Evolutionary Plot Arcs for a Series of Neurally-Generated Episodes

Pablo Gervás and **Gonzalo Méndez**
Facultad de Informática
Universidad Complutense de Madrid
Madrid, 28040, Spain

Abstract

Neural generators produce fluent text for short stories but face difficulties for longer narrative structures. Knowledge-based generators produce coherent narrative structures but have problems generating fluid text. Series built of episodes involve an overarching plot arc for the series and specific plots for each episode. The present paper explores a combination of evolutionary generation of season plot arcs–based on knowledge resources for plot structures–with neural generation of episode descriptions–prompted with fragments of season plot arc.

Introduction

The last decade of research into story generation has shown a progressive shift of interest from knowledge-based solutions to neural solutions. The knowledge-based solutions used in the past had proven capable of generating story outlines that obeyed expectations on narrative structure, but their results were usually rendered as clumsy texts that lacked fluency and nuance. Recent neural solutions based on self-attention mechanisms have proven capable of producing fluent and convincing text that correspond to prompts that outline the content desired for the story. However, they face difficulties in generating longer stories that maintain a coherent plot over their duration. The present paper explores the possibilities of combining the two types of solution into a single procedure in which an overarching narrative structure is generated using knowledge-based solutions, and the result is used to prompt a neural generator based on a Large Language Model (LLM).

Previous Work

Three different topics have been reviewed to inform the work presented in this paper: academic analyses of the narrative structure of series, an existing evolutionary solution for generating a plot outline for a story and neural solutions for story generation.

Structure of Serial Narrative

Serial narratives have been formally defined as "a continuing narrative distributed in installments over time" (O'Sullivan, 2019). Their defining feature is the fact that they are delivered in segments. Narratives of this type have a long history, probably starting way back with oral storytellers but well documented from 19th century serial novels (Okker, 2003) to contemporary television series (Fox, 2014). Existing analyses of the narrative structure of serial narrative may have relevant insights on how plot outlines for such narratives might be constructed computationally.

Garin (2017) identifies a number of important issues relevant to serial narrative. One is the difference between series where the episodes are almost interchangeable in terms of where in the sequence they are presented and series where each episode only makes sense at a particular point in the sequence. This relates to whether each episode is *self-contained*–the episode has its own beginning, development and end– or *serialised*–plots carry over the scope of a season. Narratives built of self-contained episodes are often referred as *series* and narratives that are serialised are referred as *serials*. However, different authors (Allrath and Gymnich, 2005; Garin, 2017) argue that these two possible types of episode act as opposing force that pull the concept of a series in different directions. This point is tightly connected with the issue of closure, explicitly addressed by Allrath and Gymnich (2005). In the particular case of TV series, seasons are designed in principle so they have the potential to run endlessly–at least as long as they achieve satisfactory audience ratings. The season is never intended to have a full closure, so it is important that individual episodes offer some kind of closure, though it must never be a definitive one. These considerations also lead to a tendency towards episodes where a set of multiple plot lines are explored in parallel (Allrath and Gymnich, 2005).

With respect to chronology, alterations to the expected timeline are identified as frequent in series (Allrath and Gymnich, 2005) and Rennett (2011) explores the use of time manipulation devices–such as flashbacks, flash forward and time travel–as tools for building the narrative discourse of a series as a combination of views on different moments in time.

Evolutionary Story Generation

Gervás (2022) defines an evolutionary algorithm for constructing stories as a combination of a set of axes of interest–small sequences of events connected by plot-relevant causality and sharing characters in roles important to the plot (Gervás, 2019).

An *axis of interest* of AoI is a data structure that lists a number of plot atoms that represent events that should be connected – say a villainy that is then avenged– in the chronological order in which they should appear in the story – villainy is presented at the start of the story and avenged towards the end –, and that encodes as a set of variables the narrative roles – villain, victim, avenger – of the characters in the events that must be instantiated correctly for the connection to be coherent. The representation for the plot of a story will be a *story draft*, which includes plot atoms from different axes of interest, stitched together by having the characters of the story take on the various roles in the different plot atoms.

As fitness function, the evolutionary algorithm uses a metric that computes the degree of satisfaction of knowledge-based constraints on role-sharing and appropriate sequencing across the plot atoms involved. Table 1 shows examples of two AoIs–each listing its constituent plot atoms–, an example of simple plot built by combining them in a coherent manner, and an example of constraint that applies to them.

Neural Story Generation

A wide range of neural and non-neural approaches to story generation have been developed prior to the advent of the new generation of Large Language Models (LLMs) (Alhussain and Azmi, 2021). However, these LLMs have increased the quality of the text rendered by story generation systems, and have posed new challenges like the generation of long stories (Mikhaylovskiy, 2023) for which it is difficult to generate consistent texts. It is for this reason that state of the art approaches still combine different techniques and rely on LLMs to render the final text of the story.

