Sentiment Transfer using Seq2Seq Autoencoders

Ayush Singh, Ritu Palod
{singh.ay, palod.r}@husky.neu.edu
CSYE7245 Fall 2017, Northeastern University

Abstract

Expressing in language is subjective. Everyone has a different style of reading and writing, apparently it all boil downs to the way their mind understands things (in a specific format). Language style transfer is a way to preserve the meaning of a text and change the way it is expressed. Progress in language style transfer is lagged behind other domains, such as computer vision, mainly because of the lack of parallel data, use cases, and reliable evaluation metrics. In response to the challenge of lacking parallel data, we explore learning style transfer from non-parallel data. We propose a model combining seq2seq, autoencoders, and adversarial loss to achieve this goal. The key idea behind the proposed models is to learn separate content representations and style representations using adversarial networks. Considering the problem of evaluating style transfer tasks, we frame the problem as sentiment transfer and evaluation using a sentiment classifier to calculate how many sentiments was the model able to transfer. We report our results on several kinds of models.

Related Work

Style Transfer in Computer Vision

Style transfer has made significant progress in computer vision in recent years. (Gatys et al., 2016) separated the content and style of images to recombine them generating new images using a linear model to change the color of the pictures. Their methods use only one image to represent a style. However, it does not work for text because a single sentence or short article does not store enough style information.

(Zhu et al., 2017) proposes CycleGAN to do image-image translation. It firstly learns a mapping $G : X \rightarrow Y$ using an adversarial loss, and then a reverse mapping $F : Y \rightarrow X$ with a cycle loss $F(G(X)) \approx X$ which performs unpaired image to image translation. CycleGAN shows qualitative results, nevertheless, cycle training is hard to implement with discrete text.

Style Transfer in Natural Language Processing

(Jhamtani et al., 2017) explores automatic methods to transform text from modern English to Shakespearean English using parallel data. The model was based on seq2seq and enriched it with pointer network (Vinyals et al., 2015), however, paired word dictionary is a scarce resource that does not exist in most style transfer tasks, and it required parallel corpora.

(Fu et al., 2017) proposed a variational auto-encoder (VAE) based model to revise a new sequence to improve its associated outcome. However, there is no significant evaluation for style transfer. It uses nonparallel data. (Shen et al., 2017) explored style transfer for sentiment modification, decipherer word substitution ciphers and recovery of word or-

der. They used VAE as the base model and used an adversarial network to align different styles. However, their evaluation only considered the classification accuracy.

There are similar supervised exploratory works by (Ficler and Goldberg, 2017) experiment with controlling several stylistic aspects of the generated text, in addition to its content. The method is based on conditioned recurrent neural networks (CRNN) language model, where the desired content as well as the stylistic parameters serve as conditioning contexts but use hand labeled features and is a supervised task. (Li et al., 2016) encodes personas in distributed embeddings that capture individual characteristics such as background information and speaking style. But again, they use twitter conversations (parallel text) to learn responses. These works are not directly applicable in our case since we want to learn the style and content in an unsupervised manner and using non-parallel text.

Dataset

We are using the Yelp dataset.

1. A large-scale dataset (4.7 million reviews) suitable for effectively training neural networks.

2. Crowd-sourced to collect the natural language reviews written by human beings from over four continents, that avoids over-fitting and improves generalization.

3. Text reviews are correlated by the stars given by a user to a business location, which is perfect for our case since we can separate styles based on sentiment of the text.

4. To make data suitable for sentiment analysis, we used all reviews with stars more than 3 as *positive*, less than 3 as *negative* and equal to 3 as *neutral*.
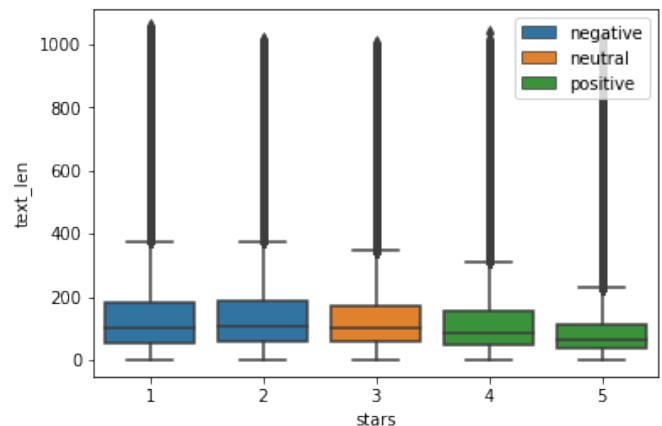

