Controllable Unsupervised Text Attribute Transfer via Editing Entangled Latent Representation

Ke Wang, Hang Hua, Xiaojun Wan

Presenter: Chiyu Zhang

Context

Unsupervised text style (attribute) transfer:

- Transform a text to alter a specific attribute (e.g. sentiment) without using any parallel data
- Preserving its attribute-independent content.
- Conforming to the target attribute
- Maintaining linguistic fluency

Context

The dominant approaches are:

- to separately model attribute and content representations.
- Such as using ***multiple attribute-specific decoders*** or ***combining*** the ***content representations*** with different ***attribute representations*** to decode texts with target attribute.

Context

Shortcomings:

- First, disentangle **attribute** and attribute-independent **content**, this may undermine ***the integrity (i.e., naturality)*** and result in poor ***readability*** of the generated sentences.
- Second, require modeling each new attribute separately and thus lack ***flexibility and controllability.***

This work

- They present ***a controllable unsupervised*** text attribute transfer framework, which can edit the ***entangled latent representation.***
- Their ***latent representation*** is an entangled representation of both ***attribute and content***.

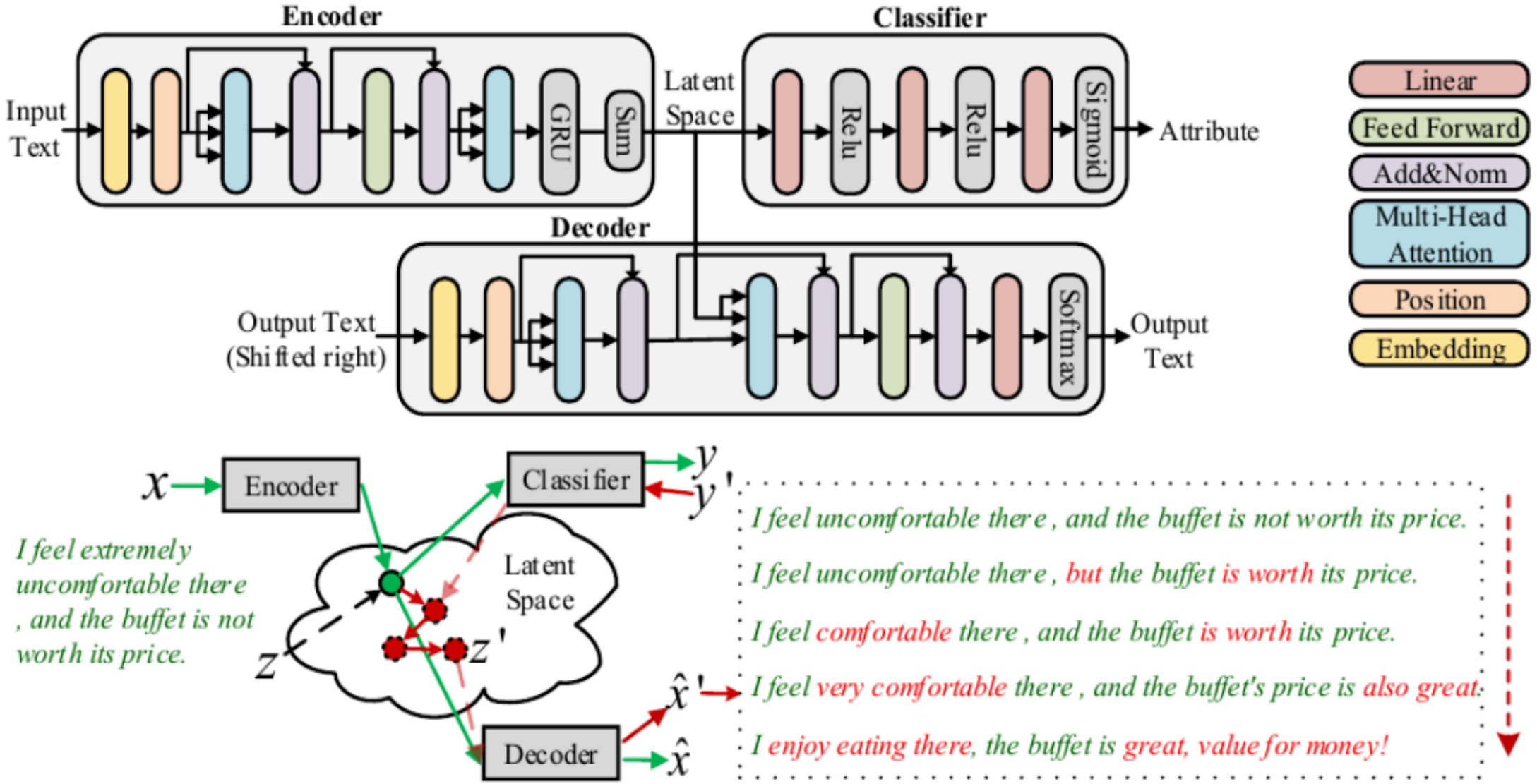
Overview

- They present ***a controllable unsupervised*** text attribute transfer framework, which can edit the ***entangled latent representation.***
- build a Transformer-based ***autoencoder*** to learn an entangled latent representation for both ***attribute and content;***
- Train attribute ***classifier***;
- use proposed ***Fast-Gradient-Iterative-Modification (FGIM)*** algorithm to iteratively ***edit*** the latent representation, until the ***latent representation*** can be identified as target attribute by the ***classifier***.

Model Architecture

