Part1

2024年3月20日 14:54

Image video basics

bit depth/ color space/

Data acquire

CCD/ CMOS lens+ccd+processing

Video
scene↑
shot↑
frame↑

Arithmetic 1:
24bit RGB video, 640*480/30fps
640*480*3*8*30

interlaced/ progressive

Video Signal Processing

Standards

Jpeg MPEG H.261(DCT) 264 265

Video Analysis

Detection/ pose estimation/ action recognition

Part2

2024年3月20日 14:57

Terms and Concepts

compression ratio = before/after always greater than 1

Shannon Entropy

$$\mathrm{H(S)} = -\sum_{i=1}^{n} p_i log_2 p_i$$

it means that: for a symbol S, has n different possibilities, when the S_i is unlikely to happen, it has more information.
so the symbol S average information is H(S) bits. This is the limitation of encoding.

Entropy Coding

Entropy coding is universal for all kinds of information, it consider only the binary bit stream, lossless compression.

Huffman coding: variable length coding

```
The codeword used for each character/symbol is determined by tracing the path from
the root node to the leaf node.
```

[Huffman Coding 1](#)

Image & Video Compression Basics

spatial/ temporal/ coding redundancy
frequency/ color masking
sensitive to low freq and luminance not in high freq and chrominance

some metrics:
MSE:
SNR:
PSNR:

Transform-based Coding / Compression

compact components/ transformed and easy to code/ quantization and coding

Discrete Cosine Transform (DCT)

DCT is used to transform s_{ij} to S_{uv}

$$S_{uv} = a(u)a(v)\sum_{i=0}^{N-1} \sum_{j=0}^{N-1} s_{ij} cos\frac{(2i+1)u\pi}{2N} cos\frac{(2j+1)v\pi}{2N}$$

$$\mathrm{a(k)} = \begin{cases} \sqrt{\frac{1}{N}}, & k = 0 \\ \sqrt{\frac{2}{N}}, & other \end{cases}$$

not based on image, universal
noted that S00 = 1/N sigma(sij)

Part 2 Solution to Exercise on 2DDCT
matrix implementation

$$F(u, v) = \mathbf{T} \cdot f(i, j) \cdot \mathbf{T}^T. \tag{8.27}$$

We will name $\mathbf{T}$ the *DCT-matrix*.

$$\mathbf{T}[i, j] = \begin{cases} \frac{1}{\sqrt{N}}, & \text{if } i = 0 \\ \sqrt{\frac{2}{N}} \cdot \cos\frac{(2j+1) \cdot i\pi}{2N}, & \text{if } i > 0 \end{cases} \tag{8.28}$$

Part 2 Solution to Exercise on 2DDCT Using Matrix Implementation

JPEG Standard

baseline jpeg

Step1	Step2	Step3	Step4	Step5	Step6
8*8block	DCT	Quantization and Compression	ZigZag	Entropy	Dataframe
		coefficient truncation and scale			

Zig-Zag scanning is used to serialize 2D mat to 1d seq.

Since the DC is usually large, use differential coding DPCM
Differential coding is used as average intensity between 2 consecutive blocks is similar.

while AC have many 0s, use run-length coding(skip value pair) RLC and then Huffman coding.

Part 2 Solution to Exercise on Basis Function and Quantization
Part 2 Solution to Exercise on JPEG

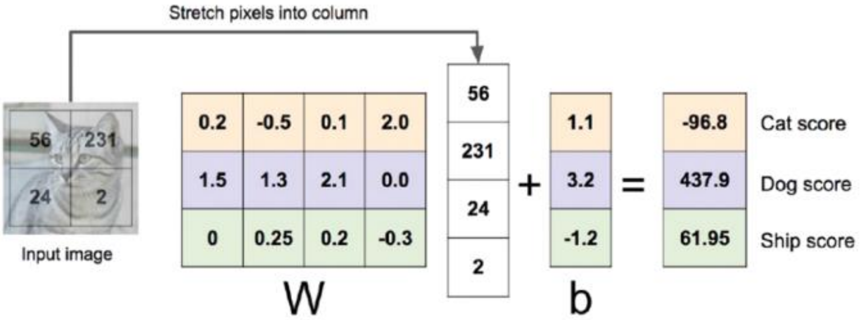
Part3A to 136

2024年3月20日 14:57

DNN

why different deep neural networks?
for different use. solve unique problems

cnn: progressively extract features, for classification and regression.(supervised)
rnn: for sequence, prediction and translation.
linear classifier:



some loss function:

