On the Relationship between Self-Attention and Convolutional Layers

Jean-Baptiste Cordonnier, Andreas Loukas, Martin Jaggi

Presented by Yenson Lau — Layer 6 AI

August 11th, 2021

Today's talk

- Attention can simultaneously attend to every word in an input sequence
- NN architectures using self-attention (SA) only (without convolution) can compete with SA + convolutional architectures on vision tasks
- Do SA layers process images in a similar manner to convolutional layers?

Contributions of the paper

- A constructive theoretical proof that SA can express convolutional layers using relative positional encoding:

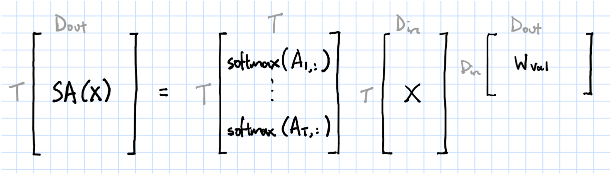
> **Main theorem.** *A* multi-head self-attention *(MHSA) layer with N_h heads of dimension D_h each, final output dimension D_{out}, and a* relative positional encoding *of dimension $D_p \geq 3$ can express any 2D convolutional layer of kernel size $\sqrt{N_h} \times \sqrt{N_h}$ and $\min(D_h, D_{out})$ output channels.*

- Experiments showing that the first few layers of DNNs using SA do behave similarly to the theoretical construction.

Self-Attention

- Let $\boldsymbol{X} \in \mathbb{R}^{T \times D_{in}}$ be an input matrix consisting of T tokens in D_{in} dimensions. A self-attention layer $\mathrm{SA} : D_{in} \to D_{out}$ is expressed as

$$\mathrm{SA}(\boldsymbol{X})_{t,:} \doteq \mathrm{softmax}(\boldsymbol{A}_{t,:})\boldsymbol{X}\boldsymbol{W}_{val}, \tag{1}$$



- We refer to the the elements of the $T \times T$ matrix

$$\boldsymbol{A} \doteq \boldsymbol{X}\boldsymbol{W}_{qry}\boldsymbol{W}_{key}^{T}\boldsymbol{X}^{T} \tag{2}$$

the *attention scores*, and the softmax outputs as *attention probabilities*.

Self-Attention

- So far the learnable parameters are the *query*, *key*, and *value* matrices

$$\boldsymbol{W}_{qry} \in \mathbb{R}^{D_{in} \times D_k}, \quad \boldsymbol{W}_{key} \in \mathbb{R}^{D_{in} \times D_k}, \quad \text{and} \quad \boldsymbol{W}_{val} \in \mathbb{R}^{D_{in} \times D_{out}}.$$

  For simplicity, ignore residual connections, normalization or constant factors.

- Note that $\boldsymbol{A}$ is *permutation equivariant* – shuffling the order of the tokens (rows) in $\boldsymbol{X}$ shuffles the token scores in $\boldsymbol{A}$. *This is not desired behavior.*
  - e.g. "the cat ate the fish" is very different from "the fish ate the cat"

- To overcome this, we can add a (fixed or learned) *positional encoding* $\boldsymbol{P}_{t,:}$, for each token in the sequence, to the input matrix for computing attention scores

$$A \doteq (\boldsymbol{X} + \boldsymbol{P})\boldsymbol{W}_{qry}\boldsymbol{W}_{key}^T(\boldsymbol{X} + \boldsymbol{P})^T, \tag{3}$$

  where the encoding matrix $\boldsymbol{P}$ has size $T \times D_{in}$.

- In *absolute encoding*, a fixed or learned vector $\boldsymbol{P}_{t,:}$ is assigned to each token t. Therefore, for a query token q (the token to compute the representation of) and key token k (the pixel we are attending to),

$$\begin{aligned} \boldsymbol{A}^{abs}_{q,k} &= (\boldsymbol{X}_{q,:} + \boldsymbol{P}_{q,:})\boldsymbol{W}_{qry}\boldsymbol{W}^{T}_{key}(\boldsymbol{X}_{q,:} + \boldsymbol{P}_{q,:})^{T} \\ &= \boldsymbol{X}_{q,:}\boldsymbol{W}_{qry}\boldsymbol{W}^{T}_{key}\boldsymbol{X}^{T}_{k,:} + \boldsymbol{X}_{q,:}\boldsymbol{W}_{qry}\boldsymbol{W}^{T}_{key}\boldsymbol{P}^{T}_{k,:} \\ &\quad + \boldsymbol{P}_{q,:}\boldsymbol{W}_{qry}\boldsymbol{W}^{T}_{key}\boldsymbol{X}^{T}_{k,:} + \boldsymbol{P}_{q,:}\boldsymbol{W}_{qry}\boldsymbol{W}^{T}_{key}\boldsymbol{P}^{T}_{k,:} \end{aligned} \tag{4}$$