One such approach is the combination of automated planning to create coherent and believable narratives with LLMs that use these narratives as an scaffolding to generate richer texts in a variety of domains (Simon and Muise, 2022). In this work, after manually extracting characters, objects, and locations from the story source, the authors create a domain and a problem encoding, and the output of the planner is fed to an LLM to generate the narrative. The authors report that almost all nouns (characters, objects, and locations) and verbs (actions) of the plan are reflected in the generated story, and the resulting narrative is more coherent than stories that are generated using only plain text prompts to the LLM.

Planning is also used by Xie and Riedl (2024) to generate suspenseful stories as the authors claim that LLMs are still unreliable when it comes to suspenseful story generation. They propose an iterative-prompting-based planning method that is grounded in cognitive psychology and narratology. This method works in a fully zero-shot manner and does not rely on any supervised story corpora. The authors have used two different LLMs to generate the final versions of the stories, ChatGPT and LLaMa2, and they report better results in both cases using the hybrid method rather than using the LLMs alone.

Other authors (Elliott, 2024) use a hybrid model that combines a symbolic AI cognitive-appraisal model of emotions embodied in the Affective Reasoner (AR) and ChatGPT. The emotion and narrative structure is generated by AR techniques and then fed to ChatGPT to add world knowledge and language structure. The authors report that the resulting stories are polished and cohesive, but the basic structural elements remain under computational control, and they also present a series of hybrid-generated stories that illustrate how the emotions of characters, their actions and their narrative perspectives remain consistent after being processed by ChatGPT to generate the stories.

Finally, some authors have also tested the use of LLMs as a co-creation tool (Yuan et al., 2022). In this work, the authors describe Wordcraft, a text editor in which users collaborate with an LLM to write stories. Results show that LLMs enable co-creation in different ways, such as engaging in open-ended conversations about the story, respond to the user's requests (such as "rewrite this text to be more Dickensian"), and generate suggestions to unblock writers in the creative process.

Generate Outlines for a Season of Episodes with an Overarching Plot

The task of constructing a season of episodes interconnected by an overarching plot faces a number of specific challenges beyond the generation of a story to be delivered as a single installment.

It shares features with the generation of a simple story in the sense that both the overarching plot and the specific plot for each episode should be valid instances of stories. However, there are two additional challenges to be addressed that arise specifically in the process of constructing the stories to serve as plots for the different episodes. First, the development of the overarching plot must be distributed in an even manner over the sequence of episodes. Second, the plot for each episode should mainly start within the episode and it should end with the episode.

There are two possible approaches for the construction of an overarching plot for the season. One approach involves establishing it before the episodes are drafted and then using it to inform the process of constructing the episodes. Another approach involves progressive construction of the episodes, with the overarching plot emerging as a result of the sequencing of the episodes. Whereas human authors seem capable of doing it in either way, we have opted for the first approach because it is computationally simpler.

This paper explores the construction of a season of episodes in the following terms:

- an outline for the overarching plot of the season is selected
- the number of episodes that the season will have is established
- specific spans of the season plot are assigned to each episode
- for each episode so determined, the episode is fleshed out with additional content that defines a specific subplot for the episode

(a)

AoI =	ABDUCTION
`Kidnapping`	`abductor` = x `abducted` = y
`Rescue`	`abducted` = y `rescuer` = z

AoI =	CALLTOACTIONREWARD
`Call`	`called` = x `caller` = y
`Reward`	`rewarded` = x

(b)

ABDUCTION	`Kidnapping`	(`abductor`=*villain*,`abducted`=*victim*)
CALLTOACTIONREWARD	`Call`	(`called`=***hero***,`caller`=*sender*)
ABDUCTION	`Rescue`	(`abducted`=*victim*,`rescuer`=***hero***)
CALLTOACTIONREWARD	`Reward`	(`rewarded`=***hero***)

(c)

AoIs involved	`Abduction`	(relation)	`CallToActionReward`
Shared roles	`hero`	=	`hero`
Sequencing constraints	`Abduction`	<	`CallToAction`
	`Rescue`	>	`CallToAction`
	`Rescue`	<	`Reward`

Table 1: Representational elements for a simple plot: (a) shows the axes of interest (AoIs) for ABDUCTION and CALLTOACTIONREWARD, (b) shows a story draft that combines them into a coherent plot, (c) shows a constraint to the effect that: the hero of both AoIs must be the same, the abduction must take place before the call to action, the rescue must take place after the call to action and before the reward. Co-occurrence in variable names within AoIs is relevant, so the character that plays `abducted` in the `Kidnapping` must be the same as the one that plays `abducted` in the `Rescue`. In the story draft, variable names for the different AoIs have been instantiated to relevant narrative roles in the story.