- The whole framework can be divided into three sub-models:
- **an encoder E** which encodes the text ***x*** into a latent representation ***z***,
- a **decoder D** which decodes text ***x*** from ***z***,
- an **attribute classifier C** that classifies attribute of the latent representation ***z***.

$$\boldsymbol{z} = E_{\theta_e}(\boldsymbol{x});\ \boldsymbol{y} = C_{\theta_c}(\boldsymbol{z});\ \hat{\boldsymbol{x}} = D_{\theta_d}(\boldsymbol{z})$$


Encoder
Classifier
Input Text
GRU
Sum
Latent Space
Relu
Relu
Sigmoid
Attribute
Decoder
Output Text (Shifted right)
Softmax
Output Text
Linear
Feed Forward
Add&Norm
Multi-Head Attention
Position
Embedding
x
Encoder
Classifier
y
y'
I feel extremely uncomfortable there , and the buffet is not worth its price.
Latent Space
z
z'
x̂'
x̂
Decoder
I feel uncomfortable there , and the buffet is not worth its price.
I feel uncomfortable there , but the buffet is worth its price.
I feel comfortable there , and the buffet is worth its price.
I feel very comfortable there , and the buffet's price is also great.
I enjoy eating there, the buffet is great, value for money!

Model Architecture

The whole framework can be divided into three sub-models:

- **an encoder E** which encodes the text ***x*** into a latent representation ***z***,
- a **decoder D** which decodes text ***x*** from ***z***,
- an **attribute classifier C** that classifies attribute of the latent representation ***z***.

$$\boldsymbol{z} = E_{\theta_e}(\boldsymbol{x});\ \boldsymbol{y} = C_{\theta_c}(\boldsymbol{z});\ \hat{\boldsymbol{x}} = D_{\theta_d}(\boldsymbol{z})$$

They formulate the text attribute transfer task as an ***optimization problem***:

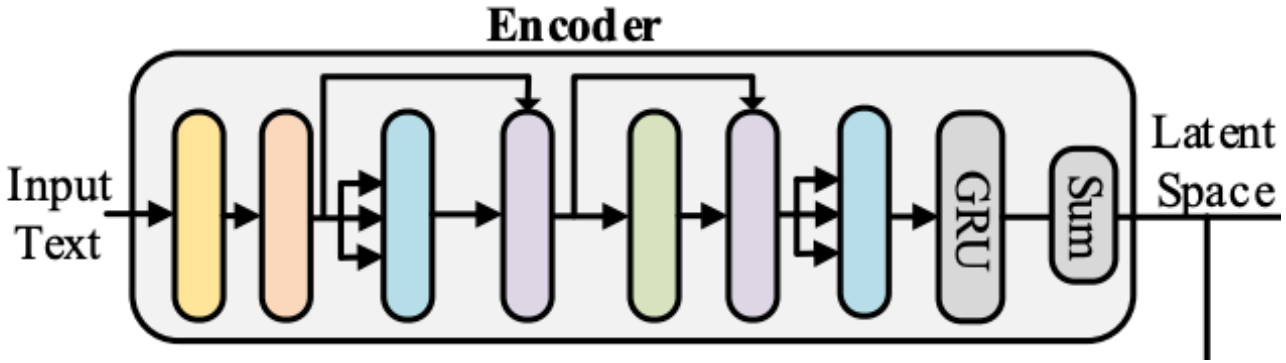
$$\hat{\boldsymbol{x}}' = D_{\theta_d}(\boldsymbol{z}')\ where\ \boldsymbol{z}' = argmin_{\boldsymbol{z}^*} ||\boldsymbol{z}^* - E_{\theta_e}(\boldsymbol{x})||\ s.t.\ C_{\theta_c}(\boldsymbol{z}^*) = \boldsymbol{y}'$$

