Narrative Plan Generation with Self-Supervised Learning

Mihai Polceanu, [1] **Julie Porteous,** [2] **Alan Lindsay,** [3] **Marc Cavazza** [1]
[1] School of Computing and Mathematical Sciences, University of Greenwich, London, UK
[2] School of Computing Technologies, RMIT University, Melbourne, Australia
[3] School of Mathematical and Computer Sciences, Heriot-Watt University, Edinburgh, UK
m.polceanu@gre.ac.uk, julie.porteous@rmit.edu.au, alan.lindsay@hw.ac.uk, m.cavazza@gre.ac.uk

Abstract

Narrative Generation has attracted significant interest as a novel application of Automated Planning techniques. However, the vast amount of narrative material available opens the way to the use of Deep Learning techniques. In this paper, we explore the feasibility of narrative generation through self-supervised learning, using sequence embedding techniques or auto-encoders to produce narrative sequences. We use datasets of well-formed plots generated by a narrative planning approach, using pre-existing, published, narrative planning domains, to train generative models. Our experiments demonstrate the ability of generative sequence models to produce narrative plots with similar structure to those obtained with planning techniques, but with significant plot novelty in comparison with the training set. Most importantly, generated plots share structural properties associated with narrative quality measures used in Planning-based methods. As plan-based structures account for a higher level of causality and narrative consistency, this suggests that our approach is able to extend a set of narratives with novel sequences that display the same high-level narrative properties. Unlike methods developed to extend sets of textual narratives, ours operates at the level of plot structure. Thus, it has the potential to be used across various media for plots of significant complexity, being initially limited to training and generation operating in the same narrative genre.

Introduction

Narrative Generation has attracted significant attention in the Entertainment AI community for its potential to develop new media genres such as Interactive Narrative or provide new gameplay content in computer games (Riedl and Bulitko 2012). There has been substantial work in the use of Planning technologies (Riedl and Young 2010) to support the generation of consistent plots, in particular preserving the causality of narrative actions (Young et al. 2013). Planning technologies can support generation in various media, from 3D graphics to text or even video, as they are essentially media neutral. It has also been demonstrated, through various user evaluations of the end-product narrative, that plan-based narratives are recognized as realistic by users and that sophisticated aspects such as sub-plots (Porteous, Charles, and Cavazza 2016) or narrative trajectories (Porteous et al. 2011), which play an important role in story quality, can be captured by Planning representations. With the increasing availability of large-scale narrative resources (Finlayson 2013), an alternative approach could be to harness Deep Learning (DL) methods that have demonstrated their appropriateness for sequence generation.

In this paper, we wish to first demonstrate the basic ability of DL methods, learning from a dataset of well-formed plots, to produce novel narrative sequences, which preserve those story properties that justified the previous use of planning in narrative generation. We thus use as a dataset plot sequences produced by a narrative planning approach that adopts a plot-centric perspective (Riedl and Young 2010), to train Generative Recurrent Neural Networks. We then analyze the generated plots for their plan well-formedness, diversity, and narrative properties. This method for training DL networks with "synthetic" data is also inspired from the use of computer graphics to train DL models in computer vision (Qiu et al. 2017).

Previous and Related Work

The use of planning techniques is well-established in narrative generation (Riedl and Young 2010), and has been used to represent complex structures such as narrative arcs (Porteous et al. 2011) and discourse-level phenomena (Young et al. 2013). It is primarily intended to generate a plot structure in which planning operators represent core narrative actions. Wang et al. (2017) have complemented traditional plan-based narrative generation with a deep Reinforcement Learning (RL) component, aimed at personalizing narrative generation. Finlayson (2015) is one of the earliest examples of applying Machine Learning (ML) to plot structures, using Propp's narrative functions as elementary narrative units. Finlayson's system operates through annotation of textual stories with narrative functions and aims to extract a plot structure independent of its linguistic realization. Subsequent ML approaches directly process textual stories without narrative annotation, thereby acquiring simultaneously plot structure and semantic aspects: their plot structures are however often of limited length. Xu et al. (2018) generate a short textual narrative from a one-sentence thematic description using a combination of Seq2Seq (Sutskever, Vinyals, and Le 2014) encoder-decoders with RL to con-

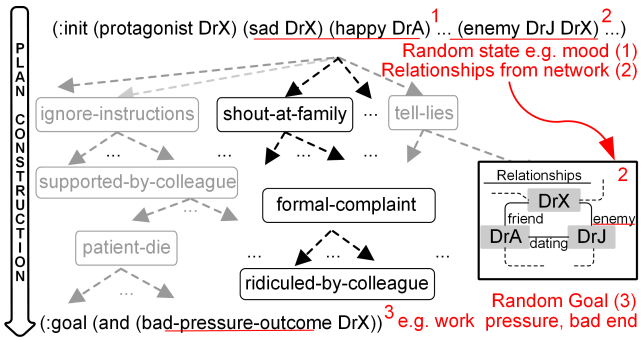

Figure 1: Narrative plan and function labelling for an example medical drama domain. Planning domain and problem properties. Plans are obtained for problem instances with initial state and goal created randomly using thematic categories (e.g. character mood) and relationships (e.g. social network as shown).

