Google Research

AdvAug: Robust Adversarial Augmentation for Neural Machine Translation

Yong Cheng, Lu Jiang, Wolfgang Macherey, Jacob Eisenstein

Introduction

Neural Machine Translation (NMT)

$\mathbf{y}$ It was indeed a miracle that the plane did not touch down at home or hospital.

f → NMT (RNN/CNN/Transformer et al.)

$\mathbf{x}$ 这架飞机没有撞上住家或医院，实在是奇迹。

Neural Machine Translation (NMT)

$\mathbf{y}$ It was indeed a miracle that the plane did not touch down at home or hospital.

f → NMT (RNN/CNN/Transformer et al.)

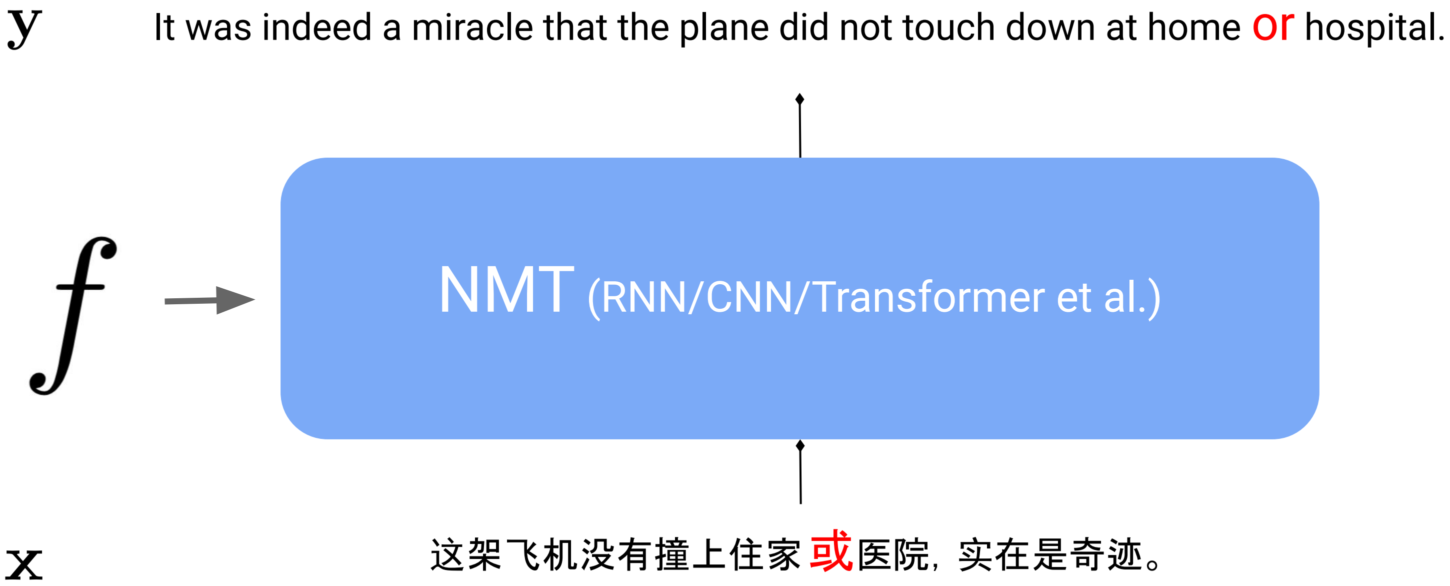
$\mathbf{x}$ 这架飞机没有撞上住家或医院，实在是奇迹。

Training Loss:

