Artificial Language Processing: Getting Creative with NLPs

Thomas Zacharias, Ashutosh Taklikar *

February 2020

Abstract

Fictional languages have become increasingly popular over the recent years appearing in novels, movies, TV shows, comics, and video games. While some of these fictional languages have a complete vocabulary, most do not. Here, we propose a deep learning solution to the problem. Using style transfer method as a machine translation tool, we demonstrate the generation of new words for the required language, while maintaining the style of the creator, hence extending the vocabulary of that language. We further use StyleGAN to generate new scripts for the designed fictional language.

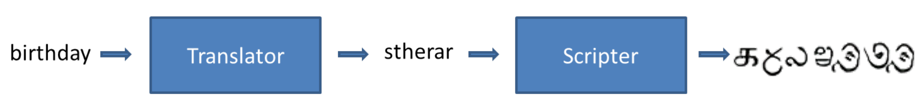

Figure 1: **Net designed system.** The image shows our net system with the translator that enables us to understand the style of the author, translate words that were never originally translated, and then write them in our designed script using our scripter.

*Both authors procrastinated equally in this work.

1 Introduction

Languages can be broadly classified into natural and constructed languages. Natural languages are languages that evolved over time within a community under some cultural framework. These are constantly evolving in humans through use and repetition without conscious planning or premeditation. They differ from constructed languages that are purposefully designed and are a result of controlled intervention and language planning. Constructed languages are used for various applications such as human communication (international auxiliary language and code); to give fiction or an associated constructed setting an added layer of realism; for experimentation in the fields of linguistics, cognitive science, and machine learning; for artistic creation; and for language games.

Fictional languages are constructed languages designed for a particular fictional setting - a book, movie, television show, or video game. Typically, the creation of a single person, they differ from natural languages in that they do not evolve over time out of a particular culture or community. The only native speakers of these fictional languages are the fictional characters that have been created for the fictional setting. That said, there is a large demand for fictional languages in the entertainment industry with many of these languages being used in western blockbusters, video games, comic book, and novels. Quenya and Sandarin in The Lord of the Rings, Na'vi in Avatar, Dothraki in Game of Thrones, and Klingon in Star Trek are a few examples of fictional languages with complete vocabulary and grammatical rules that anyone can learn. Some of these languages such as Na'vi and Dothraki are constructed by professional linguists called conlangers whose skills are usually in high demand in Hollywood. However, there are plenty of fictional works where the artists are unable to employ conlangers and are required to come up with a language of their own. While there are exceptionally talented authors like J.R.R. Tolkein who created multiple complete fictional languages such as Common Eldarin, Quenya and Goldogr for "The Lord of the Rings", most authors usually create a limited dictionary for their fictional language consisting of a few hundred English words with their corresponding translations to the fictional language. Such languages are incomplete in two broad ways- 1) they do not have a complete vocabulary; 2) their grammatical structure and rules are not well defined. One famous example is the "Ancient language" used in Eragon. Such languages even with their limited vocabulary and grammar are extremely popular among the fandom which is especially evident in meetups such as Comic-Con.

In the first part of this work, we use neural networks to extend the vocabulary of fictional languages- i.e., take a limited dictionary of a few hundred words and their translations and try to extrapolate the vocabulary of the language while maintaining the style of the author. In other words we wanted to create a black box where the user can input a word he wants translated and our network will produce translated word, even if the word was never translated by the author. To our knowledge, neural networks have not been used for such an application until now. We use two different approaches involving Recurrent Neural Networks(RNNs)- one using a simple RNN and another using style transfer- and analyze their results. In the second part of this project, we try to create a script for the fictional languages using StyleGAN. Apart from the obvious applications in the entertainment industry, these methodologies and ideas could also be relevant to natural languages such as Modern Hebrew, whose vocabulary is continuously evolving and growing.

2 Methods

2.1 Language translator- Word generator using a simple RNN

There have been plenty of works that have used Recurrent Neural Networks (RNNs) for character generation on different datasets [7]. One of these implementations uses RNNs to generate new baby names by training the network on a list containing popular preexisting names. As a Naive approach, we decided to train a simple RNN character-level language model on all the words available in the fictional language. Once the training was done, we could give the trained RNN the first character of our new fictional word and it would provide us with an output. This seemed to be an easy way to generate new fictional words. In order to create something more like a dictionary, where an input English word is mapped to the fictional language, we decided that the initial character would be based on some sort of a combination of the characters of the English word we wanted translated. We used the mean of the embeddings of the characters of the English word, but any other indicator could have also been used.

