Deep Learning

15 Encoder-Decoder Models & Attention

Dr. Konda Reddy Mopuri
Dept. of AI, IIT Hyderabad
Jan-May 2023

Language Model

1. $y^* = \text{argmax } P(y_t / y_1, y_2 \ldots y_{t-1})$

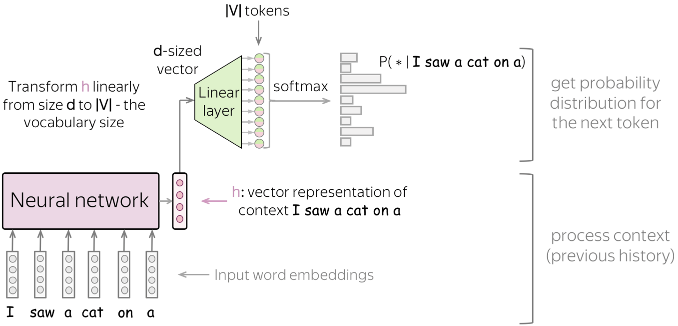
Language Model

1. $y^* = \text{argmax } P(y_t / y_1, y_2 \dots y_{t-1})$
2. We have an NN (e.g. RNN or LSTM) first consuming the i/p sequence $(y_1^{t-1}) \rightarrow$ representation for the context

Language Model

1. $y^* = \text{argmax } P(y_t/y_1, y_2 \dots y_{t-1})$
2. We have an NN (e.g. RNN or LSTM) first consuming the i/p sequence $(y_1^{t-1}) \rightarrow$ representation for the context
3. Then, predict the probability distribution $P(y_t/y_1, y_2 \dots y_{t-1})$ over the vocabulary

Language Model



Credits: Elena Voita

Encoder-Decoder Framework

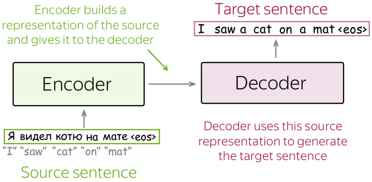
1. Standard modeling paradigm for sequence-to-sequence tasks

Encoder-Decoder Framework

1. Standard modeling paradigm for sequence-to-sequence tasks
2. Consists of two components: **Encoder** and **Decoder**

Encoder-Decoder Framework

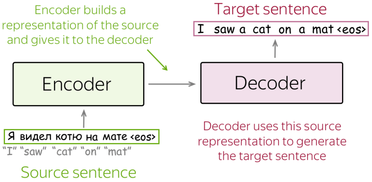
1. **Encoder**: reads source sequence to produce its representation



Credits: Elena Voita

Encoder-Decoder Framework

1. **Encoder**: reads source sequence to produce its representation
2. **Decoder**: uses the source representation given by the encoder to infer the target sequence



Credits: Elena Voita

Encoder-Decoder Model

1. Language modeling learns $p(y)$, where $y = (y_1, y_2, \ldots y_n)$ is a sequence of tokens

Encoder-Decoder Model

1. Language modeling learns $p(y)$, where $y = (y_1, y_2, \ldots y_n)$ is a sequence of tokens
2. Seq2Seq need to model the conditional probability $p(y/x)$ of a sequence y given a sequence x (source or context)

Encoder-Decoder Model

1. Language modeling learns $p(y)$, where $y = (y_1, y_2, \ldots y_n)$ is a sequence of tokens
2. Seq2Seq need to model the conditional probability $p(y/x)$ of a sequence y given a sequence x (source or context)
3. Note that x need not be a sequence always (e.g. image in captioning)

