트랜스포머와 판별기를 이용한 비병렬 데이터의 텍스트 스타일 변환

박다솔○, 차정원
창원대학교
{dasol_p ○,jcha}@changwon.ac.kr

Text Style Transfer of Non-parallel Data using Transformer and Discriminator

Da-Sol Park○, Jeong-Won Cha
Changwon National University

요 약

텍스트 스타일 변환은 문장 내 컨텐츠는 유지하면서 문장의 스타일을 변경하는 것이다. 스타일의 정의가 모호하기 때문에 텍스트 스타일 변환에 대한 연구는 대부분 지도 학습으로 진행되어왔다. 본 논문에서는 병렬 데이터 구축이 되지 않은 데이터를 학습하기 위해 비병렬 데이터를 이용하여 스타일 변환을 시도한다. 트랜스포머 기반의 문장 생성기를 이용하여 문장을 생성하고, 해당 스타일을 분류하는 판별기로 이루어진 모델을 제안한다. 제안 모델을 통해, 감정 변환의 성능은 정확도(Accuracy) 56.9%, self-BLEU 0.393(긍정→부정), 0.366(부정→긍정), 유창성(fluency) 798.23(긍정→부정), 1381.05(부정→긍정)을 보였다. 본 연구는 비병렬 데이터에 대해 스타일 변환을 적용함으로써, 병렬 데이터가 없는 다양한 도메인에도 적용가능 할 것이다.

주제어: 비병렬 데이터, 텍스트 스타일 변환, 기계학습, 자연어 처리

1. 서론

텍스트 스타일 변환(Text Style Transfer)은 문장을 원하는 스타일로 변경하는 동시에 주변 콘텐츠 정보들을 일관성 있게 유지하는 것이 목표로 한다. 즉, 문장 내 콘텐츠 정보는 유지하면서 문장의 속성(Attribution)을 변경하는 것이다. 이는 문장 간 스타일 변환(대화 에이전트의 대화 스타일 변환, 격식 및 비격식 문장의 스타일 변환 등), 문장의 속성 전이(개인 정보를 보호하기 위해서 개인적 속성을 혼란스럽게 만드는 것) 등 많은 자연어 처리 분야에 적용될 수 있다.

기존의 시퀀스-투-시퀀스(Sequence-to-sequence)의 대표적인 태스크에는 번역, 대화 시스템 등이 있다[1,2]. 스타일의 정의가 모호하기 때문에 입력 문장과 변환된 스타일 문장을 병렬로 작성한 지도 학습 연구가 많이 진행되었다. 하지만 지도 학습을 위한 병렬 코퍼스를 생성하는 일은 어렵다. 따라서 병렬 데이터가 부족한 문제점을 해결하기 위한 방법이 필요하여 비병렬 데이터의 스타일 변환을 중점으로 연구되고 있다. 본 논문 또한 비병렬 데이터를 이용하여 스타일 변환을 시도한다.

기존의 텍스트 스타일 변환에 대한 연구는 셰익스피어의 문체를 적용한 연구[3], 입력 발화에 대한 공손함의 정도를 다르게 하여 발화[4] 등 생성하는 연구가 진행되고 있다. [5]는 주어진 문장에서 속성을 가지는 단어를 찾은 후 그 단어를 대체하는 방법이고, [6]은 스타일에 대한 판별기를 부착하여 적대적으로 학습하는 방법이다.

본 논문에서 제안하는 스타일 변환기는 비병렬 데이터를 이용하고 트랜스포머(Transformer)를 기반으로 하는 모델이다. 트랜스포머는 완전 연결 자기 집중 뉴럴 네트워크(Fully-connected Self-attention Neural Network)이고, 대부분의 자연어 처리 task에서 최고 성능(State-of-the-art)을 보였다. 우리는 동일한 도메인의 비병렬 데이터를 이용하여 스타일 변환을 적용해보고자 한다. 비병렬 데이터의 스타일 변환 태스크 중 감정 변환에 적용하여 제안 모델의 강인함을 보여준다.

2. 제안 모델

제안하는 스타일 변환 모델 구조는 그림 1과 같다. 트랜스포머로 구성된 문장 생성기(Generator)와 스타일 판별기(Discriminator)를 이용한다. 트랜스포머는 인코더와 디코더에서 스택 자기 주의(Stacked Self-attention)와 포인트-와이즈(Point-wise), 완전 연결 계층(Fully-connected Layer)을 사용한다. 트랜스포머의 강한 능력으로, 우리 모델은 문장의 의미를 잘 보존하면서 문장의 스타일을 변환할 수 있다.

