Creative Contextual Dialog Adaptation in an Open World RPG

Mika Hämäläinen
mika.hamalainen@helsinki.fi
Department of Digital Humanities, University of Helsinki
Helsinki, Finland

Khalid Alnajjar
khalid.alnajjar@helsinki.fi
Department of Computer Science, University of Helsinki
Helsinki, Finland

ABSTRACT

Role playing games rely typically on hand-written dialog that has no flexibility in adapting to the game state such as the level of the player. This is an even bigger problem for open world RPGs that make it possible to complete the game quests and objectives virtually in any given order. We present a computationally creative method for adapting Fallout 4 dialog to the changes in the game state using word embeddings for semantics and a BRNN for sequence-to-sequence paraphrasing of syntax.

CCS CONCEPTS

• **Computing methodologies** → **Discourse, dialogue and pragmatics**; **Natural language generation**; *Natural language processing*; • **Applied computing** → Computer games.

KEYWORDS

dialog generation, contextual adaptation, video game dialog, computational creativity

ACM Reference Format:
Mika Hämäläinen and Khalid Alnajjar. 2019. Creative Contextual Dialog Adaptation in an Open World RPG. In *The Fourteenth International Conference on the Foundations of Digital Games (FDG '19), August 26–30, 2019, San Luis Obispo, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3337722.3341865

1 INTRODUCTION

A great many open world role-playing games (RPGs) offer the player a seemingly endless world to discover. The player can move freely in the world and complete quests in almost any given order. Fallout 4[1], the focus of our paper, is an example of a game of this nature. However open the game world, what makes it less immersing is the fact that all the dialog in the game has been scripted and voice-acted beforehand.

We propose a computationally creative mod for Fallout 4, the purpose of which is to take existing in-game dialog and adapt it contextually to the different variables indicating the state of the player, such as health, charisma, the number of quests completed or monsters killed. We do this by adapting the semantics of existing dialog to one of these aforementioned variables to convey the dialog in the mood set by the player state and by paraphrasing the syntax to provide more variety to the dialog.

In video game research, producing parts of the game automatically, whether they are playable levels, sprites or dialog, is usually inspected thorough the notion of procedural content generation (PCG). However, our framework comes from computational creativity (CC), which puts an emphasis on the *creativity* as opposed to *generation*[2].

The common list of desirable properties in PCG include speed, reliability, controllability, expressivity and diversity, and creativity and believability [25]. An abstract level theory of CC explaining creativity through a so called creative tripod [5], specifies three requirements for a creative system: skill, appreciation and imagination.

Appreciation of the tripod is most closely related to the *reliability* in PCG. Reliability states that the generated content should satisfy some criteria on quality. Appreciation, on the other hand, refers to the capability of the system to critically evaluate its own creations and those of others.

Imagination and *expressivity and diversity* are closely related to each other as they both refer to the capability of the system to produce novel output. A system repeating the same output with minor changes is hardly creative.

Skill relates to *creativity and believability* in PCG. Skill means that the system is simply capable of producing a creative artifact as an output, whether it is of any value is up to *appreciation* to decide. In PCG terms, creativity and believability are about the generated output not looking like it was generated by a computer, which can be achieved by skill.

Speed and *controllability* are not requirements for creativity in the creative tripod framework, although they are needed in our case. Dialog has to be generated in real time, and therefore speed is essential. The context in which the dialog occurs has to have some control over the generated dialog as it has to adapt to the context.

2 RELATED WORK

Dialogue generation in the context of digital media has been studied from the point of view of interactive storytelling (IS) [3]. In their approach, they generate suitable dialog in between two characters in an automatically generated situation. The generation is done with multiple rule-based steps starting from determining the modality of the expression, generating the semantics, finding the words in a lexicon and finally selection based on onomatopoeia. Their approach relies on predefined rules offering no adaptability without altering the existing rules.

[1] Fallout is a registered trademark of Bethesda Softworks LLC in the U.S. and/or other countries. All Rights Reserved.