Figure 1: Distribution of lengths of different sentiments of reviews

Preprocessing

Original dataset came serialized in *json* which we stored in MongoDB and then performed all our exploratory data analysis from database querying and used keras to create our data preprocessing pipeline from loading text to creating vocabulary, encoding into word index sequences and finally padding to fixed length dynamically when creating batches all comes as a prerequisite to feeding data to neural networks.
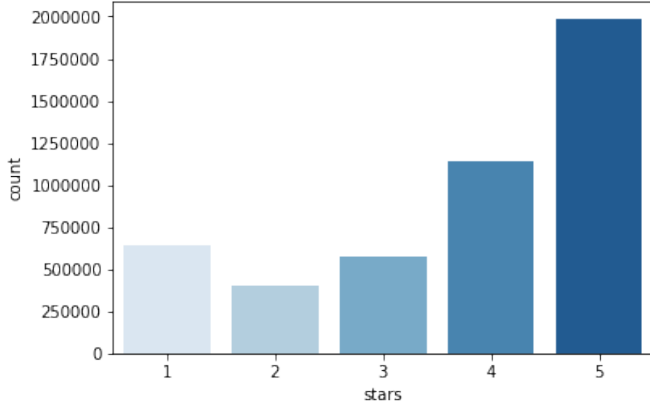

Figure 2: Plot of class imbalance in the dataset

1. Converted all characters to lower cases.

2. Replaced all the numbers with a special string _NUM_.

3. Remove punctuations and special symbols.

4. Remove Neutral reviews (Stars = 3)

5. Maximum Sentence Length 10 words as we are more interested in tips style of reviews e.g. "good Italian food" than longer descriptive ones.

6. Only keep 10k most frequent words in vocabulary and then split the dataset in positive and negative so that both of them share the same vocabulary.

7. The dataset had a class imbalance where positive reviews were 3M vs 1M negative reviews. After preprocessing we were left with 600k reviews, we then handled the class imbalance splitting almost equally.

8. The data is split so that the training (80%), dev (10%), and test (10%).

Model

We propose seq2seq autoencoder model and try to train the same model adversarially to separate style from content for style transfer in this paper and evaluate it on various configurations. We try to maximize learning the style of a text and keep the vocabulary consistent among the different styles we want to transfer in. The common ground of the two models is to learn a representation for the input sentence that only contains the content information. Figure 1 illustrates the two models. We give more details about each model in the following sections.

Background: Autoencoders

(Rumelhart et al., 1985) proposed autoencoders to learn lossy abstract representations of higher dimensional data and were later found to efficient in compressing data. It was mainly used for dimension reduction in the past, but more recently, the concepts have been widely used for generative models. They consist of an encoder that encodes information in a latent space followed by a decoder that tries to regenerate original data using latent space. The learning takes place using gradient descent and backpropagation over a certain number of epochs until network reaches a minimal loss.
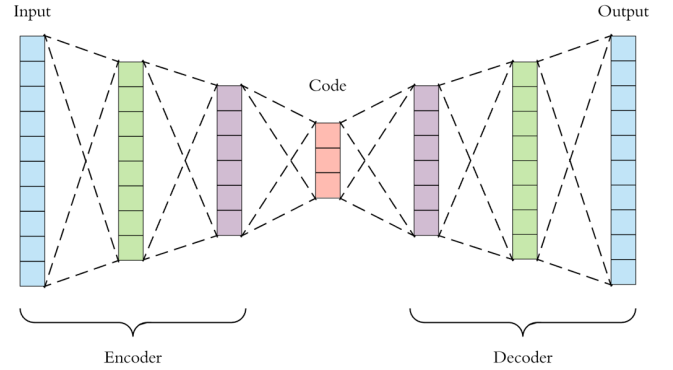

Figure 3: Network architecture of Autoencoder

Background: Word2Vec

Word2Vec are a type of word embeddings by (Mikolov et al., 2013) that are used to learn dense representations of distributional and semantic context of words in corpus but it's use is not only limited to text. The model borrows from the saying *A word is identified by the company it keeps*, and tries to predict surrounding words of a word (CBOW) or inverse where it tries to predict center word from surrounding words (skip-gram) using a multilayer perceptron.
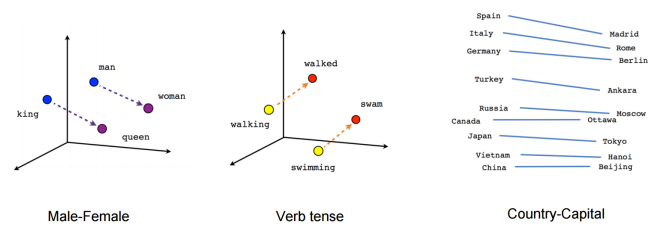

