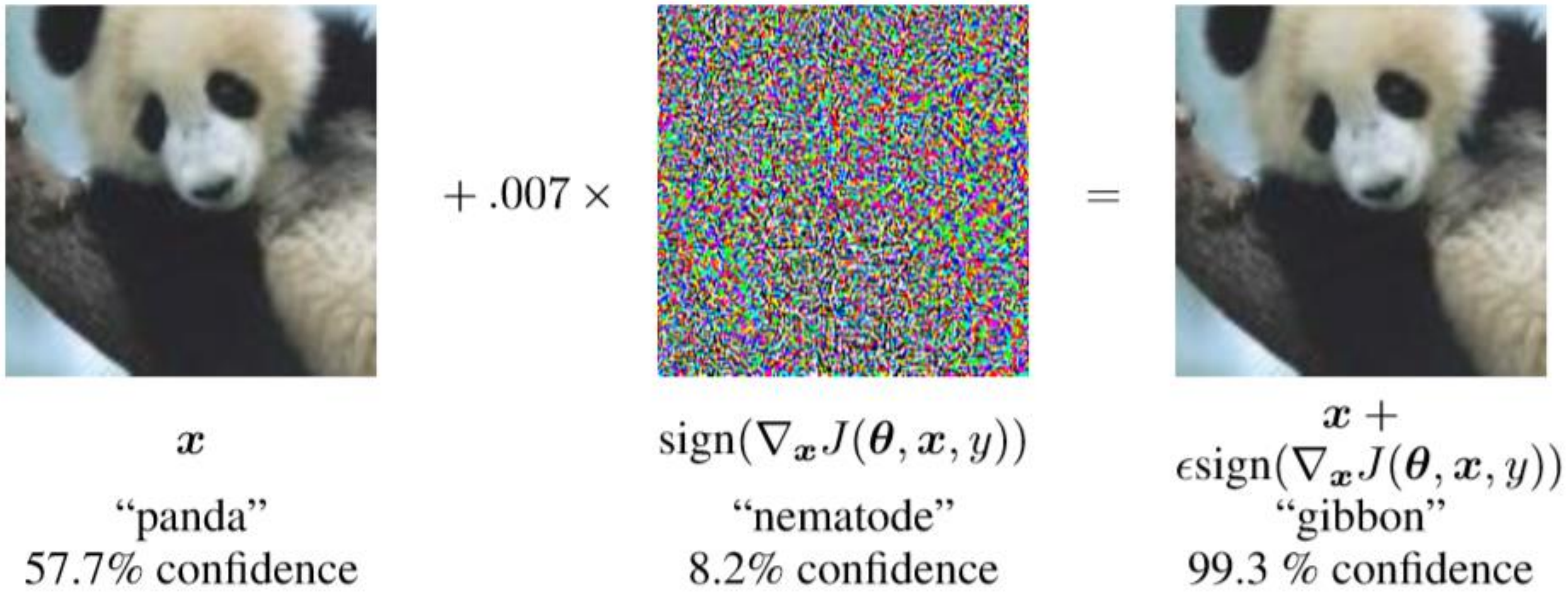
Adversarial Examples in Natural Language Processing

Zijing Ou

Adversarial Examples



Two cores in common:

- the perturbations are small
- the ability of fooling DNN models

Why does it work?

The primary cause of neural networks' vulnerability to adversarial perturbation is their linear nature [1]

adversarial input

$$\tilde{\boldsymbol{x}} = \boldsymbol{x} + \boldsymbol{\eta} \quad ||\eta||_\infty < \epsilon,$$

Consider the dot product

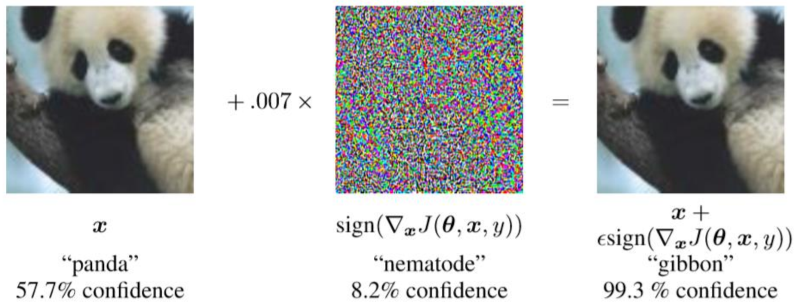
$$\boldsymbol{w}^\top \tilde{\boldsymbol{x}} = \boldsymbol{w}^\top \boldsymbol{x} + \boldsymbol{w}^\top \boldsymbol{\eta}.$$

If w has n dimensions and the average magnitude of an element of the weight vector is m

$$\boldsymbol{w}^\top \boldsymbol{\eta} \propto \epsilon m n \qquad \eta = \text{sign}(\boldsymbol{w})$$

[1] ICLR15: EXPLAINING AND HARNESSING ADVERSARIAL EXAMPLES

How to fix it?