Encoder-Decoder Model

1. Hence, Seq2Seq tasks can be modelled as conditional language models

Language Models: $P(y_1, y_2, \ldots, y_n) = \prod_{t=1}^{n} p(y_t | y_{<t})$

Conditional Language Models: $P(y_1, y_2, \ldots, y_n, | x) = \prod_{t=1}^{n} p(y_t | y_{<t}, x)$

condition on source x

Credits: Elene Voita

Encoder-Decoder Model

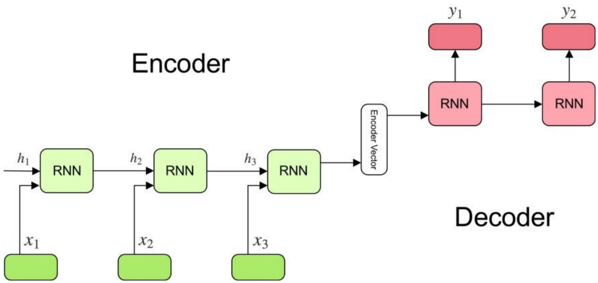
1. Basis for a lot of applications
   - Image (or video) captioning
   - Textual entailment
   - Machine translation
   - Transliteration
   - Document summarization
   - VQA: Visual Question Answering
   - Video classification
   - Chatbot for dialog

Encoder-Decoder Model

1. Basis for a lot of applications
   - Image (or video) captioning
   - Textual entailment
   - Machine translation
   - Transliteration
   - Document summarization
   - VQA: Visual Question Answering
   - Video classification
   - Chatbot for dialog
2. Let's consider machine translation...

Encoder-Decoder Model

- Simplest model is having two RNNs



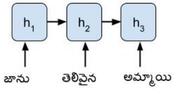
Credits: Simeon Kostadinov

Encoder-Decoder for Machine Translation

Input sequence: $x_1, x_2, \ldots x_T$

Output sequence: $y_1, y_2, \ldots y_{T'}$

Encoder: $h_t = E(x_t, h_{t-1})$



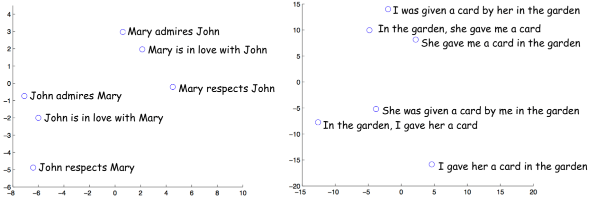
Sequence to sequence learning by Sutskever et al. NeurIPS 2014

Encoder-Decoder for Machine Translation

- Hope is that
  - final encoder state 'encodes' all the information about the source
  - this vector is sufficient for the decoder to generate the target sentence

Encoder-Decoder for Machine Translation

- Representations of sentences with similar meaning but different structure are close!



Sequence to sequence learning by Sutskever et al. NeurIPS 2014

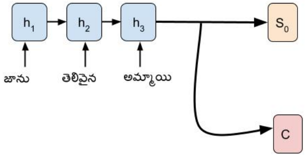
Encoder-Decoder for Machine Translation

Input sequence: $x_1, x_2, \ldots x_T$

Output sequence: $y_1, y_2, \ldots y_{T'}$

Last hidden state $h_T \rightarrow$ Initial state of the Decoder S_0 and the context information C
E.g. $S_0 \leftarrow h_T$ + dense layers, and $C \leftarrow h_T$

Encoder: $h_t = E(x_t, h_{t-1})$



Sequence to sequence learning by Sutskever et al. NeurIPS 2014

Encoder-Decoder for Machine Translation

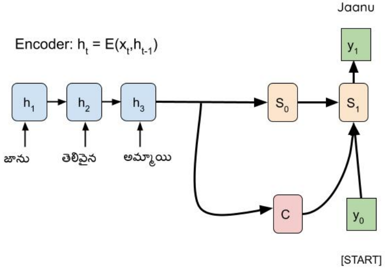
Input sequence: $x_1, x_2, \dots x_T$

Output sequence: $y_1, y_2, \dots y_{T'}$

Last hidden state $h_T \rightarrow$ Initial state of the Decoder S_0 and the context information C
E.g. $S_0 \leftarrow h_T$ + dense layers, and $C \leftarrow h_T$

