Deep Learning

7. Attention Layers

Lars Schmidt-Thieme

Information Systems and Machine Learning Lab (ISMLL)
Institute for Computer Science
University of Hildesheim, Germany

Syllabus

Tue. 21.4.	(1)	1. Supervised Learning (Review 1)
Tue. 28.4.	(2)	2. Neural Networks (Review 2)
Tue. 5.5.	(3)	3. Regularization for Deep Learning
Tue. 12.5.	(4)	4. Optimization for Training Deep Models
Tue. 19.5.	(5)	5. Convolutional Neural Networks
Tue. 26.5.	(6)	6. Recurrent Neural Networks
Tue. 2.6.	—	— *Pentecoste Break* —
Tue. 9.6.	(7)	7. Autoencoders
Tue. 16.6.	(8)	8. Attention Layers
Tue. 23.6.	(9)	9. Generative Adversarial Networks
Tue. 30.6.	(10)	10. Recent Advances
Tue. 7.7.	(11)	11. Engineering Deep Learning Models
Tue. 14.7.	(12)	Q & A

Outline

1. Modelling Pairwise Interactions

2. Attention Layers

3. Equivariant Layers and Order-Dependence

4. Example: Machine Translation

Outline

1. Modelling Pairwise Interactions

2. Attention Layers

3. Equivariant Layers and Order-Dependence

4. Example: Machine Translation

From Linear Models to Pairwise Interactions

- M predictors: $x \in \mathbb{R}^M$, scalar output $y \in \mathbb{R}$
- linear model (= polynomial model of order 1):

$$y(x) := \sum_{m=1}^{M} w_m x_m = w^T x$$

- Q: How can we model pairwise interactions between predictors?

Polynomial Model of Order 2

- M predictors: $x \in \mathbb{R}^M$, scalar output $y \in \mathbb{R}$
- linear model (= polynomial model of order 1):

$$y(x) := \sum_{m=1}^{M} w_m x_m = w^T x$$

- polynomial model of order 2:

$$y(x) := \sum_{m=1}^{M} \sum_{m'=1}^{M} W_{m,m'} x_m x_{m'} = x^T W x, \quad W \in \mathbb{R}^{M \times M}$$

Matrix Factorization

- two nominal predictors x_1, x_2
  - represented by indicators in $x_1 \in \{0,1\}^A$ and $x_2 \in \{0,1\}^B$
  - e.g., for recommender systems:
    ratings y depending on users (A) and items (B)

$$\begin{aligned} y(x_1, x_2) &:= (Wx_1)^T (Vx_2), \quad W \in \mathbb{R}^{K\times A}, V \in \mathbb{R}^{K\times B}, K \in \mathbb{N} \text{ latent dimension} \\ &= x_1^T W^T V x_2 \\ &= x^T \begin{pmatrix} 0 & \frac{1}{2} W^T V \\ \frac{1}{2} V^T W & 0 \end{pmatrix} x, \quad x := \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \end{aligned}$$

- $=$ polynomial model of order 2,
  with low rank approximation of the interaction matrix $\tilde{W}$
  of special form:

$$= x^T \tilde{W} x, \quad \tilde{W} := \begin{pmatrix} 0 & \frac{1}{2} W^T V \\ \frac{1}{2} V^T W & 0 \end{pmatrix}$$

Factorization Machine

- M predictors: $x \in \mathbb{R}^M$, scalar output $y \in \mathbb{R}$

$$\begin{aligned} y(x) &:= \sum_{m=1}^{M} \sum_{m'=1}^{M} W_{.,m}^T V_{.,m'} x_m x_{m'}, \quad W, V \in \mathbb{R}^{K \times M} \\ &= x W^T V x \end{aligned}$$

- $=$ polynomial model of order 2, with low rank approximation of the interaction matrix $\tilde{W}$:

$$= x^T \tilde{W} x, \quad \tilde{W} := W^T V$$

Outline

1. Modelling Pairwise Interactions

2. Attention Layers

3. Equivariant Layers and Order-Dependence

4. Example: Machine Translation

Pairwise Feature Interactions for all Pairs of Instances

- previous section: pairwise feature interaction for a single instance.
  - pairwise interaction
  - of M features
  - of a single pair (x, x) of a single input vector x with itself
  - yielding a scalar output.

