# Pokérator - Unveil your inner Pokémon

**Dominique Geissler, Elisa Nguyen, Daphne Theodorakopoulos, Lorenzo Gatti**
Human Media Interaction Lab, University of Twente
Enschede, The Netherlands
{d.m.geissler,t.q.e.nguyen,d.theodorakopoulos}@student.utwente.nl, l.gatti@utwente.nl

## Abstract

The Pokérator is a generator of Pokémon names and descriptions, based on user input. The names are generated by blending words based on syllables or characters according to a bigram language model. An accompanying description is generated by filling a template with ConceptNet answers. This sentence is then used as a prompt for text generation with the GPT-2 language model which was finetuned on Pokédex entries. The evaluation of the generated Pokémon names shows that the names are not realistic, but appreciated and creative.

## Introduction

While many computational creativity systems produce "art for art's sake", ever more systems are starting to focus on creativity in applied domains. These applications range from headline generation (Alnajjar, Leppänen, and Toivonen 2019; Gatti et al. 2016) over cover art for music albums (Cruz 2019) to mnemonic devices (Bodily, Glines, and Biggs 2019). Gaming is also one of the domains where computational systems, either autonomous or in a co-creation setting, are becoming popular. In addition to prominent examples like *ANGELINA* (Cook, Colton, and Gow 2017), which can generate full games on their own, a number of creative systems are focused on helping human content creators produce assets, resources and flavour text, i.e. text that fits well with the style of the game and adds to the depth of the story, but has no practical effect on its mechanics.

In the Pokémon universe, the role-playing video games where human trainers battle each other's little "monsters", i.e. Pokémon, both the Pokémon names and the entries of the Pokédex (an encyclopedia storing knowledge and trivia about every Pokémon) are prime examples of flavour text. The Pokémon universe looks like a good application scenario for a creative generator: the monsters' names have a very distinct look and appearance, and are not random but related to the characteristics of the Pokémon themselves (Kawahara, Noto, and Kumagai 2018). Naming a new Pokémon is thus a creative task that requires both intelligence and knowledge of the domain. A creative generator for names and descriptions would be beneficial for the authors of the game.

Current Pokémon games let the user customise the name, gender and look of their playable character. Previous research shows that this type of customisation can result in higher player engagement (Ng and Lindgren 2013). With the Pokérator, customisation could go beyond the playable character and expand to the first Pokémon a player receives.

This work aims at producing a personalised Pokémon name and description, starting from user-provided concepts. For the names, the generator aims at capturing the intuition behind the names of many monsters, i.e. blending two words together (e.g. "Snorlax" is a blend of "snoring" and "relax"). The descriptions are to reflect these characteristics by describing them further. Next to implementing the Pokérator, this work aims at evaluating the quality of the output of the system.

## Related Work

The Pokérator is concerned with creative naming; *Namelette* (Özbal and Strapparava 2013) is an interactive system that tackles a similar problem: it can generate brand, company or product names. It creates neologisms from user input based on characteristics as well as phonetic similarities. Related information about the words are derived from ConceptNet (Speer, Chin, and Havasi 2017) and WordNet (Fellbaum 2010). These are blended together to create a new name. An n-gram language model (LM), trained on the words in the CMU Pronouncing Dictionary (Weide 1998) computes the phonetic likelihood of the name. *Namelette* can also perform latinisation of the name, by adding a latin suffix to the name. In many ways, *Namelette* works similar to the Pokérator as they both rely on word relations, blend words to create new names and evaluate based on n-gram LMs.

*JAPE* (Binsted and Ritchie 1994) is a program for punning in a question-answer format. It creates puns based on schemata, descriptions as well as templates and uses WordNet data to create the puns. Even though our system does not aim at creating puns, it uses a similar syllable-merging process for Pokémon name generation and relies on templates for text generation.

Like the Pokérator, *Churnalist* (van Stegeren and Theune 2019) aims at automatically creating flavour text for computer games. The system generates fictional newspaper headlines by feeding user input and related words into a headline database and replacing the subjects. Similarities are the usage of related words and templates; the Pokérator however uses GPT-2 to produce somewhat longer texts.

The Patent Claim Generator (Lee and Hsiang 2019) aims

Proceedings of the 11th International
Conference on Computational Creativity (ICCC'20)
ISBN: 978-989-54160-2-8

at contributing to the sparsely explored field of "augmented inventing", i.e. having the computer produce innovations. It generates patent claims using OpenAI's GPT-2 model (Radford et al. 2019). Similarly to our work, the large pre-trained LM GPT-2 has been adapted, in this case to the field of patent claims, to be able to generate a particular type of text.

## Method

**User Input.** In order to get the initial words for name and description generation, the user is first asked 8 "personal" questions requiring one-word answers (e.g. name, hobby, favourite animal/plant/food). The questions are intended for the user to build a relation towards their own "inner Pokémon".