A number of different configurations are tested:

- evolutionary knowledge-based algorithm used to generated overarching plot + evolutionary algorithm is used to flesh out the subplot for each episode
- evolutionary knowledge-based algorithm used to generated overarching plot + neural generator is used to flesh out the subplot for each episode
- neural generator used to generated a season of episodes

Evolutionary Free Plot Generator

An existing evolutionary story generator is used to create a plot outline that will serve as overarching plot for a series of episodes.

An example of overarching plot for a season generated by the evolutionary story generator is (numbers are inserted to identify the different facts in the outline, for ease of comparison with the generated tellings):

```
   (1) Hans is set an interdiction by Alexander.
(2) Alexander kidnaps Jeff.  (3) Alexander goes
elsewhere while Hans remains behind.  (4) Hans
violates the interdiction.  (5) Hans rescues Jeff
from Alexander.  (6) Hans fights with Alexander.
(7) Hans is pursued by Alexander.  (8) Hans avoids
pursuit.  (9) Hans achieves victory over Alexander.
```

We are taking this plot as the outline that should drive the story generation in the remaining experiments.

Refining Prompts for Neural Story Generation

The story outline produced by the evolutionary story generator can be passed on to a neural generator to construct a fluent text rendering of the plot. However, the actual design of the prompt to use plays a very important role in the process. The neural generator used for this work is the Gemini Pro 1.0 model Gemini-Team (2023) accessed via its available web interface[1].

Our first attempt considered the following prompt:

```
   <prompt 1>
Rewrite the following facts as a short story
```

which lead to the following outcome

```
   <outcome 1>
Hans, a brave soldier, was ordered by his king
Alexander to stay in a certain area and wait for his
orders (1). However, Hans felt the urge to go out and
explore the area, so he disobeyed his orders (4).
Alexander was furious and decided to take revenge on
Hans.

First, Alexander kidnapped Jeff (2), one of Hans's friends,
and held him hostage. He then ordered Hans to come and
meet him. Hans was hesitant at first, but he knew the
consequences if he disobeyed Alexander. He reluctantly
agreed and went to meet him. Alexander was waiting for
him, furious that Hans disobeyed his orders. He
threatened to kill Jeff if Hans did not comply with
his orders. Hans was terrified and tried to escape,
but Alexander caught him and held him captive.

Soon after, Alexander went elsewhere while Hans
remained behind (3). Alexander was now free to pursue
Hans (7). Hans was now in a dangerous situation. The
king's soldiers were after him, and he had no way of
escaping. He was surrounded and outnumbered. Hans
fought bravely against the soldiers and managed to
survive. Alexander had been gone for a while, and
Hans was now free to go. However, he was still pursued
by the soldiers. They were relentless and would not
let him go. He was running out of options and was about
```

[1] https://gemini.google.com

```
to give up when a miracle happened. Hans saw a nearby
castle and decided to go there. He hoped that the
castle would protect him from the soldiers and that
Alexander was not far behind. The castle turned out to
be a safe haven for Hans (8). He was able to hide there,
waiting for the soldiers to go away.

Finally, Alexander returned to the castle. He was
furious that Hans had disobeyed his orders and had
almost killed him. He demanded that Hans come out and
face him. Hans was now trapped, but he had nowhere to
run. Alexander demanded that he come out and face him.
He was now at the mercy of Alexander and had no choice
but to obey. Alexander threatened to kill Jeff if Hans
did not comply. Hans was now in a dangerous situation.
He had to either obey Alexander or face the
consequences.

Finally, Hans came out of his hiding place and faced
Alexander. He was outnumbered and outgunned, but he
was determined to fight for Jeff. He fought bravely
and managed to defeat Alexander (9). Alexander was now
captured and imprisoned for his crimes. Hans was now
a hero, and his actions had saved Jeff's life (5). The
king was grateful for Hans's bravery and decided to
reward him for his service to the kingdom. The end.
```

A second attempt used the following prompt:

```
<prompt 2>
Use the following events to write a short story
```

which leads to the following outcome

```
<outcome 2>
Hans goes to Alexander's castle to get a message
for his brother.  He is intercepted by Alexander's
guards who take him to Alexander's room.  Alexander
has Hans tied up and threatens to kill him unless
Hans tells him about Hans' brother.  Hans tells
Alexander about Hans' brother and Alexander decides
to kill his brother too.  Alexander kidnaps his
brother (2) and leaves him.  Alexander returns to
his castle and Hans is free.  He goes to his village
and finds out that Alexander has gone elsewhere
(3).  He goes to his village and finds out that
Alexander's men have gone to Hans' village to find
out what happened to Hans.  Hans goes to his village
and rescues his brother (5).  He then goes to the
castle where he finds out that Alexander has left.
Hans then goes to Alexander's castle and fights
him (6).  Alexander is defeated (9) and they both
escape.
```

For the first prompt, all the facts in the input have been included, but the order has been scrambled. Of the final outcome, the given facts are only 35% of all the events mentioned. For the second prompt, only 5 out of the 9 facts in the input are included, the relative order of those included is also scrambled, and the given facts are 47% of all the events mentioned. So the phrasing of the first prompt appeared to induce the addition of a significant number of additional events, whereas the phrasing of the second appeared to allow the option of leaving out many of the events in the input.