nect to a plot-like ("skeleton") extraction module. Roemmele and Gordon (2018) describe an encoder-decoder approach to predict causal relations in stories. Martin et al. (2018) use a recurrent multilayer encoder-decoder network (event2event) to generate the next event from a given narrative sequence. This work has been extended (Tambwekar et al. 2019) to improve plot generation by including a RL component, so that the generated plot could be goal-driven, as with planning approaches. Ammanabrolu et al. (2020) have further enhanced the approach by extending the event-to-sentence models to ensure semantic and narrative coherence. Liu et al. (2020) use story character embeddings as a skeleton and use next-step prediction to generate actions at the event level. In a similar fashion but applied to the visual domain, Asai and Muise (2020) narrow the gap between high-dimensional image data and symbolic planning states by using an encoder-decoder architecture combined with a classical planner to solve visual n-puzzle tasks. Recent work by Yao et al. (2019) has investigated simultaneously short plot structures and textual descriptions in their "Plan-and-Write" system. Like Martin et al. (2018) they decompose story generation into story structure modeling and structure to surface generation. Storyline planning uses bidirectional Gated Recurrent Units (GRU (Cho et al. 2014)) while story generation uses a Seq2Seq approach with Long Short-Term Memory (LSTM (Hochreiter and Schmidhuber 1997)) decoding and bidirectional-LSTM encoding. Finally, Wang and Wan (2019) developed a story completion model that generates the final sentence of a short textual story, using a transformer-based conditional Variational Autoencoder (VAE). Our work aims to revisit the larger-scale plot generated by planning approaches from a DL perspective, separating plot structure from its media or linguistic realization; we will still seek inspiration from some of the above-described sequence generation models, but instead learn representations of plot structures with generative neural networks.

Plot Dataset Preparation

For the purpose of training a DL model, 4 synthetic plot datasets were created for use in experiments: using 4 different previously published narrative planning domains to generate well-formed narrative plans via random search sampling. Details of the narrative planning domains and the process of narrative plan dataset generation are given in the next two sections.

Narrative Planning Domains

Our approach is based on the formal equivalence between narrative annotation (e.g., based on a narrative ontology such as narrative functions) and planning operators to describe the sequence of actions constituting a plot. In a plot-based approach, a planning operator corresponds to an elementary narrative action that may coordinate several characters: this corresponds to the structure of a narrative function (without necessarily fully subscribing to a Proppian approach). The sequence of operators in a narrative plan constitutes the plot representation.

As an example, consider a medical drama domain, similar to (Porteous, Charles, and Cavazza 2016), shown in Figure 1. In this domain the operators are drawn from the narrative modelling of this drama and represent narrative actions such as {*shout-at-colleague, ignore-instructions*} and {*misdiagnose-patient*} which are actually similar to the Proppian functions used in narrative annotation work (Finlayson 2015) being domain-specific instantiations of prototypical narrative functions such as "villainy" or "erroneous judgement". There are long distance effects within the story arc: early choices (e.g. the action {*shout-at-family*}) force later choices (those not greyed out in the search).

For the experiments in this paper we selected a range of narrative planning domains based on the following criteria: they had appeared in the literature; provided a range of narrative contexts; and had been modelled independently by different authors for different purposes. These domains are: *Aladdin* (Riedl and Young 2010); *Western* (Ware 2014); *Red* (Riedl 2009); and *Medical* (Porteous, Charles, and Cavazza 2016). For experiments we used PDDL 2.1 variants of these, which have been made available by the authors (*Aladdin*, *Red* and *Medical*), or automatically translated from an ADL encoding[1] (*Western*).

For these domain models the number of ground actions (i.e., when all parameters in actions are instantiated to domain objects) and the average length of goal-directed narrative plans (e.g., using a classical planner such as METRIC-FF[2]) are shown in Table 1.

The process of Narrative Dataset preparation is illustrated in Figure 2 (part 1). Input to the narrative generator is the narrative domain model and a planning problem. The random selection of initial state predicates and goal conditions for these problem instances was mitigated via the use of thematic categories such as narrative themes (e.g. romance,

[1] Using the translation from (Gazen and Knoblock 1997).

[2] Available to download from: https://fai.cs.uni-saarland.de/hoffmann/metric-ff.html

pressure of work), narrative outcomes (happy or sad), personal relationships (e.g. enemy, friend), to ensure balanced coverage of narrative topics and themes in the resulting generated plots. The same process is repeated for all domains in our experiments to generate a dataset for each.

Narrative Plan Dataset Generation

We developed a bespoke mechanism (referred to herein as Narrative Generator) for generating plot datasets in this work based on the fact that applying a classical planner[3] to the given problems does not generate a sufficiently diverse set of plans, also considering that strict optimality is not a concern when generating plan-based narratives. The aim with our approach was to generate plans which balance goal directedness while allowing for sufficient variation. The generated plans are constituted by a sequence of operators corresponding to narrative actions, therefore each plan in turn corresponds to one plot (we will use the terms plan and plot interchangeably in this context). Domains which were originally defined using the Action Description Language (ADL), which allows for parametrizable actions, were instantiated[4] into PDDL.

The Narrative Generator constructs plans for problem instances through a process of forward random search sampling (Figure 2.1), without using a structural distance heuristic as follows: forward search from the initial state (current state) with repeated random selection of the next applicable action and advancing the current state via action application. The search terminates with either success (reach a state satisfying the goal condition) or failure (a dead end i.e., no applicable actions, or a previously visited state). On success, unique plans are added to the plot dataset, otherwise the search is repeated for the same problem instance. This process was repeatedly called and resulted in a training dataset (DS) of 3072 ($3 * 2^{10}$) unique plans each (excluding validation dataset), for each domain, which were randomly shuffled. Subsets of each dataset were used to explore model performance with less data (*i.e.* dataset sizes: $small = 1024$, $medium = 2048$, $large = 3072$). The properties of datasets of narrative plans are listed for each domain in Table 1.