Optimization problem

$$\hat{\boldsymbol{x}}' = D_{\theta_d}(\boldsymbol{z}') \; where \; \boldsymbol{z}' = argmin_{\boldsymbol{z}^*} ||\boldsymbol{z}^* - E_{\theta_e}(\boldsymbol{x})|| \; s.t. \; C_{\theta_c}(\boldsymbol{z}^*) = \boldsymbol{y}'$$

They transform the original problem to find an **optimal representation *z'*** that conforms to the **target attribute *y'*** *(requirement ii)* and is **"closest" to *z*** *(requirement i)*, then they decode **the target text $\hat{x}$' from *z'*** *(requirement iii)*.

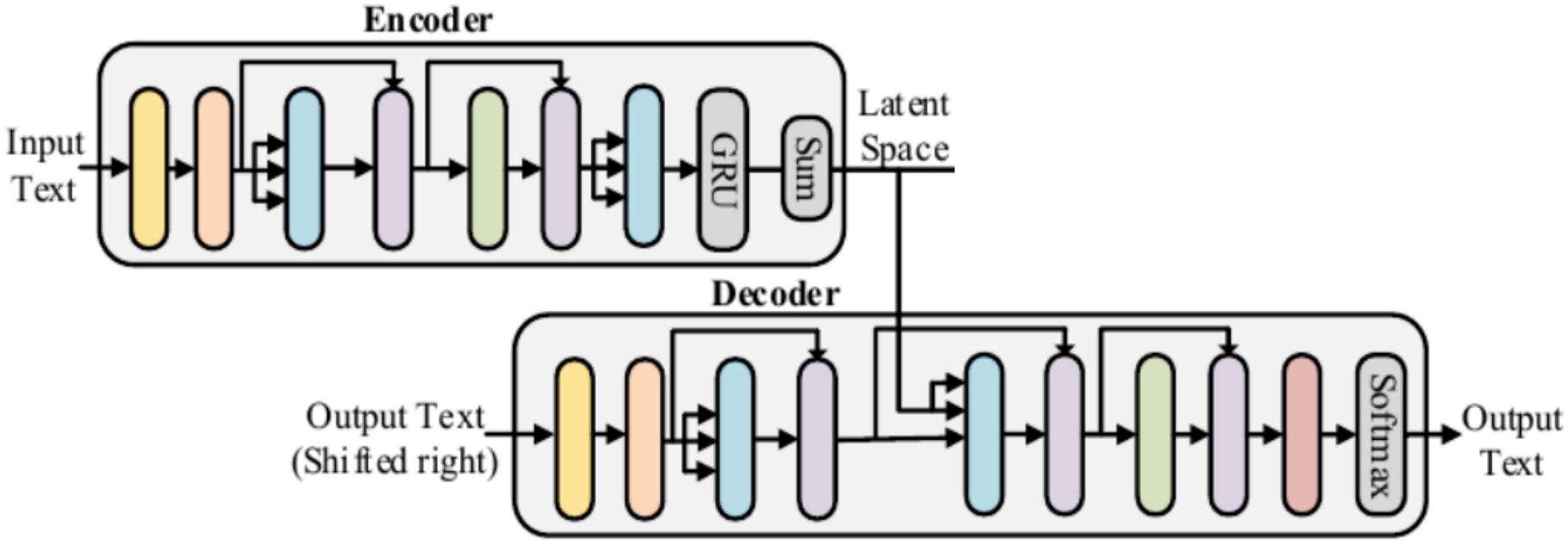
Transformer-based Autoencoder



- A **Transformer-based autoencoder** with low reconstruction bias to learn the **latent representation** of source text.
- Pass source text ***x*** through the original **Transformer's encoder** ($E_{transformer}$) and get the intermediate representations ***U***.
- Add extra **positional embeddings *H* to *U***.
- Pass ***U*** through a **GRU layer with self-attention.**
- Apply a **sigmoid activation function** on the GRU hidden representations and **sum them** to get the **final latent representation *z*.**

$$\boldsymbol{z} = E_{\theta_e}(\boldsymbol{x}) = Sum(Sigmoid(GRU(\boldsymbol{U} + \boldsymbol{H}))), where\ \boldsymbol{U} = E_{transformer}(\boldsymbol{x}).$$

Decoder



The target text $\hat{x}$ can be decoded from z.

