Network Compression (2)

Aaron Zhao, Imperial College London

Introduction

Neural Network Compression

- Winograd Transformed Convolution
- Low-rank Approximation
- Chaining Compression Algorithms
- Knowledge Distillation

Winograd Transformation

Winograd transformation

Very simple initiative:

Let's replace expensive operations (multiplication) with cheap ones (addition).

Let's replace the following

$$y = x \times x$$

With this one

$$y = x + x$$

Winograd: an illustration

$$F(2,3) = \begin{bmatrix} d_0 & d_1 & d_2 \\ d_3 & d_4 & d_5 \end{bmatrix} \begin{bmatrix} g_0 \\ g_1 \\ g_2 \end{bmatrix}$$

$F(2,3)$ has in total $2 \times 3 = 6$ multiplications

$$F(2,3) = \begin{bmatrix} d_0 & d_1 & d_2 \\ d_3 & d_4 & d_5 \end{bmatrix} \begin{bmatrix} g_0 \\ g_1 \\ g_2 \end{bmatrix} = \begin{bmatrix} m_1 + m_2 + m_3 \\ m_2 - m_3 - m_4 \end{bmatrix}$$

$m_1 = (d_0 - d_2) g_0$

$m_2 = (d_1 + d_2) \frac{g_0 + g_1 + g_2}{2}$

$m_3 = (d_2 - d_1) \frac{g_0 - g_1 + g_2}{2}$

$m_4 = (d_1 - d_3) g_2$

Now we have in total 4 multiplications but a bit more additions

Winograd: an illustration (ii)

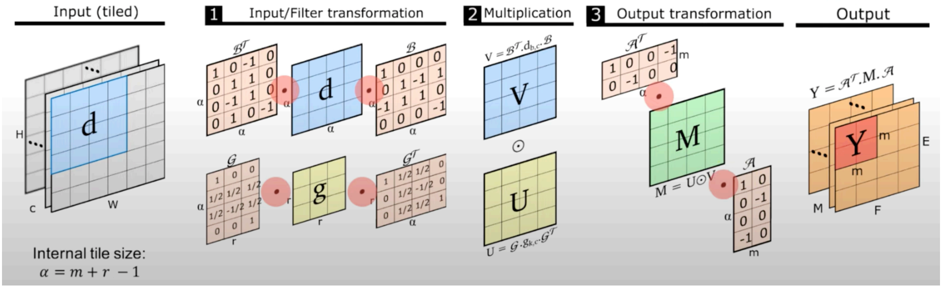


Figure 1:

Rewrite it in matrix form

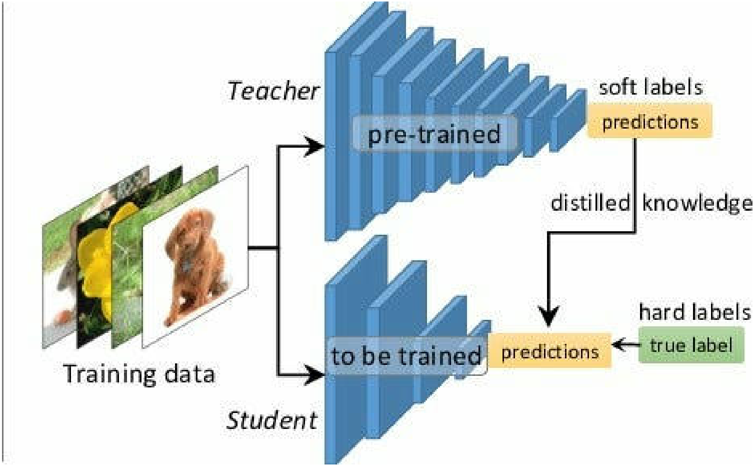
$$Y = A^T \left[\left(GgG^T \right) \cdot \left(B^T dB \right) \right] A$$

Basically, g and d would be transformed values to the winograd space to do a point-wise multiplication.

Knowledge Distillation

The Teacher-student Paradigm

The general idea is to have a capable model (teacher) and use its output to 'teach' a smaller student model.



Soft-labels

We construct a soft label, this is normally the final layer of the embedding from the larger model:

$$y_{soft} = f_w(X)$$

Then we use this soft label to form a loss function with the output of the small model (student).

$$\mathcal{L}_{kd} = g\big(y_{soft}, f'_{w'}(X)\big)$$

where $f'_{w'}(X)$ is the output of the student model.