Our narrative generator proved effective at generating plots of different lengths that exhibit high variation. With regard to plan length, from the properties listed above it can be seen that the random generation method tends to produce longer plans, in particular through repetition of non-critical actions, which is in contrast to those generated by a classical planner such as METRIC-FF[2] which generates plans of much shorter length for the same domain, as the default planning mechanism tends towards optimality (shortest plan) even if it has been shown not to be a requirement for narrative generation (Porteous et al. 2011). The narrative generator also proved very effective at generating plots with high variation in terms of operators used, despite domain difficulty.

Finally, for evaluation purposes, we manually tagged the decisive operators in each of those domains used for our experiment. These are referred to as *beats*[5] and their distribution can serve as a means to measure the narrative interest within the generated plots (Mateas and Stern 2005; Young et al. 2013).

	Aladdin	*Medical*	*Red*	*Western*
l_{DS}	23.4 (5.2)	26.8 (2.8)	11.5 (3.7)	37.7 (3.8)
a_{DS}	38	109	70	285
l_{CP}	11.3 (0.5)	20.0 (0.0)	6.0 (0.5)	9.0 (0.0)
a_{ADL}	11	N/A	5	19
a_{PDDL}	72	141	106	350
$repeat$	✓	✗	✓	✓
BF^{π}	7.08 (2.75)	27.65 (14.47)	5.10 (2.81)	41.80 (3.22)

Table 1: Synthetic dataset characteristics. Top part illustrates the average length of plans (and standard deviation) in the Dataset (l_{DS}) created with the Narrative Generator as well as the number of actions (a_{DS}) that appear in these plans. At the bottom are the average lengths (and standard deviations) of plans found by the Classical Planner (l_{CP}), followed by the number of parametrizable operators (a_{ADL}) and their instantiations (a_{PDDL}), whether these actions can be performed more than once ($repeat$), and finally the average branching factors (and standard deviation) for each domain (BF^{π}, analysis of branching factor sampled from a plan).

Neural Architectures

Our goal is to obtain a generative method that performs well without relying on prior expert knowledge of the difficulty and properties of the domain. To avoid bias from specific hyperparameters (*e.g.* number of layers, hidden state size, etc) which might lead to an advantage when tuning for each particular dataset, we use instead a common framework (Neural Model, depicted in Figure 2.2) across all experiments with identical hyperparameters. The Neural Model explored herein follows, in a broad sense, the *sequence-to-sequence* architecture, an encoder-decoder architecture which has been commonly used for both supervised and unsupervised learning and has proven particularly successful in language modeling tasks (Seq2Seq (Sutskever, Vinyals, and Le 2014)). In order to experiment with various regularities and dependencies in narrative plots, we opted for a spectrum of sequence embedding encoder-decoder models inspired from previous work. To this end, the framework is then instantiated to obtain the architecture variants used in this paper. In the following we discuss each model instance and provide essential implementation details.

The Neural Model (Figure 2, part 2) consists of an encoder which maps a narrative plot to a single latent vector, and a decoder which generates a plot from this vector. Local

[3]Using an iterative top-k planning approach (Katz et al. 2018).

[4]adl2strips software available to download from: http://fai.cs.uni-saarland.de/hoffmann/adl2strips.zip

[5]We have adopted a conservative approach to the distribution of beats, only tagging as beats some of the most dramatic operators that introduce significant new information in the narrative progression also relying on the PDDL structure.

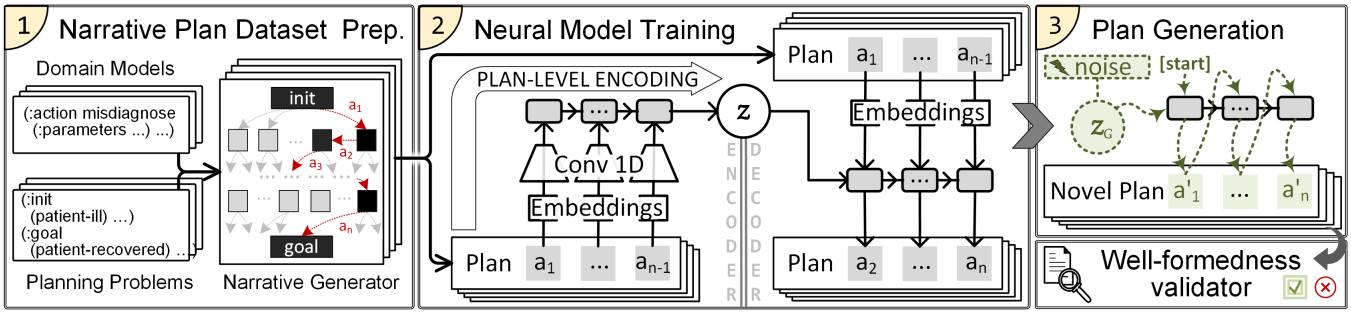

Figure 2: System Overview: Narrative Plan Dataset Preparation (1), Model Training (2) and Plan generation (3). Unique narrative plans are generated via forward random search sampling (see text) (1). The resulting dataset is used for model training which consists in a Plan-level encoding, which is afterwards used to initialize the hidden state of the decoder. During training, the decoder is trained semi-supervised on the same input of the encoder, but by forcing next-step prediction (2). For sampling new sequences from a trained model, the decoder subnetwork is used with noise (z_G) as substitute for the latent vector to recurrently generate complete narrative plots (illustrated with dotted arrows in 3). Finally, generated sequences are checked for plan well-formedness.

structure in plans, for instance, non-critical actions that are performed in-between major *beat* events contribute to story variation but have less effect on the overall outcome. The encoder uses a one-dimensional convolutional layer (Conv 1D) over the operator embeddings, before feeding information into the RNN that traverses the sequence. This helps capturing intermediate-level regularities in sequences, which in narrative plots could correspond to various narrative phases of story evolution. The encoding vector z is obtained by transforming the last hidden state of the RNN into a vector using a fully-connected layer. Decoding a latent vector consists in initializing the hidden state of a separate RNN with a linear transformation of the previously obtained latent vector (z) and traversing the same input sequence while trying to predict the next operator in the sequence (*i.e.* output is shifted by one).

