Reversible Recurrent Neural Networks

Matthew MacKay, Paul Vicol, Jimmy Ba, Roger Grosse

University of Toronto & Vector Institute

Motivation

- RNNs are memory intensive to train
  - This limits the flexibility of RNN models that can be trained and the lengths of sequences we can backpropagate through
- Reversible RNNs are RNNs for which the hidden-to-hidden transition can be reversed
  - Reduce memory usage during training, as hidden states need not be stored.

Summary

- We show perfectly reversible RNNs are fundamentally limited since they cannot forget information from their hidden state.
- We provide a scheme for storing a small number of bits in order to allow perfect reversal with forgetting.
- We introduce the RevGRU and RevLSTM models, which are reversible analogues of standard the GRU and LSTM
- The reversible models achieve similar performance to the standard models on language modeling and neural machine translation, while saving 5–15$\times$ activation memory cost

Reversible Recurrent Architectures

- Separate the hidden state h of a RevGRU into two groups, h_1 and h_2, with updates:

$$z_1, g_1 = F(h_2, x) \qquad h_1 \leftarrow z_1 \odot h_1 + (1 - z_1) \odot g_1 \qquad (1)$$

$$z_2, g_2 = G(h_1, x) \qquad h_2 \leftarrow z_2 \odot h_2 + (1 - z_2) \odot g_2 \qquad (2)$$

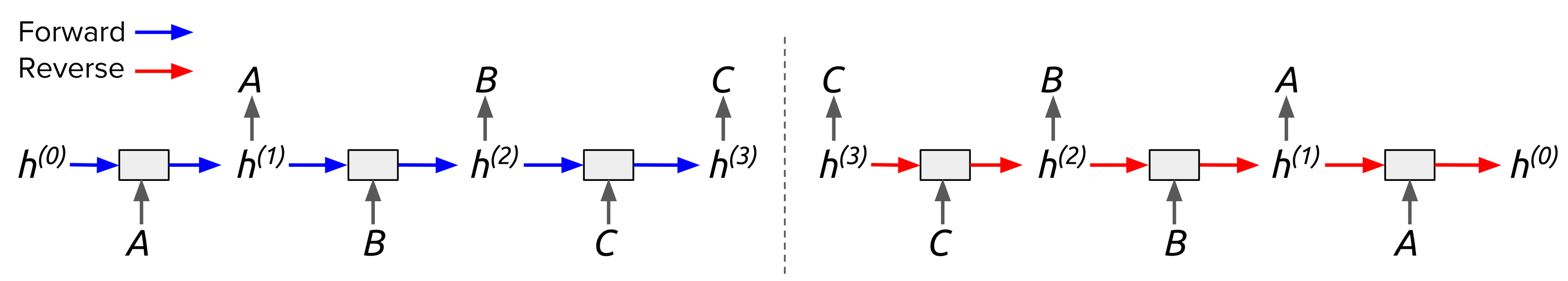
where F and G are analogous to standard GRU updates and x is the current input.

- Reversible in exact arithmetic, e.g. reconstruct h_2 by recomputing z_2, g_2 and using:

$$h_2 \leftarrow [h_2 - (1 - z_2) \odot g_2] \odot 1/z_2$$

- In practice, cannot reconstruct perfectly since forgetting (multiplication by z) discards information

Impossibility of No Forgetting



- Can achieve perfect reconstruction with no memory usage by removing the forgetting step, but this limits model capability
- Consider the repeat task: repeat each input token on next timestep
- Unrolling the reverse computation reveals a sequence-to-sequence computation in which the decoder must reproduce the input sequence from the final encoder hidden state
  - This becomes infeasible for long sequences

Reversibility with Forgetting

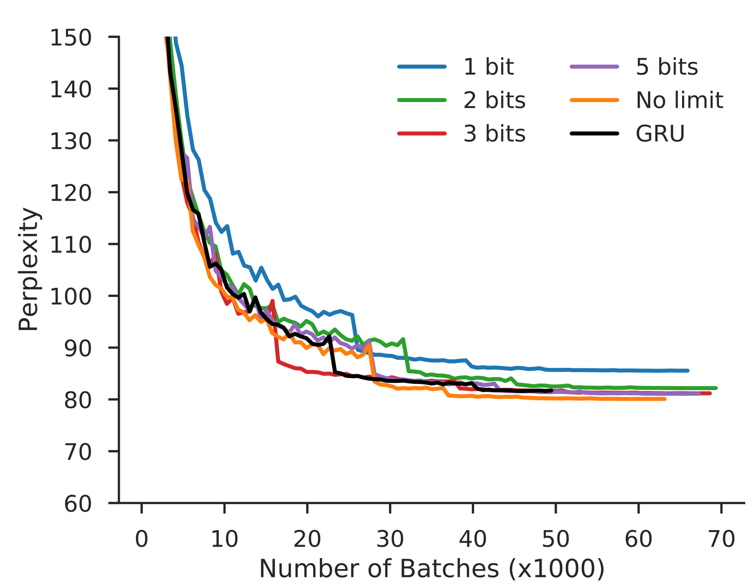
- We allow forgetting and use a buffer to efficiently store forgotten information
  - Neglecting buffer overflow, $z = 2^{-k}$ corresponds to storing exactly k bits
  - We limit the amount forgotten by restricting z to lie in an interval $(a, 1)$ for $a > 0$

