Neural Style Transfer in Text

Zhiping (Patricia) Xiao

University of California, Los Angeles

February 23, 2021

Outline

UCLA

Overview

Style transfer in text is the task of rephrasing the text from one style to another without changing other aspects of the meaning.

In computer vision (CV), style transfer refer to changing images from one style (e.g. photo) to another (e.g. Monet painting).

Observing examples such as:
"let's talk about abortion issues"
Democratic ver.: "let's stand by the pro-choice majority"
Republican ver.: "we need some pro-life policy"
— I think it would be interesting to build models to learn such transferring automatically.

UCLA

Summary:

- ▶ Style Transfer in Text: Exploration and Evaluation (AAAI'18)

Papers:

- ▶ Style Transfer Through Back-Translation (ACL'18) (code)
- ▶ CycleGAN-based Emotion Style Transfer as Data Augmentation for Speech Emotion Recognition (INTERSPEECH'19) (code)
- ▶ Cycle-Consistent Adversarial Autoencoders for Unsupervised Text Style Transfer (COLING'20)

UCLA

The progress in language style transfer is lagged behind other domains (e.g. CV).

Special Challenges in **Text** Style Transfer:

▶ Lack of parallel data;
    ▶ Solution #1: build better data sets with human experts
    ▶ Solution #2: focus on **unpaired** approaches (i.e. don't need samples like "a in style A is b in style B")

The progress in language style transfer is lagged behind other domains (e.g. CV).

Special Challenges in **Text** Style Transfer:
- Lack of parallel data;
  - Solution #1: build better data sets with human experts
  - Solution #2: focus on **unpaired** approaches (i.e. don't need samples like "a in style A is b in style B")
- Lack of reliable evaluation metrics.
  - Solution #1: human evaluations
  - Solution #2: design metrics to evaluate some important properties (e.g. style difference & content preservation)

Selected Related Works

Style Transfer Through Back-Translation (ACL'18)

Why back-translation:

- ▶ serves as the *Encoder*, represents the **meaning** of the input sentence;
- ▶ weakens the style attributes. [1]

How to do style-transfer:

1. pre-trained (supervised training) style classifier
2. multi-task decoder
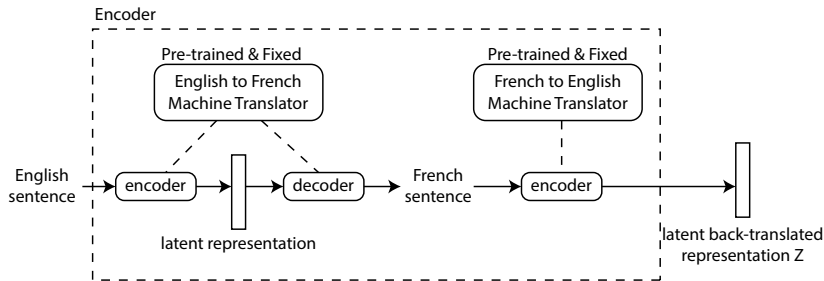
[1]Discussed in https://arxiv.org/abs/1610.05461

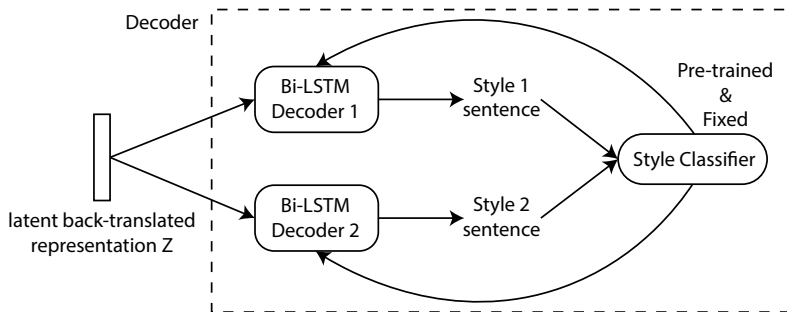Figure: The Encoder. Machine Translation Models are fixed.

Figure: The Decoder. Could be regarded as a multi-task decoder. The style classifier is a convolutional neural network (CNN) trained in a supervised manner, on held-out training data (never used to train style transfer decoders later on).

How the challenges are solved:

- ▶ Lack of parallel data: train against style classifier;
- ▶ Lack of reliable evaluation metrics:
    1. Style transfer accuracy: the proportion of our generated sentences of the desired style (according to the pre-trained style classifiers).
    2. Preservation of meaning: conducted human evaluations.
    3. Fluency (the readability and the naturalness): conducted human evaluations.

