A Tool for Generating Monster Silhouettes with a Word-Conditioned Variational Autoencoder

Adrian Gonzalez,[1] Matthew Guzdial,[2] Felix Ramos[1]
[1] Department of Computer Science, Cinvestav IPN, Unidad Guadalajara, Mexico
[2] Computing Science Department, University of Alberta
adrian.glez@cinvestav.mx, guzdial@ualberta.ca, felix.ramos@cinvestav.mx

Abstract

A character's appearance is crucial to communicating game mechanics to the audience. Creating a game character's design is a time-consuming task and requires design knowledge, skills, and experience. Research on how an AI system might be able to support this design process is an underexplored area. In this work we present a prototype of a variational autoencoder-based creativity support tool that modifies game character silhouettes by using words to describe the design's desired properties.

Introduction

Creating a game character is a demanding task in which design, gameplay and narrative are intertwined. Reaching a point in which all three aspects are successfully conveyed often requires numerous iterations. However, even further modifications to a final design might be needed, for instance, changes in a character's look can play an important role in portraying their growth or showing the player a new game mechanic, such as the protagonist *X*'s armor upgrades in the game *Megaman X*. Therefore, a tool that could assist during this iterative modification process could be valuable.

The challenging task of character design could benefit from the progress in automatic image synthesis techniques. Image generation methods can produce high quality images in domains such as human faces or animals (Karras, Laine, and Aila 2018). However, many approaches lack an interface that allows non-researcher users to intuitively control the system (Voynov and Babenko 2020). While there exist programs that possess them, as in (Simon 2020), their controls do not necessarily correspond to human-interpretable concepts. We argue that using natural language to adjust a model's output can increase the controllability for users unfamiliar with image generation methods.

In this demonstration we present a creativity support tool for creating monster silhouettes using natural language to shape the result. It relies on a Conditional Variational Autoencoder (Sohn, Lee, and Yan 2015) conditioned on both silhouette images and word embedding vectors generated by a text embedding model (Bengio et al. 2003). We used images and text from the Pokémon games because their visual designs convey gameplay information like their strengths and weaknesses (Gonzalez, Guzdial, and Ramos 2020).

With this tool we aim to evaluate users' response to natural language-based interfaces for image generation models in the games domain. This will also help us to determine this natural language-based paradigm's suitability in other design tasks such as coloring or composition (Gonzalez 2020).

Related Work

Creativity Support Tools (CSTs) aim to aid their users' creative process through the use of Artificial Intelligence (Shneiderman 2007; Frich et al. 2019). Some notable image generation-based CSTs are Artbreeder (Simon 2020) and MonsterGAN (Huang 2020). Both present the user with several pictures from which they can select, modify, or combine with others. However, the controls for these systems are often hand-labeled latent vectors, where the vectors do not necessarily correspond to the given labels. This makes the changes seem unpredictable. In contrast, our prototype uses the user's design intentions, expressed in natural language, to guide the generation model.

Neural networks that generate images from a text description allow people without the technical skills or knowledge to create illustrations that, to some extent, fit their described criteria. A notable project that works with text descriptions is DALL-E (Ramesh et al. 2021). However, some limitations for these systems can be blurry pictures (Reed et al. 2016), prolonged or unstable training stages (Zhang et al. 2017), and massive hardware requirements (Ramesh et al. 2021), all of which might discourage potential end users. Instead of using a model like DALL-E we opted for a simpler model and individual terms to describe the desired result. We argue that early design stages, like silhouette design, can allow the use of VAE's blurry images, thus we developed a CST to generate monster silhouettes.

User Interface Overview

We show our application's user interface in Figure 1. Video demonstration available[1]. It consists of two square images and eight text inputs. The image on the left shows the noise image passed as input to our system. The text fields allow the user to type their desired design elements, these can be

[1] Video: https://youtu.be/r9Ds4C98VDM

any words such as *strong* or *flying*. The image on the right displays how our model transforms the first image by using the eight words given as input. In Figure 1b we present how our system interprets a different noise image using the same text input as in Figure 1a. Figure 1c shows the opposite scenario, same noise, but different words.

Additionally, there are three buttons: “Randomize image” which changes the input noise image on the left; “Randomize Words” that replaces all the current text inputs with ones associated with a random Pokémon from the dataset; and “Save Image” which saves the current result, input image and words. Even though it is possible, we do not allow the user to provide an image as an starting point, since we plan on comparing the current CVAE backend against a GAN model that does not allow it either.

