in the CUD. Despite often due to errors in *Voz*, the choice of the mapping yields shallow structures in the CUD even when using the annotated stories. As mentioned earlier, the representation formalism has a substantial impact on computational analogy (Zhu and Ontañón 2014). Adding additional layers of annotation (e.g., force dynamics) that could be extracted by a narrative information extraction system should have a positive effect on the analogical reasoning process.

Dataset Size. The number of stories in our dataset seems to be insufficient, specially given that some stories are very specific and different from the rest which implies there are no plausible sources for analogy. The approach described in this paper could be generalized and switching to a larger corpus should be possible given an information extraction system capable of extracting higher-level narrative information that could be used to generate *Computer-Understandable Description* structures such as the ones described.

Conclusions and Future Work

With the long-term goal of bridging the gap between narrative analysis and narrative generation work, in this paper we presented our work towards a text-based, end-to-end computational narrative system. Specifically, we presented our work on automatically mapping the output of *Voz*, a narrative information extraction pipeline, into *Riu*, an analogy-based story generation system. Our experimental evaluation shows promising results but indicates that there are still a number of open research problems to address. Namely, issues with the performance of narrative information extraction and the limitations of the shallow representation formalism produced by our mapping. On the other hand, we also observed that some errors in *Voz* have a smaller affect than anticipated.

In our future work we would like to extract and encode information that has been shown in the past to improve the performance of *Riu* such as *force dynamics* structures (Zhu and Ontañón 2014) and generalize *Voz* so it can handle a broader domain of input stories. We would also like to investigate mapping the output of other narrative information extraction and story generation systems in order to generalize a framework for end-to-end computational narrative.

Acknowledgements

Special thanks to Mark A. Finlayson for kindly sharing his dataset of translated and annotated Russian stories.

References

Andersen, S., and Slator, B. M. 1990. Requiem for a theory: the 'story grammar' story. *Journal of Experimental & Theoretical Artificial Intelligence* 2(3):253–275.

Chambers, N., and Jurafsky, D. 2008. Unsupervised learning of narrative event chains. In *2008 Anniversary Meeting of the Association for Computational Linguistics*, volume 94305, 789–797.

Chatman, S. B. 1980. *Story and discourse: Narrative structure in fiction and film*. Cornell University Press.

Chaturvedi, S.; Srivastava, S.; Daume III, H.; and Dyer, C. 2015. Modeling dynamic relationships between characters in literary novels.

Elson, D. K. 2012. *Modeling Narrative Discourse*. Ph.D. Dissertation, Columbia University.

Falkenhainer, B.; Forbus, K. D.; and Gentner, D. 1989. The structure-mapping engine: Algorithm and examples. *Artificial intelligence* 41(1):1–63.

Finlayson, M. A. 2012. *Learning narrative structure from annotated folktales*. Ph.D. Dissertation, Massachusetts Institute of Technology.

Gervás, P.; Díaz-Agudo, B.; Peinado, F.; and Hervás, R. 2005. Story plot generation based on cbr. *Knowledge-Based Systems* 18(4):235–242.

Gervás, P. 2009. Computational Approaches to Storytelling and Creativity. *AI Magazine* 49–62.

Lenat, D. B. 1995. CYC : A Large-Scale Investment in Knowledge Infrastructure. *Communications of the ACM* 38(11):33–38.

Levin, B. 1993. *English verb classes and alternations: A preliminary investigation*. University of Chicago press.

Li, B.; Lee-Urban, S.; Johnston, G.; and Riedl, M. 2013. Story generation with crowdsourced plot graphs. In *27th AAAI Conferece on Artificial Intelligence*.

Malec, S. A. 2001. Proppian structural analysis and xml modeling. *Computers, Literature and Philology (CLiP 2001)*.

Mani, I. 2012. *Computational Modeling of Narrative*, volume 5 of *Synthesis Lectures on Human Language Technologies*. Morgan & Claypool Publishers.

Meehan, J. 1981. Tale-spin. *Inside computer understanding: Five programs plus miniatures* 197–226.

Ontañón, S., and Zhu, J. 2011. The sam algorithm for analogy-based story generation. In *7th AIIDE Conference*.

Propp, V. 1973. *Morphology of the Folktale*. University of Texas Press.

Reagan, A. J.; Mitchell, L.; Kiley, D.; Danforth, C. M.; and Dodds, P. S. 2016. The emotional arcs of stories are dominated by six basic shapes. *EPJ Data Science* 5(1):31.

Riedl, M. O., and Sugandh, N. 2008. Story planning with vignettes: Toward overcoming the content production bottleneck. In *Joint International Conference on Interactive Digital Storytelling*, 168–179. Springer.

Riedl, M. O., and Young, M. R. 2004. An intent-driven planner for multi-agent story generation. In *3rd International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 1*, 186–193. IEEE Computer Society.

Riedl, M. O. 2009. Incorporating authorial intent into generative narrative systems. In *AAAI Spring Symposium: Intelligent Narrative Technologies II*, 91–94.

Swanson, R., and Gordon, A. S. 2012. Say anything: Using textual case-based reasoning to enable open-domain interactive storytelling. *ACM Transactions on Interactive Intelligent Systems* 2(3):16:1–16:35.

Theune, M.; Faas, S.; Heylen, D.; and Nijholt, A. 2003. The virtual storyteller: Story creation by intelligent agents.

Turner, S. R. 1993. *Minstrel: a computer model of creativity and storytelling*. Ph.D. Dissertation, University of California at Los Angeles.

Valls-Vargas, J.; Zhu, J.; and Ontañón, S. 2014. Toward automatic role identification in unannotated folk tales. In *10th AIIDE Conference*.

Valls-Vargas, J.; Zhu, J.; and Ontañón, S. 2015. Narrative hermeneutic circle: Improving character role identification from natural language text via feedback loops. In *24th IJCAI*, 2517–2523.

Valls-Vargas, J.; Zhu, J.; and Ontañón, S. 2016. Error analysis in an automated narrative information extraction pipeline. *IEEE TCIAIG* PP.

Young, R. M.; Ware, S.; Cassell, B.; and Robertson, J. 2013. Plans and planning in narrative generation: a review of plan-based approaches to the generation of story, discourse and interactivity in narratives. *Sprache und Datenverarbeitung, Special Issue on Formal and Computational Models of Narrative* 37(1-2):41–64.

Zhu, J., and Ontañón, S. 2014. Shall i compare thee to another story? - an empirical study of analogy-based story generation. *IEEE TCIAIG* 6(2):216–227.