Constructing narrative using a generative model and continuous action policies

Emmanouil Theofanis Chourdakis and **Joshua D. Reiss***
Queen Mary University of London
Mile End Road, E1 4NS
London, UK
{e.t.chourdakis,joshua.reiss}@qmul.ac.uk

Abstract

This paper proposes a method for learning how to generate narrative by recombining sentences from a previous collection. Given a corpus of story events categorised into 9 topics, we approximate a deep reinforcement learning agent policy to recombine them in order to satisfy narrative structure. We also propose an evaluation of such a system. The evaluation is based on coherence, interest, and topic, in order to figure how much sense the generated stories make, how interesting they are, and examine whether new narrative topics can emerge.

1 Introduction

In this work reinforcement learning is used in conjunction with a shallow generative artificial neural network (ANN) to generate novel stories. First, a SkipGram (Mikolov and others, 2013) based model is derived that generates parts of the narrative in a local neighbourhood (a few consecutive events at time). An artificial agent is then used to extend its use to the whole narrative while globally adhering to the story structure learned by that model.

2 Previous Work

Data-driven approaches for story generation can be found in (McIntyre and Lapata, 2009; Li and others, 2013). In (McIntyre and Lapata, 2009), the authors present an end-to-end system to generate stories by deriving models of interest and coherence and a generator that creates stories by consulting a knowledge base of story elements and their possible interactions. They improved their work in (McIntyre and Lapata, 2010) by generating stories with genetic algorithms instead of specified models for interest. In (Li and others, 2013), the authors recombine events found in a story corpus with a planning algorithm to create novel stories which consist of events in the form of simple sentences. Their novelty relies on that they crowd-source the corpus in natural language sentences and do not need to provide a pre-defined knowledge base. In that work, they use paraphrase identification using weighted dependencies (Lintean and Rus, 2009) in order to group similar events which they use to construct graphs of narration and a planning algorithm to generate new stories. (Riedl and Harrison, 2016) use that work together with Reinforcement Learning in order to teach artificial agents human values. Deep Reinforcement Learning has been explored in the context of natural language generation before in the context of text-based games. In (Narasimhan and others, 2015) the authors introduce a recurrent neural network which they call LSTM-DQN, to characterise the states of the worlds of text-based Multi-User Dungeon games. They then use Deep Q-learning (Mnih and others, 2015) to learn optimal policies for such games. In (He and others, 2016) the authors introduce a novel type of ANN called Deep Reinforcement Relevance Network which allows for separate ANNs to use for the states and actions of the agents allowing actions of arbitrary number or complexity to be taken by the agent. In this work we use such

*Correspondence should be sent to the first author. This paper has been sponsored by RPPtv Ltd.

a network with an actor-critic method and devise a data driven approach for story generation to learn how to construct narratives from a collection of stories.

3 Methodology

3.1 Event Representation

We used 519 stories from the SCHEHERAZADE system (Li and others, 2013)[1] which contains simple stories pertaining to 9 topics with an average length of 7-16 events per story per topic. These stories consist of simple sentences, each describing a single event. Using the Stanford NLP parser, we extract the Universal Dependencies (Chen and Manning, 2014; Nivre and others, 2016) of each sentence as a list of relations in the form $rel(head, modifier)$ where rel is the relation type and $head$, $modifier$ are literals. We further lemmatize each $head$ and $modifier$ using WordNet (Miller, 1995) in order to reduce the total number of literals we have to deal with. Narratives are sequences of events which in turn are simple sentences that describe a character action or a stative. We use universal dependencies and a shallow ANN in order to derive a useful and compact representation for each event. Having derived a set of all the dependencies found in the corpus each event is represented as a vector $\boldsymbol{v}_k$ of the form $[H_{dep1}\ H_{dep2}\ \ldots M_{dep1}\ M_{dep2}]^T$ where H_{dep} corresponds to the head of dependency dep, M_{dep} to the modifier and each of those elements can take as values an integer that serves as the index for the literals found in the corpus.

After we extract a vector $\boldsymbol{v}_k$ for each event k in our corpus, we use an ANN to learn a compact representation of our events such that two similar events have similar representations. Instead of measuring grammatical similarity as in (Li and others, 2013) we consider as similar events the ones that are used in a similar context. For this we use a model similar to the SkipGram (Mikolov and others, 2013). This model derives a low-dimensional fixed-length representation space that maps events that are used similarly, close in that space thus implicitly "grouping" them together. It also gives probabilities of each event happening, based on previous events. The SkipGram model can be seen in Figure 1a.

