Attention-based Model and Its Application in Scene Text Recognition

Baoguang Shi (石葆光)

Huazhong University of Science and Technology

March 23, 2016

Part 1: Introduction to Attention-Based Models

Introduction

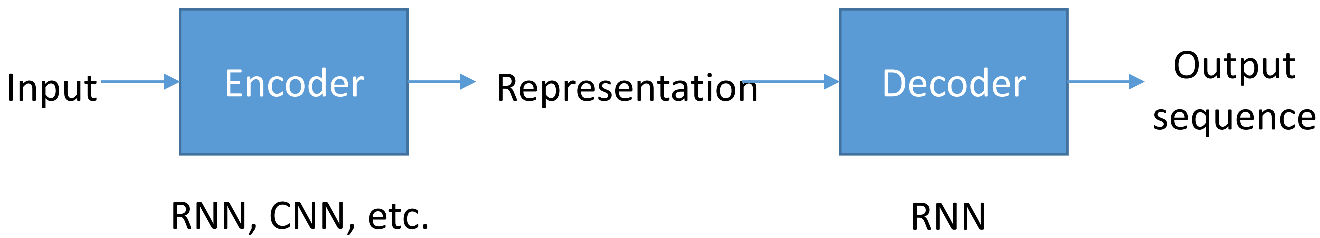
- The problem to solve: predicting a *sequence* given an input, with deep neural networks.
- Input can be: image, speech, sentence, etc.
- Why it matters?
  - Speech recognition: speech signal sequence => transcription sequence
  - Image captioning: image => word sequence
  - Machine translation: word sequence (in one language) => word sequence (in another)
  - …

Main Difficulties

- Outputs are variable-length sequences
- Inputs may also have unfixed number of dimensions

Attention-based models [1-2]

- An encoder-decoder framework

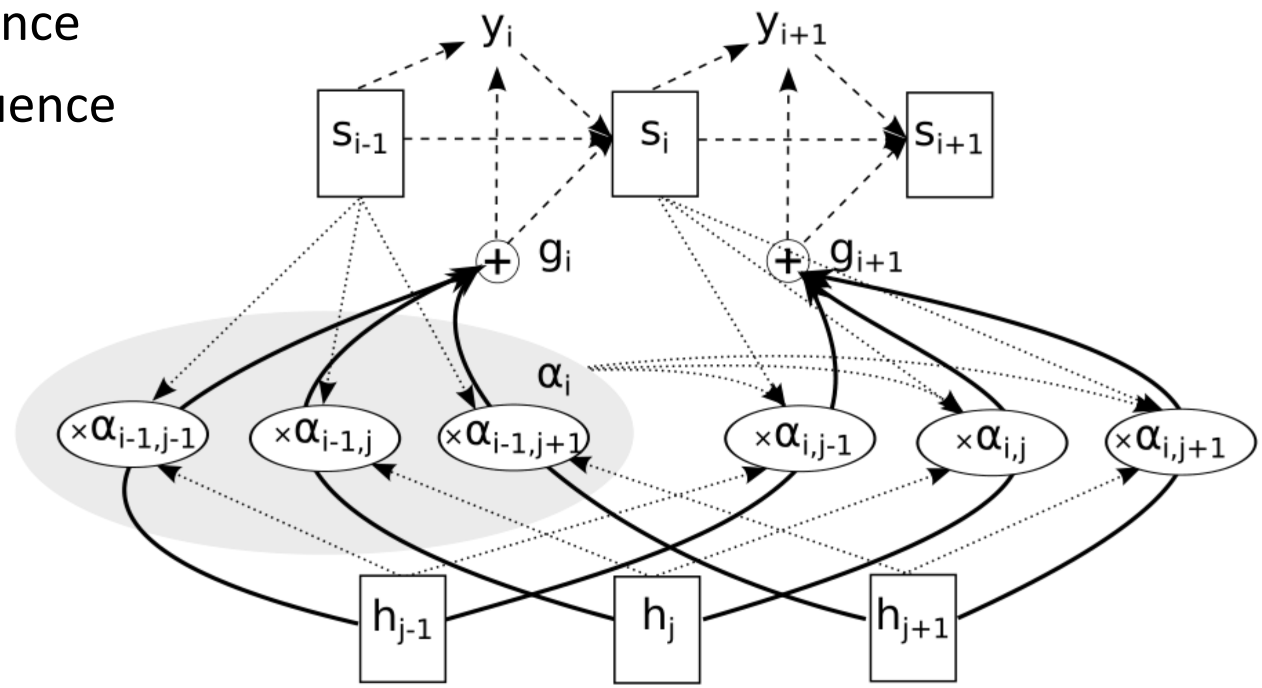


- At each step,
  - Select *relevant contents* in the representation (*attend*)
  - Generate a token

[1] Dzmitry Bahdanau, Kyunghyun Cho, Yoshua Bengio: *Neural Machine Translation by Jointly Learning to Align and Translate*. ICLR 2015.
[2] Jan Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, KyungHyun Cho, Yoshua Bengio: *Attention-Based Models for Speech Recognition*. CoRR abs/1506.07503 (2015)

