TTIC 31230, Fundamentals of Deep Learning

David McAllester, Winter 2020

Machine Translation and Attention

Review of Auto-regressive Language Modeling

An auto-regressive language model defines

$$P_\Phi(w_t \mid w_0, \cdots, w_{t-1})$$

Training an Auto-regressive Language Model

At train time the full sentence is given and the loss function is given by

$$\mathcal{L} = -\ln P_\Phi(w_0, \ldots, w_T) = \sum_t -\ln P_\Phi(w_t \mid w_0, \ldots, w_{t-1})$$

Sampling from an Auto-regressive Language Model

Draw w_0 from $P_\Phi(w_0)$,

$t = 0$

While $w_t \neq <\mathrm{EOS}>$

  Draw word w_{t+1} from $P_\Phi(w_{t+1} \mid w_0, \cdots, w_t)$.

  increment t

Machine Translation

$$w_0, \ldots, w_{T_{\mathrm{in}}} \Rightarrow \tilde{w}_0, \ldots, \tilde{w}_{T_{\mathrm{out}}}$$

Translation is a **sequence to sequence** (seq2seq) task.

Sequence to Sequence Learning with Neural Networks, Sutskever, Vinyals and Le, NeurIPS 2014, arXiv Sept 10, 2014.

Machine Translation

We define a model

$$P_\Phi\left(\tilde{w}_0, \ldots, \tilde{w}_{T_{\mathrm{out}}} \mid w_0, \ldots, w_{T_{\mathrm{in}}}\right)$$

$$\begin{aligned}
\Phi^* &= \operatorname*{argmin}_\Phi \; E_{\mathrm{Pop}} \; \left[- \ln \; P_\Phi\left(\tilde{w}_0, \ldots, \tilde{w}_{T_{\mathrm{out}}} \mid w_0, \ldots, w_{T_{\mathrm{in}}}\right)\right] \\
&= \operatorname*{argmin}_\Phi \; E_{\langle x, y \rangle \sim \mathrm{Pop}} \; [- \ln P_\Phi(y|x)]
\end{aligned}$$

Translation Using Thought Vectors

The final state of a **right-to-left (backward)** RNN is viewed as a **"thought vector"** representation of the input sentence.

We use the input thought vector $\overleftarrow{h}_{\mathrm{in}}[0, J]$ as the initial hidden state of a **left-to-right (forward)** RNN language model generating the output sentence.

Computing the input thought vector backward provides a good start to the forward generation of the output.

Machine Translation Decoding

We can sample a translation

$$w_t \sim P(w_t \mid \overleftarrow{h}_{\mathrm{in}}[0, J],\ w_0, \ldots, w_{t-1})$$

But typically we do a greedy decoding

$$w_t = \operatorname*{argmax}_{w_t}\ P(w_t \mid \overleftarrow{h}_{\mathrm{in}}[0, J],\ w_0, \ldots, w_{t-1})$$

Machine Translation Using Vector Sequences

Neural Machine Translation by Jointly Learning to Align and Translate Dzmitry Bahdanau, Kyunghyun Cho, Yoshua Bengio, ICLR 2015 (arXiv Sept. 1, 2014)

Machine Translation Using Vector Sequences

$$P_\Phi(w_{t_{\text{out}}} \mid \overleftrightarrow{h}_{\text{in}}[T_{\text{in}}, J],\ w_0, \cdots, w_{t_{\text{out}}-1})$$

$$= \operatorname*{softmax}_{w_{t_{\text{out}}}}\ e[w_{t_{\text{out}}}, J]\, \vec{h}_{\text{out}}[t_{\text{out}} - 1, J]$$

Computing $\vec{h}_{\mathrm{out}}[t_{\mathrm{out}}, J]$

$$\text{Attention } P(t_{\mathrm{in}}|t_{\mathrm{out}}): \quad \alpha[t_{\mathrm{out}}, t_{\mathrm{in}}] = \mathrm{softmax}_{t_{\mathrm{in}}}\; e[w_{t_{\mathrm{out}}}, J]\; \overset{\leftrightarrow}{h}_{\mathrm{in}}[t_{\mathrm{in}}, J]$$

$$\text{Weighted Sum}: \quad \tilde{h}_{\mathrm{in}}[t_{\mathrm{out}}, J] = \alpha[t_{\mathrm{out}}, T_{\mathrm{in}}]\; \overset{\leftrightarrow}{h}_{\mathrm{in}}[T_{\mathrm{in}}, J]$$

$$\text{RNN}: \quad \vec{h}_{\mathrm{out}}[0, J] = \overleftarrow{h}_{\mathrm{in}}[0, J/2]; \vec{h}_{\mathrm{in}}[T_{\mathrm{in}}, J/2]$$

$$\text{RNN}: \quad \vec{h}_{\mathrm{out}}[t_{\mathrm{out}}, J] = \mathrm{CELL}(\vec{h}_{\mathrm{out}}[t_{\mathrm{out}}, J],\; e[w_{t_{\mathrm{out}}}, I],\; \tilde{h}_{\mathrm{in}}[t_{\mathrm{out}}, J])$$