Given these considerations, we decided to use the following prompt:

Prompt	Facts included	Relative order	Facts added
1	9/9	1-4-2-3-7-8-9-5	17/9
2	5/9	2-3-5-6-9	10/9
3	7/9	1-4-5-6-9-8-7	0/9

Table 2: Summary of performance of the neural generator in rendering a given set of facts, under three different prompts.

```
<prompt 3>
Use the following events to write a story; make
sure all the given events are mentioned in the
story; keep the relative order or the events; do not
add new events or characters to the story
```

This results in the following outcome (for ease of understanding, sentences estimated to correspond to facts in the input prompt have the number for the corresponding plot fact inserted after them; these numbers are not generated by the text generator):

```
<outcome 3>
Hans was a young and brave knight who had been
set an interdiction by Alexander (1), the emperor
of the Roman Empire.  He was forbidden to leave the
city of Rome and travel to the east.  However, Hans
disobeyed the interdiction (4) and went to the east
to find Jeff, his best friend.  After several months
of searching, Hans found Jeff in a nearby village
and managed to rescue him from Alexander (5).  Hans
then fought with Alexander (6) and defeated him (9),
earning his freedom and glory.  However, he had to
avoid being pursued by Alexander (8), who was still
looking for him (7).  In the end, Hans achieved
victory over Alexander and was able to return to the
city of Rome.
```

The performance of the neural generator as means of rendering a specific input is quite poor. A summary of the relative comparison between the set of input facts and the two outcomes is given in Table 2.

Neural Plot Generator

Our next experiment explores how the neural generator performs as means of generating a series of episodes. Again, the prompts to be used have to be refined to achieve acceptable results.

To allow for a certain comparability with the outcomes of the evolutionary generator, we include in the prompt certain restrictions regarding the nature of the plot to be produced.

The first version considered for the prompt was:

```
<prompt 4>
Write a short story that serves as an overarching plot
for a series of episodes. The story must have three
male characters. One of the characters must kidnap
another, and the remaining character must rescue him.
```

This results in the following outcome:

```
<outcome 4>
1. Introduction: The story begins with a group of
friends on a camping trip in the woods. The setting is
dark and mysterious, with a campfire providing the only
```

light. Suddenly, one of the friends goes missing.

2. The kidnapping: The missing friend is found tied up and gagged in the forest. The other friends are shocked and afraid, but they must quickly come up with a plan to rescue him.

3. The rescue: The group devises a plan to distract the kidnapper while one of them sneaks up from behind. They succeed in freeing their friend, and together, the group celebrates their victory. The story ends with a lesson about the dangers of being alone in the woods at night. As a male character, I can play the role of the kidnapper or the rescuer. The kidnapper's character would be cunning and calculating, with a strong sense of power and control. The rescuer's character would be resourceful and determined, with a willingness to take risks to save their friend.

In this outcome, the model has obeyed the instruction of breaking the story into three episodes, and it has included both the kidnapping and the rescue. But the episodes lack any internal structure and the plot generated for the whole story is extremely basic.

Since prompt 4 appears inadequate to produce the type of results we expected, so we refined it with the goal of obtaining from the model an outline that might satisfy the requirements for a series of episodes described in the Previous Work Section.

An LLM is prompted to generate a story that will serve as overarching plot for a series of episodes.

The prompt used is:

<prompt 5>

Write a short story that serves as a plot for a series of three episodes. The story must have three male characters. One of the characters must kidnap another, and the remaining character must rescue him. Indicate in the story where each episode would start and end. Ensure each episode has its own subplot and that the story has an overarching plot, with the three characters appearing in each of the episodes. Give each character a name. Each episode must be at least 50 words.

The use of prompt 5 results in the following outcome:

<outcome 5>

Episode 1 - The Abduction A man named John is kidnapped by his ex-girlfriend, Jane. He is shocked and confused, and he does not know what to do. He calls his friend, Peter, for help. Peter is an experienced detective, and he quickly figures out the situation. He calls his friend, Paul, who is an expert in criminal investigations. Paul agrees to help Peter, and he flies to John's hometown to find Jane. He starts asking around and looking for clues. He eventually finds Jane and confronts her. Finally, he arrests Jane for kidnapping John. John is relieved that he is safe, and he thanks Peter and Paul for their help. The end of the episode.