The application was developed using Python 3.6, Tensorflow and Keras to represent the CVAE model, and the PyQt5 library (PyQt 2015) for the UI. An online version is being developed to gather user feedback.

Backend VAE

To train our CVAE we used Pokémon images coupled with eight words that describe them. We obtained each Pokémon’s words by scraping its biology section from the Pokémon *wikia* page Bulbapedia (Bulbapedia 2021). We took each Pokémon’s top eight words (according to TF-IDF), and assigned them to the Pokémon’s image. Our model’s input data consists of a 32*32 pixel image of a white silhouette in black background and eight 128-dimensional vectors representing the natural language words.

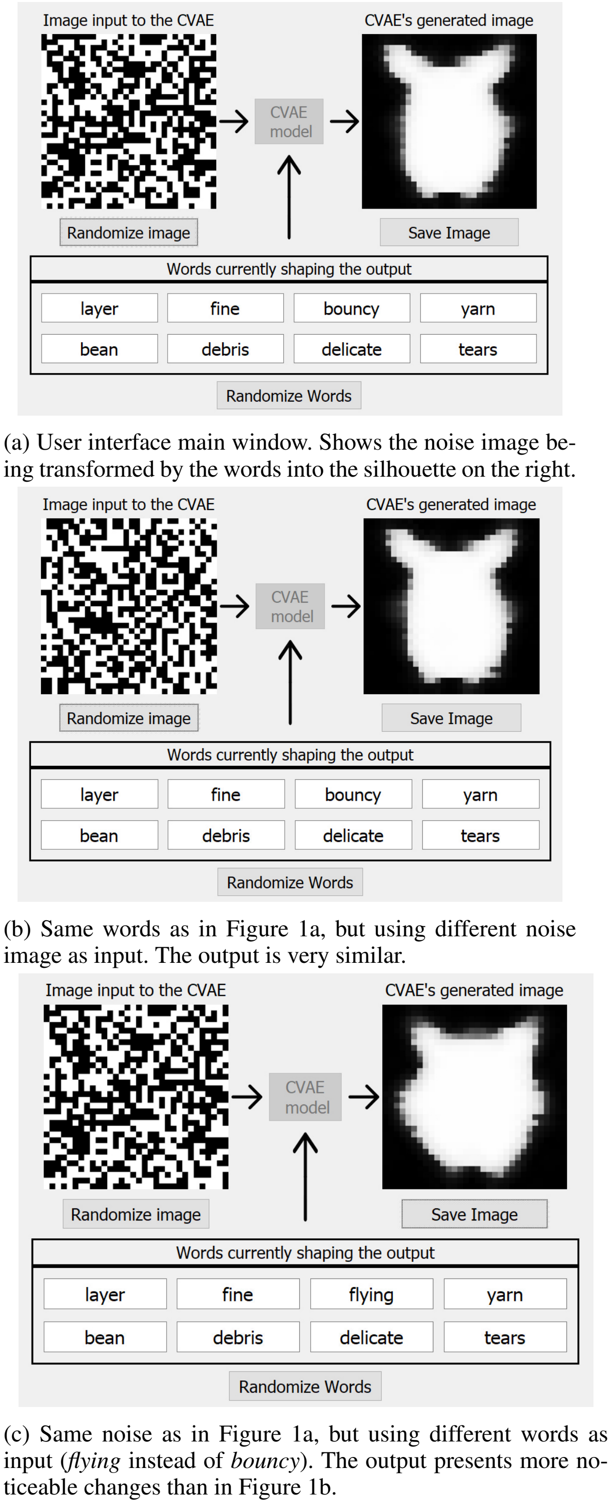
Our models is a Conditional Convolutional Variational Autoencoder (Sohn, Lee, and Yan 2015) similar to the one shown in (Krishnamoorthy 2020). Instead of using fully connected layers to transform the conditional labels, we used the word embedding representations directly. We reshaped them as a 32*32 array and concatenated them as a new channel for the image. We employed Tensorflow-hub’s Neural-Net Language Model (Google 2021) to obtain the word embedding vectors. We used OpenCV’s bilateral filtering (OpenCV 2021) to remove isolated noise from our model’s outputs and make the silhouette sharper, as VAEs are known to produce blurry results (Kingma and Welling 2014).