First Look

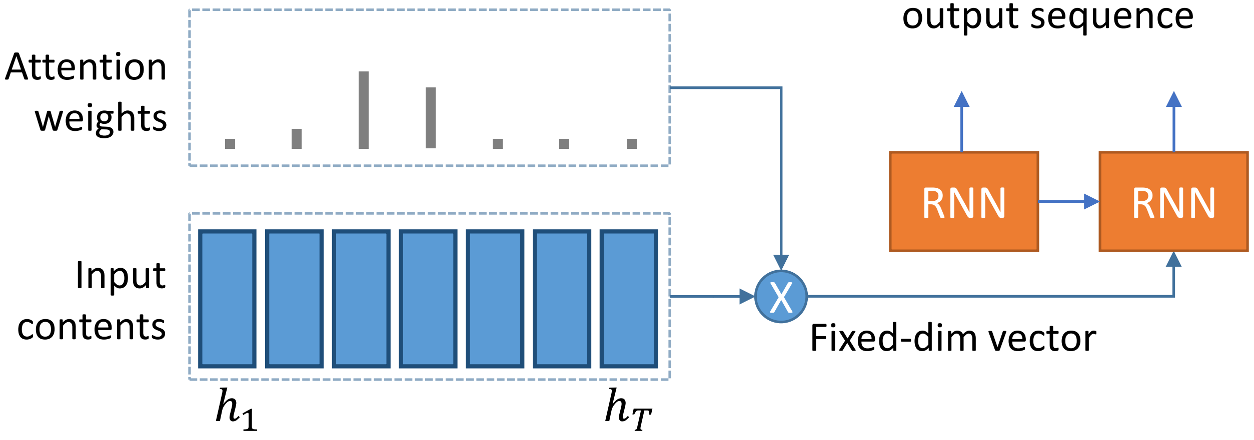
- h : input sequence
- y : output sequence



[1] Jan Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, KyungHyun Cho, Yoshua Bengio: *Attention-Based Models for Speech Recognition*. CoRR abs/1506.07503 (2015)

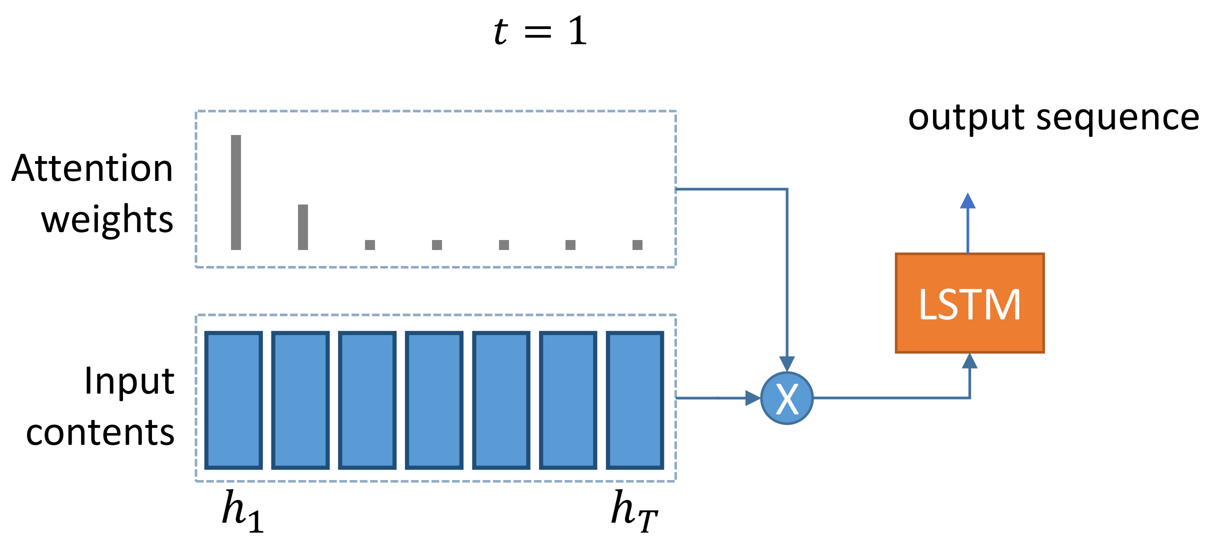
Key: Attention Weights

- At each step,
  - Calculates a vector of attention weights (non-negative, sum to 1)
  - Linearly combines the input vectors into a *glimpse* vector
- Convert variable-length inputs into a fixed-dimensional vector