Episode 2 - The Rescue John is now in Peter's hands, and Peter is trying to figure out what to do. He has to think of a way to rescue John without getting caught. After some brainstorming, he comes up with a plan. Peter tells John that they will meet at a specific location at a specific time. John is skeptical, but he follows Peter's instructions. Finally, they arrive at the location, and Peter gives John a signal to start the rescue. John is surprised to see that it is Peter who has come to rescue him. Peter explains that he was able to get inside Jane's house and follow her trail. He was able to track her to a nearby town and then to her car, where John was waiting for him. They start driving back to John's hometown and John is relieved that he is safe. The end of the episode.

Episode 3 - The Overarching Plot Peter and John are now friends, and they decide to work together to catch Jane. They agree to meet at a specific location every month to share updates and information. Finally, they decide to start a private detective agency together. The end of the episode.

This outcome has a more interesting first episode, with its own plot structure. However, all the elements presented as requirements are consumed in the plot of that episode. The plots for the other two episodes do not make much sense. We take this to be an indication that the neural generator performs better when sufficient restrictions are provided in the input to give a solid structure in the set. Based on this hypothesis we consider a further experiment in which, rather than hand-craft a more elaborate prompt in which additional details are provided to structure the remaining episodes, we use an adaptation of the existing evolutionary plot outline generator to generate specific outlines for each episode.

Evolutionary Constrained Plot Generator

The evolutionary story generator described at the start of this section is adapted to generate episode subplots that are built around a number of events from a given overarching season plot.

For each episode, the span of plot atoms from the overarching plot for the season that needs to be considered will act as a seed for the construction of the episode. Revised versions of the mutation and cross-over operators were developed to ensure that the span of plot atoms from the season's plot arc is not altered in any way during the construction of the episode. During construction of an episode, at each generation, each individual draft in the population is merged with the context of the season for the episode under construction before it is scored by the fitness function. The metrics on relative order of plot atoms and character assignment are then applied to the complete draft of the season. This ensures that the draft for the episode is at every point evaluated in the context of all the elements from the season that it needs to take into account.

This procedure ensures that the following constraints are met: each episode is constructed with a coherent plot line that runs from beginning to end, each episode includes the corresponding span of the season plot, the events in the season plot and each episode are linked by a set of shared characters, the span of the season plot corresponding to an episode is not altered during the construction of the episode.

The following example (referred in the prompts that follow as *Template-based rendering of full season plot*) of how the evolutionary procedure fleshes out the season plot by generating detailed content for the plot of each episode shows in Bold the events that correspond to the plot of the season.

[Episode 1]

(1) Hans suffers poverty. (2) Hans has aspiration. (3) **Hans is set an interdiction by Alexander.** (4) **Alexander kidnaps Jeff.** (5) Hans dresses up as a member of the opposite sex. (6) Hans is transformed. (7) **Alexander goes elsewhere while Hans remains behind .** (8) Hans fulfills their aspiration. (9) Hans is recognised.

[Episode 2]

(10) **Hans violates the interdiction.** (11) **Hans rescues Jeff from Alexander.** (12) Hans is tested by Joseph. (13) Hans meets Edward. (14) Hans reacts to the test by Joseph. (15) Hans falls in love with Edward. (16) **Hans fights with Alexander.** (17) Hans is validated by Joseph. (18) Hans sees validation recognised. (19) Hans lives happily ever after with Edward.

[Episode 3]

(20) **Hans is pursued by Alexander.** (21) Alexander calls to action Lem. (22) **Hans avoids pursuit.** (23) Alexander meets Lem. (24) **Hans achieves victory over Alexander.** (25) Alexander falls in love with Lem. (26) Lem is rewarded. (27) Alexander lives happily ever after with Lem.

The outline of the season can be passed on to a neural generator to construct a fluent text rendering of the plot, using the following prompt that includes the refinements recognised as necessary in the previous experiment:

<prompt 6>

Use the following events to write a short story: <Template-based rendering of full season plot>. Keep the order of the events and do not add new events or characters to the story

This results in the following outcome:

<outcome 6>

(1) Hans suffers poverty: Hans lives in a small village in Germany. He has no money, no education, and no prospects in life. He is an orphan, and his parents are dead. He is alone in the world, and he is very lonely.

(2) Hans has aspiration: Hans wants to study and become a scientist. He knows that he is talented, and he wants to make a difference in the world. He wants to help others, and he wants to change the world for the better.

(3) Hans is set an interdiction by Alexander: Alexander is a powerful man who rules the country. He is ruthless and corrupt, and he does not care about the poor. He has ordered Hans to stop his studies, and he has threatened him with punishment if he does not obey.

(4) Alexander kidnaps Jeff: Hans is a member of Alexander's army, and he is sent to fight in the army. He is a good soldier, and he does his job well. However, he is captured by Alexander's enemies, and he is taken hostage.

