M2CAI WORKFLOW CHALLENGE 2016

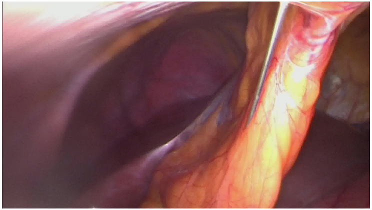
Fine tuning CNN with HMM smoothing

21th October 2016

Rémi Cadène, Thomas Robert, Nicolas Thome, Matthieu Cord

University Pierre and Marie Curie - LIP6 - MLIA

M2CAI Workflow Dataset



Videos resolution is 1920 x 1080, shot at 25 frames per second at the IRCAD research center in Strasbourg, France.

- 27 training videos ranging from 15mn to 1hour
- 15 test videos

M2CAI Workflow Dataset

1 of 8 classes for each frames :

- TrocarPlacement
- Preparation
- CalotTriangleDissection
- ClippingCutting
- GallbladderDissection
- GallbladderPackaging
- CleaningCoagulation
- GallbladderRetraction

M2CAI Workflow Goal and Measure

Goal

Online prediction : $P(y|x_i, x_{i-1}, x_{i-2}, ...)$
x_i := frame i, and y := classes

Useful to

- Monitor surgeons
- Trigger automatic actions

Measures

- Jaccard similarity coefficient : $J(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$
- Accuracy top1 : nb frames well classified / nb total frames

Two fold approach

1. Frames classifier using Deep Learning

- From Scratch Convolutional Neural Network (CNN)
- Features Extraction CNN
- Fine tuning CNN

2. Smoothing predictions

1. Averaging predictions over last 15 frames
2. Hidden Markov Model (HMM) as a "denoizer"

Creating a trainset and valset of images

Creating validation set by random split

- Training set : 22 videos
- Validation set : 5 videos $\{2, 9, 10, 13, 27\}$

Extracting one frame every 25 frames (1 frame per second)

- Training set : 59,493 images
- Validation set : 8,062 images
- Testing set : 28,732 images

Training CNN From Scratch

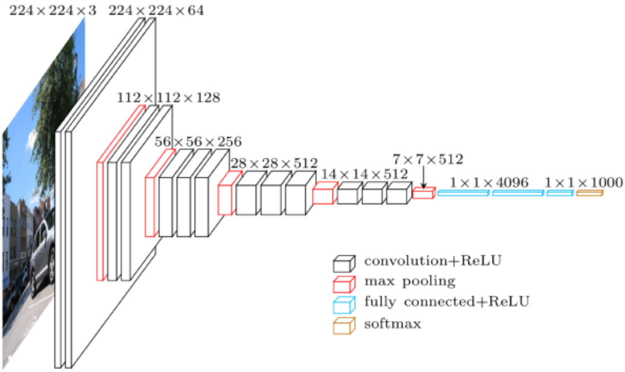
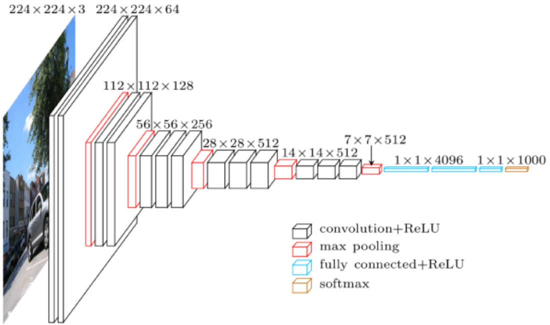


Figure 1 – Vgg16 [**simonyan2014very**], top2 ILSVRC2014

Using representations learned on ImageNet

Pre-trained CNN as Features Extractor

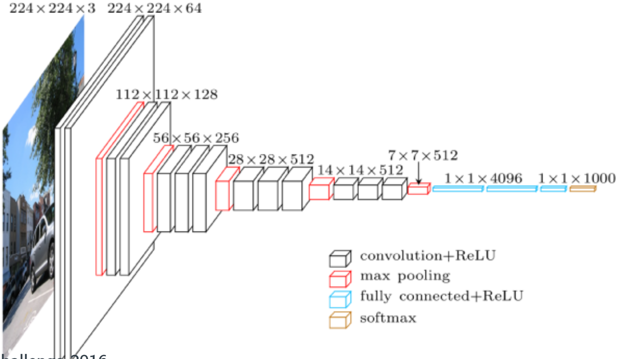
1. Extracting features somewhere
2. Training a Support Vector Machine



Adapting representations learned on Imagenet

Fine tuning a pre-trained CNN

- Same process than CNN From Scratch
- But smaller learning rate for pre-trained layers



Which CNN to use ? Possible in production ?