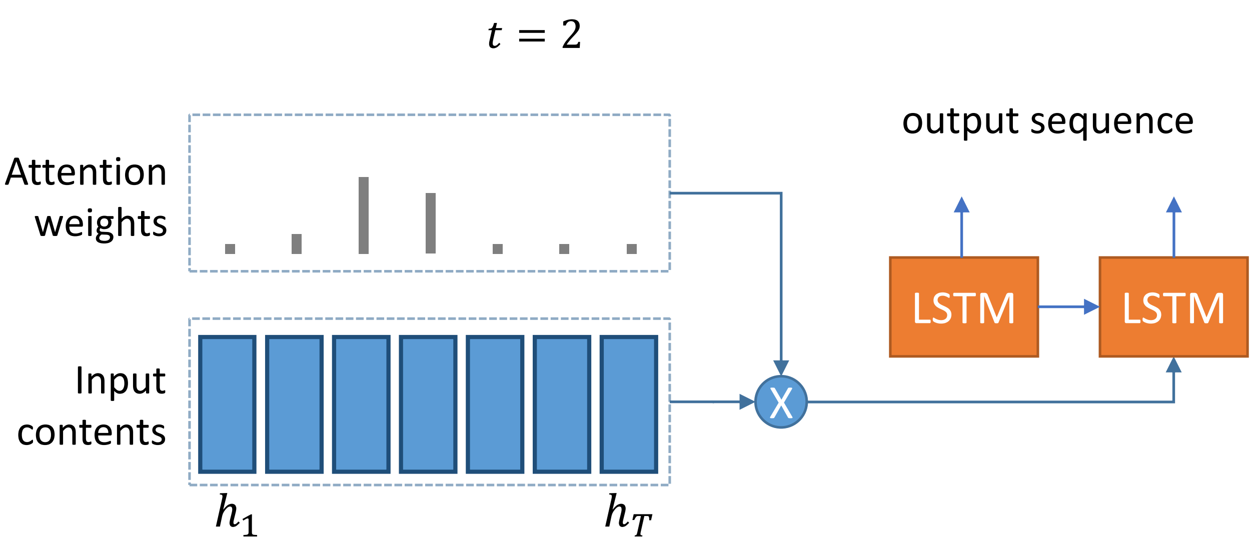
Different Weights at Every Step

- Attend to different contents



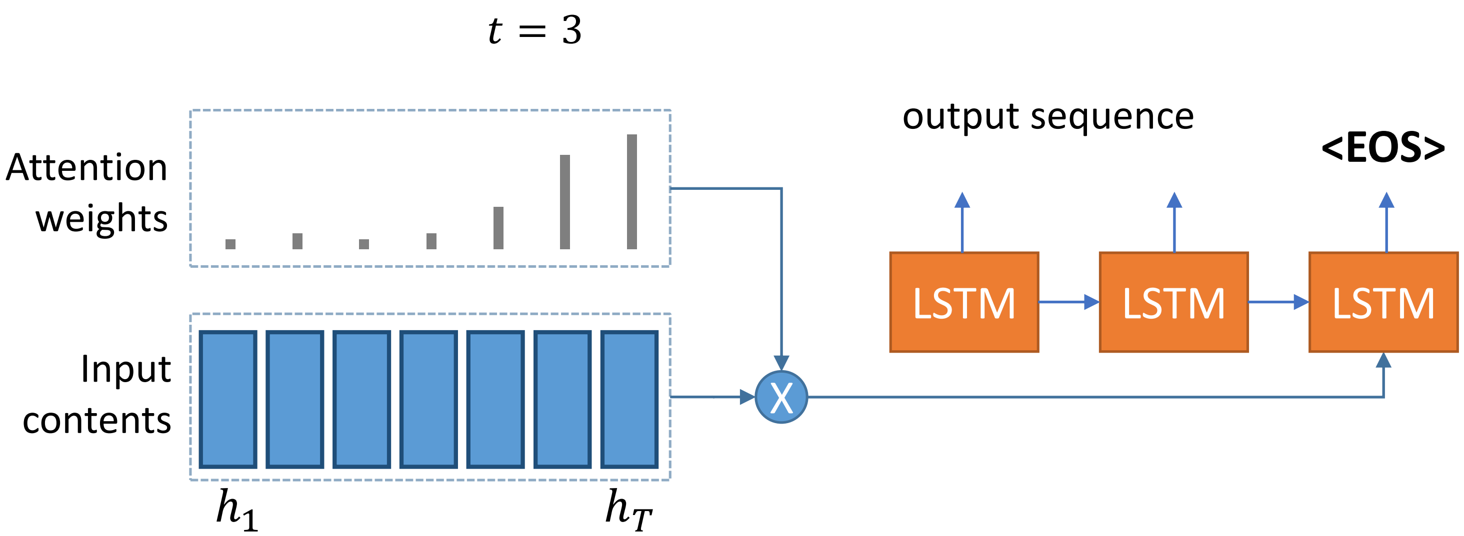
Different Weights at Every Step

- Attend to different contents

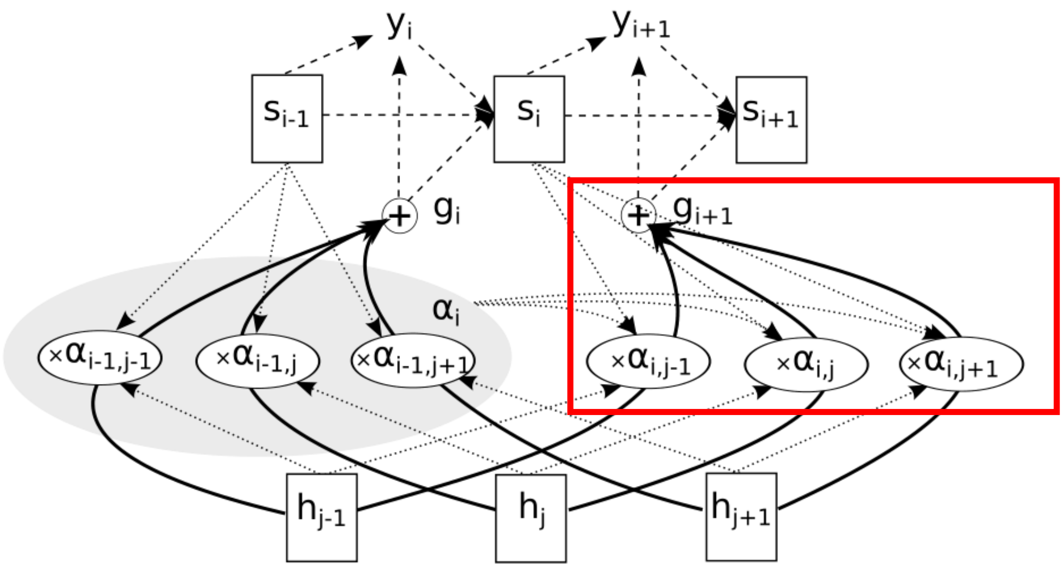


Different Weights at Every Step

- Attend to different contents



Detailed architecture



$$\alpha_i = Attend(s_{i-1}, \alpha_{i-1}, h)$$

$$f_i = F * \alpha_{i-1}$$

$$e_{i,j} = w^\top \tanh(W s_{i-1} + V h_j + U f_{i,j} + b)$$

$$\alpha_{i,j} = \exp(e_{i,j}) \Big/ \sum_{j=1}^{L} \exp(e_{i,j}) \, .$$

$$g_i = \sum_{j=1}^{L} \alpha_{i,j} h_j$$

The Attention Mechanism

- Allows us to predict a sequence from input contents.
- Allows the model to be trained ent-to-end.

What do attention weights tell us?