(7) Alexander goes elsewhere while Hans remains behind: Hans is taken to Alexander's enemy, who tortures him and tries to kill him. However, Hans survives, and he is rescued by Alexander's soldiers.

(5) Hans dresses up as a member of the opposite sex: Alexander's army has a strict code of behavior. All soldiers are expected to dress in a particular way, and they are not allowed to talk or socialize with the locals. Hans is captured by Alexander's enemy, who wants him to dress up as a woman. He does this reluctantly, and he tries to escape from the enemy's camp.

(6) Hans is transformed: Alexander's enemy has taken Hans captive, and he has transformed him into a woman. He has given him new clothes, and he has taught him how to behave and talk.

(9) Alexander recognises Hans: Hans is taken to a castle, where Alexander is staying. Hans is recognised by Alexander, who is shocked to see him in a woman's body. Alexander is furious, and he orders Hans to be killed.

(8) Hans fulfills their aspiration: Hans is brought to Alexander's castle, where Alexander has a feast for his army. Hans is dressed in his old clothes, and he is allowed to speak and behave as he wishes. He is recognised by Alexander, who is shocked to see him in a woman's body.

Facts included	Relative order	Facts added
9/27	1-2-3-4-7-5-6-9-8	7/27

Table 3: Summary of performance of the neural generator in rendering the set of facts for prompt 6.

A summary of the relative comparison between the set of input facts and the generated outcome is given in Table 3. The model appears to identify the first third of the events provided in the input as suitable for a plot, and indeed construct an acceptable story setting those out in almost the original relative order. However, it then ignores the remaining 18 events in the input. It also appears not to understand the role played by the labels in the input that separate the different episodes. The instruction to refrain from adding events to the story other than those mentioned in the input has been disregarded.

Our next refinement of the prompt attempts to address some of the issues observed with the preceding version of the prompt: the following prompt intends to ensure that the boundaries between episodes are respected:

<prompt 7>

Write a story about a poor man named Hans; use the following events to drive the story; make sure all the given events are mentioned in the story; keep the relative order of the events; do not add new events or characters to the story; the story must be at least 100 words long: <Template-based rendering of full season plot>

For reasons unknown to us, the model is incapable of returning text outcomes to this type of prompt.

As a result we consider using the neural generator to generate separately the textual rendering for the differ-

Facts included	Relative order	Facts added
5/9	2-3-10-4-5-8-8	1/9

Table 4: Summary of performance of the neural generator in rendering the set of facts for prompt 8.

ent episodes, using the templated-based rendering of each episode. The prompt created along these lines for episode 1 would be:

```
<prompt 8>
Write a story about a poor man named Hans; use
the following events to drive the story; make sure
all the given events are mentioned in the story;
keep the relative order of the events; do not add
new events or characters to the story; the story
must be at least 100 words long:  Hans suffers
poverty.  Hans has aspiration.  Hans is set an
interdiction by Alexander.  Alexander kidnaps Jeff.
Hans dresses up as a member of the opposite sex.
Hans is transformed.  Alexander goes elsewhere
while Hans remains behind .  Hans fulfills their
aspiration.  Hans is recognised.
```

This results in the following outcome (numbers used to identify the events in the season plot have been inserted into the text of the original outcome to help identify the relation between input and outcome):

```
<outcome 8>
One day, Hans was living in a poor village.  He
was an ambitious man, who had a dream of becoming a
successful person (2).  One day, Alexander, a rich
man, came to the village to set an interdiction (3).
He asked all the people to stop doing any work, and
told them that they would not be able to earn money
for three years.  Hans was a very hardworking man,
so he did not listen to Alexander (10).  Alexander
then kidnapped Jeff (4), a young man who was also
living in the village, and took him with him to a
faraway place.  Hans then dressed up as a woman (5),
and pretended to be the wife of Jeff.  He went to
work as a servant in a rich house.  He worked hard,
and finally fulfilled his aspiration of becoming a
successful person (8).  After three years, Alexander
came back to the village.  He saw that the villagers
were still living in poverty.  He then let Hans out
of his interdiction, and told them that they could
work and earn money again.  Hans was finally set
free from his poverty, and he fulfilled his dream of
becoming a successful person (8).
```

A summary of the relative comparison between the set of input facts and the generated outcome is given in Table 4.

Neural Series Plot Generator

A final experiment we consider is to explore ability of the neural model to generate the plot outlines for a series of episodes.