We note a few disadvantages of the above method-

1. Not based on a dictionary- The network is set to generate new words in the fictional language independent of their English translations. However, we would like some sort of a mapping between the two languages and hence decided to give the starting letter as the mean of the embeddings of the characters in the English word. Multiple English words could have the same embedding and hence the mapping need not be one to one. This would mean

that our network will output the same fictional words for different English words- something that is undesirable.

2. Limited dataset- We would like to work with hundreds of words, while most character level language models work with a lot more (on the order of magnitudes higher). The lack of training data will make the RNN less robust.

Aware of these constraints we decided to also try a different approach.

2.2 Language translator- Word generator using style transfer from text

Style transfer has become very popular in recent years especially in computer vision [4, 14]. In these works, the neural network sees two images- the content image and the style image and tries to transfer the style of one image to the other. Figure 2 shows an example of style transfer being used in images.

Figure 2: Example of style transfer in images

Style transfer in text [23, 19] is based on a similar idea- you input a particular text to your trained network and the network tries to generate new text with a different style. This has been used to demonstrate sentiment modification, word substitution decipherment, word order recovery, sentence rewriting, etc. [23, 19]. We propose using style transfer on languages- to try get the network to learn the underlying style of languages and act as a tool that is able to translate one language to another. Note that up until now (to our knowledge), style transfer had only been used on the sentence level. While there have been proposals to use it for machine translation [23], we were unable to find any works that have actually implemented this.

We will now delve deeper into the work of Shen et al. [23] on style transfer whose architecture we decided to implement to achieve language style transfer for our case. The novelty in their work lies in introducing a refined alignment of sentence representation across text corpora. Their network consists of an encoder generator system. The encoder takes a sentence and its style indicator as input and outputs a style-independent content representation. The generator then takes the style independent content representation applies a certain style and presents an output sentence. If the generator used the same style as the encoder, then the system would essentially act as a variational auto-encoder outputting sentences similar to the original input. If the generator uses a different style, however, the system would act as a transfer model with the output sentences containing a the new style. They also propose aligning the outputs with examples from the other sentence domain at a distributional level thus realizing cross-alignment (see Figure 3).

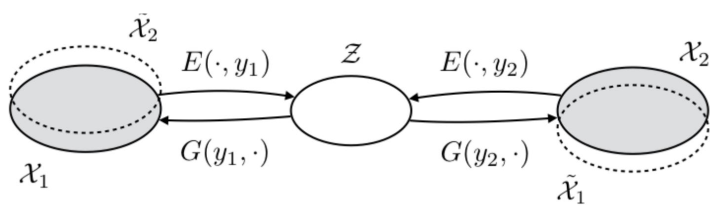

Figure 3: **Cross-alignment** [23]. An overview of the cross-alignment method. χ_1 and χ_2 are two sentence domains with different styles y_1 and y_2, and Z is the shared latent content space. Encoder E maps a sentence to its content representation, and generator G generates the sentence back when combining with the original style. When combining with a different style, transferred $\tilde{\chi_1}$ is aligned with χ_2) and $\tilde{\chi_2}$ is aligned with χ_1 at the distributional level.

One of the use cases they propose is for sentiment modification. The goal was to change the underlying sentiment i.e. applying "style transfer" between negative and positive sentences. Datasets containing 250K negative sentences and 350K positive sentences were collated from readily available user ratings from Yelp restaraunt reviews. During training the network was shown the two files containing positive and negative sentiments and trained according to the loss function introduced in their paper. During testing, either positive or negative sentences were given and the network was able to convert it to the opposite sentiment. Shen et al. were able to quantitatively analyze the performance of their network during training by measuring how often a transferred sentence had the correct sentiment according to a pre-trained sentiment classifier. This however was not an good measure for their generation task and they resorted to using human evaluations on the success of sentiment and fluency instead.