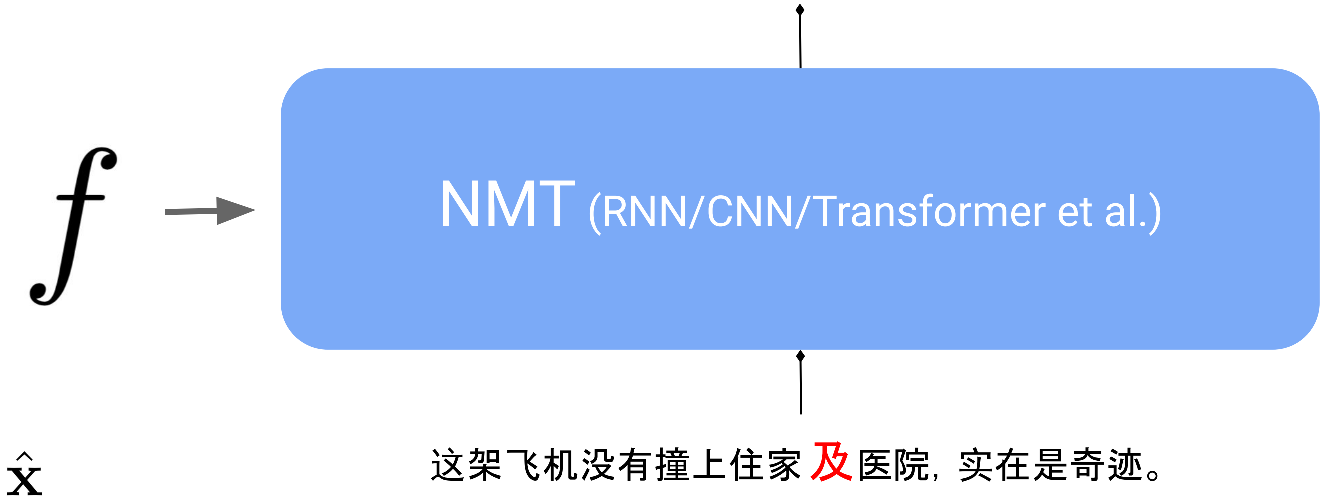
$$\mathcal{L}_{clean}(\boldsymbol{\theta}) = \mathop{\mathbb{E}}_{P_\delta(\mathbf{x},\mathbf{y})}[\ell(f(e(\mathbf{x}), e(\mathbf{y}); \boldsymbol{\theta}), \ddot{\mathbf{y}})]$$

$$P_\delta(\mathbf{x},\mathbf{y}) = \frac{1}{|S|}\sum_{(\mathbf{x}',\mathbf{y}')\in\mathcal{S}} \delta(\mathbf{x}=\mathbf{x}', \mathbf{y}=\mathbf{y}')$$