Generating sequences with the common encoder-decoder framework generally consists in next-step predictions conditioned on past inputs, but is limited in generative ability (*i.e.* the output is deterministically conditioned only on past input). Recently, Ghosh et al. (2020) have shown that Regularized Autoencoders (RAE) allow random sampling of the latent space of deterministic models which increases the stability of training generative models. We combine Seq2Seq model with the RAE sampling approach to obtain the first framework instance used in our experiments, a generative recurrent autoencoder (***AE***) model. To quantify the importance of encoder-decoder architecture and of the adopted sampling method, we utilize a simplified variant of this approach as a baseline, which represents a middle ground between previous works and ***AE***, by applying the same sampling method but removing the encoder. This results in an LSTM generative model whose initial hidden state is computed from a random latent code z, denoted as ***LSTM*** in experiments. As a second baseline, we compare our results with a representative of the VAE generative model family. Since recurrent VAE models which incorporate LSTM were proposed for modeling speech and handwriting and the importance of including randomness to generate novel outputs was highlighted (Chung et al. 2015), there have been extensive efforts to improve their sampling quality and training stability (Shen and Su 2018). One such strategy (named ***VAE*** in our experiments) to address this challenge is to gradually adjust the importance of the KL divergence term in the Evidence Lower Bound (ELBO) objective (*i.e.* annealing) in combination with input dropout in the decoder as shown by (Bowman et al. 2016), which is suitable for generating diversity from limited sample sizes. The construction of z in ***VAE*** follows the variational interpretation ($z = \mu + \sigma * N(0, 1)$, which represents a sample from a multivariate Gaussian distribution with mean μ and variance σ). Other variants from the family of Variational Autoencoders were considered, notably β-VAE (Higgins et al. 2017) which introduces a hyperparameter to control the importance of the KL divergence in the ELBO objective similar to (Bowman et al. 2016), while Zhao et al. (2019) introduced InfoVAE, a generalization of β-VAE and Adversarial Autoencoders (AAE) (Makhzani et al. 2016). Generative Adversarial Networks (GAN) have also been extended to sequence modeling (SeqGAN (Yu et al. 2017)) which employ RL training, with varying levels of convergence stability.

To generate novel samples from the trained models (Figure 2.3), we use samples from a random distribution as a substitute for the latent vector (z_G) to prime the decoding RNN. The resulting plan is then constructed in an autoregressive fashion. Models used in our experiments use weight decay through the AdamW optimization algorithm (Loshchilov and Hutter 2019) and Dropout (Hinton et al. 2012) for model-wise regularization. All hyperparameters were kept the same for all models, which led to an identical decoder responsible for the process of generating novel sequences. To stabilize training of ***VAE***, dropout was applied on the tokens of the sequence, forcing the model to learn to reconstruct missing inputs as described in (Bowman et al. 2016) and used GRU and sigmoid KL term annealing centered at 10000 epochs. The latent space (z) and RNN

Domain		Average f_G		
name	*dataset*	***AE***	***LSTM***	***VAE***
Aladdin	small	**3.46 (0.40)**	**3.78 (0.59)**	**3.39 (1.06)**
	medium	**1.56 (0.59)**	1.02 (0.14)	**1.63 (0.36)**
	large	**0.91 (0.29)**	0.44 (0.09)	**0.94 (0.36)**
Medical	small	**2.14 (0.73)**	0.84 (0.26)	**2.43 (1.74)**
	medium	**1.94 (0.28)**	0.38 (0.19)	**1.55 (0.94)**
	large	**1.57 (0.28)**	0.17 (0.08)	0.96 (0.48)
Red	small	2.18 (0.26)	**3.10 (0.60)**	1.04 (0.37)
	medium	**1.38 (0.23)**	**1.21 (0.36)**	0.66 (0.16)
	large	**0.93 (0.25)**	0.71 (0.14)	0.45 (0.15)
Western	small	**0.21 (0.17)**	**0.06 (0.14)**	<0.01
	medium	**0.23 (0.24)**	0.01 (0.01)	<0.01
	large	**0.25 (0.25)**	**0.11 (0.25)**	<0.01
Cross-domain robust		✓	✗	✗

Table 2: Model performance comparison across domains. The generative factor (f_G) is the ratio between the number of well-formed plans (out of 10000 generated samples) and the dataset size. Average and standard deviation reported, with statistically significant best values in bold ($p < 0.05$, two-sample t-test).

hidden state were kept at fixed sizes of 16 and 64 neurons respectively to preserve a consistent information bottleneck throughout all experiments. A kernel of size 5 was used in all experiments in the 1-dimensional convolutional layer followed by a size 3 max pooling layer in the encoder to capture local structure in plans. Hyperparameter values were chosen so that the decoder, which is identical in all tested variants, does not overfit on any of the datasets (*i.e.* the generative potential of all variants is essentially the same, as only the decoder is used in the generation step).