Components (decoders and classifier) are *somewhat* "adversarial", but are not trained end-to-end.

By design, the style-transfer models are trying to generate "just-as-good" fake samples, such that the fake ones are hard to be distinguished from genuine ones.

It is natural to come up with a style-classifier, and a generator to work against it.

A framework: estimating generative models via an adversarial process.

- ► simultaneously train two models:
  - ► a generative model **G**: captures the data distribution, generates synthetic data that looks like real;
  - ► a discriminative model **D**: tell the probability that a sample came from the training data rather than **G**.
- ► training procedure: corresponds to a minimax two-player game
  - ► for **G**: to maximize the probability of **D** making a mistake;
  - ► for **D**: to minimize the mistake that **D** makes regarding the current **G**.

UCLA

A unique solution exists:

- ▶ **G**: recovering the training data distribution;
- ▶ **D**: equal to $\frac{1}{2}$ everywhere.

in the case where **G** and **D** are both multilayer perceptrons (MLP), the entire system can be trained end-to-end.

Objective of vanilla GAN is:

$$\min_G \max_D \mathbb{E}_{x_{real}}\big(\log(D(x_{real}))\big) + \mathbb{E}_{x_{fake}}\big(\log(1 - D(G(x_{fake})))\big)$$

D (discriminator, output 1 for real data and 0 for fake data) is trained first and then **G** in each iteration. While training **D** we optimize:

$$\max_D \mathbb{E}_{x_{real}}\big(\log(D(x_{real}))\big) + \mathbb{E}_{x_{fake}}\big(\log(1 - D(G(x_{fake})))\big)$$

which is minimizing:

$$\ell_D = -\mathbb{E}_{x_{real}}\big(\log(D(x_{real}))\big) - \mathbb{E}_{x_{fake}}\big(\log(1 - D(G(x_{fake})))\big)$$

and while training **G** we minimize:
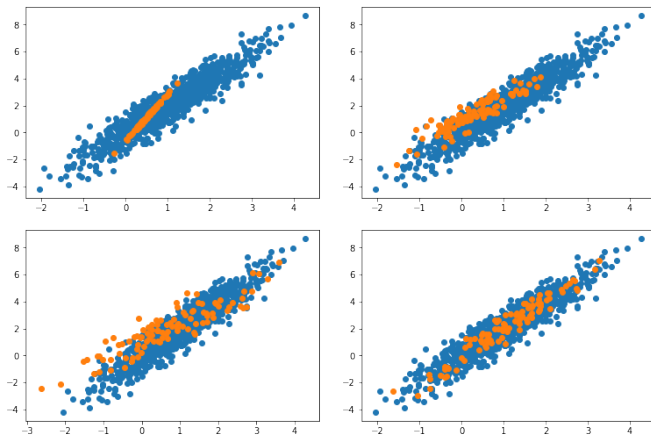
$$\ell_G = \log(1 - D(G(x_{fake})))$$

UCLA

Figure: Example. Noise distribution: normal. [3]
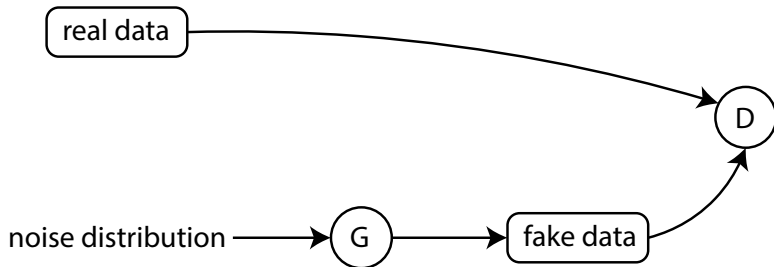
---

[3] URL of tutorial online (with code).