[1] http://boyangli.co/openstory.php

Choosing such a model allows us to capture relations between neighbouring events, in a similar way to that of the original SkipGram that captures analogies of words in language. We can then use these learned relations to generate events that satisfy them and thus create "coherent" narratives. It also allows us to implicitly group events. This means that, in the process of generating a narrative, when choosing on an event to include, we do have a probability of including a different, but similar, event. Finally, we can use it with events not found in the corpus it has been trained with. As long as we can feed it a vector representation of the new event it will be mapped close to similar events in the corpus. We will see that by using the model generatively to predict the context from a starting event we can already make sensible narratives.

3.2 Generative Model

In Section 3.1 we introduced our SkipGram Model. This model has been trained to give an approximation of the context of an event, given that event. The context of an event in our case consists of the events that immediately surround it. By starting from a random event that can begin a narrative, the model gives the probability of the next event. An example of a narrative generated can be seen in Figure 2b. Generating narratives this way, while it appears adequate, suffers from a serious limitation. Since the model is trained on an event and its immediate surroundings, it is not possible to capture longer distance dependencies in the narrative. In other words, we cannot interrupt a coherent sequence of events and come at it later so the model is "forced" to keep very close to the corpus in order to maintain coherence.

3.3 Deep Reinforcement Learning

Reinforcement learning is the field that studies how an abstract mathematical entity, called an *agent*, can interact with an environment $\mathcal{E}$ in order to maximise a numerical quantity (Sutton and Barto, 1998). We call an instance of the environment at time t a *state* s_t, the quantity to maximise a *utility* U_t. The agent interacts with the environment by executing a series of *actions* a_i and receiving a series of immediate *rewards* r_t. The utility U_t is related to the immediate rewards r_t by the expression: $U_t = \sum_{n=1}^{t} r_n$. The series of actions the agent takes based on the

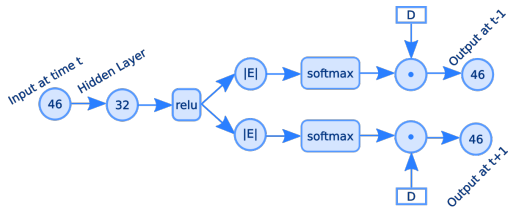
state of the environment is modelled by a *policy* π. The policy can be seen as a probability distribution $\pi(a_t = a_i|s_t)$. The problem of reinforcement learning therefore is to find a policy that maximises the utility for the agent in a given environment. In order to generate policies, RL algorithms usually approximate a *value function* $V(s_t)$ or an *action-value function* $Q(s_t, a_t)$. $V(s_t)$ gives a measure of how beneficial is for the agent to exist at the state s_t and $Q(s_t, a_t)$ how beneficial it is for the agent to be at state s_t and execute action a_t. Deep Reinforcement Learning (DRL) approximates Q, V, $\mathcal{E}$, or π with a Deep Neural Network. A popular approach for training agents works by suggesting an action a_t using a model called an *actor* and evaluates it using a model called a *critic*. The method we use in this work is called Deep Deterministic Policy Gradient (Lillicrap and others, 2016) with the actor and critic models being the deep neural networks that appear in Figures 1b and 1c respectively. The model of the critic is inspired by the Deep Reinforcement Relevance Network given in (He and others, 2016). The actor approximates an event to be included in the narrative and the critic evaluates it based on the current state of the narrative. The state of the narrative is at every point a simple concatenation of the embeddings (as given by the hidden layer in 1a) of the events included in that narrative until that point. At every step the reward is calculated based on the distance of the expected action-event to the selected event so that it awards adding events to the narrative when those are close to the ones we expect to see, and punishes by a small amount unexpected events. Punishing unexpected events might appear counter-intuitive at first glance since story generation systems are expected to generate unexpected events. This is compensated by the stochastic nature of policies found by actor-critic methods which will also assign a small probability to an unexpected event happening.

