Blank Language Models

Tianxiao Shen*  Victor Quach*  Regina Barzilay  Tommi Jaakkola  (*: equal contribution)
tianxiao@mit.edu
**Left-to-Right Language Model**

✔ Generate from scratch

✘ Start with partially specified text
  - text editing
  - template filling
  - text restoration
  - …

![Diagram showing the process of generating text from a left-to-right language model.](image-url)
Blank Language Model (BLM)

Input: They also have _____ which _____.
Output: They also have ice cream which is really good.

- Fine-grained control over generation location
- Respect preceding and following context
- Variable number of missing tokens
BLM - Overview

- Dynamic canvas where “___” controls where tokens can be placed
- At each step,
  1. select a “___”
  2. predict a word w
  3. replace that blank with “w”, “___ w”, “w ___”, or “___ w ___”
- Stop when there is no “___”
BLM - Overview

• Dynamic canvas where “___” controls where tokens can be placed
• At each step,
  1. select a “___”
  2. predict a word w
  3. replace that blank with “w”, “___ w”, “w ___”, or “___ w ___”
• Stop when there is no “___”

They also have _____ which ______.
BLM - Overview

• Dynamic canvas where “___” controls where tokens can be placed
• At each step,
  1. select a “___”
  2. predict a word w
  3. replace that blank with “w”, “__ w”, “w __”, or “___ w ___”
• Stop when there is no “___”

____ really _____

They also have _____ which _____. 
BLM - Overview

- Dynamic canvas where “___” controls where tokens can be placed
- At each step,
  1. select a “___”
  2. predict a word w
  3. replace that blank with “w”, “___ w”, “w ___”, or “___ w ___”
- Stop when there is no “___”

They also have _____ which _____ really ______.
BLM - Overview

- Dynamic canvas where “___” controls where tokens can be placed
- At each step,
  1. select a “___”
  2. predict a word w
  3. replace that blank with “w”, “___ w”, “w ___”, or “___ w ___”
- Stop when there is no “___”

ice_____

↓

They also have _____ which _____ really _____.
BLM - Overview

- Dynamic canvas where “___” controls where tokens can be placed
- At each step,
  1. select a “___”
  2. predict a word w
  3. replace that blank with “w”, “___ w”, “w ___”, or “___ w ___”
- Stop when there is no “___”

They also have ice _____ which _____ really _____. 
BLM - Overview

• Dynamic canvas where “___” controls where tokens can be placed
• At each step,
  1. select a “___”
  2. predict a word w
  3. replace that blank with “w”, “___ w”, “w ___”, or “___ w ___”
• Stop when there is no “___”

is

They also have ice _____ which _____ really _____.
BLM - Overview

- Dynamic canvas where “___” controls where tokens can be placed
- At each step,
  1. select a “___”
  2. predict a word w
  3. replace that blank with “w”, “___ w”, “w ___”, or “___ w ___”
- Stop when there is no “___”

They also have ice _____ which is really ______.
BLM - Overview

- Dynamic canvas where “___” controls where tokens can be placed
- At each step,
  1. select a “___”
  2. predict a word $w$
  3. replace that blank with “$w$”, “___ $w$”, “$w$ ___”, or “___ $w$ ___”
- Stop when there is no “___”

```
cream

They also have ice _____ which is really _____.
```
BLM - Overview

- Dynamic canvas where “___” controls where tokens can be placed
- At each step,
  1. select a “___”
  2. predict a word w
  3. replace that blank with “w”, “___ w”, “w ___”, or “___ w ___”
- Stop when there is no “___”

They also have ice cream which is really good.
BLM - Overview

- Dynamic canvas where “___” controls where tokens can be placed
- At each step,
  1. select a “___”
  2. predict a word w
  3. replace that blank with “w”, “___ w”, “w ___”, or “___ w ___”
- Stop when there is no “___”

They also have ice cream which is really good.
BLM - Overview

- Dynamic canvas where “___” controls where tokens can be placed
- At each step,
  1. select a “___”
  2. predict a word w
  3. replace that blank with “w”, “___ w”, “w ___”, or “___ w ___”
- Stop when there is no “___”

Grammar
- Nonterminal: ___
- Terminals: w ∈ V
- Production rules: ___ → ___? w ___?
  (dist. depends on model and context)
BLM - Architecture

1) Choose a blank

They also have ____ which ____.