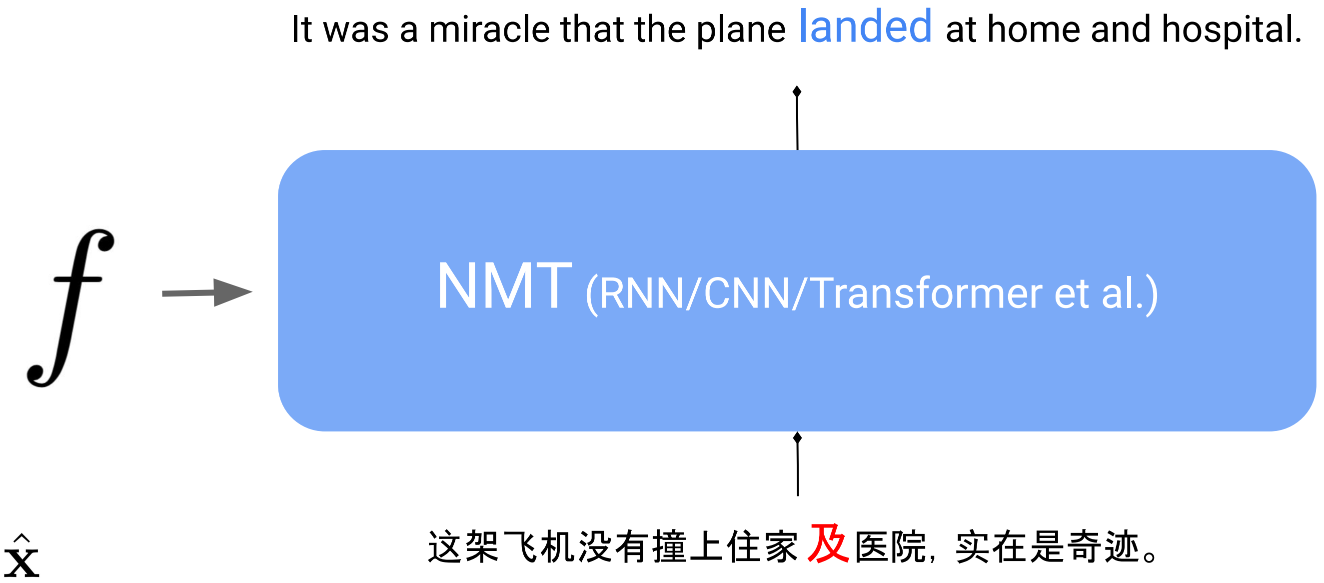
Sensitive to Input Perturbations



Sensitive to Input Perturbations



Sensitive to Input Perturbations



Previous Work

- One potential solution is data augmentation which introduces noise to training examples guided by the principle that the noisy examples are still semantically valid translation pairs.
    - Continuous noise which is modeled as a real-valued vector applied to word embeddings (Miyato et al., 2016, 2017; Cheng et al., 2018; Sano et al., 2019).
    - Discrete noise which adds, deletes, and/or replaces characters or words in the observed sentences (Belinkov and Bisk, 2018; Sperber et al., 2017; Ebrahimi et al., 2018; Michel et al., 2019; Cheng et al., 2019; Karpukhin et al., 2019).

Background Work

- Generating Adversarial Examples for NMT (Cheng et al. 2019).
    - Adversarial examples are generated by solving: $\hat{\mathbf{x}} = \underset{\hat{\mathbf{x}}:\mathcal{R}(\hat{\mathbf{x}},\mathbf{x})\leq\epsilon}{\operatorname{argmax}} \ \ell(f(e(\hat{\mathbf{x}}), e(\mathbf{y}); \boldsymbol{\theta}), \ddot{\mathbf{y}})$

    The set of adversarial examples from $(\mathbf{x}, \mathbf{y})$:

$$A_{(\mathbf{x},\mathbf{y})} = \{(\hat{\mathbf{x}}, \hat{\mathbf{y}}) | \hat{\mathbf{x}} \leftarrow \pi(\mathbf{x}; \mathbf{x}, \mathbf{y}, \xi_{src}), \hat{\mathbf{y}} \leftarrow \pi(\mathbf{y}; \hat{\mathbf{x}}, \mathbf{y}, \xi_{tgt})\},$$

- Data Mixup (Zhang et al. 2018).
    - Given a pair of images $(\mathbf{x}', \mathbf{y}')$ and $(\mathbf{x}'', \mathbf{y}'')$, *mixup* minimizes the sample loss from a vicinity distribution $P_v(\tilde{\mathbf{x}}, \tilde{\mathbf{y}})$ defined in the RGB-label space:

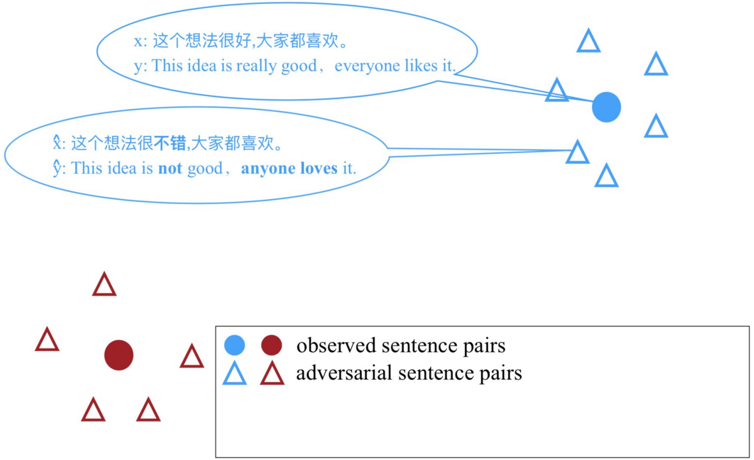
$$\tilde{\mathbf{x}} = \lambda\mathbf{x}' + (1 - \lambda)\mathbf{x}'', \qquad \lambda \sim \mathrm{Beta}(\alpha, \alpha)$$
$$\tilde{\mathbf{y}} = \lambda\mathbf{y}' + (1 - \lambda)\mathbf{y}''.$$

Our work: *AdvAug*

- We introduce a novel *vicinity distribution* to describe the space of adversarial examples centered around each training example.