Plot Generation Experiments

The main objective of our approach is to generate *novel*, *well-formed* plans that preserve narrative consistency and also retain the structural characteristics necessary for *narrative interest*, from a limited training dataset. In the following we first evaluate the ability of generative models to capture the constraints that underlie planning domains, followed by an analysis of the quality of generated samples.

Generated Plan Novelty

In order to attain target properties for the plans generated by our approach, we rely on the exploration of the learned latent representations of plans within the trained model; *i.e.* performing common random sampling of the latent code allows us to check whether generated sequences are novel and well-formed plans. We trained the aforementioned model variants on each Planning domain, relying on the validation Negative Log Likelihood (NLL) loss and number of unique sequences over total generated (efficiency). A well-formedness validator[6] was used as an oracle to quantify the amount of plans among the unique sequences (accuracy).

Model performance on the task of generating novel narrative plots relies not only on the percentage of well-formed plans, but on a balance between the accuracy, the capability to generate a reasonable number of novel sequences with desirable narrative properties and the ability to capture long-term dependencies. In our plan novelty experiments, the results of which are detailed in Table 2, we aim to evaluate this balance by introducing a "generative factor" (f_G), defined as the ratio between the number of well-formed plans generated by each model variant and the size of the dataset, which measures novelty with respect to the training dataset. Let us consider the practical significance of f_G: a value of 2 for a small dataset (1024 training plots) indicates the ability to generate over 2000 novel plots, which compares favourably with the generative ability of plan-based approaches[7].

Although well-formedness is not a strict requirement for narrative storytelling, imposing this additional constraint sets stricter requirements on the model performance in our experiments. The previously proposed f_G integrates well-formedness accuracy as well as the efficiency of plan generation. In Table 2 we report the f_G values obtained by the three model variants on different amounts of data. Statistically significant ($p < 0.05$) best values in bold show that the most promising of the tested architectures is ***AE*** which is able to generate a significant proportion of completely novel sequences which are formally well-formed plans (all constraints of the planning domain are satisfied), without posing stability issues. We note that for *Aladdin* and *Medical*, ***AE*** matches ***VAE*** which is able to accurately generate more plans compared to our ***LSTM*** baseline, while on *Red* and *Western* ***AE*** matches the stable increased performance of ***LSTM***. We also note that ***LSTM*** and ***VAE*** achieve good results, both in terms of accuracy and overall f_G, on *Red* and *Aladdin* respectively, but their performance is not consistent throughout the entire suite of tests, which is due to a lack of context information at initialization in the case of ***LSTM*** and training instability in the case of ***VAE***. In contrast, ***AE*** is consistently and accurately capturing long-term dependencies between actions and is robust across all planning domains as it can be seen from significantly high f_G for diverse planning domains and dataset sizes. These results show that generative models are indeed a suitable approach to narrative plan generation, as all variants are able to obtain quality results on at least one planning domain. An analysis of the remaining accuracy potential of all models showed that this is due to narrative sequences which revisit one or more planning states (*i.e.* subsequent actions lead to a state S_j which is equivalent to state S_i where i precedes j in time), or repeated actions which do not satisfy the strict planning constraint of well-formedness but still constitute acceptable

[6]We use an alternative implementation to (Rintanen 2008) to compute predecessor states by action formula (i.e., pre- and post-condition) manipulation and test applicability of the sequence in the resulting state to show domain rule compliance.

[7]As for future availability and size of real-world datasets we can note anecdotally that The Simpsons have reached over 600 episodes and NCIS over 300.

	Aladdin	*Medical*	*Red*	*Western*
B_{CP}	0.20 (0.04)	0.10 (0.00)	0.39 (0.10)	0.44 (<0.01)
B_{AE}	0.15 (0.03)	0.12 (0.02)	0.39 (0.09)	0.29 (0.06)
D_{CP}	0.35 (0.08)	0.15 (0.0)	0.29 (0.13)	0.22 (0.03)
D_{AE}	0.43 (0.16)	0.11 (0.02)	0.19 (0.05)	0.22 (0.07)

Table 3: Analysis of beat content in plots from each source: Classical Planner (CP) and ***AE***, illustrating averages and standard deviations of the number of beats per length (B), and the distance between beats per length (D).

	Aladdin	*Medical*	*Red*	*Western*
Kt	+0.412 $p < 10^{-3}$	+0.570 $p < 10^{-4}$	+0.388 $p < 10^{-4}$	+0.119 $p < 0.03$
Ld	11.66 (2.33) m:0 M:18	16.15 (3.28) m:3 M:25	9.49 (2.8) m:0 M:17	10.80 (2.2) m:3 M:20

Table 4: Agreement and diversity analysis between Classical Planner and ***AE***. Comparison of operator ranks using Kendall's tau (Kt) with statistic value (+ indicates positive agreement). Plan diversity indicated by pair-wise Levenshtein distance (Ld) average and standard deviation, and minimum (m) and maximum (M) distances.

narratives. This is likely due, at least in part, to the modest training dataset sizes, with more data being necessary to derive all the rules behind the narratives, as indicated by the significant increase in accuracy with larger amounts of data which we observed in our experiments. We also note there is a slight decrease of f_G with larger datasets, which is primarily due to a limit in generating new plans as the larger the dataset the more plans are visited, as well as a limit of the chosen model capacity which is suitable for increase for ***AE***, but leads to overfitting in ***LSTM*** and to relatively high instability in ***VAE***.

