Deep Dungeons and Dragons: Learning Character-Action Interactions from Role-Playing Game Transcripts

Citation for published version:

Digital Object Identifier (DOI):
10.18653/v1/N18-2111

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Peer reviewed version

Published In:
The 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies

General rights
Copyright for the publications made accessible via the Edinburgh Research Explorer is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The University of Edinburgh has made every reasonable effort to ensure that Edinburgh Research Explorer content complies with UK legislation. If you believe that the public display of this file breaches copyright please contact openaccess@ed.ac.uk providing details, and we will remove access to the work immediately and investigate your claim.
**Abstract**

An essential aspect to understanding narratives is to grasp the interaction between characters in a story and the actions they take. We examine whether computational models can capture this interaction, when both character attributes and actions are expressed as complex natural language descriptions. We propose role-playing games as a testbed for this problem, and introduce a large corpus of game transcripts collected from online discussion forums. Using neural language models which combine character and action descriptions from these stories, we show that we can learn the latent ties. Action sequences are better predicted when the character performing the action is also taken into account, and vice versa for character attributes.

1 Introduction

Imagine a giant, a dwarf, and a fairy in a combat situation. We would expect them to act differently, and conversely, if we are told of even a few actions taken by a character in a story, we naturally start to draw inferences about that character’s personality. Communicating narrative is a fundamental task of natural language, and understanding narrative requires modelling the interaction between events and characters.2

In this paper, we propose that collaboratively-told stories that arise in certain types of games provide a natural test bed for the problem of inferring interactions between characters and actions in narratives. We present a corpus of role-playing game (RPG) transcripts where characters and action sequences are described with complex natural language texts. Table 1 shows an example character description, and an action text for the same character. This example shows how the ties between characters and their actions are subtly present in the text descriptions, and learning the latent ties between them is a difficult task. Based on our corpus, and using neural language models, this work demonstrates an initial success on this problem.

The ability to understand and generate narratives is a useful skill for natural language systems, for example, to plan a coherent answer to a question, or to generate a summary of a document. Prior work on narrative processing has focused on inducing disjoint sets of character and event types (as topic models), capturing the relationship between characters in the same story, or extracting character-action pairs as low level noun-verb tuples. However, these models do not aim to match or infer characters and actions from each other.

We make two contributions towards closing this gap. We introduce a corpus of thousands of RPG transcripts, and show that neural language models can learn the interactions between characters and actions in narratives.

---

1http://groups.inf.ed.ac.uk/cup/ddd/

2In this paper, the word character will always be used in the sense of “character in a story” rather than the sense of “character in a token”.

**Table 1**: Example descriptions from our RPG corpus.

---
transcripts and demonstrate predictive cues between characters and actions by building neural language models with facility for adding side information. We show that a language model over action text obtains lower perplexity when we also make available a representation of the character who produced each token. Likewise, a language model for character descriptions benefits from information about the actions the character made. Our findings open up new possibilities for making sophisticated inference over narrative texts.

2 Related work

In work on narratives, both characters and actions have received significant attention, albeit separately. There is work on inducing types of characters (Bamman et al., 2013, 2014) or relationships between characters (Chang et al., 2009; Elson et al., 2010; Chaturvedi et al., 2016; Iyyer et al., 2016). Often these approaches are based on probabilistic topic models or more recently distributed word representations computed by neural networks. Others focus on learning regular and repetitive event sequences in stories (Chambers and Jurafsky, 2009; McIntyre and Lapata, 2009), together with some information about the agent of the actions. These extractions are fairly low-level, in the form of noun-verb pairs. There are also models for clustering stories either based on their characters (Frermann and Szarvas, 2017), or sentiment and topic (Elsner, 2012, 2015).

The above approaches mine types of actions or characters. This work focuses on inferring the latent ties between actions and characters, and whether one aspect can help predict the other. Flekova and Gurevych (2015) present recent work related to this latter idea. They classify characters based on their speech and actions into an introvert or extrovert class. In contrast, we focus on attributes of characters and actions beyond such coarse traits, and when these attributes are expressed as complex descriptions.

3 A corpus of RPG transcripts

Traditionally, RPGs are played orally with players seated around a table. But there are also online forums where users play RPGs by posting text descriptions instead.

