Attention Distillation for Detection Transformers: Application to Real-Time Video Object Detection in Ultrasound

Jonathan Rubin1, Ramon Erkamp1, Ragha Srinivasa Naidu1, Anumod Odungatta ThodiyilF, Alvin Chen1
1Philips Research North America, Cambridge MA, United States
2Philips Innovation Campus, Bangalore, India

Introduction
- We introduce a method for efficient knowledge distillation of transformer-based object detectors.
- Attention distillation makes use of the self-attention matrices generated in the layers of detection transformer (DETR) models.
- Localization information from the attention maps of a large teacher network are distilled into smaller student networks capable of running at much higher speeds.
- We apply the approach to the clinically important problem of detecting medical instruments (e.g. needle insertion procedures) in real-time from ultrasound video sequences, where inference speed is critical on computationally resource-limited hardware.

Data
- Ultrasound video sequences acquired from ~12,200 needle insertions (~2 million individual frames) were used for model training and evaluation.
- Data were collected over two years from ex vivo tissues (porcine, bovine, and chicken) as well as human cadavers, and comprised a range of ultrasound transducers, systems, ultrasound imaging settings (gain, depth, and tissue presets), needle types, needle sizes, insertion angles, and bevel orientations.
- A total of 30,770 labeled video clips were used as the training set, and 5,023 labeled clips from independent data collection sets, and 5,023 labeled clips from independent data collection sizes, insertion angles, and bevel orientations.

Attention Distillation

- We apply attention distillation by making use of self-attention matrices generated within the encoder-decoder detection transformer architecture.
- Multi-headed scaled dot-product attention Carion et al. (2020) is applied to learned query, Q, and key, K, matrices.

\[ \lambda = \text{softmax} \left( \frac{QK^T}{\sqrt{d_k}} \right) \]

\[ L_{\text{attention distillation}} = (1 - \alpha) \cdot L_{\text{box}}(b_i, \hat{b}_i) + \alpha \cdot \left( K \mathcal{L} (A_i \| \hat{A}_i) + T^2 \cdot K \mathcal{L} \left( \sigma \left( \frac{\sigma(b_i^D)}{\sigma(\hat{b}_i^D)} \right) \right) \) \]

In the equation above, \( \alpha \) is a hyper-parameter that controls mixing between the bounding box loss and the attention distillation loss, where \( b_i \) and \( \hat{b}_i \) refer to the ground truth and predicted bounding box coordinates, and \( A_i \) and \( \hat{A}_i \) are class prediction probabilities given by the student and teacher networks, respectively.

- The first component of the loss applies knowledge distillation to the self-attention maps created by teacher and student networks.
- The second component of the loss applies knowledge distillation to the class label predictions. \( T \) is a temperature hyper-parameter that controls smoothing, as in Hinton et al. (2015), and \( \alpha \) is the softmax operation.

Results

2D-to-2D Attention Distillation for Images
- Our teacher network (DETR-R50-6/8) is a detection transformer with ResNet-50 backbone and six encoder and decoder layers. We trained smaller student networks (DETR-R50-1/1) comprising an identical backbone but consisting of only one encoder and decoder.

3D-to-2D Attention Distillation for Videos
- We apply attention distillation to compress a 3D detection transformer, which infers on a video sequence, into a 2D student model that processes single frames independently.

Table 1 & 2. Model sizes, inference speeds, and MAP of attention distillation student DETR models compared to a large 2D teacher model (DETR) and 3D teacher model (DETR-R50). DETR-R50-50 refers to the model type, where 50 indicates the number of encoder and decoder layers. All student models were trained with \( \alpha = 0.7 \). For comparison, a baseline model trained without attention distillation (\( \alpha = 0 \)) is also shown, as well as comparison to a Faster R-CNN model. Reported inference speeds (FPS) are based on model inference on a P100 GPU.

References