PROJECT REPORT
STYLE TRANSFER IN TEXT: EVALUATION AND EXPLORATION

April 29, 2019

Harshita Agrawal (2018201014)
Supriya Priyadarshani (2018202009)
Rajat Kumar Verma (2018201040)
Anjul Gupta (2018201021)

Contents

1 Problem Description

Using massive amounts of parallel data has been essential for recent advances in text generation tasks, such as machine translation and summarizing. However, in many text generation problems, we can only assume access to non-parallel or mono-lingual data. Problems such as decipherment or style transfer are all instances of this family of tasks. In all of these problems, we must preserve the content of the source sentence but render the sentence consistent with desired presentation constraints.
The goal of controlling one aspect of a sentence such as style independently of its content requires that we can disentangle the two. However, these aspects interact in subtle ways in natural language sentences, and we can succeed in this task only approximately even in the case of parallel data. The paper proposes two models to test achieve the goal of style transfer.The goal of controlling one aspect of a sentence such as style independently of its content requires that we can disentangle the two. However, these aspects interact in subtle ways in natural language sentences, and we can succeed in this task only approximately even in the case of parallel data. The paper proposes two evaluation metrics that measure the two aspects of style transfer: transfer strength and content preservation.

2 Paper Overview

Style transfer in an important problem in the field of NLP as it can automate conversion of paper title to news title, which reduces the human effort in academic new report. The progress in style transfer of language is lagged behind other domains such as computer vision, largely because of the lack of parallel corpus and reliable evaluation metrics. In this paper two models for style transfer are proposed to approach the following problems.

- lacking parallel training data
- hard to separate style from the content

The models achieve goal by multi-task learning and adversarial training of deep network.

2.1 Models Overview

- **Auto-encoder Model:** Auto-encoder seq-seq is used as the base model since minimum changes are expected from input to output. In the auto-encoder seq2seq model, an encoder is learned to generate intermediate representation of input sequence $X = (x_1, .., x_{T_x})$ of length T_x Then a decoder is used to recover the input X using intermediate representation.

- **Multi-decoder Model:** Implement a multi-decoder seq2seq where the encoder is used to capture the content c of the input X, and the multi-decoder contains $n(n \geq 2)$ decoders to generate outputs in different styles.
- **Style Embed Model:** Use the same encoding strategy, but introduces style embeddings that are jointly trained with the model. The style embeddings are used to augment the encoded representations, so that only one decoder needs to be learned to generate outputs in different styles

2.2 Evaluation Metrics

Considering the problem of lacking principle evaluation metrics, the paper proposes two novel evaluation metrics that measure two aspects of style transfer:

- Transfer Strength
- Content Preservation

Results show that the proposed content preservation metric is highly correlate to human judgments, and the proposed models are able to generate sentences with similar content preservation score but higher style transfer strength comparing to autoencoder.

2.3 Related Work

1. **Style Transfer in NLP**
    - **Mueller, Gifford, and Jaakkola (2017)** proposed a variational auto-encoder (VAE) based model to revise a new sequence to improve its associated outcome. However, there is no significative evaluation for style transfer. It uses nonparallel data.
    - **Shen et al. (2017)** explored style transfer for sentiment modification, decipherment of word substitution ciphers and recovery of word order. They used VAE as the base model and used an adversarial network to align different styles. However, their evaluation only considered the classification accuracy.

2. **Adversarial Network for Domain Seperation**

   - **(Ganin and Lempitsky 2015)** proposed deep domain adaptation approach to encourage domain-invariant features. This model can be trained on labeled source domain data and unlabeled target domain data.
   - **(Bousmalis et al. 2016)** used adversarial networks to learned shared representations between two domains which do not contain the individual features of each domain.

   The major difference between the below mentioned work and this paper is that they do not need to generate new sentences. How adversarial networks work on controlled generation is largely untested.