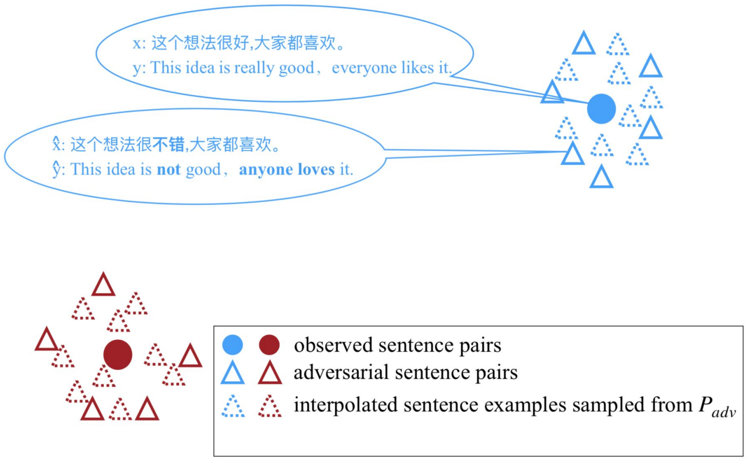
Our work: *AdvAug*

- We introduce a novel *vicinity distribution* to describe the space of adversarial examples centered around each training example.
  - First generate adversarial sentences in the discrete data space,



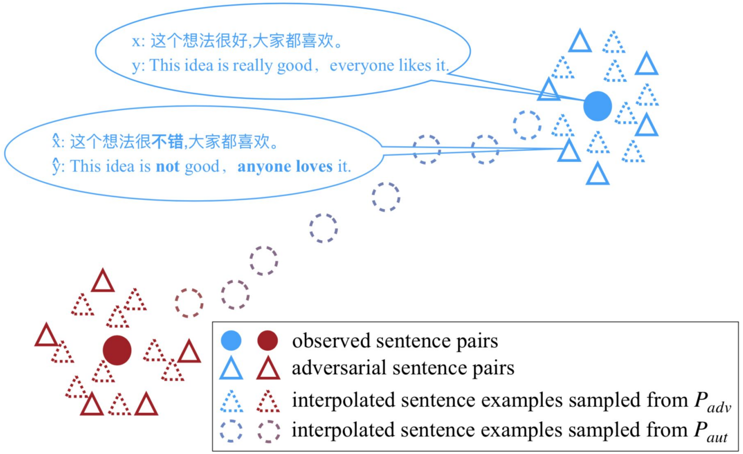
Our work: *AdvAug*

- We introduce a novel *vicinity distribution* to describe the space of adversarial examples centered around each training example.
  - First generate adversarial sentences in the discrete data space, and then sample *virtual* adversarial sentences from the vicinity distribution according to their interpolated embeddings.