Language Modelling

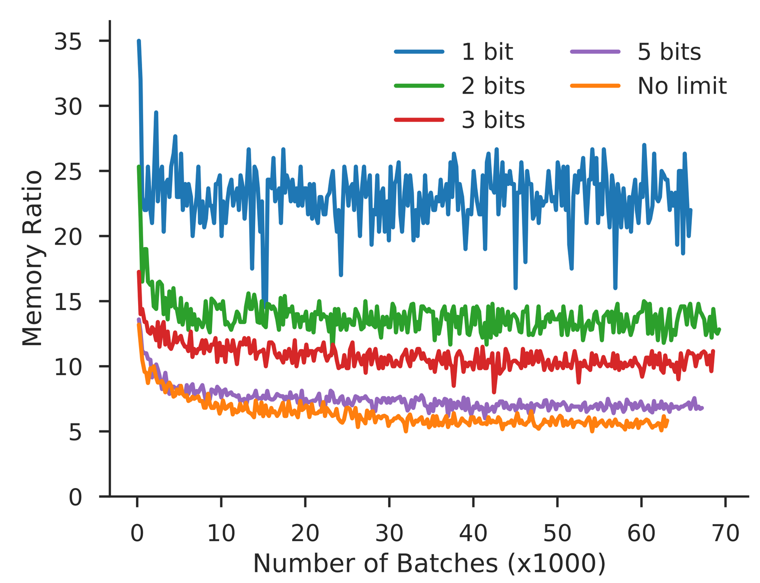
- Validation perplexities and memory savings on Penn TreeBank word-level language modeling.

Reversible Model	2 bit	3 bits	5 bits	No limit	Usual Model	No limit
1 layer RevGRU	82.2 (13.8)	81.1 (10.8)	81.1 (7.4)	81.5 (6.4)	1 layer GRU	82.2
2 layer RevGRU	83.8 (14.8)	83.8 (12.0)	82.2 (9.4)	82.3 (4.9)	2 layer GRU	81.5
1 layer RevLSTM	79.8 (13.8)	79.4 (10.1)	78.4 (7.4)	78.2 (4.9)	1 layer LSTM	78.0
2 layer RevLSTM	74.7 (14.0)	72.8 (10.0)	72.9 (7.3)	72.9 (4.9)	2 layer LSTM	73.0