Normally, this is actually added to the actual loss as an additional regularization term:

$$\mathcal{L} = \mathcal{L}_{ce} + \mathcal{L}_{kd}$$

Logits-based KD

$$\mathcal{L}_{kd} = g\big(y_{soft}, f'_{w'}(X)\big)$$

where $f'_{w'}(X)$ is the output of the student model.

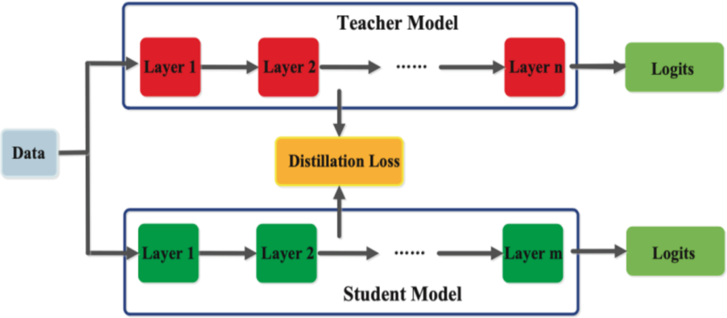
Function g is commonly l_p norms, but can also be other functions.

Activation-based KD

Activation-based KD forms an additional regularization loss through all (or a subset of) activation values to form the loss.

$$\mathcal{L}_{kd} = \sum_{i=1}^{L} g(y_{act}^{i}, f_{act}^{i}(X))$$

where L is the total number of layers.



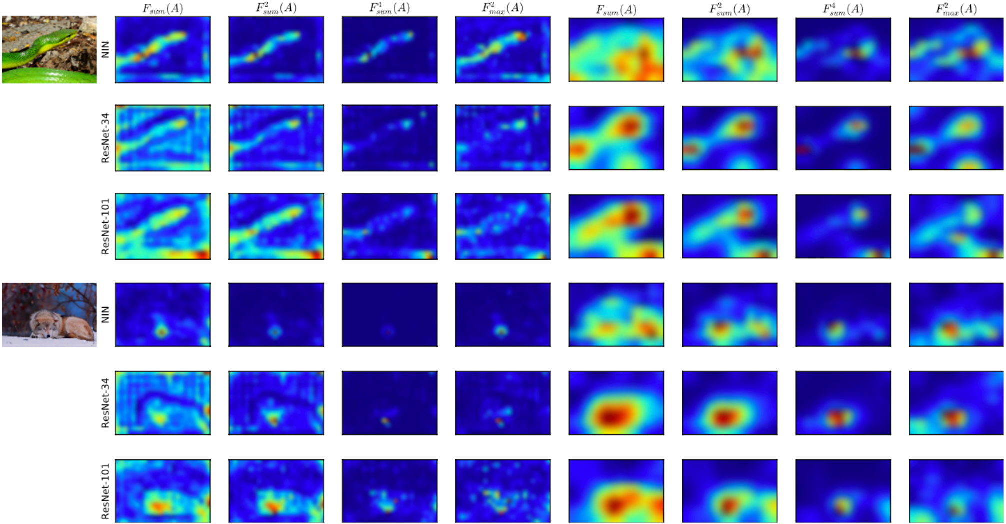
Attention-map for activation-based KD

As we have explained, the design of the regularization loss g can be fairly complex. Let's consider a layer's activation functions $x_{act} \in \mathcal{R}^{C \times H \times W}$.

We define an attention map as a function $p : \mathcal{R}^{C \times H \times W} \to \mathcal{R}^{H \times W}$. For instance, this can be:

- Sum of absolute values: $p(x) = \sum_{i=1}^{C} |x_i|$
- Sum of absolute values with power m: $p(x) = \sum_{i=1}^{C} |x_i|^m$

Attention-map for activation-based KD (ii)



Attention-map for activation-based KD (iii)

Performant networks tend to have similar attention maps

$$\mathcal{L} = \mathcal{L}_{CE} + \frac{\beta}{2} \sum_{i=1}^{L} \left\| \frac{Q_S^i}{\| Q_S^i \|_2} - \frac{Q_T^i}{\| Q_T^i \|_2} \right\|_p$$

where Q_S^i and Q_T^i are vectorized version of activation values for the student and the teacher networks respectively.

