

# Narrative Planning Model Acquisition from Text Summaries and Descriptions

Thomas Hayton<sup>1</sup>, Julie Porteous<sup>2</sup>, João F. Ferreira<sup>3</sup>, Alan Lindsay<sup>4</sup>

<sup>1</sup>Teesside University, Middlesbrough, United Kingdom, t.hayton@tees.ac.uk

<sup>2</sup>School of Science, RMIT University, Melbourne, Australia, julie.porteous@rmit.edu.au

<sup>3</sup>INESC-ID & Instituto Superior Técnico, Universidade de Lisboa, Portugal, joao@joaoff.com

<sup>4</sup>University of Huddersfield, United Kingdom, a.lindsay@hud.ac.uk

## Abstract

AI Planning has been shown to be a useful approach for the generation of narrative in interactive entertainment systems and games. However, the creation of the underlying narrative domain models is challenging: the well documented AI planning modelling bottleneck is further compounded by the need for authors, who tend to be non-technical, to create content. We seek to support authors in this task by allowing natural language (NL) plot synopses to be used as a starting point from which planning domain models can be automatically acquired. We present a solution which analyses input NL text summaries, and builds structured representations from which a PDDL model is output (fully automated or author in-the-loop). We introduce a novel sieve-based approach to pronoun resolution that demonstrates consistently high performance across domains. In the paper we focus on authoring of narrative planning models for use in interactive entertainment systems and games. We show that our approach exhibits comprehensive detection of both actions and objects in the system-extracted domain models, in combination with significant improvement in the accuracy of pronoun resolution due to the use of contextual object information. Our results and an expert user assessment show that our approach enables a reduction in authoring effort required to generate *baseline* narrative domain models from which variants can be built.

## Introduction

Domain modelling for automated planning is challenging in general but is further compounded when non-technical authors are needed to create the content to populate the model. This is particularly true for domain models for use in Interactive Entertainment systems and games. Whilst these applications are our focus in this work we note that this is also true of other application domains such as requirements engineering (Deeptimahanti and Babar 2009).

AI planning has been widely used for generating narrative in Interactive Entertainment systems e.g. (Aylett, Dias, and Paiva 2006; Riedl and Young 2010; Porteous, Charles, and Cavazza 2013). To date, the modelling of these domain models has been handled manually: a common strategy being to start by building a *baseline* plot and then building up

interactive models via systematic consideration of alternatives around the baseline (Porteous, Cavazza, and Charles 2010). Indeed many prototype systems have sought inspiration from existing narrative works (e.g. *Who's Afraid of Virginia Woolf?* (Mateas and Stern 2005), *Madame Bovary* (Cavazza et al. 2009), *Aladdin* (Riedl and Young 2010)) and games (e.g. *Hitman* (Pizzi et al. 2010)). Our motivation in this work is to assist authors and reduce this authoring burden, by developing an automated route to baseline narrative planning model development.

The solution presented in the paper is an automated approach that takes as input natural language (NL) sentences summarising the main elements of a story (i.e. from plot synopses) and from this generates planning action representations, a narrative planning domain model, corresponding to the baseline plot. This is a non-trivial task, as NL sentences often have multiple clauses and conjunctions that use many pronouns and multiple references to the same object or characters. This approach is fully implemented in a prototype system, the main tasks of which are: (i) identify objects in the domain; (ii) remove all pronoun references from the text using a novel sieve-based approach to pronoun coreferencing which exploits contextual object information; and (iii) use NLP techniques to identify actions in the input sentences and construct structured representations which are used to build the final output narrative domain model.

The contribution of this work is an approach that can generate baseline narrative planning domain models from input NL plot synopses in a fully automated way. Thus this helps reduce the overall authoring burden.

In the paper we start with background on the work. This is followed with detail of the key aspects of the approach: (1) Object Identification; (2) Input Sentence Pre-Processing; and (3) Narrative Domain Model Acquisition. In the evaluation we consider the performance of our approach on each of these aspects. The results are encouraging and demonstrate comprehensive detection of narrative objects and events (actions) in the system-generated models, in combination with marked improvement in the accuracy of pronoun resolution. We further show that the models are able to re-generate the original baseline story and discuss results of a user study and the ability of the approach to reduce authoring burden.

### ① Input Plot Synopses (Jungle Book)

S1: **Mowgli** |,|a young orphan boy |,|is found in a basket in the deep jungles of India by **Bagheera** |,|a black panther who promptly takes *him* to a **mother wolf** who has just had cubs.

S2: Shortly afterwards |,|a group of monkeys kidnap **Mowgli** |and|take *him* to their leader |,|**King Louie** the Orangutan. **King Louie** offers to help **Mowgli** stay in the jungle if *he* will tell **Louie** how to make fire.