- this section: pairwise feature interaction for **all pairs** of instances.
  - pairwise interaction
  - of M features
  - of **all pairs** (x_t, x_s) of two elements of a sequence of length T
  - yielding a $T \times T$ matrix output.

Attention Layer

- sequence of length T, each element with M predictors: $X \in \mathbb{R}^{T \times M}$
- output again of dimensions $T \times K$ (layer)

$$\operatorname{attn}(X)_{t,s} := a((X_{t,.}^T W^T V X_{s,.})), \quad V, W \in \mathbb{R}^{L \times M}, L \in \mathbb{N}$$
$$Z(X)_t := \sum_{s=1}^{T} \operatorname{attn}(X)_{t,s} \, (XU^T)_{s,.}, \quad U \in \mathbb{R}^{K \times M}$$

  - pairwise interactions,
    with low rank approximation of the interaction matrix
    and **nonlinear activation** a
  - output a weighted sum of outputs $(XU^T)_{s,.}$ from all elements s,
    weighted by their pairwise interaction / attention.

Attention Layer / Matrix Notation

- elementwise notation:

$$\text{attn}(X)_{t,s} := a((X_{t,.} W^T V X_{s,.}^T)), \quad V, W \in \mathbb{R}^{K \times M}$$

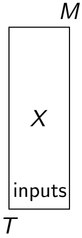
$$Z(X)_t := \sum_{s=1}^{T} \text{attn}(X)_{t,s} \, (XU^T)_{s,.}, \quad U \in \mathbb{R}^{K \times M}$$

⇝ more compact matrix notation:

