Internship of Research Master 2 in Computer Science

Work on COCO 2018 Keypoint Detection Task

Author :
Qixiang PENG

Stage chief :
Dr. Gang YU

Host organization : Megvii Research

Secretariat - tel : 01 69 15 81 58
Email Address: alexandre.verrecchia@u-psud.fr

Contents

Abstract

This report summarizes my internship in Megvii Research: Work on COCO 2018 Keypoint Detection Task[1]. This challenge is designed to push the state of the art in multi-person pose estimation.

The topic of multi-person pose estimation has been largely improved recently, especially with the development of convolutional neural network. However, there still exist a lot of challenging cases, such as occluded keypoints, invisible keypoints and complex background.

Nowadays, two solutions are adopted widely: Bottom-Up approaches and Top-Down approaches. In this challenge, our team proposed a novel multi-stage top-down method. More specifically, each stage is based on the Resnet and the later stage will pay more attention to the harder pose samples.

Based on the proposed algorithm, we achieve state-of-art results on the COCO keypoint benchmark, with average precision at 77.8% mAp on the COCO test-dev dataset and 76.4% on the COCO test-challenge dataset, which is a 4.8% mAp relative improvement compared with 73.0% from the COCO 2017 keypoint challenge.

Our team was invited to make a presentation in ECCV2018, September 8 - 14 in Munich, Germany[2].

Keywords

COCO 2018 Keypoint Detection, human pose estimation, multi-stage, top-down.

[1] More detail about this challenge can be found in https://competitions.codalab.org/competitions/12061

[2] More detail about the conference can be found in https://eccv2018.org/

CHAPTER 1

Introduction to Company and Team

1.1 Introduction to Company

Founded in October 2011, Megvii is an Articial Intelligence company specialized in providing enterprises and developers with intelligent solutions and data services, and is dedicated to the mission of Create machines that can see and think. With the "cloud + end" system of Megvii Cloud and Megvii SensorNet as its core products, Mevgii has successfully offered solutions for over 800 enterprises in nance, security, office, real estate and other business sectors.

Megvii holds more than 350 domestic and international patents. Over seven years of development, Megvii has gathered a workforce of over 1000 people, among whom more than 70% are RD stas. The core team of Megvii is composed of top geeks who are alumni of universities like Tsinghua, Columbia, Oxford, etc., and adventurers formerly working for Google, Alibaba, Huawei and IBM. Over 80 people in Megvii have been awarded golden prizes of informatics at national and international levels. Research teams from Megvii have been and are holding the first places in more than ten international AI benchmarks.

The name Megvii is from mega vision, which means our work is concentrated on offering computer vision technologies that enable your applications to read and understand the world better. In fact, FACE++, the best product of megvii, now, is the biggest platform of face detection over the world.

Here is the link to official website: https://www.faceplusplus.com/

1.2 Introduction to Team

During the internship, I worked in Detection Team in Megvii Research. The team leader is Gang YU[1].

In general, our team is in charge of 4 main issues:

1. **Detection:** Face Detection, Pedestrian/human Detection, Vehicle/Plate Detection, General Object Detection, Object Detection in Video, 3D Object detection (combined with Point Cloud)

2. **Segmentation:** Semantic Segmentation, Instance Segmentation, Panoptic Segmentation, Video Segmentation, 3D Segmentation

3. **Skeleton:** Human Pose Estimation, Hand Pose Estimation

4. **Action:** Action Recognition in Video

Our team has a solid technical accumulation, especially in the detection aspect. We have the winner solution of COCO2017 Detection: MegDet [24]. From a product perspective, we built a small repo of imagenet base model for training and exploring models with less than 100M FLOPs. In addition to Detection, our skeleton solution also took the first in the COCO2017 Human Pose competition: CPN [6]. In terms of Segmentation, we also have some better work published. In addition, we have sufficient GPU resources, as well as very large internal data sets for exploring the upper-bound of various research tasks.

[1]His google scholar is: https://scholar.google.com/citations?user=BJdigYsAAAAJhl=en

CHAPTER 2

Human Pose Estimation and COCO dataset

In this chapter, several brief presentations will be given to explain the context of the human pose estimation and describe the COCO dataset.

2.1 Human Pose Estimation

Localizing body parts for human body is a fundamental yet challenging task in computer vision, and it serves as an important basis for high-level vision tasks, e.g., activity recognition [30, 32], human re-identification [33], and human-computer interaction. In generala human pose estimation model aims to predict the 2D coordinates of different human parts given a 2D human image. Achieving accurate localization, however, is difficult due to the highly articulated human body limbs, occlusion, change of viewpoint, and foreshortening.

