


Building a Size Constrained Predictive Model for Video Classification

Miha Skalic
David Austin
Munich, Sep 9th 2018



Next Top GB Model

- **Miha Skalic**
 - Last year PhD in Biomedicine doing deep learning for Drug Design
 - Barcelona, Spain
- **David Austin**
 - Senior Principal Engineer at Intel
 - Chandler, Arizona

The label assignment challenge

- Over 6 million training videos, 3800 categories
- Samples are sequences of of varying lengths vector
 - 1 second image + audio embedding

- Build a predictive model as a single Tensorflow graph
- Size limit: < 1GB

Primary Design Strategy

- Use frame level features
- Stick to proven model architectures, 1GB is reasonable for multiple models once compressed
- Focus on known model compression techniques
- Once all known techniques are exhausted, experiment with ways to expand their capability

Used models

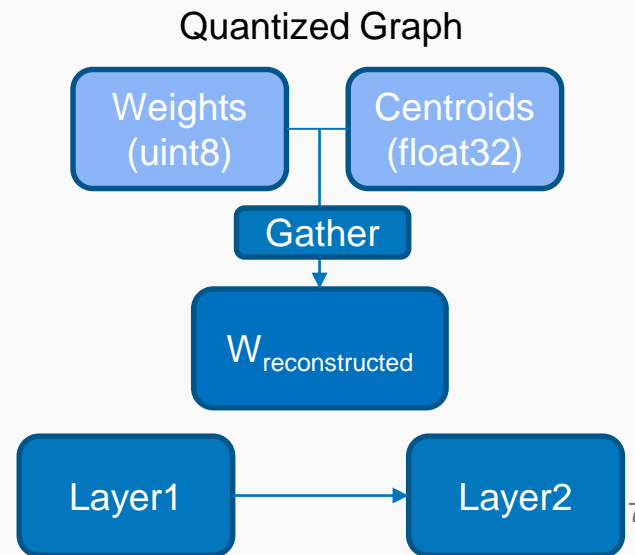
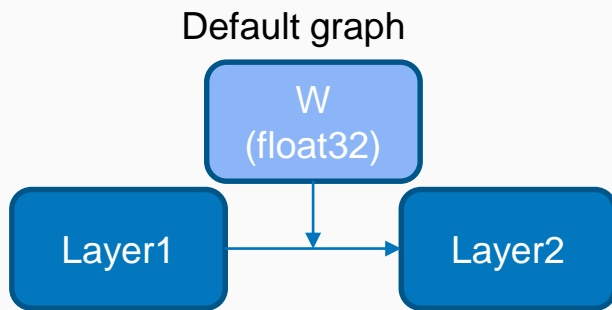
- RNNs (LSTM + GRU)
 - Aggregate frames information into RNN memory state
- Deep Bag of Frames
 - (wide) FC -> pool -> FC
- netVLAD
 - Soft assignment of descriptors to cluster
- netFV
 - Similar to netVLAD, learnable 2nd order statistics in cluster

Model quantization (I)

- Bigger is better:
 - 2017 best solution contained several >1GB models
- Usage of partial 8-bit weights quantization
 - Reducing model size to 1 / 4
- Partial weights quantization
 - Only quantize weights that are big (17k elements in our case)
 - BatchNorm on 1000 unit layer (float32) -> 2x 3KB
 - 1000x1000 FC layer mapping -> 3 MB
 - 12k x 4k FC -> 144MB (as float32), 36MB as unit8

Model quantization (II)

- Uniform min-max quantization
- Variable casts to uint8
- 256 float32 centroids
- At train time Gather
 - ~6% longer runtime



Distillation

- Training student networks on soft-labels (teacher outputs) yield better performance
 - Smaller models can be trained
 - Multiple teacher networks per student networks

$$L = \lambda * BCE(y_p, y_t) + (1 - \lambda) * BCE(y_p, y)$$

$\lambda=0.5$, y_p – predicted probability, y_t – teacher prediction,

In Model Weights averaging

- Exponential moving average
 - Train until convergence, then initialize averaging
 - Best performance gains on smaller models (RNNs, big models NetVlad little to no gain)
 - Equal weights averaging performed similarly