[2] See [26] for an extended discussion on the concept of *mere generation* in CC

FDG '19, August 26–30, 2019, San Luis Obispo, CA, USA

ACM ISBN 978-1-4503-7217-6/19/08…$15.00
https://doi.org/10.1145/3337722.3341865

Altering dialog to produce more variety in story-based games has been studied by [14]. Their approach does not alter any existing dialog in the game, but rather they suggest dialog to be written in an abstract level of intents such as greetings. An abstract intent is divided into 9 contextual bins holding pieces of ready-made dialog. During the game, an appropriate piece of dialog is selected from one of the bins that matches the context. They train a classifier to place existing dialog into the bins automatically. Their approach does not generate anything novel on its own, but rather provides a way of showing different pieces of existing dialog in different contexts. This requires that the original game has its dialog written according to their abstraction.

In [16], Lessard *et al.* have described their work on generating NPC dialogues that are suitable in changing game-state conditions. Their system is called *Hammurabi* and it is an experimental political management game. The authors have used an existing tool for generating metadata-driven context-free grammars, called Expressionist [21]. By incorporating the tool, the authors generated context-free grammars where some replacements could by filled/extended by game authors while leaving out some replacements to be filled during run-time based on a context (e.g. the personality of an NPC).

Morrison and Martens [19] have proposed a model of group conversation where multiple NPCs, along with the player, participate in a conversation. In the model, the player and NPCs can take the same conversational moves. The generation component considers a multitude of features, such as the current topic, character personality and character emotions. An essential goal of their work is to simulate such group conversations and observe any effects caused by the features. Their results indicate that s character's personality influences on how often it speaks and that a character would change its belief when grouped with characters of different believes.

In terms of creative generation of game-related natural language, Fan *et al.* [7] have build a hierarchical neural network for generating stories. The model is trained to produce contexts (a list of words) and stories based on these contexts. Such model could be employed in a game to generate dialogues instead of stories; however, obtaining sufficient training data for such a task is challenging. Other game-related approaches to creative narrative generation is discussed by Liapis *et al.* in [2].

3 DIALOG ADAPTATION

This section is dedicated to describing the creative dialog adaptation. First and foremost, we describe how the dialog generator integrates to Fallout 4, and then we describe the two methods for altering the existing dialog. The input for the adaptation is two utterances of existing in-game dialog - one to be uttered by the player, and the other by an NPC - and the game state variable according to which the adaptation is conducted.

The dialog is changed one utterance at a time, first by paraphrasing its syntax and then by adapting its lexicon to semantically relate to the change in the game state. This method always produces 5 candidate dialogues, out of which the top ranking one is picked to be shown to the player.

3.1 Integration to Fallout 4

In order to interact with the player in the video game, our code needs to be able to communicate with Fallout 4 to gather information about the game state and display the generated dialog. For this purpose, we have developed a Fallout 4 mod.

Our mod adds an NPC (non-player character) capable of creative dialog in *Diamond City*. When the player approaches to the NPC and initiates a dialog, a Papyrus script attached to the NPC writes a debug log output indicating that the player is in dialog with the NPC together with information about the game state. This information consists of 8 values of the player object (such as health, agility and stamina) and 104 values of the game stats (such as quests completed, people killed, locations discovered and so on). The increase in these values is used for generating the adapted dialog.

Because Fallout 4 mods do not support communication outside of the game itself, we read and parse the debug log for the data originating from our mod with a constantly running background script external to the game. When the script reads the new game state, it passes it on to the generative Python script to do the actual dialog generation.

The generated dialog cannot be shown in the game itself, as Fallout 4 only supports dialog that has been scripted beforehand. Therefore, we have developed an overlay application that stays hidden until the dialog has started. Once the new dialog is generated, it will be shown on top of the game window. The overlay application changes the shape of its window to only show the generated dialog making it look visually as though the text was shown by Fallout 4 itself. The overlay application listens to joy-pad button presses to keep itself in sync with the dialog state in Fallout 4, requesting for more dialog to be generated as needed.

As a result of this complicated mod consisting of external Python scripts and an external overlay application, we can read the game state in the beginning of the dialog and we can successfully show the externally generated dialog as though it was a part of the game.