Our initial attempt considered the following prompt:

```
<prompt 9>
Write the story of a series of three episodes. The
story must have a complex plot and three male
characters. One of the characters must kidnap another,
and the remaining character must rescue him. Specify
what happens in each of the episodes.
```

This results in the following outcome:

```
<outcome 9>
Episode 1: The first episode begins with the
kidnapping of a young man by a gang of thugs. The thugs
hold the young man captive in a dark alley, threatening
to harm him if anyone comes near. They force him to
wear a mask and blindfold him.

Episode 2: The young man's friends come to his rescue,
but they are outnumbered and outmatched by the thugs.
They manage to free the young man, but he is badly
beaten and unconscious.

Episode 3: The young man awakens to find himself
chained to a wall inside a warehouse. He is surrounded
by a menacing figure, who reveals himself as the leader
of the thugs. He threatens to kill the young man if he
tries to escape. The young man struggles to break free,
but the leader has a hold of a knife. Just as it seems
that the young man is doomed, a mysterious figure
appears and rescues him. The mysterious figure is
revealed to be a masked hero, who defeats the leader
and sets the young man free. The masked hero reveals
himself to be the young man's best friend, who has
returned from a mission to save him. The friends
celebrate their victory and vow to continue their
mission of fighting injustice together.
```

This outcome again shows that the model has difficulty following the concept of assigning a specific subplot to each episode. The kidnapping and rescue are used to provide an overarching plot for the three episodes, but the rescue appears first in episode 2 and again in episode 3.

In view of these observations, we reformulate the prompt as:

```
<prompt 10>
Write the story of a series of three episodes. The
story must have a complex plot and three male
characters. One of the characters must kidnap another,
and the remaining character must rescue him. Specify
what happens in each of the episodes.
Indicate in the story where each episode would start
and end. Ensure each episode has its own subplot and
that the story has an overarching plot, with the three
characters appearing in each of the episodes.
Give each character a name. Each episode must be
at least 50 words.
```

This results in the following outcome:

```
<outcome 10>
The Janus Coin:  A Three-Part Miniseries
Episode 1:  The Price of Power
Characters:  Elias (ambitious scholar), Kai
(streetwise thief), Dr.  Volkov (enigmatic
scientist)
Plot:  Elias, a brilliant but underfunded
scholar, desperately seeks funding for his research
on ancient artifacts.  He stumbles upon an auction
```

```
for a mysterious coin { the Janus Coin { rumored
to grant immense power.  Dr.  Volkov, a wealthy
and reclusive scientist, wins the coin in a bidding
war against Elias.  Elias, fueled by desperation,
convinces Kai, a charming but morally flexible
thief, to steal the coin.  Kai succeeds, but is
captured by Volkov's guards.  The episode ends
with Elias staring at the stolen coin, a mixture of
triumph and unease clouding his eyes.
  Episode 2:  Shadows and Secrets
  Characters:  Kai, Dr.  Volkov, Elias
  Plot:  Kai wakes up imprisoned in Volkov's hidden
laboratory.  He discovers Volkov conducting bizarre
experiments with the Janus Coin, believing it
to hold the key to immortality.  Kai befriends
Dr.  Anya, Volkov's assistant, who reveals her
growing unease with his increasingly ruthless
methods.  In desperation, Kai contacts Elias using
a hidden communicator within his cell.  Elias,
riddled with guilt and fearing the coin's power,
decides to rescue Kai.  The episode ends with Elias
infiltrating Volkov's compound, determined to free
Kai and stop Volkov's experiment.
  Episode 3:  The Price We Pay
  Characters:  Elias, Kai, Dr.  Volkov (with a
shocking transformation)
  Plot:  Elias finds Kai, but their escape is cut
short by Volkov.  As they fight their way out,
the Janus Coin activates, unleashing a surge of
energy that transforms Volkov into a monstrous being
fueled by the coin's power.  In the ensuing chaos,
Dr.  Anya sacrifices herself to give Elias and Kai
a chance to escape.  Now faced with a monstrous
Volkov, Elias realizes the true price of power.  He
and Kai must work together to defeat Volkov, even
if it means destroying the coin and ending their
individual dreams of success.  The final battle ends
with the laboratory crumbling, leaving the fate of
the coin, Volkov, and the characters hanging in the
balance.
```

The outcome for prompt 10 constitutes a fairly good story, and it satisfies most of the constraints established in the prompt. It is clear that this is a better story than those obtained by any of the other attempts. However, it is more of a serial than a series, in the sense that the episodes do not really have a full subplot, merely a span of the plot for the season.

Discussion

An important insight arising from the experiments described is that the specific neural model being used in this paper requires that some basic assumptions concerning the task of rendering a set of events as narrative text to be made explicit in the prompt. The assumptions that need to be explicitly stated are: make sure all the given events are mentioned in the story, keep the relative order for the events, and do not add new events or characters to the story. Without these specific instructions in the prompt, the model produces outcomes that violate these requirements. Even with these instructions declared explicitly in the prompt, the requirement are sometimes violated.