Decoder: $s_t = D(y_{t-1}, s_{t-1}, C)$

Encoder: $h_t = E(x_t, h_{t-1})$



Sequence to sequence learning by Sutskever et al. NeurIPS 2014

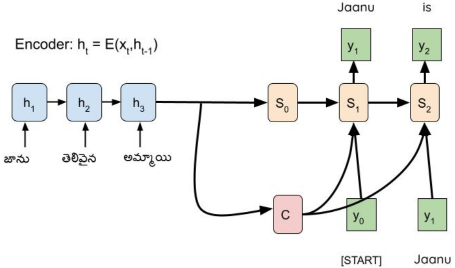
Encoder-Decoder for Machine Translation

Input sequence: $x_1, x_2, \ldots, x_T$

Output sequence: $y_1, y_2, \ldots, y_{T'}$

Last hidden state $h_T \rightarrow$ Initial state of the Decoder S_0 and the context information C
E.g. $S_0 \leftarrow h_T$ + dense layers, and $C \leftarrow h_T$

Decoder: $s_t = D(y_{t-1}, s_{t-1}, C)$



Sequence to sequence learning by Sutskever et al. NeurIPS 2014

Encoder-Decoder for Machine Translation

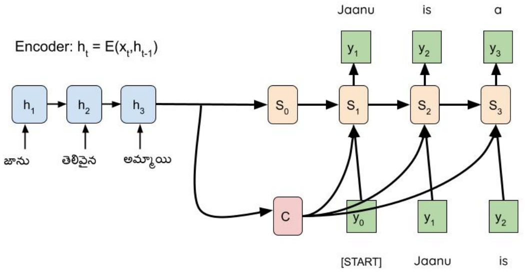
Input sequence: $x_1, x_2, \dots x_T$

Output sequence: $y_1, y_2, \dots y_{T'}$

Last hidden state $h_T \rightarrow$ Initial state of the Decoder S_0 and the context information C
E.g. $S_0 \leftarrow h_T$ + dense layers, and $C \leftarrow h_T$

Decoder: $s_t = D(y_{t-1}, s_{t-1}, C)$

Encoder: $h_t = E(x_t, h_{t-1})$



Sequence to sequence learning by Sutskever et al. NeurIPS 2014

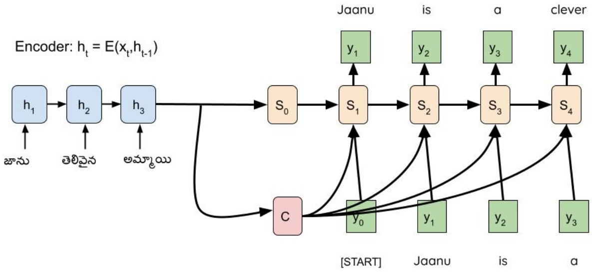
Encoder-Decoder for Machine Translation

Input sequence: $x_1, x_2, \ldots x_T$

Output sequence: $y_1, y_2, \ldots y_{T'}$

Last hidden state $h_T \rightarrow$ Initial state of the Decoder S_0 and the context information C
E.g. $S_0 \leftarrow h_T$ + dense layers, and $C \leftarrow h_T$

Decoder: $s_t = D(y_{t-1}, s_{t-1}, C)$



Sequence to sequence learning by Sutskever et al. NeurIPS 2014

Encoder-Decoder for Machine Translation

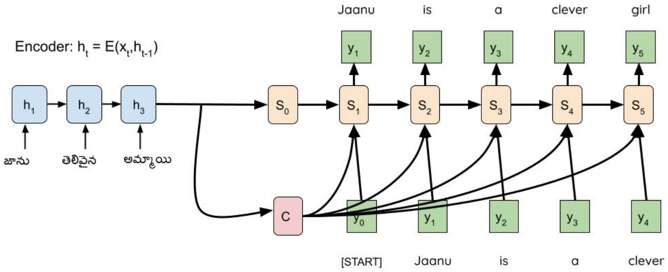
Input sequence: $x_1, x_2, \ldots\ x_T$

Output sequence: $y_1, y_2, \ldots\ y_{T'}$

Last hidden state $h_T \rightarrow$ Initial state of the Decoder S_0 and the context information C
E.g. $S_0 \leftarrow h_T$ + dense layers, and $C \leftarrow h_T$