3.2 Dialog data

In order to adapt existing dialog, it is important to get access to the in-game dialog to begin with. Fortunately, Fallout 4 Fandom [3] has listed all the text in the Fallout 4 dialogues together with meta-data, such as scene and topic id and preceding dialog.

We clean the additional annotation from these dialogues leaving only the utterances that appear in the game. Furthermore, we filter out all the utterances that consist of less than four tokens. This will leave us with the utterances with enough substance to do any meaningful adaptation. The final dialog corpus consists of 74,943 utterances.

3.3 Syntactic change

In order to provide more variety in the dialog, we paraphrase its syntax by training a sequence-to-sequence NMT (neural machine translation) model with OpenNMT [15]. The only difference from the OpenNMT defaults is that instead of an RNN (recurrent neural network) based encoder, we use BRNN (bidirectional recurrent neural network) to better capture the context of each token. In addition, we use the copy attention mechanism, which allows for

[3] https://fallout.fandom.com/wiki/Category:Fallout_4_dialogue_files

transferring unknown tokens from the source to the target. While this might lead into code switching in a bilingual translation task, it is useful when the translation is done within one language.

While using sequence-to-sequence models in paraphrasing as an end-to-end solution has received research attention in the past [9, 17], we are not interested in paraphrasing directly on the level of text, but rather on the more abstract level of syntax. There are two reasons for this, firstly an NMT model will output an N amount of best candidates for a given input, which is not very *imaginative* as the output stays the same, secondly in the absence of parallel data tagged with Fallout 4 game states, it is not possible to train an NMT model to solve the problem we are dealing with. However, what we achieve by paraphrasing the syntax is that we can have more structural variety in the final output as it does not have to follow the exact structure of the original utterance,

To train the NMT model, a parallel corpus of syntactically equivalent English sentences is needed. For this, we use the Opusparcus [6] corpus, which consists of aligned English sentences from different subtitles for the same movies. The train set for English is quite extensive, consisting of over 40M parallel sentences, even too much so for our needs. We take the first 100k sentence pairs from the train set that satisfy the criterion of their syntax being different.

We turn the parallel sentences from Opusparcus in parallelized syntactic structures by parsing the sentences with Spacy[4] [12]. We replace the words in the sentences with a placeholder if they are part of an open class part-of-speech (nouns, adjectives, verbs, adverbs, proper nouns and numerals) this placeholder includes the part-of-speech and the morphological reading of the word. For words that belong to a closed class part-of-speech or are auxiliary verbs as indicated by *aux* dependency, they are left in the sentence. This makes it possible for the model to learn to paraphrase in a more abstract syntactic level, without losing the meaning bearing information of closed class words such as articles and prepositions.

With the syntactic parallel data, the model is trained for the default 100,000 training steps. The original dialog in Fallout 4 is also converted in the abstract syntax format, and translated to 5 best paraphrased syntactic structures with the NMT model.

The problem that arises from the corpus is that the NMT model learns to truncate the syntactic structures. This is due to the fact that subtitles are shorter than the maximum length of an utterance in Fallout 4, and the model has only learned to predict short structures on the target side. However, as proposed by [22], a better target representation can be included into the model by back translation.

We train the exact same NMT model as before with the difference of flipping the target and source sides, and use this model to translate from the syntactic structures in Fallout 4 utterances to their paraphrases. With this synthetic data, we continue to train our initial model using the synthetically generated paraphrases as the source and the actual dialog structures as the target side, for an additional 100,000 steps. This way, the model will learn longer target side representations and will be capable of producing longer target sentences reducing the problem of truncation.