### ② After pre-processing (Pronouns resolved)

S1: [Mowgli,] [a young orphan boy,][is found in a basket in the deep jungles of India by Bagheera,][a black panther who promptly takes Mowgli to a MotherWolf who has had cubs]

S2: [Shortly afterwards,][a group of monkeys kidnap Mowgli][take Mowgli to their leader,][Louie the Orangutan.][Louie offers to help Mowgli stay in the jungle if Mowgli will tell Louie how to make fire]

Figure 1: Example: ① input NL; ② pre-processed output. Highlighted: disambiguated names (**King Louie**→**Louie**); pronoun resolution (*him*→**Mowgli**); sentence breaks (| |)

## Background

The focus of our work is to build a specialised approach to domain model acquisition, which is supported at each point by our observations of the process and available information typical to the construction of narrative domain models. Thus input is a story plot synopsis in the form of NL sentences, and the target output is a planning domain model. Note that we assume a planning problem in PDDL (McDermott et al. 1998) separated into: the domain model, a definition of the problem domain that defines the world and its behaviours, and an explanation of the specific problem to be solved within that world. A domain model is a tuple,  $\mathbb{D} = \langle \mathbb{O}, \mathbb{P} \rangle$ , defining the sets of operators,  $\mathbb{O}$ , and predicates,  $\mathbb{P}$ . An operator,  $\mathcal{O} \in \mathbb{O}$ , is represented by an *operator header*: a unique symbol (operator name) and a list of typed variables (parameters). The *operator body* consists of three sets of predicates: the preconditions, and the add and delete effects. An action,  $\mathcal{A}$ , is a planning operator,  $\mathcal{O}$ , that has been instantiated with problem constants (parameters, preconditions and effects) and an *action header* is a name and a list of constants (instantiated parameters).

We assume that the input to domain model acquisition is sourced from publicly available online resources such as Wikipedia. From analysis of these resources we also assume that input plot synopses are written from a third person perspective. As illustration, some sentences from a summary of the film the Jungle Book (from Wikipedia) are shown in Figure 1. Throughout the paper we use The Jungle Book and other synopses for illustration (for details see: Evaluation).

### Step 1: Object Identification

The first aspect of the approach is to identify all objects that appear in the input NL plot synopses. For this we use Stanford CoreNLP (Manning et al. 2014), and the syntactic parsing annotations it produces. In particular: the part-of-speech

(POS) tags for each word (VB verbs, NN nouns and JJ adjectives); and the dependency parse graph relations. For each sentence, all words that have a relation to their parent of either: subject, object, noun modifier, dependent, conjunction, clausal complement, appositional modifier, adverb modifier or adverbial clause modifier are considered. It is possible that a word with a compound relation can be an object if its parent hasn’t already been identified as an object. If the word is either a noun or adjective it is identified as an object. Finally the relations around the word are analysed to see if more detail can be included in the object name. This is done by checking the object word’s children for modifiers and compounds. An example of this is ‘*jungles*’ in S1, Figure 1; with the object being identified as ‘*deep jungles of India*’. The output of this phase is a set of strings which are clustered on the basis of shared substrings from which the largest shared substring is extracted as a unique object name. As illustration, some examples of the object clusters along with the extracted name and type tag are:

Identified Object Clusters	Name	Type Tag
“man-eating Bengal tiger”, “Bengal tiger”, ...	Tiger	MCHAR
“King Louie”, “Louie”	Louie	MCHAR
“laid-back fun-loving bear Baloo”, “Baloo”	Baloo	MCHAR

If an object hasn’t been detected automatically from the text, a user can add objects and aid in disambiguation to ensure a complete list is available going forward. Next, each object is tagged with one of the following types: MCHAR (male character), FCHAR (female character), OTHER (object/location), OTHERP (plural object/location), or GROUP (group/organisation). These types are inferred using online resources, such as (Kantrowitz 2016; iluEnglish 2017) but the system also allows for manual user tagging. Object typing is used in pronoun resolution (next section).

### Step 2: Input Sentence Pre-Processing

**Sentence Segmentation:** Input sentences are segmented into single units, around sentence breaks for later use during coreferencing and domain model extraction. Sentence breaks can be in the form of either punctuation or a coordinating conjunction. Commas, semicolons and colons are the only punctuation marks taken as breaks, provided they are not being used to separate a list of objects. Coordinating conjunctions such as “and” and “but” are also sentence breaks, provided they are **not** being used as follows: (i) to join two words together (e.g. “hide and seek”); or (ii) directly following punctuation that itself denotes a break (e.g. “He thanks them, but”). For illustration, some examples are shown in Figure 1.

**Coreferencing Pronouns:** For later extraction of domain actions we further pre-process the input NL sentences by replacing pronouns with the name of the object they refer to. Whilst online resources, such as CoreNLP (Manning et al. 2014) and spaCy (SpaCy 2.1+ 2019), can be used without modification for this coreference resolution task, we observe that their performance can improve in our context, because there is contextual information which can be used: namely the objects in the input that have already been identified.

Hence in our work we have developed a novel approach which uses this narrative contextual information and in our evaluation we show that this approach is able to outperform both CoreNLP and spaCy.

**Pronoun Types:** As our target input NL is third person synopses, the types of pronouns are restricted to: Subject Pronouns (He, She, It, They); Objective Pronouns (Him, Her, It, Them); Possessive Adjectives (His, Her, Its, Their); Possessive Pronouns (His, Hers, Theirs); and Reflexive Pronouns (Himself, Herself, Itself, Themselves).

For male character references the possessive form is “his” and the objective form “him”, whereas for females, both the possessive and objective forms are “her”. To resolve this we inspect the CoreNLP POS tags for the next word(s) in the sentence: if this is a noun, or set of adjectives describing a noun, then “her” is possessive, otherwise it is objective.