Decoder: $s_t = D(y_{t-1}, s_{t-1}, C)$

Encoder: $h_t = E(x_t, h_{t-1})$



Sequence to sequence learning by Sutskever et al. NeurIPS 2014

Encoder-Decoder for Machine Translation

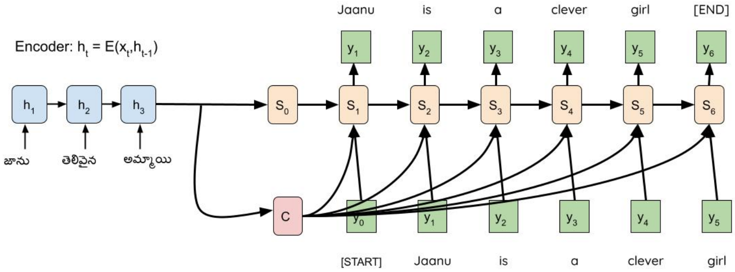
Input sequence: $x_1, x_2, \dots x_T$

Output sequence: $y_1, y_2, \dots y_{T'}$

Last hidden state $h_T \rightarrow$ Initial state of the Decoder S_0 and the context information C

E.g. $S_0 \leftarrow h_T$ + dense layers, and $C \leftarrow h_T$

Decoder: $s_t = D(y_{t-1}, s_{t-1}, C)$



Sequence to sequence learning by Sutskever et al. NeurIPS 2014

Encoder-Decoder for Machine Translation

1. Encoder got only a single vector to encode the entire source sequence

Encoder-Decoder for Machine Translation

1. Encoder got only a single vector to encode the entire source sequence
2. Harsh compression, may lead to encoder forgetting something!

Encoder-Decoder for Machine Translation

1. Encoder got only a single vector to encode the entire source sequence
2. Harsh compression, may lead to encoder forgetting something!
3. Different information may be relevant for the decoder at different time steps

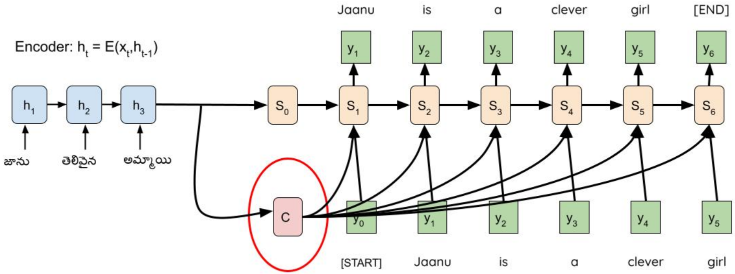
Encoder-Decoder for Machine Translation

Input sequence: $x_1, x_2, \dots x_T$

Output sequence: $y_1, y_2, \dots y_{T'}$

Last hidden state $h_T \rightarrow$ Initial state of the Decoder S_0 and the context information C
E.g. $S_0 \leftarrow h_T$ + dense layers, and $C \leftarrow h_T$

Decoder: $s_t = D(y_{t-1}, s_{t-1}, C)$



Bottleneck: Entire input is summarized by this vector!

Sequence to sequence learning by Sutskever et al. NeurIPS 2014

Encoder-Decoder for Machine Translation

Input sequence: $x_1, x_2, \dots x_T$

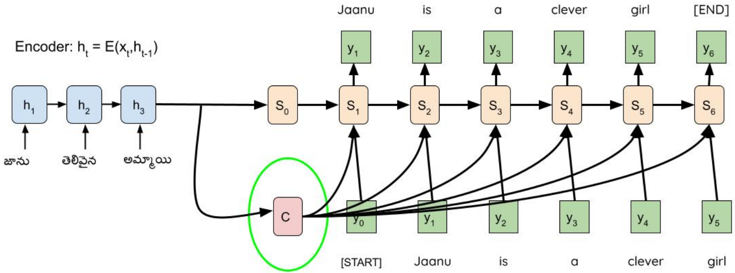
Output sequence: $y_1, y_2, \dots y_{T'}$

Last hidden state $h_T \rightarrow$ Initial state of the Decoder S_0 and the context information C

E.g. $S_0 \leftarrow h_T$ + dense layers, and $C \leftarrow h_T$

Decoder: $s_t = D(y_{t-1}, s_{t-1}, C)$

Encoder: $h_t = E(x_t, h_{t-1})$



Solution: use different context at each time step!