- Square loss:

$$L(x,y) = \sum_i (y_i - f(x_i))^2$$

- Mean Square Error (MSE):

$$MSE = \frac{1}{N}\sum_i (y_i - f(x_i))^2$$

- Mean Absolute Error (MAE):

$$MAE = \frac{1}{N}\sum_i |y_i - f(x_i)|$$

other loss:
softmax loss,

$$L = -\sum_j y_j \log p_j =$$

y is the ground truth(label, 0 and 1)p is the softmax possibilities

CNN

Conv layer	later layer with high level features	after first conv, no anymore RGB, add together after elementwise product with conv kernel.	of course we need padding
activation layer	relu sigmoid tanh	often combined with conv layer	
Pooling	reduce dimension	max pooling average pooling	
FC	feature/embedding		
softmax	possibilities		

	distribution		

training use SGD,Adam optimizer

Alex net/ VGG/ ResNet

Performance metrics: acc, memory footprint, speed flops,

[Part 3A Solution to Exercises on CNN]

RNN

when seq2seq modelling: encoder decoder
consider old state

$$\underbrace{\boxed{h_t}}_{\text{new state}} = \underbrace{\boxed{f_W}}_{\text{some function with parameters W}}(\underbrace{\boxed{h_{t-1}}}_{\text{old state}}, \underbrace{\boxed{x_t}}_{\text{input vector at some time step}})$$

the formula is

$$\mathbf{h}_t = \tanh(\mathbf{W}_{hh}\mathbf{h}_{t-1} + \mathbf{W}_{xh}\mathbf{x}_t)$$

$$\mathbf{y}_t = \mathbf{W}_{hy}\mathbf{h}_t$$

same W for each timestep, use many step, but slow

batch training: full, stochastic, minibatch

use truncated backpropagation, because the sequence could be very long.
[Part 3A Solution to Exercises on RNN]
gradient vanishing and exploding problem: when multiply over times, cause it. CLIPPING or change RNN

LSTM

h for short mem, c for long mem, and gates "ifog"

Vanilla RNN

$$h_t = \tanh\left(W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right)$$

LSTM

$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

$$c_t = f \odot c_{t-1} + i \odot g$$

$$h_t = o \odot \tanh(c_t)$$

Part 3A Solution to Exercises on LSTM(1)

sigmoid:1/(1+e^-x)