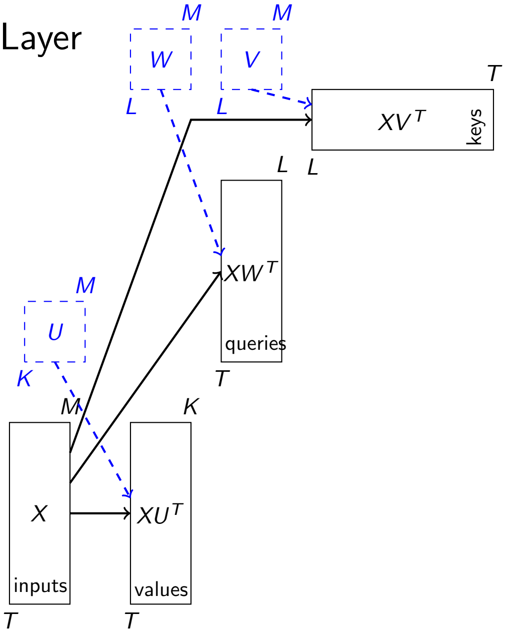
$$\text{attn}(X) = a(XW^T V X^T)$$

$$Z(X) := \text{attn}(X)\, XU^T$$

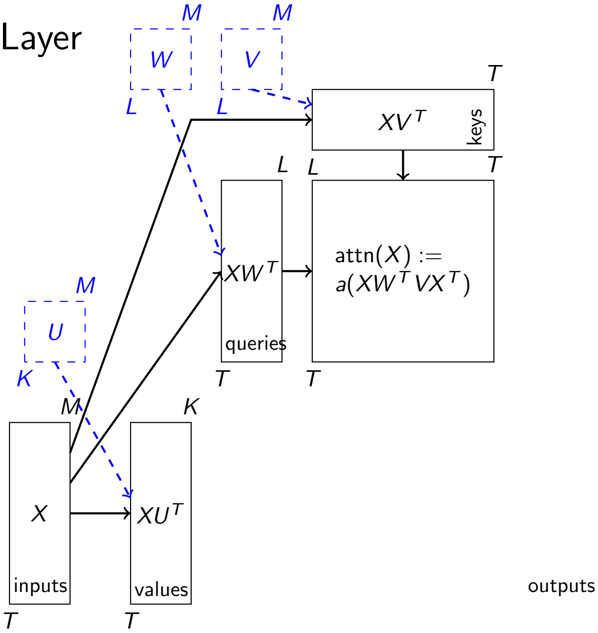
Attention Layer



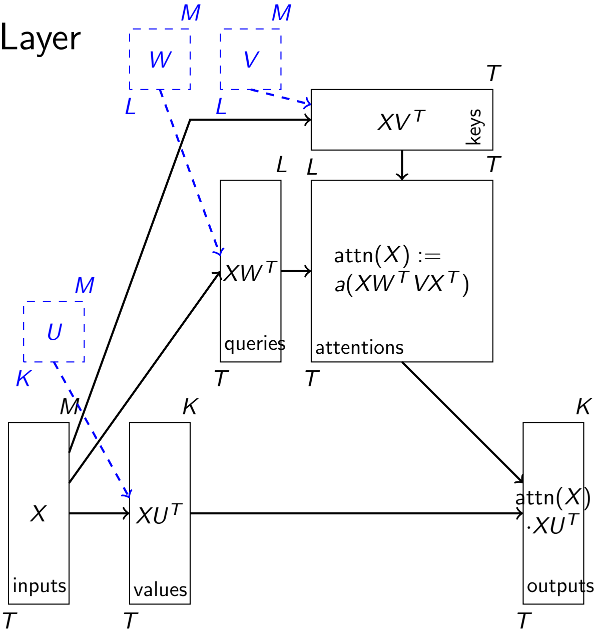
Attention Layer



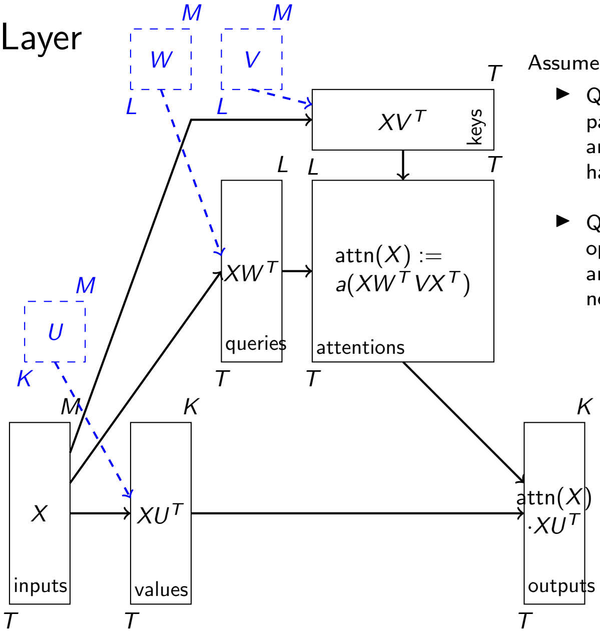
Attention Layer



Attention Layer



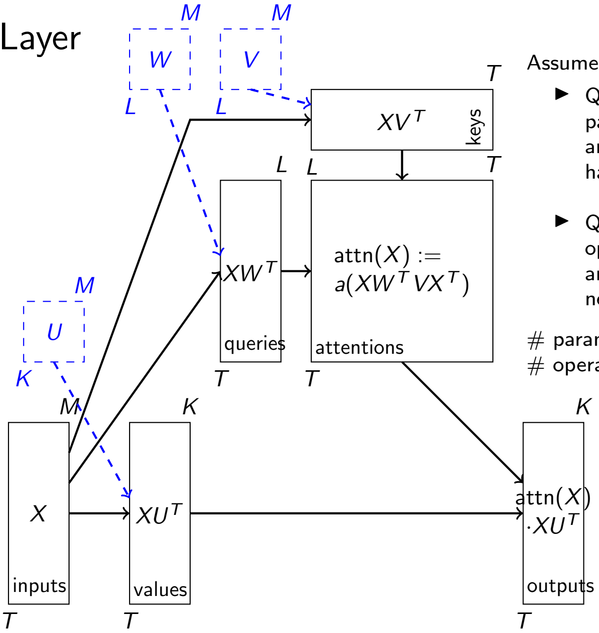
Attention Layer



Assume $M = K = L$.

- Q: How many parameter does an attention layer have?
- Q: How many operations does an attention layer need?

Attention Layer



Assume $M = K = L$.