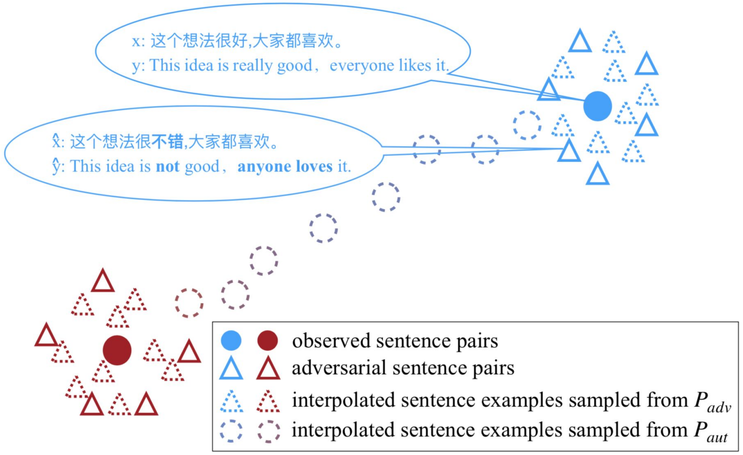
Our work: *AdvAug*

- We introduce a novel *vicinity distribution* to describe the space of adversarial examples centered around each training example.
  - First generate adversarial sentences in the discrete data space, and then sample *virtual* adversarial sentences from the vicinity distribution according to their interpolated embeddings
- We also use a similar *vicinity distribution* over the authentic training data.



Our work: *AdvAug*

- We introduce a novel *vicinity distribution* to describe the space of adversarial examples centered around each training example.
  - First generate adversarial sentences in the discrete data space, and then sample *virtual* adversarial sentences from the vicinity distribution according to their interpolated embeddings
- We also use a similar *vicinity distribution* over the authentic training data.
- We train on the embeddings sampled from the two *vicinity distributions*.



Approach

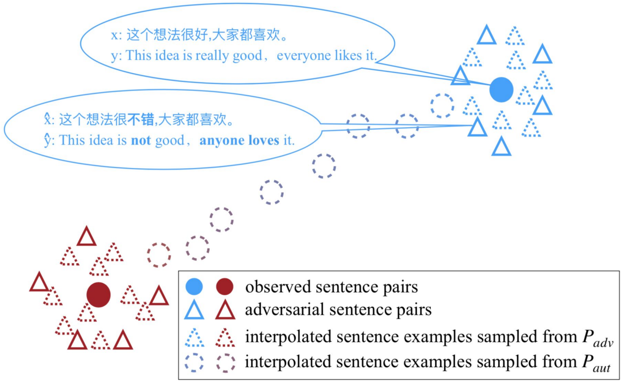
AdvAug

- We propose two *vicinity distributions* to reinforce the model over virtual data points surrounding the observed examples in the training set.
  - P_{adv} for the (dynamically generated) adversarial examples

$$P_{adv}(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}) = \frac{1}{|\mathcal{S}|} \sum_{(\mathbf{x},\mathbf{y})\in\mathcal{S}} \mu_{adv}(\tilde{\mathbf{x}}, \tilde{\mathbf{y}} | A_{(\mathbf{x},\mathbf{y})})$$

  - P_{aut} for the (observed) *authentic* examples

$$P_{aut}(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}) = \frac{1}{|\mathcal{S}|} \sum_{(\mathbf{x},\mathbf{y})\in\mathcal{S}} \mu_{aut}(\tilde{\mathbf{x}}, \tilde{\mathbf{y}} | \mathbf{x}, \mathbf{y})$$



- Training objective combines two losses on them: $\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}}\{\mathcal{L}_{aut}(\boldsymbol{\theta}) + \mathcal{L}_{adv}(\boldsymbol{\theta})\}$

How to Compute μ_{adv}

- μ_{adv} in P_{adv} can be calculated from:

$$\mu_{adv}(\tilde{\mathbf{x}}, \tilde{\mathbf{y}} | A_{(\mathbf{x},\mathbf{y})}) = \frac{1}{|A_{(\mathbf{x},\mathbf{y})}|^2} \sum_{(\mathbf{x}',\mathbf{y}')\in A_{(\mathbf{x},\mathbf{y})}} \sum_{(\mathbf{x}'',\mathbf{y}'')\in A_{(\mathbf{x},\mathbf{y})}} \mathbb{E}_{\lambda}[\delta(e(\tilde{\mathbf{x}}) = m_\lambda(\mathbf{x}', \mathbf{x}''), e(\tilde{\mathbf{y}}) = m_\lambda(\mathbf{y}', \mathbf{y}'')]$$

- The convex combination $m_\lambda(\mathbf{x}', \mathbf{x}'')$ is applied over the aligned embeddings by padding tokens to the end of the shorter sentence.