Figure 4: Word Vectors latent relations

Background: Seq2Seq

Seq2Seq proposed by (Sutskever et al., 2014) has a similar architecture of autoencoder with the difference that source and target are different. Seq2Seq is used in state of the art neural machine translation systems where the encoder is trained to encode sequential data in source language while decoder is trained to predict target language itself one-time step ahead.
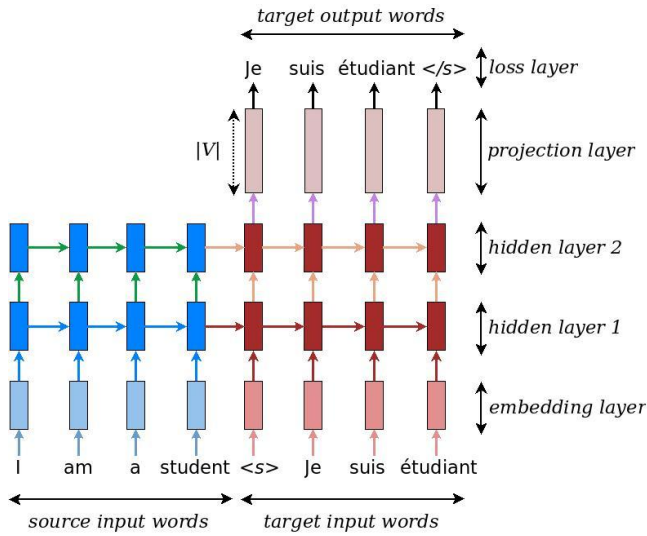
Figure 5: Seq2Seq encoder-decoder framework

The way translation becomes from one language to another becomes possible is that the last state of the encoder is used as the initial state of the decoder which gives it enough context to translate.
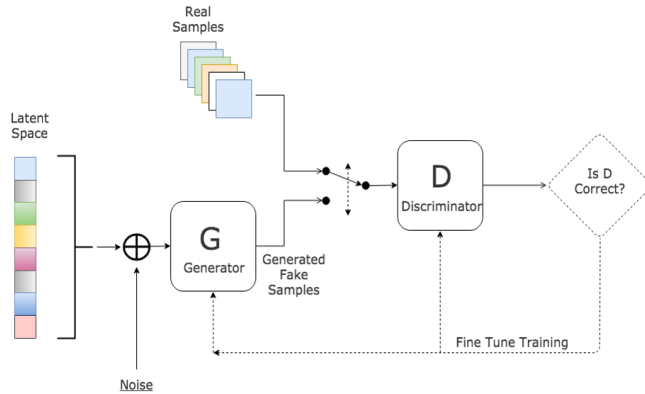
Generative Adversarial Networks

Figure 6: GAN: generator-discriminator network

The main idea behind a GAN (Goodfellow et al., 2014) is to have two competing neural network models. One takes noise as input and generates samples (and so is called the generator). The other model (called the discriminator) receives samples from both the generator and the training data, and has to be able to distinguish between the two sources. These two networks play a continuous game, where the generator is learning to produce more and more realistic samples, and the discriminator is learning to get better and better at distinguishing generated data from real data. These two networks are trained simultaneously, and the hope is that the competition will drive the generated samples to be indistinguishable from real data.

Baseline: Seq2Seq Autoencoder Model

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 trained to recover the input X using the intermediate representation. For the style transfer problem, we use the auto-encoder seq2seq model as our base model, since we expect minimum changes from the input to the output. Our intuition is to first learn embeddings of words in a sequence and then encode sequential structure using recurrent neural networks and finally use autoencoders to learn higher representations of our data. Unlike traditional seq2seq we let our source and target be exactly the same based on the intuition that when the model is fully trained and is used for inference, the encoder would encode a different style sentence in the style encoder was trained on and the same with decoder transferring the sentiment effectively. Recall that we are assuming our vocabulary stays the same, we found our intuition to be on the right track from experimental results.

The drawback of this model is that the autoencoder will capture both the style and the content representation which is good if the style we are transferring into also shares the same context e.g. Sentiments where both positive and negative would have the same vocabulary and review context but not generalizable enough for e.g. in case of news to informal style.