Classical approaches tackling the problem of human pose estimation mainly adopt the techniques of pictorial structures [7] or graphical models [5]. More specifically, the classical works [1, 10, 16, 27] formulate the problem of human keypoints estimation as a tree-structured or graphical model problem and predict keypoint locations based on hand-crafted features. Recent works [11, 15, 22, 31] mostly rely on the development of convolutional neural network(CNN) [13, 18], which largely improve the performance of pose estimation. And the rest of report also focuses on the solution based on CNN.

Nowadays there exists two main topics in human pose estimation: single person pose estimation and multi-person pose estimation. Obviously, multi-person is more challenging than single person pose estimaion but single person is the fundamentation for multi-person pose estimation, as shown in Figure.1.

2.1.1 Single person Pose Estimation

The works based on CNN usually adopts two methods: regress the coordinates of keypoints directly and regress the cofidence score map of keypoints. Toshev *et al.* firstly introduce CNN to solve pose estimation problem in the work of DeepPose [29], which proposes a cascade of CNN

(a) Single person pose estimation

(b) Multi-person pose estimation

Figure 1: Human Pose Estimation. (a) is the example of single person pose estimation, only one person in a image. (b) is the example of multi-persion pose estimation. An image includes several peoples. We need to detect all the keypoints and group them into the right person ID.

keypoint coordinate regressors to deal with pose estimation. Tompson *et al.* [28] attempt to solve the problem by predicting heatmaps of keypoints using CNN and graphical models. Using heatmap as the supervised label can provide more robust information, hence recently most of work focus on predicting heatmaps. For example, latest works such as Wei *et al.* [31] and Newell *et al.* [22] show great performance via generating the score map of keypoints using very deep convolutional neural networks.

2.1.2 Multi-person Pose estimation

Multi-person pose estimation is gaining increasing popularity recently because of the high demand for the real-life applications. However, multi-person pose estimation is challenging owing to occlusion, various gestures of individual persons and unpredictable interactions between different persons. The approach of multi-person pose estimation is mainly divided into two categories: bottom-up approaches and top-down approaches.

2.1.2.1 Bottom-Up Approaches

Bottom-up approaches [15, 21, 25] directly predict all keypoints at first and assemble them into full poses of all persons. DeepCut [25] interprets the problem of distinguishing different persons in an image as an Integer Linear Program (ILP) problem and partition part detection candidates into person clusters. Then the final pose estimation results are obtained when person clusters are combined with labeled body parts. DeeperCut [15] improves DeepCut [25] using deeper ResNet [13] and employs image-conditioned pairwise terms to get better performance. Zhe Cao *et al.* [3] map the relationship between keypoints into part affinity fields (PAFs) and assemble detected keypoints into different poses of people. Newell *et al.* [21] simultaneously produce score maps and pixel-wise embedding to group the candidate keypoints to different people to get final multi-person pose estimation.

2.1.2.2 Top-down Approaches

Top-down approaches [12, 14, 23] interpret the process of detecting keypoints as a twostage pipeline, that is, firstly locate and crop all persons from image, and then solve the single person pose estimation problem in the cropped person patches. Papandreou *et al.* [23] predict both heatmaps and offsets of the points on the heatmaps to the ground truth location, and then uses the heatmaps with offsets to obtain the final predicted location of keypoints. Mask-RCNN [12] predicts human bounding boxes first and then crops the feature map of the corresponding human bounding box to predict human keypoints. If top-down approach is utilized for multi-person pose estimation, a human detector as well as single person pose estimator is important in order to obtain a good performance.

2.1.2.3 Human detection

Human detection approaches are mainly guided by the RCNN family [8, 9, 26], the upto-date detectors of which are [12, 19]. These detection approaches are composed of two-stage in general. First generate boxes proposals based on default anchors, and then crop from the feature map and further refine the proposals to get the final boxes via R-CNN network.

2.2 COCO dataset

COCO [20] is a large-scale object detection, segmentation, and captioning dataset[1]. For different tasks like object detection, keypoint detection, stuff segmentaion, etc, COCO dataset can be seperated as different sub-datasets. Here we only introduce the human keypoint dataset.