- *Relative positional encoding* instead considers the positional difference between the query and key tokens:

$$\boldsymbol{A}^{rel}_{q,k} = \boldsymbol{X}^{T}_{q,:}\boldsymbol{W}_{qry}\boldsymbol{W}^{T}_{key}\boldsymbol{X}_{k,:} + \boldsymbol{X}^{T}_{q,:}\boldsymbol{W}^{T}_{qry}\hat{\boldsymbol{W}}_{key}\boldsymbol{r}_{\delta} + \boldsymbol{u}^{T}\boldsymbol{W}_{key}\boldsymbol{X}^{T}_{k,:} + \boldsymbol{v}^{T}\hat{\boldsymbol{W}}_{key}\boldsymbol{r}_{\delta}. \tag{5}$$

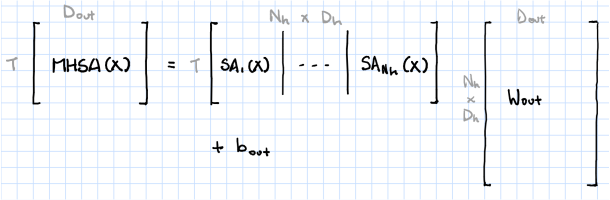
  - The learnable vectors $\boldsymbol{u}$, $\boldsymbol{v}$ are unique for each head
  - The relative positional encoding $\boldsymbol{r}_{\delta} \in \mathbb{R}^{D_p}$ depends only on the shift $\delta \doteq k - q$, and is shared by all layers and heads.
  - Key weights are split into $\boldsymbol{W}_{key}$ for the input and $\hat{\boldsymbol{W}}_{key}$ for positional encoding.

- Note the attention scores are now *shift equivariant* rather than permutation equivariant. This is the desired behavior for convolutional / vision tasks.

- In practice it is beneficial to replicate the SA mechanism into multiple heads, by concatenating the output of N_h heads of output dimension D_h and projecting it to dimension D_{out}.

$$\mathrm{MHSA}(\boldsymbol{X}) \doteq \mathrm{hcat}_{h\in[N_h]}\left[\mathrm{SA}_h(\boldsymbol{X})\right] \boldsymbol{W}_{out} + \boldsymbol{b}_{out}. \tag{6}$$

Here $\boldsymbol{W}_{out} \in \mathbb{R}^{N_h D_h \times D_{out}}$ is the projection matrix and $\boldsymbol{b}_{out} \in \mathbb{R}^{D_{out}}$ is a bias term.



Self-Attention for Images

From 1D sequence to 2D

- Replace query and key tokens with pixels $\boldsymbol{q}, \boldsymbol{k} \in [W] \times [H]$, and the input with $\boldsymbol{X} \in \mathbb{R}^{W \times H \times D}$. Each attention score now associates a query and key pixel.
- For a pixel $\boldsymbol{p} \in (i, j)$, $\boldsymbol{X}_{\boldsymbol{p},:} \doteq \boldsymbol{X}_{i,j,:}$ and $\boldsymbol{A}_{\boldsymbol{p},:} \doteq \boldsymbol{A}_{i,j,:,:}$.
- Then, analogously to the 1D case,

$$\mathrm{SA}(\boldsymbol{A})_{\boldsymbol{q},:} = \sum_{\boldsymbol{k}} \mathrm{softmax}(\boldsymbol{A}_{\boldsymbol{q},:})_{\boldsymbol{k}} \, \boldsymbol{X}_{\boldsymbol{k},:} \, \boldsymbol{W}_{val}, \tag{7}$$

  and relative positional encoding retains the same form as Equation (5). Similarly MHSA retains the same form as Equation (6).