2) Predict a word

Linear & Softmax

2) Predict a word

Linear & Softmax

really

3) Create new blanks

MLP

really

Fill and repeat

really

really

really
## BLM - Likelihood

<table>
<thead>
<tr>
<th>Step $t$</th>
<th>Canvas $c$</th>
<th>Location $b$</th>
<th>Word $w$</th>
<th>Action $a$</th>
<th>(Left blank $l$,</th>
<th>Right blank $r$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.</td>
<td><em><strong>#1</strong></em></td>
<td>#1</td>
<td>have</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>1.</td>
<td><em><strong>#1_____have</strong></em>#2___</td>
<td>#1</td>
<td>They</td>
<td>No</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>2.</td>
<td>They <em><strong>#1_____have</strong></em>#2___</td>
<td>#2</td>
<td>.</td>
<td>Yes</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>3.</td>
<td>They <em><strong>#1_____have</strong></em>#2____ .</td>
<td>#2</td>
<td>which</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>4.</td>
<td>They <em><strong>#1_____have</strong></em>#2____ which___#3____ .</td>
<td>#1</td>
<td>also</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>5.</td>
<td>They also have <em><strong>#1</strong></em>_ which___#2____ .</td>
<td>#2</td>
<td>really</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>6.</td>
<td>They also have <em><strong>#1</strong></em>_ which___#2____ really___#3____ .</td>
<td>#1</td>
<td>ice</td>
<td>No</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>7.</td>
<td>They also have ice <em><strong>#1</strong></em>_ which___#2____ really___#3____ .</td>
<td>#2</td>
<td>is</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>8.</td>
<td>They also have ice <em><strong>#1</strong></em>_ which is really___#2____ .</td>
<td>#1</td>
<td>cream</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>9.</td>
<td>They also have ice cream which is really___#1____ .</td>
<td>#1</td>
<td>good</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>10.</td>
<td>They also have ice cream which is really good .</td>
<td>-End-</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Trajectory:** generating process from an initial “___” to complete text
BLM - Likelihood

<table>
<thead>
<tr>
<th>Step $t$</th>
<th>Canvas $c$</th>
<th>Location $b$</th>
<th>Word $w$</th>
<th>Action $a$ (Left blank $l$, Right blank $r$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.</td>
<td>_ #1_</td>
<td>#1</td>
<td>have</td>
<td>#1 $3$  Yes $l$  Yes $r$</td>
</tr>
<tr>
<td>1.</td>
<td>_ #1_ have _ #2_</td>
<td>#1</td>
<td>1 They</td>
<td>#1 $1$  No $l$  Yes $r$</td>
</tr>
<tr>
<td>2.</td>
<td>They _ #1_ have _ #2_</td>
<td>#2</td>
<td>10 .</td>
<td>#2 $10$  Yes $l$  No $r$</td>
</tr>
<tr>
<td>3.</td>
<td>They _ #1_ have _ #2_ .</td>
<td>#2</td>
<td>6 which</td>
<td>#2 $6$  Yes $l$  Yes $r$</td>
</tr>
<tr>
<td>4.</td>
<td>They _ #1_ have _ #2_ which _ #3_</td>
<td>#1</td>
<td>2 also</td>
<td>#1 $2$  No $l$  No $r$</td>
</tr>
<tr>
<td>5.</td>
<td>They also have _ #1_ which _ #2_</td>
<td>#2</td>
<td>8 really</td>
<td>#2 $8$  Yes $l$  Yes $r$</td>
</tr>
<tr>
<td>6.</td>
<td>They also have _ #1_ which _ #2_ really _ #3_</td>
<td>#1</td>
<td>4 ice</td>
<td>#1 $4$  No $l$  Yes $r$</td>
</tr>
<tr>
<td>7.</td>
<td>They also have ice _ #1_ which _ #2_ really _ #3_.</td>
<td>#2</td>
<td>7 is</td>
<td>#2 $7$  No $l$  No $r$</td>
</tr>
<tr>
<td>8.</td>
<td>They also have ice _ #1_ which is really _ #2_.</td>
<td>#1</td>
<td>5 cream</td>
<td>#1 $5$  No $l$  No $r$</td>
</tr>
<tr>
<td>9.</td>
<td>They also have ice cream which is really _ #1_.</td>
<td>#1</td>
<td>9 good</td>
<td>#1 $9$  No $l$  No $r$</td>
</tr>
<tr>
<td>10.</td>
<td>They also have ice cream which is really good</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