Transformer

ViT learnable class embedding->transformer encoder-> MLP-> classification

Part 3A Solution to Exercises on Model Comparison

Part 1 Intro

2024年4月30日 17:01

EE6427 Lecture Part 1 AY2324S2

Part1

Part 2 Compression

2024年5月1日 1:58

EE6427 Lecture Part 2 AY2324S2

Part2

Part 3A AI models

2024年5月1日 1:58

EE6427 Lecture Part 3A AY2324S2

Part3A to 136

Part 3B Video Analysis

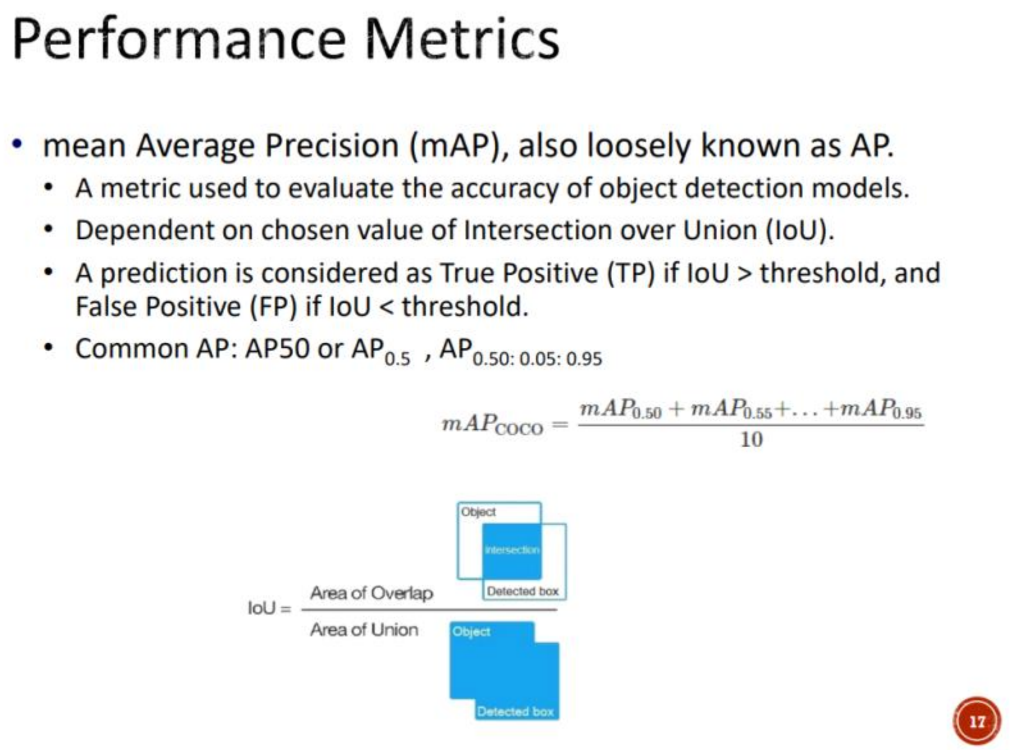
2024年5月1日 1:58

EE6427 Lecture Part 3B AY2324S2

1. Object Detection/ Tracking

object detection: regression + classification

performance metric



one stage detector: don't have region proposal generator
YOLO, SSD
YOLO: backbone: feature extraction. different scale
Neck: make it complex, upsampling, fusion feature.
Head: bounding box, classification

two stage detector: proposal first and then regression and classification
RPN , Faster RCNN Mask RCNN
high acc, low speed

lightweight detector
small and fast but less acc

mobileNet: depthwise separable convolution reduce computation

swin transformer: hierarchical, patch attention only within current small region.
window based multihead self attention, in next layer shift the window.

DETR: parallel CNN+Transformer

2. Pose Estimation

track->motion prediction-> data association
motion modeling: kalman filtering/ particle filtering

data association: base on IOU, calc the distance.

Trackformer: track by attention, CNN Transformer encoder/decoder. object query, track query,

3. Human Action Recognition

Performance metrics:
- PCK (Percentage of Correct Keypoint)
- PCP (Percentage of Correct Parts)
- AP and AR based on OKS (Object Keypoint Similarity)

single-person: regression method: use CNN
body part detection method: use heatmap

multi-person: HRNet top down method
bottom up: detect body parts, assemble.

TransPose: resnet, Transformer encoder, n times, estimate body keypoints

HAR:

Two stream networks: use both spatial and temporal(optical flow) information, score fusion and prediction.
Early years.

optical flow, orthogonal info against RGB, but computational intensive, storage requirements.

3D CNN: 3D tensor,
Slowfast Network.
Pros: can extract spatial and temporal info simultaneously

Efficient video modeling: TSM, no optical flow, no 3D tensor convolution
TSM: traditional 2DCNN, realtime online shift: unidirectional
offline shift: bidirectional

Transformer based:

video swin transformer: long range dependency, computational intensive.

Video Swin Transformer

- Patching merging
  - Perform 2× spatial downsampling and concatenate features of each 2×2 spatially neighboring patches.
  - Do not downsample along the temporal dimension.
  - Apply a linear layer to project the concatenated features.

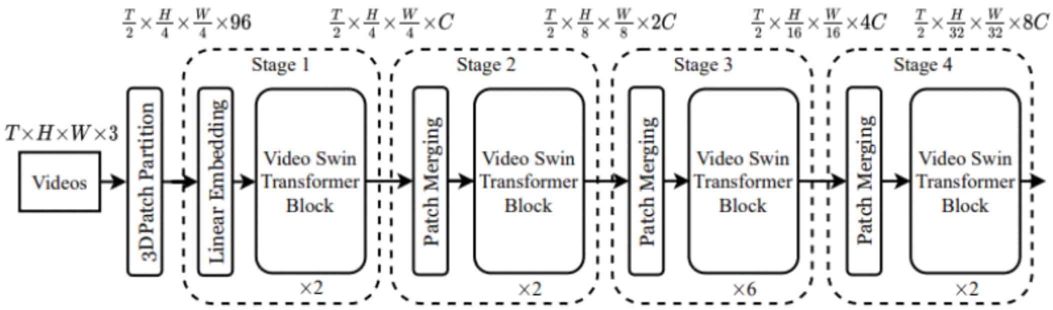


Figure 1: Overall architecture of Video Swin Transformer (tiny version, referred to as Swin-T).