- Indicate the importance of inputs for each output token
- Provides the *soft-alignment* between inputs and outputs

Attention weights (2D)

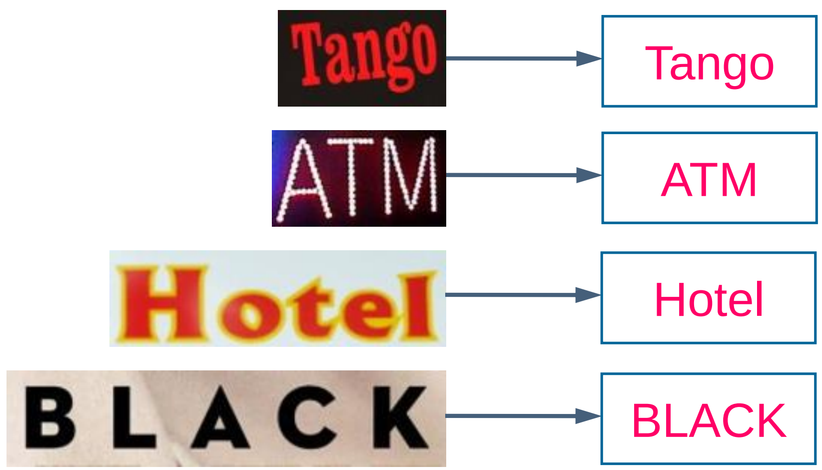
(b) A woman is throwing a frisbee in a park.

[1] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C. Courville, Ruslan Salakhutdinov, Richard S. Zemel, Yoshua Bengio: Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. ICML 2015

Part 2: Attention-based Models for Scene Text Recognition

Scene Text Recognition

- A problem of image to sequence learning

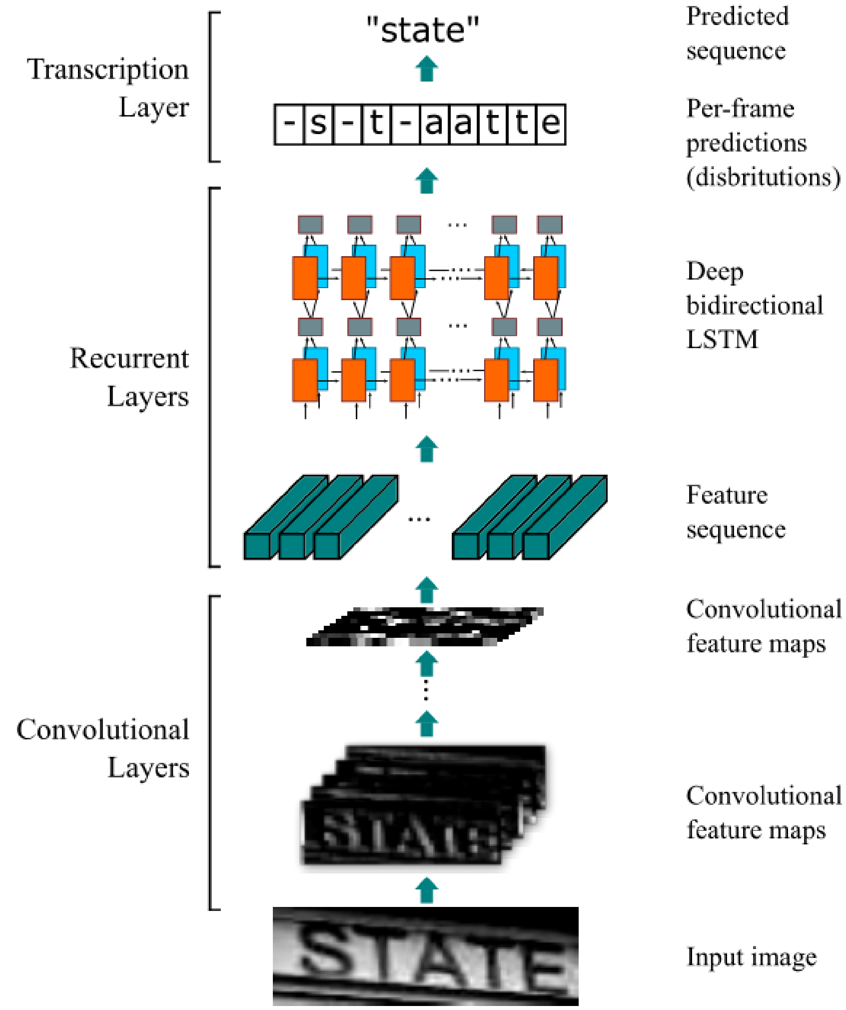


Traditional Approaches

- Character detection
- Character recognition
- Generate word from characters

Our Previous Work: CRNN

- Convolutional Recurrent Neural Network
  - Convolutional layers
  - Bidirectional LSTM
  - CTC layer