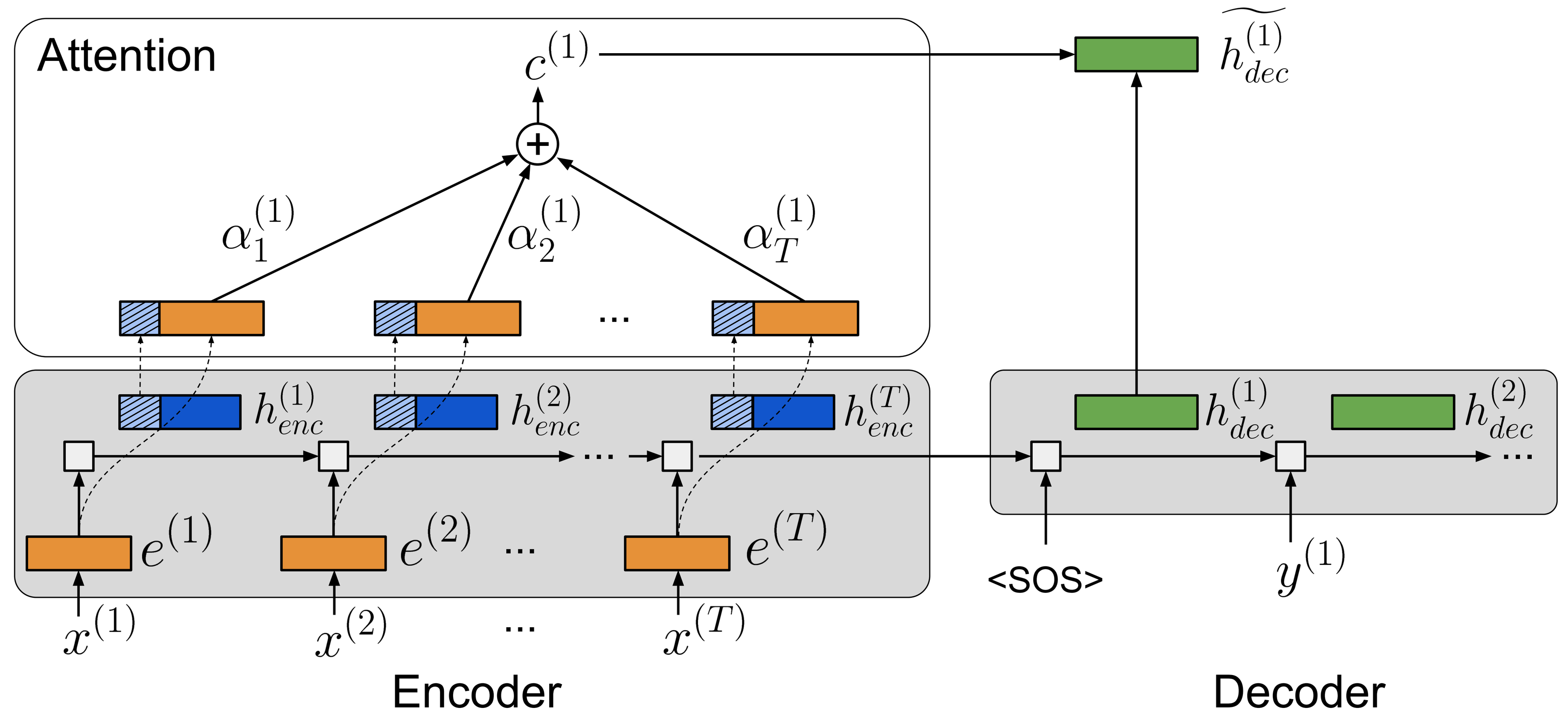
- RevGRU, validation perplexities



- RevGRU, memory savings



Memory Savings with Attention



- Standard models use attention over encoder hidden states
  - **Problematic:** Must retain the hidden states in memory to use them for attention.
- We perform attention over the concatenation of word embeddings and slices of the encoder hidden states
  - Embeddings are computed directly from the input tokens; they don't need to be stored.
  - Only the *hidden state slices* that are attended to must be stored.

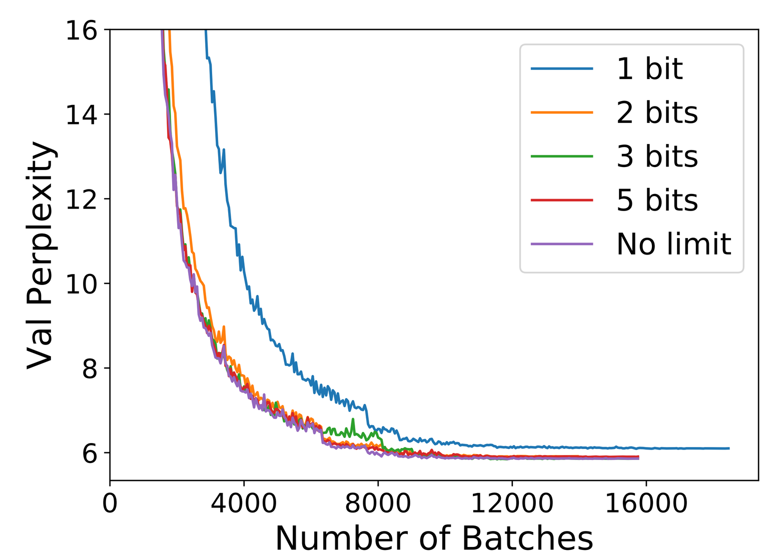
Neural Machine Translation Experiments

- Performance on the Multi30K dataset for several variants of attention and restrictions on forgetting.

Model	Attention	1 bit		2 bit		3 bit		5 bit		No Limit	
		P	M	P	M	P	M	P	M	P	M
RevLSTM	300H	26.44	1.0	36.10	1.0	37.05	1.0	37.30	1.0	36.80	1.0
	Emb	31.92	20.0	31.98	15.1	31.60	13.9	31.42	10.7	31.45	10.1
	Emb+20H	36.80	12.1	36.78	9.9	37.23	8.9	36.45	8.1	37.30	7.4
RevGRU	300H	34.86	1.0	33.49	1.0	33.01	1.0	33.03	1.0	33.08	1.0
	Emb	28.51	13.2	28.76	13.2	28.86	12.9	27.93	12.8	28.59	12.9
	Emb+20H	34.00	7.2	34.41	7.1	34.39	6.4	34.04	5.9	34.94	5.7

- **P** denotes the test BLEU scores; **M** denotes the average memory savings of the encoder during training. **20H** denotes a 20-dimensional slice of the hidden state.

- RevLSTM, Emb+20H, validation



- RevLSTM, Emb+20H, memory