Applying Artistic Style Transfer to Natural Language

Thaminda Edirisooriya
tediris@cs.stanford.edu

Morgan Tenney
mtenney@cs.stanford.edu

Abstract

Machine learning has been employed in text generation with varying levels of success, with such projects as chatbots and Twitterbots impersonating an actual human being. However, these methods often work through identity mimicry. They say phrases that their archetype would have said, assuming a specific persona. Not yet has the problem been approached with the goal of purely capturing style - that is, being able to take any sentence, and while preserving its meaning transform it into the style of a different speaker or author. In the visual arts, however, this problem has been explored. Artistic style transfer takes two images, and applies the style of one to the content of the other. We aim to apply the methodology of visual artistic style transfer to the realm of natural language. We define a style cost and a content cost, and by minimizing the two we acquire the style transferred sentence. By using an author classifier, we are able to rate the similarity of the styles of an input text and a source text, effectively acting as a cost function for style. For content, we apply average pooling to the input and proposed output phrases to get a rough approximation of content similarity while enabling flexibility in sentence structure.

1 Introduction

Machine learning practices have for some time been applied to projects related to conversation simulation, such as chatbots. Recent practice has involved the use of personas, or identities for the artificial conversationalist to assume, in order to give the chatbot a consistent manner of speaking and question answering. However, not much work has been done in the realm of conversational style independently of identity. When we began this project, this was the issue we sought to address: could we give a text-based chatbot a style of speaking independently of its assumed identity?

Methods of applying style to unrelated content are not common in the realm of natural language processing, but some algorithms have been very successful in the visual arts. Artistic style transfer is an intuitive and effective method for applying the style of one piece to the content of another, by defining and then optimizing cost functions for content and for style.

While we initially set out to create a chatbot with customizable style, we ultimately decided that the style adoption was the key component of our project, and dedicated our work towards the quality of the style transfer. Our problem statement thus became: given an input phrase, rewrite the same phrase using the style of the specified author or speaker.

2 Background/Related Work

Before we discuss the work that inspired our approach to capturing voice, we should describe a similar project with a different approach. Nguyen et al. (2017) aimed to create a chatbot with personality and identity, using transcripts from television shows to train their model. The chatbot speaks with the same style as the original character, and answers questions similarly to how the original character would do so. They capture a personality using the third and final phase of their training method. After first training on show transcripts and the Cornell Movie-Dialogs Corpus, then limiting to only the show transcripts, they finish training by using only the utterances of the characters they are trying to impersonate, fine-tuning their

model to the characters personality. While the end goal is similar to ours, our approach has a fundamentally different foundation, as well as having the goal of functioning independently from identity.

Our approach is based off of the model described by Gatys et al. (2015) use of convolutional neural networks for artistic style transfer in graphic art. This process combines features from two images: one that has lower-level features such as color and texture that we want to transfer, and the other that captures content (ie. the buildings in photograph A in Figure 1) that we want to preserve. Imagine that we have some function $L_{content}(x, c)$ that can measure how different *in content* two images x and c are from one another. In other words, it is a loss function that gets closer to zero the more similar *in content* the two images are. Also imagine that we have some function $L_{style}(x, s)$ that can measure how different *in style* two images x and s are. In theory, our problem is solved if we can successfully solve the optimization problem:

$$x^* = \underset{x}{\operatorname{argmin}}(\alpha L_{content}(c, x) + \beta L_{style}(s, x))$$

where c is the content image, s is the style image, x is the desired combination of the two, and α and β are constants that allow emphasis on content relative to style. To find the function $L_{content}$, Gatys et al. (2015) applied convolutional neural networks (CNNs) used for visual object recognition. The intuition behind this choice is simple: a neural network trained to identify objects will have learned to identify important high-level features that define an image's content independently of low-level features such as color and lighting. Beginning with a random white-noise image, they perform gradient descent and modify the image until the CNN response of the white-noise image matches the response of content image c.

On top of this response from the CNN, they build a style representation that effectively captures the function L_{style}. They use a Gram matrix to capture the style of an image, and minimize the mean-squared distance between the Gram matrix of the original style image and the Gram matrix of the image to be generated.

Thus, to generate an image that captures the content of one image and the style of another, they jointly minimize the distance of a white noise image from the content and style representations in the layers of the CNN.

We aimed to apply the ideas of this method to conversation, changing the style and 'personality' of a sentence while staying true to its original content. An important distinction to keep in mind is that we are not necessarily capturing an identity, like Nguyen et al. (2017), but rather focusing solely on the style of speech.

3 Approach

3.1 Data

For this project we made use of Project Gutenberg's database of literary transcripts. We selected books by a number of different authors, and ran some preliminary processing. We parsed the transcript line-by-line, replaced each word with the id of its embedding, and filtered out the shortest sentences. Each line was then associated with an ID corresponding to the author that had written it.

3.2 Model

3.2.1 Text Classification

Similarly to how the visual artistic style transfer method begins with a trained network for object identification, we had to begin with a network capable of author identification.

The author identification network was a simple Gated Recurrent Unit (GRU) network, each state of which took a word from the provided transcript phrase as input. We fed the final hidden state as input to a linear classifier, which classified the text by author.

We modified the text classifier to accept either word vectors or word ID's as input, since processing word vectors proved essential to training the style transfer mechanism.

3.2.2 Seq2Seq

With our text classifier complete, our next step was to create a basic Seq2Seq text generator. This we broke down into its two primary components: the encoder and the decoder. Conceptually, we needed the encoder to be able to take an input string and capture its *content* in its encoding. The decoder, on the other hand, needed to be trained to take an encoding of a phrase's content and be able to write the phrase in the specified *style* while preserving meaning. It's similar in concept to language translation, where the new language is merely a different manner of speaking.