The object types previously inferred are used to guide coreferencing and pronouns are associated with objects of a matching type. The correspondence is: MCHAR (He, His, Him, Himself); FCHAR (She, Her, Hers, Herself); OTHER (It, Its, Itself); OTHERP (They, Their, Them, Themselves); and GROUP (They, Their, Them, Themselves). For example, “she” matches FCHAR, “he” MCHAR and so on.

**Coreferencing Algorithm:** As input sentences are third-person synopses, we assume any object(s) matching a pronoun have been mentioned in the text *before* the pronoun. Thus, by backwards search through the input sentences all objects the pronoun references can be found. The pronouns are resolved in the same order they appear in the text. This is important as all object references prior to a pronoun need to be known for well-reasoned decisions. Object references can be either a named reference or a pronoun reference.

A multi-sieve approach has been developed for this decision making process (see Algorithm 1). It uses two sets of rules: *Sieve1* and *Sieve2*. For each pronoun, a list of potential objects is populated with all the objects that are referenced in the same sentence before the pronoun, providing they match the type of the pronoun. The next step is to apply the 6 rules of *Sieve1*, only one of which can be true, if any. As a result, either an object reference has been found and the search terminated, an object has been flagged as ineligible for selection, or nothing has changed. At this stage, if an object has not been returned, the 3 rules of *Sieve2* are applied. If no match is returned after this, the list of potential object references is expanded by looking to the previous sentence, adding all of the objects it references, providing they match the pronoun type. The rules of *Sieve2* are applied once again, and this process of expanding the potential object list by looking back to the previous sentence and re-applying *Sieve2* continues until a match is found.

**Coreferencing Rules:** Below we describe the rules that make up *Sieve1* and *Sieve2* and illustrate with example sentences (pronoun of interest and matching identifier **high-lighted**). Note: rules are based solely on sentence structure, the words, and their types.

#### **SIEVE 1**

##### **RULE 1: OBJECTIVE INFINITIVE VERB**

The infinitive form is the verb in its basic form (also accepts

---

#### **Algorithm 1: Pronoun Coreference Algorithm**

---

```

Function Main (Input) :
  for s in Sentences (Input) do
    for p in Pronouns (s) do
      // Find all objects that match
      // the pronoun's type
      objects = FindMatchingObjects (s,p)
      // Start the sieve mechanism
      (Sieve 1)
      Sieve1 (Input,objects,p)
    end
  end

Function Sieve1 (Input,objects,p) :
  ApplyRule OBJECTIVE-INFINITIVE-VERB:
  // Rule 1
  if match is found then return match
  ApplyRule OBJECTIVE-AFTER-BREAK:
  // Rule 2
  if match is found then return match
  ApplyRule REFLEXIVE: // Rule 3
  if match is found then return match
  ApplyRule OBJECTIVE: // Rule 4
  remove unsuitable objects from the list objects
  ApplyRule AND-POSSESSIVE: // Rule 5
  if match is found then return match
  ApplyRule INVOLVED-IN-ACTION: // Rule 6
  remove unsuitable objects from the list objects
  // At this point, no match was
  // found. So, start Sieve2.
  Sieve2 (Input,objects,p)

Function Sieve2 (Input,objects,p) :
  ApplyRule SINGLE-MATCH: // Rule 1
  if match is found then return match
  ApplyRule MULTIPLE-MATCH: // Rule 2
  if match is found then return match
  ApplyRule PLURAL-MULTIPLE-MATCH:
  // Rule 3
  if match is found then return match
  // At this point, no match was
  // found. Keep executing Sieve2 on
  // previous sentences until a match
  // is found
  s = PreviousSentence (Input,p)
  if s is defined then
    objects.add(FindMatchingObjects (s,p))
    Sieve2 (Input,objects,p)
  end

```

---

split infinitives with adverbs inserted between ‘to’ and root).  
**APPLY WHEN:** 1. Pronoun is objective; 2. Pronoun is directly preceded by a verb (infinitive); 3. Last referenced object matches pronoun type; 4. At least two different objects have been referenced before pronoun in the sentence.

**ACTION:** The last referenced object is returned as the match.

**EXAMPLE:** LisaCuddy, the Dean of Medicine, comes looking for House **to berate him**. (**him** = House)

##### **RULE 2: OBJECTIVE AFTER BREAK**

**APPLY WHEN:** 1. Pronoun is objective; 2. Pronoun occurs after sentence break and no object reference exists between the pronoun and the sentence break; 3. The last referenced

object matches the pronoun's type; 4. At least two different objects have been referenced before the pronoun in the sentence.

**ACTION:** The last referenced object is returned as the match.

**EXAMPLE:** Bagheera *speaks to* Baloo *and convinces* him the jungle isn't safe for Mowgli. (*him* = Baloo)

### **RULE 3: REFLEXIVE**

**APPLY WHEN:** 1. The pronoun is reflexive; 2. Only one reference matching the pronoun's type exists in the sentence.

**ACTION:** The only matching reference returned as match.

**EXAMPLE:** Shaggy *trips over* *himself*. (*himself*=Shaggy)

### **RULE 4: OBJECTIVE**

**APPLY WHEN:** 1. The pronoun is objective.

**ACTION:** Last referenced object is ineligible for selection.

**EXAMPLE:** House *thinks the patient has a brain tumor, but* Wilson *asks* him *to take the case*. (*him* != Wilson)

### **RULE 5: AND POSSESSIVE**

**APPLY WHEN:** 1. The pronoun is possessive; 2. The pronoun is preceded by the word 'and'; 3. The last referenced object matches the pronoun type.