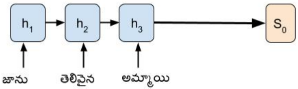
Sequence to sequence learning by Sutskever et al. NeurIPS 2014

Encoder-Decoder for Machine Translation with Attention

Input sequence: $x_1, x_2, \ldots. x_T$

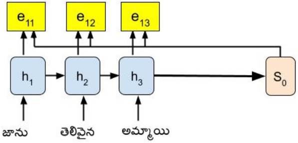
Input sequence: $y_1, y_2, \ldots. y_{T'}$

Encoder: $h_t = E(x_t, h_{t-1})$



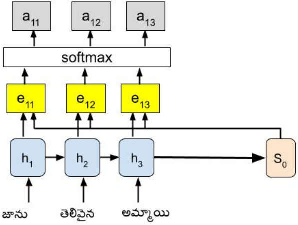
Encoder-Decoder for Machine Translation with Attention

Compute the alignment scores
$e_{t,i} = f_{att}(s_{t-1}, h_i)$ f_{att} - couple of dense layers



Neural Machine Translation with aligning by Bahdanau et al. ICLR 2015

Encoder-Decoder for Machine Translation with Attention

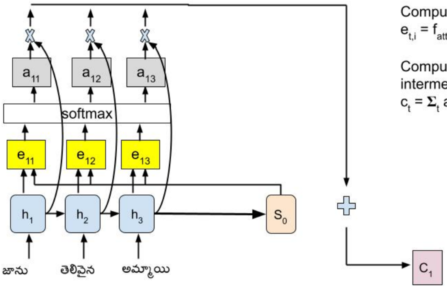


Compute the alignment scores

$e_{t,i} = f_{att}(s_{t-1}, h_i)$ f_{att} - couple of dense layers

Neural Machine Translation with aligning by Bahdanau et al. ICLR 2015

Encoder-Decoder for Machine Translation with Attention



Compute the alignment scores

$e_{t,i} = f_{att}(s_{t-1}, h_i)$ f_{att} - couple of dense layers

Compute the context as a linear combination of intermediate hidden states

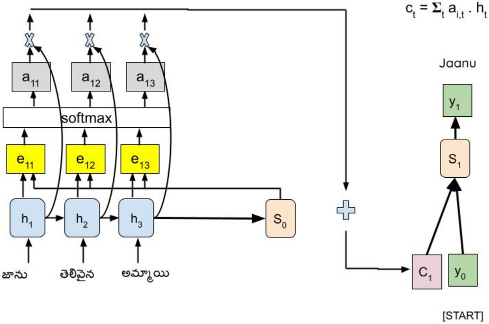
$c_t = \Sigma_t \, a_{i,t} \cdot h_t$

Neural Machine Translation with aligning by Bahdanau et al. ICLR 2015

Encoder-Decoder for Machine Translation with Attention

Compute the alignment scores
$e_{t,i} = f_{att}(s_{t-1}, h_i)$ f_{att} - couple of dense layers

Compute the context as a linear combination of intermediate hidden states
$c_t = \Sigma_t \, a_{i,t} \cdot h_t$