3. **Related Evaluation Metrics**

   - **BLEU (Bilingual Evaluation Understudy)** outputs a number between 0 and 1 that indicates how similar the candidate text is to the reference texts, with values closer to 1 representing more similar texts. BLEU was one of the first metrics to claim a high correlation with human judgements of quality,and remains one of the most popular automated and inexpensive metrics. It has been argued that although BLEU has significant advantages, there is no guarantee that an increase in BLEU score is an indicator of improved translation quality.
   - **ROGUE (Recall-Oriented Understudy for Gisting Evaluation)** includes measures to automatically determine the quality of a summary by comparing it to other (ideal) summaries created by humans. The measures count the number of overlapping units such as n-gram, word sequences, and word pairs between the computer-generated summary to be evaluated and the ideal summaries created by humans.

3 Problem Design

Sequence to sequence (seq2seq) neural network models have demonstrated great success in many generation tasks, such as machine translation, dialog system and image caption, because parallel data is available for those tasks. However, it is hard to get parallel data for tasks such as style transfer. Also, there is one more challenge, that is separating style from its content. So we presented two models that

can work with non parallel data by separating content and style using adversarial networks. We also have presented two evaluation metrics based on transfer strength and content preservation Earlier evaluation metrics such as BLEU was used (which worked with parallel data).

Initially the Auto-encoder Model was designed which serves as the base model since we expect the minimum changes from input to the output. Then the other two models were implemented. The models were then tested (after training) , and then evaluated based on the two evaluation metrics.

4 Models

4.1 Model Auto-encoder

A GRU Autoencoder is an implementation of an autoencoder for sequence data using an Encoder-Decoder GRU architecture. They are an unsupervised learning method, although technically, they are trained using supervised learning methods, referred to as self-supervised. GRU's are improved version of standard recurrent neural network. It contains two gates - Update Gate and Reset Gate. The update gate helps the model to determine how much of the past information (from previous time steps) needs to be passed along to the future. A GRU unit is composed of following things

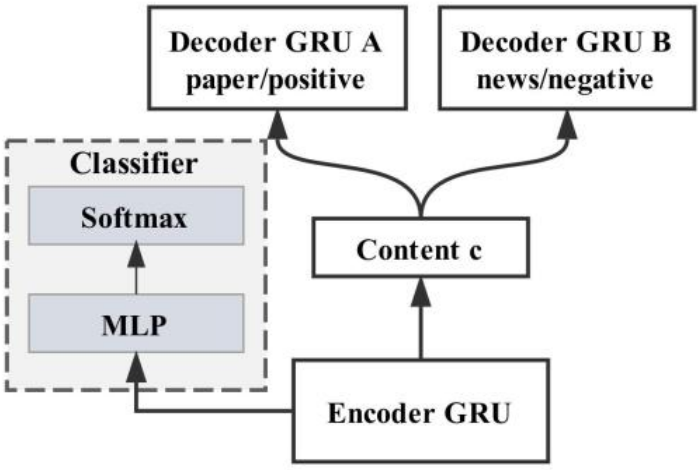
$$
\begin{aligned}
\mathbf{s}_j &= \mathbf{z}_j \odot \mathbf{h}_j + (1 - \mathbf{z}_j) \odot \mathbf{s}_{j-1}, \\
\mathbf{h}_j &= \tanh\left(\mathbf{W}\mathbf{E}[x_{j-1}] + \mathbf{r}_j \odot (\mathbf{U}\mathbf{s}_{j-1})\right), \\
\mathbf{r}_j &= \sigma\left(\mathbf{W}_r\mathbf{E}[x_{j-1}] + \mathbf{U}_r\mathbf{s}_{j-1}\right), \\
\mathbf{z}_j &= \sigma\left(\mathbf{W}_z\mathbf{E}[x_{j-1}] + \mathbf{U}_z\mathbf{s}_{j-1}\right),
\end{aligned}
$$

where s_j is the activation unit of GRU at time j, h_j is an intermediate state that computes the candidate activation, r_j is a reset gate that controls how much to reset from the previous activation for the candidate activation and z_j is the update gate that controls how much to update the current activation based on the previous activation and candidate activation. E is a word embedding matrix that is used to convert the input words to vector representations. $E, W, U, W_r, U_r, W_z, U_z$ are the model parameters.

