Fast Inference from Transformers via Speculative Decoding

Yaniv Leviathan, Matan Kalman, Yossi Matias
ICML 2023
Presenter: Lingzhi Zhao

Some slides are from the authors’
Motivation

- Decoding K tokens takes K **serial runs**

- Can we somehow decode several tokens in parallel?
Previous Approaches

• Reduce the inference cost for all inputs equally
  • Distillation (Hinton, 2015), sparsification (Jaszczur, 2021), quantization (Hubara, 2016)

• Adaptive computation
  • Han, 2021, Sukhbaatar, 2019
  • Different inference steps require different size of model
Observation 1

• Some tokens are easier than others

• Hebrew: הָיוּ בָּרָק אָוָּבָהַּ הנשיא  

  English: The president was Barack Obama.

Easy - e.g. can guess based on just the last token.

Hard - e.g. requires looking several tokens back, knowledge of Hebrew
Observation 2

- Decoding from large transformers is memory bound

<table>
<thead>
<tr>
<th>Hardware can do</th>
<th>Transformers need</th>
</tr>
</thead>
<tbody>
<tr>
<td>Floating point operations per byte read</td>
<td>Floating point operations per byte read</td>
</tr>
</tbody>
</table>
Speculative Decoding

- Sample generations from more efficient *approximation* models as speculative prefixes for the slower *target* models
- Consider two models $M_q$, target model and $M_p$, more efficient approximation model

\[
p_1(x) = M_p(pf) \quad \rightarrow \quad x_1
\]
\[
p_2(x) = M_p(pf, x_1) \quad \rightarrow \quad x_2
\]
\[
\vdots
\]
\[
p_5(x) = M_p(pf, x_1, x_2, x_3, x_4) \quad \rightarrow \quad x_5
\]

Run approximation model $\gamma$ steps
Speculative Decoding

- Consider two models $M_q$, target model and $M_p$, more efficient approximation model

\[
q_1(x), q_2(x), q_3(x), q_4(x), q_5(x), q_6(x) = M_q(pf, x_1, x_2, x_3, x_4, x_5)
\]

Run target model once
Speculative Decoding

• Case 1: if $q(x) \geq p(x)$, then accept the generated token from the approximation model

• Case 2: if $q(x) < p(x)$, then accept with probability $\frac{q(x)}{p(x)}$
  • If rejected, sample $x$ from an adjusted distribution $(q(x) - p(x))^+$
Theoretical Analysis: Number of Parallel Tokens

• The expected number of tokens generated by speculative decoding is

\[ E(\# \text{ generated tokens}) = \frac{1 - \alpha^{\gamma+1}}{1-\alpha} \]

• \( \alpha \): expected acceptance rate

• Optimally choose the number of tokens \( \gamma \) to attempt to parallelize
Theoretical Analysis: Walltime Improvement

• The expected improvement factor in total walltime:

\[ \frac{1 - \alpha^{\gamma+1}}{(1-\alpha)(\gamma c+1)} \]

• \( c \): the ratio between the time for a single run of the approximation model and the time for a single run of the target model
How to choose $\gamma$

- The optimal $\gamma$ should maximize the walltime reduction

*Figure 3. The optimal $\gamma$ as a function of $\alpha$ for various values of $c$.***
Evaluation

- Implement SD in T5X codebase; two tasks: translation and text summarization;
- Target model (11B); approximation models (800M, 250M, 77M)
- Batch size 1 on a single TPU-v4

Table 2. Empirical results for speeding up inference from a T5-XXL 11B model.

<table>
<thead>
<tr>
<th>TASK</th>
<th>$M_q$</th>
<th>TEMP</th>
<th>$\gamma$</th>
<th>$\alpha$</th>
<th>SPEED</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENDE</td>
<td>T5-SMALL ★</td>
<td>0</td>
<td>7</td>
<td>0.75</td>
<td>3.4X</td>
</tr>
<tr>
<td>ENDE</td>
<td>T5-BASE</td>
<td>0</td>
<td>7</td>
<td>0.8</td>
<td>2.8X</td>
</tr>
<tr>
<td>ENDE</td>
<td>T5-LARGE</td>
<td>0</td>
<td>7</td>
<td>0.82</td>
<td>1.7X</td>
</tr>
<tr>
<td>ENDE</td>
<td>T5-SMALL ★</td>
<td>1</td>
<td>7</td>
<td>0.62</td>
<td>2.6X</td>
</tr>
<tr>
<td>ENDE</td>
<td>T5-BASE</td>
<td>1</td>
<td>5</td>
<td>0.68</td>
<td>2.4X</td>
</tr>
<tr>
<td>ENDE</td>
<td>T5-LARGE</td>
<td>1</td>
<td>3</td>
<td>0.71</td>
<td>1.4X</td>
</tr>
<tr>
<td>CNNNDM</td>
<td>T5-SMALL ★</td>
<td>0</td>
<td>5</td>
<td>0.65</td>
<td>3.1X</td>
</tr>
<tr>
<td>CNNNDM</td>
<td>T5-BASE</td>
<td>0</td>
<td>5</td>
<td>0.73</td>
<td>3.0X</td>
</tr>
<tr>
<td>CNNNDM</td>
<td>T5-LARGE</td>
<td>0</td>
<td>3</td>
<td>0.74</td>
<td>2.2X</td>
</tr>
<tr>
<td>CNNNDM</td>
<td>T5-SMALL ★</td>
<td>1</td>
<td>5</td>
<td>0.53</td>
<td>2.3X</td>
</tr>
<tr>
<td>CNNNDM</td>
<td>T5-BASE</td>
<td>1</td>
<td>3</td>
<td>0.55</td>
<td>2.2X</td>
</tr>
<tr>
<td>CNNNDM</td>
<td>T5-LARGE</td>
<td>1</td>
<td>3</td>
<td>0.56</td>
<td>1.7X</td>
</tr>
</tbody>
</table>

- T5-small (77M) has a good balance between acceptance rate and number of generated tokens, and achieves fast inference time
Evaluation

- Approximation tends to produce $\alpha$ between 0.5 and 0.9
- Even trivial unigram and bigram approximations yield non negligible $\alpha$ values with negligible runtime
What SD is good at

• Decode faster from autoregressive models: 2x-3x in typical scenarios

• Only different decoding algorithm: no architecture changes, no re-training

• Identical output distribution
What SD is limited at

- Adaptively choosing $\gamma$ during runtime could further improve its performance

- Fine-tune the approximation model to generate more similar distributions with the target model

- Lack comparisons with state-of-the-arts