The whole $\frac{\beta}{2} \sum_{i=1}^{L} \left\| \frac{Q_S^i}{\| Q_S^i \|_2} - \frac{Q_T^i}{\| Q_T^i \|_2} \right\|_p$ part is an additional loss term that encourages the student network to have the same ‘attention map’ as the teacher network.

Chaining Compression Algorithms

Chaining compression techniques

In fact, many compression techniques are working on fairly orthogonal spaces. A great idea to harvesting more compression rate is chaining a bunch of them together.

Multiplying gains when you chain a bunch of compression algorithms together.

The following example is what I have tried to run fine-grained pruning and fixed-point quantizations on a CIFAR10 network. The CIFAR10 network is a variant of VGG.

Chaining compression techniques (ii)

Method	Bitwidth	Density	Compression rate	top-1/top-5 accuracy
baseline	32	100.00%	-	91.37%/99.67%
fixed-point (fixed)	4	100.00%	8.00×	89.64%/99.74%
dynamic fixed-point (DFP)	4	100.00%	8.00×	90.63%/99.68%
fine-grained pruning (pruned)	32	15.65%	6.39×	91.12%/99.70%
pruned + fixed	6	15.65%	33.92×	90.59%/99.68%
pruned + DFP	6	15.65%	33.92×	91.04%/99.70%

Chaining compression techniques (iii)

The best quantization method pushes the compression rate to $8\times$ with less than 1% loss in accuracy.

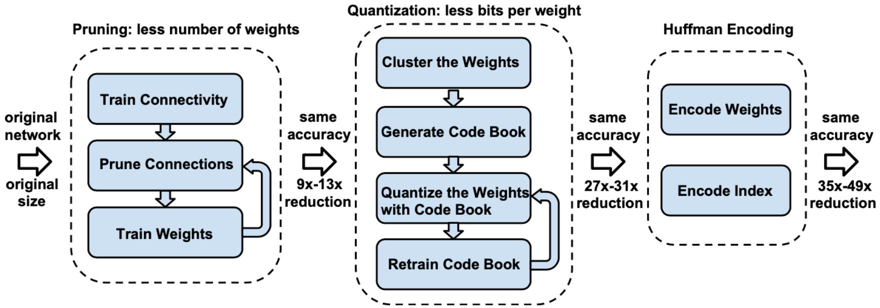
Adding the pruning optimization on top of the quantization, we push the compression rate to $33.92\times$ with a even better accuracy.

Why?: sometimes pruning serves as a regularization so you might have a better accuracy!

Lossy and Lossless Compressions

Distinguish between lossy and lossless compressions

- Quantization – lossy compression
- Pruning – lossy compression
- Decoding – lossless compression



Compression and Re-training

- Quantization – Lossy Compression
- Pruning – Lossy Compression
- Huffman Decoding – Lossless Compression

Re-training becomes a critical operation to increase the performance degradation from lossy compression.

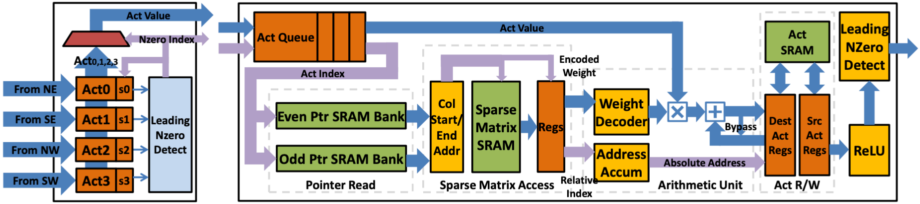
However, lossless compressions do not cause any performance degradation and thus has no need to use re-training.

Chaining these optimizations gives multiplying gains! Many of the compressions are (partially) orthogonal.