Adversarial Training [1]

$$-\log p(y \mid \boldsymbol{x} + \boldsymbol{r}_{\text{adv}}; \boldsymbol{\theta}) \text{ where } \boldsymbol{r}_{\text{adv}} = \underset{\boldsymbol{r}, \|\boldsymbol{r}\| \le \epsilon}{\arg\min} \log p(y \mid \boldsymbol{x} + \boldsymbol{r}; \hat{\boldsymbol{\theta}})$$

With a linear approximation

$$\boldsymbol{r}_{\text{adv}} = -\epsilon \boldsymbol{g} / \|\boldsymbol{g}\|_2 \text{ where } \boldsymbol{g} = \nabla_{\boldsymbol{x}} \log p(y \mid \boldsymbol{x}; \hat{\boldsymbol{\theta}})$$

The objective function

$$\tilde{J}(\boldsymbol{\theta}, \boldsymbol{x}, y) = \alpha J(\boldsymbol{\theta}, \boldsymbol{x}, y) + (1 - \alpha) J(\boldsymbol{\theta}, \boldsymbol{x} + \epsilon \text{sign}\left(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y)\right))$$

[1] ICLR15: EXPLAINING AND HARNESSING ADVERSARIAL EXAMPLES

Virtual Adversarial Training [2]

$$\mathrm{KL}[p(\cdot \mid \boldsymbol{x}; \hat{\boldsymbol{\theta}}) || p(\cdot \mid \boldsymbol{x} + \boldsymbol{r}_{\text{v-adv}}; \boldsymbol{\theta})]$$

$$\text{where } \boldsymbol{r}_{\text{v-adv}} = \underset{\boldsymbol{r}, \|\boldsymbol{r}\| \leq \epsilon}{\arg\max} \, \mathrm{KL}[p(\cdot \mid \boldsymbol{x}; \hat{\boldsymbol{\theta}}) || p(\cdot \mid \boldsymbol{x} + \boldsymbol{r}; \hat{\boldsymbol{\theta}})]$$

With a approximation

$$\boldsymbol{r}_{\text{v-adv}} = \epsilon \boldsymbol{g} / \|\boldsymbol{g}\|_2 \text{ where } \boldsymbol{g} = \nabla_{\boldsymbol{s}+\boldsymbol{d}} \mathrm{KL}\left[p(\cdot \mid \boldsymbol{s}; \hat{\boldsymbol{\theta}}) || p(\cdot \mid \boldsymbol{s} + \boldsymbol{d}; \hat{\boldsymbol{\theta}})\right]$$

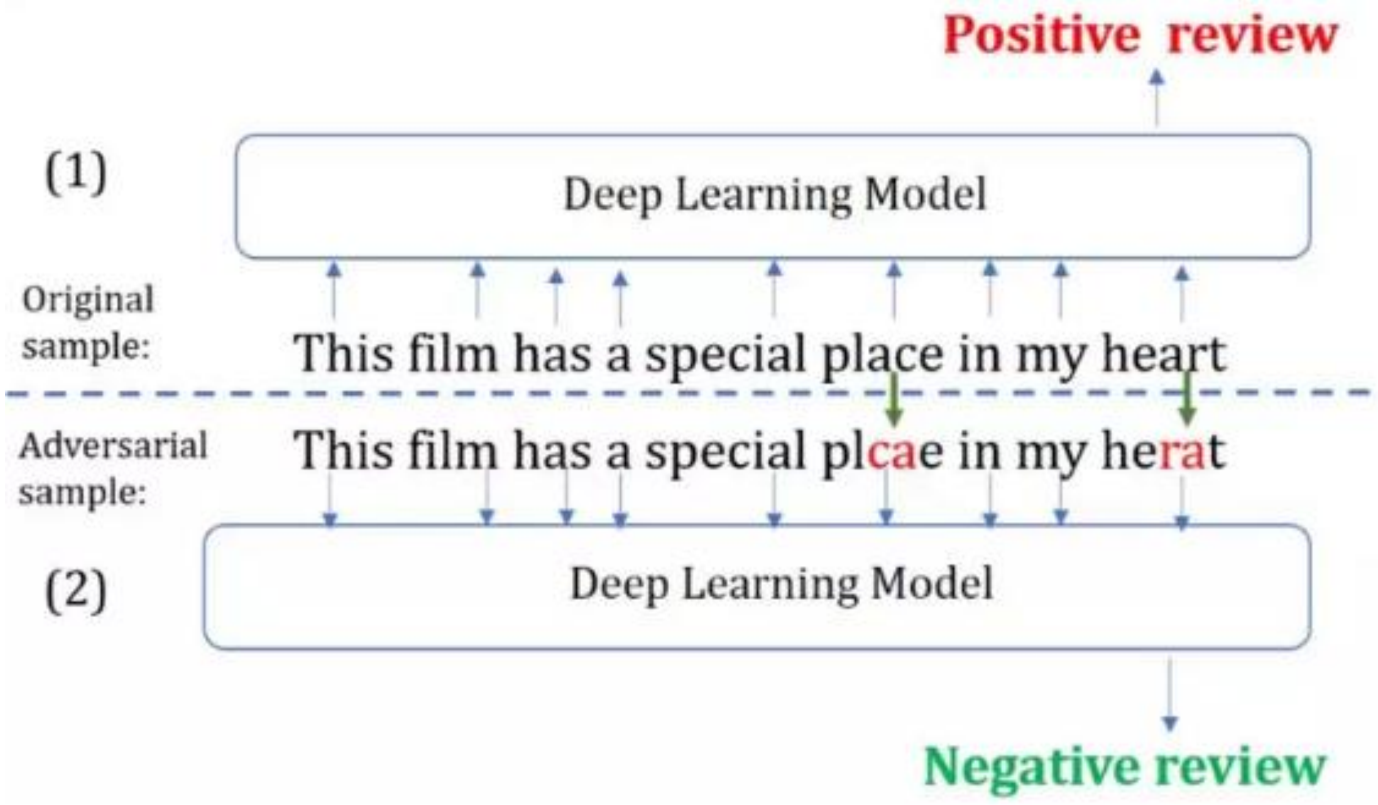
Where **d** is a small random vector with the same dimension as **s**