All experiments were performed on a setup of 8 NVIDIA™ GTX 1080 Ti GPUs, 32 Intel™ Xeon CPUs with 128GB RAM running Ubuntu OS. The PyTorch 1.6.0 library (Paszke et al. 2019), which made possible the implementation and control of key aspects for each of the models and experiments herein, and the NVIDIA™ CUDA Toolkit 10.1 were used for all experiments. In the novelty experiments, random initialization was used for neural network parameters and sampling, and experiments were rerun deterministically with consistent random number generator seeds $(0-7)$ and results were averaged, while quality analysis was performed on the previously trained ***AE*** model.

Evaluating the Quality of Generated Plots

We evaluate our approach on its ability to generate novel plots not part of any training set, as well as the narrative quality of those generated plots, the latter being measured through structural properties identified by previous work in narrative generation (Porteous, Charles, and Cavazza 2016; Amos-Binks, Roberts, and Young 2016). Previous work in DL-based generation has relied on user evaluations to rate generated narratives (Martin et al. 2018; Yao et al. 2019), however this is only achievable in practice if the number of narratives is limited, and the narrative has a discourse-level presentation, often as text, instead of symbolic plot sequences. Our system generates a sizeable number of plot sequences represented as a list of operators, which make it difficult to envision user evaluation as described above. Previous work in plan-based narrative generation has identified a number of structural criteria for narrative quality that can be subject to automated testing. Interestingly, even user evaluations have confirmed the importance of structural quality criteria, such as plot coherence (Yao et al. 2019), which are largely guaranteed when the plot is a well-formed plan. As a tentative measure of narrative quality, we use overall structural properties that give an indication over story pacing using the distribution of *beats* (Mateas and Stern 2005; Young et al. 2013) within the plot sequence. In addition, we also investigate the composition of plots in terms of individual actions (operators) for ***AE***-generated and plan-generated plot sequences.

We have annotated specific operators as *beats*, *i.e.* events that have significant impact in the story from a narrative point of view, post-training (*i.e.* the model does not have access to beat information). Narrative sequences generated during our experiments using ***AE*** were used in comparison to plans found by a Classical Planner (CP) under the aforementioned quality metrics (results in Table 3). The first metric (B) consists in the overall occurrence of beats within the generated plot, which acts as a high-level measure of dramatic content (these have to be normalised according to plot length). The second metric (D) which is the average distance between beats, approximates story pacing. These two metrics are used not as absolute values but for their comparison to plan-based generation, as in the latter such structural properties have been shown to be indicators of narrative quality and believability. The generated plots maintain characteristics within an acceptable range (*i.e.* the number of beats per length (B) does vary between ***AE*** and ***CP*** but lies within close proximity, with no statistically significant difference for *Red* and roughly within one and two standard deviations for *Medical* and *Aladdin* respectively, while the higher discrepancy in *Western* is likely to be caused by the difference in plan length (previously reported in Table 1, *i.e.* l_{DS} and l_{CP}). The distance between *beats* (D) varies as well, but again lies roughly within one standard deviation across most sources, with no major discrepancies. Considering the average length of generated plot sequences, these results are compatible with recommendations for story pacing (Porteous et al. 2011)) of existing classical approaches to narrative plot generation, and strongly suggest that ***AE***-based generation is able to produce story paced in a similar way to plan-generated sequences thus preserving important narrative properties associated with story quality.

Moreover, to estimate whether operators have similar prevalence in generated plans, we ranked each operator according to its occurrence frequency in each source. The obtained histograms were used to compute the Kendall rank

correlation coefficient (Kt) to verify whether there is indeed a significant level of agreement between plans found by the classical planner and those generated by ***AE*** in terms of their use of available narrative actions represented by those operators. Results in Table 4 (top) strongly indicate positive agreement between the frequency rankings of operators from the two sources, which means that operators used in CP plans are as prevalent in the model output, suggesting that ***AE*** generation preserves narrative content by making use of narrative actions without introducing bias in action selection.

Finally, to evaluate how varied novel generated plots are, in comparison with plans found by CP, we rely on a distance metric between plot sequences themselves. Measuring distances between narratives is a standard method to investigate their diversity (Roberts et al. 2007). Here we relied on the Levenshtein distance (Ld) (Levenshtein 1966; Porteous, Charles, and Cavazza 2016). Results in Table 4 show high average numbers of edits between CP and ***AE*** plots with particularly wide variability. For instance, high numbers of possible edits (Table 4 maximum values), such as for the $Medical$ domain, indicate that the model learns to restructure entire plots without sacrificing quality criteria. It is interesting to note that, in some instances where the random search dataset contained a number of plans that coincided with those generated by the classical planner (*Aladdin* and *Red*), the model generated novel plans (*i.e.* not part of its training dataset) which were also generated by the planner (*e.g.* Ld minimum of 0 for $Aladdin$ and Red). This suggests an ability to generalize over the constraints which underlie the planning domains (*i.e.* there is no overlap of CP samples and the training dataset). This indicates the fact that in addition to generating plot sequences exhibiting relevant structural properties, the model is also able to produce plans with high levels of diversity.

Finally, beyond the overall number of beats and their average distribution within plot sequences, we also have observed anecdotal, yet interesting, phenomena regarding comparative beats localization. For instance, in the *Medical* domain, when comparing the set of generated sequences of similar lengths (20+) produced by ***AE*** to those generated by ***CP***, we noted a striking similarity in the occurrence of beats primarily towards the last 2/3 of the plot. This suggests that ***AE*** generation has been able to capture some narrative evolution properties of this more constrained domain, which tends to produce narratives with a climax and resolution.