A sentence $x$ with $n$ words can be realized by $n!$ trajectories, each trajectory corresponds to a different word insertion order

$$p(x; \theta) = \sum_{\sigma \in S_n} p(x, \sigma; \theta) = \sum_{\sigma \in S_n} \prod_{t=0}^{n-1} p(a_t^{x, \sigma} | c_t^{x, \sigma}; \theta)$$

order

action, canvas at step $t$
BLM - Training

\[
\log p(x; \theta) = \log \sum_{\sigma \in S_n} \prod_{t=0}^{n-1} p(a_t^x, \sigma | c_t^x, \sigma; \theta) \quad \text{intractable}
\]

\[
\log \left( \frac{1}{m} \sum_{i=1}^{m} a_i \right) \geq \frac{1}{m} \sum_{i=1}^{m} \log a_i
\]

\[
\geq \log(n!) + \frac{1}{n!} \sum_{\sigma \in S_n} \sum_{t=0}^{n-1} \log p(a_t^x, \sigma | c_t^x, \sigma; \theta)
\]
BLM - Training

\[
\log p(x; \theta) = \log \sum_{\sigma \in S_n} \prod_{t=0}^{n-1} p(a_t^{x,\sigma} | c_t^{x,\sigma}; \theta) \quad \text{intractable}
\]

\[
\geq \log(n!) + \frac{1}{n!} \sum_{\sigma \in S_n} \sum_{t=0}^{n-1} \log p(a_t^{x,\sigma} | c_t^{x,\sigma}; \theta)
\]

1. Uniformly sample \( \sigma \) from \( S_n \)
2. Uniformly sample \( t \) from 0 to \( n - 1 \)
3. Construct canvas \( c_t^{x,\sigma} \)
4. Compute estimated loss \(-\log(n!) - n \cdot \log p(a_t^{x,\sigma} | c_t^{x,\sigma}; \theta)\)
BLM - Training

$c_t^{x,\sigma}$ depends only on the first $t$ elements of $\sigma$

$\rightarrow$ combine into one pass loss calculations of trajectories that are the same in the first $t$ steps but different at the $t + 1$ step

$$\geq \log(n!) + \frac{1}{n!} \sum_{\sigma \in S_n} \sum_{t=0}^{n-1} \log p(a_t^{x,\sigma} | c_t^{x,\sigma}; \theta)$$
BLM - Training

$c_t^{x,\sigma}$ depends only on the first $t$ elements of $\sigma$

$\rightarrow$ combine into one pass loss calculations of trajectories that are the same

in the first $t$ steps but different at the $t+1$ step

$$\geq \log(n!) + \sum_{t=0}^{n-1} \frac{1}{n!} \sum_{\sigma \in S_n} \log p(a_t^{x,\sigma} | c_t^{x,\sigma}; \theta)$$

$$= \log(n!) + n \cdot \mathbb{E}_t \mathbb{E}_{\sigma_{1:t}} \mathbb{E}_{\sigma_{t+1}} \mathbb{E}_{\sigma_{t+2:n}} \left[ \log p(a_t^{x,\sigma} | c_t^{x,\sigma}; \theta) \right]$$

$$= \log(n!) + \mathbb{E}_t \mathbb{E}_{\sigma_{1:t}} \left[ \frac{n}{n-t} \sum_{\sigma_{t+1}} \log p(a_t^{x,\sigma} | c_t^{x,\sigma}; \theta) \right]$$
BLM - Training