The objective function

$$\frac{1}{N} \sum_{n=1}^{N} \log p(y^{(n)} | x^{(n)}, \theta) + \lambda \frac{1}{N} \sum_{n=1}^{N} \mathrm{LDS}(x^{(n)}, \theta)$$

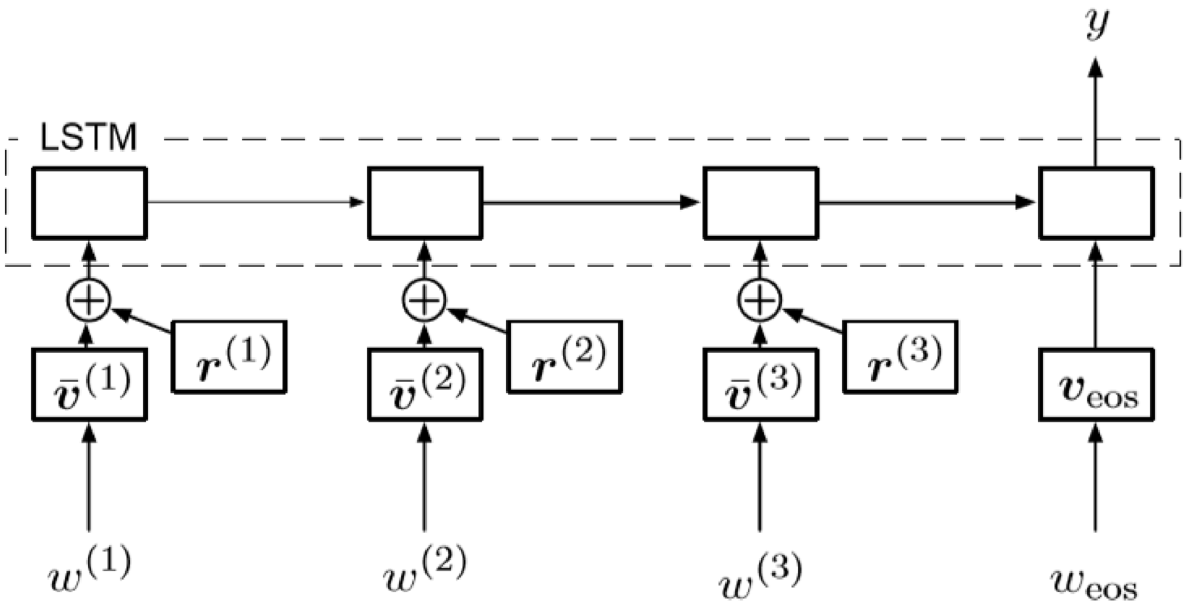
[2] ICLR16: Distributional smoothing with virtual adversarial training

Can it work in text?



We cannot calculate the perturbed inputs for tasks in the NLP field since the inputs consist of **discrete symbols**, which are not a continuous space used in image processing

Applying AdvT to word embedding space [3]



$$\bar{\boldsymbol{v}}_k = \frac{\boldsymbol{v}_k - \mathrm{E}(\boldsymbol{v})}{\sqrt{\mathrm{Var}(\boldsymbol{v})}} \text{ where } \mathrm{E}(\boldsymbol{v}) = \sum_{j=1}^{K} f_j \boldsymbol{v}_j, \mathrm{Var}(\boldsymbol{v}) = \sum_{j=1}^{K} f_j \left(\boldsymbol{v}_j - \mathrm{E}(\boldsymbol{v})\right)^2$$

where f_i is the frequency of the i-th word, calculated within all training examples.

$$r^{(n)} = r_{adv}^{(n)} \ or \ r_{v-adv}^{(n)}$$

[3] ICLR17: Adversarial training methods for semi-supervised text classification

Drawback of applying AdvT to word embedding space

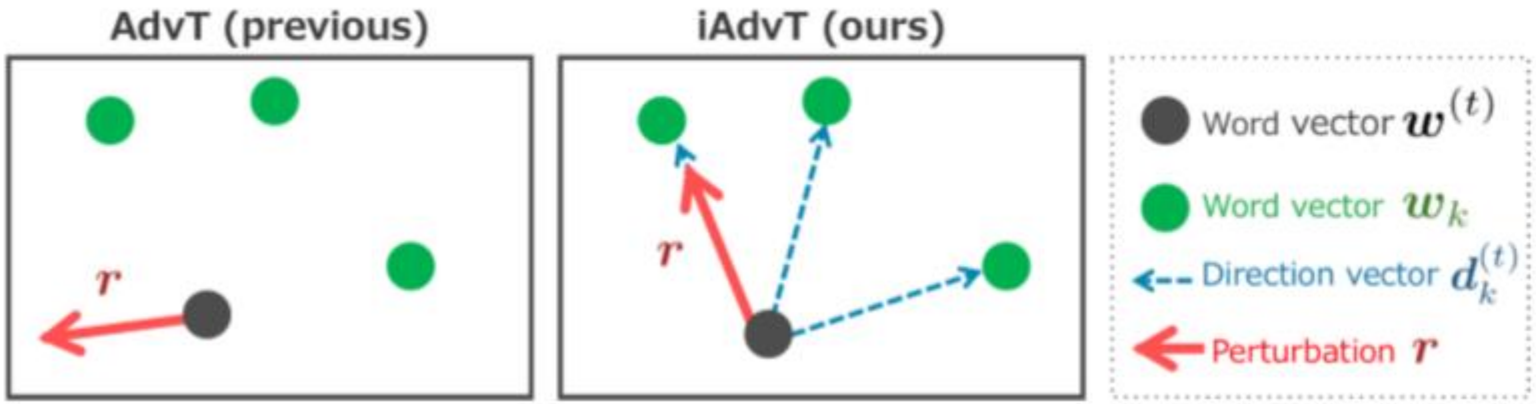
lacks interpretability!!!