**ACTION:** The last referenced object is returned as the match.

**EXAMPLE:** Mowgli *joins the elephant patrol led by* Hathi *and* his *wife Winifred*. (*his* = Hathi)

### **RULE 6: INVOLVED IN AN ACTION**

**APPLY WHEN:** 1. The words in between the pronoun and the next named object (going forwards in the sentence), consists of at least one verb, no breaks, conjunctions or nouns.

**ACTION:** The next named object becomes ineligible for selection.

**EXAMPLE:** Louie *offers to help* Mowgli *stay in the jungle if* he *will tell* Louie *how to make fire*. (*he* != Louie)

## **SIEVE 2**

### **RULE 1: SINGLE MATCH**

**APPLY WHEN:** 1. Candidate objects list contains one match.

**ACTION:** The object is returned as the match.

**EXAMPLE:** Mowgli *is playing with* his ... (*his* = Mowgli)

### **RULE 2: MULTIPLE MATCH**

**APPLY WHEN:** 1. The candidate objects list contains more than one match.

**ACTION:** Going backwards in the text, find the last sentence break that occurred. Select the first candidate object to be referenced after this break. If no reference is found, the next sentence break back is used, and the first object reference to occur after this break is selected. Repeat backwards, moving through sentence breaks until match is returned.

**EXAMPLE:** Baloo *and Bagheera* *head home, content that* Mowgli *is happy with* his *own kind*. (*his* = Mowgli)

### **RULE 3: PLURAL MULTIPLE MATCH**

It is possible for a plural pronoun to be referencing a group, an object plural, or multiple characters or objects.

**APPLY WHEN:** 1. The pronoun is plural. 2. The candidate objects list contains more than one match.

**ACTION:** The selection process is the same as in the 'Multiple Match' rule, with an addition. If the match returned is a singular character or object, and other different characters or objects also exist as candidates; they are all returned as matches. Associating the pronoun with multiple references.

Seg1	Mowgli retrieves WaterPot for YoungGirl
CoreNLP	->retrieves/VBZ (root) ->Mowgli/NNP (nsubj) ->WaterPot/PRP (dobj) ->YoungGirl/PRP\$ (nmod:for) ->for/IN (case)
Extracted	ACTION : retrieves Objects : Mowgli WaterPot YoungGirl
Seg2	Bagheera volunteers to escort Mowgli back.
CoreNLP	->volunteers/NNS (root) ->Bagheera/NNP (compound) ->escort/NN (nmod:to) ->to/TO (case) ->Mowgli/PRP (dep) ->back/RB (advmod) ->././ (punct)
Extracted	ACTION : volunteers-to-escort Objects : Bagheera Mowgli

Figure 2: Identifying Actions from input NL segments. Shows: NL input, generated CoreNLP dependency graph with annotations and the identified Actions (Seg1, Seg2).

**EXAMPLE:** He *informs the* gang, *but when* they *return, the painting is back on the wall*. (*they* = gang)

Based on our assumption that matching object(s) for all pronouns will have been mentioned by name in the text *before* the pronoun—a valid assumption for third-person authored synopses—the output of this phase is segmented NL sentences with no pronouns. For some examples see Figure 1.

## **Step 3: Domain Model Acquisition**

In this phase, words in the pre-processed NL segments that represent narrative events and that can become actions in the output domain model are identified. Any objects that could be associated are also linked to the action at this stage. Where available, extra information is then added to action names to aid readability.

### **Identifying Narrative Actions**

Stanford CoreNLP and the syntactic parsing annotations it produces are used to identify actions (for illustration some examples are shown in Figure 2). For each input NL segment, a CoreNLP dependency graph is generated and each node is analysed. If, during analysis, one of the following is found in the dependency graph an **action** is extracted:

- **Main Verbs:** principle/lexical verbs such as `retrieves` in Seg1, Figure 2 are extracted. Auxiliary verbs, such as 'has' and 'be', are ignored as they are only ever used in sentences with main verbs or adjectives and thus are deemed not useful for forming narrative actions.
- **Nouns** if the following apply: (i) they aren't tagged as objects in the dependency graph (i.e. contain `obj`) or subject (i.e. containing `subj`); (ii) they have at least one character as a child in the graph. For example Seg2 in Figure 2.

## Identifying Objects associated with Actions

The next stage is to identify the objects which are associated with each action and which will form the parameters of the action in the output narrative model. All objects that are referenced in the same segment as an action are added as parameters. As an example consider the action `retrieves` in Seg1 Figure 2, for which the following 3 parameters are identified from the associated objects: `Mowgli`, `WaterPot`, `YoungGirl`. It is possible for a related object to not be mentioned in the same segment. To identify these, the dependency graph of the full sentence is checked to see if the action has a subject or object relating to a different segment. When an objective pronoun is present in the segment, the action has to include another object, different to that referenced by the pronoun. Objects from the previous segment(s) are added until this condition is met. Adding the objects referenced within previous segments is also done when no associated objects have been found.