Reduced Seq2Seq Model

Another one of our intuitions is that if we let the model see a large vocabulary which it will from the whole dataset, it will learn highly variant style in our case both positive and negative. This would not be ideal for our case since we do not want to build a language model rather a style and content model.

In our experiments, we found that limiting the vocabulary works best for this seq2seq autoencoder, even outperforming full dataset model.

Reversed Seq2Seq Model

(Sutskever et al., 2014) found that if we only reverse the source sequence and let target be intact, it reduces the distance between translation words and hence increase memory ability on long sentences while reducing computation time. Reversed model is relevant when using unidirectional LSTMs since bidirectional would automatically consider both directions of a sequence.

In our experiments, we did not find reversing technique to be any superior, and would like to point that it is case specific.

Bidirectional Seq2Seq Model

Traditional LSTM are unidirectional which means they take only the word appearing after them into context. Adding another layer in parallel that takes the previous word into context and merging/concatenating/averaging both layers allows the model to take both neighbor words into context and is found to be outperforming unidirectional LSTMs.
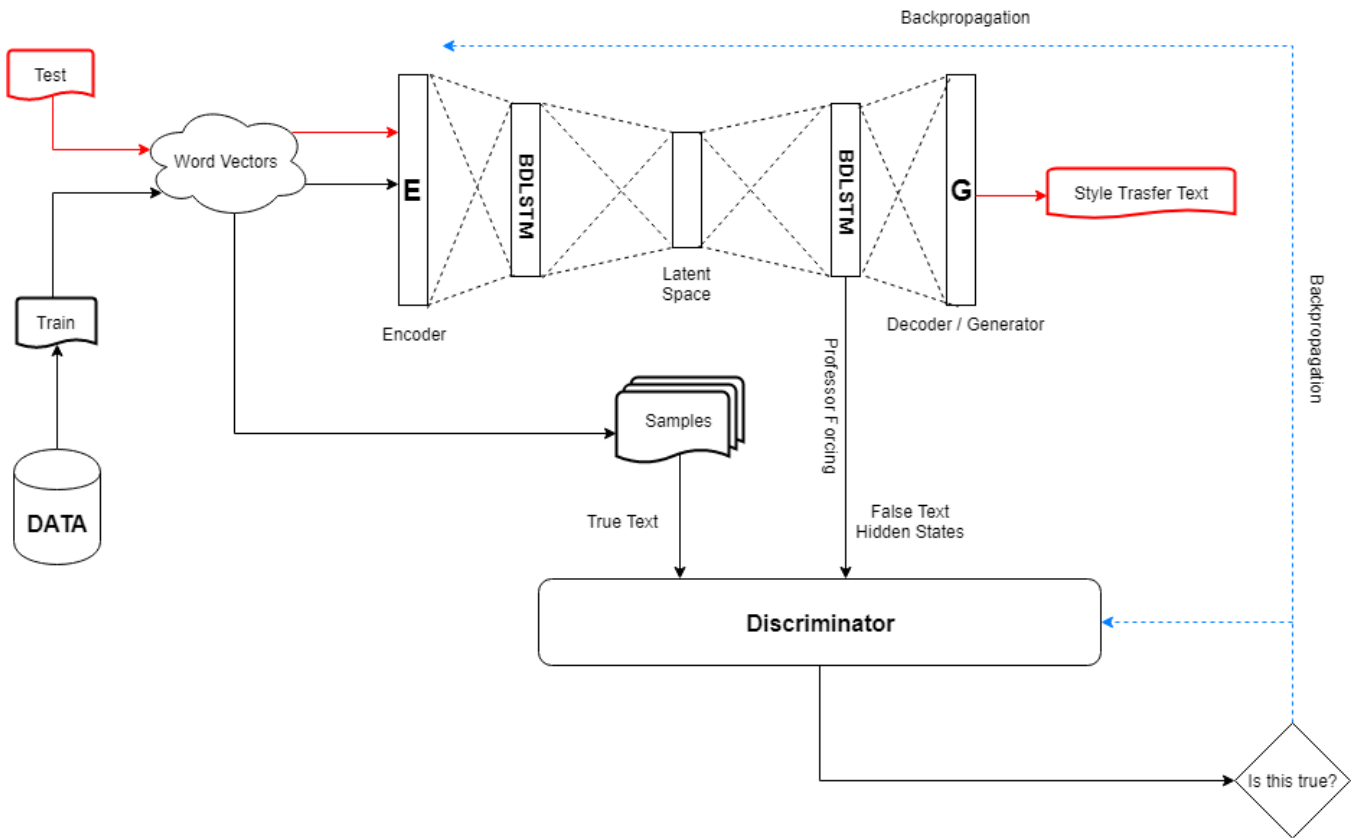
Figure 7: Adversarial Seq2Seq Autoencoder Network

In our experiments, we found that not only did bidirectional networks plateaued faster but also gave the best accuracy when used in our evaluation model (sentiment classifier) but for our style transfer the bidirectional overtrained itself and was even able to predict exact match unseen data.