Figure 1: You can see the results of artistic style transfer being applied to the picture A. The painting providing the style is shown in the bottom left of each panel. Painting B is 'The Shipwreck of the Minotaur' by J.M.W. Turner; C is 'The Starry Night' by Vincent van Gogh; D is 'Der Schrei' by Edvard Munch. (Gatys et al., 2015)

We now proceed to define a loss function that allows the Seq2Seq model the ability to capture and maintain sentence content, which we do by scoring sentence similarity; we need to ensure that the decoder could output different words than the original encoded phrase had input, so long as the meaning of the output phrase was similar enough. We do this by first mapping the decoder output to a vector with the same dimensions as our given word vectors, and then averaging consecutive word vectors in both the input sequence, as well as our synthetic output, using average pooling, and then comparing the output sentence to the input - specifically, we generate the cosine distances of all pooled parts of the sequences, and use it as our *content loss*.

We choose this method of decoding so that, at test time, we can generate the final output sequence by mapping each decoded vector to its closest word vector in the vocabulary. It also allows us to back-propogate the gradient of the content loss into the Seq2Seq model directly, allowing the model to build an understanding of word meaning.

3.2.3 Style Transfer

With the basic Seq2Seq model running, the final step was to apply the style transfer. First, we implemented a method of rating style. For the desired style archetype, we get the average text classifier final state for the author whose style we are trying to emulate, over all of their data. We also run our generated sentence through the text classifier, and we apply a cost to the result relative to how different the hidden states are from one another.

We also experimented with attempting to emulate the style by modifying the output phrase until it was classified under the desired author, rather than comparing the final hidden state to the au-

thor's average hidden state. However, this led to some rather interesting overfitting; in general, the decoder became fixated on a few of the author's keywords. Without named-entity filtering, these were often character names, leading the decoder to swap out the names of entities to match those within the book written by the desired author. While entity replacement is interesting, this wasn't the result we were looking for, so we returned to using the comparison to the average text classifier final state.

We compute our final total loss using the following function:

$$Loss = \alpha l_c + \beta l_s$$

where c_l is the content loss as defined earlier, and l_s is the style loss defined in this section. By optimizing on both of these, we build a system for a given author that can transform input sentences into the desired style.

4 Experiments

As discussed earlier, devising a loss function that allows for the training of the Seq2Seq model on both losses required experimentation - in our initial system, the decoder output mapped to a vocabulary dimension vector using an affine transform ($y = xW + b$), and we decoded our result by taking the index in the vocabulary with the highest value. However, we quickly discovered that this prevented easy propogation of style loss back into the model - since we would have to evaluate the style score on the specific sequence, we would only propogate loss into the chosen words within the vocabulary.

By switching to a model where we instead produce word vector-like outputs, which are decoded using cosine distance to the actual pre-computed word vectors, we can feed the generated vectors into our style loss computation, and thus train the weights that compute the output sequence directly.

We also tried multiple ways of computing style loss - in one experiment, we used softmax cross entropy loss of the desired author class and the generated score vector for our synthetic sequence. We found, however, that the model, instead of copying the style of the author, simply tried to fool the model by using words with high frequency in the original texts. By instead comparing the feature vectors before the score classification, we achieved better results. While our initial method failed to work, it should be noted that using an adversarial network to penalize the overfit behavior of our model could have alleviated this issue, but more experiments are required to determine this.

5 Conclusion

5.1 Future Steps

Artistic style transfer in natural language has potential, but there are a lot of steps that could be taken that would likely make for great improvements.

First, regarding the problem of named entities: in speeches and in novels, named entities such as people, locations, or unique objects make it too easy for the classifier to fit to specific words as opposed to the written style. In our experiments, we allowed words not within the GLoVe dataset to be appended onto the vocabulary when they were encountered in the text, which included names. However, disabling this functionality and instead allowing these terms to be tagged unknown might improve the style mimicry overall.

Next, regarding the looseness of the content cost evaluation: we think there are a few alternatives to our simple average pooling that could prove more effective at preserving meaning in the encoding-decoding process. For example, using convolutions over the input and output phrases might be an improvement, as it could take into account sentence structure while still allowing some flexibility in word ordering. However, there haven't been any significant problems with average pooling thus far, simple as it is, so this step is lower priority than fixing the named-entity problem.

Finally, regarding data. Our experiments were run on five different authors, with one book by each. Getting different pieces of writing would probably improve results, especially if the books are separate stories from one another as this would help to separate the author's style from their stories' content.

5.2 Final Words

Style transfer in natural language is difficult and fascinating, a discrete problem without a concrete solution. With more work and development, language style transfer could be applied to problems such as language modernization - for example, 'translating' Shakespeare for easier understanding.

6 References

References

Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. 2015. A neural algorithm of artistic style. *CoRR* abs/1508.06576. http://arxiv.org/abs/1508.06576.

Harish Narayanan. ???? Convolutional neural networks for artistic style transfer. https://harishnarayanan.org/writing/artistic-style-transfer/.

Huyen Nguyen, David Morales, and Tessera Chin. 2017. A neural chatbot with personality .

Sean Stanko, Devin Lu, and Irving Hsu. 2013. Whose book is it anyway? using machine learning to identify the author of unknown texts .

Yizhe Zhang, Zhe Gan, and Lawrence Carin. 2016. Generating text via adversarial training .