2.2.1 Dataset statistic

The coco keypoint dataset was split up into 3 parts: train-set, validation-set, test-set. And test-set is further divided into 2 parts: test-dev for evaluating the model in the usual time and test-challenge for evaluating the model during the competition. It should be noted here that only labels of train-set and validation-set are available. In order to extend the train-set, we extracted a small subset of the validation-set called mini-val for evaluating our model, and the rest of validation-set and train-set merged into one larger train-set called train-val.

	train-set	validation-set	test-dev	test-challenge	train-val	mini-val
images numbers	39935	19010	20000	20000	56599	2346
instances numbers	105968	50197	unknown	unknown	149813	6352
keypoints numbers	1161667	548831	unknown	unknown	1642283	68215

Table 1: The statistics of datasets

2.2.2 Annotation format

Each person(instance) annotations contains a series of fields. **1.** id: Indicates the global id of this instance. **2.** image_id: Indicates which images this person belongs to. **3.** bbox: Indicates the location of this person in the image. **4.** keypoints: A length 3*17 array, indicates the details of the keypoints. Each keypoint has a 0-indexed location x,y and a visibility flag v defined as v=0: not labeled (in which case x=y=0), v=1: labeled but not visible, and v=2: labeled and visible. **5.** s: Indicates the square root of the object segment area.

There are 17 keypoints: 0: nose, 1: left eye, 2: right eye, 3: left ear, 4: right ear, 5: left shoulder, 6: right shoulder, 7: left elbow, 8: right elbow, 9: left wrist, 10: right wrist, 11: left hip, 12: right hip, 13: left knee, 14: right knee, 15: left ankle, 16: right ankle.

```
annotation{
  "id" : int,
  "image_id" : int,
  "bbox" : [x, y, weight, height],
  // x, y is the coordinate of the top-left point of the bbox.
  "keypoints" : [x1, y1, v1, ...]
  "s" : float
}
```

2.2.3 Metric

The core idea behind evaluating keypoint detection is to mimic the evaluation metrics used for object detection, namely average precision (AP) and average recall (AR) and their variants. At the heart of these metrics is a similarity measure between ground truth objects and predicted

[1]More details about COCO dataset can be found in http://cocodataset.org/

objects. In the case of object detection, the IoU(Intersection over Union) serves as this similarity measure (for both boxes and segments). Thesholding the IoU defines matches between the ground truth and predicted objects and allows computing precision-recall curves. To adopt AP/AR for keypoints detection, we only need to define an analogous similarity measure. Here an object keypoint similarity (OKS) which plays the same role as the IoU.

As mentioned before, for each object, ground truth keypoints have the form $[x_1, y_1, v_1, ..., x_k, y_k, v_k]$, where x,y are the keypoint locations and v is a visibility flag defined as v=0: not labeled, v=1: labeled but not visible, and v=2: labeled and visible. Each ground truth object also has a scale s which is defined as the square root of the object segment area.

For each object, the keypoint detector must output keypoint locations and an object-level confidence. Predicted keypoints for an object should have the same form as the ground truth: $[x_1, y_1, v_1, ..., x_k, y_k, v_k]$. However, the detector's predicted v_i are not currently used during evaluation, that is the keypoint detector is not required to predict per-keypoint visibilities or confidences.

The object keypoint similarity (OKS) is defined as:

$$OKS = \frac{\sum_i [exp(-d_i^2/2s^2k_i^2)\delta(v_i > 0)]}{\sum_i [\delta(v_i > 0)]} \text{.....(1)}$$

The d_i are the Euclidean distances between each corresponding ground truth and detected keypoint and the v_i are the visibility flags of the ground truth (the detector's predicted v_i are not used). To compute OKS, we pass the d_i through an unnormalized Guassian with standard deviation sk_i, where s is the object scale and k_i is a per-keypont constant that controls falloff. For each keypoint this yields a keypoint similarity that ranges between 0 and 1. These similarities are averaged over all labeled keypoints (keypoints for which $v_i > 0$). Predicted keypoints that are not labeled ($v_i = 0$) do not affect the OKS. Perfect predictions will have OKS=1 and predictions for which all keypoints are off by more than a few standard deviations si will have OKS 0. The OKS is analogous to the IoU. Given the OKS, we can compute AP and AR just as the IoU allows us to compute these metrics for box/segment detection.

The final metric for COCO keypoint dataset is shown as below:

```
Average Precision (AP):
  AP                % AP at OKS=.50:.05:.95 (primary challenge metric)
  AP^OKS=.50        % AP at OKS=.50 (loose metric)
  AP^OKS=.75        % AP at OKS=.75 (strict metric)
AP Across Scales:
  AP^medium         % AP for medium objects: 32^2 < area < 96^2
  AP^large          % AP for large objects: area > 96^2
Average Recall (AR):
  AR                % AR at OKS=.50:.05:.95
  AR^OKS=.50        % AR at OKS=.50
  AR^OKS=.75        % AR at OKS=.75
AR Across Scales:
  AR^medium         % AR for medium objects: 32^2 < area < 96^2
  AR^large          % AR for large objects: area > 96^2
```

Figure 2: The final metrics for COCO keypoint task.

