Exam ss20 ret - Exam ss20 ret

Computer Vision III: Detection, Segmentation and Tracking (Technische Universität München)

Computer Vision Group
Department of Informatics
Technical University of Munich

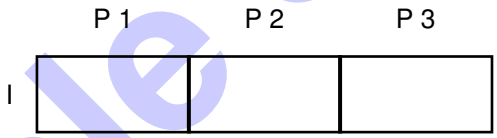
Esolution
Place student sticker here

Note:

- During the attendance check a sticker containing a unique code will be put on this exam.
- This code contains a unique number that associates this exam with your registration number.
- This number is printed both next to the code and to the signature field in the attendance check list.

Computer Vision III: Detection, Segmentation, and Tracking

Exam:	IN2375 / Retake	**Date:**	Wednesday 30th September, 2020
Examiner:	Prof. Leal-Taixe	**Time:**	11:45 – 13:15



Left room from _____ to _____

from _____ to _____

Early submission at _____

Notes _____

Computer Vision Group
Department of Informatics
Technical University of Munich

Retake

Computer Vision III: Detection, Segmentation, and Tracking

Prof. Leal-Taixe
Computer Vision Group
Department of Informatics
Technical University of Munich

Wednesday 30th September, 2020
11:45 – 13:15

Working instructions

- This exam consists of **12 pages** with a total of **3 problems**.
  Please make sure now that you received a complete copy of the exam.
- The total amount of achievable credits in this exam is 60 credits.
- Detaching pages from the exam is prohibited.
- Allowed resources:
- **Answers are only accepted if the solution approach is documented.** Give a reason for each answer unless explicitly stated otherwise in the respective subproblem.
- Do not write with red or green colors nor use pencils.
- Physically turn off all electronic devices, put them into your bag and close the bag.

Problem 1 Multiple Choice Questions: (12 credits)

- For all multiple choice questions any number of answers, i.e. either zero (!), one, all or multiple answers can be correct.
- For each question, you'll receive 2 points if all boxes are answered correctly (i.e. correct answers are checked, wrong answers are not checked) and 0 otherwise.

How to Check a Box:

- Please **cross** the respective box: ☒ (interpreted as **checked**)
- If you change your mind, please **fill** the box: ■ (interpreted as **not checked**)
- If you change your mind again, please place a cross to the left side of the box: ☒■ (interpreted as **checked**)

a) Mark all the true statements for video object segmentation metrics:

- [ ] The F-measure computed on the boundary pixels is used to measure region similarity.
- [ ] The Jaccard index will be the same if our prediction mask covers 0.5 of the true object and we have no false positives, than if we cover the whole object and we have a false positive region which is equal to 0.5 times the true object region.
- [ ] The Jaccard Index measures segmentation contour accuracy.
- [ ] Decay is a measure used to quantify the performance loss (or gain) over the temporal domain.

b) Which of the following statements is true about Message Passing Networks:

- [ ] They are linear functions of node and edge feature vectors
- [ ] None of the statements is correct
- [ ] They are invariant to node permutations
- [ ] They can only encode pairwise interactions between node features

c) Check all that apply for object detectors:

- [ ] ne-stage object detectors are faster but less accurate than two-stage object detectors.
- [ ] aster R-CNN is a one-stage object detector.
- [ ] ast R-CNN uses anchors.
- [ ] -CNN does one forward pass through the backbone CNN for every proposal.

d) Which of the following statements is true about Message Passing Networks (check all that apply):

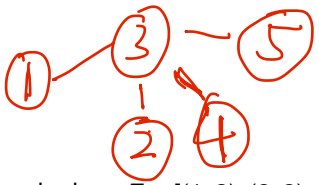
- [ ] They are invariant to node permutations.
- [ ] They can only encode pairwise interactions between node features.
- [ ] None of the statements is correct.
- [ ] They are linear functions of node and edge feature vectors.

e) Mean Average Precision (mAP) is used to evaluate object detectors. Which of the following statements is true (mark all that apply):

- [x] To compute AP, we consider bounding box predictions with intersection-over-union below a certain threshold τ to be false negatives.
- [ ] Average Precision (AP) does not change with the number of low-score False Positives (FP) when all True Positives are covered by high-score predictions.
- [ ] First the Average Precision is computed for each bounding box, then the mean of AP is taken over all bounding boxes.
- [x] First the Average Precision is computed for each class, then the mean of AP is taken over all classes.

f) Check all that apply for The Group Loss for Deep Metric Learning:

- [x] It uses a classification-based loss function.
- [x] It uses proxies to represent the classes.
- [ ] It proposes an advanced sampling strategy.
- [ ] It uses all the relations between the samples in the minibatch.