4 Evaluation

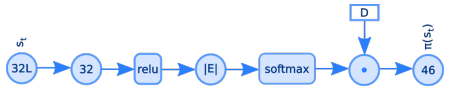
In order to evaluate the system's capability to generate interesting narratives human evaluation is necessary. Towards this goal, an evaluation experiment has been designed which is based on similar evaluation approaches found in data-driven story generation approaches (Li and others, 2013; McIntyre and Lapata, 2010) and asks 20 subjects to evaluate 40 narratives from which 10 are from our corpus of human-made narratives, 10 narratives generated by randomly combining events from the corpus, 10 are narratives generated by the SkipGram Model given in Figure 1a and 10 by the DDPG agent. Each subject evaluates 8 narratives based on number of edits (rearranging, deleting, or adding new events) required to make the narrative more coherent, interest rated on a scale from 1 to 5 (1 being "Not at all interesting" and 5 being "Very Interesting") as well as asked to give one word that better describes the topic of the narrative. This last task can helps us figure out whether new topics emerge from our system by combining events from different topics. Since this is work in progress, we lack experiment results. In the absence of human evaluation results we could do some qualitative examining of generated narratives. Figures 2a and 2c show narratives found in our original corpus and in Figures 2b and 2d narratives generated by the generative model and the DDPG agent respectively. We can see that the narrative in 2b tries to follow the narrative found in 2c however it deviates in its conclusion. Instead of kneeling in front of Sally and proposing, the narrative ends with John kissing Sally. An important note here is that for the most first part of the narrative, the generative model followed almost exactly the story found in the corpus. This is a weakness of the model that arises from learning relations only between neighbouring events. A more interesting narrative is the one found in 2d. This narrative combines events from the narrative in Figure 2a, the one in 2c, as well as others found in the corpus. Narratives generated by the DDPG agent tend to explore more events while narratives generated by the generative model tend to stick to the corpus.

5 Discussion/Future Work

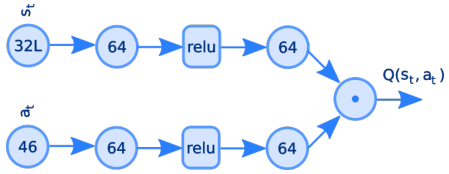
We have presented a system that can learn narrative structure from a collection of stories presented in natural language. This work builds on the work of (Li and others, 2013) and tries to improve it in several ways. First, instead of grouping events based on grammatical similarity we use similarity based on context. In that work, events are also parsed into universal dependencies and grammatical similarity



(a) SkipGram model



(b) actor



(c) critic

Figure 1: The Skipgram model, and the models for the actor and the critic. Circles represent fully connected neuron layers with the number of neurons being the number inside the circle. The smoothed rectangles represent the activation functions with `relu` being the linear rectifier and `softmax` a softmax output. $|E|$ is the number of events in our database, and D the narrative corpus as a matrix of features. The dot symbolises the dot product. L is the number of the events making up the narrative, $\pi(s_t)$ the policy at state s_t, $Q(s_t, a_t)$ the state-action value for the policy π.

between the heads and modifiers of the same dependencies is used to cluster events. This requires similar sentence structure for different events in order for such similarity to be meaningful. We get past this limitation by deriving a fixed length representation by using the model in Figure 1a and thus we are able to compare sentences of variable structure. Since our similarity is based on how events are used in a narrative, we can interchangeably use two sentences that refer to the same event but are grammatically different (e.g. "Sally agreed to marry John." and "Sally said yes.").

The second is that the use of an agent allows for complex relations in the narrative, unknown in advance, to be captured as showcased by (Narasimhan and others, 2015; He and others, 2016), as well as arbitrary authorial goals when carefully introduced in the reward function. One could for example augment the corpus of available events with information about emotions (Francisco and Hervás, 2007)

" *John loved Sally John wanted to marry Sally John bought an engagement ring John took Sally to the park John and Sally enjoyed a picnic John got down on one knee John presented the ring to Sally Sally started to cry John asked Sally to marry John Sally agreed Sally put on the ring John and Sally hugged* "

(a) An example narrative from the corpus.

" *John entered Sally's house. John and Sally entered the living room. John and Sally sat on the sofa. John picked up Sally's hand. John kissed Sally's hand. Sally smiled at John. John let go of Sally's hand. John stood up. John kissed Sally.* "

(b) An example narrative generated by using the SkipGram Model generatively.

" *Sally opened the door. John entered Sally's house. John and Sally entered the living room. John and Sally sat on the sofa. John picked up Sally's hand. John kissed Sally's hand. Sally smiled at John. John let go of Sally's hand. John stood up. John kneeled in front of Sally. John took a ring box out of his pocket. Sally pressed both hands against her cheeks. John proposed to Sally. Sally took the ring box from John. Sally opened the ring box. Sally took the ring out of the ring box. John took the ring from Sally. John put the ring on Sally's left third finger.* "

(c) An example narrative from the corpus.

