Universal Transformers

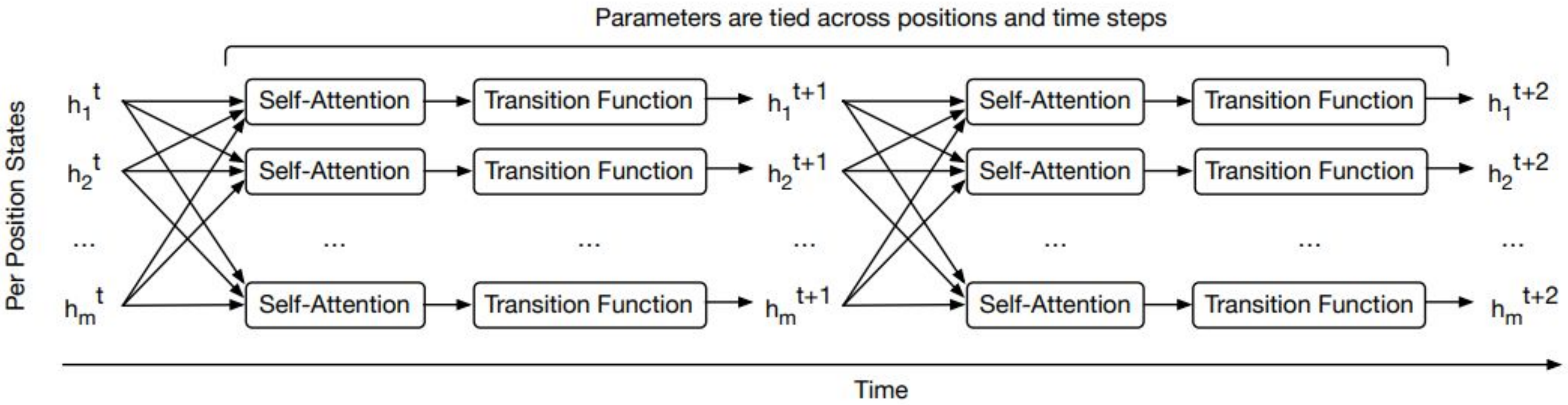
presenter: Baosong Yang

Motivation

- Failing to generalize in many tasks
- Not computationally universal, limiting its theoretical expressivity

- Core difference to Transformer:
  - Repeatedly revises its representations of all symbols in the sequence with each recurrent step. (Using RNN to replace the stacked architecture)
  - Adopting ACT (adaptive computation time)