Performance Comparison

Methods	Representative Models			Input Size	GFLOPs × views	Accuracy (Top1 %) (Kinetics 400)	FPS	Remarks
	Model	Year	Venue					
Two-stream networks	TSN	2016	ECCV	8×3×224×224	16×250	72.45	18.6	Simple design, significant computation and storage requirement for optical flows
3D CNNs	I3D	2017	CVPR	32×3×224×224	108×N/A	74.87	0.8	Complex, very large computation consumption in training
	SlowFast	2019	CVPR	32×3×224×224	234×30	81.8	0.8	
Efficient video modeling	TSM	2019	ICCV	16×3×224×224	33×30	74.1	18.1	Simple, efficient training, fast runtime, relatively accurate
	TDN	2021	CVPR	24×3×224×224	198×30	79.4	-	
Transformers	VTN	2021	ICCV	250×3×224×224	4218 × 1	78.6	-	Accurate, large data requirement, high computational cost
	Video Swin-L	2022	CVPR	32×3×384×384	2107 × 50	84.9	0.6	
Skeleton based networks	ST-GCN	2018	AAAI	-	-	30.7 (Kinetics 400) 86.9 (NTU60_XSub)	-	Leverage on human pose, lower accuracy in broad domain applications.
	PoseC3D	2021	ArXiv	8×3×224×224	-	47.4 (Kinetics 400) 94.3 (NTU60_XSub)	-	

Part 4 Video Compression

2024年5月1日 1:58

EE6427 Lecture Part 4 AY2324S2

Video Coding:

motion estimation, motion compensation.

Motion Estimation (2)

- The difference between two macroblocks can be measured by Mean Absolute Difference (MAD):

$$MAD(i,j) = \frac{1}{N^2}\sum_{k=0}^{N-1}\sum_{l=0}^{N-1}\left|C(x+k, y+l) - R(x+i+k, y+j+l)\right|$$

N: size of the macroblock,
k and l : indices for pixels in the macroblock,
i and j : horizontal and vertical displacements,
$C(x+k, y+l)$: pixels in macroblock of Target frame,
$R(x+i+k, y+j+l)$: pixels in macroblock of Reference frame.

- Goal: to find the motion vector *MV* = (*u*, *v*) such that *MAD(i, j)* is minimum:

$$(u, v) = [(i, j) \mid MAD(i, j) \text{ is minimum}, \quad i \in [-p, p], j \in [-p, p]]$$

Motion Estimation Methods

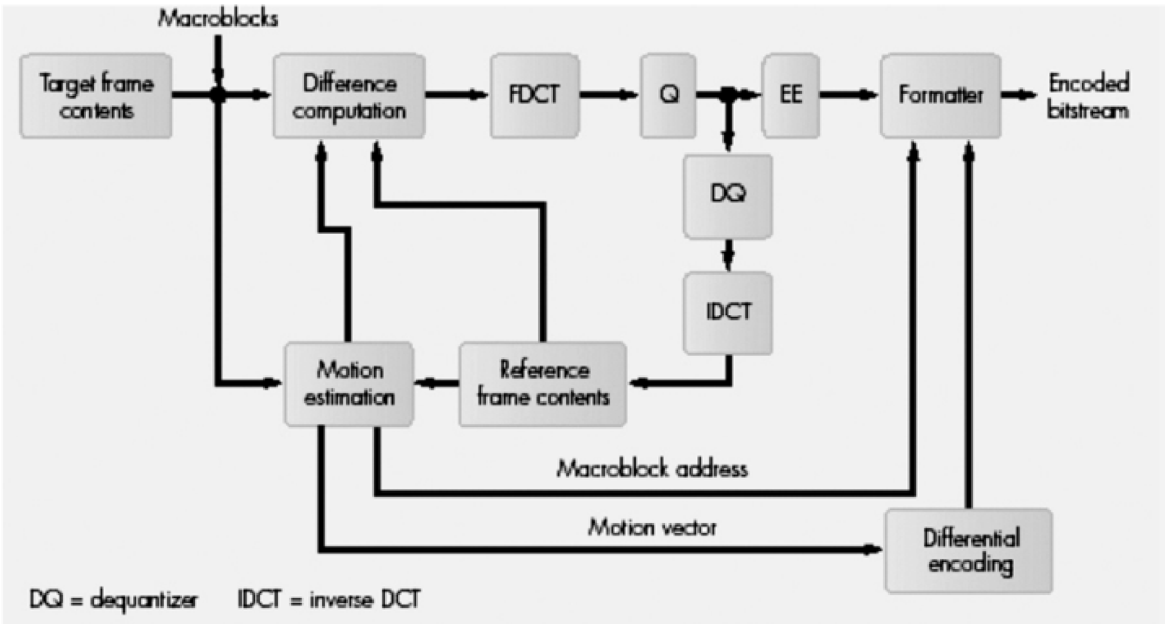
- Full Search
- Three-step Search
- 2D-Log Search
- Hierarchical Search