Problem 2 Short questions (28 credits)

You are given an undirected graph $G = (V, E)$ with 5 nodes, $V := 1, 2, 3, 4, 5$ and edges $E = [(1, 3), (2, 3), (3, 4), (3, 5)]$. Every node i corresponds to an object detection observed at a time step i. Neural message passing is being performed on G. Assume that a round of *node-to-edge* updates has been performed, and you have access to the intermediate *messages*: $m(1, 3) = (1, 0)'$
$m(2, 3) = (0, 1)'$
$m(3, 4) = (1, 1)'$
$m(3, 5) = (0, 0)'$

0 1 a) For the *edge-to-node* updates, we are using the *mean* operator. Perform a regular *edge-to-node* update on node 3.

Solution: (1 / 4) * (1 + 0 + 1 + 0, 0 + 1 + 1 + 0)' = (0.5, 0.5) '

0 1 2 b) You are given an additional $4x2$ weight matrix $N_v = ((1, 1, 1, 1), (1, 1, 1, 1))$. Perform a time-aware *edge-to-node* update on node 3.

First, we compute the mean separately for messages from nodes in past frames (1 and 2), and nodes in future frames (4, 5):
$(1/2) * (1 + 0, 0 + 1)' = (0.5, 0.5)'$ $(1/2) * (1 + 0, 1 + 0)' = (0.5, 0.5)'$ (1p if both computations correct)
Now, we concatenate these two vectors and multiply the result with matrix N_v: $((1, 1, 1, 1), (1, 1, 1, 1)) * (0.5, 0.5, 0.5, 0.5) = (2, 2)'$ (1p)

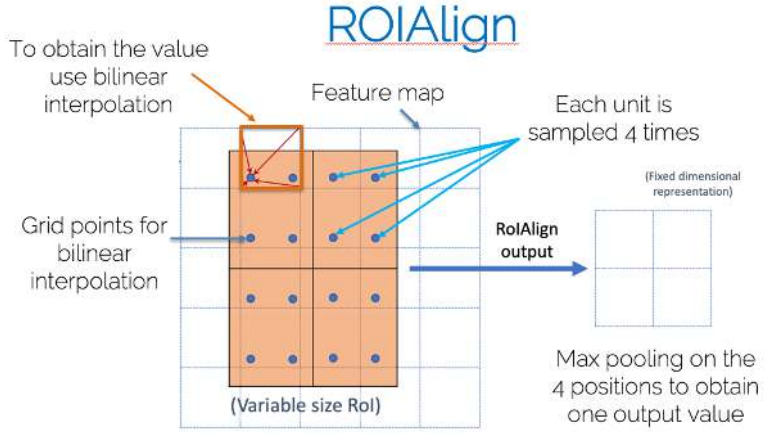
0 1 2 3 c) How many multiplications are done in a layer of depth-wise separable convolutions with $5x5$ kernels on a feature map of $9x9x7$ (no padding, stride of 1). There is no need to solve the multiplication, just write down the operations (multipliers) (1p) and their meaning (1p). What is the advantage of using depth-wise separable convolutions in networks like DeepLabv3+ (1p)?

7 kernels of size 5x5x1 applied to 5x5 locations = (7x5x5) x (5x5) locations (1p for expressing the right multiplication, 1p for the meaning of the variables).
DeepLabv3+ takes advantage of depth-wise separable convolutions to spare multiplications, hence reducing the computational cost. (1p)

d) Explain the concepts of RoIAlign (1p) and RoI pooling (1p). Use an example.