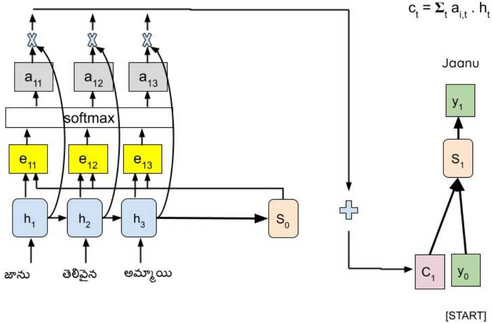
Decoder: $s_t = D(y_{t-1}, C_t)$

Neural Machine Translation with aligning by Bahdanau et al. ICLR 2015

Encoder-Decoder for Machine Translation with Attention

Compute the alignment scores
$e_{t,i} = f_{att}(s_{t-1}, h_i)$ f_{att} - couple of dense layers

Compute the context as a linear combination of intermediate hidden states
$c_t = \Sigma_t a_{i,t} \cdot h_t$

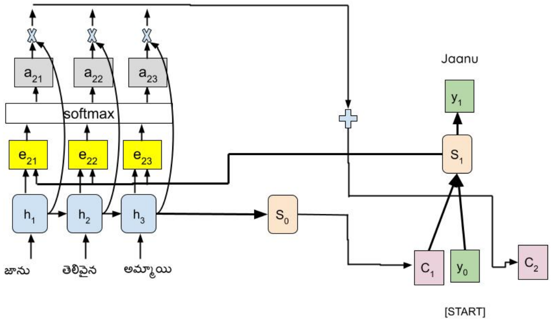


Decoder: $s_t = D(y_{t-1}, C_t)$

All these operations are differentiable!
Attention is learned using backprop!!

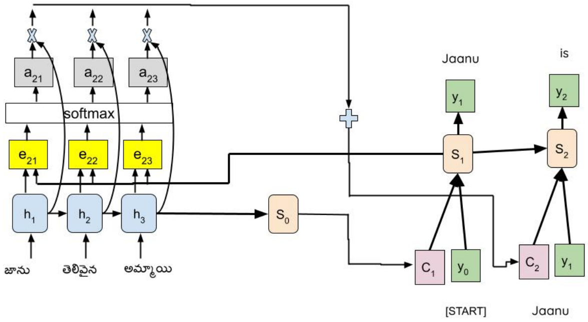
Neural Machine Translation with aligning by Bahdanau et al. ICLR 2015

Encoder-Decoder for Machine Translation with Attention



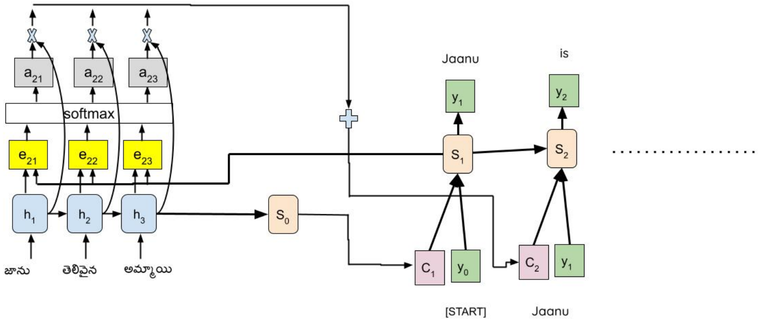
Neural Machine Translation with aligning by Bahdanau et al. ICLR 2015

Encoder-Decoder for Machine Translation with Attention



Neural Machine Translation with aligning by Bahdanau et al. ICLR 2015

Encoder-Decoder for Machine Translation with Attention



Neural Machine Translation with aligning by Bahdanau et al. ICLR 2015

Encoder-Decoder for Machine Translation with Attention

- Employs a different context at each time step of decoding

Neural Machine Translation with aligning by Bahdanau et al. ICLR 2015

Encoder-Decoder for Machine Translation with Attention

- Employs a different context at each time step of decoding
- No more bottleneck-ing of the input

Neural Machine Translation with aligning by Bahdanau et al. ICLR 2015

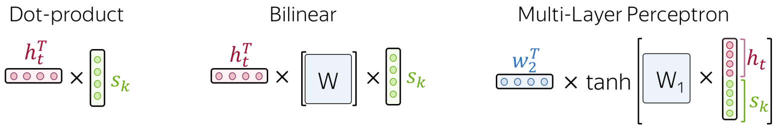
Encoder-Decoder for Machine Translation with Attention