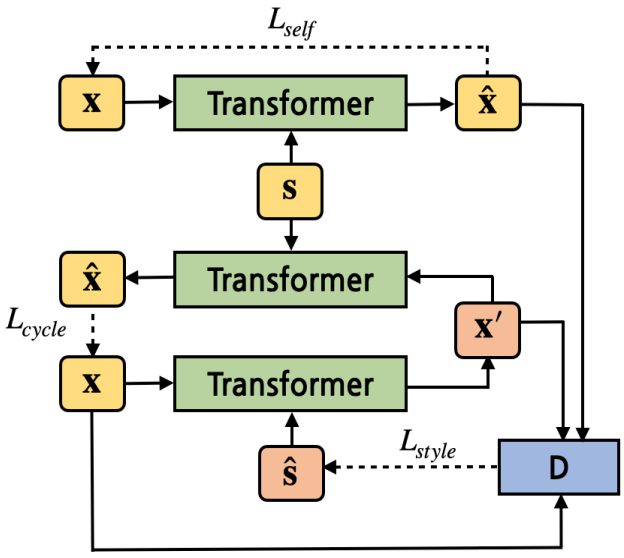

그림 1 제안 모델 구조

각 시간 t에서 입력 문장인 x와 입력 문장의 스타일 변수인 s와 목적 스타일 변수인 $\hat{s}$을 사용한다고 가정한다. 비병렬 코퍼스로 진행하기 때문에 우리는 $s \neq \hat{s}$이고, $f_\theta(x,\hat{s})$를 직접적으로 지도(Supervision)할 수 없다. 따라서 $s \neq \hat{s}$일 때, 입력 문장과 스타일 변수를 이용한 매핑 함수인 $f_\theta(x,s)$를 학습하는 것이 목표이다.

지도 학습을 할 수 있는 방법은 2가지가 있다. 첫 번째는 기존 입력 문장인 x와 목적 스타일 변수인 $\hat{s}$를 이용하여 $x' = \hat{y} = f_\theta(x,\hat{s})$인 문장을 생성한다. 이를 다시 기존 스타일 변수 s를 사용하여 생성하는 문장이 $\hat{x}$이 x가 되도록 한다. x와 $\hat{x}$의 손실함수를 순환 손실함수(Cycle Loss)[7]라 하고, 이러한 방법을 역번역(Back-translation)[8] 방식이라 한다. 두 번째는 스타일 판별기 네트워크(D)를 학습하여 생성 문장의 스타일을 분류한다.

본 논문에서 2.1은 문장 생성기에 대해 설명하고, 2.2에서 스타일 판별기에 대해 설명한다. 또 2.3에서 전체 모델에서 사용하는 손실함수에 대해 설명한다. 마지막으로 2.4에서 각 모듈의 학습 방법을 설명한다.

2.1. 문장 생성기

트랜스포머 인코더인 $Enc(\mathrm{x},\theta_E)$은 입력 문장 $\mathrm{x}=(x_1,x_2,\ldots,x_n)$을 연속적인 표현인 $\mathrm{z}=(z_1,z_2,\ldots,z_n)$에 매핑한다. 그리고 트랜스포머 디코더인 $Dec(\mathrm{z},\theta_D)$는 출력 문장 $\mathrm{y}=(y_1,y_2,\ldots,y_m)$를 위해 연속적으로 확률을 예측하고 식 (1)과 같이 표현된다.

$$z = Enc(\mathrm{x},\theta_E)$$
$$Dec(\mathrm{z}|\theta_D) = p_\theta(\mathrm{y}|\mathrm{x}) = \prod_{t=1}^{m} p_\theta(y_t|z,y_1,\ldots,y_{t-1}) \quad (1)$$

기존 트랜스포머 프레임워크에서 스타일을 컨트롤하는 것을 가능하게 하기 위해 트랜스포머 인코더의 입력에 스타일 임베딩을 추가한다. 따라서 네트워크는 입력 문장 x와 스타일 변수 s 또는 $\hat{s}$를 이용하여 출력 조건의 확률을 계산하여 출력 문장을 생성할 수 있으며 식 (2)과 같이 표현할 수 있다.