CHAPTER 3

Solution to COCO 2018 keypoint task

Similar to [12,23], our algorithm adopts the top-downpipeline: a human detector is first applied on the image to generate a set of human bounding-boxes and detailed localization of the keypoints for each person can be predicted by a single-person skeleton estimator. In addition, I use GAN to generate new train data. The extended train-set makes model more robust.

3.1 Human detector

We adopt the state-of-art object detector algorithms based on FPN [19]. ROIAlign from Mask RCNN [12] is adopted to replace the ROIPooling in FPN. To train the object detector, all eighty categories from the COCO dataset are utilized during the training process but only the boxes of human category is used for our multi-person skeleton task.

In our pipeline, we need a object detector which can find out the candidate human bounding-boxes as many as possible, not the one which can predict the human bounding-boxes very accurately. In another words, we need a object detector with high recall, not high precision.

3.2 Human Pose Estimator

Before starting the discussion of our algorithm, I first briefly review the design structure for the single person pose estimator based on each human bounding box.

3.2.1 Stacked Hourglass

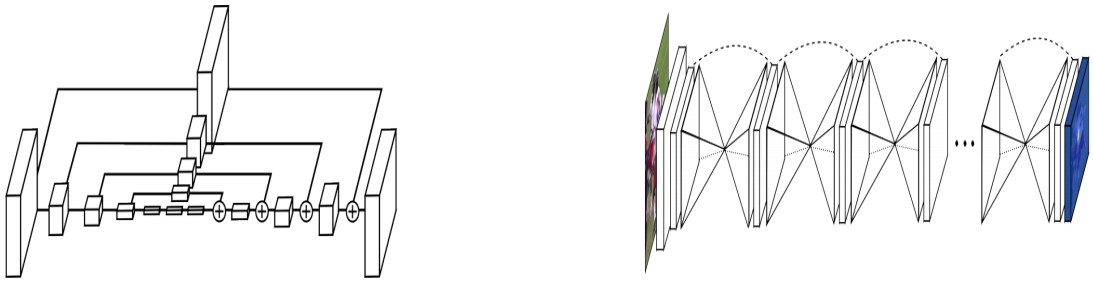
Stacked hourglass [22], which is a prevalent method for pose estimation, stacks eight hourglasses which are down-sampled and up-sampled modules with residual connections to enhance the pose estimation performance.

The design of the hourglass is motivated by the need to capture information at every scale. While local evidence is essential for identifying features like faces and hands, a final pose estimate requires a coherent understanding of the full body. The persons orientation, the arrangement of their limbs, and the relationships of adjacent joints are among the many cues that are best recognized at different scales in the image. The hourglass is a simple, minimal design that has the capacity to capture all of these features and bring them together to output pixel-wise predictions.

Then the network is handled further by stacking multiple hourglasses end-to-end, feeding the output of one as input into the next. This provides the network with a mechanism for repeated bottom-up, top-down inference allowing for reevaluation of initial estimates and features across the whole image. The key to this approach is the prediction of intermediate heatmaps upon which a loss can be applyed. Predictions are generated after passing through each hourglass where the network has had an opportunity to process features at both local and global contexts. Subsequent hourglass modules allow these high level features to be processed again to further evaluate and reassess higher order spatial relationships.

3.2.2 Cascaded Pyramid Network

CPN [6] thought that stacking two hourglasses is sufficient to have a comparable performance compared with the eight-stage stacked hourglass module. Hence, CPN involves two sub-networks: GlobalNet and RefineNet.



(a) One Hourglass (b) Stacked Hourglasses

Figure 3: Hourglass Module. (a): The illustration of a single hourglass module. Each box in the figure corresponds to a residual module. The number of features is consistent across the whole hourglass. (b) The illustration of multiple stacked hourglass modules which allow for repeated bottom-up, top-down inference

GlobalNet is based on the ResNet backbone. The last residual blocks of different conv features conv25 are denoted as C_2, C_3, ..., C_5 respectively. 3 3 convolution filters are applied on C_2, ..., C_5 to generate the heatmaps for keypoints. The shallow features like C_2 and C_3 have the high spatial resolution for localization but low semantic information for recognition. On the other hand, deep feature layers like C_4 and C_5 have more semantic information but low spatial resolution due to strided convolution (and pooling). Thus, usually an U-shape structure is integrated to maintain both the spatial resolution and semantic information for the feature layers. GlobalNet can effectively locate the keypoints like eyes but may fail to precisely locate the position of hips. The localization of keypoints like hip usually requires more context information and processing rather than the nearby appearance feature.