## Translation to PDDL

Next the output PDDL domain model is constructed, using the extracted information, in a form that is sufficient for re-constructing the original story.

**Actions** are named using those extracted from the CoreNLP dependency graphs. As we are targeting narrative planning models, the following common object types are assumed: character, group, object and location.

**Parameters** correspond to the associated objects identified earlier with the relevant object types (as identified and disambiguated during the pre-processing). An additional parameter is added to all actions, an object of the type `causality` which appears as a predicate argument in action pre- and post-conditions to provide a baseline of causality as described below.

**Pre- and Post-conditions:** in a similar approach to (Yordanova 2016), default predicates are added to the pre- and post-conditions of actions. Named `can-Action`, they introduce a baseline level of causality, sufficient to ensure generation of a narrative plan corresponding to the original input synopsis. There are two types of such predicates: i) *enabling character pre-conditions*, one for each of the actions associated objects; and ii) *enabling causality conditions*, established as effects of actions and required for the `NextAction` in the input synopses. These predicates are named `can-NextAction` with argument of type `?x-causality`. For example, in Figure 3, the action `arrive` has the effect (`can-rescue ?c`) which is required as a pre-condition of the next action `rescue`.

## Author Domain Model Refinement

With author input at this stage the clarity and flexibility of the system generated baseline domain model can be improved. In particular: action names that lack detail or are difficult to understand can be renamed; actions deemed redundant can be deleted completely or merged with others; and associated objects can be similarly amended.

In addition, to extend the baseline domain model to allow for the generation of story variants, action pre- and



	<pre>(:action <b>arrive</b> :parameters (?c1 ?c2 ?c3 - char ?s - causality) :precondition (and   (can-arrive ?c1) (can-arrive ?c2)   (can-arrive ?c3) (can-arrive ?s)) :effect (and (<b>can-rescue</b> ?s)))</pre>
	<pre>(:action <b>rescue</b> :parameters (?c1 ?c2 ?c3 - char ?s - causality) :precondition (and   (can-rescue ?c1) (can-rescue ?c2)   (can-rescue ?c3) (<b>can-rescue</b> ?s)) :effect (and (can-ensuing ?s)))</pre>

Figure 3: Example system generated actions: with associated objects Baloo, Bagheera and Mowgli (hence 3 parameters of type `char`). The parameter of type `causality` is used in the predicate `can-rescue` which enables generation of a baseline plan that respects event ordering in the input synopsis.

post-conditions can be amended. The `can-Action` and `can-NextAction` predicates alongside any predicates an author wants to add would enable this.

## Evaluation

The aim of the evaluation was to assess the performance of our approach on: (i) pronoun coreference resolution with multiple pronoun references across sentences; (ii) identification of narrative actions and associated objects from multi-clause sentences; and (iii) generating useful baseline narrative planning models, as assessed by expert users.

For the evaluation we used the following synopses: Scooby Doo, Friends, House, Jungle Book, Toy Story (TS), Titanic, Merchant of Venice (MoV), Christmas Carol (CC), Lord of the Flies (LoF) and The Odyssey (OD) [Synopses]. These were chosen because they are publicly available online resources, not crafted to fit our approach, and provide suitably challenging input, from a representative set of genres with different subject matter, varying levels of detail and a variety of writing styles, vocabulary and language.

The results of experiments are listed in Figure 4. For each synopsis, part ① of the figure lists the total number of input **S**(entences), the total number of **P**(ronouns), narrative **A**(ctions) and **O**(bjects). In the next subsections these counts are used as a gold standard for comparison.

### (i) Pronoun Coreference Resolution

The results of experiments with our coreferencing algorithm are shown in Figure 4 ②. For our sieve-based Algorithm 1 against CoreNLP and SpaCy the table shows: the number of pronouns correctly resolved (green), the number of pronouns that were not resolved correctly (red) and the overall %correct (black). The results show that our algorithm outperforms CoreNLP and spaCy consistently across the domains: this is what we expect with the use of narrative contextual information and the fact that all named entities have been typed. Overall the results are very encouraging; however the results for the House domain are particularly interesting. In the House domain, this causes particular problems for CoreNLP, with 16.4%. The reason for CoreNLP’s poor



	①				②			③	④
	S	P	A	O	Alg. 1	c-NLP	spaCy	Actions	Objects
<b>Scooby</b>	52	79	179	116	64 15 81%	30 49 38%	26 53 32%	112 19 96.6%	178 10 99.4%
<b>Friends</b>	5	5	32	18	4 1 80%	1 4 20%	3 2 60%	16 0 88.9%	32 1 100%
<b>House</b>	28	55	100	61	51 4 92.7%	9 46 16.4%	18 37 32.7%	59 3 96.7%	98 6 98%
<b>Jungle</b>	29	35	124	75	30 5 85.7%	13 22 37.1%	15 20 42.9%	70 2 93.3%	124 10 100%
<b>TS</b>	24	40	166	90	25 15 62.5%	16 24 40.0%	11 29 27.5%	88 10 97.8%	166 6 100%
<b>Titanic</b>	38	57	192	95	52 5 91.2%	18 39 31.6%	15 42 26.3%	89 8 93.7%	190 5 99%
<b>MoV</b>	28	32	111	63	28 4 87.5%	12 20 37.5%	7 25 21.9%	62 2 98.4%	110 3 99.1%
<b>CC</b>	34	52	169	76	42 10 80.8%	28 24 53.8%	40 12 76.9%	72 8 94.7%	168 8 99.4%
<b>LoF</b>	57	51	261	152	45 6 88.2%	32 19 62.7%	37 14 72.5%	146 20 96.1%	258 8 98.9%
<b>Odyssey</b>	39	67	212	94	55 12 82.1%	30 37 44.8%	25 42 37.3%	90 7 95.7%	210 7 99.1%