**Word Creation.** The user's answers are the input for creating the new Pokémon name. To restrict the search space and keep computational costs low, the system starts by selecting two words at random. In the next step, the inputs are blended. First, the words are tokenised into syllables using the Natural Language Toolkit (NLTK) syllable tokeniser (Loper and Bird 2002). Then, a list is created by merging the first syllables of one word with the last syllables of the other word. This is done for all possible combinations. The longer word can never completely be part of the blended output as it would be too recognisable. However, the shorter word may because it could be only one syllable and thus be skipped. If both words are of equal length, they are both taken into consideration (e.g. [starfish, yellow] $\rightarrow$ [starlow, starfishlow, yelstarfish, yelfish]). In case both words only consist of one syllable, the merge is done on character level. The first letters of the first word until the first vowel are merged with the last letters of the second word starting from the first vowel. Moreover, a suffix chosen randomly from common Pokémon suffixes is added (e.g. [green, cat] $\rightarrow$ [gr-at, c-een] $\rightarrow$ [gratgon, ceenlow]).

**Name Ranking.** After generation, the system ranks the names to use the best one as Pokémon name with the help of a syllable-based and a character-based LM. Input words with more than one syllable are first split into syllables, then grouped into bigrams and evaluated with the LMs. If the original words had only one syllable, they are split into characters and grouped into bigrams for evaluation. We trained four LMs for evaluation: Two were trained on Pokémon names stemming from a dataset which contains information on the 802 existing Pokémon (Banik 2017), and two on the 133k English words contained in the CMU Pronouncing dictionary (Weide 1998). We used two sets to ensure that the generated name looks like a Pokémon but also seems like an English word. For each dataset, one LM was created on the basis of syllables, and one on the basis of characters. For each word in the datasets, we created bigrams and subsequently calculated the probability of each bigram using Naive Bayes with Laplace Smoothing. The probability of a generated word is calculated by multiplying the individual bigram probabilities. For example, P("starfishlow") = P("fish"|"star") $\times$ P("low"|"fish"). The probabilities from the Pokémon and the CMU dictionary are weighted: P("starfishlow") = 0.4 $\times$ PokéLM + 0.6 $\times$ EnglishLM. The weights were chosen based on an internal evaluation of about 20 examples. This ensures that the word is pronounceable and not completely alien from English orthography, while still considering the peculiarity of Pokémon names[1] (e.g. "Exeggcute", "Kakuna"). The generated word with the highest probability is returned as the name of the Pokémon. In the above example, this is "starlow".

**Description: Prompt for text generation.** The description of a Pokémon in the Pokédex is usually a short text of up to three sentences, describing one feature or characteristic of the Pokémon. In this work, the description is created using OpenAI's GPT-2 model and an input sentence. The input sentence is generated based on word relations and templates. One of the words that compose the generated Pokémon name is taken as an input to ConceptNet (Speer, Chin, and Havasi 2017) in order to retrieve related words. ConceptNet offers a number of related words as an answer to one query as well as so-called surface texts, i.e. sample sentences including both the input word and the output word, specifying the relationship of the words (e.g. "Something you find at [[sea]] is [[a starfish]]"). The offered related words are filled into templates. As there are different relations, we prepared multiple templates for each relation. To ensure proper grammar, the retrieved word needs to fulfil a part-of-speech (POS) expected by the template sentence, e.g. a template for the word relation "AtLocation" is *"It likes to be at <AtLocation>."*, which expects a noun. In order to ensure the correct POS of the output word, the surface text is POS tagged. From the available word relations that satisfy the described requirements, a fitting one is chosen randomly and the input sentence is built. In the example of the Pokémon "Starlow", the input sentence for the next step is "It likes to be at sea.".

**Description: Text generation.** To generate the Pokémon description, the pre-trained LM GPT-2 is used. GPT-2 (Radford et al. 2019) is an unsupervised LM which has proved useful for different Natural Language Processing tasks, including language generation. We finetuned the LM on a dataset of real Pokédex entries. The 802 descriptions (about 1,600 sentences) were scraped from the Pokédex website[2].The previously created input sentence is used as a prompt for the generation. The model returns a description of 100 characters which is stripped off after the first three complete sentences. The final description is composed of these sentences and excludes the prompt sentence as it is rather simple, not particularly creative, and would introduce a lot of repetitions due to the limited number of templates. Any mention of a Pokémon in the generated description is replaced by the generated Pokémon name. Finally, the generated Pokémon with its name and description is displayed to the user. In our example, the generated final description would be: "Starlow continually molts the shell and discharges toxic spores. This Pokémon feeds on toxic gases

[1]Given the relatively small number of Pokémon, using only a Pokémon-based LM would result in low probability scores, due to the limited amount of syllable transitions that could be covered.