Model

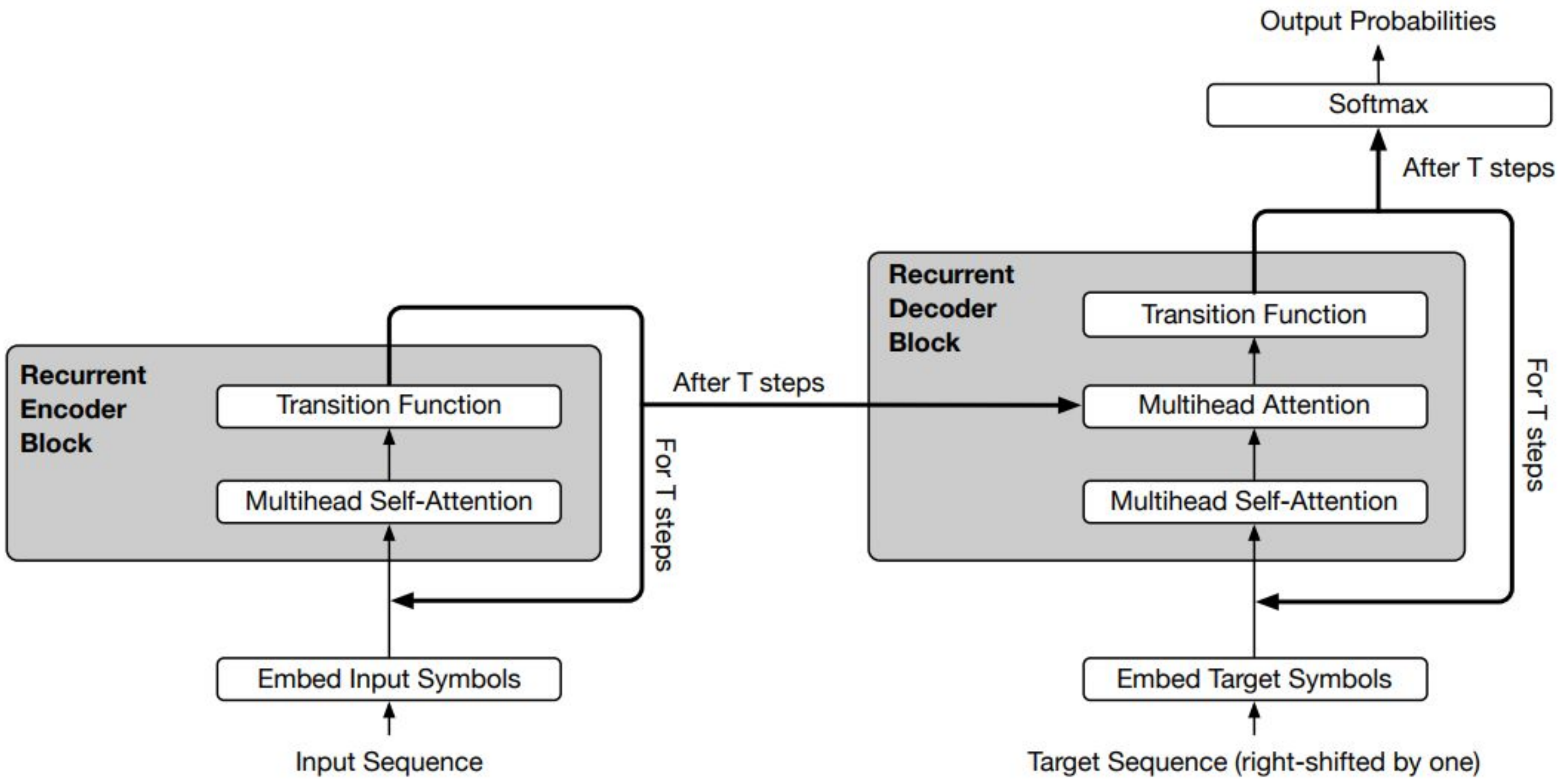


- Recurrent transition function
  - either a separable convolution
  - or a fully-connected neural network that consists of a single rectified-linear activation function between two affine transformations
- Parameters are tied across time steps (layers)

Model

$$H^t = \text{LayerNorm}(A^{t-1} + \text{Transition}(A^t))$$
$$\text{where } A^t = \text{LayerNorm}(H^{t-1} + \text{MultiHeadSelfAttention}(H^{t-1} + P^t)),$$

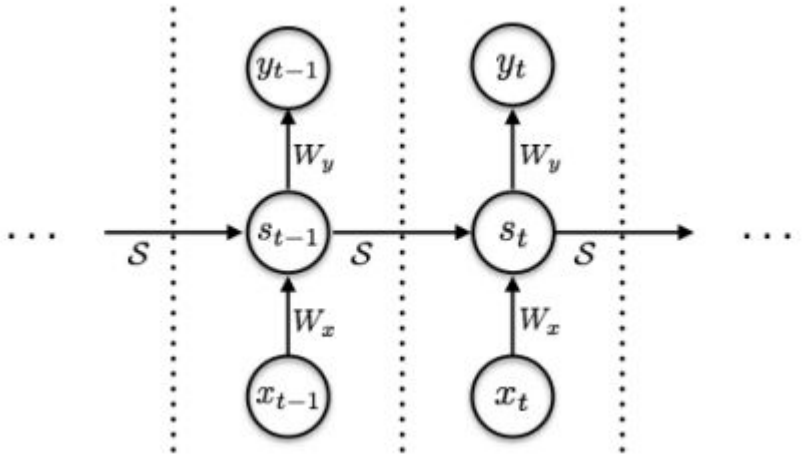
$$P^t_{pos,2j} = \sin(pos/10000^{2j/d}) \oplus \sin(t/10000^{2j/d})$$
$$P^t_{pos,2j+1} = \cos(pos/10000^{2j/d}) \oplus \cos(t/10000^{2j/d}).$$