Figure 4: Results: ① shows the number of **S**(entences), **P**(ronouns), narrative **A**(ctions) and **O**(bjects) for each input synopsis for use as gold standard comparison; ② results of Pronoun Coreference Resolution for Algorithm 1 against CoreNLP and spaCy (% correctly resolved pronouns, #pronouns resolved (green), #not resolved (red)). Our approach outperforms CoreNLP and spaCy across all domains; ③ lists the results for Narrative Action Identification (% correct, #correct (green), #additional errors (red)); ④ lists the results for Object Identification (% correct, #correct (green), #incorrect (red)). See text for further detail.

results on this domain is because House is recognised as an organisation, instead of a male doctor. As House is the main character in the synopsis this had a very noticeable effect.

## (ii) Identification of Narrative Actions and Objects

Figure 4 shows system performance on identification of narrative actions ③ and associated objects ④ from the input NL text synopses. The table lists the number of input sentences (**S**), and then the number of correctly identified Narrative Actions and Objects, against the total number of actions and objects in the input sentences, and the number of errors (where errors are actions and objects which not judged as such in the text). In the table, the errors are shown in red.

These results show a consistently high detection rate for both actions and objects across these domains. The few occasions where objects aren't detected often results from words being assigned an incorrect POS tag by CoreNLP.

## (iii) Evaluation of Baseline Narrative Domain

Our system generated baseline models were sufficient to ensure generation of narrative plans corresponding to an original input synopsis when used with a suitable narrative planning problem. As illustration, part of a Jungle Book synopsis

## Plot Synopsis

A group of monkeys kidnap Mowgli and take him to their leader, King Louie the orangutan. King Louie offers to help Mowgli stay in the jungle if he will tell Louie how to make fire like other humans. However, since he was not raised by humans, Mowgli does not know how to make fire. Bagheera and Baloo arrive to rescue Mowgli.

### (i) System Generated

...  
31:(kidnap-group Monkeys Mowgli)  
32:(stay-in-the-jungle-2 Louie Mowgli)  
33:(tell Louie Mowgli)  
34:(make-fire Louie Mowgli)  
35:(not-was-raised-humans Mowgli)  
36:(not-does-know Mowgli)  
37:(make-fire-2 Mowgli)  
38:(arrive Bagheera Baloo Mowgli)  
39:(rescue Bagheera Baloo Mowgli) ...

### (ii) Hand Crafted

...  
11: (kidnap Mowgli Louie Baloo)  
12: (rescue Mowgli Baloo Bagheera Louie) ...



Figure 5: Example Jungle Book plans. For the plot synopsis fragment (i), the figure shows the corresponding part of the plan generated using actions extracted by our system (ii); and (iii) hand-crafted. Also listed is the action position in the output plan: we observe that the automated method generates more actions at a finer level of granularity, compared to the more general hand-crafted model. This is consistent with the other domains we evaluated (see text for details).

and the corresponding part of a plan generated using the system generated domain model is shown in Figure 5.

We wanted also to qualitatively assess these system generated domain models, and to do so we conducted an evaluation with four experts in the area of PDDL domain modelling: all with publications at international AI conferences; and two also expert in interactive narrative systems development. Given the time required on the part of the experts we restricted the evaluation to: House and Jungle Book. For the evaluation, three domain models were created for each synopsis: (i) **HC** hand-crafted by a PDDL expert; (ii) **A** automatically generated by the system (no user interaction); and (iii) **SA** semi-automatically generated (user interacting to delete and merge actions and change names: the aim being to make minimal changes to improve clarity and flexibility of the model). The numbers of actions in the different models for each domain along with the number of user interactions to create **SA** are shown in Figure 5. We observe that the number of actions in the plans generated by **A** are higher than in the **HC**: an average of more than 2 actions per input sentence. An advantage of the detail in **A** is the possibility for finer grained action interactions that might be abstracted away with **HC**, but this must be traded off against introduction of detail that fails to enhance narratives.

For the evaluation the experts read the original plot synopsis, studied plan traces generated by the different domain models and rated them according to the goodness of fit, of action names and objects to the input synopsis. Users were

Domain	Sentences	#Actions			Semi-Automated		
		HC	A	SA	Del	Merge	Change
House	28	9	69	45	8	14	31
Jungle	29	18	83	58	10	14	28

Figure 6: Domain Model Comparison: #input Sentences; #Actions in output plans to recreate synopsis (HC, A, SA); #user interactions for SA (semi-automated). Details: see text

unaware of how each model had been generated. The order in which the different models were shown to users was randomised and domain models were labelled numerically. Rankings were recorded on an online questionnaire using a 5 point likert scale (1=very poor fit and 5=very good). The results are shown in Figure 7. These rankings support our expectation that the semi-automated models would provide the best fit. The hand-crafted models were ranked rather poorly overall and this is to be expected as the task is laborious and time consuming so there is a tendency to create fewer actions. Similarly the automated domain models were also ranked rather poorly and our expectation is that this was because the system tends to generate more actions. We also asked the participants for comments about the domain models (free text responses). The following gives a flavour of responses for the semi-automated model: “... large amount of detail to the actions ... easy to follow”; “ ... for the purpose of setting an interactive narrative it seems to me that a more granular breakdown of the story allows for a better range of stories” ; “a plan of interesting detail, less abstract than the second plan ... more coherent than the first” (first refers to A and second to HC).