UCLA

Figure:
$$\min_G \max_D \mathbb{E}_{x_{real}}\big(\log(D(x_{real}))\big) + \mathbb{E}_{x_{fake}}\big(\log(1 - D(G(x_{fake})))\big)$$
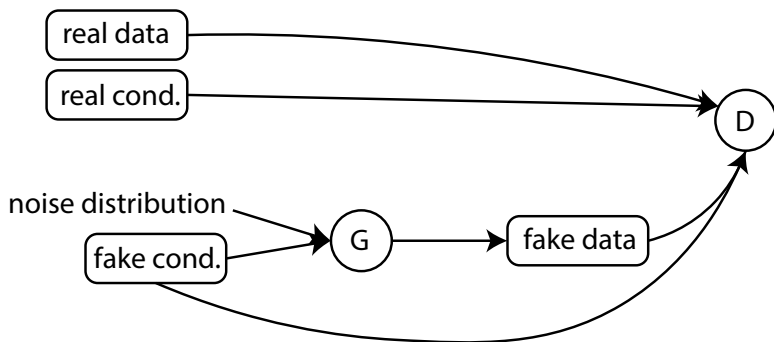
UCLA

Figure: For a condition y,
$$\min_G \max_D \mathbb{E}_{x_{real}} \big(\log(D(x_{real}|y)) \big) + \mathbb{E}_{x_{fake}} \big(\log(1 - D(G(x_{fake}|y))) \big)$$
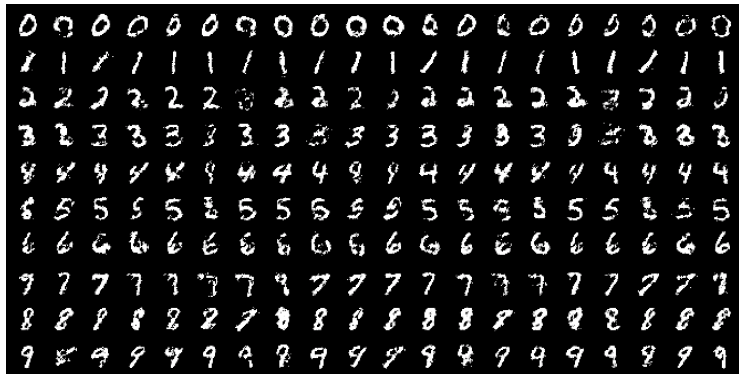
Figure: Generated MNIST digits, each row conditioned on one label.

For more on applications, see pix2pix [5] (condition: outline of shapes).
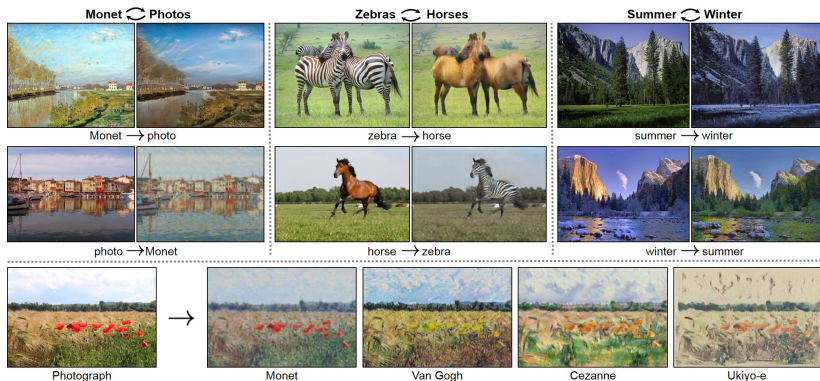
[5] https://arxiv.org/abs/1611.07004

Figure: CycleGAN example. Cycle refers to the bi-directional transfer. There are two generators indeed.

CycleGAN framework solved the "**lack of parallel data**" problem, by not requiring paired samples from different styles.

▶ Input data: data from domain $X = \{x_i\}_{i=1}^N$, and domain $Y = \{y_j\}_{j=1}^M$. Two sets of data, two different styles.

▶ Two generator / translators: $G : X \to Y$ and $F : Y \to X$.

▶ Associated adversarial discriminators D_X and D_Y.

▶ G, F, D_X, D_Y are trained together, end-to-end.

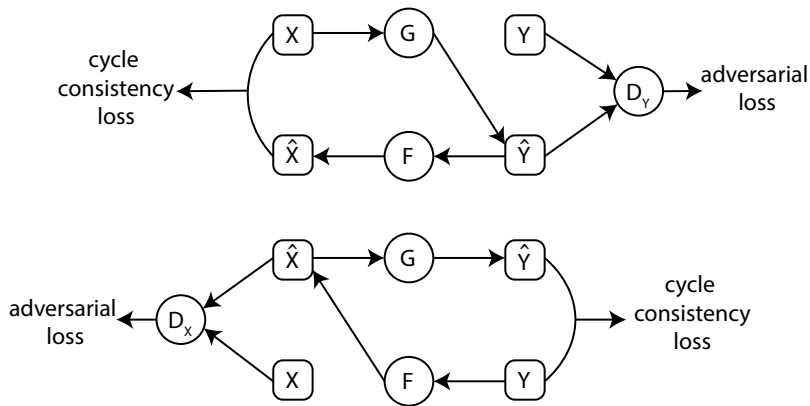▶ To enhance cycle consistency, encourage $F(G(x)) \approx x$ and $G(F(y)) \approx y$.