4.2 Multi-decoder Model

The multi-decoder model for style transfer is similar to an auto-encoder with several decoders, with the exception that the encoder now tries to learn some content representa-

tions that do not reflect styles. The style specific decoders (one for each style) then take the content representations and generate texts in different styles. The main challenge is to separate content representation from the style.



We use a similar adversarial network to separate the content representation c from the style. The adversarial network is composed of two parts. The first part aims at classifying the style of x given the representation learned by the encoder. The loss function minimizes the negative log probability of the style labels in the training data

$$L_{adv1}\left(\Theta_c\right) = -\sum_{i=1}^{M} \log\, p\left(l_i | Encoder(\mathbf{x}_i; \Theta_e); \Theta_c\right),$$

The second part of the adversarial network aims at making the classifier unable to identify the style of x by maximizing the entropy (minimize the negative entropy) of the predicted style labels.

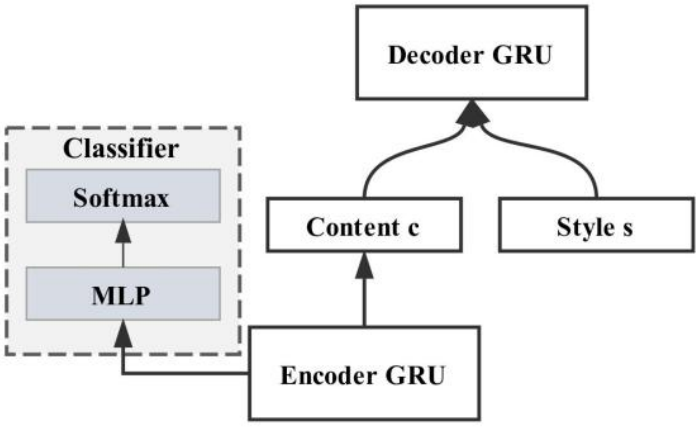
$$L_{adv2}\left(\Theta_e\right) = -\sum_{i=1}^{M}\sum_{j=1}^{N} H\left(p\left(j | Encoder(\mathbf{x}_i; \Theta_e); \Theta_c\right)\right)$$

While the encoder is trained to produce content representations, the multiple decoders are trained to take the representations produced by the encoder and generate outputs in different styles.

4.3 Style-embedding Model

Our second model uses style embeddings to control the generated styles. In this model, the encoder and the adversarial network parts are the same as the multi-decoder model,

to generate content representations c. In addition, style embeddings are introduced to represent the styles. A single decoder is trained in this model, which takes the concatenation of the content representation c and the style embedding e of a sentence as the input to generate texts in different styles.



The loss function of style-embedding is defined below where L_{gen2} is the loss function for the seq2seq generation. The only difference is that it also contains the parameter E for style embeddings, that are jointly trained with the modelThe total loss is similar to the multi-decoder model, where L_{adv1} and L_{adv2} are the same equations mentioned in multi-decoder model.

$$L_{total2}(\theta_e, \theta_d, E) = L_{gen2}(\theta_e, \theta_d, E) + L_{adv1}(\theta_c + L_{adv2}(\theta_e)$$

5 Evaluation Metrics

5.1 Transfer Strength

The main task of transfer strength model is to transfer source style to target style, so transfer strength evaluates whether the style is transferred. We use LSTM-sigmoid classification. Transfer strength accuracy is defined as N_{right}/N_{total}, where N-total is the number of test data and N-right is the number of correct case which is transferred to target style.

5.2 Content Preservation

But, it is easy to train a model that has 100 percent transfer strength by only generating the target style words. Therefore, we propose a metric for content preservation, which can evaluate the similarity between source text and target text. Content preservation

rate is defined as cosine distance between source sentence embedding and target sentence embedding.