## Related Work

There are a number of approaches that aim to learn planner action models from natural language (NL) input. These vary in the type and source of the inputs that they take, e.g., textual action descriptions (Lindsay et al. 2017; Yordanova 2016) and wikis and webpages (Branavan et al. 2012; Sil and Yates 2011). The approaches also vary in how the causal relations are identified: using NLP techniques applied directly to the text (Sil and Yates 2011); exploiting a feedback loop to support surface linguistic cues (Branavan et al. 2012); by using time series analysis (Yordanova 2016) or targeting an existing model acquisition system (Lindsay et al. 2017). The importance of specialising the process of domain model acquisition to a specific user group has also been observed and examples of areas that have been investigated are puzzle games (Ersen and Sariel 2015) and narrative generation for Interactive Storytelling (Li et al. 2013). Janghorbani et al (2019) introduced a domain model assistant for authoring virtual agents and which automated aspects, including acquiring affordances for pre- and post-conditions from complex, compound sentences. Our approach differs from these various works as the focus is acquiring baseline narrative models from input synopses from external sources. A complementary line of research has investigated mining weblogs and story corpora to obtain narrative content for open story generation. (Swanson and Gordon 2012) use textual case-based reasoning to select narrative content in response

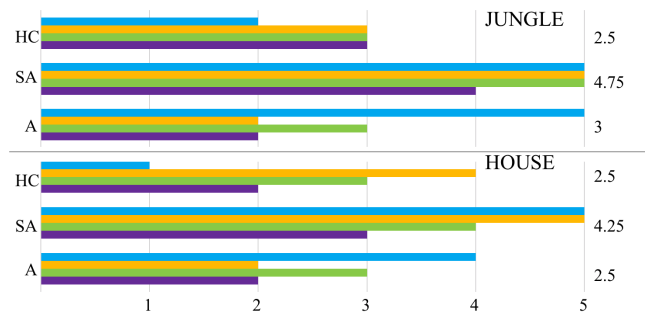


Figure 7: Results of Expert Evaluation. Models created: Automated (A); Semi-Automated (SA); and Hand-crafted (HC). Experts were asked to assess the goodness of fit of the model to the input synopsis (1=v. poor, 5=v. good) with averages shown down RHS. Overall SA rated highest

to user text-based interaction. Recent work (Martin et al. 2018) has explored exploiting the improvements in neural networks as a framework for story generation. These approaches provide a wide scope for content generation, but they rely on a large corpus. In contrast, our approach aims to support the telling of a single specific story (not necessarily represented in any existing corpus) and has an explicit story-world representation, which can provide various guarantees (e.g., on the pedagogic content of the generated narrative).

Finally, the sieve method that we present in this paper for pronoun coreferencing is inspired by the sieve architecture developed by Raghunathan et al. (2010), Lee et al. (2011), and Lee et al. (2013). The sieve architecture follows from the classic line of work focused on rule-based approaches (Mitkov et al. 2007; Haghighi and Klein 2009).

## Conclusion

We presented a specialised approach to domain model acquisition for domains where non-technical experts are required for creation of content to populate the model. In the paper we overviewed our approach and demonstrated its operation using input synopses drawn from online resources. We showed comprehensive detection of objects and actions for a range of synopses. One aspect was the development of a novel algorithm for coreference resolution for narrative applications which consistently outperforms more general tools, such as CoreNLP and spaCy, in this context.

In future work we will extend this method to incorporate multiple synopses into an existing model (e.g. multiple episodes of Scooby Doo), to build models that enable generation of novel narrative variants through recombination.

## Acknowledgements

The work reported in this article was supported by national funds through Fundação para a Ciência e a Tecnologia (FCT) with reference UID/CEC/50021/2019; and by funds from DSI Collaborative Grant CR-0016.