Discussion

Our next step with this project will be to perform a human subject study to gather feedback on the tool’s helpfulness and controllability during early stages of the character design process. Measuring the users’ level of expertise will also be crucial in understanding the tests’ results. Considering the current system is trained on Pokémon data, the experiment will likely be designed around tasks that have consistent results in that domain, such as creating *starter* or *legendary* Pokémon.

Acknowledgements

This work was supported by CONACYT through the doctoral scholarship number CVU-777880.



layer fine bouncy yarn
bean debris delicate tears

(a) User interface main window. Shows the noise image being transformed by the words into the silhouette on the right.

layer fine bouncy yarn
bean debris delicate tears

(b) Same words as in Figure 1a, but using different noise image as input. The output is very similar.

layer fine flying yarn
bean debris delicate tears

(c) Same noise as in Figure 1a, but using different words as input (*flying* instead of *bouncy*). The output presents more noticeable changes than in Figure 1b.

Figure 1: User interface examples.

References

Bengio, Y.; Ducharme, R.; Vincent, P.; and Janvin, C. 2003. A Neural Probabilistic Language Model. *J. Mach. Learn. Res.* 3: 1137–1155. ISSN 1532-4435.

Bulbapedia. 2021. Bulbapedia. https://bulbapedia.bulbagarden.net/wiki/Main_Page. Accessed: 2021-06-12.

Frich, J.; MacDonald Vermeulen, L.; Remy, C.; Biskjaer, M. M.; and Dalsgaard, P. 2019. *Mapping the Landscape of Creativity Support Tools in HCI*, 1–18. New York, NY, USA: Association for Computing Machinery. ISBN 9781450359702. URL https://doi.org/10.1145/3290605.3300619.

Gonzalez, A. 2020. Artificial Intelligence as an Art Director. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment* 16(1): 337–339. URL https://ojs.aaai.org/index.php/AIIDE/article/view/7454.

Gonzalez, A.; Guzdial, M.; and Ramos, F. 2020. Generating Gameplay-Relevant Art Assets with Transfer Learning. *arXiv preprint arXiv:2010.01681* .

Google. 2021. Tensorflow-hub Neural-Net Language Model. https://tfhub.dev/google/nnlm-en-dim128/2. Accessed: 2021-06-12.

Huang, K. 2020. MonsterGAN. https://www.kylehuang.design/monstergan. Accessed: 2021-06-13.

Karras, T.; Laine, S.; and Aila, T. 2018. A Style-Based Generator Architecture for Generative Adversarial Networks. *CoRR* abs/1812.04948. URL http://arxiv.org/abs/1812.04948.

Kingma, D. P.; and Welling, M. 2014. Auto-Encoding Variational Bayes. *CoRR* abs/1312.6114.

Krishnamoorthy, A. 2020. PyTorch-VAE. https://github.com/AntixK/PyTorch-VAE/blob/master/models/cvae.py. Accessed: 2021-06-14.

OpenCV. 2021. OpenCV. https://opencv.org/about/. Accessed: 2021-06-12.

PyQt. 2015. PyQt5. https://doc.bccnsoft.com/docs/PyQt5/. Accessed: 2021-06-12.

Ramesh, A.; Pavlov, M.; Goh, G.; Gray, S.; Voss, C.; Radford, A.; Chen, M.; and Sutskever, I. 2021. Zero-shot text-to-image generation. *arXiv preprint arXiv:2102.12092* .

Reed, S.; Akata, Z.; Yan, X.; Logeswaran, L.; Schiele, B.; and Lee, H. 2016. Generative Adversarial Text to Image Synthesis. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, 1060–1069. JMLR.org.

Shneiderman, B. 2007. Creativity Support Tools: Accelerating Discovery and Innovation. *Commun. ACM* 50(12): 20–32. ISSN 0001-0782. doi:10.1145/1323688.1323689. URL https://doi.org/10.1145/1323688.1323689.

Simon, J. 2020. Artbreeder. https://artbreeder.com. Accessed: 2021-06-13.

Sohn, K.; Lee, H.; and Yan, X. 2015. Learning Structured Output Representation using Deep Conditional Generative Models. In Cortes, C.; Lawrence, N.; Lee, D.; Sugiyama, M.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Voynov, A.; and Babenko, A. 2020. Unsupervised Discovery of Interpretable Directions in the GAN Latent Space. *ArXiv* abs/2002.03754.

Zhang, H.; Xu, T.; Li, H.; Zhang, S.; Wang, X.; Huang, X.; and Metaxas, D. N. 2017. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, 5907–5915.