- Code & model released at https://github.com/bgshih/crnn

Our approach

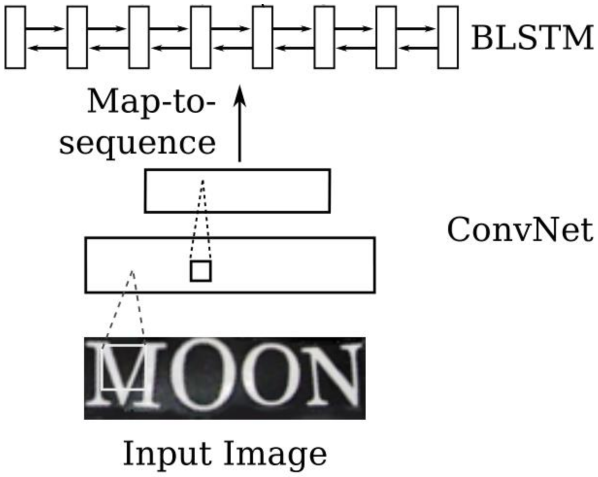
- Scheme: sequence-to-sequence learning
  - Sequence-based image representation
  - To character sequence

[‘S’,‘A’,‘L’,‘E’,<EOS>]

- Encoder: Convolutional layers + LSTM, extract sequence-based representation
- Decoder: Attention-based RNN, generate character sequence

Encoder

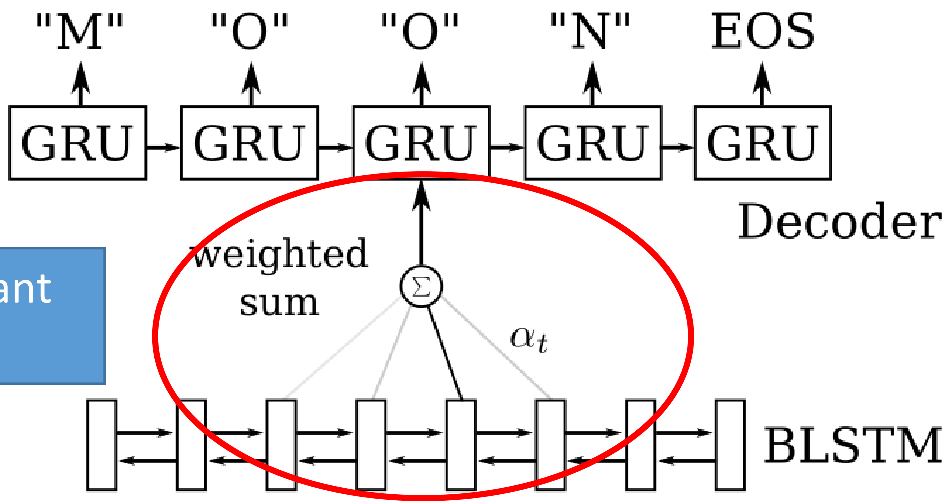
- Extract a sequence-based representation of image
  - Structure: Convolutional layers + Bidirectional-LSTM
  - Convolutional layers extract feature maps of size $C \times H \times W$
  - Split feature maps along columns, into W vectors with CH dimensions (map-to-sequence conversion)
  - A Bidirectional-LSTM models the context within the sequence



Decoder

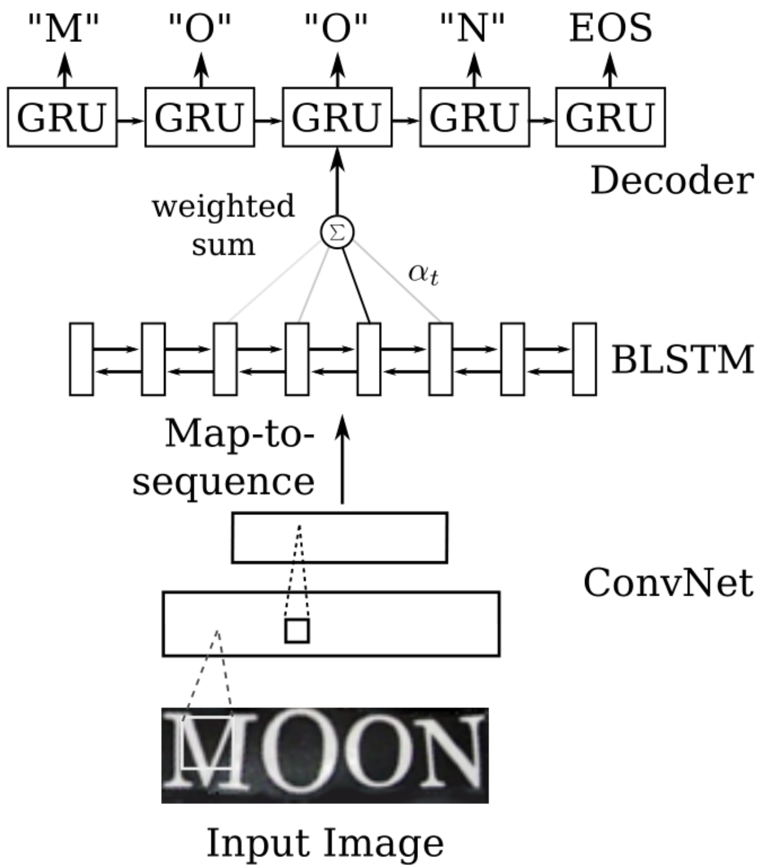
- Attention-based RNN, whose cells are Gated Recurrent Units (GRU)