Based on the feature pyramid representation generated by GlobalNet, a RefineNet to explicitly address the hard keypoints was attached. In order to improve the efficiency and keep integrity of information transmission, the RefineNet transmits the information across different levels and finally integrates the informations of different levels via upsampling and concatenating as HyperNet [17]. In addition, more bottleneck blocks are stacked into deeper layers, whose smaller spatial size achieves a good trade-off between effectiveness and efficiency.

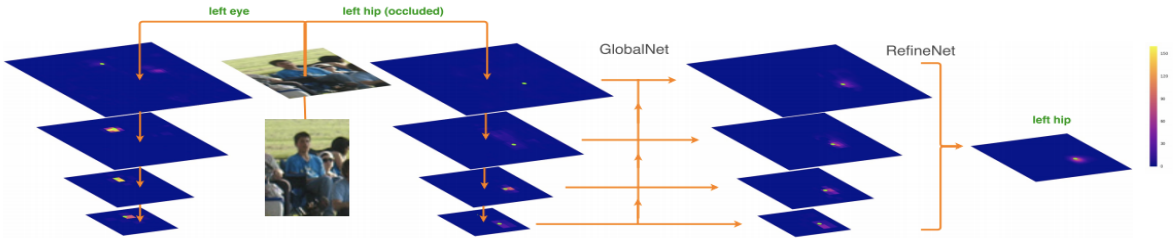


Figure 4: Cascaded Pyramid Network. Output heatmaps from different features. The green dots means the groundtruth location of keypoints. GlobalNet handles the easy case like left eye well. RefineNet integrates more information to locate the hard case, like occluded left hip.

3.2.3 Our Pose Estimator

Motivated by the works [6, 22] described above, we propose an effective and efficient network to address the problem of pose estimation. As shown in Figure 5, our model is a multi-stage network.

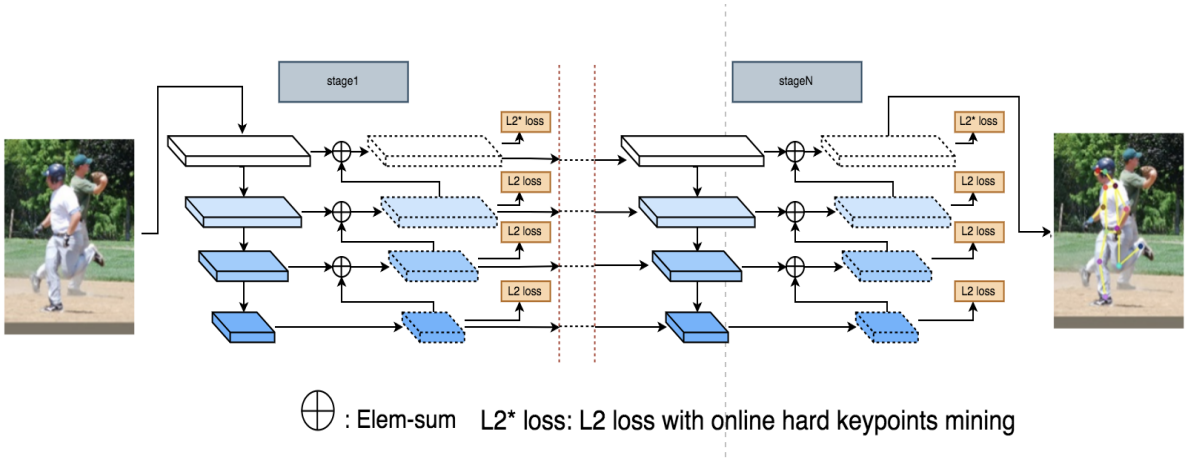


Figure 5: Our multi-stage network. Each stage is a U-shape structure based on the variant resnet. Different stages focus on diffferent poses. The later stage studies the harder pose.

3.2.3.1 Singe Stage

Every single stage is a U-shape structure.

As we all known, classification task needs semantic information and spatial information is necessary for location task. The keypoint task contains both classification and location task because we need to not only classifer different body parts but also give precise coordinates. In other words, we need both semation information and spatial information. This is why we adopt U-shape structure.