Reduced Bidirectional Seq2Seq Model

After seeing that reduced vocabulary model actually works to our advantage, we tried the same approach of reduced vocabulary with bidirectional LSTMs.

Adversarial Seq2Seq Autoencoder Model

To separate content from the style we train the seq2seq autoencoder with an adversarial loss where we improve the latent semantic space of the autoencoder to capture a style by training the discriminator to learn to differentiate between a real style using decoder part of autoencoder and a randomly generated fake style. Since the text decoder outputs discreet values, differentiating on them to back propagate errors would be impossible so we use the last hidden layer of the decoder as input to discriminator and compare that to fake text generated randomly which allows the discriminator to better distinguish original style from the rest. This network although looks promising on paper was found to be slow to train, given the time constraints we prioritized experimenting more with Seq2Seq Autoencoder.

Evaluation

Evaluation plays an important role in style transfer as they provide criteria to compare different models. Automatic evaluation metrics speed up development. In this paper, we will focus on showcasing results of negative to positive transfer for succinctness, in experiments we have found the performance of from positive to negative equal. We divide our evaluation into two types: machine and human evaluation comprising of

1. Soundness (generated texts being textually entailed with original version)

2. Coherence (free of grammatical errors, proper word usage, etc.)

3. Effectiveness (the generated texts actually match the desired style)

For machine evaluation, BLEU (Papineni et al., 2002) is a popular evaluation metric in neural machine translation and ROUGE (Lin, 2005) is popular in text summarization. But since both of them rely on human-generated text, which in our case is style which can be different for different persons. ROUGE-2 compares percentage bi-gram overlaps between human generated and system generated responses which might be good but in case of sentiment transfer we can simply use state of art classifier to detect whether the transferred sentence successfully transferred the sentiment.
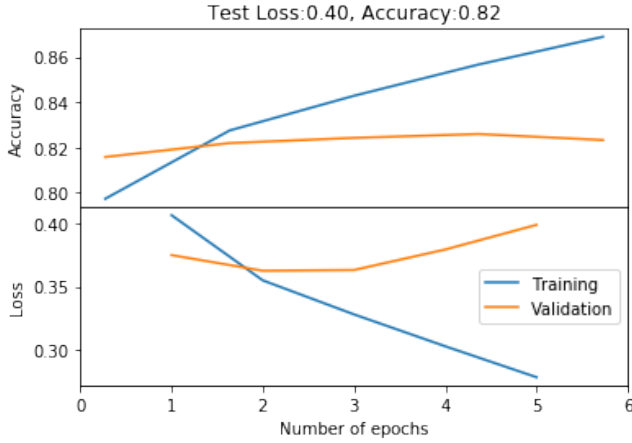
Figure 8: Accuracy & loss of Sentiment Classifier

Our sentiment classifier is a bidirectional dual layer 1024 units LSTM (dropout 0.2) with a multi-layer perceptron on top trained on 2M equally distributed positive and negative reviews which has an accuracy of 82.6% on unseen data in just 5 epochs. The train and validation loss intersected in just 2 epochs but we let the network runs for 3 more epochs just to be sure.

Experiments

Note that for the purpose of easy evaluation we are framing our problem in terms of sentiment transfer but our training methods are agnostic of the style and are end to end fully unsupervised.

Model

Before we implement the above model, as a part of understanding of how the basic components work we also implemented the following:

1. Autoencoders for MNIST dataset and text data where we realized that in case of text we need some network to take the sequential nature of text into account. This is how we developed an intuition of using Seq2Seq Autoencoders.

2. Word2Vec using noise contrastive estimation to learn word embeddings but quickly realized that in case of non-parallel data we would not have the luxury to pre-train word vectors, so we implement on the fly training of embeddings.

3. Seq2Seq inspired from Google Neural Machine Translation system which includes some of the state of the art techniques like gradient clipping and source sentence reversing.

4. Generative Adversarial Networks on MNIST dataset where we learned how tricky it is to tune GANs as we experience a common phenomenon called *mode collapse* ourselves where generator starts generating only one type of distribution which it thinks is the best way to fool discriminator. To alleviate the problem we used mini-batching to let discriminator see a batch of examples instead of one.