- It abandons the generation of adversarial examples interpretable by people.
- We exclusively regard it as a regularizer to stabilize the model.
- It can’t generate adversarial examples.

A Trade-Off

well-formed **vs** low-cost (gradient-based)

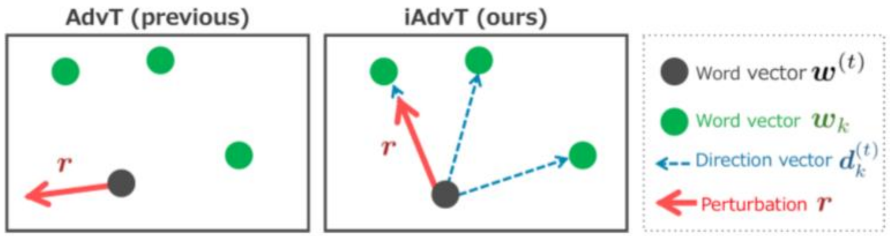
Interpretable Adversarial Perturbation in Embedding Space [4]



How to realize that?

[4] IJCAI18: Interpretable Adversarial Perturbation in Input Embedding Space for Text

Interpretable Adversarial Perturbation in Embedding Space



We define direction vector $d_k^{(t)}$

$$d_k^{(t)} = \frac{\tilde{d}_k^{(t)}}{||\tilde{d}_k^{(t)}||_2}, \quad \text{where} \quad \tilde{d}_k^{(t)} = w_k - w^{(t)}.$$

Let $\alpha^{(t)}$ be a |V|-dimensional vector, and let $\alpha_k^{(t)}$ be the k-th factor of $\alpha^{(t)}$

Then, we define

$$r(\alpha^{(t)}) = \sum_{k=1}^{|\mathcal{V}|} \alpha_k^{(t)} d_k^{(t)}.$$

$$\tilde{X}_{+r(\alpha)} = \left(w^{(t)} + r(\alpha^{(t)})\right)_{t=1}^{T}$$

$$\alpha_{\mathtt{iAdvT}} = \underset{\alpha, ||\alpha|| \leq \epsilon}{\operatorname{argmax}} \left\{ \ell(\tilde{X}_{+r(\alpha)}, \tilde{Y}, \mathcal{W}) \right\}$$ Approximately, $$\alpha_{\mathtt{iAdvT}}^{(t)} = \frac{\epsilon g^{(t)}}{||g||_2}, \quad g^{(t)} = \nabla_{\alpha^{(t)}} \ell(\tilde{X}_{+r(\alpha)}, \tilde{Y}, \mathcal{W})$$

The Loss function is

$$\mathcal{J}_{\mathtt{iAdvT}}(\mathcal{D}, \mathcal{W}) = \frac{1}{|\mathcal{D}|} \sum_{(\tilde{X}, \tilde{Y}) \in \mathcal{D}} \ell(\tilde{X}_{+r(\alpha_{\mathtt{iAdvT}})}, \tilde{Y}, \mathcal{W})$$

But, it still has some problem……

- It is difficult to make small perturbations along the direction of gradients
- The fluency of the generated examples cannot be guaranteed

Fortunately, there are some excellent models in 9102….

- ACL19: Generating Fluent Adversarial Examples for Natural Languages
- ACL19: Generating Natural Language Adversarial Examples through Probability Weighted Word Saliency

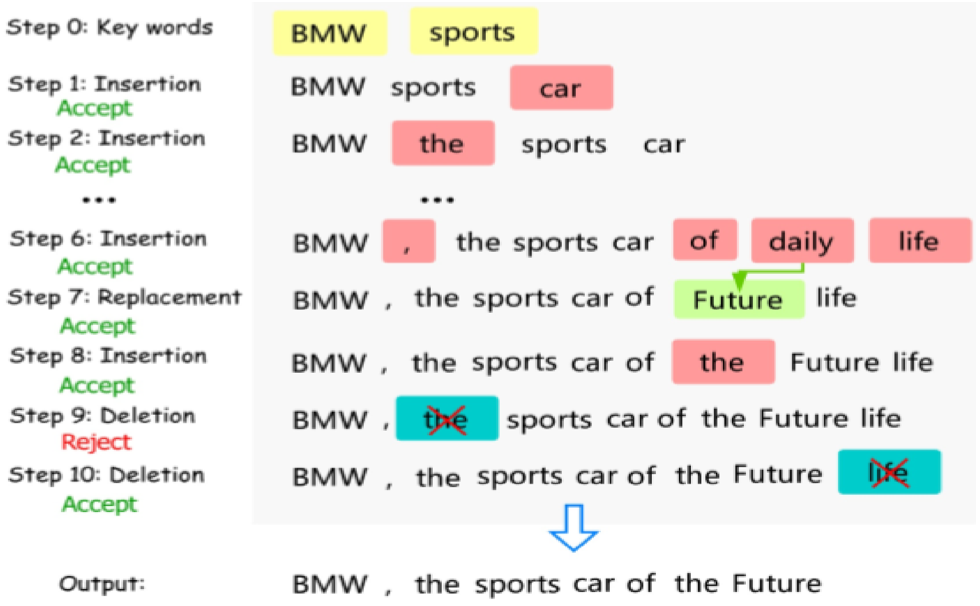
They are both based on statistical models.