Attention

$$\alpha[t_{\mathrm{out}}, t_{\mathrm{in}}] = \operatorname*{softmax}_{t_{\mathrm{in}}} \; e[w_{t_{\mathrm{out}}}, J] \; \overset{\leftrightarrow}{h}_{\mathrm{in}} [t_{\mathrm{in}}, J]$$

$$\tilde{h}_{\mathrm{in}}[t_{\mathrm{out}}, J] = \alpha[t_{\mathrm{out}}, T_{\mathrm{in}}] \; \overset{\leftrightarrow}{h}_{\mathrm{in}} [T_{\mathrm{in}}, J]$$

$\tilde{h}_{\mathrm{in}}[t_{\mathrm{out}}, J]$ is a convex combination of vectors $\overset{\leftrightarrow}{h}_{\mathrm{in}} [t_{\mathrm{in}}, J]$.

More generally, attention computes a convex combination of vectors where the combination weights are computed by a softmax of an inner product with a "query" vector (such as $e[w_{t_{\mathrm{out}}}, J]$ above).

Attention in Image Captioning

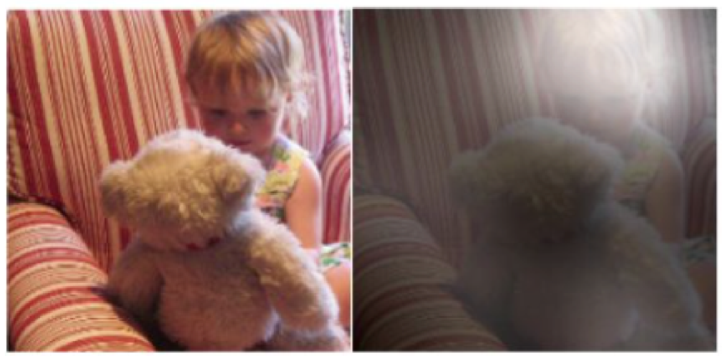
We can treat image captioning as translating an image into a caption.

In translation with attention involves an attention over the input aligning output words with positions in the input.

For each output word we get an attention over the image positions.

Attention in Image Captioning

A woman is throwing a frisbee in a park.



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

Xu et al. ICML 2015

Further Comments on Decoding

We can sample a translation

$$w_t \sim P(w_t \mid \overleftarrow{h}_{\mathrm{in}}[0, J],\; w_0, \ldots, w_{t-1})$$

Typically we do a greedy decoding

$$w_t = \operatorname*{argmax}_{w_t}\; P(w_t \mid \overleftarrow{h}_{\mathrm{in}}[0, J],\; w_0, \ldots, w_{t-1})$$

or we might try maximize total probability.

$$w_0, \ldots, w_{T_{\mathrm{out}}} = \operatorname*{argmax}_{w_0, \ldots, w_{T_{\mathrm{out}}}}\; P_\Phi\left(w_0, \ldots, w_{T_{\mathrm{out}}} \mid \overleftarrow{h}_{\mathrm{in}}[0, J]\right)$$

Greedy Decoding vs. Beam Search

We would like

$$W_{\mathrm{out}}[T_{\mathrm{out}}]^* = \operatorname*{argmax}_{W_{\mathrm{out}}[T_{\mathrm{out}}]} \; P_\Phi(W_{\mathrm{out}}[T_{\mathrm{out}}] \mid W_{\mathrm{in}}[T_{\mathrm{in}}])$$

But a greedy algorithm may do well

$$w_t = \operatorname*{argmax}_{w_t} \; P_\Phi(w_t \mid W_{\mathrm{in}}[T_{\mathrm{in}}],\; w_0, \ldots, w_{t-1})$$

But these are not the same.

Example

"Those apples are good" vs. "Apples are good"

$$P_\Phi(\text{Apples are Good } \texttt{<eos>}) > P_\Phi(\text{Those apples are good } \texttt{<eos>})$$

$$P_\Phi(\text{Those}|\varepsilon) > P_\Phi(\text{Apples}|\varepsilon)$$

Beam Search

At each time step we maintain a list the K best words and their associated hidden vectors.

This can be used to produce a list of k "best" decodings which can then be compared to select the most likely one.

END