The model does show the ability to identify a small set of events provided in the input as material relevant to the story it should construct. However, there appears to be a limit to the number of events it can consider in this way. When the input includes more than 10 events, the model builds a story with the first 10 and disregards the rest. If the story produced is longer than the span covered by the first 10 events in the input, the model extends the text built from those events with further events that were not either not present in the input or correspond to reiterations of events in that first set of 10. This issue may be related with the procedure and/or the dataset employed to train the model for the task of story generation. If the examples considered in that procedure included a number of events in that particular range, it may be possible that the model stops responding when the range is exceeded.

One aspect that the model also has difficulty in observing is the concept of a narrative built as a series of episodes, with an overarching plot for the whole and specific subplots for the episodes. In spite of various attempts to refine the prompt to ensure compliance with this format, the outcomes consistently fail to achieve this. This happens even in the case of relying exclusively on the neural model to produce the full plot for the series, which yields a story of surprising quality. This may also be related to having used during the training of the model schemas that correspond to simple stories rather than stories deployed in the form of a series of episodes.

It is interesting to note that when the neural model is prompted to generate a series of episodes, it ends the final episode with an open ending. This is in keeping with the observed characteristics of existing TV series, as described in the Previous Work section. At least that part of the concept of series appears to be well understood by the model.

Conclusions

The model for neural generation of stories that has been tested in this paper is capable of producing very fluent text that expands given events into spans of text of several sentences that make sense, sound convincing and have a certain narrative appeal. However, it faces difficulties when asked to adapt the resulting story to include a certain set of events described in the prompt. The difficulties that have been identified in this paper include: failing to include all the events, not respecting the relative order in which the events are provided, and adding events not mentioned in the prompt. The refinements to the prompt introduced in the hope of solving these difficulties do somewhat reduce their incidence but fail to eliminate them altogether.

The model also struggles with the concept of building a story that has an overarching plot but also includes episodes such that each has its own subplot. Attempts to refine the prompts to work around this limitation have failed to produce successful outcomes.

Overall, it appears that neural models are very capable of producing high quality stories when left unconstrained, but react badly to any attempt to guide the outcomes in particular directions.

The experiments presented in this paper have been carried out with a specific model and they would need to be replicated with a broader range of models.

Author Contributions

Pablo Gervás wrote the code for the evolutionary generation of the plots for a series of episodes. Gonzalo Méndez carried out the experiments for the neural generation of the texts. Both authors contributed equally to the elaboration of results and the writing of the paper.

Acknowledgments

This paper has been partially supported by the CANTOR project (PID2019-108927RB-I00) funded by the Spanish Ministry of Science and Innovation.

References

Alhussain, A. I., and Azmi, A. M. 2021. Automatic story generation: A survey of approaches. *ACM Comput. Surv.* 54(5).

Allrath, G., and Gymnich, M. 2005. Towards a narratology of tv series. In Allrath, G., and Gymnich, M., eds., *Narrative strategies in television series*. Springer.

Elliott, C. 2024. A hybrid model for novel story generation using the affective reasoner and chatgpt. In Arai, K., ed., *Intelligent Systems and Applications*, 748–765. Cham: Springer Nature Switzerland.

Fox, B. 2014. *Contemporary Television Series: Narrative Structures and Audience Perception*. Cambridge Scholar Publishers.

Garin, M. 2017. Infinite wounds: redefining narrative structure and serial dynamics in television series. *L'Atalante. Revista de estudios cinematográficos* 24:27–41.

Gemini-Team. 2023. Gemini: A family of highly capable multimodal models.

Gervás, P. 2019. Generating a search space of acceptable narrative plots. In *10th International Conference on Computational Creativity (ICCC 2019)*.

Gervás, P. 2022. Evolutionary stitching of plot units with character threads. In *WIVACE 2022 XVI International Workshop on Artificial Life and Evolutionary Computation*.

Mikhaylovskiy, N. 2023. Long story generation challenge. In Mille, S., ed., *Proceedings of the 16th International Natural Language Generation Conference: Generation Challenges*, 10–16. Prague, Czechia: Association for Computational Linguistics.

Okker, P. 2003. *Social Stories: The Magazine Novel in Nineteenth-Century America*. University of Virginia Press.

O'Sullivan, S. 2019. Six elements of serial narrative. *Narrative* 27(1):49–64.

Rennett, M. 2011. Narrative philosophy in the series: Fate, determinism and the manipulation of time. *Looking for Lost: critical essays on the enigmatic series* 25–42.

Simon, N., and Muise, C. 2022. Tattletale: storytelling with planning and large language models. In *ICAPS Workshop on Scheduling and Planning Applications*.

Xie, K., and Riedl, M. 2024. Creating suspenseful stories: Iterative planning with large language models.

Yuan, A.; Coenen, A.; Reif, E.; and Ippolito, D. 2022. Wordcraft: Story writing with large language models. In *27th International Conference on Intelligent User Interfaces*, IUI '22, 841–852. New York, NY, USA: Association for Computing Machinery.