Heatmap-based Pose Estimation using Residual blocks

Doriand Petit
Master ISI
Sorbonne Université
Paris, France
doriand.petit@etu.sorbonne-universite.fr

Louis Simon
Master ISI
Sorbonne Université
Paris, France
louis.simon@etu.sorbonne-universite.fr

***Abstract*—As part of our Machine Learning for Human-Computer Interaction class, we have worked on a deep convolutional model for key points estimation. In this report, we present an architecture inspired by ResNet. The present architecture features a set of residual blocks followed by a single convolutional classifying head. Our neural networks outputs a set of 16 heatmaps, one for each key point. Predictions can then be turned into cartesian coordinates by selecting the maximum's index on each heatmaps. The developed architecture is tested on a toy data set made of simple stickman with additional noise (small shapes and background).**

Using residual blocks, we are able to extract relevant information from the input image with a larger number of layers. Due to computational limitations, we were unable to train particularly deep networks. We hypothesize that building deeper networks can greatly improve the key points estimation prediction results. We also compare our architecture with others proposed by students from our class.

***Index Terms*—Key point estimation, ResNet, Hourglass**

I. Introduction

Pose estimation is a critical field of study in computer vision. Its uses are numerous, from sports analysis to scene understanding. Precisely, it corresponds to a set of approaches made to detect human bodies in images and to mark joints as key points. This can also be extended to 3D images, with for instance, special captor such as Kinects. Finally, Pose Estimation can result in Gesture Analysis, a related field of study that tries to understand what does a movement means from one or a series of pose estimations.

For years, researchers have looked for methods to estimate human pose key points from pictures or videos. At first, classical image processing approaches were used, but nowadays, Deep Neural Networks have clearly shown the best results, as they permit to build more precise, accurate, and fast systems. They create and integrate low, middle and high-level features; the depth of a neural network can further develop the complexity of the model and of the analysis. Our approach will also feature Deep Neural Networks, and some advanced blocks and architectures will be of use, ResNet and HourGlass being the most noticeable ones.

While some Deep Learning methods often directly estimate key points coordinates, it is also possible to try predicting complete heatmaps for each key point of a pose, and the precise coordinates can be extracted from them. This can result in more accurate inference and it is also sometimes better to keep heatmaps for some particular uses, *e.g.*, interpretability. Our approach performs this heatmap extraction process to predict the key points positions.
In this report, we perform keypoints estimation on toy data set made of randomly generated RGB stickmans. Each stickman has 16 key points used for labelling. Some image from the dataset include noise such as small random circles, whose shape are similar to stickman's heads, and *Mario Bros* backgrounds.

II. Related work

Deep Pose. Classic approaches to human pose estimation had pretty constraining limitations in terms of results. Deep pose [1] marks a crucial shift as the first major paper using Deep Learning to tackle this problem. It obtained State-Of-The-Art performances and was the spearhead to a whole new generation of papers. Their approach used Convolutional Neural Networks as well as a "cascade" system, in order to refine the estimations. Another important feature provided by Deep Pose is the estimation of hidden key points, as all images do not provide a complete view of a person's pose. However, due to the nature of our own problem (2D stickmans), this particular question will not be further studied.

Open Pose. The years following the release of Deep Pose were full of pose estimations papers, some of them being not so important while others were true breakthroughs. Open Pose [2] can be counted in the second category. Apart from groundbreaking results, it also offers important features. Most of all, Open Pose is the first real-time approach to 2D human pose estimation. This is obviously a crucial addition to the field, as most uses really benefit from the possibility of working online. The approach also considers multi-person pose detection, as well as a more complete key point estimation, including body, foot and facial key points. While all these features are important improvements on human pose estimation, most of them are not relevant to our current problem, which focuses on a single stickman pose estimation.

III. Network Architecture

As a way to leverage the training process of our reasonably deep network, we utilize residual blocks, the main component of the well-known *ResNet*. We also make use of the *Hourglass* architecture which was originally designed for human pose estimation. The ResNet architecture as well as ours are described in the followings. We will also discuss the use of vectors versus heatmaps as predictions.