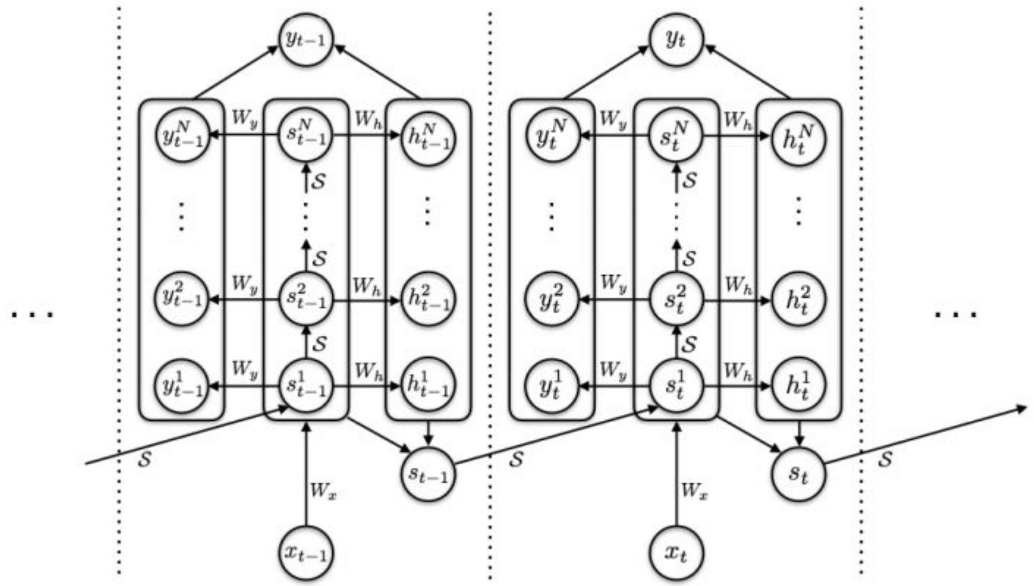


Adaptive Computation Time

- Dynamically modulating the number of computational steps needed to process each input symbol.
- Based on a scalar pondering value predicted by the model at each step.
- Once the per-symbol recurrent block halts, its state is simply copied to the next step until all blocks halt, or we reach a maximum number of steps.

Adaptive Computation Time





- input: the hidden state of the t-th element in n-th layer
- output: probility of halting

$$h_t^n = \sigma\left(W_h s_t^n + b_h\right) \qquad p_t^n = \begin{cases} R(t) \text{ if } n = N(t) \\ h_t^n \text{ otherwise} \end{cases} \qquad N(t) = \min\{n' : \sum_{n=1}^{n'} h_t^n >= 1-\epsilon\}, \quad R(t) = 1 - \sum_{n=1}^{N(t)-1} h_t^n$$

- Differentiable: $s_t = \sum_{n=1}^{N(t)} p_t^n s_t^n \qquad y_t = \sum_{n=1}^{N(t)} p_t^n y_t^n$

Experiments

- bABI QA: The dataset consists of 20 different tasks. Given a story, answer questions.

Model	10K examples		1K examples	
	train single	train joint	train single	train joint
Previous best results:				
QRNet [24]	0.3 (0/20)	-	-	-
Sparse DNC [23]	-	2.9 (1/20)	-	-
GA+MAGE [8]	-	-	8.7 (5/20)	-
MemN2N [26]	-	-	-	12.4 (11/20)
Our Results:				
Transformer [31]	15.2 (10/20)	22.1 (12/20)	21.8 (5/20)	26.8 (14/20)
Universal Transformer (this work)	0.23 (0/20)	0.47 (0/20)	5.31 (5/20)	8.50 (8/20)
Adapt. Univ. Transformer (this work)	**0.21 (0/20)**	**0.29 (0/20)**	**4.56 (3/20)**	**7.85 (5/20)**

Table 1: Average error and number of failed tasks ($>5\%$ error) out of 20 (in parentheses; lower is better in both cases) on the bAbI dataset under the different training/evaluation setups. We indicate state-of-the-art where available for each, or '-' otherwise.