Attention: select relevant contents



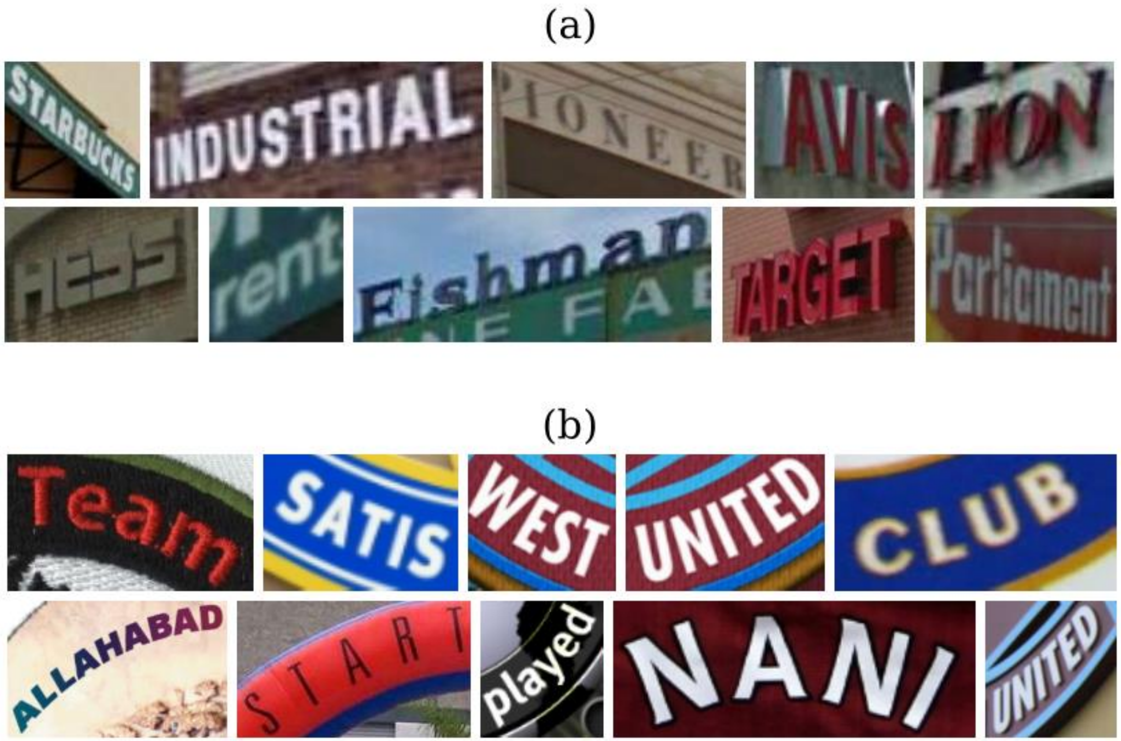
Sequence Recognition Network: The Whole Structure

- Components
  - Convolutional layer
  - Bidirectional-LSTM
  - Attention-based decoder



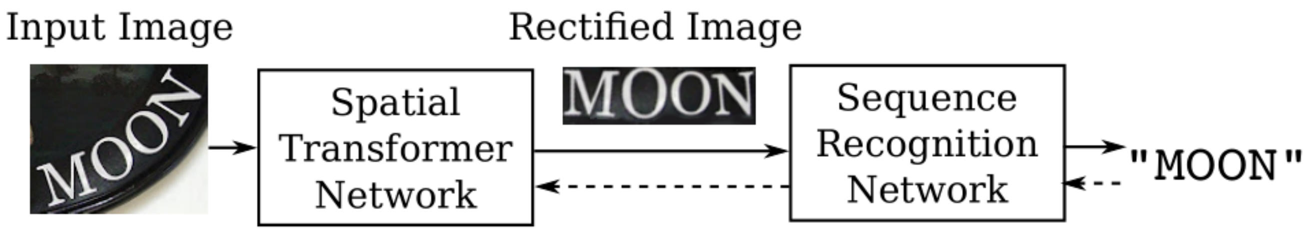
However…

- This scheme does not work well on *irregular text*



Rectification + Recognition

- Rectifying images using a Spatial Transformer Network [1] (STN).
- Recognizing rectified images using the network mentioned above (SRN).
- STN and SRN are trained jointly.