A. ResNet

ResNet [3] is arguably a major milestone in research on deep convolutional network. It gained its popularity after obtaining outsanding results at ILSVRC 2015. The main idea behind this architecture is to tackle the problem of vanishing/exploding gradient which is a key issue in deep learning research [4]. ResNet is able to partially solve this problem by learning a residual mapping $\mathcal{F}(x)$ from one layer to another and combining it with its input to obtain the normally expected mapping $\mathcal{H}(x) = \mathcal{F}(x) + x$. Input is then added to the residue via a identity skipping connection. Authors hypothesize that it is easier to optimize the residual mapping than the original one, thus leading to faster and more reliable training.

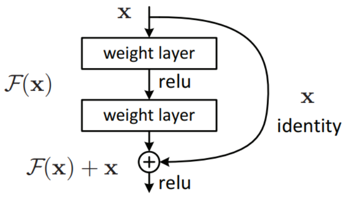
x
weight layer
relu
weight layer
$\mathcal{F}(\mathbf{x})$
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Fig. 1. Residual block (*courtesy* [3])

Formally, ResNet is built using elementary residual blocks (fig 1) which gathers two weighted layers with an activation function after the first one. A batch normalization layer is also often added after each weighted layers. In our case, weighted layers are 2D convolutional layers with a certain stride and number of filters. As dimensions of $\mathcal{F}(x)$ and x might not be the same size, *i.e.* if we want to downsample or upsample the input image or increase/decrease the number of filters, additional transformations can be applied to x, *e.g.* 1×1 convolutions.
Using this mechanism of residual blocks, one can train deeper networks than classical ones (e.g. VGG19). In their article, authors describe a set of Deep Convolutional Network with depth ranging from 18 to 152 layers. As networks gets deeper, training errors tend to decrease w.r.t networks' depth with ResNet whereas it usually increase with plain networks, i.e., architecture without residual blocks.

B. Vectors / Heatmaps

An important aspect of our approach relates to the prediction of heatmaps, from which we extract keypoints. As stated above, using heatmaps is fairly useful when dealing with multiple persons in the same scene. Moreover, the mapping between images and cartesian coordinates is not as trivial as it seems. Nonetheless, there are few ways to recover cartesian coordinates from heatmaps, and vice and versa. In order to compute the points' coordinates, we decided to keep the argmax of each heatmap as a keypoint. Actually, most of the issues we had during this project comes from these maps, whether it was technical bugs on the expected shape of a heatmap, or more Deep Learning oriented issues such as the question of ”abusive up sampling”, which we will discuss in a future part.

C. Ours

We have tried several approaches to solve the problem of keypoints estimation. Tested architectures include classical ResNet, hourglass/autoencoder and stacked hourglass.

Classical ResNet: Our first attempt of designing of ConvNet for keypoints estimation was to build a classical ResNet as described in the original article and up sample the last feature map in order to get sufficient resolution on the predictions. The ResNet part is equivalent to the truncated 18-layers described in the article. One can argue that using ResNet for heatmap prediction, i.e., a mapping from input image into a supposedly same size image, might be inappropriate with a ResNet as this network was originally designed for image classification. Last feature maps are of size 7×7 which makes it fairly tricky to produce high resolution heat maps without abusively up sampling. Resulting predictions tend to whether detect only few keypoints, *e.g.* only the head and or the hips are detected, or even equal if the stickman is to small compared to the image's size. As a way to tackle this problem of low resolution heatmap and extensive mapping, we also tried to implement hourglass/auto encoder-like architectures.

Hourglass: The main advantage of HourGlass architectures in our situation is the shape of the Neural Network's output. Indeed, the nature of the architecture being an auto-encoder, the predicted heatmaps have a satisfying shape withoutany unwanted up sampling chains which can have tricky effects on our predictions, as explained in the previous part. We thus began to develop a related model. Lots of choices had to be made, especially on the ”decoder” part, where we could use up samples in-between ”classical” convolutions for instance, or create an ”inverse residual bloc”. We decided to go for the second option, assuming that the advantages of residual blocks would also apply for our transposed residual blocks which contain a set of transposed convolutions, in the residual block's fashion. The resulting blocks performs a *safe* mappping thanks to its residual structure and doubles the size of its inputs. While we tested the addition of skip connections in our HourGlass models in the early stages of our research, just like true HourGlass architectures, we did not keep them later on in the final ones. After testing these architectures, we also decided to take the opportunity to try to use Stacked HourGlass architectures, in hope of even better results. However, because of the models' complexity, the maximum depth of the tries did not go further than 2 stacked HourGlass. Moreover, some implementations led to unexpected predictions, *e.g.*, one would only predict the position of the head.