$$e(\tilde{x}_i) = \lambda e(x'_i) + (1-\lambda) e(x''_i), \forall i \in [1, |\tilde{\mathbf{x}}|] \qquad \lambda \sim \mathrm{Beta}(\alpha, \alpha)$$

Loss for P_{adv}

- The translation loss on vicinal adversarial examples can be integrated over P_{adv}

$$\mathcal{L}_{adv}(\boldsymbol{\theta}) = \mathop{\mathbb{E}}_{P_{adv}(\tilde{\mathbf{x}},\tilde{\mathbf{y}})}[\ell(f(e(\tilde{\mathbf{x}}), e(\tilde{\mathbf{y}}); \boldsymbol{\theta}), \boldsymbol{\omega})]$$

- Two techniques are used for computing it:
  - Minimize the KL-divergence between the model predictions at the word level .

$$\sum_{j=1}^{|\mathbf{y}|} D_{KL}(f_j(e(\mathbf{x}), e(\mathbf{y}); \hat{\boldsymbol{\theta}}) || f_j(e(\tilde{\mathbf{x}}), e(\tilde{\mathbf{y}}); \boldsymbol{\theta})) \text{ so } \boldsymbol{\omega} = f(e(\mathbf{x}), e(\mathbf{y}); \hat{\boldsymbol{\theta}})$$

  - Employ curriculum learning to do importance sampling.

$$\mathbf{L} = \frac{1}{\sum_{i=1}^{m} I(\ell_i > \eta)} \sum_{i=1}^{m} I(\ell_i > \eta)\ell_i$$

Loss for P_{aut}

- The translation loss on authentic data can be compute as

$$\mathcal{L}_{aut}(\boldsymbol{\theta}) = \mathop{\mathbb{E}}_{P_{aut}(\tilde{\mathbf{x}}, \tilde{\mathbf{y}})} [\ell(f(e(\tilde{\mathbf{x}}), e(\tilde{\mathbf{y}}); \boldsymbol{\theta}), \tilde{\boldsymbol{\omega}})]$$

- μ_{aut} in the vicinity distribution P_{aut} is

$$\mu_{aut}(\tilde{\mathbf{x}}, \tilde{\mathbf{y}} | \mathbf{x}, \mathbf{y}) = \frac{1}{|\mathcal{S}|} \sum_{(\mathbf{x}', \mathbf{y}') \in \mathcal{S}} \mathop{\mathbb{E}}_{\lambda}[\ \delta(e(\tilde{\mathbf{x}}) = m_{\lambda}(\mathbf{x}, \mathbf{x}'), e(\tilde{\mathbf{y}}) = m_{\lambda}(\mathbf{y}, \mathbf{y}'),\ \tilde{\boldsymbol{\omega}} = m_{\lambda}(\boldsymbol{\omega}, \boldsymbol{\omega}'))]$$

  - λ is sampled twice, a constant 1.0 and a sample from a Beta distribution.
  - $\boldsymbol{\omega}$ is also interpolated.

Experiments

Results on Chinese-English Translation

Method	Loss Config	MT06	MT02	MT03	MT04	MT05	MT08
Vaswani et al.	L_{clean}	44.57	45.49	44.55	46.20	44.96	35.11
Miyato et al.	-	45.28	45.95	44.68	45.99	45.32	35.84
Sano et al.	-	45.75	46.37	45.02	46.49	45.88	35.90
Cheng et al.	-	46.95	47.06	46.48	47.39	46.58	37.38
Sennrich et al.	-	46.39	47.31	47.10	47.81	45.69	36.43
Ours	L_{mixup}	**45.12**	**46.32**	**44.81**	**46.61**	**46.08**	**36.00**
	L_{aut}	**46.73**	**46.79**	**46.13**	**47.54**	**46.88**	**37.21**
	$L_{clean} + L_{adv}$	**47.89**	**48.53**	**48.73**	**48.60**	**48.76**	**39.03**
	$L_{aut} + L_{adv}$	**49.26**	**49.03**	**47.96**	**48.86**	**49.88**	**39.63**
Ours + BT	$L_{aut} + L_{adv}$	**49.98**	**50.34**	**49.81**	**50.61**	**50.72**	**40.45**