$$z = Enc(\mathrm{x},\mathrm{s},\theta_E)$$
$$Dec(\mathrm{z}|\theta_D) = p_\theta(\mathrm{y}|\mathrm{x},\mathrm{s}) = \prod_{t=1}^{m} p_\theta(y_t|z,y_1,\ldots,y_{t-1}) \quad (2)$$

2.2. 스타일 판별기

판별기는 입력 문장의 트랜스포머의 인코더 서브 블록을 4번 수행한 후 선형 층(Linear Layer)을 이용하여 스타일을 분류하는 네트워크를 구성하였다. 판별기의 목적은 입력을 문장으로 받아 해당 문장의 스타일을 분류하는 것이다. 즉, 판별기는 K개의 클래스를 가지는 분류기이며 K개의 다른 스타일을 구별하는 것이다. 판별기의 입력 문장은 총 3개이며 (1) 실제 문장 x, (2) $f_\theta(\mathrm{x},\mathrm{s})$에 의해 생성된 $\hat{\mathrm{x}}$, (3) $f_\theta(\mathrm{x},\hat{\mathrm{s}})$에 의해 생성된 $\hat{\mathrm{x}}$이다. 판별기는 입력 문장에 대해 스타일 레이블을 올바르게 예측하도록 학습한다. 적용한 스타일은 2개이며 긍정, 부정이다.

2.3. 모델의 손실 함수

모델은 총 3가지의 손실함수로, 자기-재생성 손실함수, 순환-재생성 손실함수, 스타일 손실함수를 사용한다. 입력 문장 x와 스타일 변수 s를 이용하여 문장 생성기를 통해 문장 $\hat{\mathrm{x}}$을 생성하고, 식 (3)를 통해 자기-재생성 손실함수를 계산한다.

$$L_{self}(\theta) = -p_\theta(\hat{\mathrm{x}}=\mathrm{x}|\mathrm{x},\mathrm{s}) \quad (3)$$

입력 문장 x와 스타일 변수인 $\hat{s}$를 이용하여 문장 생성기를 통해 문장 x'를 생성하고, 이를 스타일 변수인 $\hat{s}$를 입력하여 문장 $\hat{\mathrm{x}}$를 생성한다. 입력 문장 x와 생성된 문장 $\hat{\mathrm{x}}$의 순환-재생성 손실함수를 계산하고 식은 (4)와 같다. 문장 생성기는 음의 로그 우도(negative log-likelihood)가 최소화하여 입력 문장을 재구성하도록 학습된다.

$$L_{cycle}(\theta) = -p_\phi(\hat{\mathrm{x}}=\mathrm{x}|f_\theta(\mathrm{x},\hat{s}),s) \quad (4)$$

스타일 손실함수는 정답 문장 x와 자기-재생성에 의해 생성된 문장 $\hat{\mathrm{x}}$와 순환-재생성의 중간 결과물인 $\hat{\mathrm{x}}$을 이용하여 입력에 대한 스타일 분류를 진행한다. 스타일 손실함수는 식 (5)과 같다.

$$L_{style}(\theta) = -p_\phi\left(c=\hat{s}|f_\theta(\mathrm{x},\hat{s})\right) \quad (5)$$

2.4. 각 모델 업데이트

제안 모델은 적대적 생성 네트워크(GANs)과 동일하게 문장 생성기와 판별기를 적대적으로 학습을 진행한다.

판별기 학습은 먼저 판별기를 설정한 횟수만큼 진행하여 스타일 손실함수(Style Loss)를 계산한다. 그리고 스타일 손실함수 값을 이용한 경사하강법(Gradient descent)을 수행하여 판별기를 업데이트한다. 이때, 문장 생성기는 그래디언트를 받지 않도록 설정한다.

문장 생성기 학습은 문장 생성기를 설정한 횟수만큼 진행하여 총 3개의 손실 함수를 계산한다. 자기-재생성 손실 함수, 순환-재생성 손실 함수, 스타일 손실함수를 더하여 최종 손실함수 값으로 사용한다. 그 후 최종 손실함수 값을 이용한 경사 하강법을 수행하여 문장 생성기를 업데이트한다. 이 때, 스타일 손실함수에 사용된 판별기는 그래디언트를 받지 않도록 설정한다.