- Q: How many parameter does an attention layer have?
- Q: How many operations does an attention layer need?

# parameters: $3M^2$
# operations: $O(T^2M)$

Query-Key-Value Metaphor

- originally used activation function:
  - rowwise softmax, scaled:

$$a(Z) := (\mathrm{softmax}(\frac{Z_{t,.}}{\sqrt{L}}))_{t=1:T}$$

- query-key-value metaphor:

$$\mathrm{query}_t := W x_t, \quad \mathrm{key}_t := V x_t, \quad \mathrm{value}_t := U x_t$$

$$z_t = \sum_{s=1}^{T} \mathrm{softmax}(\frac{\mathrm{query}_t^T\,\mathrm{key}_s}{\sqrt{L}})\,\mathrm{value}_s$$

- Vaswani et al. 2017

Multi-Head Attention Layer

- let's call the layer so far **single head attention**:

$$\text{sha}(X; w, v, u) := \text{attn}(X; w, v)\,(XU^T)$$

- **multi-head attention** (with H heads):
  - split the input feature-wise into H parts of size M/H
  - single-head attention for each split
  - concatentate output features again

$$\text{mha}(X) := \text{concat}_2((\text{sha}(X_{1:T,\text{slice}(h)}; W^h, V^h, U^h))_{h=1:H})\,Q^T,$$

$$Q \in \mathbb{R}^{K^{\text{out}} \times HK}$$

$$\text{slice}(h; M/H) := (1 + \frac{hM}{H}, 2 + \frac{hM}{H}, \ldots, \frac{M}{H} + \frac{hM}{H})$$

Note: concat_2 concatenates along columns: $T \times HK$.

Multi-Head Attention Layer

- in effect, the (elementwise) pairwise feature interaction matrix is blockdiagonal:

$$\tilde{W} = \begin{pmatrix} (W^1)^T V_1 & 0 & \dots & 0 \\ 0 & (W^2)^T V_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & (W^H)^T V_H \end{pmatrix}$$

Attention vs. Convolutions

layer	# parameters	# operations	path length
convolutional	KM^2	TKM^2	T/K
attention	$3M^2$	T^2M	1

Attention vs. Convolutions

- **restricted attention**: attention restricted to diagonal band:

$$\text{attnres}(X;K)_{t,s} := \mathbb{I}(|t-s| \leq \frac{K}{2})\ \text{attn}(X)_{t,s}$$

  - **window size** $K \in \mathbb{N}$ allows to trade off # operations vs. path length
  - **path length**: number of operations between z_t and x_s for any s, t.

layer	# parameters	# operations	path length
convolutional	KM^2	TKM^2	T/K
attention	$3M^2$	T^2M	1
restricted attention	$3M^2$	TKM	T/K

T sequence length, M input/embedding dimension, K window size

Outline

1. Modelling Pairwise Interactions

2. Attention Layers

3. Equivariant Layers and Order-Dependence

4. Example: Machine Translation

Separable Layers

- a layer $z : \mathbb{R}^{T\times M} \rightarrow \mathbb{R}^{T\times K}$ is called **separable (in T)** if its output is the stacking of the outputs of some function on single slices each:

$$f(X) = (g(X_{t,.}))_{t=1:T}, \quad \exists g : \mathbb{R}^{M} \rightarrow \mathbb{R}^{K}$$
$$\text{or equiv. } f(X)_t = g(X_{t,.}), \quad \forall t = 1, \ldots, T$$

- i.e., the t-th output element depends only on the t-th input element, and not on any other input elements at s.

- example: convolutions with kernel size 1:

$$f(X) := XW^T, \quad W \in \mathbb{R}^{K\times M}$$
$$\text{as } f(X)_t = X_{t,.}W^T$$

- counterexample: convolutions with kernel size > 1.