The improvement of the back-translation approach can be seen in Figure 1. The figure shows the average length of the predicted sentences in relation to the original length in the top 5 output

[4]The model used was en_core_web_md

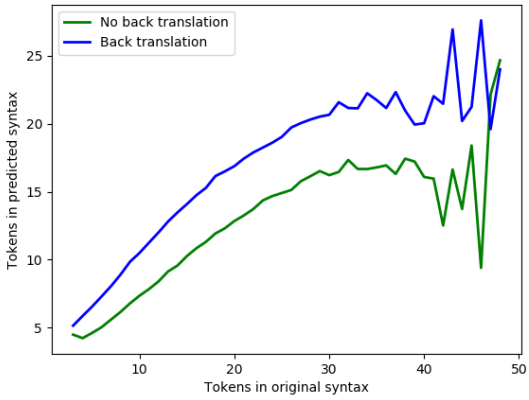

Figure 1: Average length of the predicted sentences in contrast to the original sentence

for each sentence as predicted by the model before and after the training with the synthetic back-translated data. The results are shown for frequent lengths in the dialogues, meaning that the length has occurred over 10 times in the corpus. Table 1 illustrates how the abstract syntax and its paraphrase look like.

After the paraphrasing with the final model, the placeholders are replaced with the actual words from the original dialog. At this point, no changes have been done to words, but rather to the structure of the utterances. When a new structure is requested to an existing utterance, the system will pick at random out of the top 5 paraphrases output by the NMT model.

3.4 Semantic change

The semantic change process is in charge of substituting words in the paraphrased utterances with new words that are semantically related to the desired context as indicated by the game state. The new words should fit semantically in the utterance to be altered as well. The process begins by accepting as input a syntactically paraphrased utterance s, its syntax ξ and the desired context ω to which we would like to adapt the utterance. In our case, the context is a list of words (i.e. $\omega = \{w_0, w_1, w_2, \dots w_i\}$, where $i > 0$) that relates to a changed variable in the game state. A mapping between the game state variables and four contextual keywords is done manually. An example of such a mapping for the game sate variable *Days passed* is *old, time, longevity* and *mature*.

Once the process is provided with the required parameters, it performs the semantic change in the following steps:

(1) Find and select words in the utterance s to change.
(2) Get possible candidate words to replace existing ones.
(3) Search for candidate words that are suitable for the intended context.
(4) Perform the semantic change.

In the remainder of this section we elucidate these steps.

The first step, that is finding and selecting words to replace, looks at the part-of-speech tags of words provided in the syntax ξ and highlights words A that have certain parts-of-speech. We define

Original utterance	Original syntax	New syntax
Another one of his stogies.	*Another+DET NUM+CD of+ADP ADJ+PRP$ NOUN+NNS .+PUNCT*	*Another+DET ADJ+JJ of+ADP ADJ+PRP$ NOUN+NNS .+PUNCT*
I really mean it, I owe you one.	*I+PRON ADV+RB VERB+VBP it+PRON ,+PUNCT I+PRON VERB+VBP you+PRON NUM+CD .+PUNCT*	*I+PRON ADV+RB do+VERB ADV+RB VERB+VB it+PRON ,+PUNCT I+PRON VERB+VBP you+PRON NUM+CD .+PUNCT*

Table 1: Example of the syntactic paraphrasing

these parts-of-speech to be verbs, nouns, proper nouns, adjectives and numbers. Thereafter, a random subset B of the highlighted words A is selected, such that $B \subseteq A$ and $|B| > 0$, to be replaced by the process. As an example, let s be "You sure you want to be stuck with this ugly mug ?", the words matching the defined parts-of-speech are *sure, want, be, stuck, ugly* and *mug*. Out of these, the process selected to change four words only to be replaced, which are *sure, be, stuck* and *mug*.

The next step is finding semantically similar words to the selected words B. We consider words that are semantically similar to the original words as we want the new replacements to fit into the original utterance by a similar semantic meaning. To obtain the semantically similar words, we use a pre-trained (on common crawls) FastText model [18]. For each word to be replaced in B, we retrieve the k most similar words to it. We empirically set k to 300 as it was not very restrictive and it covers a wide range of similar words with the least amount of perceivable noise. Examples of retrieved words for words in B, following the earlier example, are:

sure: {afraid, glad, sorry, aware, clear, confident, …, conclusive, relevant, pretend}
be: {have, remain, seem, are, appear, is, …, depend, partially, sort}
stuck: {sticking, jammed, wedged, trapped, screwed, sucked, stick, …, screwing, poking, Sticking, positioned}
mug: {mugs, Mug, cup, coffee, pint, cups, tankard, …, wineglass, handgun, lunch, 568ml}

In the retrieved words that are semantically similar, we search for suitable words to be used based on the following criteria: 1) the word is not identical to the original word 2) it has a part-of-speech that matches the original word, 3) it is relatively similar to the original word and 4) it is strongly similar to the intended context. In case the original word is a placeholder (i.e. a completely new word added by the paraphrasing model), we only consider the second and fourth criteria.