" *John loved Sally. John presented the ring to Sally. John let go of Sally's hand. Sally and John laughed. Sally and John kissed. John told Sally how beautiful she is. Sally blushed.* "

(d) An example narrative generated by using the DDPG agent.

Figure 2: Examples of narratives.

and then reward events with the desired emotional content. The use of an agent that can also create narrative allows usage in a multiagent, or even interactive environment. This is not very obvious in the current work because experiments have not been yet conducted but an example would be an agent that learned from narratives of topic "proposal", another that learned from "affairs" to work together (i.e. by alternating between the choices of the two agents after a couple of sentences), to produce something in the lines of a "family drama".

The current research leaves some things to be desired. While he have designed an experiment for the evaluation of the system, we have yet to run it through human subjects, who are the ones who can judge if a system exhibits creativity. We cannot therefore have a discussion about whether our system is creative. The narrative generation capacity is limited among other things by the corpus itself. We can only make as many novel stories as can be made by recombining the available events. Given that the vectors of the events (Section 3.1) in the corpus constitute only a limited subset of values in that vector space we should be able to generate novel events mapped from within that space once we had a way to map from narrative to surface text. In (Kumagai and others, 2016), the authors present a system that can generate language given syntactic structure as well as semantic information. Our event vector representation maintains syntactic structure data which could be combined with that work to generate surface text. Another issue is that learning is done exclusively on the narrative-structure level without taking into account any consideration any extra information in the stories. One could use characterisation of story events and heuristics of narration similar to the *STellA* system presented in (León and Gervás, 2014). We speculate that such heuristics can be used as rewards in the context of reinforcement learning and thus guide learning. More technical issues relate to problems that can be met both in reinforcement and in deep learning. Training the networks and the agent is sensitive to hyper-parameters as well as network architecture. Since this is work in progress both the architecture and the hyperparameters have been chosen intuitively by hand and by no means we can claim these are optimal. Better design parameters can be chosen in a robust way through exhaustive cross validation.

References

Danqi Chen and Christopher D Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the EMNLP*.

Virginia Francisco and Raquel Hervás. 2007. Emotag: Automated mark up of affective information in texts. *Proceedings of Doctoral Consortium at the 8th EUROLAN summer school*.

Ji He et al. 2016. Deep reinforcement learning with a natural language action space. In *Proceedings of the 54th Annual Meeting of the ACL*.

Kaori Kumagai et al. 2016. Human-like natural language generation using monte carlo tree search. In *Proceedings of CC-NLG*.

Carlos León and Pablo Gervás. 2014. Creativity in story generation from the ground up: Nondeterministic simulation driven by narrative. In *Proceedings of the 5th ICCC*.

Boyang Li et al. 2013. Story generation with crowdsourced plot graphs. In *27th AAAI Conference*.

Timothy P Lillicrap et al. 2016. Continuous control with deep reinforcement learning. In *Proceedings of ICLR*.

Mihai Lintean and Vasile Rus. 2009. Paraphrase identification using weighted dependencies and word semantics. In *Proceedings of the 22nd FLAIRS*.

Neil McIntyre and Mirella Lapata. 2009. Learning to tell tales: A data-driven approach to story generation. In *Proceedings of ACL-IJCNLP*, pages 217–225. ACL.

Neil McIntyre and Mirella Lapata. 2010. Plot induction and evolutionary search for story generation. In *Proceedings of the 48th Annual Meeting of the ACL*.

Tomas Mikolov et al. 2013. Efficient estimation of word representations in vector space. In *Proceedings of ICLR*.

G. A. Miller. 1995. WordNet: A Lexical Database for English. *Communications of the ACM*, 38(11).

Volodymyr Mnih et al. 2015. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533.

Karthik Narasimhan et al. 2015. Language understanding for text-based games using deep reinforcement learning. In *Proceedings of CEMNLP*.

Joakim Nivre et al. 2016. Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the 10th International Conference on Language Resources and Evaluation*, pages 1659–1666.

Mark O Riedl and Brent Harrison. 2016. Using stories to teach human values to artificial agents. In *Workshops at the 30th AAAI Conference*.

Richard S. Sutton and Andrew G. Barto. 1998. *Introduction to Reinforcement Learning*. MIT Press, 1st edition.