- Given an image tensor $\boldsymbol{X} \in \mathbb{R}^{W \times H \times D_{in}}$ and kernel tensor $\boldsymbol{W} \in \mathbb{R}^{K \times K \times D_{in} \times D_{out}}$,

$$\mathrm{Conv}(\boldsymbol{X})_{i,j,:} \doteq \sum_{(\delta_1,\delta_2)\in\Delta_K} \boldsymbol{X}_{i-\delta_1,j-\delta_2,:}\boldsymbol{W}_{\delta_1,\delta_2,:,:} + \boldsymbol{b} \in \mathbb{R}^{D_{out}}, \tag{8}$$

  where

$$\Delta_K \doteq \left[-\left\lfloor\frac{K}{2}\right\rfloor,\dots,\left\lfloor\frac{K}{2}\right\rfloor\right] \times \left[-\left\lfloor\frac{K}{2}\right\rfloor,\dots,\left\lfloor\frac{K}{2}\right\rfloor\right]$$

  is the set of all shifts represented by a $K \times K$ kernel.

- We depart from the original notation slightly by using the *unflipped* convolution. In the paper, the summand is flipped to $\boldsymbol{X}_{i+\delta_1,i+\delta_2,:}\boldsymbol{W}_{\delta_1,\delta_2,:,:}$. The theorem in the paper is not changed by this.

- **Main theorem.** *A* multi-head self-attention *(MHSA) layer with N_h heads of dimension D_h each, final output dimension D_{out}, and a* relative positional encoding *of dimension $D_p \geq 3$ can express any 2D convolutional layer of kernel size $\sqrt{N_h} \times \sqrt{N_h}$ and* $\min(D_h, D_{out})$ *output channels.*

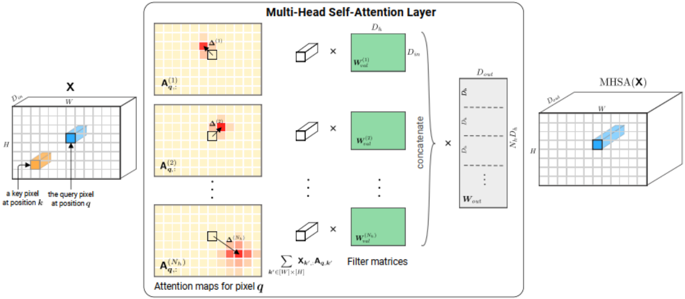


Figure 1: Illustration of a Multi-Head Self-Attention layer applied to a tensor image $\mathbf{X}$. Each head h attends pixel values around shift $\mathbf{\Delta}^{(h)}$ and learn a filter matrix $\boldsymbol{W}_{val}^{(h)}$. We show attention maps computed for a query pixel at position $\boldsymbol{q}$.

- The theorem is a consequence of two lemmas.

- **Lemma 1.** *Consider a MHSA layer consisting of $N_h = K^2$ heads, $D_h \geq D_{out}$ and let $\boldsymbol{f} : [N_h] \to \Delta_K$ be a bijective mapping of heads onto shifts. Further, suppose that for every head*

$$\operatorname{softmax}(\boldsymbol{A}^{(h)}_{\boldsymbol{q},:})_{\boldsymbol{k}} = \begin{cases} 1 & \text{if } \boldsymbol{f}(h) = \boldsymbol{q} - \boldsymbol{k}, \\ 0 & \text{otherwise.} \end{cases} \tag{9}$$

Then for any convolutional layer with a $K \times K$ kernel and D_{out} output channels, there exists $\{\boldsymbol{W}^{(h)}_{val}\}_{h \in [N_h]}$ such that $\operatorname{MHSA}(\boldsymbol{X}) = \operatorname{Conv}(\boldsymbol{X})$, for any kernel tensor $\boldsymbol{W} \in \mathbb{R}^{K \times K \times D_{in} \times D_{out}}$, and for all $\boldsymbol{X} \in \mathbb{R}^{W \times H \times D_{in}}$.