[2]https://www.pokemon.com/us/pokedex/

and toxins. Starlow is capable of swimming in the sea.". Due to the small size of the training corpus, GPT-2 can easily be overfitted. It may return a description which partially matches one in the Pokédex corpus. To avoid this and ensure novelty of the output, the ROUGE-5 precision score (Lin 2004) is calculated, i.e. the amount of overlap of 5-grams between the generated text and the training dataset. If the ROUGE-5 precision is larger than 0, meaning at least one 5-gram was detected, the description is discarded. A new description will be generated until this requirement is met. [3]

## Preliminary evaluation

The generated names were evaluated in a within-subject study with 33 participants, recruited through convenience sampling[4]. Only participants that had already played Pokémon, excluding those having played the latest generation (Generation VIII), were able to do to the evaluation. This was done to ensure at least a basic level of familiarity with Pokémon. The evaluation consisted of an online survey.

Participants were presented with 4 original Pokémon names (random selection from latest generation) and 4 generated names. For the generated Pokémon names, 4 results that looked convincing (e.g. not presenting the errors mentioned in the Discussion section) were chosen for the evaluation. Participants were asked to classify which of the names were generated and which names were original. This tested how realistic the generated Pokémon names sounded in comparison to original names. In addition, participants were asked to rate the names on two dimensions: likeability and creativity. The two variables were measured using a 5-point Likert scale.

In a follow-up study, 26 participants participants were asked to interact with the Pokérator to generate their own individual Pokémon. We collected their impressions of the system, and checked if it could help them "unveil their inner Pokémon".

## Results and discussion

**Evaluation results.** Regarding the evaluation of Pokémon names, users can identify most of the generated names as such (68% accuracy on average). This gives an indication that the generated names are not similar enough to real Pokémon names, or that it was too obvious that they were constructed from two words. This led to the names being easily distinguishable and indicates that improvements on this front are needed. It is worth noting, however, that original names were often mistaken as generated by the participants (on average, only 44% of non-generated names are correctly classified as "original"), suggesting an important effect of familiarity that should be further investigated.

However, in the dimensions of likeability and creativity (Table 1), no significant difference (using a paired t-test on the average per-participant ratings) could be found. A potential explanation is that generated names are liked as much as (unfamiliar) Pokémon names - again suggesting a strong effect of familiarity -, and that the Pokérator could be reasonably successful at producing creative names.

| | Original | Generated |
|---|---|---|
| Likeability | 3.37 | 3.14 |
| Creativity | 3.39 | 3.20 |

Table 1: Average ratings of Pokémon names

Finally, from the users that could interact with the system, we collected some feedback. About 20% of the participants stated to have found their inner Pokémon.

**Error analysis.** During the name creation process, words are blended together and the final name is selected based on the likelihood of the syllable/character arrangement making up a real word. We did not consider that the original words and their n-grams have a higher probability as they occur in the training data. This leads to a higher probability of word blends containing a full original word.

Another issue is the overfitting of the GPT-2 model which can lead to (parts of) generated descriptions being copied to the output. On the one hand, the relatively small size of the training set can easily lead to overfitting of the GPT-2 model. On the other hand, shorter training can lead to descriptions which are further from potential Pokédex entries. Currently, this problem is tackled by using ROUGE as an 'overfit detector' that will trigger the generation of a new description.

**Limitations.** Currently, the syllables are extracted from the input words using the syllable tokeniser from NLTK. However, the quality of the output of this tokeniser varies greatly, limiting smooth syllable concatenations. In addition, the method for naming in this work limits the possible names to only blended words, whereas real Pokémon names are not always blended words (e.g. "Ekans" which is "Snake" in reverse). For the description, the first limitation is the dependency on answers from ConceptNet. Since it is a crowd-sourced database some words have limited sets of relations to choose from while others have questionable relations, e.g. among the <AtLocation> words for "cat" are "my dogs mouth", "a hat that comes back" and "the Milky Way galaxy". A second limitation is the use of hardcoded templates as it only offers a certain number of simple sentence skeletons to choose from. This can have an effect on the quality of the generated description. In addition, there is a low connection between the description and the name and subsequently the user as only one word from the user-given input is used for generation description. This limits the level of self-identification of the user with their Pokémon. As for the evaluation, the main limitations - apart from the lack of data on descriptions - are its size and the generalisation of the results. We hand-picked a limited number of names, and these are not necessarily representative of the output but rather contain the top percentage of generations.

[3] The code and trained models can be downloaded from https://github.com/ElisaNguyen/Pokerator

[4] We designed and ran an analogous evaluation of the Pokédex descriptions. However, due to a bug in the code that stops GPT-2 from repeating descriptions taken from the training data, its results were biased (i.e., the "generated" condition contained also human-written descriptions), and are thus not included in the current work.