Equivariant Layers

- a layer $z : \mathbb{R}^{T\times M} \rightarrow \mathbb{R}^{T\times K}$ is called **equivariant (in T)** if
  - first permuting the input in T and then applying z yields the same result as
  - first applying z and then permuting its output in T (with the same permutation π)

$$z(\pi(X)) = \pi(z(X)), \quad \forall X \in \mathbb{R}^{T\times M}, \forall \pi \in \text{permutations}(\{1, 2, \ldots, T\})$$

$$\pi(X) := (X_{\pi(t),1:M})_{t=1:T}$$

- i.e., the t-th output element depends only on
  - the t-th input element and
  - the (multi)set of all input elements $\{X_{1,.}, X_{2,.}, \ldots, X_{T,.}\}$ (but not their order) — often called **context** in ML.

- examples:
  - all separable layers, e.g., convolutions with kernel size 1.
  - attention layers

- counterexample: convolutions with kernel size > 1.

Make Equivariant Layers Order-Dependend Again

- add the position as additional channel:

$$\tilde{X} := \text{concat}_2(X, (t)_{t=1:T}) \in \mathbb{R}^{T\times(M+1)}$$

- alternatively, one can superimpose the encoded input with a **positional encoding**
  (to make it easier to regress on the position):

$$\begin{aligned} \text{PE}(t)_{2k} &:= \sin(\frac{t}{10000^{2k/K}}), \quad k = 1, \ldots, \frac{K}{2} \\ \text{PE}(t)_{2k+1} &:= \cos(\frac{t}{10000^{2k/K}}) \\ \tilde{X} &:= XW^T + (\text{PE}(t))_{t=1:T}, \quad W \in \mathbb{R}^{K\times M} \end{aligned}$$

Outline

1. Modelling Pairwise Interactions

2. Attention Layers

3. Equivariant Layers and Order-Dependence

4. Example: Machine Translation

Parallel Corpora (Europarl Corpus)

- **parallel corpus**: same text in two (or more) languages.
- example:

English text:

Resumption of the session
I declare resumed the session of the European Parliament adjourned on Friday 17 December 1999, and I would like once again to wish you a happy new year in the hope that you enjoyed a pleasant festive period. Although, as you will have seen, the dreaded 'millennium bug' failed to materialise, still the people in a number of countries suffered a series of natural disasters that truly were dreadful. You have requested a debate on this subject in the course of the next few days, during this part-session.

German text:

Wiederaufnahme der Sitzungsperiode
Ich erkläre die am Freitag, dem 17. Dezember unterbrochene Sitzungsperiode des Europäischen Parlaments für wiederaufgenommen, wünsche Ihnen nochmals alles Gute zum Jahreswechsel und hoffe, daß Sie schöne Ferien hatten. Wie Sie feststellen konnten, ist der gefürchtete ”Millenium-Bug ” nicht eingetreten. Doch sind Bürger einiger unserer Mitgliedstaaten Opfer von schrecklichen Naturkatastrophen geworden. Im Parlament besteht der Wunsch nach einer Aussprache im Verlauf dieser Sitzungsperiode in den nächsten Tagen.

WMT-2014 Task

- Shared Task *Machine Translation* of the ACL 2014 Ninth Workshop on Statistical Machine Translation (WMT-2014)
- parallel datasets:

corpus	size	fr-en	de-en
Europarl v7	628 MB	+	+
Common Crawl Corpus	876 MB	+	+
UN corpus	2.3 GB	+	
News Commentary	77 MB	+	+
10^9 French English corpus	2.3 GB	+	

BLEU score (1/3)

- BLEU: Bilingual Evaluation Understudy

- model a text as a sequence of sentences,
  and a sentence as a sequence of tokens.

- ***n*-gram frequencies** of a sequence y:
  - how often does each sequence a of length n occur in a sequence y:

  $f_n(y) := (|\{i \mid i = 1, \ldots, |y|,\ y_{i:i+|a|-1} = a\}|)_{a \in A^n}$

  e.g., $f_1(\text{"hello"}, \text{"world"}, \text{"hello"}) = \{\text{"hello"} : 2, \text{"world"} : 1\}$

  e.g., $f_2(\text{"hello"}, \text{"world"}, \text{"hello"}) = \{\text{"hello world"} : 1, \text{"world hello"} : 1\}$

- compare two sentences/sequences $y, \hat{y} \in A^*$
  - ***n*-gram precision**: which fraction of predicted n-grams occur in the ground truth:

$$p_n(y, \hat{y}) := \frac{\mathbb{1}_A^T \min(f_n(\hat{y}), f_n(y))}{|\hat{y}| - n + 1}$$

