**Non-Monotonic Sequential Text Generation**

**Sequentia**

Can we train sequence models that generate with learned, non-monotonic orders?

In this work, we:

1. Propose a Non-Monotonic Text Generation framework for training text generation policies with imitation learning.
2. Define a class of oracles, including a coaching oracle that yields policies with learned generation orders.
3. Evaluate on conditional and unconditional generation tasks.

**Sequential Text Generation**

Sequentially generate tokens \( Y = (y_1, \ldots, y_N) \).

**Introduction**

In this work, we propose a Non-Monotonic Text Generation framework for training text generation policies with imitation learning. We define a class of oracles, including a coaching oracle that yields policies with learned generation orders. We evaluate on conditional and unconditional generation tasks.

**As Imitation Learning**

- **Actions**: \( a \in V \cup \{ \emptyset \} \) word or end-token.
- **State**: \((x, T_t)\) are predictions so far.
- **Goal**: Learn \( \pi_\theta(a|s) \) that mimics an oracle policy \( \pi^*(a|s) \).

**Learning**

- Sample states, compute action costs with an oracle policy \( \pi^* \)
- Minimize action costs (KL-divergence) at sampled states

\[
\mathbb{E}_{t \sim U[2|Y|+1]} \mathbb{E}_{s_t \sim d_{s_t}} D_{KL}(\pi^*(\cdot|s_t)||\pi_\theta(\cdot|s_t))
\]

**Special cases**: \( \pi^* \) left-right, \( \pi^* \) uniform, \( \pi^* \) coaching, \( \cdots \)

The coaching oracle weights valid actions (and hence generation orders) by the learned policy:

The resulting loss reinforces preferred generation orders:

\[
KL(\pi_{\text{coaching}}||\pi_{\emptyset})
\]

**Unconditional Generation**

Generate a word at an arbitrary position, then recursively generate words to its left and then words to its right, yielding a binary tree.

<table>
<thead>
<tr>
<th>Oracle</th>
<th>%Unique</th>
<th>BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>left-right</td>
<td>17.8</td>
<td>47.0</td>
</tr>
<tr>
<td>uniform</td>
<td>98.3</td>
<td>40.0</td>
</tr>
<tr>
<td>annealed</td>
<td>93.1</td>
<td>56.2</td>
</tr>
</tbody>
</table>

**Conditional Generation**

<table>
<thead>
<tr>
<th>Oracle</th>
<th>Validation BLEU</th>
<th>Test BLEU</th>
<th>Validation F1</th>
<th>Test F1</th>
<th>Validation EM</th>
<th>Test EM</th>
<th>YSI</th>
<th>Ribs</th>
</tr>
</thead>
<tbody>
<tr>
<td>left-right</td>
<td>28.00 (0.00)</td>
<td>28.00 (0.00)</td>
<td>30.10</td>
<td>65.22</td>
<td>82.29</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>uniform</td>
<td>21.40 (0.86)</td>
<td>21.40 (0.86)</td>
<td>26.40</td>
<td>62.41</td>
<td>80.00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>annealed</td>
<td>23.50 (0.33)</td>
<td>23.50 (0.33)</td>
<td>27.96</td>
<td>63.15</td>
<td>80.91</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tree-encoding</td>
<td>28.50 (0.08)</td>
<td>28.50 (0.08)</td>
<td>30.15</td>
<td>65.22</td>
<td>82.29</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>end-tuning</td>
<td>24.30 (0.12)</td>
<td>24.30 (0.12)</td>
<td>28.59</td>
<td>63.57</td>
<td>81.61</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Variable-length Text Filling**

**Machine Translation** (IWSLT De-En, Transformer Policy)