**Three tasks:**

1. machine translation

2. abstractive text summarization

3. language modeling

**Baseline:**

Used feature map:

ELU, RFA, T2R

Omitted models:

Linformer (fixed length) and synthesizer (need specify the maximum input length) are excluded.

**Task 1: Machine translation**

**Setup DataSet:**

1. WMT16 En-De: (4.5M train pair, average target length 29.5 tokens)
2. WMT14 En-Fr (36M, 31.7)
3. WMT17 Zh-En (20M, 28.5)

**Model structure:**

Use 6 large-sized transformer with 6 layers, 16 attention heads, 1024 dimensions, and 4096 hidden dimensions for both encoder and decoder.

**Parameters:**

1. Dropof = 0.3
2. Weight decay=0.01
3. Label smoothing =0.1
4. Increased batch size of approximately 460K tokens by accumulating gradients without updating parameters.
5. Random initialization for 30K (60K for the large En-Fr dataset) steps using Adam with a learning rate of and
6. Beam search decoding with beam size 5 and length penalty 1.0
7. The checkpoints from the last five epochs are averaged to obtain the final model.
8. Tokenized BLEU evaluation.
9. Applied to both cross and causal attention & Memory sizes k = (32, 4) for cross and causal attention.

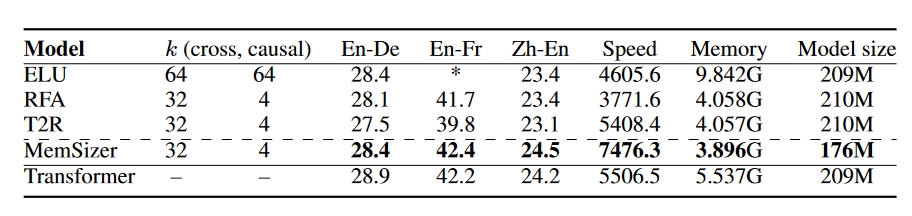
**Result**

1. kernel-based transformers suffer from additional overhead when the generated sequence is relatively short.
2. ELU has a much larger feature size k, leading to increased memory overhead.
3. MemSizer outperforms RFA and T2R while being comparable to ELU, in terms of test BLEU score.

**Conclusion:**

MemSizer achieves faster generation time and more efficient GPU memory utilization.

**Image:**

****

**Task 2: Abstractive Text Summarization**

**Setup DataSet:**

1. CNN/DailyMail, training, validation, and testing (287,113/13,368/11,490 documents). average length: 766/53
2. XSUM, training, validation, and testing 227K (204,045/11,332/11,334), average length: 431/23

**Model structure:**

Use the BART-large configuration with 12 layers, 16 attention heads, 1024 model dimensions, and 4096 hidden dimensions for both the encoder and decoder.

**Parameters:**

1. Dropof = 0.1

2. Weight decay=0.01

1. Label smoothing =0.1
2. Each model is trained from random initialization for 50K steps using Adam.
3. Random initialization for 30K (60K for the large En-Fr dataset) steps using Adam with a learning rate of and
4. Beam search decoding with beam size 5 and length penalty 1.0
5. Use the standard ROUGE metrics (F1 scores ROUGE-1/2/L) Tokenized BLEU evaluation.
6. Memory sizes k = (32, 4) for cross and causal attention.

**Result**

1. MemSizer outperforms RFA and T2R on both datasets in terms of ROUGE scores. (Note: ELU is omitted, as it is diverged)
2. On the XSUM dataset, MemSizer even achieves better results than the vanilla transformer while being much faster and memory efficient. On CNN/DailyMail dataset, there are still gaps behind vanilla transformer.

Reason: Maybe come from suffering from the limited capacity of the reduced memory bank

1. Similar to machine translation, the kernel-based transformers suffer from additional overhead when the generated sequence is relatively short

**Conclusion:**

Overall, MemSizer achieves the largest speed-up (22% speed-up compared to the vanilla transformer) and the smallest memory consumption (75% reduction compared to the vanilla transformer)

**Image:**

**表格

描述已自动生成**

**Task 3: Language Modeling**

**Setup DataSet:**

1. WikiText-103 language model (LM) benchmark, 103M tokens sampled from English Wikipedia.

**Model structure:**

32 layers, 8 heads, 128 head dimensions, 1024 model dimensions, 4096 fully connected dimensions and dropofs

**Parameters:**

1. Dropof = 0.2
2. Set the memory size k to be 32.
3. The word embedding and softmax matrices are tied.
4. Learning Rate:
5. Batch size is 256.