We wanted to take what they did for sentiment transfer and apply it to our problem- the problem of translation. One of the reasons we chose this paper was because the training data is not necessarily parallel. The number of words in the English vocabulary is much larger than the set of a few hundred words available in the fictional language. By using a network that works with non-parallel text, we could exploit this to our advantage. Apart from this, one way to increase the size of the fictional vocabulary could be by using examples from the language that the fictional language is based on. E.g. in the case of the Eragon, the Ancient language is based on ancient Norse and Celtic. By incorporating words from these languages, the number of examples that the network sees during training can be increased by a large amount.

The key differences/challenges while implementing their model:

1. Sentence level vs word level. Their model focused on style transfer in the sentence level while we needed style transfer at the word level. This was implemented in our code and the way we designed our datsets.

2. Measure to analyze output data. They were able to use a pre-trained sentiment classifier network to have some quantitative measure to analyze their results. This measure could used in our case for obvious reasons.

We therefore came up with our own measure which we discuss now in detail. Our proposal to obtain a measure to analyse the generated words[1]:

1. Qualitative analysis. Qualitatively, a) we would like the words to be relatively short since most languages contain short words and b) the words should contain vowels in them so that they can be pronounced.

2. Quantitative analysis. A quantitative way to analyze the viability of generated words for a fictional language does not seem intuitively possible. To overcome this dilemma we propose the following- instead of training the network on the vocabulary of English and the fictional language, we shall train it on a subset of the vocabulary of two natural languages say English and Hebrew where Hebrew will act as our fictional language. If the network is able to generate actual Hebrew/English words that it never saw before, then we could say that the idea is a success i.e. given English and a fictional language, the network will be able to new words that the author would have come up with if he had completed the vocabulary. It would also be considered successful if the output words sound/feel like they belong to that language. This requires human evaluation but we would like to remind you that a complete quantitative analysis (without human intervention) was not

[1]The authors are not linguists. Their views are based on personal intuition, their experiences, and a bunch of google searches.

possible for the sentiment transfer example either.

Armed with these linguistic epiphanies and a whole lot of caffeine, we later (see section 4.2) test our hypothesis.

2.3 Generating new scripts using StyleGAN

In 2018, a team from Nvidia proposed an alternate architecture for GANs that greatly improved the quality of generated images and enabled intuitive, scale specific control through the latent space [9]. Inspired by style transfer literature, the authors called their network the StyleGAN. Before this, the state of the art GANs at that time [8, 16, 2] acted as black boxes with no intuitive control over specific features. StyleGAN introduced a completely re-designed generator architecture that enabled novel methods to control the generated images. Instead of feeding the input latent code only at the beginning of the network, they introduced a mapping network that first transforms it to intermediate latent codes which are then affine transformed to produce different styles that control the layers of the synthesis network via adaptive instance normalization (AdaIN) [6, 3, 5]. In addition to this, random noise is also added to the synthesis network. The authors show that this design allows the intermediate latent space to be less entangled than the input latent space something also demonstrated by Yujun et al. [20]. Figure 4 shows a visual comparison between the generative networks of the traditional GAN and the StyleGAN that Karras et al. [9] propose.

StyleGAN was famously used to generate realistic images of faces of people that do not exist (see Figure 5). By training it on custom datasets people have managed to generate images of new beds, paintings, watches, and even Pokémon (bringing new meaning to the phrase "gotta catch ém all"!) [11].

Inspired by Robert Munro [17], we wanted to use StyleGAN to generate new scripts based on the existing scripts procured from different human languages. This could be done by training the network on images of scripts from different languages.

Note that an improved version of StyleGAN, StyleGAN2 [10], was introduced in 2019. The work exposes and examines characteristic artifacts in the generated outputs of the StyleGAN and proposes changes in the model architecture and training methods to address them. While being a more robust network, we decided to work with the original StyleGAN because our training images were fairly simple- characters from different languages.