- *Proof of Lemma 1.* Rewrite Equation (6) as

$$\operatorname{MHSA}(\boldsymbol{X}) = \boldsymbol{b}_{out} + \sum_{h \in [N_h]} \operatorname{softmax}(\boldsymbol{A}^{(h)})\ \boldsymbol{X}\ \underbrace{\boldsymbol{W}^{(h)}_{val}\ \boldsymbol{W}^{(h)}_{out}}_{\boldsymbol{W}^{(h)}}\ . \tag{10}$$

Here $\boldsymbol{W}^{(h)}_{val} \in \mathbb{R}^{D_{in} \times D_h}$ is the value matrix for head h, or in MATLAB notation,

$$\boldsymbol{W}^{(h)}_{out} = \boldsymbol{W}[(h-1) * D_h + 1 : hD_h\ ,\ :\ ] \quad \in \quad \mathbb{R}^{D_h \times D_{out}}.$$

- *Proof of Lemma 1 (cont).* Consider a single "query" pixel from $\mathrm{MHSA}(\boldsymbol{X})$:

$$\mathrm{MHSA}(\boldsymbol{X})_{\boldsymbol{q},:} = \boldsymbol{b}_{out} + \sum_{h\in[N_h]} \left(\sum_{\boldsymbol{k}\in[T]} \mathrm{softmax}(\boldsymbol{A}^{(h)}_{\boldsymbol{q},:})_{\boldsymbol{k}} \ \boldsymbol{X}_{\boldsymbol{k},:} \right) \boldsymbol{W}^{(h)}. \tag{11}$$

  The summand $\sum_{\boldsymbol{k}} \mathrm{softmax}(\boldsymbol{A}^{(h)}_{\boldsymbol{q},:})_{\boldsymbol{k}} \ \boldsymbol{X}_{\boldsymbol{k},:}$ is a weighted average on the rows of $\boldsymbol{X}$.

- Applying the assumption from Equation (9), each of the softmax weights pick out a single row $\boldsymbol{k} = \boldsymbol{q} - \boldsymbol{f}(h)$:

$$\mathrm{MHSA}(\boldsymbol{X})_{\boldsymbol{q},:} = \boldsymbol{b}_{out} + \sum_{h\in[N_h]} \boldsymbol{X}_{\boldsymbol{q}-\boldsymbol{f}(h),:} \boldsymbol{W}^{(h)}. \tag{12}$$

  If we set $K = \sqrt{N_h}$, then it is clearly possible to index all the shifts using $h \in [N_h]$, i.e. $\boldsymbol{f}([N_h]) = \Delta_K$. Therefore we can rewrite the expression as

$$\mathrm{MHSA}(\boldsymbol{X})_{\boldsymbol{q},:} = \boldsymbol{b}_{out} + \sum_{h\in[N_h]} \boldsymbol{X}_{\boldsymbol{q}-\boldsymbol{f}(h),:} \boldsymbol{W}_{\boldsymbol{f}(h),:,:} = \mathrm{Conv}(\boldsymbol{X})_{\boldsymbol{q},:}. \qquad \square \tag{13}$$

- Lemma 1 says that, with the right choice of relative positional encoding (and a very restricted set of attention parameters), *a MHSA layer is exactly equivalent to a convolutional layer*.
- The number of linearly independent filters expressible by $\boldsymbol{W}$ is clearly limited by $\min(D_h, D_{out})$. Therefore, to express D_h convolutional filters, it is best to simply let $D_{out} = N_h D_h$ and have $\boldsymbol{W}^{(h)} \in \mathbb{R}^{D_h \times D_h}$.
- There is a generalized version of this lemma in the Appendix of the paper that delves into the space of filters spannable for a given set of dimensions D_h, N_h, D_{out}, etc., for interested readers.

- **Lemma 2.** *There exists a relative encoding scheme* $\{\boldsymbol{r}_\delta \in \mathbb{R}^{D_p}\}_{\boldsymbol{\delta}\in\mathbb{Z}^2}$ *with* $D_p \geq 3$ *and parameters* $\boldsymbol{W}_{qry}, \boldsymbol{W}_{key}, \hat{\boldsymbol{X}}_{key}, \boldsymbol{u}$ *with* $D_p \leq D_k$ *such that, for every* $\boldsymbol{\Delta} \in \Delta_K$ *there exists some vector* $\boldsymbol{v}(\boldsymbol{\Delta})$ *that yields* $\mathrm{softmax}(\boldsymbol{A}_{\boldsymbol{q},:})_{\boldsymbol{k}} = 1$ *if* $\boldsymbol{k} - \boldsymbol{q} = \boldsymbol{\Delta}$, *and zero otherwise.*