- Employs a different context at each time step of decoding
- No more bottleneck-ing of the input
- Decoder can 'attend' to different portions of the input at each time step

---

Neural Machine Translation with aligning by Bahdanau et al. ICLR 2015

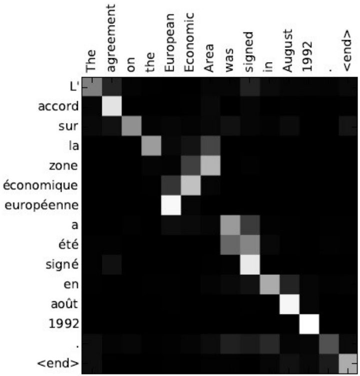
Encoder-Decoder for Machine Translation with Attention

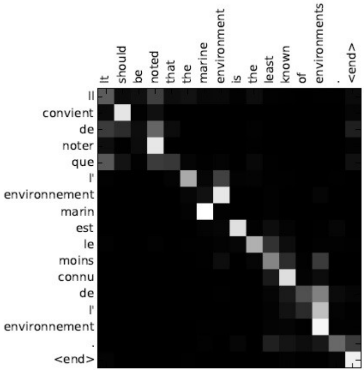


$$\text{score}(h_t, s_k) = h_t^T\ s_k \qquad \text{score}(h_t, s_k) = h_t^T\ W s_k \qquad \text{score}(h_t, s_k) = w_2^T \cdot \tanh(W_1[h_t, s_k])$$

Computing Attention
(Credits: Elene Voita)

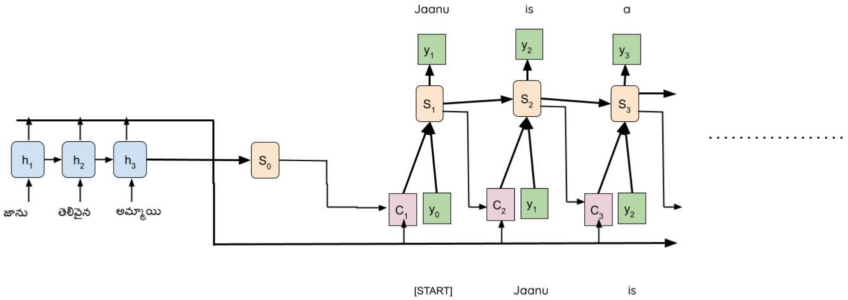
Encoder-Decoder for Machine Translation with Attention





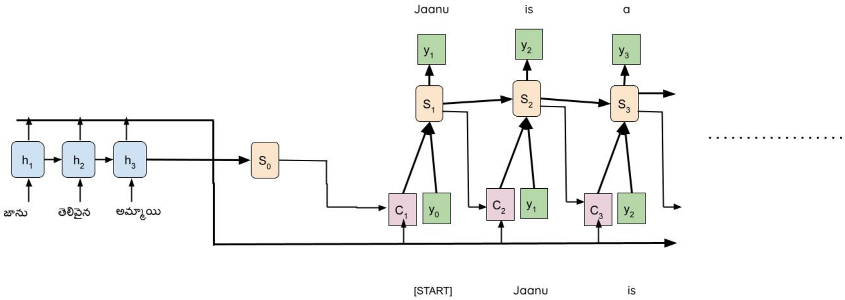
Neural Machine Translation with aligning by Bahdanau et al. ICLR 2015

Encoder-Decoder for Machine Translation with Attention



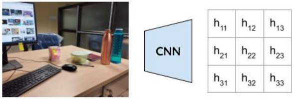
- Decoder doesn't consider the h_i to be an ordered set

Encoder-Decoder for Machine Translation with Attention



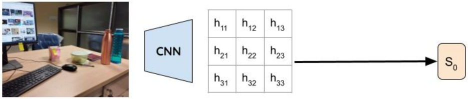
- Decoder doesn't consider the h_i to be an ordered set
- This architecture can be exploited to process a set of inputs h_i