1. Uniformly sample $t$ from 0 to $n - 1$
2. Uniformly sample $\sigma_{1:t}$
3. Construct canvas $c^x_t;\sigma$
4. Compute estimated loss $- \log(n!) - \frac{n}{n - t} \sum_{\sigma_{t+1}} \log p(a^x_t,\sigma | c^x_t;\sigma; \theta)$

$$= \log(n!) + \mathbb{E}_t \mathbb{E}_{\sigma_{1:t}} \left[ \frac{n}{n - t} \sum_{\sigma_{t+1}} \log p(a^x_t,\sigma | c^x_t;\sigma; \theta) \right]$$

$n/2$ action losses per pass :)
1. Uniformly sample $t$ from 0 to $n-1$
2. Uniformly sample $\sigma_{1:t}$
3. Construct canvas $c_t^{x,\sigma}$
4. Compute estimated loss $- \log(n!) - \frac{n}{n-t} \sum_{\sigma_{t+1}} \log p(a_t^{x,\sigma} | c_t^{x,\sigma}; \theta)$

$n/2$ action losses per pass :)
BLM - Training

1. Uniformly sample $t$ from 0 to $n - 1$
2. Uniformly sample $\sigma_{1:t}$
3. Construct canvas $c_t^{x, \sigma}$
4. Compute estimated loss $- \log(n!) - \frac{n}{n - t} \sum_{\sigma_{t+1}} \log p(a_t^{x, \sigma} | c_t^{x, \sigma}; \theta)$

$n/2$ action losses per pass :)
They also have ice cream which is really good.

1. Uniformly sample $t$ from 0 to $n - 1$
2. Uniformly sample $\sigma_{1:t}$
3. Construct canvas $c_t^{x,\sigma}$
4. Compute estimated loss $- \log(n!) - \frac{n}{n - t} \sum_{\sigma_{t+1}} \log p(a_t^{x,\sigma} | c_t^{x,\sigma}; \theta)$

$n/2$ action losses per pass :)
BLM - Training

1. Uniformly sample $t$ from 0 to $n - 1$
2. Uniformly sample $\sigma_{1:t}$
3. Construct canvas $c_{x,\sigma}^t$
4. Compute estimated loss

$$- \log(n!) - \frac{n}{n - t} \sum_{\sigma_{t+1}} \log p(a_{x,\sigma}^t | c_{x,\sigma}^t; \theta)$$

$n/2$ action losses per pass :)
BLM - Training

1. Uniformly sample $t$ from 0 to $n - 1$
2. Uniformly sample $\sigma_{1:t}$
3. Construct canvas $c_t^{x,\sigma}$
4. Compute estimated loss $- \log(n!) - \frac{n}{n - t} \sum_{\sigma_{t+1}} \log p(a_t^{x,\sigma} | c_t^{x,\sigma}; \theta)$

$n/2$ action losses per pass :)
BLM - Inference

✓ Simple greedy decoding or beam search to fill in the blanks in input

Note: not search for sentence $x$ with the maximum marginal likelihood $p(x; \theta)$, but for sentence $x$ and trajectory $\sigma$ that have the maximum joint likelihood $p(x, \sigma; \theta)$
## Experiments

<table>
<thead>
<tr>
<th>Text infilling</th>
<th>Input: They also have _____ which _____.</th>
<th>Output: They also have <strong>ice cream</strong> which <strong>is really good</strong>.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ancient text restoration</td>
<td>Input: τε εγγονον εισαι????????σοφιαι</td>
<td>Output: τε εγγονον εισαιου του σοφιαι</td>
</tr>
<tr>
<td>Sentiment transfer</td>
<td>Input: The employees were <strong>super nice</strong> and <strong>efficient</strong>!</td>
<td>Output: The employees were <strong>rude</strong> and <strong>unprofessional</strong>!</td>
</tr>
<tr>
<td>Language modeling</td>
<td>Output: They also have ice cream which is really good.</td>
<td></td>
</tr>
</tbody>
</table>
Text Infilling

Dataset: Yahoo Answers (100K documents, max length 200 words)
Test data: randomly mask tokens with ratio \( r \), contiguous masked tokens \( \rightarrow “” \)

Metrics:
- Accuracy: BLEU score against original document
- Fluency: perplexity evaluated by a pre-trained left-to-right LM
Text Infilling

Baselines:

- **BERT+LM**
  - use BERT representation of each blank as seed for a left-to-right LM that learns to generate tokens in the corresponding blank
  - at test time, multiple blanks are filled in one after another
Text Infilling