We collected a corpus of RPG threads from one such website roleplayerguild.com. Here each game play is recorded in two threads. In one of these, each player posts a detailed text description of the role (character) she is going to play in the game, which we call a character description. This description includes the character’s physical appearance, personality, family background, as well as special and supernatural powers, and possessions. A second thread consists of the actual game play where each player contributes a post when his turn comes. Each post describes how the character that is assumed by that specific player responds to the game situation. Thus the story develops collaboratively. We call each post in the story thread an action description. An example from our corpus of a character description and an action description is shown in Table 1.

A noteworthy aspect of these RPGs is that character attributes are determined by writing the descriptions before the game starts. The story thread itself then focuses predominantly on the actions and does not reiterate character attributes. Moreover, we know unambiguously which character is associated with each action post. Such mapped pairs of clean character descriptions and associated actions would be difficult to obtain from novels or other stories without sophisticated analysis.

Our corpus contains 1,544 RPGs spanning a variety of themes—fantasy, apocalyptic, romance, anime, military, horror, and adventure. There are a total of 56,576 posts, comprising of 25.3M tokens. The maximum number of posts in a story is 753, minimum 2, and the average is 26. Note that many stories are in progress and some are long running. There are 9,771 unique characters in the corpus, and their descriptions amount to 8.5M tokens. There is a minimum of 1, average 6, and maximum 24 characters in a single story.

Even though each character or action description focuses on a single character, it nevertheless contains descriptions of background settings of the scene, and interactions of other characters (eg. descriptions of the parents of a character). Hence we preprocess the texts to only retain parts most related to the character in focus. To this end, in character descriptions, we only keep those sentences which mention the character’s name or the personal pronouns ‘he’ or ‘she’. The use of pronouns reflects an intuition that since the description is of one key character, the pronoun is most likely to refer to this salient entity. We also take sentences which mention personality describing words such as ‘personality’, ‘skill’, ‘specialize’,
'ability', 'profile', 'talent', etc., even when they do not contain the name. For action descriptions, we only keep sentences which start with the character's name. We do not use pronouns since action text may refer to other salient characters as well.

Finally, we replace the main character (contributor) of a post with an “ENT” (for entity) token. Other proper names in a post are replaced with a “NAME” token and numbers with a “NUM” token. We drop all punctuation and any text with “NAME” token and numbers with a “NUM” to-

Other proper names in a post are replaced with a character's name. We do not use pronouns since action descriptions as side information. For each token \( x \), the side information is the character description indicated by \( z \), i.e., \( C_z \). We follow the approach by Mikolov and Zweig (2012), and Hoang et al. (2016), where a feature embedding vector \( e \) representing side information is input to both the RNN's hidden and output layers, or to one of them. During development, we found that concatenating the feature embedding with the token embedding at the input layer, and with the hidden state at output layer gave the best performance. More formally, ACTION-LMS computes:

\[
\begin{align*}
    h_i &= LSTM(h_{i-1}, x_i) \\
    P(x_i|x_1 \ldots x_{i-1}) &= softmax(W_{hv}h_i + b_v)
\end{align*}
\]

where \( e_i \) is a representation of the character which produced the token \( x_i \). The hidden state \( h_{i-1} \) now summarizes both the action tokens up to \( i - 2 \) and the character information up to \( i - 1 \). The output layer weight matrix is \( W_{hv} \in \mathbb{R}^{|V|\times|h|} \) where \( |h| \) is the size of the RNN hidden unit, and \( |e| \) the feature embedding size.

In our work, the feature embedding itself comes from a feedforward neural network trained jointly within the LM. This feature network takes as input the average value of pretrained embeddings\(^3\) for the tokens in the character description (we remove stopwords\(^4\)). This initial vector is passed through hidden layers to yield the feature embedding \( e \) (reminiscent of deep averaging networks by Iyyer et al. (2015)).

The language models for character descriptions are similar in structure. First, we call the unconditioned model \( P(C_i) \) for a character description \( C_i \) as CHAR-LM; this is again an LSTM language model. Second, we implement CHAR-LMS

\(^3\) 300 dimension word2vec (Mikolov et al., 2013) embeddings trained on the 1 billion word Google News Corpus.