The U-shape structure is based on the ResNet backbone. The last residual blocks of different conv features conv25 are denoted as C_2, C_3, ..., C_5 respectively, like the cuboids with full line in Figure 5.

In the down-sampling procedure, the resolution of feature map zooms out 5 times(conv1 5). For example, if the size of input was $256 * 192$, the lowest resolution could be $8 * 6$. Hence, every pixel in $8 * 6$ size feature map owns a huge receptive field and the $8 * 6$ size feature map has high-dimensinal semantic information but lacks spatial information. On the contrary, the high solution feature map has more spatial information and less semantic information.

And then the up-sampling procedure will be passed. With a coarser-resolution feature map, we upsample the spatial resolution by a factor of 2 (using bilinear interpolation for accuracy). The upsampled map is then merged with the corresponding down-sampling map (which undergoes a $1 * 1$ convolutional layer to reduce channel dimensions) by element-wise addition. This process is iterated until the finest resolution map is generated. To start the iteration, we simply attach a 11 convolutional layer on C_5 to produce the coarsest resolution map. Afterwards, we append a $1 * 1$ convolution on each merged map to generate the final feature map, which is to reduce the aliasing effect of upsampling and to reduce the dimensions(numbers of channels) to 256. This final set of feature maps is called $\{P_2, P_3, P_4, P_5\}$, corresponding to $\{C_2, C_3, C_4, C_5\}$ that are respectively of the same spatial sizes.

Finally, every final feature maps will pass 2 Conclusion layes($3*3$ and $1*1$) to generate the 17 dimensinal heapmaps.

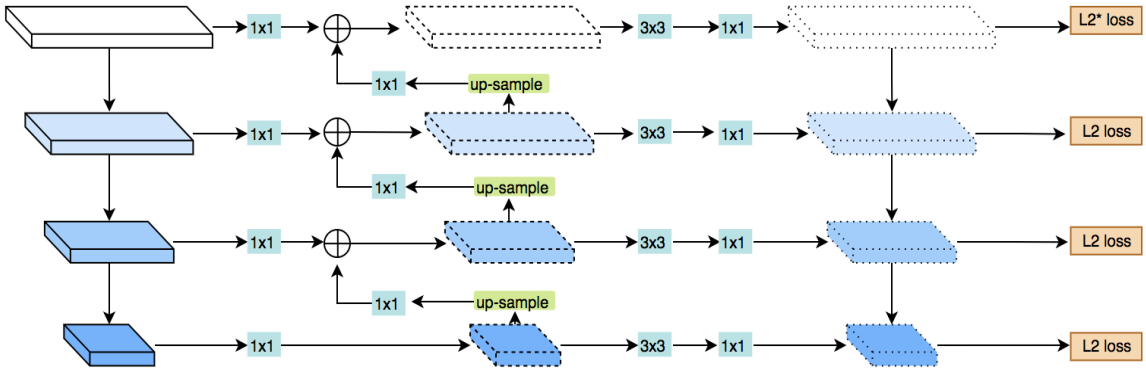


Figure 6: Details of U-shape structure for single stage

Intermediate Supervision

The U-shape architecture provides a top-down, bottom-up inference allowing for generation of heatmaps with different resolution. Hence, we can apply a loss upon the prediction of intermediate heatmaps. Predictions are generated after passing through each final feature map $\{P_2, P_3, P_4, P_5\}$ where the network has had an opportunity to process features at both local and global contexts. This is similar to other pose estimations methods that have demonstrated strong performance with multiple iterative intermediate supervision [4, 31].

Every single stage is a variant Resnet.

Nowadays, Resnet [13] is the most common basemodel in all kinds of Computer-vision task. The core idea of Resnet is the residual learning which can avoid the gradient vanishing effectively. Hence, Resnet can be designed very deeply, like resnet50, resnet101, resnet152.

$$y = F(x, \{W_i\}) + x.....(2)$$

Eqn 1 explained the residual learning. Here x and y are the input and output vectors of the layers considered. The function $F(x, \{W_i\})$ represents the residual mapping to be learned.