In some cases, the semantic model might return the same word but in a different form (e.g. capitalized). To prevent replacing a word with itself, we prune out any candidate words that match the original word, while having both converted into lower case.

The second criteria is enforced to filter out candidate words that do not have a part-of-speech matching the original word, as using such words would yield grammatically incorrect utterances. To get the possible parts-of-speech a candidate word could have, we used the part-of-speech tagger provided in Spacy.

As some of the retrieved semantically similar words might not be semantically similar in reality (as we are selecting the top k words without inspecting their similarity score), we introduce a criteria that dynamically calculates a minimum threshold of semantic similarity score to the original word that all candidate words must obey. The motivation for calculating the threshold dynamically instead of using a fixed threshold (e.g. 0.5) is that the distribution of the semantic similarity scores of the k most similar words to $b, b \in B$ is not fixed for all words in B. We define this threshold as the median of the semantic similarity scores of the retrieved k most similar words, for each word b.

Similarity to the third criteria, the last criteria calculates a dynamic threshold but based on the semantic similarity to the context ω. As the context is defined by a list of words (i.e. $\{w_0, w_1, \ldots, w_i\}$), we represent the context as the centroid of these words as follows:

$$centroid = \frac{\sum_{w \in \omega} v(w)}{|\omega|} \tag{1}$$

Where v is the vector representation of the passed word in the semantic model. Once we have a vector representation of the context, we measure the semantic similarity of candidate words for a word b and the centroid by calculating the dot product. In this criteria, instead of using the median, we define the dynamic threshold to be the 75th percentile of the semantic similarity scores to the context. The threshold is higher here as we want the new word to be strongly similar to the desired context, yet similar to the meaning of the original word.

By the end of the search step, only apt candidate words are kept. Following our earlier example, below are random examples of retained candidate words:

sure: {smart, familiar, hopeful, obvious, proud, …, quick, confident, necessary}
be: {look, ought, require, being, prove, need, …, make, keep, prefer}
stuck: {muddled, clinging, left, ended, staying, messed, putting, …, stayed, dragging, lacking, broke, perplexed}
mug: {girlfriend, smirk, pistol, photograph, mugs, …, photo, bum, plastic, tea-bag, pewter}

Out of the retained candidate words, a random word is chosen to substitute the existing one. When the new semantically changed words are used to replace the existing words in the utterance, their morphology is matched to that of the original word. This inflection step is done by using a Python library called Pattern [23]. As a result of this process, the example input "You sure you want to be stuck with this ugly mug ?" gets transformed into "You smart you want to look muddled with this ugly girlfriend ?".

3.5 Ranking the output

We pick two utterances following each other at random from the corpus. The first one will be considered a player utterance and the second an NPC utterance. We use the syntactic change followed by semantic change to produce 10 different small candidate dialogues to be presented to the player. As some of the dialogues are bound to be better than the others, we need the system to show *appreciation* to its generated output and to rank the output based on how good they are. For this purpose, we use a language model.

We train a 3-gram language model by using the entire Opusparcus on KenLM [10]. The tool uses a modified Kneser-Ney smoothing [11] to estimate the probability of an input sentence according to the language model. The output dialogues are ranked based on the language model and the top candidate is picked to be displayed to the player.

3.6 The Flow of the Dialog

When the player has initiated a dialog with the creative NPC, the first thing that is presented to the player are the dialog options. In dialog, Fallout 4 shows typically 4 options that can be chosen from to advance in the dialog sequence. In our case, we pick 3 options out of the game state variables that have increased since they were last discussed with the NPC, leaving the fourth option as a dialog ending *bye* option. A screen-shot of this dialog state can be seen in Figure 2.