Figure: CycleGAN training pipeline.

Adversarial losses:

$$\min_G \max_{D_Y} \mathcal{L}_{\mathrm{GAN}}(G, D_Y, X, Y)$$

$$= \min_G \max_{D_Y} \Big(\mathbb{E}_x\big(\log(1 - D_Y(G(x)))\big) + \mathbb{E}_y\big(\log(D_Y(y))\big) \Big)$$

$$\min_F \max_{D_X} \mathcal{L}_{\mathrm{GAN}}(F, D_X, X, Y)$$

$$= \min_F \max_{D_X} \Big(\mathbb{E}_x\big(\log(D_X(x))\big) + \mathbb{E}_y\big(\log(1 - D_X(F(y)))\big) \Big)$$

Cycle-consistency loss:

$$\mathcal{L}_{\mathrm{cyc}}(G, F) = \mathbb{E}_x\big(\|F(G(x)) - x\|_1\big) + \mathbb{E}_y\big(\|G(F(y)) - y\|_1\big)$$

UCLA

Full objective:

$$\mathcal{L}(G, F, D_X, D_Y) = \mathcal{L}_{\text{GAN}}(G, D_Y, X, Y)$$
$$+ \mathcal{L}_{\text{GAN}}(F, D_X, X, Y)$$
$$+ \lambda \mathcal{L}_{\text{cyc}}(G, F)$$

Aim to solve:

$$G^*, F^* = \arg \min_{G,F} \max_{D_X, D_Y} \mathcal{L}(G, F, D_X, D_Y)$$

Full objective:

$$\mathcal{L}(G, F, D_X, D_Y) = \mathcal{L}_{\text{GAN}}(G, D_Y, X, Y)$$
$$+ \mathcal{L}_{\text{GAN}}(F, D_X, X, Y)$$
$$+ \lambda \mathcal{L}_{\text{cyc}}(G, F)$$

Aim to solve:

$$G^*, F^* = \arg \min_{G,F} \max_{D_X, D_Y} \mathcal{L}(G, F, D_X, D_Y)$$

Can be viewed as training two autoencoders jointly:

- $F \circ G : X \to X$
- $G \circ F : Y \to Y$

CycleGAN-based Emotion Style Transfer as Data Augmentation for Speech Emotion Recognition (INTERSPEECH'19)

It is not transferring the whole sentence into another sentence of another style (i.e. emotion). Instead, it is transferring the features into another style's features.

- ▶ Where is CycleGAN used: generate synthetic feature vectors.
- ▶ Why CycleGAN: to obtain a better classifier
  - ▶ classifier trained on the combination of real and synthetic feature vectors achieves better classification performance than those rely on the real features.
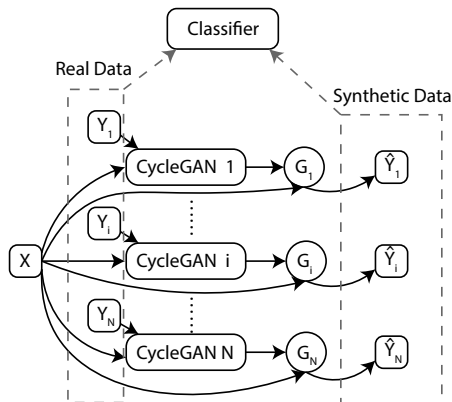
UCLA

Figure: The data-augmented emotion classifier. X is an external unlabeled dataset, and Y_i represents the features of emotion-i samples in the labeled dataset.

The classification loss can be defined as a softmax cross-entropy loss:

$$\mathcal{L}_{\text{cls}} = \sum_i t_i \log(C(G_i(X)))$$

where t_i is the class label and C is the classifier.