However, the development of these ideas did not go exactly as predicted. We already mentioned that our technical limitations did not let us exploit the whole potential of residual blocks, and the same issue appeared on the training of these complex deep networks. We will see later on how different our models' results were and try to compare them between themselves and between others' architectures.

D. Issues Encountered

This project has also met its fair share of issues. The first one affected our work the most. While creating our first model's head, we decided to use Tensorflow's `Reshape` layer in order to obtain the expected shape from already given post-processing functions. However, we used it wrongly and the absence of error hid this fact from us. This resulted in nonsensical heatmaps, which we did not see for a long time.

Unfortunately, this issue was not alone. After solving this bug, we still obtained strange predictions, which was even stranger as, this time, the heatmaps looked like they would give at the very least decent keypoints. It took us some time to figure out that our predictions were good (or at least, decent), but the visualizations were not. Because of some unexpected factor, the original visualization did not always displayed our true predictions (we discovered this fact by comparing the points coordinates to the visualization), and, after modification, we finally saw some acceptable predictions, or even good ones.

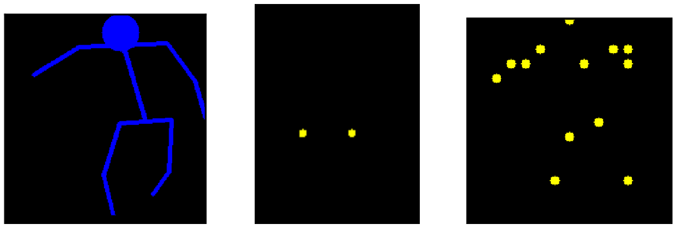

Fig. 2. Ground Truth - Visualization before Correction - After Correction

This was when we decided to finally send our final predictions only to receive similar test scores to what we obtained earlier from our bugged version. This last issue very probably comes from a small mistake of inverting our keypoints x and y coordinates. Unfortunately, it was too late to compute a last testing score set, and we will not know what could have been our best score. However, because the visualization now looks quite good, we have quite a lot of comments to do on our results and experiments.

IV. Experiments

A. Our Results

Classical ResNet: We first evaluate our simplest model containing only residual blocks and up samples. While we can't present the actual test scores (as explained in the previous part), we can still test the architecture with a validation set and some predicted heatmaps. Fig 3 shows that a very simple and shallow ResNet can give decent results, even with a very quick training. However, there are obvious limitations, as some more complex stickman's poses fail to be entirely detected. Moreover, too "easy" trainings on these too simple models can cause over-fitting problems, and very similar heatmaps between the 16 keypoints.

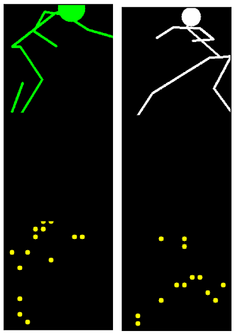

Fig. 3. Examples of Good Predictions with the simple ResNet Architecture

HourGlass: As mentioned before, the HourGlass architecture has been elected to try to obtain output heatmaps with better resolutions. Also, we took the opportunity to go for deeper networks, as learning more complex forms can give better results, especially for the more complex test datasets with non-uniform backgrounds. Unfortunately, it turned out that the decoding part was not simple as expected. The vanilla residual block downsamples by a factor 2 its input. Therefore, with have tried several strategies to upsample feature maps in the decoding part: no stride and maxPooling, and transpose convolution. Some side effects, such as extensive shape detection or detection of a limited number of key points, were observed with the first type of block. On the other hand, the transpose convolution block led to sparse heatmaps, i.e., a grid with pixels equals to zero and non-zero pixels corresponding to key points.
With have also tried to stack hourglasses to get deeper and perform a better feature extraction. Using this strategy led to slightly better results but the accuracy/training period tradeoff led us to not consider this type of architecture.
As the hourglass architecture tend to better encode, in theory, the information, the decoding part is not as simple as it seems. Moreover, this deep architecture is not as trivial to train. We hypothesize that using dropout or tuning training hyper parameters could lead to better inference.

B. Comparison between Architectures

Thanks to our colleagues' works, we were able to compare our own approach with other classical architectures used in pose estimation. It is however very important to note that each and every implementation has not been made with the same degree of precision when it comes to models' optimality. In fact, we also cannot really consider that the learning of models are equivalents, as they probably all have different image shapes, learning rates and numbers of epochs.