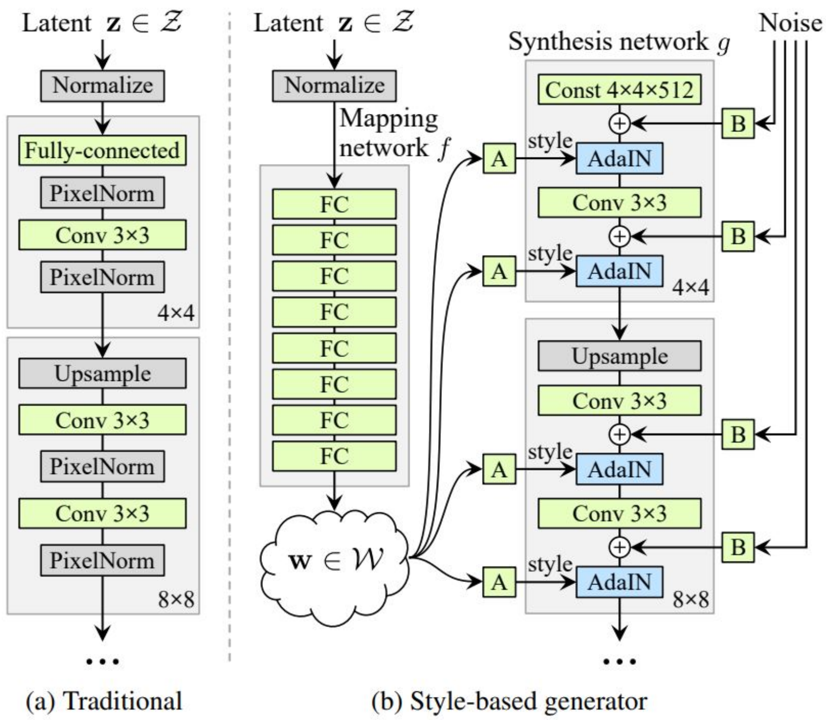

Figure 4: **StyleGAN generator** A visual comparison between a traditional generator and the style based generator introduced by Karras et al. [9]

3 Data

3.1 Language translator- Word generator using a simple RNN

The dataset to train the RNN should contain words from the fictional vocabulary that needs to be extended. As discussed earlier, in order to measure how well the network performs, we used the network on real languages instead. We selected the 1000 most common English words spoken in the United States [18] and used google translate to translate them to different languages. Of the 1000 words, 850 were selected for training, 50 for validation, and 100 for testing. It is important to note that we worked with hundreds of words during training which is approximately the number of words we would find in the vocabulary of a fictional language. We performed experiments for various Indian languages, but for the purposes of

Figure 5: **StyleGAN on faces [9]** These people do not exist. They were produced by the generator of the StyleGAN network. Various aspects of the image are under the control of the user.

the report, we shall later show the results we got for Hebrew.

3.2 Language translator- Word generator using style transfer

The dataset for style transfer requires two files containing the content in the two different styles (languages) that we want transferred. Similar to the dataset for the RNN, we selected the 1000 most common English words in the United States [18] and paired it with different languages. During testing, we could input words that the network never saw before in either language, and observe its output after it had transferred the style.

3.3 Script generator

StyleGAN like most GANs works with images. In order to generate new scripts based on the scripts of the known human languages, we required images of the known scripts. Unicode is the standard for encoding text represented in most of the world's writing systems. JPG images could be produced using Python libraries for every single Unicode character. By choosing which Unicode blocks to keep and which to leave, it is possible to generate scripts based on specific languages up to the choice of the user. We chose to work with Arabic, Hebrew, Devanagri, Malayalam, Hindi, and

Latin scripts. Figure 6 shows a collation of images of a number of characters from these scripts.

Figure 6: **Images of real characters.** Images of some of the characters from the real scripts used to train the StyleGAN.

4 Experiments and results

4.1 Language translator- Word generator using a simple RNN

As mentioned earlier, the RNN was trained on different language datasets that we generated. For the purpose of this report, we shall focus on Hebrew and Arabic. Other datasets we generated and worked with are provided in the github link in the appendix (see section 7). After training on Hebrew words, during testing the RNN was given the embedding of the English words we wanted translated. Figure 7 shows the English words we wanted translated and the proposed Hebrew and Arabic translations.