Baselines:

• BERT+LM

• Masked Language Model (MLM) with oracle length
  - replace blanks with the target number of $\langle$mask$\rangle$ tokens
  - fill masks autoregressively by most-confident-first heuristic
Text Infilling

Baselines:

• BERT+LM

• Masked Language Model (MLM) with oracle length

• Insertion Transformer (Stern et al., 2019)
  - not allow users to specify where to insert
  - force it to generate at valid locations, disable ⟨eos⟩ unless all blanks are filled,
    prioritize slots that haven’t been filled yet; otherwise failure rate > 88%
Text Infilling

Baselines:

• BERT+LM

• Masked Language Model (MLM) with oracle length

• Insertion Transformer (Stern et al., 2019)

• Seq2seq-full (Donahue et al., 2020)
  - output the full document from input
  - may miss existing tokens in input or generate tokens in incorrect locations

• Seq2seq-fill (Donahue et al., 2020)
  - output tokens to be placed in blanks, using ‘|’ to indicate separation
  - may not generate the correct number of ‘|’
Text Infilling

Baselines:

• BERT+LM
• Masked Language Model (MLM) with oracle length
• Insertion Transformer (Stern et al., 2019)
  • Seq2seq-full (Donahue et al., 2020)
    - output the full document from input
    - may miss existing tokens in input or generate tokens in incorrect locations
  • Seq2seq-fill (Donahue et al., 2020)
    - output tokens to be placed in blanks, using ‘|’ to indicate separation
    - may not generate the correct number of ‘|’

failure rate from 15% to 47%
Text Infilling

- BERT+LM
- MLM (oracle length)
- InsT
- BLM

Bar chart showing BLEU scores for different mask ratios (10%, 20%, 30%, 40%, 50%) for BERT+LM, MLM (oracle length), InsT, and BLM.
Text Infilling

- BERT+LM
- MLM (oracle length)
- InsT
- BLM

Mask ratio:
- 10%
- 20%
- 30%
- 40%
- 50%

Perplexity

Original data

Bar chart comparing the perplexity of different models at various mask ratios.
<table>
<thead>
<tr>
<th>Method</th>
<th>Generated Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original</td>
<td>when time flies, <em>where</em> does it go? <em>to</em> the center of the <strong>universe</strong> to be recycled <em>and</em> made into new time.</td>
</tr>
<tr>
<td>Blanked</td>
<td>when time flies, _____ does it go? _____ the center of the _____ to be recycled _____ made into new time.</td>
</tr>
<tr>
<td>BERT+LM</td>
<td>when time flies, <em>where</em> does it go? <em>to</em> the center of the <strong>earth</strong> to be recycled <em>came</em> made into new time.</td>
</tr>
<tr>
<td>MLM (oracle len)</td>
<td>when time flies, <em>where</em> does it go? <em>from</em> the center of the <strong>earth</strong> to be recycled <em>converted</em> made into new time.</td>
</tr>
<tr>
<td>InsT</td>
<td>when time flies, <em>where</em> does it go? <em>for</em> the center of the <strong>earth</strong> <em>has</em> to be recycled <em>and</em> made into new time.</td>
</tr>
<tr>
<td>BLM</td>
<td>when time flies, <em>where</em> does it go? <em>for</em> the center of the <strong>earth</strong> to be recycled <em>and</em> made into new time.</td>
</tr>
</tbody>
</table>

Mask ratio 10%
## Text Infilling

| Original | when time **flies**, where **does it go?** to the **center** of the universe to **be** recycled **and** made into **new time**. |
| Blanked | when time _____, where _____? _____ the _____ of _____ universe to _____ recycled _____ made into _____.
| BERT+LM | when time is, where **to?** i need to find the **way** of the universe to **be** recycled and **made into** a **lot**. |
| MLM (oracle len) | when time is, where **is the universe?** from the **creation** of the universe to **be** recycled and **made into** the universe. |
| InsT | when time **was created**, where **was it?** what was the **name** of the universe to **be** recycled and **made into** **space**. |
| BLM | when time **was created**, where **did it come from?** it was the **first part** of the universe to **be** recycled and **made into** **space**. |