CGMH [5] ★★★★★

Motivation

RNN-based language generation techniques are non-trivial to impose constraints

- Hard constraints, such as the mandatory inclusion of certain keywords in the output sentences
- Soft constraints, such as requiring the generated sentences to be semantically related to a certain topic



[5] AAAI19: CGMH Constrained Sentence Generation by Metropolis-Hastings Sampling

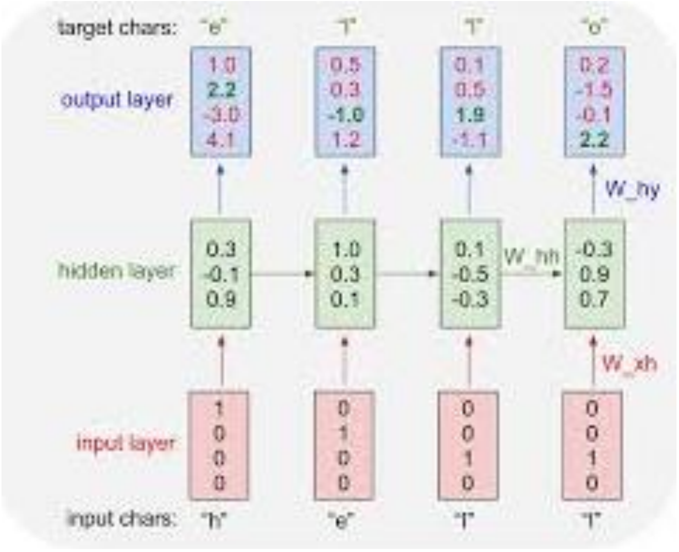
CGMH [5] ★★★★★

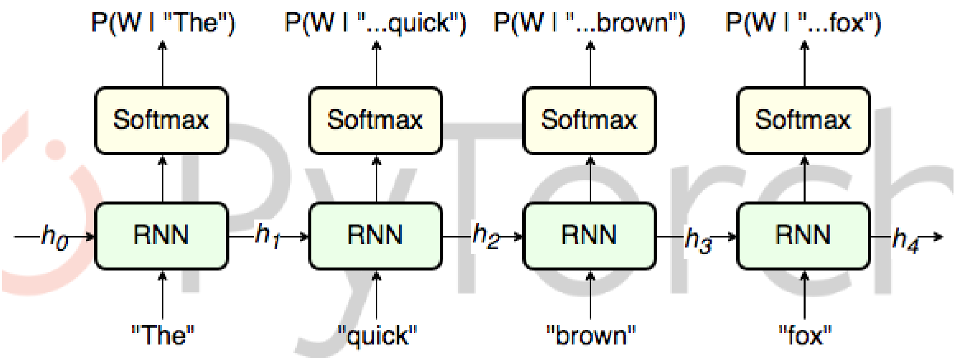
Language model

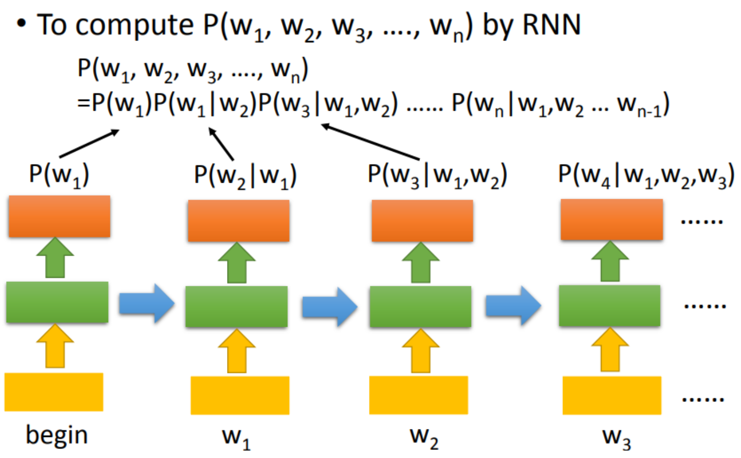
A statistical language model is a probability contribution over sequences of words.

$$P(w_1, w_2, \ldots, w_n) = P(w_1)P(w_2|w_1) \cdots P(w_n|w_1, \ldots, w_{n-1})$$

RNN-based Language model







CGMH [5] ★★★★★

detail balance condition

When the probability transition matrix of aperiodic Markov chains satisfies

$$\pi(i)p(j|i) = \pi(j)p(i|j)$$

The final state $\pi(.)$ is the stable distribution

Metropolis Hastings(MH) Sampling

1. Initialise x^0
2. For i $= 0$ to $N-1$

   $u \sim U(0,1)$

   $x^* \sim q(x^*|x^{(i)})$

   if $u < \alpha(x^*) = \min\left(1, \frac{\pi(x^*)q(x|x^*)}{\pi(x)q(x^*|x)}\right)$ $\quad x^{(i+1)} = x^*$

   else

   $x^{(i+1)} = x^{(i)}$

The MH framework is flexible, cause

- The **proposal distribution** could be **arbitrary**, as long as the Markov chain is irreducible and aperiodic
- The **stationary distribution** could be **arbitrary**, because MH algorithm can guarantee detail balance condition