Experiments

- Subject-Verb Agreement: Number agreement between subject and verb

Model	Number of attractors						
	0	1	2	3	4	5	Total
Previous best results [33]:							
Best Stack-RNN	0.994	0.979	0.965	0.935	0.916	0.880	0.9923
Best LSTM	0.993	0.972	0.95	0.922	0.900	0.842	0.9911
Best Attention	0.994	0.977	0.959	0.929	0.907	0.842	0.9918
Our results:							
Transformer	0.9733	0.9412	0.9316	0.9167	0.9014	0.8834	0.9616
Universal Transformer	0.9934	0.9712	0.9690	0.9400	0.9206	0.8915	0.9917
Adapt. Univ. Transf. (small)	0.9932	**0.9801**	**0.9714**	**0.9608**	**0.9521**	**0.9314**	0.9920
Adapt. Univ. Transf. (base)	**0.9943**	0.9720	0.9516	0.9567	0.9314	0.9034	**0.9924**

Table 2: Accuracy on the subject-verb agreement number prediction task (higher is better).

Experiments

- The LAMBADA task is a language modeling task consisting of predicting a missing target word given its (target) sentence and a broader context of 4-5 preceding sentences.

Model	LM Perplexity & (Accuracy)			RC Accuracy		
	control	dev	test	control	dev	test
Neural Cache [10]	**129**	139	-	-	-	-
Dhingra et al. [7]	-	-	-	-	-	0.5569
Transformer	154 (0.14)	5336 (0.0)	9725 (0.0)	0.4102	0.4401	0.3988
LSTM	138 (0.23)	4966 (0.0)	5174 (0.0)	0.1103	0.2316	0.2007
Universal Transformer	131(0.32)	279 (0.18)	319 (0.17)	**0.4801**	0.5422	0.5216
Adaptive Universal Transformer	130 (0.32)	**135** (0.22)	**142** (0.19)	0.4603	**0.5831**	**0.5625**

Table 3: LAMBADA language modeling (LM) perplexity (lower better) with accuracy in parentheses (higher better), and Reading Comprehension (RC) accuracy results (higher better). '-' indicates no reported results in that setting.

Experiments

- Algorithmic Tasks:

Model	Copy		Reverse		Addition	
	char-acc	seq-acc	char-acc	seq-acc	char-acc	seq-acc
LSTM	0.45	0.09	0.66	0.11	0.08	0.0
Transformer	0.53	0.03	0.13	0.06	0.07	0.0
Universal Transformer	0.91	0.35	0.96	0.46	0.34	0.02
Neural GPU*	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**

Table 4: Accuracy (higher better) on the algorithmic tasks, trained on decimal strings of length 40 and evaluated on length 400 from [17]. *Note that the Neural GPU was trained with a special curriculum to obtain the perfect result, while other models are trained without any curriculum.

Experiments

- Learning to Execute:

Model	Copy		Double		Reverse	
	char-acc	seq-acc	char-acc	seq-acc	char-acc	seq-acc
LSTM	0.78	0.11	0.51	0.047	0.91	0.32
Transformer	0.98	0.63	0.94	0.55	0.81	0.26
Universal Transformer	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**

Table 5: Character-level (*char-acc*) and sequence-level accuracy (*seq-acc*) results on the Memorization LTE tasks, with maximum length of 55.

Model	Program		Control		Addition	
	char-acc	seq-acc	char-acc	seq-acc	char-acc	seq-acc
LSTM	0.53	0.12	0.68	0.21	0.83	0.11
Transformer	0.71	0.29	0.93	0.66	**1.0**	**1.0**
Universal Transformer	**0.89**	**0.63**	**1.0**	**1.0**	**1.0**	**1.0**

Table 6: Character-level (*char-acc*) and sequence-level accuracy (*seq-acc*) results on the Program Evaluation LTE tasks with maximum nesting of 2 and length of 5.

Experiments

- Machine Translation:

Model	**BLEU**
Universal Transformer *small*	26.8
Transformer *base* [31]	28.0
Weighted Transformer *base* [1]	28.4
Universal Transformer *base*	**28.9**

Conclusion

- An new architecture for Transformer.
- More parameters or more biases?
- Theoretical support for context-specific?