**Mask ratio 50%**
Ancient Text Restoration

Dataset (Assael et al., 2019): ancient Greek inscriptions (18M characters/3M words)
• number of characters to recover is assumed to be known

Length-aware BLM (L-BLM)
• \([t]\) denotes blank of length \(t\)
• predict length \(l \in \{0, \cdots, t - 1\}\) of the new left blank,
  new right blank has length \(t - 1 - l\)

Baselines (Assael et al., 2019):
• Pythia: character-level seq2seq model to fill in one slot at a time
• Pythia-word: use both character and word representations
Ancient Text Restoration

![Bar Chart]

- **Human**
- **Pythia**
- **Pythia-word**
- **L-BLM**

**Mask ratio**
- 1%
- 25%
- 40%
- 50%

**Character error rate**

- **Single-slot**
- **Multi-slot**
Sentiment Transfer

Two-step approach:
1. Remove expressions of high polarity
   - train a sentiment classifier and mask words with attention weight above average
2. Complete the partial sentence with expressions of the target sentiment
   - train two instances of BLM, one for each sentiment

<table>
<thead>
<tr>
<th>Input</th>
<th>BLM</th>
</tr>
</thead>
<tbody>
<tr>
<td>everyone that i spoke with was very helpful and kind.</td>
<td>everyone that i spoke with was rude and unprofessional.</td>
</tr>
<tr>
<td>there is definitely not enough room in that part of the venue.</td>
<td>there is always enough parking in that part of the venue.</td>
</tr>
<tr>
<td>it is n’t terrible, but it is n’t very good either.</td>
<td>it is n’t fancy, but it is still very good either.</td>
</tr>
</tbody>
</table>
Sentiment Transfer

Yelp reviews

- MLM (Wu et al. 2019)
- BLM

Accuracy

- MLM: 90
- BLM: 100

BLEU

- MLM: 14
- BLM: 22
Language Modeling

To compute $p(x; \theta) = \sum_{\sigma \in S_n} p(x, \sigma; \theta)$, use Monte-Carlo estimate

$$\frac{n!}{m} \sum_{i=1}^{m} p(x, \sigma_i; \theta)$$

- estimated perplexity is likely to be higher than actual perplexity
- as $m$ increases, it converges to actual perplexity
# Language Modeling

Datasets: Penn Treebank (1M tokens), WikiText-2 (2M), WikiText-103 (103M)

<table>
<thead>
<tr>
<th>Model</th>
<th>PTB</th>
<th>WT2</th>
<th>WT103</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSTM (Grave et al., 2016)</td>
<td>82.3</td>
<td>99.3</td>
<td>48.7</td>
</tr>
<tr>
<td>TCN (Bai et al., 2018)</td>
<td>88.7</td>
<td>-</td>
<td>45.2</td>
</tr>
<tr>
<td>AWD-LSTM (Merity et al., 2017)</td>
<td>57.3</td>
<td><strong>65.8</strong></td>
<td>-</td>
</tr>
<tr>
<td>Transformer (Dai et al., 2019)</td>
<td>-</td>
<td>-</td>
<td>30.1</td>
</tr>
<tr>
<td>Adaptive (Baevski and Auli, 2018)</td>
<td>-</td>
<td>-</td>
<td>18.7</td>
</tr>
<tr>
<td>Transformer-XL (Dai et al., 2019)</td>
<td><strong>54.5</strong></td>
<td>-</td>
<td><strong>18.3</strong></td>
</tr>
<tr>
<td>Inst (our implementation)</td>
<td>77.3</td>
<td>91.4</td>
<td>39.4</td>
</tr>
<tr>
<td>BLM</td>
<td>69.2</td>
<td>81.2</td>
<td>42.5</td>
</tr>
</tbody>
</table>

Room for improvements!
Conclusion

• Blank language model for flexible text generation
• BLM generates sequences by dynamically creating and filling in blanks
• Effective on text infilling, ancient text restoration and style transfer

Future Work

• Template filling, information fusion, assisting human writing…
• Conditional BLM: edit and refine machine translation, dialogue system…
• Representation learning

https://github.com/Varal7/blank_language_model

Thank you!