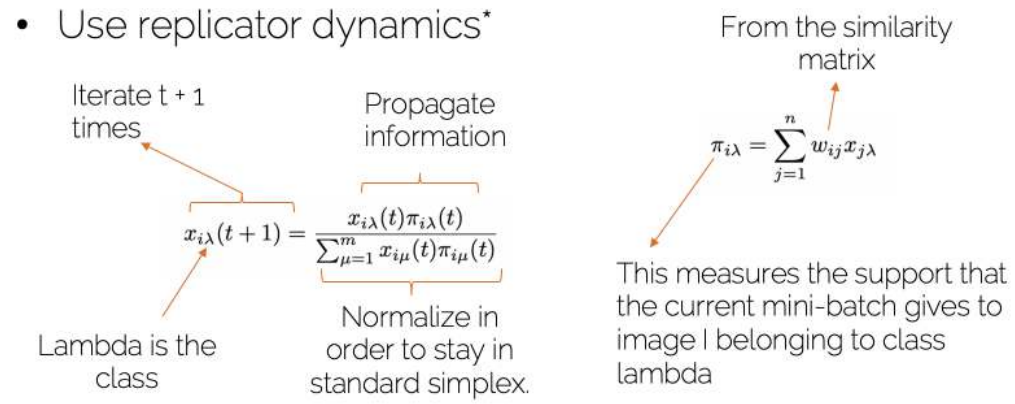
0
1
2

RoI simply max pools in the position where the bounding box is placed, without taking into account quantization effects.
RoI align want to get rid of those effects by really computing the value each pooled unit should have. For this, the units are samples 4 times, and the values are computed through bilinear interpolation (1p) (see image, as example, this is another 1p).



e) Write the equation for the refinement in The Group Loss paper (1p), make sure to explain all the variables. What is the purpose of the numerator (1p)? What is the purpose of the denominator and why is it needed (1p)? What is the main conceptual difference to softmax function typically used in neural networks (1p)?

0
1
2
3
4



It refines the matrix of priors considering the similarities between all the mini-batch images, as encoded in the similarity matrix, as well as their labeling preferences. The numerator propagates the information between the samples in the minibatch (1p).
The denominator is a normalization term that ensures that the labeling remains on the standard simplex (probability space) (1p).
The main conceptual difference to softmax function, is that while softmax function considers only the local information (the features of a sample), the refinement procedure in the Group Loss combines the local information (priors) with the global information (the similarity between all pairs on the minibatch) (1p).

0 1 2 3 f) Generalize the concept of dilated convolution to *dilated convolutions with arbitrary offsets* by writing their equation (1p). How are these convolutions called (1p)? Explain why they are important on segmentation networks like UPSNet (1p).

Regular convolution

$$\mathbf{y}(\mathbf{p}_0) = \sum_{\mathbf{p}_n \in \mathcal{R}} \mathbf{w}(\mathbf{p}_n) \cdot \mathbf{x}(\mathbf{p}_0 + \mathbf{p}_n)$$

Deformable convolution

$$\mathbf{y}(\mathbf{p}_0) = \sum_{\mathbf{p}_n \in \mathcal{R}} \mathbf{w}(\mathbf{p}_n) \cdot \mathbf{x}(\mathbf{p}_0 + \mathbf{p}_n + \Delta\mathbf{p}_n)$$

Deformable convolutions (1p)

They generalize the concept of dilated convolutions by also learning the offset, which is generated by a sibling branch of the regular convolutions. The main advantage of them is that they can pick values at different locations for convolutions conditioned on the input image of the featured space. Thus, it is not limited to rectangular shapes that standard convolutions use, making it more precise especially near the boundaries of objects with non-regular shapes. (1p)

0 1 2 g) You want to build a model that predicts the future pedestrian trajectory y_i based on the past trajectory x_i for different pedestrians *i*. For your model you choose an encoder-decoder architecture. What type of neural network would you choose for the encoder and decoder (1p) and why (1p)?

Recurrent Neural Network (RNN, GRU, LSTM) or (Transformer Network) (1p) => no points for FC network
RNN are more efficient because they account for the temporal/ sequential nature of the input and output data (1p)

0 1 2 h) In your validation set you find many samples that approach a crossroad where the trajectories can only turn left or right. However, you realize that your model always predicts a straight trajectory. Why is your model not capable of predicting a set of trajectories going left or right in these scenes? Briefly argue with respect to the model architecture (1p) and training loss (1p).

The model is deterministic (one-to-one mapping) (1p). Hence, it produces unrealistic averaged trajectories by minimizing MSE/L2 loss. (1p)

i) Modelling human behavior is very challenging and pedestrian trajectories also depend on (i) agent-environment and (ii) social interactions. Name one approach on how state-of-the-art pedestrian trajectory methods presented in the lecture tackle the aforementioned (i) and (ii) interactions (1p for each). Specify input and method/operation for both of the interactions.

0
1
2

(i) Agent-Environment Interactions: (1p)
Soft-Attention on visual features of scene using of CNN (e.g. VGG)
(ii) Social Interactions: (1p, only one example needed)
Social pooling of hidden state in neighborhood (Social LSTM)
Social Max-Pooling of hidden states of pedestrians
Soft-attention hidden state of pedestrians (SoPhie)
Graph Attention on hidden states (Social-BiGAT)

0 1 2 j) We discussed in the lecture that even very simple algorithms can provide a reasonably good segmentation of LiDAR point clouds, eg., connected components algorithm based on the bird-eye view of the point cloud. What is the key property of these signals that allows for this (1p)? Why such a technique does not work well for images (1p)?