As mentioned before, keypoint detection task needs both semantic information and spatial information. But are they same important? We did several experiments using res50, res101 and res152 as single stage network respectively. Single stage has mAp 73.3% with res50, 73.7% with res101, 73.8% with res152. Compared with res50, res101 got 0.4% mAp gain. However, res152 only got 0.1% more mAp than res101. As shown in Figure 8, the difference better res50, res101 and res152 is the number of residual blocks in Conv4. Conv4 learns more semantic information, and increasing the complexity in Conv4 doesn't seem to work. Hence, we got a conclusion: **for keypoint detection task, spatial information is more important than semantic information.**

Then we designed a variant Resnet. In normal Res101, The number of residual block of different Conv stage is $\{3, 4, 23, 3\}$. Our variant Res101 adopts $\{4, 8, 16, 3\}$, which maintains same FLOPs as normal Res101 but adds more complexity in Conv2 and Conv3. And it has 74.1% mAp, more 0.4% mAp than normal Res101 in the case of the same complexity.

(a) One Hourglass

(b) Stacked Hourglasses

Figure 7: Residual block. (a): The residual-block for residual learning. (b) The realization of residual block in Resnet50, 101, 152

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
		3×3 max pool, stride 2				
conv2_x	56×56	[3×3, 64; 3×3, 64] ×2	[3×3, 64; 3×3, 64] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3
conv3_x	28×28	[3×3, 128; 3×3, 128] ×2	[3×3, 128; 3×3, 128] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×8
conv4_x	14×14	[3×3, 256; 3×3, 256] ×2	[3×3, 256; 3×3, 256] ×6	[1×1, 256; 3×3, 256; 1×1, 1024] ×6	[1×1, 256; 3×3, 256; 1×1, 1024] ×23	[1×1, 256; 3×3, 256; 1×1, 1024] ×36
conv5_x	7×7	[3×3, 512; 3×3, 512] ×2	[3×3, 512; 3×3, 512] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3

Figure 8: Different Resnet Architectures

3.2.3.2 Multi-Stage

Hard pose samples are really rare.

In keypont detection, the object keypoint similarity (OKS), seen in Eqn 1, is required to define positives and negatives. At beginning, we trained a one-stage model and did the failure case analyse. We found that most normal poses can be handled well, but a few hard poses like Parkour or playing skateboard are detected with low OKS. Two main factors are responsible for this: 1)hard poses are inherently challenging and 2) overfitting for normal poses during training, due to lack of hard poses. Hence, motivated by cascade r-cnn [2], a multi-stage keypoint estimator architecture is proposed to the second factor. It consists of a sequence of single estimator as mentionde before trained with increasing OKS thresholds, to be sequentially more selective against close false positives. The estimators are trained stage by stage, leveraging the observation that the output of a estimator is a good distribution for training the next higher quality estimator.

For example, if we take 3 stage architecture, the input of first stage are all samples, the input of second stage are those whose OKS is smaller than 0.8 accroding to the output of first stage, and the input of third stage are those whose OKS is smaller than 0.6 accroding to the output of second stage. The resampling of progressively guarantees that the later stage pays morr attention to the

harder pose, reducing the overfitting problem.

We don't need pre-train a model using Imagenet.

Since R-cnn [9], we always initialize our basemode by the parameters pre-trained on Imagenet, then finu-tune the model by the specific dataset. It seems like a common sense, which no one doubts. Howerver, is that really suitable for keypoint detection? **Our answer is no.**

We did several comparative experiments, Table 2 shows the results.

	3xres50	4xres50	2xres101	2xresInc101
Initialization using Gaussian	77.5	77.8	77.0	77.5
Initialization using pre-trained parameters	76.9	77.0	76.2	76.9

Table 2: comparative experiments results for initialization problem.

We can find that initialization using Gaussian is generally better than initialization using pre-trained parameters. Hence, we get a conclusion: **Imagenet task is a pure classification task, which prefers to semantic information. This is why using parameters pre-trainded on Imagenet for initialization results in a worse performance. The best solution is that we pre-train our model in a large auxiliary keypint dataset.**

CHAPTER 4

Experiments

CHAPTER 5

Conclusion

CHAPTER 6

Acknowledgement

At end of everthing, i would like to point out that i couldnt nish this internship successfully without someones, and here i give my most sincere thanks to them.

Firstly, i will express thanks to Gang YU, my team leader. Its him that taught me the detailed knowledge of computer vision and deep learing, like how to design a network , how to write the code using framework, how to train a model, etc. He also explained papers to me clearly and carefully. In fact, he leads me into the deep of CVDL domain.

Next, i would like to thank for Zhicheng WANG, my virtual team leader. During all the COCO competition, he gave me lots of help, like correcting my wrong opinions and reviewing my codes. We argued about the experiment results and make the plan together.

Finally but with same importance, my colleagues helped me a lot when i encountered some problems about software or hardware, i will always appreciate that.