BLEU score (2/3)

- compare two sentences/sequences $y, \hat{y} \in A^*$
  - ***n*-gram precision**: which fraction of predicted *n*-grams occur in the ground truth:

$$p_n(y, \hat{y}) := \frac{\mathbb{1}_A^T \min(f_n(\hat{y}), f_n(y))}{|\hat{y}| - n + 1}$$

$$\text{e.g., } p_1((\text{"hello","world"}), (\text{"hello", "hello"})) = \frac{1}{2}$$

$$\text{e.g., } p_1((\text{"hello","world", "how", "do", "you" "do"})$$

$$(\text{"hello", "world", "how", "do", "going"})) = \frac{4}{5}$$

$$\text{e.g., } p_2(\ldots) = \frac{3}{4}, \quad , p_3(\ldots) = \frac{2}{3}, \quad p_4(\ldots) = \frac{1}{2}$$

BLEU score (3/3)

- **brevity penalty** (as precision favors short predictions $\hat{y}$):

$$\text{brev}(y, \hat{y}) := e^{-\max(0, |y| - |\hat{y}|)\frac{1}{|\hat{y}|}}$$

- **BLEU score** (sometimes called BLEU-4):
  - geometric mean of the 1- to 4-gram precisions
  - times brevity penality:

$$\text{bleu}(y, \hat{y}) := \text{brev}(y, \hat{y})\,(p_1(y, \hat{y}) p_2(y, \hat{y}) p_3(y, \hat{y}) p_4(y, \hat{y}))^{\frac{1}{4}} \in [0, 1]$$

$$\begin{aligned}\text{e.g., }\; \text{bleu}((&\text{"hello","world", "how", "do", "you" "do"})\\ &(\text{"hello", "world", "how", "do", "going"}))\\ &= e^{-\frac{1}{5}}(\frac{4}{5} \cdot \frac{3}{4} \cdot \frac{2}{3} \cdot \frac{1}{2})^{\frac{1}{4}} \approx 0.819 \cdot 0.669 \approx 0.548\end{aligned}$$

  - to get a BLEU > 0, at least one 4-gram has to match.

The Transformer Network

- construct output sequence $\hat{y}$ token by token,
  starting from a sequence containing just a **start token** $\hat{y}_0$,
  with a network g (called **decoder**):

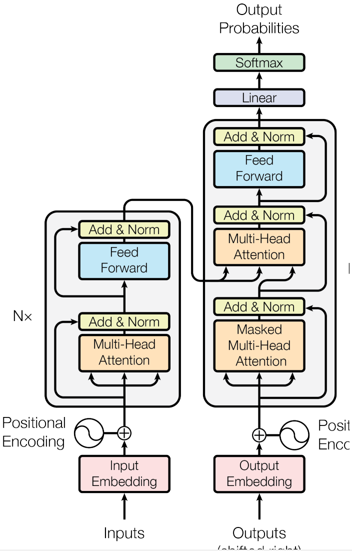
$$p(\hat{y}_1 \mid \emptyset) := g(\hat{y}_0)$$
$$p(\hat{y}_t \mid \hat{y}_{1:t-1}) := g(\hat{y}_{0:t-1})$$

- use input sequence x as additional input,
  encoded once with a network h (called **encoder**):

$$p(\hat{y}_1 \mid \emptyset, x) := g(\hat{y}_0, h(x))$$
$$p(\hat{y}_t \mid \hat{y}_{1:t-1}, x) := g(\hat{y}_{0:t-1}, h(x))$$

  - specifically the encoder is using 6 attenion layers
  - the decoder is using also 6 attention layers,
    - the output of the encoder provides keys and values,
    - the decoder provides the queries.