How to propose proposal & stationary distribution

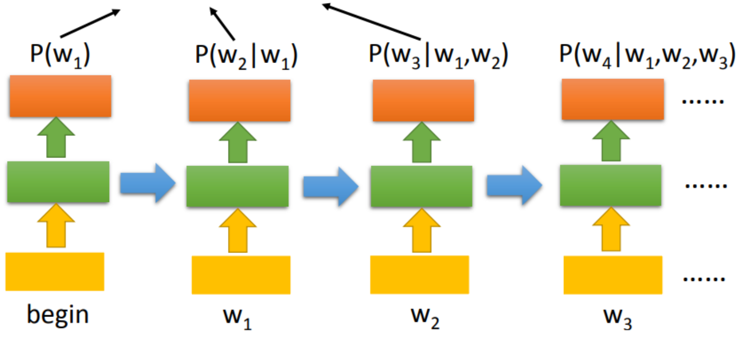
CGMH [5] ★★★★★

proposal distribution

$$[p_{replace}, p_{insert}, p_{delete}] = [\frac{1}{3}, \frac{1}{3}, \frac{1}{3}]$$

- To compute $P(w_1, w_2, w_3, \ldots, w_n)$ by RNN

$P(w_1, w_2, w_3, \ldots, w_n)$
$=P(w_1)P(w_1|w_2)P(w_3|w_1,w_2) \ldots\ldots P(w_n|w_1,w_2 \ldots w_{n-1})$



Replacement

The sentence at the current step is

$$\mathrm{x} = [w_1, \cdots, w_{m-1}, w_m, w_{m+1}, \cdots, w_n]$$

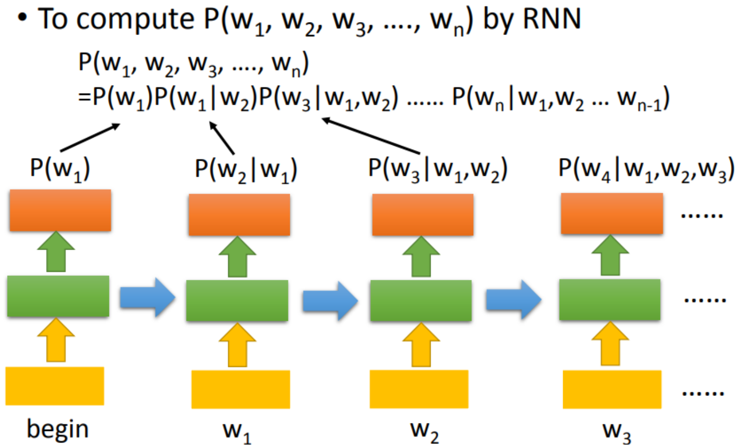
Choose a new word for the m-th position by the conditional probability

$$g_{\text{replace}}(\mathrm{x}'|\mathrm{x}) = \pi(w_m^* = w^c|\mathrm{x}_{-m}) = \frac{\pi(w_1, \cdots, w_{m-1}, w^c, w_{m+1}, \cdots, w_n)}{\sum_{w \in \mathcal{V}} \pi(w_1, \cdots, w_{m-1}, w, w_{m+1}, \cdots, w_n)}$$

However, it is difficult to compute $\pi(w_m^* = w^c|w_{-m})$ for all $w^c \in V$

CGMH [5] ★★★★★

Replacement



Build a **pre-selector** Q to discard w_c with low forward or backward probability

$$Q(w^c) = \min(\, \pi(w_1, \ldots, w_{m-1}, w_m^* = w^c), \\ \pi(w_m^* = w^c, w_{m+1}, \ldots, w_n))$$

Q is easy to compute by a **forward** and a **backward** language model, and $\pi(w_1, \cdots, w_{m-1}, w^c, w_{m+1}, \cdots, w_n)$ is no greater than Q

After pre-selection, we compute the conditional probability of selected words by

$$g_{\text{replace}}(\mathrm{x}'|\mathrm{x}) = \pi(w_m^* = w^c|\mathrm{x}_{-m}) = \frac{\pi(w_1, \cdots, w_{m-1}, w^c, w_{m+1}, \cdots, w_n)}{\sum_{w \in \mathcal{V}} \pi(w_1, \cdots, w_{m-1}, w, w_{m+1}, \cdots, w_n)}$$

Finally, sample a word for replacement

CGMH [5] ★★★★★

Insertion

First insert a special token, placeholder <PHD>

Then use $g_{replace}(.)$ to sample a real word to replace the placeholder

$$g_{\text{replace}}(\mathrm{x}'|\mathrm{x}) = \pi(w_m^* = w^c|\mathrm{x}_{-m}) = \frac{\pi(w_1, \cdots, w_{m-1}, w^c, w_{m+1}, \cdots, w_n)}{\sum_{w \in \mathcal{V}} \pi(w_1, \cdots, w_{m-1}, w, w_{m+1}, \cdots, w_n)}$$