One can see that there is a lot of repetition both at the word level and the character level. We believe that this is due to the small size of the dataset- giving a larger dataset should improve the results. Note that since our final goal was to work with fictional words and most vocabularies do not extend beyond hundreds of words, we did not try seeing the results for a larger training dataset. We would like to note, however, that the words generated are short and have vowels- both of these qualities are desirable (as discussed earlier). But apart from this, it is fair to say that these results can not be used and this naïve approach is a complete failure.

English word we want translated	Proposed Hebrew translation	Proposed Arabic translation
s u c c e s s	ולה	ال
c o m p a n y	ולה	لال
s u b t r a c t	ולה	ال
e v e n t	ולהת	ال
p a r t i c u l a r	ול	ال
d e a l	ולה	الة
s w i m	ולה	الال
t e r m	ולה	لا
o p p o s i t e	ולה	لال
w i f e	ולי	لال
s h o e	ול	ال
s h o u l d e r	ולה	ال
s p r e a d	וב	ال
a r r a n g e	ולהת	ال
c a m p	ולה	ايرة
i n v e n t	ולה	ال
c o t t o n	ולה	الة
b o r n	ולהת	ال
d e t e r m i n e	ולהת	ال
q u a r t	ולה	ايرة
n i n e	ולה	ال
t r u c k	ולהת	ال
n o i s e	ול	ال

Figure 7: **English words with proposed translations.** (Left) English words given to the network during testing. (Center) Output of the RNN trained on Hebrew words- the proposed Hebrew translation for the English word. (Right) Output of the RNN trained on Arabic words- the proposed Arabic translation for the English word

4.2 Language translator- Word generator using style transfer

As mentioned earlier, to perform style transfer we need to train the network on two datasets with content written in different styles. To perform style transfer on languages- two datasets with words written in the two different languages are required. Since the style transfer is non-parallel, the two datasets do not have to be a dictionary but after multiple experiments we recommend using something like a dictionary so that each word in each dataset is unique. Similar to the RNN, we took the 850 most used English words in the United States, and their corresponding translations to different languages and fed it to the network for training. For the purpose of this report, we will focus on the results of the English-Hebrew training pair. Once the network was trained, we inputted English and Hebrew words that the network never saw and had the network "translate" them to the other language.

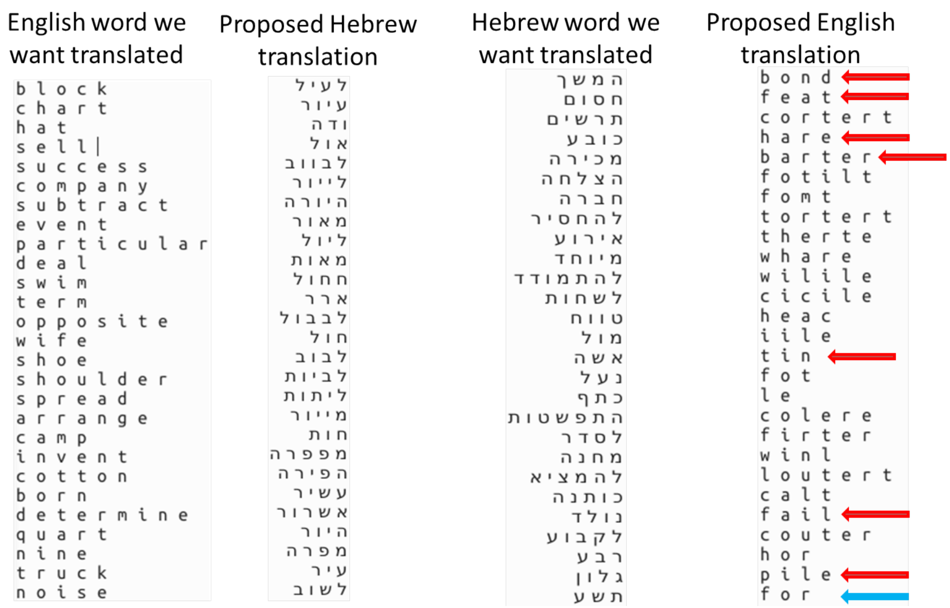

Figure 8: **English words with proposed translations.** (First column) English words given to the network during testing- the network never saw these words. (Second column) Output after the style transfer- the proposed Hebrew translation for the English word. (Third column) Hebrew words given to the network during testing- the network never saw these words. (Fourth column) Output of the style transfer- the proposed English translation for the Hebrew word. Words marked with arrows are generated English words that exist in the language with red arrows pointing to words the network never saw and blue to the words that the network saw.