Results show that the proposed content preservation metric is highly correlated to human judgments, and the proposed models are able to generate sentences with similar content preservation score but higher style transfer strength comparing to auto-encoder.

$$
\begin{aligned}
v_{min}[i] &= \min\{w_1[i], \dots w_n[i]\} \\
v_{mean}[i] &= \text{mean}\{w_1[i], \dots w_n[i]\} \\
v_{max}[i] &= \max\{w_1[i], \dots w_n[i]\} \\
v &= [v_{min}, v_{mean}, v_{max}] \\
score &= \frac{v_s^\top v_t}{\|v_s\| \cdot \|v_t\|} \\
score_{total} &= \sum_{i=1}^{M_{test}} score_i
\end{aligned}
$$

$$
l_{style} = \begin{cases} paper(positive) & output \leq 0.5 \\ news(negative) & output > 0.5 \end{cases}
$$

Although a single integrated metric that combines transfer strength and content preservation as F1 score seems plausible to measure the performance of the systems, it is not the best for style transfer, since sometimes the transfer strength is more important, while in other cases the content preservation is the focus. A weighted integration would be ideal for different scenarios.

6 Feature Engineering

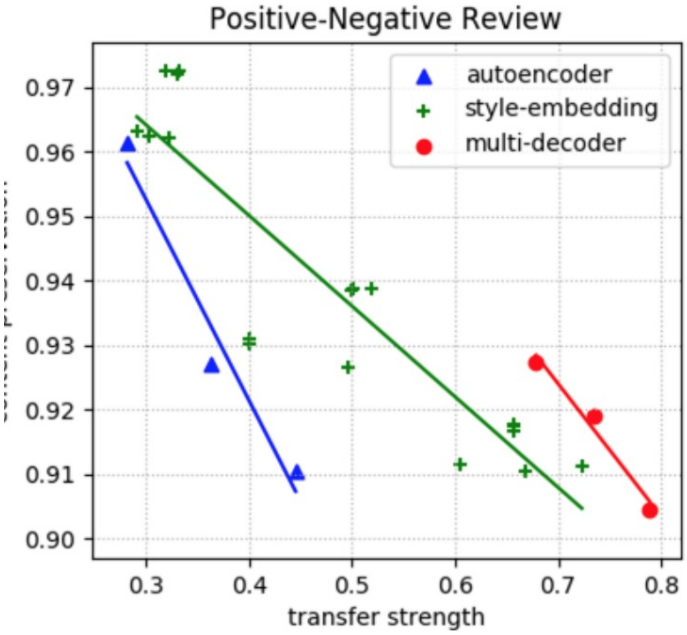
1. **Implemented**
   - Extended positive-negative review dataset on amazon, twitter, yelp and imdb reviews.
   - Prepared a dataset for paper-news title.

2. **Proposed**

- The transfer of style from one author's novel to some other.
- A weighted integrated metric that combines transfer strength and content preservation as F1 score to measure the performance of the systems, since sometimes the transfer strength is more important, while in other cases the content preservation is the focus.

7 Observation

- Reversing the sentences even though did not get us to highest accuracy. - Fixed number of epochs to 300 to make sure network plateau, we avoided early stopping since gradient descent guarantees local minima not global and gets stuck. - Since LSTMs tend to not suffer from the vanishing gradient problem, they can have exploding gradients, so we have used GRU in place of LSTM.



```
 python3 eval.py
dir_name model_type      transfer_strength       content_reservation     mixture
================================================================================
test1    embedding8      0.2765                  0.9438803062989316      0.21385374980619115
test1    embedding6      0.6055                  0.8960117473061905      0.36132591967211813
test1    embedding1      0.4765                  0.9156584887943305      0.31340632077628094
test1    embedding7      0.4905                  0.9145880600519364      0.3192721198121155
test1    embedding       0.605           0.8965986599545084      0.3612431229053648
test1    embedding2      0.289           0.9434864821132846      0.22123373950780323
test1    embedding5      0.2815                  0.9440204290794388      0.21683991917252363
test1    embedding4      0.493           0.9153460001569551      0.32042238059900546
test1    embedding3      0.609           0.8956981594839047      0.3625180078061582
================================================================================
```

For auto-encoder, two tasks share all the parameters, so it does not have the ability to generate different style sequence. For style-embedding, two tasks share encoder and decoder with separate style embedding, so it has weak ability to generate different style sequence. For multi-decoder, two tasks share encoder with two separate decoders, so it shows high ability to generate different style sequence.

References

https://github.com/fuzhenxin/textstyletransferdata