Results

Seq2Seq Models	Loss	Accuracy %
Vanilla	3.12	41.70
Reduced	2.12	66.72
Reversed	1.71	59.41
Bidirectional	2.86	45.46
Reduced Bidirectional	**1.02**	**66.91**

Table 1: Accuracy of Different Seq2Seq Models

Evaluating results of all the models using the sentiment classifier we mentioned in the evaluation section, we found *reduced bidirectional* to be the best model for transferring sentiments. Along with that, we have the following observations:

1. Reducing vocabulary not only increased vocabulary and reduced loss but also trained much much faster as this only used 20k records. Adding bidirectional on top reduced the loss by 100% which makes reduced bidirectional our best model for sentiment transfer.

2. Reversing the sentences even though did not get us to highest accuracy but if you notice the loss, it not too far behind. This is a completely new finding that we never saw mentioned in literature, we corroborate the findings of (Sutskever et al., 2014) in the case of Seq2Seq autoencoders as well.

3. Using bidirectional is found record-setting good in general but in case of autoencoders, it is good if the only goal is to encode representation, not really style transfer since it is so good that it overfits the text capturing both content and style in the semantic space which does not serve well for transferring style.

4. Adversarially training the network takes a lot of iterations (10^5) and low learning rate (10^{-4}), we trained it on a small corpus and got results that verified the architecture is right but to get sensible style transfer meant training on the big dataset which took a lot of time. We think freezing the discriminator layers and training generator on a larger batch would increase learning time. We are keeping that as the future scope.

5. Scope Update: Initially we wanted to transfer in Wikipedia style but given that difference between a sample text and it's Wikipedia counterpart made evaluation tricky, which is why we switched to sentiment transfer on Yelp dataset because the difference in two reviews of 1 vs 5 ratings is stark and can be evaluated easily. Usage of the model for transferring in Wikipedia style should be approached once the sentiment transfer model show promises and the evaluation scheme is robust enough.

6. Coming up with evaluation scheme using a neural classifier fit perfectly in our case when we switched to sentiment transfer.

Training Speed and Performance

We noticed a general trend in networks, the higher the number of parameters to train the longer each epoch took. Deepening the networks slowed the training time the most while not really adding much to accuracy score whereas widening the network not only kept the training time from slowly drastically but even added 3 points to accuracy pushing it to 99.83 on train set%.

Lastly, we want to notify that deep learning is compute intensive, so we kept our hyperparameters in a decent range training everything on an NVIDIA GeForce GTX 1080Ti with tensorflow 1.4, CUDA 8.0, cuDNN 6.1, python 3.6

1. 2 layers of LSTMs with hidden units to 1024 initialized with a uniform distribution between -0.1 and 0.1 with a seed of 2. Notice in case of bidirectional LSTM the hidden units are 512 which are concatenated as the output of LSTM which makes the number of parameters equal in all models.

2. 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.

3. Embedding size 300 which is enough to learn a dense representation of our vocabulary size (10k), also embeddings are allowed to be re-trained in each epoch. We are using pretrained embeddings since the original ones were trained on google news which is a formal style of text whereas Yelp reviews are more informal.

4. Batch sizes were calculated so that there are around 512 mini-batches per epoch, this makes sure we have equal mini batches in all reduced and full dataset models.

5. Adam Optimizer which is basically *rmsprop* (suitable for text) with momentum with a Learning Rate: 0.01

6. We are using Greedy Decoder with *GreedyEmbeddingHelper* which uses the *argmax* of the output (treated as logits) and passes the result through an embedding layer to get the next input.

7. We are using a sequence loss which is basically a weighted cross-entropy loss for a sequence of logits.

8. Although LSTMs tend to not suffer from the vanishing gradient problem, they can have exploding gradients, to address that we clipped our gradients at (-5.0, 5.0)