3. 실험 및 실험 설정

제안 모델의 강인함을 보여주기 위해 설정한 태스크는 감정 변환 데이터에 적용하였으며, 이 변환은 긍정문을 부정문으로 또는 부정문을 긍정문으로 변환된 문장을 생성하는 것이다. 감정 변환 실험을 위해 공개된 영어 데이터셋인 Yelp 데이터셋을 이용한다. Yelp 데이터셋은 Yelp 데이터셋 챌린지에서 제공하였으며 긍정 또는 부정으로 레이블링된 레스토랑 및 비지니스 리뷰로 구성되어 있다. 우리는 번역 작업을 통해 한국어 Yelp 데이터셋을 구축하고 이를 이용하였고, 복합 문장은 제외한 단일 문장만을 사용하였다. 표 1은 한국어 Yelp의 문장의 예시이다. 그리고 표 2는 실험에 사용된 데이터셋 통계를 보여준다.

표 1 한국어 Yelp 데이터셋의 예시

긍정 문장	부정 문장
품위 있는 서비스	그것들은 또한 너무 비싼 것 같다.
도전적이지만 재미있는 코스!	매우 무례하고 일단 그들이 돈을 받으면 그들은 윤리가 없다.
데이트하기엔 분위기가 완벽했다.	이곳은 형편없다.
우리는 아마도 한 달에 두 번 여기에 올 것이다.	고객 서비스에 매우 실망했다.

표 2 실험에 사용된 데이터셋 통계

문장 분류	긍정	부정
학습	30,000	30,000
검증	3,000	3,000
평가	1,000	1,000

모델에서 한 문장이 가지는 최대 토큰(wordpiece) 길이가 20이고, 임베딩 사이즈는 256이며, 배치 사이즈는 32로 설정하였다. 문장 생성기 및 판별기의 학습률은 0.0005이다. 옵티마이저(optimizer)는 Adam을 사용하였고, 조기 종료(Early stop)는 5로 설정하였다. 문장 생성기의 학습 시 미니 배치 10번으로 학습하고, 판별기의 학습시 미니 배치 5번으로 설정하고 학습하였다.

평가 방법은 정답이 없는 비병렬 데이터이므로 레퍼런스과 비교하여 성능을 측정할 수 없기 때문에 성능은 3개의 성능 지표를 통하여 수행한다. 스타일 변환에 나타나는 주요한 3가지 특성을 지표로 나타냈으며 이는 스타일 정확도, 컨텐츠 보존, 유창성이다.

스타일 정확도는 각 스타일에 대한 데이터를 이용하여 학습한 fastText[9] 스타일 분류 모델의 정확도를 측정한다.

컨텐츠 보존을 측정하기 위해 우리는 입력 문장과 변환된 문장 사이의 BLEU 점수를 계산한다. 높은 BLEU 점수는 변환된 문장이 입력 문장 내 단어들을 사용하여 주요 콘텐츠가 유지되었다고 볼 수 있다. 이를 'self-BLEU[10]'라고 명칭한다.

유창성 지표로 변환된 문장의 펄플렉시티(perplexity, PPL)를 사용한다. 본 논문에서는 n-gram 언어 모델을 쉽게 구축하고 적용할 수 있는 툴킷인 SRILM[11]을 이용한다. 데이터셋의 각 스타일별로 5-gram LM으로 학습한 모델 생성하여 PPL을 계산한다.

4. 실험 결과 및 분석

각 데이터에 대해서 긍정→긍정, 긍정→부정, 부정→긍정, 부정→부정와 같이 총 4개의 문장이 생성된다. 우리는 상반된 스타일 문장 생성 결과 뿐 아니라 동일한 스타일 문장 생성 결과 또한 평가를 진행한다.

표 3은 스타일 정확도 성능표이다. '상반된 스타일 변환 문장'은 스타일 변수가 긍정→부정, 부정→긍정인 문장들만 사용하였고, '모든 스타일 변환 문장'은 긍정→긍정, 긍정→부정, 부정→긍정, 부정→부정 결과인 모든 문장들을 사용하여 정확도를 측정하였다. 표 4는 self-BLEU와 PPL 성능이다. 입력 문장에 대해 구조는 동일하고 감정 단어들이나 감정 형용사에 대한 변화가 존재하는 문장을 출력하기 때문에 높은 BLEU를 보이며, 긍정→긍정과 부정→부정 문장들은 실제 입력으로 들어가는 문장과 거의 동일한 문장을 출력한다. 이 때 스타일은 동일하지만 다른 표현으로 바꾸려는 결과가 오히려 문법적 오류를 발생하기 때문에 PPL이 낮게 계산된다.

