SHAPE: Shifted Absolute Position Embedding for Transformers

Shun Kiyono\textsuperscript{1,2}, Sosuke Kobayashi\textsuperscript{2,3}, Jun Suzuki\textsuperscript{2,1}, Kentaro Inui\textsuperscript{2,1}

\textsuperscript{1}RIKEN \quad \textsuperscript{2}Tohoku University \quad \textsuperscript{3}Preferred Networks, Inc.
Absolute Position Embedding (APE)

Code

```python
pos_embed(pos_idx)
```

John yelled at Kevin

Position

0 1 2 3 4
SHAPE is “Shifted” APE
Adding a single line of code is all you need

Code

```python
pos_idx += self.training * torch.randint(0, K)
pos_embed(pos_idx)
```

John yelled at Kevin

Shifted by random offset k
Absolute Position Embedding (APE)

- Represent each position with unique embedding
  - e.g., sinusoidal wave [Vaswani+2017]
- 😊 Simple, fast, and easy to implement
- 😢 Poor performance on unseen lengths
  - i.e., APE is bad at extrapolation

John yelled at Kevin

Figure from [Neishi+2019]
Relative Position Embedding (RPE)

- Consider distance between token pair in self-attention
- 😄 Robust to unseen length by *shift invariance*
- 😢 Computationally more expensive
- 😢 Incompatible with lightweight self-attention variants
  - Performer, Linformer, etc...

**Additional Features for Attention**

John yelled at Kevin!

- *Key, Value* 
  - $a_{0-1}^{\text{Key,Value}}$
  - $a_{3-1}^{\text{Key,Value}}$

- Position
  - John: 0
  - yelled: 1
  - at: 2
  - Kevin: 3
Research Question: APE with Shift Invariance?

- **Shift invariance** in RPE seems the key

→ Spatial shift does not change function’s output

e.g. RPE is shift-invariant: \( a_{0-1}^\{\text{Key,Value}\} \) is same as \( a_{34-35}^\{\text{Key,Value}\} \)

Can we achieve shift invariance while using APE?
APE+Random Shift for Shift Invariance

- **Shifted Absolute Position Embedding (SHAPE)**
  - APE is randomly shifted by offset $k \sim \mathcal{U}(0, K)$
  - Model cannot use absolute position to learn task
    - Instead learns to use relative position?

John yelled at Kevin

EMNLP 2021
SHAPE Learns Shift Invariance

- Compare cosine similarities of hidden states
- APE: each $k$ produces different hidden states
- SHAPE: hidden states are invariant to $k$
SHAPE Learns Shift Invariance

• Compare cosine similarities of hidden states
• APE: each $k$ produces different hidden states
• SHAPE: hidden states are invariant to $k$
SHAPE Learns Shift Invariance

• Compare cosine similarities of hidden states
• APE: each $k$ produces different hidden states
• SHAPE: hidden states are invariant to $k$
Experimental Configuration

• Model: Transformer with APE, RPE, or SHAPE
• Task: Machine translation (MT)
• Training data
  1. Vanilla
     • WMT 2016 EnDe [Ott+2018]
  2. Extrapolate
     • Remove sequences longer than 50 subwords from Vanilla
  3. Interpolate
     • Concatenate neighboring sequences
• Validation data: newstest2010-2013
• Test data: newstest2014-2016
• Evaluation: sacreBLEU
Machine Translation Experiment: Three Distinct Datasets

Why three? - To evaluate model performance on seen/unseen lengths

1. **Vanilla: WMT EnDe 2016 Dataset [Ott+2018]**
   - Standard setting for MT
   - Sanity check of baseline performance

2. **Extrapolate: remove sequences longer than 50 subwords from Vanilla**
   - Evaluate if model can extrapolate
   - i.e. is model robust to unseen lengths?

3. **Interpolate: concatenate neighboring sequences (omitted)**
Result: RPE and SHAPE are Comparable

• On Extrapolate
  • Both RPE and SHAPE outperform APE
  • SHAPE is comparable to RPE
  • SHAPE is as fast as APE while RPE is not

• On Vanilla
  • All models achieve comparable performance
  • No risk of performance drop

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Model</th>
<th>Valid</th>
<th>Test</th>
<th>Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>VANILLA</td>
<td>APE†</td>
<td>23.61</td>
<td>30.46</td>
<td>x1.00</td>
</tr>
<tr>
<td></td>
<td>RPE†</td>
<td>23.67</td>
<td>30.54</td>
<td>x0.91</td>
</tr>
<tr>
<td></td>
<td>SHAPE†</td>
<td>23.63</td>
<td>30.49</td>
<td>x1.01</td>
</tr>
<tr>
<td>EXTRAPOLATE</td>
<td>APE</td>
<td>22.18</td>
<td>29.22</td>
<td>x1.00</td>
</tr>
<tr>
<td></td>
<td>RPE</td>
<td>22.97</td>
<td>29.86</td>
<td>x0.91</td>
</tr>
<tr>
<td></td>
<td>SHAPE</td>
<td>22.96</td>
<td>29.80</td>
<td>x0.99</td>
</tr>
</tbody>
</table>

Additional figures are available in Appendix.
Length Analysis: Better Extrapolation

Relative BLEU improvement from baseline (APE)

- RPE and SHAPE can better extrapolate than APE
- SHAPE and RPE have comparable extrapolation ability
Conclusion

• SHAPE : shifted absolute position embedding
  • APE with shift invariance
  • As fast as APE & comparable performance to RPE
  • Easy implementation
  • No risk of performance drop from APE

Take Home PyTorch Code

```python
pos_idx += self.training * torch.randint(0, K)
pos_embed(pos_idx)
```