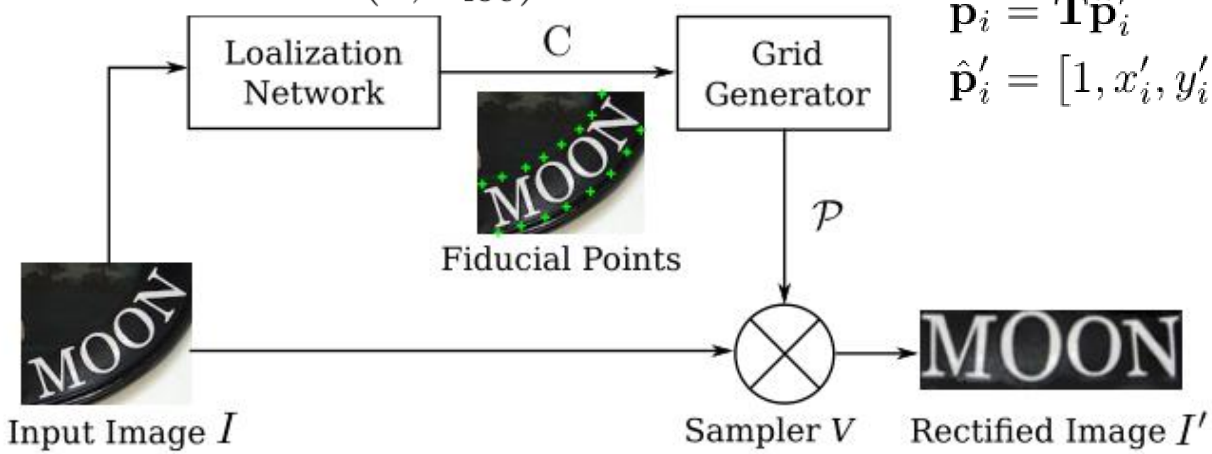


[1] Max Jaderberg, Karen Simonyan, Andrew Zisserman, Koray Kavukcuoglu: Spatial Transformer Networks. CoRR abs/1506.02025 (2015)

Rectification with STN

- Given an input image,
  - regress the locations of 20 *fiducial points* on the input image
  - calculate a TPS transformation
  - transform the input image

$$\mathbf{C} = \mathrm{CNN}(I, \boldsymbol{\theta}_{\mathrm{loc}})$$



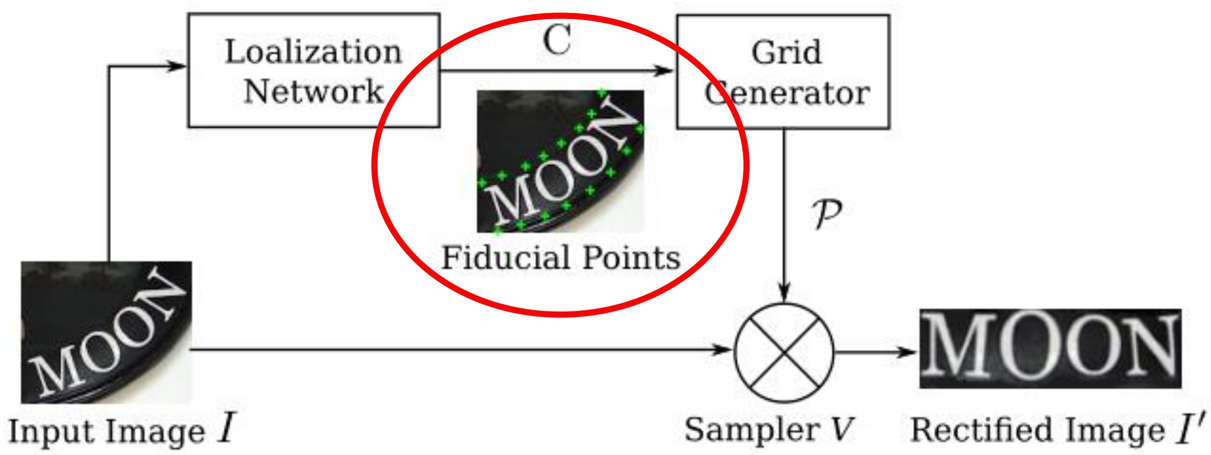
$$\mathbf{T} = \left(\boldsymbol{\Delta}_{\mathbf{C}'}^{-1} \begin{bmatrix} \mathbf{C}^{\intercal} \\ \mathbf{0}^{3\times 2} \end{bmatrix} \right)^{\intercal}$$

$$\mathbf{p}_i = \mathbf{T}\hat{\mathbf{p}}'_i$$

$$\hat{\mathbf{p}}'_i = [1, x'_i, y'_i, r'_{i,1}, \ldots, r'_{i,K}]^{\intercal}$$

End-to-end trainable

- No need to label the fiducial points manually, but let STN learns by itself



Performance

- Significant improvement on datasets that focuses on irregular text

SVT-Perspective (perspective text)

Table 2. Recognition accuracies on SVT-Perspective [26]. “50”and “Full” mean 50-word lexicon and full lexicon respectively. “None” means recognition without a lexicon.

Method	50	Full	None
Wang *et al.* [34]	40.5	26.1	-
Mishra *et al.* [22]	45.7	24.7	-
Wang *et al.* [36]	40.2	32.4	-
Phan *et al.* [26]	75.6	67.0	-
Shi *et al.* [30]	**92.6**	72.6	66.8
Ours	91.2	**77.4**	**71.8**

CUTE80 (curved text)

Table 3. Recognition accuracies on CUTE80 [26].