표 3 스타일 정확도 성능표

	상반된 스타일 변환 문장	모든 스타일 변환 문장
정확도	56.9%	75.47%

표 4 self-BLEU와 PPL 성능표

	긍정 →부정	부정 →긍정	긍정 →긍정	부정 →부정
Self-BLEU 1	0.592	0.567	0.745	0.784
Self-	0.515	0.489	0.716	0.761

BLEU 2				
Self-BLEU 3	0.449	0.424	0.683	0.736
Self-BLEU 4	0.393	0.366	0.641	0.705
PPL	798.23	1381.05	177.78	157.76

표 5와 6은 상반된 스타일 문장 생성에 성공한 예시이다. 표 5는 긍정문에서 부정문으로 변환된 문장이고, 표 6은 부정문에서 긍정문으로 변환된 문장이다. 표 5에서 식사에 대한 '맛있었다' 는 긍정의 표현을 '꽝 같았다' 라고 표현을 하였고, '최고의 서비스' 를 '최악의 서비스' 라고 문장 생성을 하였다. 마가리타 메뉴에 대해 '훌륭하다' 고 표현한 긍정 문장에 대해 '건조하다' 고 문장 생성을 하였으며, 샐러드를 '좋아한다' 는 표현을 '끔찍하다' 와 같은 부정적인 표현을 생성하였다.

표 6에서는 '끔찍하다' 라는 표현을 '좋아하다' 라고 표현하였고, 서비스가 '형편없다' 는 표현을 '멋진' 이라고 생성하였다. 또 아이들에게 추천하지 않는다는 내용을 아이들에게 강력히 추천한다고 표현하였다.

표 5 생성 성공 문장의 예(긍정→부정)

입력 문장	출력 문장
그리고 나의 식사는 맛있었다	그리고 나의 식사는 꽝같았다
역대 최고의 정육점 서비스	역대 최악의 정육점 서비스
집 마가리타는 훌륭하게 식사 위에 올랐다	집 마가리타는 건조하게 식사 위에 올랐다
나는 그들의 시그니처 샐러드를 좋아해	나는 그들의 시그니처 샐러드를 끔찍해 나는

표 6 생성 성공 문장의 예(부정→긍정)

입력 문장	출력 문장
끔찍할 정도로 끔찍한 서비스	좋아할 정도로 좋은 서비스
정말 형편없는 고객 서비스	정말 멋진 고객 서비스
아이들에게 추천하지 않는다	아이들에게 추천 훌륭한 강력히

표 7은 한국어 Yelp 데이터셋의 생성 실패의 예시이다. (1)의 경우는 긍정과 부정 모두 똑같은 문장을 생성하는 경우이다. (2)의 경우는 '일이었다구에' 와 같이 생성된 토큰의 조합이 올바르지 않아 문법적 오류를 발생한 경우이다. (3)은 입력 문장과 같이 올바른 문장으로 끝나야 하는데, 문장 도중에 생성이 중단된 경우이며 (4)는 한 문장에 중립과 같은 두 가지 스타일이 들어 있는 부분을 모두 변경해야함에도 불구하고 일부만 변경한 경우를 의미한다. 이러한 생성 실패의 경우들을 해결할 수 있는 방법을 향후 연구로 남겨둔다.

표 7 한국어 Yelp 데이터셋의 생성 실패 문장의 예

분류	입력 문장	출력 문장
(1)	(긍정)두 명의 여행 채식주의자들에게 큰 혜택	(부정)두 명의 여행 채식주의자들에게 큰 혜택
(2)	(긍정)매우 좋은 직원들과 현대적인 사무실	(부정)일이었다구에 직원들과 현대적인 사무실 일이었다
(3)	(긍정)나는 이곳이 정말 좋았다	(부정)나는 이 곳이 정말 나는
(4)	(부정) 오케이 음식 끔찍함 느리고 거만한 서비스	(긍정) 오케이 음식 좋아함 느리고 거만한 서비스

5. 결론

텍스트 스타일 변환(Text Style Transfer)이란, 문장을 원하는 스타일로 변경하는 동시에 주변 콘텐츠 정보들을 일관성 있게 유지하는 것이 목표로 한다. 기존의 텍스트 변환의 연구들은 지도 학습으로 연구가 진행되었다. 하지만 지도 학습을 위한 병렬 코퍼스를 생성하는 일은 어렵다. 따라서 병렬 데이터가 부족한 문제점을 해결하기 위한 방법이 필요하다. 우리는 이를 해결하기 위해 비병렬 데이터를 이용하여 스타일 변환을 시도한다. 비병렬 데이터는 병렬 데이터를 생성하는 것에 비해 시간 또는 비용이 상대적으로 적게 소요된다는 장점이 있으며, 특정 도메인에 대해 데이터를 수집하여 사용할 수 있다.