Figure 8 shows English words with the proposed Hebrew translations, and also Hebrew words and their proposed English translations. The red arrows mark English words that exist in the English language but the network never saw during training and the blue arrow marks English word the network generated but had already seen during training. Seeing that the network was able to generate words that exist in the English language without ever seeing them shows that it was essentially able to extend the language from a smaller subset. This was exactly our goal. Cases of repetition, where the network generates a word it saw before can be treated like this- one simply needs to ignore the repetitions and generate a new word (by giving some other input instead). At the end of the day we are simply trying to generate new words that retain the style of the original language. The dictionary mapping can be created later once the words have been generated. This is especially true since the style transfer method we use is non-parallel. Another observation is that the words that are not marked with arrows are also quite convincing- they can all be pronounced and some of them sound and feel like actual English words. The authors do not understand Hebrew so are unable to analyze in depth the proposed Hebrew words. The reason why we have added it here (apart from being interesting to the reader who does understand Hebrew) is because in this case, Hebrew essentially acts as our "fictional" language. A surface level analysis shows a variety in characters, presence of vowels, and no obvious repetitions. As far as we are concerned, the proposed Hebrew words could be the actual translations- we wouldn't know the difference. We believe that this is a convincing demonstration that our idea works. Hence, we have shown that languages can be extended using deep learning - in particular using textual style transfer.

4.3 Script generator

Once the dataset of images of different scripts was created, as mentioned earlier, we used the StyleGAN network that was open-sourced by Nvidia in 2019. Figure 9 shows a collation of some of the characters that the StyleGAN was able to generate. By selecting the desired images, one can generate a list of characters that could act as the alphabets for the fictional language. In case of the Ancient language in Eragon, the author has written words in the Latin (English) script. One could easily map the 26 alphabets in English to desired characters generated from the StyleGAN enabling one to create a complete script for the fictional language.

5 Future Work

We think it would be very interesting if a network could learn an entire language- not just the vocabulary but also the grammar. While there have been proposals to quantitatively measure output sentences [15], their use-cases are limited and style transfer is still predominantly dependent on human intervention. This is an active field of research and we look forward to new developments in the coming years. Trying other textual style transfer

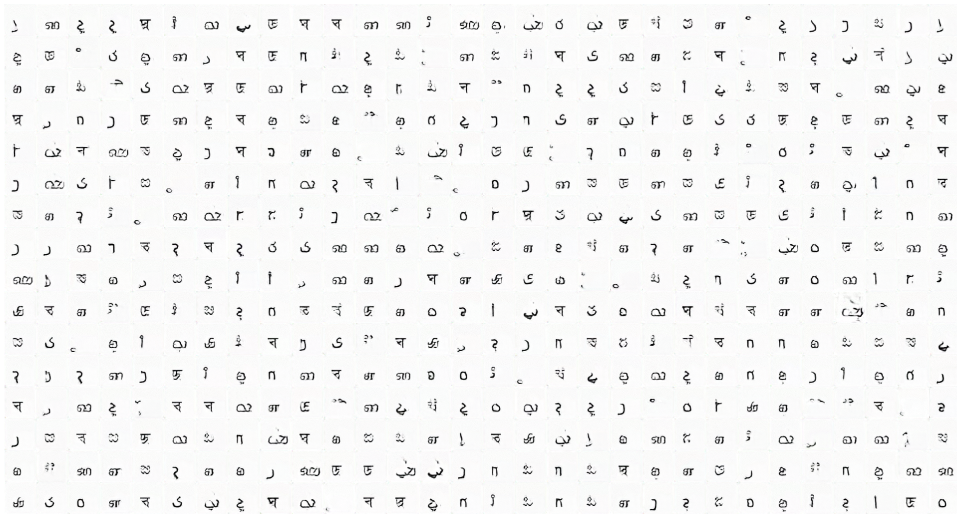

Figure 9: **Characters generated by StyleGAN.** Images of some of the fake characters generated by the StyleGAN.

networks [22, 13, 21, 12, 1] and comparing results would also be quite interesting. Style-mixing could also be a possible option when creating new scripts. We also think that it would be insightful to show our results to linguists, especially conlangers who have experience in constructing languages- their inputs could be invaluable.