Special hardware support

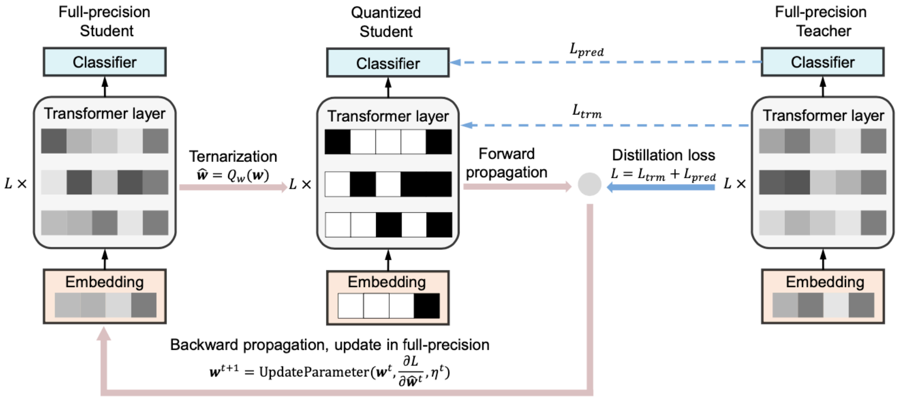
This however normally means you need special hardware support!

- Sparse Matrix Multiplication supoort
- Decoder and encoder support for Huffman decoding
- Low-precision multiplication units



Quantization with KD

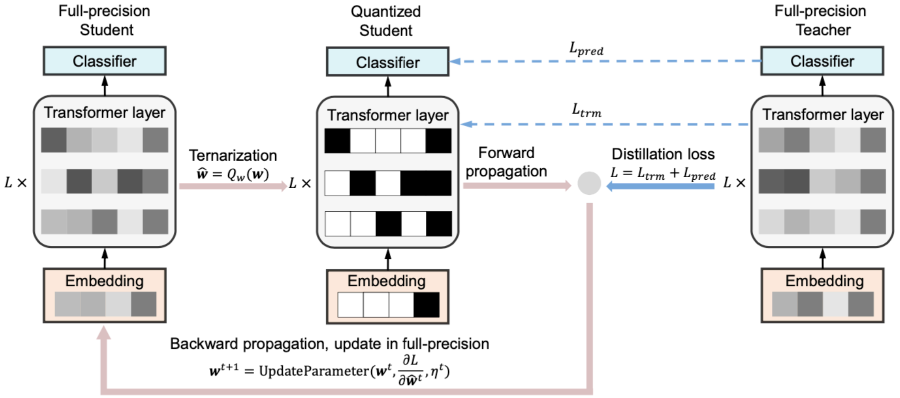
KD, as a training framework, can also be used in conjunction with compression algorithms.



Quantisation with KD

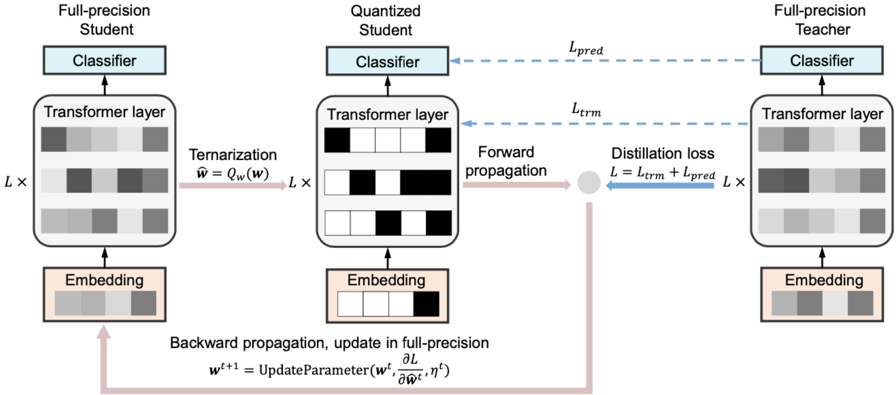
Teacher network: full-precision network

Student network: low-precision network



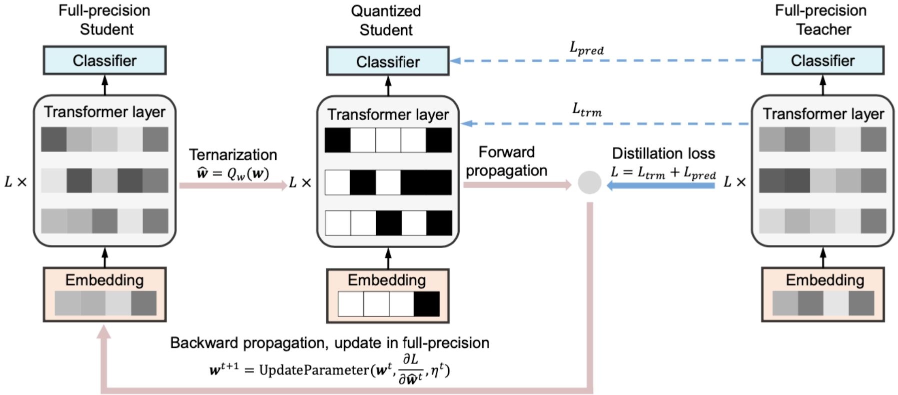
Quantisation with KD (ii)