Architecture	**Scalar / Heatmap**	**Test Score 1**	**Test Score 2**	**Test Score 3**	**Test Score 4**
ResNet	Heatmap				
ResNet	Scalar	1.51	1.47	1.44	1.43
U-Net	Heatmap	1.54	1.52	1.52	1.51
U-Net	Scalar	2.09	2.14	2.06	2.13
SE Net	Heatmap				
SE Net	Scalar	1.27	1.20	1.24	1.25
MobileNet v2	Heatmap	1.38	1.34	1.33	1.30
MobileNet v2	Scalar	1.50	1.51	1.48	1.51
Incpetion	Heatmap	1.57	1.63	1.63	1.59
Incpetion	Scalar				
HourGlass	Heatmap	1.77	1.52	1.48	1.62
HourGlass	Scalar	1.57	1.63	1.63	1.59

First of all, we can notice that most architectures present similar test errors, across all 4 datasets. However, there are still non-negligeable variations between them, that we will try to explain.

If we look at every algorithm, a first comment can be made on the use of heatmaps. The compared test scores between their predictions and the predictions made of vectors shows that they are unable to get better results. It is once again important to note that more homogeneous models might change the results. Either way, an advantage of heatmaps is clearly the help they bring to understand the models' processes.

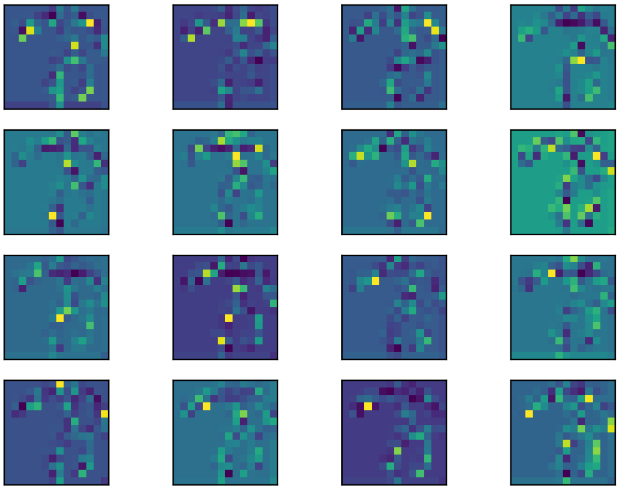

Fig. 4. Example of predicted heatmaps

We may now focus on the comparison between the various architectures, including ours, while keeping in mind the experiments' parameters' inequalities. The HourGlass architecture seems to have some struggles compared to other models, while SE Net and MobileNet v2 are better at predicting stickman poses. However, our own test score is quite low compared to others'. As we mentioned above, these scores do not come from our best predictions, which were those we visualized in this report. Hence, it is a bit difficult to compare our results with other architectures.

V. Conclusion

Using Deep Neural Networks for human pose estimation (or at least stickman pose estimation) thus gives quite good results. There are a lot of methods that can all give decent scores, whether they use special blocks or special architectures, and whether they predict heatmaps or vectors. Especially, our approach, after having solved every unexpected bugs, presents satisfying visualization. We recall that our technical limitations probably hid our model's potential, as the biggest advantage of Residual blocks are the possibility to train very deep networks, which we could not actually try. However, our ResNet-HourGlass mixed approach did not succeed very much.

While this project allowed us to explore important Deep Learning ideas, mainly ResNet but also HourGlass, we still have some regrets on our models that we would have liked to solve. Obviously, we already mentioned the shortness of our trainings, as well as our models' shallowness. Moreover, we did not have the time to really calibrate our different models, as we preferred to explore different architectures rather than try small changes such as the precise number of filters for each convolution, or the kind of padding we wanted to perform.

If we had more time, our priorities would clearly be about having a better model, with an optimal and deeper architecture, to take advantage of residual blocks, and a better training, with a lower learning rate, as well as a higher number of epochs.

References

[1] A. Toshev and C. Szegedy, "DeepPose: Human Pose Estimation via Deep Neural Networks," *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1653–1660, June 2014. arXiv: 1312.4659.

[2] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields," *arXiv:1812.08008 [cs]*, May 2019. arXiv: 1812.08008.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *arXiv:1512.03385 [cs]*, Dec. 2015. arXiv: 1512.03385.

[4] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 249–256, JMLR Workshop and Conference Proceedings, Mar. 2010. ISSN: 1938-7228.