The Transformer Network / Architecture

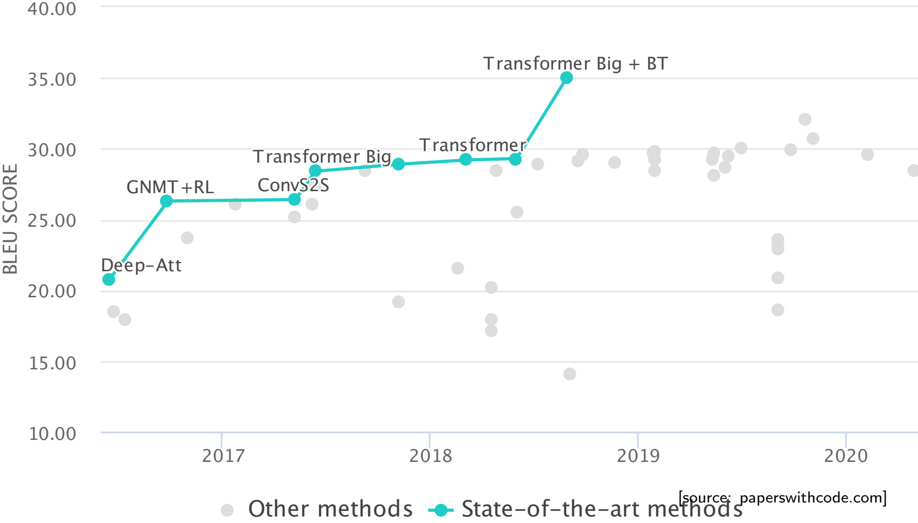


Evaluation on WMT-2014

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [15]	23.75			
Deep-Att + PosUnk [32]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [31]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [8]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [26]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [32]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [31]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [8]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	$\mathbf{3.3 \cdot 10^{18}}$	
Transformer (big)	**28.4**	**41.0**	$2.3 \cdot 10^{19}$	

[source: Vaswani et al. 2017]

Evaluation on WMT-2014 EN-DE / Leaderboard



[source: paperswithcode.com]

Summary (1/2)

- **Attention layers** model pairwise interactions between input/latent channels with a low-rank pairwise interaction weights matrix.
    - like **factorization models**
    - can be interpreted as **query–key–value** scheme
- A **multi-head attention layer** models attention between groups of input/latent channels.
- **Restricted attention layers** restrict attention in sequence dimension to a window.
- Compared to convolutions, attention layers
    - have fewer parameters: $3M^2$ vs. KM^2,
    - restricred attention layers allow to trade off # operations and **path length** through the window size K w/o increasing the number of parameters

      — # operations: TKM, path length: T/K.

Summary (2/2)

- The **transformer network** is a sequence to sequence model that
  - models the next output element as function of
    - the output sequence so far (**decoder**) and
    - the input sequence (**encoder**).
  - using attentions between
    - inputs (encoder) and
    - inputs and previous outputs (decoder)
- For **machine translation**, the transformer network is one of the most accurate models.

Further Readings

- Zhang et al. 2020, ch. 10
- Jay Alammar (2020), *Visualizing A Neural Machine Translation Model (Mechanics of Seq2seq Models With Attention* and *The Illustrated Transformer*, blog post, `https://jalammar.github.io/visualizing-neural-machine-translation-mechanics-of-seq2s`
- The attention mechanism has been introduced into the deep learning literature by Bahdanau et al. 2014, there still on top of recurrent layers.
- The described attention layer is the one of Vaswani et al. 2017.
- Factorization machines: Rendle 2010 — Matrix factorization: Srebro et al. 2005, Mnih and Salakhutdinov 2008.
- Attention mechanism are not covered by Goodfellow et al. 2016.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv:1409.0473 [cs, stat]*, September 2014.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. The Mit Press, Cambridge, Massachusetts, November 2016. ISBN 978-0-262-03561-3.

Andriy Mnih and Russ R. Salakhutdinov. Probabilistic matrix factorization. In *Advances in Neural Information Processing Systems*, pages 1257–1264, 2008.

S. Rendle. Factorization machines. In *Data Mining (ICDM), 2010 IEEE 10th International Conference On*, pages 995–1000, 2010.

N. Srebro, J. D.M Rennie, and T. S Jaakkola. Maximum-margin matrix factorization. *Advances in neural information processing systems*, 17:1329–1336, 2005.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., 2017.

Aston Zhang, Zachary C. Lipton, Mu Li, and Alexander Smola. *Dive into Deep Learning*. https://d2l.ai/, 2020.