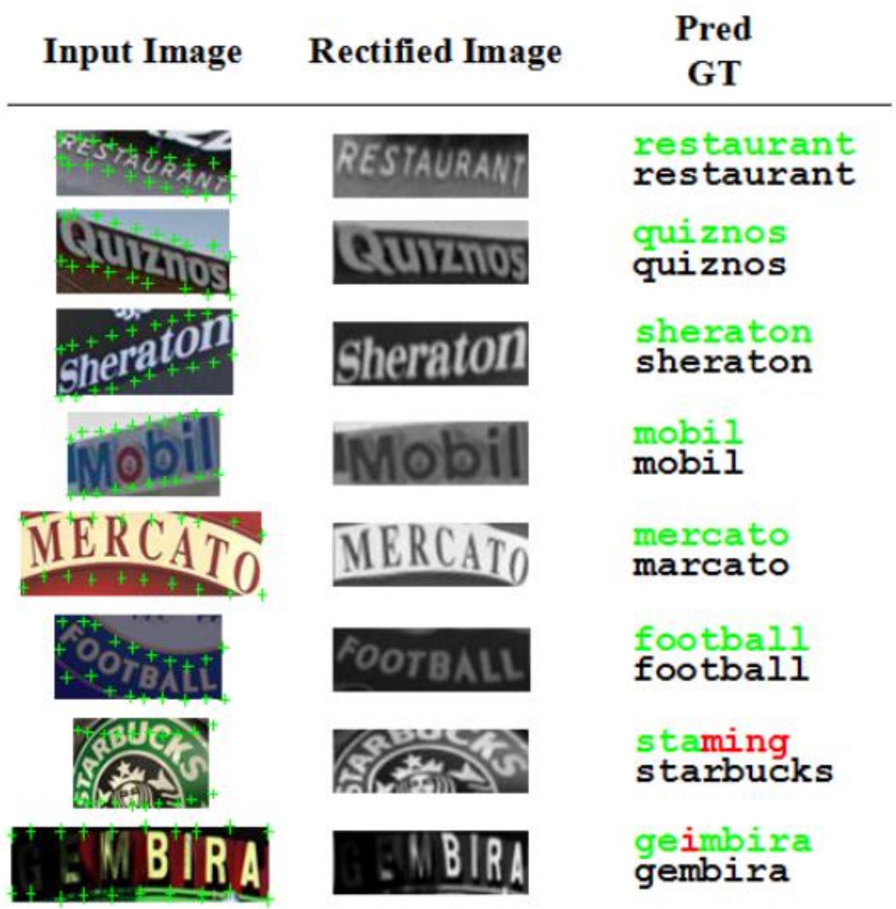
Method	Accuracy
Jaderberg *et al.* [16]	42.7
Shi *et al.* [30]	54.9
Ours	**59.2**

Performance

- State-of-the-art, or highly competitive results on general text recognition datasets

Method	IIIT5k			SVT		IC03				IC13
	50	1k	None	50	None	50	Full	50k	None	None
ABBYY [34]	24.3	-	-	35.0	-	56.0	55.0	-	-	-
Wang *et al.* [34]	-	-	-	57.0	-	76.0	62.0	-	-	-
Mishra *et al.* [22]	64.1	57.5	-	73.2	-	81.8	67.8	-	-	-
Wang *et al.* [36]	-	-	-	70.0	-	90.0	84.0	-	-	-
Goel *et al.* [10]	-	-	-	77.3	-	89.7	-	-	-	-
Bissacco *et al.* [5]	-	-	-	90.4	78.0	-	-	-	-	87.6
Alsharif and Pineau [3]	-	-	-	74.3	-	93.1	88.6	85.1	-	-
Almazán *et al.* [2]	91.2	82.1	-	89.2	-	-	-	-	-	-
Yao *et al.* [38]	80.2	69.3	-	75.9	-	88.5	80.3	-	-	-
Rodríguez-Serrano *et al.* [28]	76.1	57.4	-	70.0	-	-	-	-	-	-
Jaderberg *et al.* [18]	-	-	-	86.1	-	96.2	91.5	-	-	-
Su and Lu [32]	-	-	-	83.0	-	92.0	82.0	-	-	-
Gordo [11]	93.3	86.6	-	91.8	-	-	-	-	-	-
Jaderberg *et al.* [16]	97.1	92.7	-	95.4	80.7	**98.7**	**98.6**	93.3	**93.1**	**90.8**
Jaderberg *et al.* [15]	95.5	89.6	-	93.2	71.7	97.8	97.0	93.4	89.6	81.8
Shi *et al.* [30]	**97.6**	**94.4**	78.2	**96.4**	80.8	**98.7**	97.6	**95.5**	89.4	86.7
Ours	96.2	93.8	**81.9**	95.5	**81.9**	98.3	96.2	94.8	90.1	88.6
Ours (SRN only)	96.5	92.8	79.7	96.1	81.5	97.8	96.4	93.7	88.7	87.5

Some Results

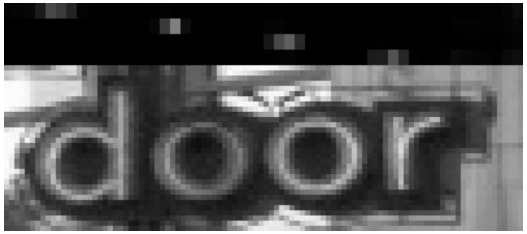


Recognition & Character Localization

- Row-t is the vector of attention weight at step t

"billiards"

"door"



"hertz"

"restaurant"

"central"

"everest"

Advantages of the Proposed Model

- Globally trainable learning system
  - Learning representation from data
  - End-to-end trainable
- Handles images of arbitrary sizes, and text of arbitrary length
  - The encoder accepts images of arbitrary widths
  - For the decoder, both input and output sequences can have arbitrary lengths
- Robust to irregular text

Takeaways

- Attention-based models predict *sequences* given input images/speeches/sentences/etc.
- Attention-weights provide soft-alignment between inputs and outputs
- The rectification + recognition scheme is effective for scene text recognition

Thanks!

- Paper: Baoguang Shi, Xinggang Wang, Pengyuan Lyu, Cong Yao, Xiang Bai, *Robust Scene Text Recognition with Automatic Rectification*. Accepted to CVPR 2016.
- Preprint available at http://arxiv.org/abs/1603.03915

- CRNN code & model: https://github.com/bgshih/crnn