9. Different sentences have different lengths. Most sentences are short (e.g., length 20-30) but some sentences are long (e.g., length >100), so a minibatch of 512 randomly chosen training sentences will have many short sentences and few long sentences, and as a result, much of the computation in the minibatch is wasted. To address this problem, we made sure that all sentences within a minibatch were roughly of the same length, which a 2x speedup.
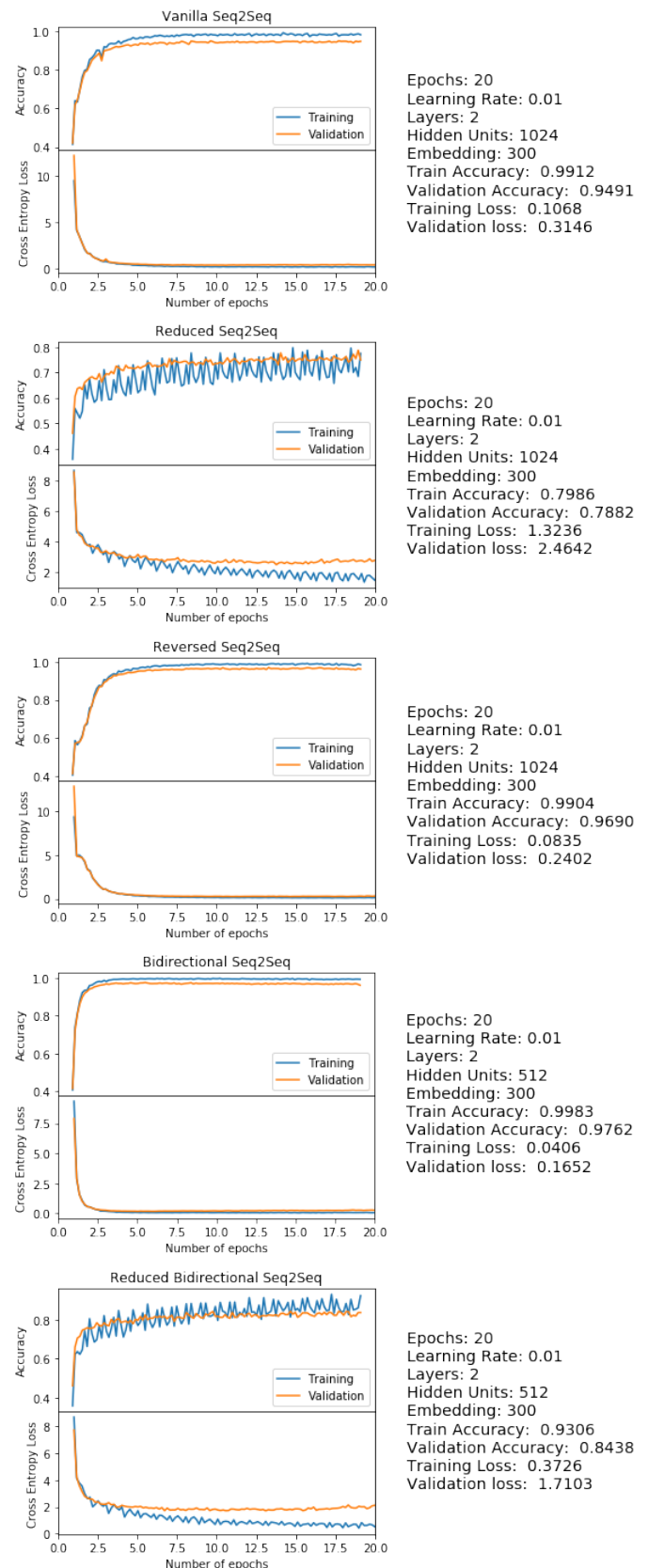

Epochs: 20
Learning Rate: 0.01
Layers: 2
Hidden Units: 1024
Embedding: 300
Train Accuracy: 0.9912
Validation Accuracy: 0.9491
Training Loss: 0.1068
Validation loss: 0.3146

Epochs: 20
Learning Rate: 0.01
Layers: 2
Hidden Units: 1024
Embedding: 300
Train Accuracy: 0.7986
Validation Accuracy: 0.7882
Training Loss: 1.3236
Validation loss: 2.4642

Epochs: 20
Learning Rate: 0.01
Layers: 2
Hidden Units: 1024
Embedding: 300
Train Accuracy: 0.9904
Validation Accuracy: 0.9690
Training Loss: 0.0835
Validation loss: 0.2402

Epochs: 20
Learning Rate: 0.01
Layers: 2
Hidden Units: 512
Embedding: 300
Train Accuracy: 0.9983
Validation Accuracy: 0.9762
Training Loss: 0.0406
Validation loss: 0.1652

Epochs: 20
Learning Rate: 0.01
Layers: 2
Hidden Units: 512
Embedding: 300
Train Accuracy: 0.9306
Validation Accuracy: 0.8438
Training Loss: 0.3726
Validation loss: 1.7103

Figure 9: Accuracy and Loss measurements for various models

Qualitative Study

To give people some intuitive sense of how our model performs, we sampled some instances from each style transfer case, and evaluate them three metrics defined in the evaluation section. Below are randomly sampled results of five models in Table 2. We can see that the auto-encoder almost always produces output text that is sound, coherent and effectively transferred style in most of the cases as the input with changing the sentiment from positive to negative or vice versa. Whereas, the other models tend to generate results that replace a few significant words or phrases while changing the sentiment. We were successfully able to do transfer sentiment while preserving most of the common content, thus fulfilling our aim.