These signals provide us reliable distance measurements to the surrounding surfaces. We can use this as a *grouping* cue: points, that are spatially nearby in the 3D point cloud, likely originate from the same object instance. (1p)
This is not the case for images, where ?depth separation? is lost due to projection. (1p)

0 1 2 k) In the first stage of PointRCNN object detector, we generate a compact set of 3D object proposals. A naive approach would be to define a regular grid over 3D space, and spawn anchor boxes from centroids of these cells. This would yield a large set of 3D object proposals. What are the two ideas (1p each), employed in PointRCNN to keep number of proposals low, while maintaining a high recall?

- First, they only spawn proposals where we have 3D LiDAR measurements.
- Second, they learn a background/foreground semantic mask, and spawn anchor boxes only from the *foreground* 3D points.

0 1 2 3 l) GNN3DMOT learns motion models in both image domain and 3D space based on LiDAR. How are appearances of tracks/detections encoded? State it for both the 2D case (1p) as well as the 3D case (1p). What is the underlying mechanism of graph neural networks that helps with learning more discriminative feature representations of detections/tracks, especially when there are several objects present that are difficult to distinguish (1p)?

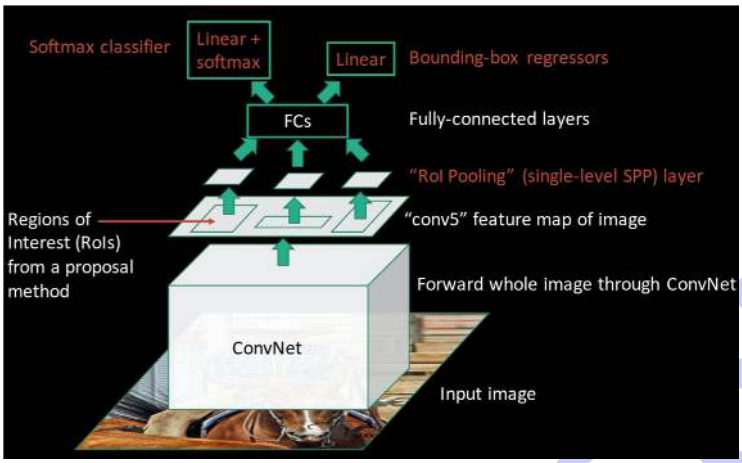
- For LiDAR, they crop points from 3D boxes and encode them using PointNet;
- For images, they extract 2D box by projecting 3D box to the image, and encode the content with a CNN (eg. VGG, ResNet)
- Feature aggregation: feature representations for detections/tracks are updated based on their neighbours in the graph. In case there will be several objects, difficult to distinguish, this will encourage object representation to be adapted and focus on the key properties of objects, that makes it distinguishable from their neighbours.

Problem 3 **Long question (20 credits)**

You are building part of the vision pipeline that should be built into a car to provide it with autonomous driving capabilities. For now you are building a prototype, so you are not concerned with computational time. You start by building the principal components of an object detector. You are given enough training data for the classes *pedestrians* and *cars*. You use a pre-existing algorithm that gives you object proposals.

a) You decide to use those pre-computed proposals and follow the idea behind *Fast R-CNN* (Girschick, "Fast R-CNN", ICCV 2015) in order to learn to predict pedestrians and cars among those proposals. Draw a diagram of the architecture in blocks (2p). Indicating the block that does image feature extraction, the type of operations involved in each block (convolutions, fully connected) and the losses used (2p). You do not need to write the mathematical formula for the losses, but specify the task they aim to solve and the shape of the loss (e.g., L1, L2...).

0 1 2 3 4



Classification head: classifies the box into one of the C semantic classes. (task needs to be specified). Loss: cross-entropy loss Regression head: estimates the bounding box change with respect to the position of the proposal. (task needs to be specified) Loss: L1, L2 both accepted

b) What operation allows you to extract features for proposals of varying sizes (1p)? Explain the operation with an example (1p).