The first distillation loss: knowledge in the embedding layer and the outputs of all Transformer layers of the full-precision teacher model to the quantized student model, by the mean squared error (MSE).



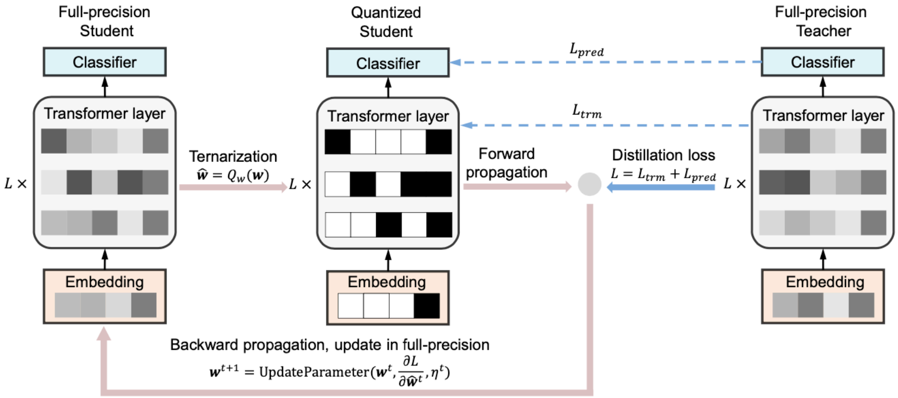
Quantisation with KD (iii)

The second distillation loss: the loss between teacher model's attention scores from all attention heads in all layers and the student model's attention scores.



Quantisation with KD (iv)

The third distillation loss: the loss between teacher model's logits and the student model's logits.



Quantisation with KD (v)

TinyBERT generally has a better accuracy given the same quantization budget.

		W-E-A (#bits)	Size (MB)	MNLI-m/mm	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE
	BERT	32-32-32	418 (×1)	84.5/84.9	87.5/90.9	92.0	93.1	58.1	89.8/89.4	90.6/86.5	71.1
	TinyBERT	32-32-32	258 (×1.6)	84.5/84.5	88.0/91.1	91.1	93.0	54.1	89.8/89.6	91.0/87.3	71.8
2-bit	Q-BERT	2-8-8	43 (×9.7)	76.6/77.0	-	-	84.6	-	-	-	-
	Q2BERT	2-8-8	43 (×9.7)	47.2/47.3	67.0/75.9	61.3	80.6	0	4.4/4.7	81.2/68.4	52.7
	TernaryBERT$_{\text{TWN}}$ (ours)	2-2-8	28 (×14.9)	83.3/83.3	86.7/90.1	91.1	92.8	55.7	87.9/87.7	91.2/87.5	72.9
	TernaryBERT$_{\text{LAT}}$ (ours)	2-2-8	28 (×14.9)	83.5/83.4	86.6/90.1	91.5	92.5	54.3	87.9/87.6	91.1/87.0	72.2
	TernaryTinyBERT$_{\text{TWN}}$ (ours)	2-2-8	18 (×23.2)	83.4/83.8	87.2/90.5	89.9	93.0	53.0	86.9/86.5	91.5/88.0	71.8
8-bit	Q-BERT	8-8-8	106 (×3.9)	83.9/83.8	-	-	92.9	-	-	-	-
	Q8BERT	8-8-8	106 (×3.9)	-/-	88.0/-	90.6	92.2	58.5	89.0/-	89.6/-	68.8
	8-bit BERT (ours)	8-8-8	106 (×3.9)	84.2/84.7	87.1/90.5	91.8	93.7	60.6	89.7/89.3	90.8/87.3	71.8
	8-bit TinyBERT (ours)	8-8-8	65 (×6.4)	84.4/84.6	87.9/91.0	91.0	93.3	54.7	90.0/89.4	91.2/87.5	72.2