Hence, $g_{insert}(.)$ is similar to $g_{replace}(.)$

CGMH [5] ★★★★★

deletion

Suppose

$$\mathrm{x} = [w_1, \cdots, w_{m-1}, w_m, w_{m+1}, \cdots, w_n]$$

we are about to delete the word w_m, then

$g_{delete}(x'|x_{t-1})$ equals 1 if $x' = [w_1, \ldots, w_{m-1}, w_{m+1}, \ldots, w_n]$, or 0 for other sentences

Notably, insertion and deletion ensure the **ergodicity** of the Markov chain

CGMH [5] ★★★★★

stationary distribution

Hard Constraints

$$\pi(\mathrm{x}) \propto p_{\mathrm{LM}}(\mathrm{x}) \cdot \mathcal{X}_{\mathrm{keyword}}(\mathrm{x})$$

- p_{LM} is a general sentence probability computed by a language model,
- $x_{keyword}$ is the indicator function showing if the keywords are included in the generated sentence

$x_{keyword} = 1$ if all constraints are satisfied (keywords appearing in the sentence), or 0 otherwise

CGMH [5] ★★★★★

stationary distribution

Soft Constraints

$$\pi(\mathrm{x}) \propto p_{\mathrm{LM}}(\mathrm{x}) \cdot \mathcal{X}_{\mathrm{match}}(\mathrm{x}|\mathrm{x}_*)$$

- $p_{LM}(x)$ is a general sentence probability computed by a language model
- $x_{match}(x|x_*)$ is a matching score

We have several choices for $x_{match}(x|x_*)$

- **Keyword matching** (KW) as a soft constraint
- **Word embedding similarity** as a soft constraint
- **Skip-thoughts similarity**(ST) as a soft constraint

CGMH [5] ★★★★★

Acceptance Rate

1. Initialise x^0
2. For i = 0 to $N-1$

   $u \sim U(0,1)$

   $x^* \sim q(x^*|x^{(i)})$

   if $u < \alpha(x^*) = \min\left(1, \frac{\pi(x^*)q(x|x^*)}{\pi(x)q(x^*|x)}\right)$ $x^{(i+1)} = x^*$

   else

   $x^{(i+1)} = x^{(i)}$

$$g_{\text{replace}}(\mathrm{x}'|\mathrm{x}) = \pi(w_m^* = w^c|\mathrm{x}_{-m}) = \frac{\pi(w_1, \cdots, w_{m-1}, w^c, w_{m+1}, \cdots, w_n)}{\sum_{w \in \mathcal{V}} \pi(w_1, \cdots, w_{m-1}, w, w_{m+1}, \cdots, w_n)}$$

$$A^*_{\text{replace}}(x'|x) = \frac{p_{\text{replace}} \cdot g_{\text{replace}}(\mathrm{x}|\mathrm{x}') \cdot \pi(x')}{p_{\text{replace}} \cdot g_{\text{replace}}(\mathrm{x}'|\mathrm{x}) \cdot \pi(x)} \approx \frac{\pi(w_m|x_{-m}) \cdot \pi(\mathrm{x}')}{\pi(w'_m|x_{-m}) \cdot \pi(\mathrm{x})} = 1$$

$$A^*_{\text{insert}}(\mathrm{x}'|\mathrm{x}) = \frac{p_{\text{delete}} \cdot g_{\text{delete}}(\mathrm{x}|\mathrm{x}') \cdot \pi(\mathrm{x}')}{p_{\text{insert}} \cdot g_{\text{insert}}(\mathrm{x}'|\mathrm{x}) \cdot \pi(\mathrm{x})} = \frac{p_{\text{delete}} \cdot \pi(\mathrm{x}')}{p_{\text{insert}} \cdot g_{\text{insert}}(\mathrm{x}'|\mathrm{x}) \cdot \pi(\mathrm{x})}$$

$$A^*_{\text{delete}}(\mathrm{x}'|\mathrm{x}) = \frac{p_{\text{insert}} \cdot g_{\text{insert}}(\mathrm{x}|\mathrm{x}') \cdot \pi(\mathrm{x}')}{p_{\text{delete}} \cdot g_{\text{delete}}(\mathrm{x}'|\mathrm{x}) \cdot \pi(\mathrm{x})} = \frac{p_{\text{insert}} \cdot g_{\text{insert}}(\mathrm{x}|\mathrm{x}') \cdot \pi(\mathrm{x}')}{p_{\text{delete}} \cdot \pi(\mathrm{x})}$$