Autoencoder reconstruction loss

$$\mathcal{L}_{ae}(D_{\theta_d}(E_{\theta_e}(\boldsymbol{x})), \boldsymbol{x}) = \mathcal{L}_{ae}(D_{\theta_d}(\boldsymbol{z}), \boldsymbol{x}) = -\sum^{|\boldsymbol{x}|}((1-\varepsilon)\sum_{i=1}^{v}\bar{p}_i \log(p_i) + \frac{\varepsilon}{v}\sum_{i=1}^{v}\log(p_i)),$$

v denotes the vocabulary size, and ε denotes the smoothing parameter

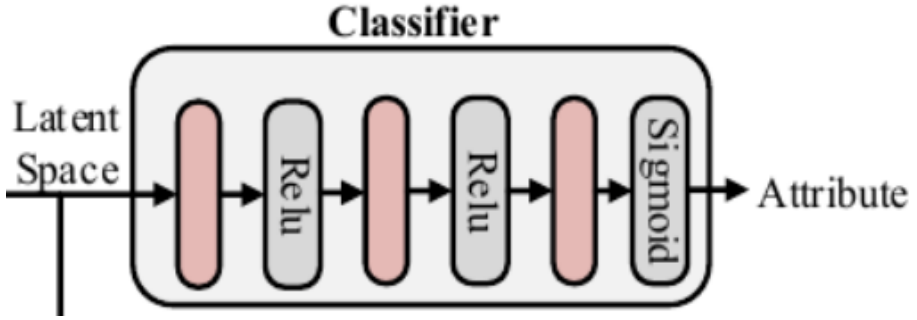
$\frac{\varepsilon}{v}\sum_{i=1}^{v}\log(p_i)$ The introduction of noise to relax confidence in the label.

p and $\bar{p}$ are the ***predicted*** probability distribution and the ***ground truth*** probability distribution over the vocabulary

Attribute Classifier for Latent Representation

- Use an attribute classifier to provide ***the direction*** (gradient) for editing the latent representation
- The classifier is ***two stacks of linear layer with sigmoid activation*** function, and the attribute classification loss is:

$$\mathcal{L}_c(C_{\theta_c}(\boldsymbol{z}), \boldsymbol{y}) = -\sum_{i=1}^{|\boldsymbol{q}|} \bar{q}_i \log q_i,$$



q represents the ***predicted*** attribute probability distribution and $\bar{q}$ is the ***true attribute*** probability distribution.

Fast-Gradient-Iterative-Modification algorithm (FGIM)

- FGIM modifies z based on the gradient of back-propagation by linearizing the **attribute classifier's** loss function on z.
- To get an optimal z', they first use z as the input of $C_{\theta c}$ and use y' as the label to calculate the gradient to z. Then they **modify z** in this direction iteratively until they get a z' that can be identified as the target attribute y' by the classifier $C_{\theta c}$.
- The newly modified latent representation z^*:

$$z^* = z - w_i \nabla_z \mathcal{L}_c(C_{\theta_c}(z), y'),$$

where w_i is the modification weight used for controlling the degree of transfer.

FGIM

- They want a modification to make the latent representation more different in attribute.
- They propose a ***Dynamic-weight-initialization method*** to allocate the initial modification weight w_i in each trial process.
- They give a set of weights $w = \{w_i\}$, and their algorithm will ***dynamically try each weight*** in w **from small to large** until they get target latent representation z'.

Algorithm 1 Fast Gradient Iterative Modification Algorithm.