Conclusions and Further Work

Our results suggest that DL methods can indeed be used to generate novel plot variants from a training set of formalized plots while preserving structural properties associated with narrative interest, such as beat distribution. This approach to narrative generation is intended to be genre-specific, like those based on Planning techniques. It makes no claim to open story generation, and its paradigm would be the generation of new episodes for drama series, or new stories in typical narrative genres. The workflow presented in this paper can be adapted for practical plot generation through automatic filtering of the ***AE***-generated sequences using quality criteria such as those used in our evaluation sections. This would hide the noisy aspects of generation to users, only presenting them with the most relevant candidates to be further sorted by automatic structural quality criteria. It should be noted that future use of DL-based generation could relax the requirement for the generated plots to be well-formed plans: this constraint has been imposed as part of this study to ensure that structural properties and long-distance dependencies could be captured and was justified by the use of plan-based "synthetic data" for DL experiments rather than actual annotated narratives.

Compared to other recently published applications of DL methods to narrative generation (Tambwekar et al. 2019; Yao et al. 2019; Ammanabrolu et al. 2020; Liu et al. 2020), which have primarily used text generation, our approach focuses on plot structure, aiming at retaining media independence, as well as full plot generation rather than story completion or continuation. This work contributes to DL-based narrative generation and takes advantage of the growing interest in narrative corpora, being increasingly available in various media (Finlayson 2013). Narrative annotation can itself be complex and time-consuming (Finlayson 2015). This is why our approach, with its ability to train on modest amounts of data[8], could prove even more relevant. Moreover, narrative annotation could be partially automated by information extraction methods (Winer and Young 2017), or using similar methods to those previously described for event extraction from textual stories or even directly analyzing media content in terms of actions using convolutional networks as initial layers, in a way similar, yet more complex to research on automatic captioning. The end-product of narrative generation will eventually consist of a media representation of the story, for instance through game engine-based animated sequences (Porteous, Charles, and Cavazza 2016). Another interesting line of improvement consists in adding controllable latent codes to enable generation focused on particular plot characteristics, either objective such as length or pace, or subjective tags such as mood, based on user preferences, by conditioning the DL model on such metrics.

One limitation of our work, which it shares with some previous plan-based generation approaches is to devise methods to explore the set of generated plots to potentially select the best candidates. A first step in this direction is the use of beat distribution in the generated narratives. Other automatic quality metrics (*e.g.* computing the "plan trajectory" in a way not dissimilar to (Porteous et al. 2011) by approximating it through the variation of the planning heuristic function calculated *post hoc* on the generated plan) could provide insight into high-level elements of the narrative, if supported by appropriate visualization and data analysis tools.

Acknowledgements

Contributions of Julie Porteous were supported, in part, by funds from DSI Collaborative Grant CR-0016.

[8]For some planning domains, we were able to train our model with even smaller datasets, *e.g.* 500 sequences.

References

Ammanabrolu, P.; Tien, E.; Cheung, W.; Luo, Z.; Ma, W.; Martin, L. J.; and Riedl, M. O. 2020. Story Realization: Expanding Plot Events into Sentences. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, 7375–7382.

Amos-Binks, A.; Roberts, D. L.; and Young, R. M. 2016. Summarizing and comparing story plans. In *7th Workshop on Computational Models of Narrative (CMN 2016)*. Schloss Dagstuhl–Leibniz-Zentrum für Informatik.

Asai, M.; and Muise, C. 2020. Learning Neural-Symbolic Descriptive Planning Models via Cube-Space Priors: The Voyage Home (to STRIPS). In *Proceedings of the 29th International Joint Conference on Artificial Intelligence*, 2676–2682. AAAI Press.

Bowman, S. R.; Vilnis, L.; Vinyals, O.; Dai, A.; Jozefowicz, R.; and Bengio, S. 2016. Generating Sentences from a Continuous Space. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*, 10–21. Berlin, Germany: Association for Computational Linguistics. doi:10.18653/v1/K16-1002.

Cho, K.; van Merriënboer, B.; Bahdanau, D.; and Bengio, Y. 2014. On the Properties of Neural Machine Translation: Encoder–Decoder Approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, 103–111. Doha, Qatar: Association for Computational Linguistics. doi:10.3115/v1/W14-4012.

Chung, J.; Kastner, K.; Dinh, L.; Goel, K.; Courville, A. C.; and Bengio, Y. 2015. A Recurrent Latent Variable Model for Sequential Data. In Cortes, C.; Lawrence, N. D.; Lee, D. D.; Sugiyama, M.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 28*, 2980–2988. Curran Associates, Inc.

Finlayson, M. A. 2013. A Survey of Corpora in Computational and Cognitive Narrative Science. *Sprache und Datenverarbeitung, Special Issue on Formal and Computational Models of Narrative* 37(1-2): 113–141.

Finlayson, M. A. 2015. ProppLearner: Deeply annotating a corpus of Russian folktales to enable the machine learning of a Russian formalist theory. *Digital Scholarship in the Humanities* 32(2): 284–300.

Gazen, B. C.; and Knoblock, C. A. 1997. Combining the expressivity of UCPOP with the efficiency of Graphplan. In *European Conference on Planning*, 221–233. Springer.

Ghosh, P.; Sajjadi, M. S. M.; Vergari, A.; Black, M.; and Scholkopf, B. 2020. From Variational to Deterministic Autoencoders. In *International Conference on Learning Representations*.

Higgins, I.; Matthey, L.; Pal, A.; Burgess, C.; Glorot, X.; Botvinick, M.; Mohamed, S.; and Lerchner, A. 2017. $\beta - VAE$: Learning Basic Visual Concepts with a Constrained Variational Framework. In *International Conference on Learning Representations*.

Hinton, G. E.; Srivastava, N.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. R. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580* .