**Training/Testing/Validation**1. Partition the training data into non-overlapping blocks of 512 contiguous tokens and train the model to autoregressively predict each token.

1. Validation and test perplexities are measured by predicting the last 256 words out of the input of 512 consecutive words to avoid evaluating tokens in the beginning with limited context.

**Result**

1. MemSizer outperforms ELU and RFA and achieves comparable performance to T2R.  
   (Shows same performance can be achieved by without approximating the softmax attention in the vanilla transformer. Also, The generation time, memory usage, and model size are significantly reduced in MemSizer.)
2. There remains a gap of 2.3 perplexity points between the MemSizer and transformer models, which might be reduced by leveraging a swap-then-finetune approach.

**Conclusion:**

MemSizer is more advantageous with cross-attention in encoder-decoder architectures.

**Image:**

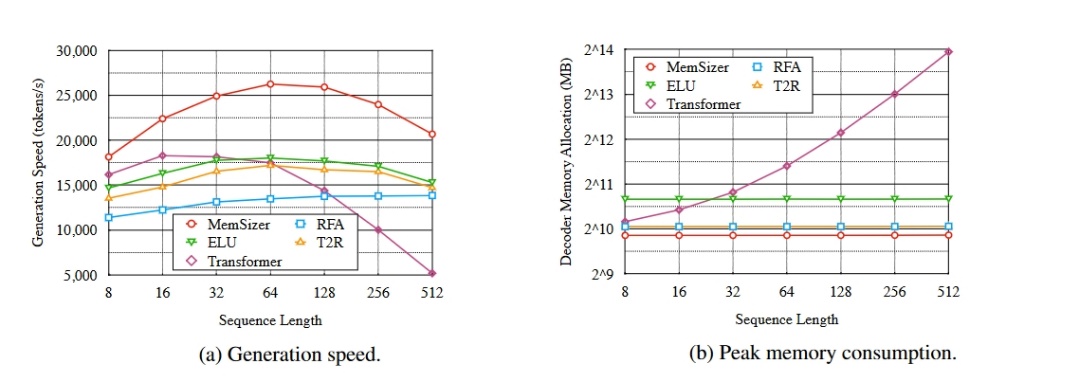
表格

描述已自动生成

**Memsizer Analysis**

1. Computational Overhead vs Sequence Length (Evaluate the time and memory efficiency against length)

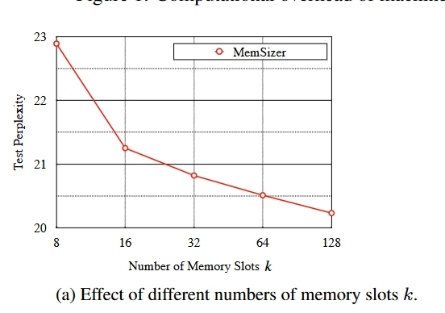
Assumption: Source Length is equal to the target length, batch size: 256, A100 GPU



Result (From image):

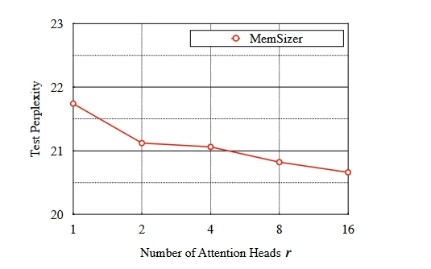
1. Dramatically outpacing the vanilla transformer model in longer sequence generation (300% speedup)
2. Outperforms other linear recurrent variants by large margins (35% faster than ELU for 512 length)
3. Maximum speedup achieved at length=64.
4. Peak memory consumption is almost constant.
5. May better when the batch size larger.
6. Number of Memory Slots: Effect of the number of memory slots

Assumption: Test perplexities using different values of k on the WikiText-103 language model task.

Result:

1. Performance gets better as k goes larger.
2. Do not see number of k has a considerable le impact on inference time and memory cost. ( In training time, however, processing time per token is roughly linear to k, may because more intermediate states need be stored for back propagation.)
3. Number of Attention Heads: Impact of number of attention heads

Assumption: Varying values of r on the WikiText-103 language model task.

Result:

1. Number of attention heads slightly affects the test perplexity, with slightly better performance.