- *Proof of Lemma 2.* Start with the relative positional encoding of Equation (5):

  $$\boldsymbol{A}^{rel}_{\boldsymbol{q},\boldsymbol{k}} = \boldsymbol{X}^T_{\boldsymbol{q},:} \boldsymbol{W}_{qry} \boldsymbol{W}^T_{key} \boldsymbol{X}_{k,:} + \boldsymbol{X}^T_{\boldsymbol{q},:} \boldsymbol{W}^T_{qry} \hat{\boldsymbol{W}}_{key} \boldsymbol{r}_\delta + \boldsymbol{u}^T \boldsymbol{W}_{key} \boldsymbol{X}^T_{k,:} + \boldsymbol{v}^T \hat{\boldsymbol{W}}_{key} \boldsymbol{r}_\delta .$$

  Since the desired attention probabilities (9) from Lemma 1 are independent of the input $\boldsymbol{X}$, set $\boldsymbol{W}_{key} = \boldsymbol{W}_{qry} = 0$.

  This leaves only the final term. Setting $\hat{\boldsymbol{W}}_{key} = [\ \boldsymbol{I}_{D_p}\ ;\ 0\ ] \in \mathbb{R}^{D_k \times D_p}$ (recall $D_p \leq D_k$) yields

  $$\boldsymbol{A}_{\boldsymbol{q},\boldsymbol{k}} = \boldsymbol{v}^T_{1:D_p} \boldsymbol{r}_\delta .$$

- *Proof of Lemma 2 (cont).* Recall that $\boldsymbol{\delta} \doteq \boldsymbol{k} - \boldsymbol{q}$. Now suppose some $\boldsymbol{v}, \boldsymbol{r}_{\boldsymbol{\delta}}$ could be found such that

$$\boldsymbol{A}_{\boldsymbol{q},\boldsymbol{k}} = -\alpha(\|\boldsymbol{\delta} - \boldsymbol{\Delta}\|^2 + c) \tag{14}$$

so that the maximum score $-\alpha c$ is achieved when $\boldsymbol{\delta} = \boldsymbol{\Delta}$, then

$$\begin{aligned} \lim_{\alpha\to\infty} \operatorname{softmax}(\boldsymbol{A}_{\boldsymbol{q},:})_{\boldsymbol{k}} &= \lim_{\alpha\to\infty} \frac{e^{-\alpha(\|\boldsymbol{\delta}-\boldsymbol{\Delta}\|^2+c)}}{\sum_{\boldsymbol{\delta}'\in\Delta_K} e^{-\alpha(\|\boldsymbol{\delta}'-\boldsymbol{\Delta}\|^2+c)}} \\ &= \frac{1_{\boldsymbol{\delta}=\boldsymbol{\Delta}}}{1 + \lim_{\alpha\to\infty} \sum_{\boldsymbol{\delta}'\neq\boldsymbol{\delta}} e^{-\alpha\|\boldsymbol{\delta}'-\boldsymbol{\Delta}\|^2}} = 1_{\boldsymbol{\delta}=\boldsymbol{\Delta}}. \end{aligned}$$

- Choosing $\boldsymbol{v}_{1:D_p} = -\alpha(1, -2\Delta_1, -2\Delta_2)$ and $\boldsymbol{r}_{\boldsymbol{\delta}} = (\|\boldsymbol{\delta}\|^2, \delta_1, \delta_2)$, yields

$$\begin{aligned} \boldsymbol{A}_{\boldsymbol{q},\boldsymbol{k}} &= \boldsymbol{v}_{1:D_p}^T \boldsymbol{r}_{\boldsymbol{\delta}} \\ &= -\alpha(\|\boldsymbol{\delta}\|^2 - 2\Delta_1\delta_1 - 2\Delta_2\delta_2) \\ &= -\alpha(\|\boldsymbol{\delta} - \boldsymbol{\Delta}\|^2 - \|\boldsymbol{\Delta}\|^2). \end{aligned}$$

Setting $c = -\|\boldsymbol{\Delta}\|^2$ recovers Equation (14) and completes the proof. □

- The encoding scheme satisfying Lemma 2 is a *quadratic encoding scheme*, and is achieved by setting

$$
\begin{aligned}
\boldsymbol{v}^{(h)} &\doteq -\alpha^{(h)}(1, -2\Delta_1^{(h)}, -2\Delta_2^{(h)}), \\
\boldsymbol{r}_{\boldsymbol{\delta}} &\doteq (\|\boldsymbol{\delta}\|^2,\ \delta_1,\ \delta_2), \\
\boldsymbol{W}_{qry} = \boldsymbol{W}_{key} &\doteq 0, \\
\hat{\boldsymbol{W}}_{key} &\doteq \boldsymbol{I}.
\end{aligned}
\tag{15}
$$