## References

- Aylett, R.; Dias, J.; and Paiva, A. 2006. An Affectively Driven Planner for Synthetic Characters. In *Proc. of 16th Int. Conf. on Automated Planning and Scheduling (ICAPS)*.
- Branavan, S. R. K.; Kushman, N.; Lei, T.; and Barzilay, R. 2012. Learning High-level Planning from Text. In *Proc. of the 50th Annual Meeting of the Association for Computational Linguistics (ACL)*.
- Cavazza, M.; Pizzi, D.; Charles, F.; Vogt, T.; and André, E. 2009. Emotional input for character-based interactive storytelling. In *Proc. of the 8th Int. Joint Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*.
- Deeptimahanti, D. K., and Babar, M. A. 2009. An automated tool for generating uml models from natural language requirements. In *Proc. of 24th IEEE/ACM International Conference on Automated Software Engineering (ASE)*.
- Ersen, M., and Sariel, S. 2015. Learning behaviors of and interactions among objects through spatio-temporal reasoning. *IEEE Transactions on Computational Intelligence and AI in Games* 7(1):75–87.
- Haghighi, A., and Klein, D. 2009. Simple coreference resolution with rich syntactic and semantic features. In *Proc. of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- iluEnglish. 2017. Gender in English: Masculine and feminine words. <https://goo.gl/UhUVfi>.
- Janghorbani, S.; Modi, A.; Buhmann, J.; and Kapadia, M. 2019. Domain authoring assistant for intelligent virtual agent. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, 104–112. International Foundation for Autonomous Agents and Multiagent Systems.
- Kantrowitz, M. 2016. Name corpus: List of male, female, and pet names. <https://goo.gl/bTcwze>.
- Lee, H.; Peirsman, Y.; Chang, A.; Chambers, N.; Surdeanu, M.; and Jurafsky, D. 2011. Stanford’s multi-pass sieve coreference resolution system at the CoNLL-2011 shared task. In *Proc. of the 15th conference on computational natural language learning: Shared task*, 28–34. ACL.
- Lee, H.; Chang, A.; Peirsman, Y.; Chambers, N.; Surdeanu, M.; and Jurafsky, D. 2013. Deterministic coreference resolution based on entity-centric, precision-ranked rules. *Computational Linguistics* 39(4):885–916.
- Li, B.; Lee-Urban, S.; Johnston, G.; and Riedl, M. 2013. Story Generation with Crowdsourced Plot Graphs. In *Proc. of the 27th AAAI Conf. on Artificial Intelligence*.
- Lindsay, A.; Read, J.; Ferreira, J. F.; Hayton, T.; Porteous, J.; and Gregory, P. 2017. Framer: Planning Models from Natural Language Action Descriptions. In *Proc. of the 27th International Conference on Automated Planning and Scheduling (ICAPS)*.
- Manning, C. D.; Surdeanu, M.; Bauer, J.; Finkel, J. R.; Bethard, S.; and McClosky, D. 2014. The Stanford CoreNLP natural language processing toolkit. In *The Annual Meeting of the Association for Computational Linguistics (System Demonstrations)*, 55–60.
- Martin, L. J.; Ammanabrolu, P. J.; Wang, X.; Hancock, W.; Singh, S.; Harrison, B.; and Riedl, M. O. 2018. Event Representations for Automated Story Generation with Deep Neural Nets. In *Proc. of the 32nd AAAI Conf. on Artificial Intelligence*.
- Mateas, M., and Stern, A. 2005. Structuring Content in the Façade Interactive Drama Architecture. In *Proc. of the 1st Conf. on AI and Interactive Digital Entertainment (AIIDE)*.
- McDermott, D.; Ghallab, M.; Howe, A.; Knoblock, C.; Ram, A.; Veloso, M.; Weld, D.; and Wilkins, D. 1998. PDDL-the planning domain definition language. Technical report.
- Mitkov, R.; Evans, R.; Orăsan, C.; Ha, L.; and Pekar, V. 2007. Anaphora resolution: To what extent does it help NLP applications? *Anaphora: Analysis, Algorithms and Applications* 179–190.
- Pizzi, D.; Lugin, J.-L.; Whittaker, A.; and Cavazza, M. 2010. Automatic Generation of Game Level Solutions as Storyboards. *IEEE Transactions on Computational Intelligence and AI in Games* 2(3):149–161.
- Porteous, J.; Cavazza, M.; and Charles, F. 2010. Applying Planning to Interactive Storytelling: Narrative Control using State Constraints. *ACM Transactions on Intelligent Systems and Technology (ACM TIST)* 1(2):1–21.
- Porteous, J.; Charles, F.; and Cavazza, M. 2013. Network-ING: using Character Relationships for Interactive Narrative Generation. In *Proc. of 12th Int. Conf. on Autonomous agents and multi-agent systems (AAMAS)*. IFAAMAS.
- Raghunathan, K.; Lee, H.; Rangarajan, S.; Chambers, N.; Surdeanu, M.; Jurafsky, D.; and Manning, C. 2010. A multi-pass sieve for coreference resolution. In *Proc. of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Riedl, M. O., and Young, R. M. 2010. Narrative Planning: Balancing Plot and Character. *Journal of AI Research* 39:217–267.
- Sil, A., and Yates, A. 2011. Extracting strips representations of actions and events. In *Recent Advances in Natural Language Processing (RANLP)*.
- SpaCy 2.1+. 2019. NeuralCoref 4.0: Coreference Resolution in spaCy with Neural Networks. <https://github.com/huggingface/neuralcoref>.
- Swanson, R., and Gordon, A. S. 2012. Say Anything: Using Textual Case-Based Reasoning to Enable Open-Domain Interactive Storytelling. *ACM Trans. Interact. Intell. Syst.* 2(3).
- Synopses. The synopses used in the study are available for download from: <http://tiny.cc/3o76bz>.
- Yordanova, K. 2016. From Textual Instructions to Sensor-based Recognition of User Behaviour. In *Proc. of 21st Int. Conf. on Intelligent User Interfaces, IUI Companion*. ACM.