Figure 2: Dialog options

The player can pick the game state variable according to which he desires the existing dialog to be altered. After the choice, the adapted dialog is shown (see Figure 3). First the player character will utter his adapted line, then the NPC. At the end of this sequence, the dialog returns back to the state in which the player can pick from the game state variables or exit the dialog.

As our method is a computationally heavy process taking dozens of seconds to produce the final output, we cannot use it for real time generation. Therefore, we are continuously generating dialog into a cache in the background, so that there will always be generated dialog available in real time.

4 EVALUATION

With computational creativity, just producing novel output is not enough, but it has to be evaluated. The difficulty of evaluation of

Figure 3: Generated dialog shown in the game

creative output in the field of CC has been pointed out in a timely manner, see [1, 13]. Therefore, we use two different methods to evaluate the generated dialog: internal and external. The internal one is based on existing research in automatically evaluating paraphrase generation and the external one is based on using human annotators to judge the output based on a set of evaluation questions.

We produce 50 utterances of dialog with randomized game state variable with our method described in the previous section. These 50 utterances will be evaluated with the two different evaluation approaches. Examples of the generated output can be seen in Table 2.

4.1 Internal evaluation

For internal evaluation, we are following the evaluation metrics already established for evaluating paraphrase generation automatically [24]. They base the evaluation on two notions: a paraphrased text should be as close to the original as possible, while at the same time, it should be as different as possible. Closeness to the original indicates similarity in meaning and deviation novelty.

Similarity is calculated with BLEU scores [20], which is a widely used metric in machine translation for assessing how closely the output of a system matches a gold standard translation. PINC [4] is used to indicate the deviation of the output from a gold standard. In other words, the higher the BLEU, the closer the paraphrase is to the original, and the higher the PINC, the more different they are.

In our case, we count the 4-gram BLEU and unigram PINC scores comparing the final output to the original utterance. These results shown in Table 3 are the average scores for the evaluation set. As BLEU indicates, there is still similarity to the original sentence, which can be seen as an indication of a similar meaning. The PINC score shows that there is difference between the input and the output, indicating that the output is clearly different and that the changes are not just minor ones. We will further validate the results in the next section with human judges.

4.2 External evaluation

In order to have a better understanding on what the BLEU and PINC scores actually indicate, we conduct a quantitative evaluation

Original	Altered	Game state	Seed words
That's the old age talkin' again I suppose.	That is again the old age fightin' I suppose.	strength	strong, powerful, bulky, muscular
They exploited technology for their own gains, pocketing the cash and ignoring the damage they'd done.	They cultivated technology for their extensive claims, pocketing the cash and ignoring the damage they'd.	endurance	persistent, sturdy, survivor, enduring
You're done putting people through your little tests.	You be documented establishing civilians through vague checks.	perception	eye, sense, vision, accuracy
Well, then you know just how I feel.	Well, then you anticipate just how I think.	luck	lucky, luck, chance, fortunate
Uh, yeah. I'll try and dial back the friendship from here on out.	Uh, I'll fix and switch back the generosity from here out on.	endurance	persistent, sturdy, survivor, enduring

Table 2: Examples of the altered titles

BLEU	PINC
0.11	*0.68*

Table 3: BLEU and PINC scores

on the generated output. The evaluation was done on an online crowd-sourcing platform called Figure Eight[5]. For each generated utterance in the evaluation set, we show the original utterance, the generated one and the seed words used to adapt the semantics of the utterance. Every generated utterance was evaluated by 35 human judges according to the following evaluation questions. The judges were not told they are evaluating computer produced utterances.

(1) How different are the two sentences?
(2) How close are the meanings of the two sentences?
(3) How closely do the words relate to the sentence 1?

The first question is meant to evaluate the same aspect as PINC, indicating the deviation of the output from the input. If the difference is high, the adaptation has done enough changes to the original input. The second question is supposed to evaluate the same aspect as the BLEU score. If the meanings are close, the adaptation will not have changed the meaning entirely. The third question evaluates whether the meaning of the adapted sentence relates to the seed words used for adapting the utterance to the context.