Full objective:

$$\mathcal{L} = \sum_{i=1}^{N} \mathcal{L}_i^{\text{GAN}}(G_i, D_{Y_i}, X, Y_i) + \sum_{i=1}^{N} \mathcal{L}_i^{\text{GAN}}(F_i, D_X, X, Y_i)$$

$$+ \lambda_{\text{cyc}} \sum_{i=1}^{N} \mathcal{L}_{\text{cyc}}(G_i, F) + \lambda_{\text{cls}} \mathcal{L}_{\text{cls}}$$

Aim to solve:

$$C^*, G_1^*, \ldots, G_N^*, F^* = \arg \min_{C, G_1, \ldots, G_N, F} \max_{D_X, D_Y} \mathcal{L}$$

UCLA

Structure:

- Encoder (sentences → features) is not needed: Directly use the "emobase2010" reference feature set, which is based on the Interspeech 2010 Paralinguistic Challenge feature set, consisting of 1,582 features.
- N CycleGANs: each corresponds to an emotion type;
- A domain classifier: classifying N emotion types.

Result: classifier performance gets improved.

UCLA

Cycle-Consistent Adversarial Autoencoders for Unsupervised Text Style Transfer (COLING'20)

Components:

► Encoder: vanilla LSTM autoencoders (1997 ver.), *one for each style*, sequence → feature, or feature → sequence;

► Transfer Nets: transforming features from one style to the other;

► Cycle-consistent Constraints: $F(G(x)) \approx x$, $G(F(y)) \approx y$.

The encoded representation of style

- X: $Z_X = \text{enc}_X(X)$;
- Y: $Z_Y = \text{enc}_Y(Y)$.

where enc_X and enc_Y are the LSTM autoencoders for style X and style Y respectively.

There are corresponding decoders dec_X and dec_Y. Objective of the autoencoders is defined as (assuming $X = \{x_i\}_{i=1}^{N}$ and $Y = \{y_j\}_{j=1}^{M}$, x_i, y_j are sequences):

$$\mathcal{L}_R(\text{enc}_X, \text{dec}_X, \text{enc}_Y, \text{dec}_Y)$$

$$= -\frac{1}{N} \sum_{i=1}^{N} \log p(\text{dec}_X(\text{enc}_X(x_i)) = x_i)$$

$$- \frac{1}{M} \sum_{j=1}^{M} \log p(\text{dec}_Y(\text{enc}_Y(y_j)) = y_j)$$

UCLA

Adversarial losses:

$$\min_G \max_{D_Y} \mathcal{L}_{\text{GAN}}(G, D_Y, X, Y)$$

$$= \min_G \max_{D_Y} \Big(\mathbb{E}_x \big(\log(1 - D_Y(G(x))) \big) + \mathbb{E}_y \big(\log(D_Y(y)) \big) \Big)$$

$$\min_F \max_{D_X} \mathcal{L}_{\text{GAN}}(F, D_X, X, Y)$$

$$= \min_F \max_{D_X} \Big(\mathbb{E}_x \big(\log(D_X(x)) \big) + \mathbb{E}_y \big(\log(1 - D_X(F(y))) \big) \Big)$$

Cycle-consistency loss:

$$\mathcal{L}_{\text{cyc}}(G, F) = \mathbb{E}_x \big(\|F(G(x)) - x\|_1 \big) + \mathbb{E}_y \big(\|G(F(y)) - y\|_1 \big)$$

UCLA

Adversarial losses:

$$\min_{G} \max_{D_Y} \mathcal{L}_{\mathrm{GAN}}(G, D_Y)$$

$$= \min_{G} \max_{D_Y} \Big(\mathbb{E}_{z_x} \big(\log(1 - D_Y(G(z_x))) \big) + \mathbb{E}_{z_y} \big(\log(D_Y(z_y)) \big)$$

$$+ \mathbb{E}_x \big(\log(1 - D_Y(G(\mathrm{enc}_X(x)))) \big) + \mathbb{E}_y \big(\log(D_Y(\mathrm{enc}_Y(y))) \big) \Big)$$

$$\min_{F} \max_{D_X} \mathcal{L}_{\mathrm{GAN}}(F, D_X)$$

$$= \min_{F} \max_{D_X} \Big(\mathbb{E}_{z_x} \big(\log(D_X(z_x)) \big) + \mathbb{E}_{z_y} \big(\log(1 - D_X(F(z_y))) \big)$$

$$+ \mathbb{E}_x \big(\log(D_X(\mathrm{enc}_X(x))) \big) + \mathbb{E}_y \big(\log(1 - D_X(F(\mathrm{enc}_Y(y)))) \big)$$