  Here the learned parameters $\boldsymbol{\Delta}^{(h)} = (\Delta_1^{(h)}, \Delta_2^{(h)})$ and $\alpha^{(h)}$ determine the center and width of attention of each head, and $\boldsymbol{\delta} = (\delta_1, \delta_2)$ is fixed and expresses the relative shift between query and key pixels.

- Although the proof requires $\alpha \to \infty$ to satisfy the assumption (9) of Lemma 1, finite precision arithmetic performs hard attention with sufficiently large enough α. E.g. for `Float32`, set $\alpha \geq 46$.

- The lemma, and thus the theorem, can be extended in a straightforward manner to cover K-dimensional convolutions, with $D_p = K + 1$.

- **Baseline.** Standard ResNet18 on CIFAR-10
- **Model.**
  - 6 MHSA layers. Each layer consists of a MHSA step, followed by dense, dropout and LayerNorm sublayers.
  - Use 2x2 invertible downsampling to reduce image size (attention coefficients scale quadratically to image size)
  - Fixed size representation of input image is the average pooling of the last layer representations, and fed to a linear classifier
- **MHSA variations.** Different types of relative positional encoding

$$\boldsymbol{A}^{rel}_{\boldsymbol{q},\boldsymbol{k}} = \boldsymbol{X}^{T}_{\boldsymbol{q},:} \boldsymbol{W}_{qry} \boldsymbol{W}^{T}_{key} \boldsymbol{X}_{k,:} + \boldsymbol{X}^{T}_{\boldsymbol{q},:} \boldsymbol{W}^{T}_{qry} \hat{\boldsymbol{W}}_{key} \boldsymbol{r}_{\boldsymbol{\delta}} + \boldsymbol{u}^{T} \boldsymbol{W}_{key} \boldsymbol{X}^{T}_{k,:} + \boldsymbol{v}^{T} \hat{\boldsymbol{W}}_{key} \boldsymbol{r}_{\boldsymbol{\delta}}.$$

  - **SA with quadratic embedding:** Retain final term only and fix the variables using Equation (15). The attention widths $\alpha^{(h)}$ and centers $\Delta^{(h)}$ are still learnt.
  - **SA with learned embedding:** Retain final term only but learn v, $\hat{\mathrm{W}}_{key}$, r_{δ}, with $D_p = D_{out} = 400$. Set $D_h = D_{out}$.
  - **SA with content-based attention:** All terms retained (might actually be just first two terms) and all variables learnable. Same dimensions as above.

- **ResNet converges faster:** Probably because SA's inductive bias is not as strong as ResNet, but may also be due to different optimization setup.
- SA models with more learnable parameters converge slower and ends up with lower testing accuracy.

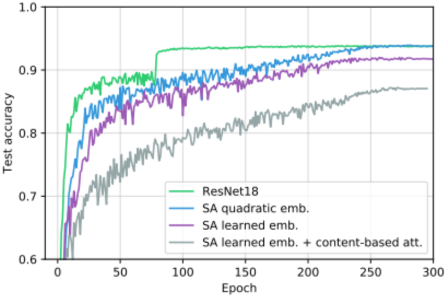


Figure 2: Test accuracy on CIFAR-10.

Experiments

Do self-attention layers actually learn convolutions? Maybe.

- **ResNet converges faster:** Probably because SA's inductive bias is not as strong as ResNet, but may also be due to different optimization setup.
- SA models with more learnable parameters converge slower and ends up with lower testing accuracy.

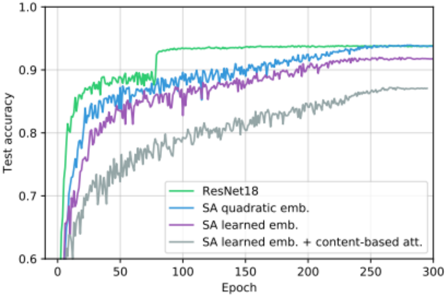


Figure 2: Test accuracy on CIFAR-10.

Thanks for listening!