Input: Original latent representation $\boldsymbol{z}$; Well-trained attribute classifier C_{θ_c}; A set of weights $\boldsymbol{w} = \{w_i\}$; Decay coefficient λ; Target attribute $\boldsymbol{y'}$; Threshold t;

Output: An optimal modified latent representation $\boldsymbol{z'}$;

for each $w_i \in \boldsymbol{w}$ **do**
  $\boldsymbol{z}^* = \boldsymbol{z} - w_i \nabla_{\boldsymbol{z}} \mathcal{L}_c(C_{\theta_c}(\boldsymbol{z}), \boldsymbol{y'})$;
  for s-steps **do**
    if $|\boldsymbol{y'} - C_{\theta_c}(\boldsymbol{z}^*)| < t$ **then** $\boldsymbol{z'} = \boldsymbol{z}^*$; Break;
    end if
    $w_i = \lambda w_i$;
    $\boldsymbol{z}^* = \boldsymbol{z}^* - w_i \nabla_{\boldsymbol{z}^*} \mathcal{L}_c(C_{\theta_c}(\boldsymbol{z}^*), \boldsymbol{y'})$;
  end for
end for
return $\boldsymbol{z'}$;

the initial weight $\boldsymbol{w_i \in w}$ will iteratively decay by multiplying a fixed decay coefficient $\boldsymbol{\lambda}$.

Advantages of FGIM

Attribute Transfer over Multiple Aspects:

- They proposed framework transfers the source text's attribute into any target attribute by using only the classifier $C_{\theta c}$ and the target attribute y.

Transfer Degree Control:

- Our model can use different modification weight in w to control the degree of modification, thus achieving the control of the degree of attribute transfer.

Experiment

- Transformer-based autoencoder: the embedding size, the latent size and the dimension size of self-attention are all set to 256.
- The inner dimension of Feed-Forward Networks (FFN) in Transformer is set to 1024.
- Besides, each of the encoder and decoder is stacked by two layers of Transformer.
- The hidden size of GRU and batch-size are set to 128.
- The smoothing parameter ε is set to 0.1.

Experiment

- For the classifier, the dimensions of the two linear layers are 100 and 50.
- For FGIM, the weight set ***w:*** {1.0, 2.0, 3.0, 4.0, 5.0, 6.0},
- the threshold t: 0.001
- the decay coefficient λ 0.9
- The optimizer is Adam and the initial learning rate is 0.001.

Datasets

Table 1: Statistics for Yelp, Amazon, Captions datasets.

Dataset	Styles	#Train	#Dev	#Test	#Vocab	Max-Length	Mean-Length
Yelp	Negative	180,000	2,000	500	9,640	15	8.89
	Positive	270,000	2,000	500			
Amazon	Negative	277,000	1,015	500	58,991	34	14.84
	Positive	278,000	985	500			
Captions	Humorous	6,000	300	300	8,693	20	14.04
	Romantic	6,000	300	300			

Experiment

Evaluation Metric

- **Acc**: measure the attribute transfer accuracy of the generated texts with a **fastText** classifier trained on the training data.
- **BLEU**: use the multi-BLEU metric to calculate the **similarity** between the generated sentences and the references written by human.
- **PPL**: measure the **fluency** of the generated sentences.

Results

Methods	Yelp			Amazon			Captions		
	Acc	BLEU	PPL ↓	Acc	BLEU	PPL ↓	Acc	BLEU	PPL ↓
CrossAlign [28]	72.3%	9.1	50.8	70.3%	1.9	66.2	78.3%	1.8	69.8
MultiDec [5]	50.2%	14.5	84.5	67.3%	9.1	60.3	68.3%	6.6	60.2
StyleEmb [5]	10.2%	21.1	47.9	43.6%	15.1	60.1	56.2%	8.8	57.1
CycleRL [38]	53.6%	18.8	98.2	52.3%	14.4	183.2	45.2%	5.8	50.3
BackTrans [26]	93.4%	2.5	49.5	84.6%	1.5	48.3	78.3%	1.6	68.3
RuleBase [17]	80.3%	22.6	66.6	67.8%	33.6	52.1	85.3%	**19.2**	35.6
DelRetrGen [17]	88.8%	16.0	49.6	51.2%	29.3	55.4	90.4%	12.0	33.4
UnsupMT [41]	95.2%	22.8	53.9	84.2%	33.9	57.9	**95.5%**	12.7	31.2
Ours	**95.4%**	**24.6**	**46.2**	**85.3%**	**34.1**	**47.4**	92.3%	17.6	**23.7**

Human Evaluation

attribute accuracy (Att), the retainment of content (Con) and the fluency of sentences (Gra).