\(^4\) We remove stopwords for side information only.
which estimates $P(C_i | X_{C_i})$, where $X_{C_i}$ is the subsequence of $X$ only containing the tokens produced by $C_i$. We obtain this conditional probability based on the same architecture as ACTION-LMS. Here the input to the feature neural network is the average pretrained embeddings of the tokens (without stopwords) in $X_{C_i}$.

4.2 Experiments

We randomly divide our corpus into 100 stories for testing, 20 for development, and the rest, 1319 for training. We compare the two ACTION language models, based on a vocabulary size of 20,000, and the CHAR models have a vocabulary of 10,000.

Some posts are long even after our filtering steps, and create a winding story line when concatenated. So we also explore whether limits on description lengths is useful. In ACTION models, a limit of $g$ means that only the first $g$ words of each post are concatenated to form $X$. For CHAR models, only the first $g$ words of the description $C_i$ is used as the sequence for the LM. The same limit $g$ is given to both the models with and without side information. When using side information, we can restrict the conditioning text as well, to a maximum of $h$ words. We tune these limit parameters, as well as the number of hidden layers, hidden unit sizes and dropout probability on a development set.

For the ACTION models, we set $g$ to 100 words. ACTION-LMS uses 2 layers with 256 hidden units each. ACTION-LMS has 1 layer with 256 hidden units for the feature network with $h$ set to 25 words, and 1 layer with 50 units for the RNN part. For the CHAR models, $g = 200$ words. CHAR-LMS has one hidden layer with 100 units. For CHAR-LMS, the best network was the same as ACTION-LMS but with $h = 100$ (the first 100 words of all the action posts by that character are combined as the side information). We apply a dropout probability of 0.65, clip gradients at 5.0, and use the Adam algorithm (Kingma and Ba, 2015) for optimization. All our models can trained in an hour. ACTION-LM with 14 epochs, CHAR-LM 62, ACTION-LMS 60 and CHAR-LMS 91 epochs. We implemented the models in TensorFlow\footnote{https://www.tensorflow.org}.

4.3 Results

First, we provide examples of the patterns captured by ACTION-LMS and CHAR-LMS by sampling from the models (Table 2). For side information, we use simple words (taken from the descriptions in our test corpus) for closer examination.

For ACTION-LMS, we seed the story line with the priming text “(bos) ENT called…”, where ENT is the token in our vocabulary referring to the main character of a post. (bos) is a beginning of sentence marker. Different inputs for the conditioning character description are shown under “Char. context”. We then sample from the LM following a greedy approach taking the most likely token at each step until either the end token (eos) or a maximum of 12 tokens is reached. The sample is shown under “generated continuation”. Similarly, we sample from CHAR-LMS where the sequence is first primed with “ENT is a”. We find that both models capture interesting ties between character attributes and actions. However, there is much scope for improved models of generation.

In this work, we have focused on the possibility of capturing the interactions. For that, we compare the impact of side information using perplexity on held-out data (Table 3). For both charac-

<table>
<thead>
<tr>
<th>Model</th>
<th>Train</th>
<th>Dev</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTION-LM</td>
<td>82.56</td>
<td>106.83</td>
<td>105.06</td>
</tr>
<tr>
<td>ACTION-LMS</td>
<td>57.38</td>
<td>94.95</td>
<td>96.91</td>
</tr>
<tr>
<td>CHAR-LM</td>
<td>59.45</td>
<td>118.78</td>
<td>108.12</td>
</tr>
<tr>
<td>CHAR-LMS</td>
<td>61.84</td>
<td>110.13</td>
<td>100.86</td>
</tr>
</tbody>
</table>

Table 2: Samples from our language models

Table 3: Perplexities of our models
ter and action LMs, adding side information leads to a significant decrease in perplexity showing that the interdependence between the two aspects can be learned computationally. Again, there is a lot of scope for improving the language models given that the development and test perplexities are much higher than those during training.

5 Conclusions

We have proposed and demonstrated the feasibility of capturing interactions between characters and their actions in stories. While our neural models show that the data can be better modeled by combining both aspects, one might eventually want to infer a missing modality by sampling or generation from the model. We plan to work on these improvements for future work, and also explore evaluation methods which go beyond language model perplexities, and capture model aspects closer to the task and domain.

References