Bibliography

[1] Mykhaylo Andriluka, Stefan Roth, and Bernt Schiele. Pictorial structures revisited: People detection and articulated pose estimation. pages 1014–1021, 2009.

[2] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. *arXiv preprint arXiv:1712.00726*, 2017.

[3] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. *arXiv preprint arXiv:1611.08050*, 2016.

[4] Joao Carreira, Pulkit Agrawal, Katerina Fragkiadaki, and Jitendra Malik. Human pose estimation with iterative error feedback. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4733–4742, 2016.

[5] Xianjie Chen and Alan L Yuille. Articulated pose estimation by a graphical model with image dependent pairwise relations. pages 1736–1744, 2014.

[6] Yilun Chen, Zhicheng Wang, Yuxiang Peng, Zhiqiang Zhang, Gang Yu, and Jian Sun. Cascaded pyramid network for multi-person pose estimation. *arXiv preprint arXiv:1711.07319*, 2017.

[7] Martin A Fischler and Robert A Elschlager. The representation and matching of pictorial structures. *IEEE Transactions on computers*, 100(1):67–92, 1973.

[8] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[9] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[10] Georgia Gkioxari, Pablo Arbelaez, Lubomir Bourdev, and Jitendra Malik. Articulated pose estimation using discriminative armlet classifiers. pages 3342–3349, 2013.

[11] Georgia Gkioxari, Alexander Toshev, and Navdeep Jaitly. Chained predictions using convolutional neural networks. In *European Conference on Computer Vision*, pages 728–743. Springer, 2016.

[12] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 2980–2988. IEEE, 2017.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[14] Shaoli Huang, Mingming Gong, and Dacheng Tao. A coarse-fine network for keypoint localization. In *The IEEE International Conference on Computer Vision (ICCV)*, volume 2, 2017.

[15] Eldar Insafutdinov, Leonid Pishchulin, Bjoern Andres, Mykhaylo Andriluka, and Bernt Schiele. Deepercut: A deeper, stronger, and faster multi-person pose estimation model. In *European Conference on Computer Vision*, pages 34–50. Springer, 2016.

[16] Sam Johnson and Mark Everingham. Learning effective human pose estimation from inaccurate annotation. pages 1465–1472, 2011.

[17] Tao Kong, Anbang Yao, Yurong Chen, and Fuchun Sun. Hypernet: Towards accurate region proposal generation and joint object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 845–853, 2016.

[18] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[19] Tsung-Yi Lin, Piotr Dollár, Ross B Girshick, Kaiming He, Bharath Hariharan, and Serge J Belongie. Feature pyramid networks for object detection. In *CVPR*, volume 1, page 4, 2017.

[20] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[21] Alejandro Newell, Zhiao Huang, and Jia Deng. Associative embedding: End-to-end learning for joint detection and grouping. In *Advances in Neural Information Processing Systems*, pages 2277–2287, 2017.

[22] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European Conference on Computer Vision*, pages 483–499. Springer, 2016.

[23] George Papandreou, Tyler Zhu, Nori Kanazawa, Alexander Toshev, Jonathan Tompson, Chris Bregler, and Kevin Murphy. Towards accurate multi-person pose estimation in the wild. In *CVPR*, volume 3, page 6, 2017.

[24] Chao Peng, Tete Xiao, Zeming Li, Yuning Jiang, Xiangyu Zhang, Kai Jia, Gang Yu, and Jian Sun. Megdet: A large mini-batch object detector. pages 6181–6189, 2018.

[25] Leonid Pishchulin, Eldar Insafutdinov, Siyu Tang, Bjoern Andres, Mykhaylo Andriluka, Peter V Gehler, and Bernt Schiele. Deepcut: Joint subset partition and labeling for multi person pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4929–4937, 2016.

[26] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[27] Ben Sapp and Ben Taskar. Modec: Multimodal decomposable models for human pose estimation. pages 3674–3681, 2013.

[28] Jonathan J Tompson, Arjun Jain, Yann LeCun, and Christoph Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. In *Advances in neural information processing systems*, pages 1799–1807, 2014.

[29] Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1653–1660, 2014.

[30] Chunyu Wang, Yizhou Wang, and Alan L Yuille. An approach to pose-based action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 915–922, 2013.

[31] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4724–4732, 2016.

[32] Weilong Yang, Yang Wang, and Greg Mori. Recognizing human actions from still images with latent poses. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2030–2037. IEEE, 2010.

[33] Liang Zheng, Yujia Huang, Huchuan Lu, and Yi Yang. Pose invariant embedding for deep person re-identification. *arXiv preprint arXiv:1701.07732*, 2017.