Hierarchical downsampling 2 times and motion estimation

MPEG:

MPEG-1: I-Frame Encoding

MPEG-1: P-Frame Encoding Flowchart

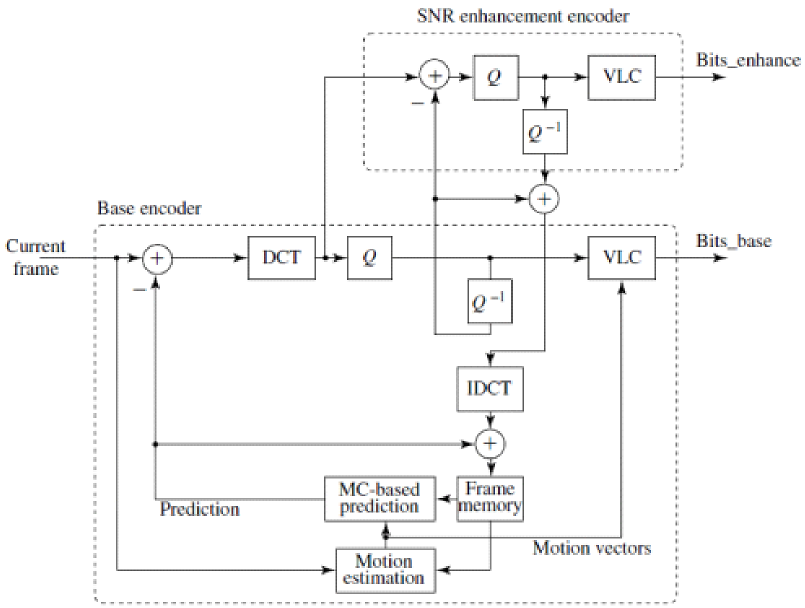


MPEG-2: Overview

- Aim to address limitations of MPEG-1: e.g., low bitrate (1.5 Mbps), progressive-scan only.
- Standardized in 1995.
- Developed for digital broadcast TV (interlaced-scan) at a high bitrate (4 Mbps).
- Defined different profiles for different applications.
- Support scalable coding.

MPEG2 scalability: base layer and enhancement layer
scalability in SNR: base layer use large quantization table.

MPEG-2: SNR Scalability



(a) Encoder

MPEG4:

object based coding. VOP

H.26X

H.261: constant step size of quantization table for I frame

H.264

• Context-Adaptive Variable Length Coding (CAVLC) and Context-Adaptive Binary Arithmetic Coding (CABAC) • More robust to data errors and data losse

H.264 motion compensation
Part 4 Solution to Exercise on H264 Motion Compensation

H.264 no B frame

integer transform derived by 4*4 DCT , transform mat is orthogonal, but need norm, involved in quantization

no need to mem this pic

H.264: Quantization and Scaling

- Let **f** be 4 × 4 input matrix, and $\hat{\mathbf{F}}$ quantized transform output.
- The forward integer transform, scaled and quantized:

$$\hat{\mathbf{F}} = \text{round}\left[(\mathbf{H} \times \mathbf{f} \times \mathbf{H}^T) \cdot \mathbf{M_f}/2^{15}\right]$$

where “×” denotes matrix multiplication, “·” denotes element-by-element multiplication, and M_f is the 4 × 4 quantization matrix derived from matrix **m** and quantization parameter QP.

Intra coding:
[Part 4 Solution to Exercise on H264 Intra Coding]