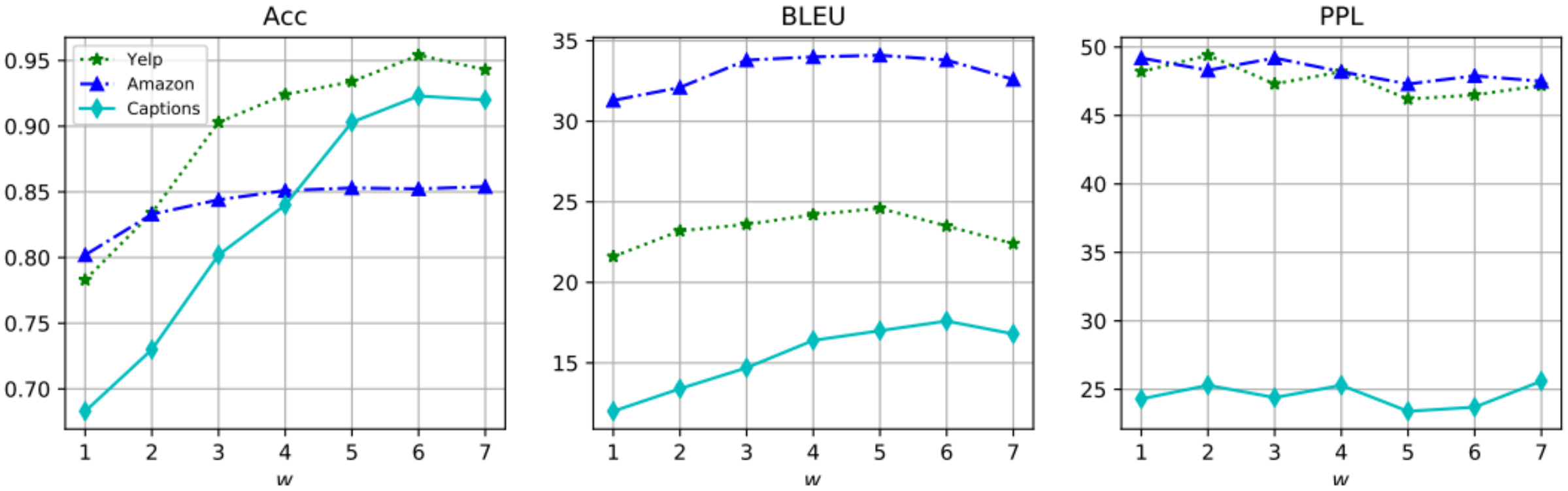
Methods	Yelp			Amazon			Captions		
	Att	Con	Gra	Att	Con	Gra	Att	Con	Gra
CrossAlign [28]	2.5	2.8	3.3	2.7	2.7	3.1	2.1	2.5	3.0
MultiDec [5]	2.3	3.1	2.7	2.6	2.9	2.9	2.5	2.6	2.9
StyleEmb [5]	2.6	3.0	2.9	3.1	2.8	3.2	2.3	3.1	3.0
CycleRL [38]	2.9	3.0	3.2	3.2	3.1	3.2	2.5	2.9	2.8
BackTrans [26]	2.0	2.4	2.9	2.6	2.8	3.4	2.4	2.8	2.8
RuleBase [17]	3.4	3.2	3.4	3.6	3.7	3.8	2.6	3.1	3.0
DelRetrGen [17]	3.2	2.9	3.0	3.7	3.6	3.4	2.5	2.9	3.2
UnsupMT [41]	3.2	3.3	3.5	3.7	4.0	3.7	2.8	2.8	3.3
Ours	**3.6**	**3.5**	**3.8**	**4.0**	**4.2**	**4.1**	**3.5**	**3.4**	**3.5**

Multi-Aspect Sentiment Transfer

- transform 150 texts into texts with all ***negative sentiments*** over five aspects $y = (0.0, 0.0, 0.0, 0.0, 0.0)$
- transform the other 150 texts into texts with all ***positive sentiments*** $y = (1.0, 1.0, 1.0, 1.0, 1.0)$.

Aspects	Acc	Att	Con	Gra
Appearance	90.2%	3.2	3.5	3.8
Aroma	89.3%	3.4	3.9	3.7
Palate	91.2%	3.1	3.8	3.7
Taste	88.2%	3.4	3.7	3.6
Overall	87.3%	3.6	4.0	3.8

Influence of the modification weight w.