0 1 2

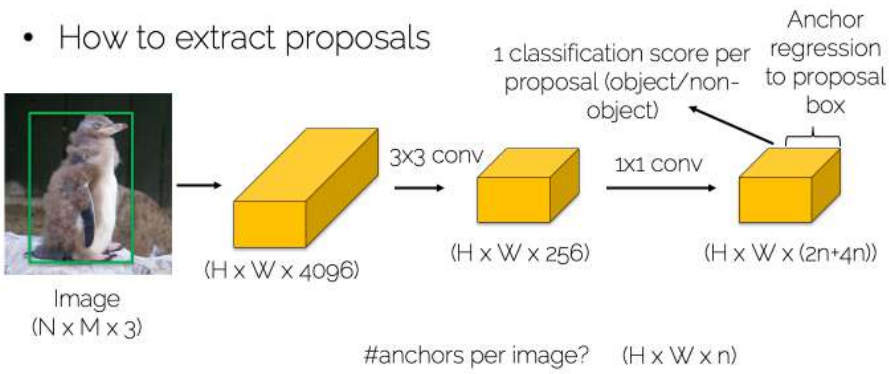
Solution: Region of Interest (RoI) pooling.

RoI pooling always pools the features into a fixed 7x7 dimension, for any feature input size.

Example: You put a 7x7 grid on top of the feature map, and you max-pool whatever falls into that feature map.

0 1 2 3 4 5

c) You observe that the region proposal algorithm that was given to you is slowing your system at test time, so you decide to embed it into the network. Draw the architecture of a Region Proposal Network (RPN) as seen in *Faster RCNN* (Ren et al, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", NIPS 2015). Make sure to specify the sizes of all the feature maps and your convolution kernels. Use H and W as the size of the feature map, C as the number of channels (use C_1, C_2 or as many as you need) and n as the number of anchors per location on the feature map (3p). How are proposals extracted from the RPN, i.e., what do the numbers in the last feature map represent (2p)?



n = number of anchors
2n = used to classify the object as object/non-object (1n also accepted)
4n = bounding box regression for each anchor

Your code is now trained to detect cars and pedestrians, so you move to the temporal domain. You now want to track these objects, so you decide to follow *Tracktor* (P. Bergmann et al. "Tracking without bells and whistles". ICCV 2019).

0 1 2

d) Explain what elements of your previous object detector can *Tracktor* re-use and how, in order to obtain tracks of cars and pedestrians from a video.

You can re-use the bounding box regressor. (1p)

Instead of using the RPN at each frame, you input the previously detected bounding boxes of frame $t-1$ as proposals for detection at frame t. Using the regressor, you now know where those boxes moved from frame $t-1$ to frame t, effectively solving the tracking problem for two frames. You repeat this procedure for every new frame. You also ran the plain detector in parallel in case pedestrians appear in the scene. (1p)

e) Your implemented *Tracktor* idea is working well in tracking isolated objects, but as soon as you have a crowded scene with many pedestrians, you observe a lot of *identity switches*. You decide to train a re-identification network for similarity learning. What architecture (with a ResNet-50 backbone) (1p) and loss (1p) do you use? Write down the formula for that loss (1p).

0 1 2 3

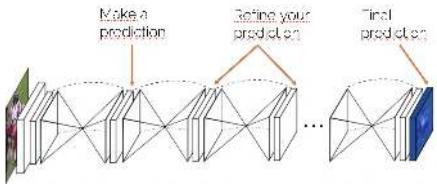
Architecture: siamese network. (1p)
Loss: Triplet loss (contrastive loss is also accepted) (1p)

Formula: $\mathcal{L}(A, P, N) = \max(0, |||f(A) - f(P)||||^2 - |||f(A) - f(N)||||^2 + m)$, where A indicates the anchor image, P the positive image, N the negative image and m the margin. (1p)

f) Your tracker is now quite stable, but you see the re-identification could be improved if we had body joint locations. You decide to reimplement the stacked hourglass architecture for human joint prediction. What is the output of such architecture and how does it represent joint locations (1p)? Explain the architecture (draw an example) and why is is called *stacked hourglass* (1p). In which part of the architecture do you compute the loss (1p)?

0 1 2 3

Representation: body joint locations as heatmaps, same dimension as an image but with high values on the position where each of the joint is likely to be found. (1p)



It is called stacked hourglass because we have several u-net (encoder-decoder which look like an hourglass) concatenated. The series of downsampling and upsampling steps helps us refine the predictions. (1p)
The loss is computed at all intermediate outputs in order to get a stronger training signal (1p).

g) You now want to scale your detection to all possible object classes in the world. Is that feasible or not? Explain your argument.

0 1

Solution: The long tail – several objects are observed far too infrequently to collect a sufficient amount of training data for them.

Additional space for solutions–clearly mark the (sub)problem your answers are related to and strike out invalid solutions.