Image captioning using RNNs with Attention



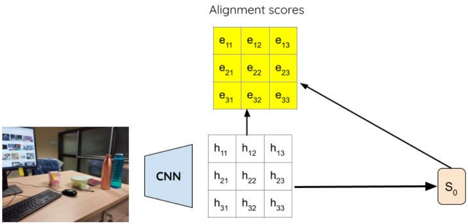
Show Attend and Tell by Xu et al. 2015

Image captioning using RNNs with Attention



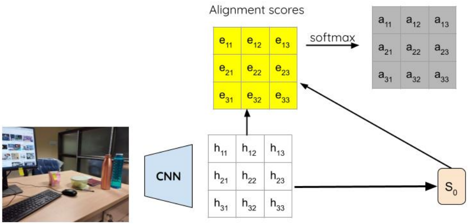
Show Attend and Tell by Xu et al. 2015

Image captioning using RNNs with Attention



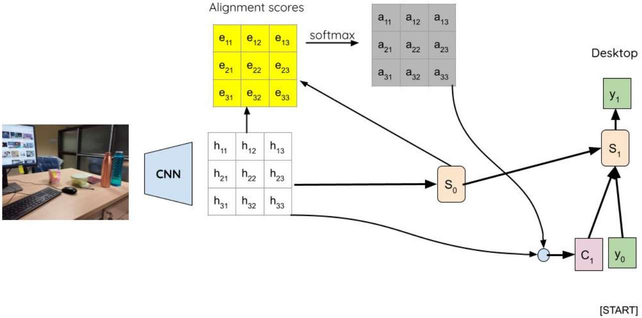
Show Attend and Tell by Xu et al. 2015

Image captioning using RNNs with Attention



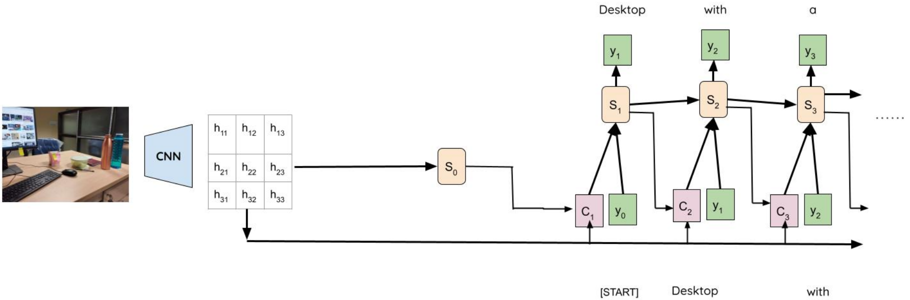
Show Attend and Tell by Xu et al. 2015

Image captioning using RNNs with Attention



Show Attend and Tell by Xu et al. 2015

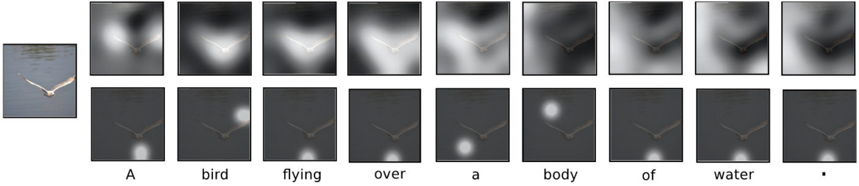
Image captioning using RNNs with Attention



Show Attend and Tell by Xu et al. 2015

Image captioning using RNNs with Attention

Figure 2. Attention over time. As the model generates each word, its attention changes to reflect the relevant parts of the image. "soft" (top row) vs "hard" (bottom row) attention. (Note that both models generated the same captions in this example.)



Show Attend and Tell by Xu et al. 2015

Image captioning using RNNs with Attention

A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

A stop sign is on a road with a mountain in the background.

A little girl sitting on a bed with a teddy bear.

A group of people sitting on a boat in the water.

A giraffe standing in a forest with trees in the background.

Show Attend and Tell by Xu et al. 2015