Type	Negative Sentiment Transfer to Positive Sentiment
Ground Truth	ok never going back to this place again .
Vanilla	ok will be going to this place again .
Reduced	plus loves going back to this place again .
Reversed	this place again place ok find phoenix back .
Bidirectional	phenomenal again this location to going going delicious up !
Reduced Bidirectional	this place again this place every time soon
Ground Truth	very disappointed !
Vanilla	very efficient !
Reduced	very impressed !
Reversed	disappointed as !
Bidirectional	beware delightful !
Reduced Bidirectional	roti .
Ground Truth	bad management .
Vanilla	pure management .
Reduced	truly management .
Reversed	honestly night .
Bidirectional	good management .
Reduced Bidirectional	management !
Ground Truth	i love the food ... however service here is horrible .
Vanilla	i love the food ... this is always and delicious .
Reduced	i love the food ... however service here is crazy .
Reversed	needless i love the food ... however service is pretty .
Bidirectional	they provide is here service however , most the club tacos !
Reduced Bidirectional	i love the food ... however service here is amazing .
Ground Truth	i will never be back .
Vanilla	i will definite be back .
Reduced	i will be going back .
Reversed	back however i will never be
Bidirectional	give back be be be again !
Reduced Bidirectional	i will be back .
Ground Truth	the service sucks , management is terrible .
Vanilla	the service , folks , is spectacular .
Reduced	the service came , management is terrible .
Bidirectional	the service great, management better .
Reversed	have fees is funny , open service occasions !
Reduced Bidirectional	management is fairly unbelievable , was amazing !
Ground Truth	they are completely unprofessional and have no experience .
Vanilla	they are fully greatly and have made better .
Reversed	all here could have and and dig cars !
Reduced	they are completely trendy and have no experience .
Bidirectional	have no experience no we are completely trip .
Reduced Bidirectional	have myself experience now experience again again area .

Table 2: Samples from different models

Conclusion

We discussed the state of research on the topic of style transfer, the reasons behind lack of research, its applications in all fields of natural language comprehension, and how some of the deep learning techniques like sequence to sequence and autoencoders can be of help. We proposed a model and experimented with different types of architectures, vocabulary size, depth of network as well as the type of cells. We created our own evaluation criterion which is suitable for the task of sentiment classification. Our best model transferred sentiment with 67% accuracy showing results that even seem reasonable from human perception from the qualitative study. The paper shows a promise on the power of autoencoders combined with seq2seq in multi-domain adaption and calls for future work in using adversarial training or single-encoder for content preservation and multi-decoder to decode in multiple styles at ones.

Acknowledgement

References

Ficler, J. and Goldberg, Y. (2017). Controlling linguistic style aspects in neural language generation. *arXiv preprint arXiv:1707.02633*.

Fu, Z., Tan, X., Peng, N., Zhao, D., and Yan, R. (2017). Style transfer in text: Exploration and evaluation. *arXiv preprint arXiv:1711.06861*.

Gatys, L. A., Ecker, A. S., and Bethge, M. (2016). Image style transfer using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2414–2423.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680.

Jhamtani, H., Gangal, V., Hovy, E. H., and Nyberg, E. (2017). Shakespearizing modern language using copy-enriched sequence-to-sequence models. *CoRR*, abs/1707.01161.

Li, J., Galley, M., Brockett, C., Spithourakis, G. P., Gao, J., and Dolan, B. (2016). A persona-based neural conversation model. *arXiv preprint arXiv:1603.06155*.

Lin, C. (2005). Recall-oriented understudy for gisting evaluation (rouge). *Retrieved August*, 20:2005.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1985). Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science.

Shen, T., Lei, T., Barzilay, R., and Jaakkola, T. (2017). Style transfer from non-parallel text by cross-alignment. *arXiv preprint arXiv:1705.09655*.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Vinyals, O., Fortunato, M., and Jaitly, N. (2015). Pointer networks. In *Advances in Neural Information Processing Systems*, pages 2692–2700.

Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint arXiv:1703.10593*.