The judges were asked to answer to the questions on a 5-point Likert scale, 1 being the furthest away from what is asked (e.g. "very similar" for Q1) and 5 agreeing most with the question (e.g. "very different" for Q1). In other words, the higher the scores, the better the quality of the output.

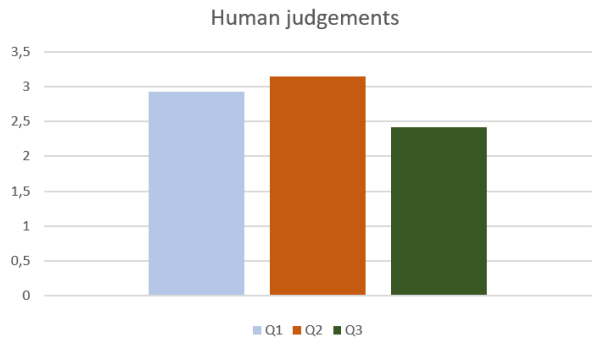

Figure 4: Average scores for the evaluation questions

[5]https://figure-eight.com/

The results of the evaluation are shown in Figure 4. The results show that despite of being previously used for the evaluation task PINC and BLEU scores reflect poorly the human judgment (see [8] for a similar observation). However, our approach can produce on the average utterances that are different from the original and yet share a meaning similarity, which is good for adding more variety on the existing dialog. On the other hand, the approach could improve on expressing the meaning of the seed words in the final output.

5 CONCLUSIONS AND FUTURE WORK

We have presented our work on creative contextual adaptation of Fallout 4 dialog. Our model can produce novel utterances out of the existing ones that share similarity in meaning with the original utterances. However, more research is needed in ensuring that the semantics imposed by the new context are better conveyed in the generated utterances.

Furthermore, we have developed a mod that allows the integration of our generative model with the game itself. The code of the mod has been released under GPL licence as open source on Zenodo[6].

In the future, it would be interesting to integrate our approach better with the game. Now we have one NPC character with whom the player can have dialog that is randomly picked from all the existing dialog in the game. Instead of one NPC, it would be more meaningful to integrate the dialog adaptation to the existing characters in the game contextually adapting the dialog they have been programmed to say. This would open up the possibility of answering to several of interesting playability related questions. Will the adapted dialog mask some of the key parts of the dialog making progression on quests difficult? How to ensure the key parts are still conveyed? Also this would make it possible to widen the context we are using for adapting the dialog to different locations as well.

Exploring the different possibilities for conceptual adaptation is needed in the future. As the results of this paper suggest, using a seed word based approach doesn't, with our methods, yield results of a high semantically relatedness to the context. Perhaps, the context should be modelled more thoroughly with full sentences or bigger text corpora to have a better semantic representation based on which the adaptation can be conducted.

REFERENCES

[1] Khalid Alnajjar and Mika Hämäläinen. 2018. A Master-Apprentice Approach to Automatic Creation of Culturally Satirical Movie Titles. In *Proceedings of the 11th International Conference on Natural Language Generation (INLG)*. 274––283.

[6]https://doi.org/10.5281/zenodo.3232863

[2] Georgios N. Yannakakis Antonios Liapis and Julian Togelius. 2014. Computational Game Creativity. In *Proceedings of the Fifth International Conference on Computational Creativity (ICCC2014)*. International Association for Computational Creativity, 46–53.

[3] Marc Cavazza and Fred Charles. 2005. Dialogue generation in character-based interactive storytelling. In *Proceedings of the First AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*. AAAI Press, 21–26.

[4] David L Chen and William B Dolan. 2011. Collecting highly parallel data for paraphrase evaluation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. 190–200.

[5] Simon Colton. 2008. Creativity Versus the Perception of Creativity in Computational Systems.. In *AAAI spring symposium: creative intelligent systems*, Vol. 8.

[6] Mathias Creutz. 2018. Open Subtitles Paraphrase Corpus for Six Languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. European Language Resources Association (ELRA).