Cycle-consistency loss:

$$\mathcal{L}_{\text{cyc}}(G, F) = \mathbb{E}_{z_x}\big(\|F(G(z_x)) - z_x\|_1\big) + \mathbb{E}_{z_y}\big(\|G(F(z_y)) - z_y\|_1\big)$$
$$+ \mathbb{E}_x\big(\|F(G(\text{enc}_X(x))) - \text{enc}_X(x)\|_1\big)$$
$$+ \mathbb{E}_y\big(\|G(F(\text{enc}_Y(y))) - \text{enc}_Y(y)\|_1\big)$$

The full objective:

$$\mathcal{L}_{\text{CAE}} = \lambda_1 \mathcal{L}_R(\text{enc}_X, \text{dec}_X, \text{enc}_Y, \text{dec}_Y)$$
$$+ \lambda_2\big(\mathcal{L}_{\text{GAN}}(G, D_Y) + \mathcal{L}_{\text{GAN}}(F, D_X)\big)$$
$$+ \lambda_3 \mathcal{L}_{\text{cyc}}(G, F)$$

Aim to solve ($k = \{X, Y\}$):

$$G^*, F^*, \text{enc}_k^*, \text{dec}_k^* = \arg \min_{G, F, \text{enc}_k, \text{dec}_k} \max_{D_X, D_Y} \mathcal{L}_{\text{CAE}}$$

UCLA

Now that we have learned:

$$G, F, \text{enc}_X, \text{dec}_X, \text{enc}_Y, \text{dec}_Y$$

we transfer sequence x_i into style Y sequence $\hat{y}_i$ as:

$$\hat{y}_i = \text{dec}_Y(z_{\hat{y}_i}) = \text{dec}_Y(G(z_{x_i})) = \text{dec}_Y\big(G(\text{enc}_X(x_i))\big)$$

Automatic metrics:

- ▶ Transfer: style-transfer success rate, a classifier in *fastText* library;
- ▶ BLEU: evaluate the content preservation;
- ▶ PPL (perplexity): evaluate the fluency of the transferred sequence;
- ▶ RPPL (reverse perplexity): evaluate representativeness with respect to the underlying data distribution, detect the mode collapse for generative models, etc.

Human evaluation: let human grade generated sentences with scores from 1 to 5 for style transfer, content preservation and fluency.

UCLA

Reinforcement-learning Based approaches:

- ▶ Reinforcement Learning Based Text Style Transfer without Parallel Training Corpus (NAACL-HLT'19) (code)
  - ▶ Adversarial training involved.
- ▶ A Dual Reinforcement Learning Framework for Unsupervised Text Style Transfer (IJCAI'19) (code)

Domain adaptation:

- ▶ Domain Adaptive Text Style Transfer (EMNLP'19) (code)
  - ▶ Assume that the target domain only has limited non-parallel data; but source domain can be unknown.

UCLA

Summary

BST v.s. CAE, Positive v.s. Negative. [7]

negative → positive	
crap fries, hard hamburger buns, burger tasted like crap!	*empathy, the best food, but it was very nice!*
the people behind the counter were not friendly whatsoever.	*the people here are really good.*
this place is bad news!	*this place is amazing!*

positive → negative	
the food is excellent and the service is exceptional!	*the food is horrible and the service is terrible.*
great as always, i love there food.	*really disappointed, i couldn't be back.*
i would recommend a visit here.	*i will not recommend this place.*

Figure: BST results

Models	Positive → Negative	Negative → Positive
Source	it has a great atmosphere , with wonderful service .	i have n't received any response to anything .
CAE	it has a horrible atmosphere , with awful service .	i have been pleased with a wonder time .
Source	the steak was really juicy with my side of salsa to balance the flavor.	my corn beef hash was mushy with uncooked veggies .
CAE	the steak was really dry with my sauce on the salsa .	my beef hash was juicy and tender .

Figure: CAE results

[7]The only similar experiment we can find.

Components of a generator:
(sequence →) encoder (→ feature →) decoder (→ sequence)
The encoder / decoder are not necessarily LSTMs.

- ▶ encoder: can be either style-specific or not; can be as simple as RNNs, or as complex as neural machine translation.
- ▶ decoder:
  - ▶ Type #1: multi-decoder model (one decoder per style)
  - ▶ Type #2: style-embedding model (style as parameter)

Adversarial framework is typically used for separating *style* from *content*, and avoid the *lack-of-parallel-data* problem.

Thank you! ☺

UCLA