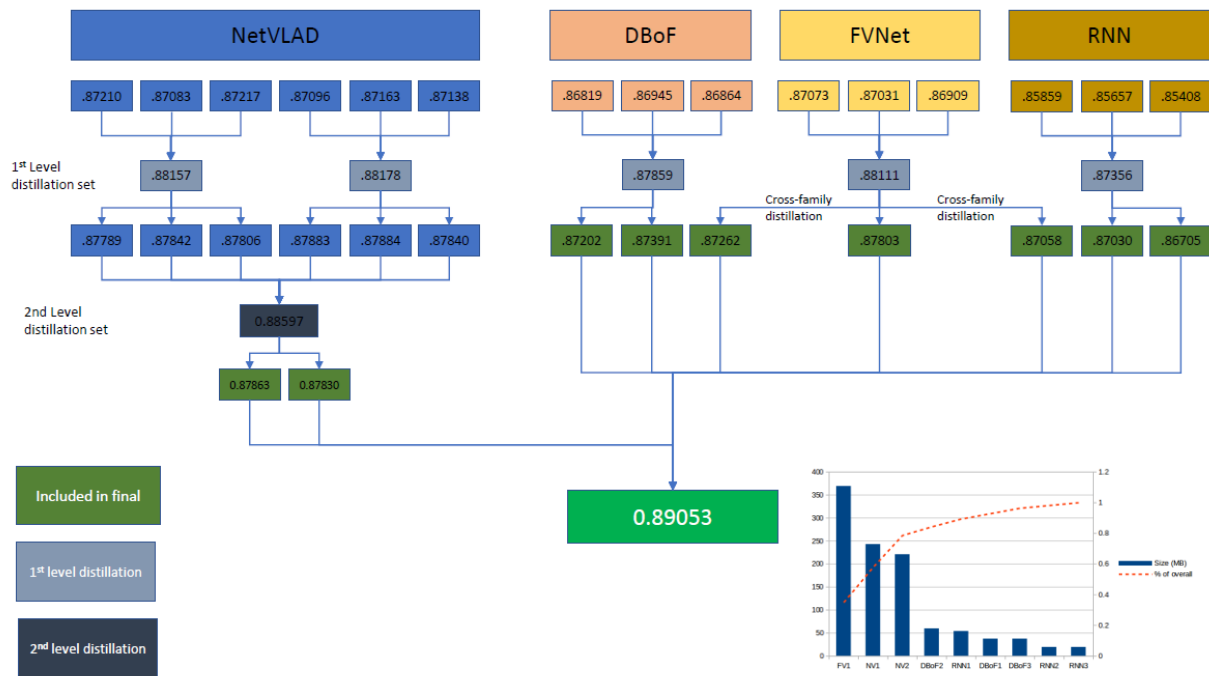
$$W_{\text{ema}} = (1 - \gamma) * W_{\text{ema}} + (\gamma * W)$$

Model Family	GAP change post weight averaging
RNN	0.00339
FVNet	-0.00186
DBoF	0.00216
NetVLAD	0.00072

Sampling as data augmentation

- During training sample (300) frames with replacement
- During prediction ensure that each frames is sampled at least once.
- +0.00150 GAP improvement

Solution outline



NetVLAD - Fastest to train (~1.5 days), best GAP, trained to multiple levels of distillation.

Distillation improved individual model performance by 0.004-0.015.

Cross family distillation did not improve individual model performance, but improved overall ensemble GAP.

Things that did not work

- Innovative and novel architectures
- Under/over sampling
- Non uniform quantization (e.g. quantile quantization)
- Mixup data augmentation

Thanks!



Code available:

<https://github.com/miha-skalic/youtube8mchallenge>