[7] Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical Neural Story Generation. *CoRR* abs/1805.04833 (2018). arXiv:1805.04833 http://arxiv.org/abs/1805.04833

[8] Mika Hämäläinen and Khalid Alnajjar. 2019. Modelling the Socialization of Creative Agents in a Master-Apprentice Setting: The Case of Movie Title Puns. In *Proceedings of the Tenth International Conference on Computational Creativity*. 266–273.

[9] Sadid A Hasan, Kathy Lee, Vivek Datla, Ashequl Qadir, Joey Liu, Oladimeji Farri, et al. 2016. Neural Paraphrase Generation with Stacked Residual LSTM Networks. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. 2923–2934.

[10] Kenneth Heafield. 2011. KenLM: Faster and Smaller Language Model Queries. In *Proceedings of the EMNLP 2011 Sixth Workshop on Statistical Machine Translation*. Edinburgh, Scotland, United Kingdom, 187–197. https://kheafield.com/papers/avenue/kenlm.pdf

[11] Kenneth Heafield, Ivan Pouzyrevsky, Jonathan H. Clark, and Philipp Koehn. 2013. Scalable Modified Kneser-Ney Language Model Estimation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*. Sofia, Bulgaria, 690–696. https://kheafield.com/papers/edinburgh/estimate_paper.pdf

[12] Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural Language Understanding with Bloom Embeddings, Convolutional Neural Networks and Incremental Parsing. *To appear* (2017).

[13] Anna Jordanous. 2014. Stepping back to progress forwards: Setting standards for meta-evaluation of computational creativity. (2014).

[14] Christopher Kerr and Duane Szafron. 2009. Supporting dialogue generation for story-based games. In *Proceedings of the Fifth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*. AAAI Press, 154–160.

[15] Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M. Rush. 2017. OpenNMT: Open-Source Toolkit for Neural Machine Translation. In *Proc. ACL*. https://doi.org/10.18653/v1/P17-4012

[16] Jonathan Lessard, Etienne Brunelle-Leclerc, Timothy Gottschalk, Marc-Antoine Jetté-Léger, Odile Prouveur, and Christopher Tan. 2017. Striving for Author-friendly Procedural Dialogue Generation. In *Proceedings of the 12th International Conference on the Foundations of Digital Games (FDG '17)*. ACM, New York, NY, USA, Article 67, 6 pages. https://doi.org/10.1145/3102071.3116219

[17] Jonathan Mallinson, Rico Sennrich, and Mirella Lapata. 2017. Paraphrasing revisited with neural machine translation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, Vol. 1. 881–893.

[18] Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2017. Advances in Pre-Training Distributed Word Representations. *CoRR* abs/1712.09405 (2017). arXiv:1712.09405 http://arxiv.org/abs/1712.09405

[19] Hannah Morrison and Chris Martens. 2017. A Generative Model of Group Conversation. In *Proceedings of the 12th International Conference on the Foundations of Digital Games (FDG '17)*. ACM, New York, NY, USA, Article 66, 7 pages. https://doi.org/10.1145/3102071.3116218

[20] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*. 311–318.

[21] James Ryan, Ethan Seither, Michael Mateas, and Noah Wardrip-Fruin. 2016. Expressionist: An Authoring Tool for In-Game Text Generation. In *Interactive Storytelling*, Frank Nack and Andrew S. Gordon (Eds.). Springer International Publishing, Cham, 221–233.

[22] Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Improving neural machine translation models with monolingual data. *arXiv preprint arXiv:1511.06709* (2015).

[23] Tom De Smedt and Walter Daelemans. 2012. Pattern for python. *Journal of Machine Learning Research* 13, Jun (2012), 2063–2067.

[24] Jörg Tiedemann and Yves Scherrer. 2018. Translational Grounding: Using Paraphrase Recognition and Generation to Demonstrate Semantic Abstraction Abilities of MultiLingual NMT. *arXiv preprint arXiv:1808.06826* (2018).

[25] Noor Togelius, Julian an Shaker and Mark J Nelson. 2016. Introduction. In *Procedural content generation in games*. Springer, Chapter 1, 1–15.

[26] Dan Ventura. 2016. Mere generation: Essential barometer or dated concept. In *Proceedings of the Seventh International Conference on Computational Creativity*. Sony CSL, Paris, 17–24.