6 Conclusion

We used various methods in deep learning to extend the vocabulary and generate a new script for fictional languages. We tried two methods to extend the vocabulary- by training a simple RNN and then trying to use textual style transfer as a translation tool. We proposed ways to analyze the outputs and saw that the results using the style transfer method were much better. We also generated new characters for our fictional language using StyleGAN- a GAN based architecture that borrows from style transfer literature.

7 Appendix

Check out our code here.

A summary of our implementations:

- Traversed through the multiverse to obtain datasets of different languages (fictional and natural) and implemented a program to generate datasets in the required format.
- Implemented a character level language model using Recurrent Neural Networks in PyTorch.
- Upreved the style transfer code [19] from python 2 Tensorflow 1.13 to python 3 Tensorflow 1.19 and updated code to a character level language model.
- Generated images of characters of existing Unicode scripts using python libraries.
- Used generated images to train Nvidia's StyleGAN [9] model.

References

[1] Reina Akama, Kazuaki Inada, Naoya Inoue, Sosuke Kobayashi, and Kentaro Inui. Generating stylistically consistent dialog responses with transfer learning. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 408–412, Taipei, Taiwan, November 2017. Asian Federation of Natural Language Processing.

[2] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis, 2018.

[3] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style, 2016.

[4] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2414–2423, 2016.

[5] Golnaz Ghiasi, Honglak Lee, Manjunath Kudlur, Vincent Dumoulin, and Jonathon Shlens. Exploring the structure of a real-time, arbitrary

neural artistic stylization network. *Procedings of the British Machine Vision Conference 2017*, 2017.

[6] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[7] Andrej Karpathy. The unreasonable effectiveness of recurrent neural networks, http://karpathy.github.io/2015/05/21/rnn-effectiveness/, 2015.

[8] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation, 2017.

[9] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2019.

[10] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan, 2019.

[11] Jamshed Khan. Stylegan: Use machine learning to generate and customize realistic images, https://heartbeat.fritz.ai/stylegans-use-machine-learning-to-generate-and-customize-realistic-images-c943388dc672, 2019.

[12] Dianqi Li, Yizhe Zhang, Zhe Gan, Yu Cheng, Chris Brockett, Bill Dolan, and Ming-Ting Sun. Domain adaptive text style transfer. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019.

[13] Juncen Li, Robin Jia, He He, and Percy Liang. Delete, retrieve, generate: a simple approach to sentiment and style transfer. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1865–1874, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[14] Fujun Luan, Sylvain Paris, Eli Shechtman, and Kavita Bala. Deep photo style transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4990–4998, 2017.

[15] Remi Mir, Bjarke Felbo, Nick Obradovich, and Iyad Rahwan. Evaluating style transfer for text. *Proceedings of the 2019 Conference of the North*, 2019.

[16] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks, 2018.

[17] Robert Munro. Creating new scripts with stylegan, https://towardsdatascience.com/creating-new-scripts-with-stylegan-c16473a50fd0, 2019.

[18] David Norman. 1000 most common english words spoken in united states of america, https://gist.github.com/deekayen/4148741, 2019.

[19] Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. Style transfer from non-parallel text by cross-alignment. In *Advances in neural information processing systems*, pages 6830–6841, 2017.

[20] Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. Interpreting the latent space of gans for semantic face editing, 2019.

[21] Yi Zhang, Jingjing Xu, Pengcheng Yang, and Xu Sun. Learning sentiment memories for sentiment modification without parallel data. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018.

[22] Jake Zhao, Yoon Kim, Kelly Zhang, Alexander M. Rush, and Yann LeCun. Adversarially regularized autoencoders, 2017.

[23] Yanpeng Zhao, Wei Bi, Deng Cai, Xiaojiang Liu, Kewei Tu, and Shuming Shi. Language style transfer from sentences with arbitrary unknown styles. *arXiv preprint arXiv:1808.04071*, 2018.