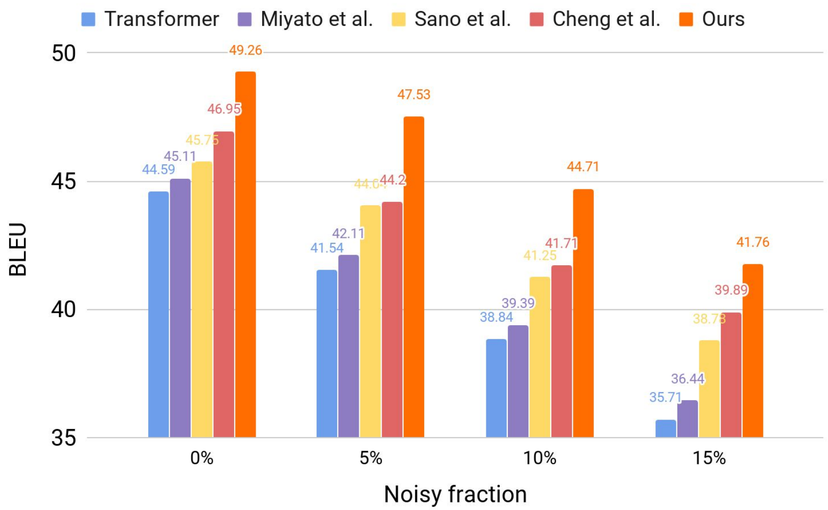
Results on English-French and English-German Translation

Method	Loss Config.	English-French		English-German	
		test2013	test2014	test2013	test2014
Vaswani et al.	L_{clean}	40.78	37.57	25.80	27.30
Sano et al.	-	41.68	38.72	25.97	27.46
Cheng et al.	-	41.76	39.46	26.34	28.34
Ours	L_{mixup}	**40.78**	**38.11**	26.28	28.08
	L_{aut}	**41.49**	**38.74**	**26.33**	**28.58**
	$L_{aut} + L_{adv}$	**43.03**	**40.91**	**27.20**	**29.57**

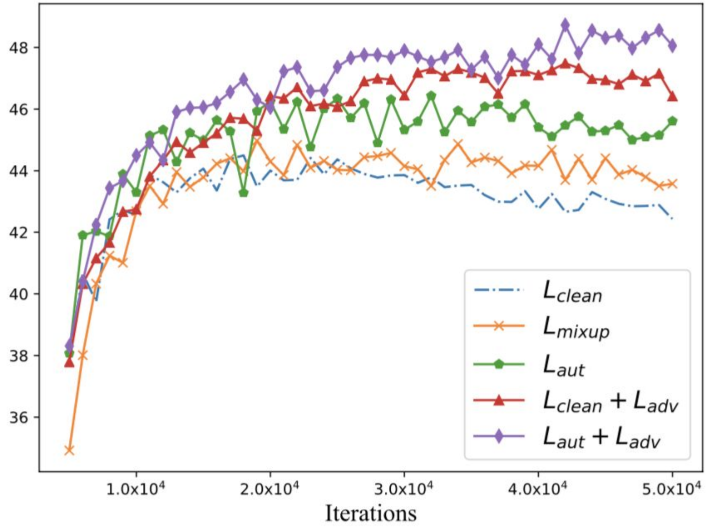
Effect of α in Beta Distribution

Loss	0.2	0.4	4	8	32
L_{mixup}	45.28	45.48	45.64	45.09	-
L_{aut}	45.95	45.92	46.70	46.73	46.54
$L_{aut} + L_{adv}$	47.06	46.88	47.60	47.89	47.81

Robustness to Noisy Inputs and Overfitting



Results on artificial noisy inputs.



BLEU scores over iterations.

Conclusions

Conclusions

- We have presented an approach to augment the training data of NMT models by introducing a new vicinity distribution defined over the interpolated embeddings of adversarial examples and authentic examples.
- We design an augmentation algorithm over the virtual sentences sampled from both of the vicinity distributions in sequence-to-sequence NMT model training.

Thanks