Latent Representation Modification Study

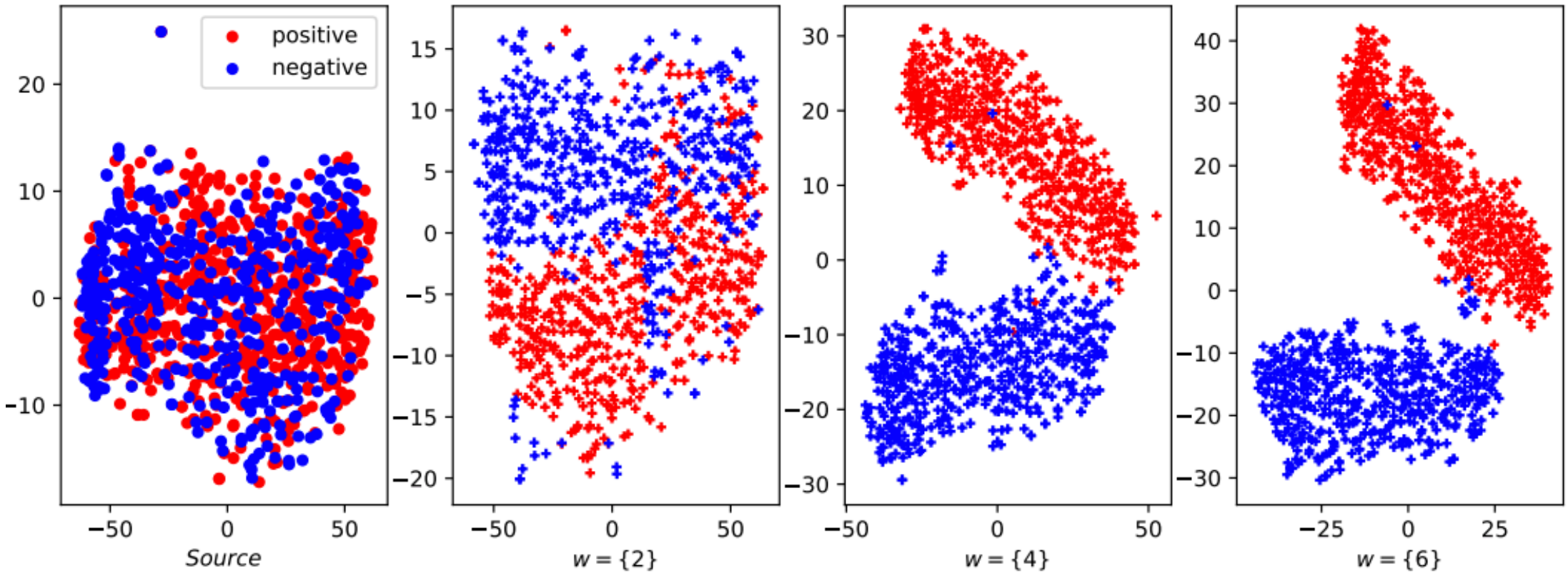


Figure 3: Visualization of representations with different modification weight w.

Latent Representation Modification Study

Table 5: Examples of generation with different modification weight w.

	Positive ->Negative	Negative ->Positive
Source:	really good service and food .	it is n't terrible , but it is n't very good either .
Human:	the service was bad	it is n't perfect , but it is very good .
$w = \{1\}$	really good service and food .	it is n't terrible , but it is n't very good either .
$w = \{2\}$	very good service and food .	it is n't terrible , but it is n't very good delicious either .
$w = \{3\}$	very good food but service is terrible !	it is n't terrible , but it is very good delicious either .
$w = \{4\}$	not good food and service is terrible !	it is n't terrible , but it is very good and delicious .
$w = \{5\}$	bad service and food !	it is n't terrible , but it is very good and delicious appetizer .
$w = \{6\}$	very terrible service and food !	it is excellent , and it is very good and delicious well .

Reference

- https://papers.nips.cc/paper/9284-controllable-unsupervised-text-attribute-transfer-via-editing-entangled-latent-representation
- https://s3.amazonaws.com/postersession.ai/638a7f70-e33e-46a5-9424-3d99536ec3b3.pdf