본 논문에서는 트랜스포머 기반의 문장 생성기와 입력 문장에 대한 스타일을 분류하는 판별기를 이용하여 감정 변환을 적용하며 한국어 Yelp의 리뷰 데이터셋에 실험을 진행하였다. 비병렬 데이터이므로 레퍼런스가 존재하지 않아 직접적인 성능 측정이 어렵다. 그렇기 때문에 스타일 변환의 주요한 3가지 특성인 스타일 정확도, 컨텐츠 보존, 유창성을 성능 지표로 사용하였다.

스타일 정확도는 fastText를 이용한 스타일 분류 모델의 결과를 이용하였고, 정확도는 56.9%를 보였다. 컨텐츠 보존을 측정하기 위해 우리는 입력 문장과 변환된 문장 사이의 self-BLEU 점수를 계산하였으며 BLEU 4 기준으로 긍정→부정 문장은 0.393을 보였고, 부정→긍정 문장은 0.366을 보였다. 유창성은 변환된 문장의 PPL를 계산하였으며 긍정→부정 문장은 798.23을 보였고, 부정→긍정 문장은 1381.05을 보였다.

입력 문장에 대해 구조는 동일하고, 감정 단어들이나 감정 형용사에 대한 변화가 존재하는 문장을 출력함으로써 self-BLEU는 높게 나오지만, 실제 입력으로 들어가는 문장과 컨텐츠는 유지되 문장을 출력한다. 이 때 스타일은 동일하지만 다른 표현으로 바꾸려는 결과가 오히려 문법적 오류를 발생하기 때문에 PPL이 높게 계산된다. 스타일 변환에서 상반되는 스타일에 동일한 문장을 생성하는 문제와 문법적 오류가 존재하였다. 문법적 오류는

생성된 토큰의 잘못된 조합 또는 문장 도중에 생성이 중단된 경우에 해당된다. 또 한 문장에 다중 클래스의 표현이 나타나는 경우 모두 변경하지 않고, 일부만 변경된 경우이다. 우리는 이러한 문제점을 해결하기 위한 연구를 향후 연구로 남겨둔다.

Acknowledgement

이 논문은 2020년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No.2019-0-01755, 마취분야용 의료 딥러닝을 활용한 인공지능(ANES AI) 및 인터랙티브 OCS KIOSK 시스템 개발)

참고문헌

[1] Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. “Sequence to sequence learning with neural networks.”, Advances in neural information processing systems., 2014.

[2] Asri, Layla El, Jing He, and Kaheer Suleman. “A sequence- to-sequence model for user simulation in spoken dialogue systems.”, arXiv prep rint arXiv:1607.00070, 2016.

[3] Jhamtani, Harsh, et al. "Shakespearizing modern language using copy-enriched sequence-to-sequence models.”, arXiv preprint arXiv:1707.01161, 2017.

[4] Santos, Cicero Nogueira dos, Igor Melnyk, and Inkit Padhi. "Fig hting offensive language on social media with unsupervised text style transfer.”, arXiv preprint arXiv:1805.07685, 2018.

[5] Li, Yang hao, et al. "Demystifying neural style transfer.”, arXiv preprint arXiv:1701.01036, 2017.

[6] Shen, Tianxiao, et al. "Style transfer from non-parallel text by cross-alignment.”, Advances in neural information processing systems, 2017.

[7] Zhu, Jun-Yan, Taesung Park, Phillip Isola, and Alexei A. Efros. "U npaired image-to-image translation using cycle-consistent adversarial networks." In *Proceedings of the IEEE international conference on computer vision*, pp. 2223-2232. 2017.

[8] Edunov, Sergey, Myle Ott, Michael Auli, and David Grangier. "Understanding back-translation at scale." *arXiv preprint arXiv:1808.09381* (2018).

[9] Wang, Alex, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. "Glue: A multi-task benchmark and analysis p latform for natural language understanding." *arXiv prep rint arXiv:1804.07461*, 2018.

[10] Zhu, Yaoming, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. "Texygen: A benchmarking platform for text generation models." In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pp. 1097-1100. 2018.

[11] Stolcke, Andreas. "SRILM-an extensible language modeling toolkit.", Seventh international conference on spoken language processing, 2002.