Model	Input	Param.	Depth	Implem.	Time (ms)
Vgg16	224	138M	16	GPU	
InceptionV3	399	24M	42	GPU	**0**
ResNet-200	224		200	GPU	
InceptionV3	399	24M	42	CPU	0

Table 1 – Forward+Backward with batches of 30 images.

Comparison of frames classifiers

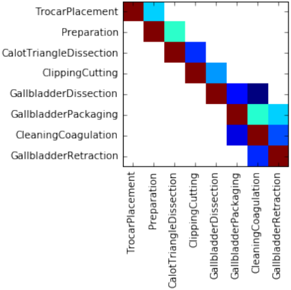
Classification Model	Accuracy (%)
InceptionV3 Extraction	60.53
InceptionV3 From Scratch	69.13
InceptionV3 Weldon	78.18
InceptionV3 Fine-tuned	79.06
ResNet200 Fine-tuned	79.24

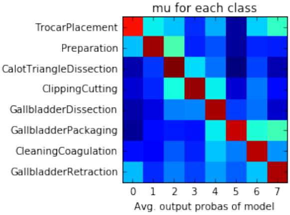
Table 2 – Accuracy on the validation set.

Gaussian Hidden Markov Model

HMM on the smoothed predictions over last 15 frames

- the initial state probabilities
- the matrix of probabilities of transition between states
- the emissions of observations





Gaussian Hidden Markov Model

Training process

Counting

Testing process

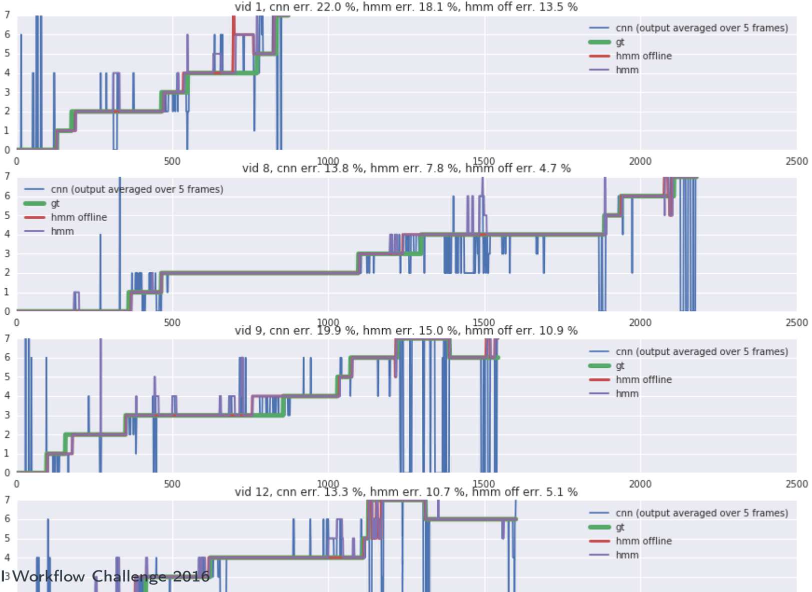
Offline testing : Viterbi algorithm to obtain the most likely sequence of states

Online testing : to predict x_t we apply Viterbi on the sequence $y_1, \ldots, y_t$

Comparison of temporal smoothing methods

Temporal Method	Accuracy Val (%)	Jaccard Val	Jaccard Test
Avg Smoothing	85.97	74.67	–
HMM Online	88.90	81.60	71.
HMM Offline	93.47	87.59	

Visualization



Conclusion

Conclusion

- Deep Learning efficient
- Fine Tuning most accurate approach
- HMM is usefull to smooth the predictions

Future work

- train on 100%
- ensembling

Code available : `github.com/Cadene/torchnet-m2caiworkflow`

References I