**Future work.** In addition to a more extensive evaluation, which should encompass descriptions in addition to a larger number of generated names, there is room for improvement in the system itself. Further work could be focused on improving the name generation process. Instead of literally using the user's answers, the name could be generated with synonyms of the answers. Another possibility is to use all the answers from the user and generate all possible combinations. These would lead to greater variability in the generated names, and might lead to better results. Besides that, other combinations of syllables could be tried out, e.g. the syllables of one word are placed in the middle the other, as happens in real Pokémon names such as "Exeggutor".

In addition, the evaluation of the different syllable combinations could be improved, e.g. by also using a phonetic LM which could lead to more realistic sounding words.

Looking at the description generation, some issues can be found with the sentence generated with ConceptNet data. Future work could focus on checking the relations retrieved from ConceptNet for grammar and content plausibility.

Finally, more features could be added to the Pokérator, e.g. the Pokémon type and suitable attacks. This would result in a more holistic and complex Pokémon generation.

## Conclusion

We investigated how to develop a creative system that can generate a new Pokémon with a description based on user input. The resulting Pokérator blends user-provided answers together, producing a Pokémon name, and uses their properties to generate a short description. The evaluation shows that generated names are not realistic, but seems to achieve similar levels of likeability and creativity as original Pokémon names. From the individual evaluation, about 20% of participants found their inner Pokémon and could identify with it. With further improvements, we hope the system could prove itself a useful tool to assist Pokémon game developers, or to extend the possibility of user personalisation in the next Pokémon games.

## Acknowledgements

Pokémon names are copyright of Nintendo/Game Freak; no copyright infringement is intended.

## References

Alnajjar, K.; Leppänen, L.; and Toivonen, H. 2019. No time like the present: Methods for generating colourful and factual multilingual news headlines. In *Proceedings of the 10th International Conference on Computational Creativity (ICCC'19)*, 258–265. Association for Computational Creativity.

Banik, R. 2017. The complete Pokémon dataset. Data retrieved from Kaggle https://www.kaggle.com/rounakbanik/pokemon/metadata.

Binsted, K., and Ritchie, G. 1994. An implemented model of punning riddles. Technical report, University of Edinburgh, Department of Artificial Intelligence.

Bodily, P.; Glines, P.; and Biggs, B. 2019. "She offered no argument": Constrained probabilistic modeling for mnemonic device generation. In *Proceedings of the 10th International Conference on Computational Creativity (ICCC'19)*, 81–99.

Cook, M.; Colton, S.; and Gow, J. 2017. The ANGELINA videogame design system - part I. *IEEE Transactions on Computational Intelligence and AI in Games* 9(2):192–203.

Cruz, M. M. 2019. *Os olhos também ouvem: Sistema de geração de imagens de acordo com texto e som para álbuns de música*. Ph.D. Dissertation, Universidade de Coimbra.

Fellbaum, C. 2010. WordNet. In *Theory and applications of ontology: computer applications*. Springer. 231–243.

Gatti, L.; Özbal, G.; Guerini, M.; Stock, O.; and Strapparava, C. 2016. Heady-lines: A creative generator of newspaper headlines. In *Proceedings of the Companion Publication of the 21st International Conference on Intelligent User Interfaces (IUI'16)*, 79–83.

Kawahara, S.; Noto, A.; and Kumagai, G. 2018. Sound symbolic patterns in Pokémon names. *Phonetica* 75(3):219–244.

Lee, J., and Hsiang, J. 2019. Patent claim generation by fine-tuning OpenAI GPT-2. *CoRR* abs/1907.02052.

Lin, C.-Y. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, 74–81.

Loper, E., and Bird, S. 2002. Nltk: The natural language toolkit. In *In Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics. Philadelphia: Association for Computational Linguistics*.

Ng, R., and Lindgren, R. 2013. Examining the effects of avatar customization and narrative on engagement and learning in video games. In *Proceedings of the 18th International Conference on Computer Games (CGAMES 2013)*, 87–90.

Özbal, G., and Strapparava, C. 2013. Namelette: A tasteful supporter for creative naming. In *Proceedings of the Companion Publication of the 2013 International Conference on Intelligent User Interfaces (IUI'13)*, 55–56.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; and Sutskever, I. 2019. Language models are unsupervised multitask learners.

Speer, R.; Chin, J.; and Havasi, C. 2017. ConceptNet 5.5: An open multilingual graph of general knowledge. *Proceedings of the 31st AAAI Conference on Artificial Intelligence (AAAI-17)* 4444–4451.

van Stegeren, J., and Theune, M. 2019. Churnalist: Fictional headline generation for context-appropriate flavor text. In *Proceedings of the 10th International Conference on Computational Creativity (ICCC'19)*, 65–72.

Weide, R. L. 1998. The CMU pronouncing dictionary. URL: http://www. speech. cs. cmu. edu/cgibin/cmudict.