CGMH [5] ★★★★★

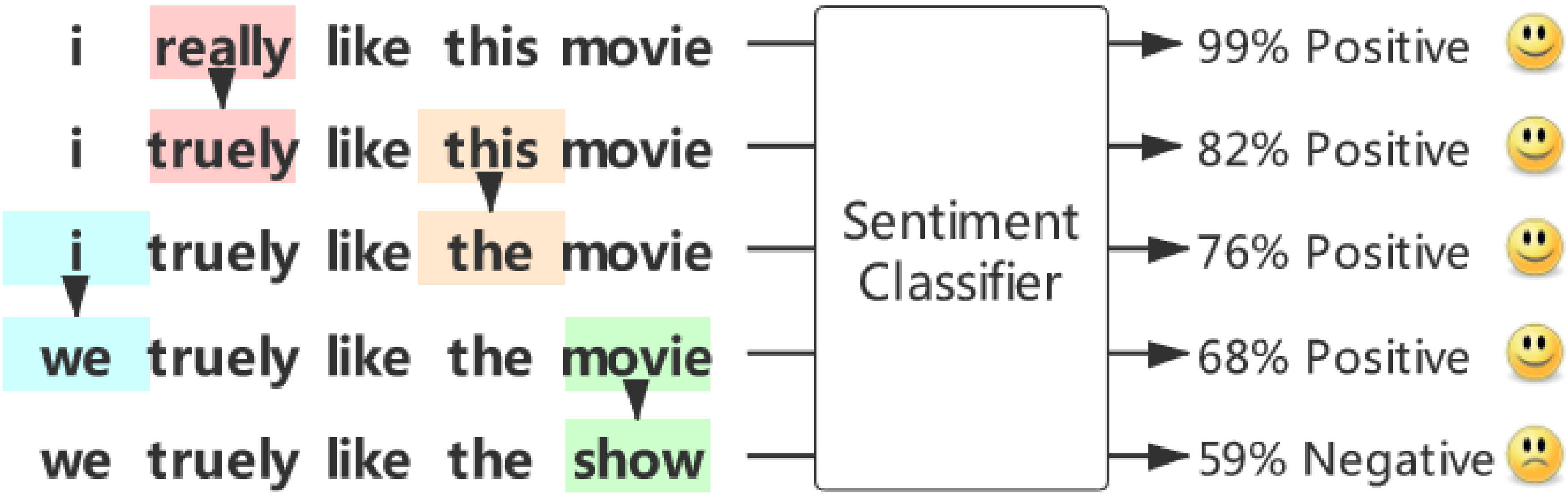
Result

Keyword(s)	Generated Sentences
friends	My good **friends** were in danger .
project	The first **project** of the scheme .
have, trip	But many people **have** never made the **trip** .
lottery, scholarships	But the **lottery** has provided **scholarships** .
decision, build, home	The **decision** is to **build** a new **home** .
attempt, copy, painting, denounced	The first **attempt** to **copy** the **painting** was **denounced** .

But, how to apply it to Adversarial Examples

MHA [6] ★★★

objective



Word change, Output change!!!

How to select a substitute word?

[6] ACL19: Generating Fluent Adversarial Examples for Natural Languages

MHA [6] ★★★

$$g_{\text{replace}}(\mathrm{x}'|\mathrm{x}) = \pi(w_m^* = w^c|\mathrm{x}_{-m}) = \frac{\pi(w_1, \cdots, w_{m-1}, w^c, w_{m+1}, \cdots, w_n)}{\sum_{w \in \mathcal{V}} \pi(w_1, \cdots, w_{m-1}, w, w_{m+1}, \cdots, w_n)}$$

Pre-selection

$$S^B(w|x) = LM(w|x_{[1:m-1]}) \cdot LM_b(w|x_{[m+1:n]})$$

w-MHA Pre-selection

$$S^W(w|x) = S^B(w|x) \cdot S(\frac{\partial \tilde{\mathcal{L}}}{\partial e_m}, e_m - e)$$

- S is the cosine similarity function
- $\tilde{L} = L(\tilde{y}|x,\ C)$ is the loss function on the target label
- e_m and e are the embeddings of the current word (w_m) and the substitute (w).

[6] ACL19: Generating Fluent Adversarial Examples for Natural Languages

MHA [6] ★★★

Result

Case 1
Premise: *three men are sitting on a beach dressed in orange with refuse carts in front of them.* **Hypothesis:** *empty trash cans are sitting on a beach.* **Prediction:** ⟨Contradiction⟩
Genetic: ***empties*** *trash cans are sitting on a beach.* **Prediction:** ⟨Entailment⟩
b*-MHA:** ***the *trash cans are sitting* ***in*** *a beach.* **Prediction:** ⟨Entailment⟩
w*-MHA:** ***the *trash cans are sitting on a beach.* **Prediction:** ⟨Entailment⟩

Case 2
Premise: *a man is holding a microphone in front of his mouth.* **Hypothesis:** *a male has a device near his mouth.* **Prediction:** ⟨Entailment⟩
Genetic: *a* ***masculine*** *has a device near his mouth.* **Prediction:** ⟨Neutral⟩
b*-MHA:** *a man has a device near his* ***car*.* **Prediction:** ⟨Neutral⟩
w*-MHA:** *a man has a device near his* ***home*.* **Prediction:** ⟨Neutral⟩

[6] ACL19: Generating Fluent Adversarial Examples for Natural Languages

Bibliography

Adversarial Examples Theory

- ICLR15: Explaining and harnessing adversarial examples
- ICLR16: Distributional smoothing with virtual adversarial training
- TPAMI18: Virtual Adversarial Training A Regularization Method for Supervised and Semi-Supervised Learning

Adversarial Examples in text

- ICLR17: Adversarial training methods for semi-supervised text classification
- IJCAI18: Interpretable Adversarial Perturbation in Input Embedding Space for Text
- ACL19: Generating Fluent Adversarial Examples for Natural Languages
- ACL19: Generating Natural Language Adversarial Examples through Probability Weighted Word Saliency
- ARXIV19: A survey on Adversarial Attacks and Defenses in Text

Something interesting in Adversarial Examples

- IJCAI19: Improving the Robustness of Deep Neural Networks via Adversarial Training with Triplet Loss
- NIPS19: Adversarial Examples Are Not Bugs, They Are Features
- NIPS19: Learning to Confuse Generating Training Time Adversarial Data with Auto-Encoder