Hochreiter, S.; and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8): 1735–1780.

Katz, M.; Sohrabi, S.; Udrea, O.; and Winterer, D. 2018. A novel iterative approach to top-k planning. In *Twenty-Eighth International Conference on Automated Planning and Scheduling*.

Levenshtein, V. I. 1966. Binary codes capable of correcting deletions, insertions and reversals. *Cybernetics and Control Theory* 10: 707–710.

Liu, D.; Li, J.; Yu, M.-H.; Huang, Z.; Liu, G.; Zhao, D.; and Yan, R. 2020. A Character-Centric Neural Model for Automated Story Generation. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, 1725–1732.

Loshchilov, I.; and Hutter, F. 2019. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*.

Makhzani, A.; Shlens, J.; Jaitly, N.; and Goodfellow, I. 2016. Adversarial Autoencoders. In *International Conference on Learning Representations*.

Martin, L. J.; Ammanabrolu, P.; Wang, X.; Hancock, W.; Singh, S.; Harrison, B.; and Riedl, M. O. 2018. Event Representations for Automated Story Generation with Deep Neural Nets. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*.

Mateas, M.; and Stern, A. 2005. Structuring Content in the Façade Interactive Drama Architecture. In *First Artificial Intelligence and Interactive Digital Entertainment Conference*, 93–98.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Kopf, A.; Yang, E.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; and Chintala, S. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In Wallach, H.; Larochelle, H.; Beygelzimer, A.; d'Alché-Buc, F.; Fox, E.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 32*, 8024–8035. Curran Associates, Inc. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

Porteous, J.; Charles, F.; and Cavazza, M. 2016. Plan-based narrative generation with coordinated subplots. In *Proceedings of the 22nd European Conference on Artificial Intelligence*. IOS Press.

Porteous, J.; Teutenberg, J.; Pizzi, D.; and Cavazza, M. 2011. Visual Programming of Plan Dynamics using Constraints and Landmarks. In *Proceedings of the 21st International Conference on Automated Planning and Scheduling*.

Qiu, W.; Zhong, F.; Zhang, Y.; Qiao, S.; Xiao, Z.; Kim, T. S.; and Wang, Y. 2017. Unrealcv: Virtual worlds for computer vision. In *Proceedings of the 25th ACM international conference on Multimedia*, 1221–1224.

Riedl, M.; and Bulitko, V. 2012. Interactive narrative: A novel application of artificial intelligence for computer games. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence*.

Riedl, M. O. 2009. Incorporating Authorial Intent into Generative Narrative Systems. In *Proceedings of AAAI Spring Symposium on Intelligent Narrative Technologies*.

Riedl, M. O.; and Young, R. M. 2010. Narrative planning: Balancing plot and character. *Journal of Artificial Intelligence Research* 39: 217–268.

Rintanen, J. 2008. Regression for Classical and Nondeterministic Planning. In *Proceedings of the 18th European Conference on Artificial Intelligence*. IOS Press.

Roberts, D. L.; Bhat, S.; Clair, K. S.; and Isbell Jr, C. L. 2007. Authorial idioms for target distributions in TTD-MDPs. In *Proceedings of the National Conference on Artificial Intelligence*, volume 22, 852.

Roemmele, M.; and Gordon, A. 2018. An Encoder-decoder Approach to Predicting Causal Relations in Stories. In *Proceedings of the First Workshop on Storytelling*, 50–59.

Shen, X.; and Su, H. 2018. Towards Better Variational Encoder-Decoders in Seq2Seq Tasks. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*.

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to Sequence Learning with Neural Networks. In Ghahramani, Z.; Welling, M.; Cortes, C.; Lawrence, N. D.; and Weinberger, K. Q., eds., *Advances in Neural Information Processing Systems 27*, 3104–3112. Curran Associates, Inc.

Tambwekar, P.; Dhuliawala, M.; Martin, L. J.; Mehta, A.; Harrison, B.; and Riedl, M. O. 2019. Controllable Neural Story Plot Generation via Reward Shaping. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. AAAI Press.

Wang, P.; Rowe, J. P.; Min, W.; Mott, B. W.; and Lester, J. C. 2017. Interactive Narrative Personalization with Deep Reinforcement Learning. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, 3852–3858.

Wang, T.; and Wan, X. 2019. T-CVAE: Transformer-based conditioned variational autoencoder for story completion. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, 5233–5239. AAAI Press.

Ware, S. G. 2014. *A plan-based model of conflict for narrative reasoning and generation*. Ph.D. thesis, North Carolina State University.

Winer, D. R.; and Young, R. M. 2017. Automated screenplay annotation for extracting storytelling knowledge. In *Thirteenth Artificial Intelligence and Interactive Digital Entertainment Conference*.

Xu, J.; Ren, X.; Zhang, Y.; Zeng, Q.; Cai, X.; and Sun, X. 2018. A Skeleton-Based Model for Promoting Coherence Among Sentences in Narrative Story Generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 4306–4315.

Yao, L.; Peng, N.; Weischedel, R.; Knight, K.; Zhao, D.; and Yan, R. 2019. Plan-and-write: Towards better automatic storytelling. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, volume 33, 7378–7385. AAAI Press.

Young, R. M.; Ware, S. G.; Cassell, B. A.; 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.

Yu, L.; Zhang, W.; Wang, J.; and Yu, Y. 2017. Seqgan: Sequence generative adversarial nets with policy gradient. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*.

Zhao, S.; Song, J.; and Ermon, S. 2019. Infovae: Balancing learning and inference in variational autoencoders. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, volume 33, 5885–5892.