Effect of task relatedness on MTL via MMoE

Cong Liu[21069508], Minyi Wan[21136404], Yi Fan[21076239], Yuheng Jia[21049846], Ziwen Li[21039204], Zening Li[18040301], Wenqi Wu[18017822], and Zhebin Sun[18017060]

Department of Computer Science
University College London, London, UK

1 Introduction

Humans can learn multiple tasks simultaneously, and use the knowledge learned from some auxiliary tasks to assist the learning of a target task. Inspired by such behaviour, a similar learning model emerged in machine learning field. By sharing the parameters or feature maps among auxiliary tasks, the model generalizes better on the target task [1]. This approach is called Multi-task learning (MTL). In this project, we adopted Multi-gate Mixture-of-Expert (MMoE) structure [2], which shares the learned feature maps, and models the task relationships between tasks simultaneously.
In this report, we would like to investigate how MTL improves the performance of target task and how the accuracy of auxiliary tasks and the correlation between them and target task affect the effect of MTL.
The data used in this project is a pre-processed version of The Oxford-IIIT Pet Dataset [8], which contains a set of images of cats and dogs of size 256×256. It consists of a training set (2210 samples), a validation set (738 samples), and a testing set (738 samples). The target task in this report is animal segmentation, with animal classification and animal object detection being auxiliary tasks.

2 Methods

Classification In this project, we implement ResNet18 for binary classification task [3] as it performs well in image classification tasks and is relatively efficient to train. Another key reason is that ResNet can lever the problem of information loss and attrition to some extent, and ensure that this binary classification task does not suffer from problems such as gradient disappearance or gradient explosion in a multi-task learning framework. The features of ResNet also provide a good guarantee of its stability in completing tasks and sharing information in a multi-task learning framework.
Object detection In common object detection task, complex network structure such as Faster-RCNN [4] may be adopted for its informative output. However the complexity may impedes the fusion of multiple architectures in MTL task and hence in this project, we use ResNet34 [3] for the object detection task as it is one of the most commonly used backbone neural networks for such tasks.

While preserving its network structure, the dimension of the output is adopted to 4, representing the coordinates of the top-left and bottom-right corners of the bounding box.

Masking based Segmentation U-net is used for the segmentation task for its good performance and sample-efficiency [5]. We adopt the architecture as shown in paper [5]. For this particular task, we use padding convolutions to preserve the size and add a Batch Normalization (BN) before ReLU in each sampling step, which is different from the original paper.

Multi-gate Mixture-of-Expert (MMoE) We applied our multi-task learning by performing the structure developed by [2]. The structure consists of two parts in general, i.e. Expert and Tower. Networks before the parameter sharing are defined as experts, and networks after that are towers. The output after an expert is given by the specific trainable network, Gate, which also takes raw image data as input to extract the feature map so that each model can select parameters properly. Furthermore, context gating, as described in [7], is used in Gate structure, aiming to provide a explicit learnable structure of weights. The structure of MMoE is shown in Fig. 1.

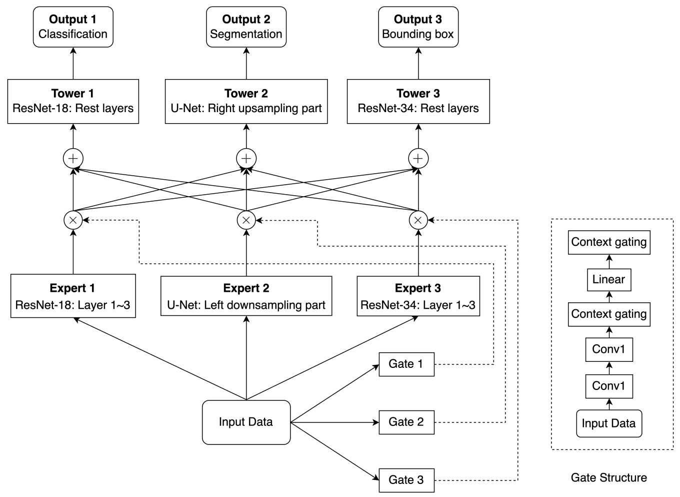

Output 1 Classification
Output 2 Segmentation
Output 3 Bounding box
Tower 1 ResNet-18: Rest layers
Tower 2 U-Net: Right upsampling part
Tower 3 ResNet-34: Rest layers
Expert 1 ResNet-18: Layer 1~3
Expert 2 U-Net: Left downsampling part
Expert 3 ResNet-34: Layer 1~3
Gate 1
Gate 2
Gate 3
Input Data
Context gating
Linear
Context gating
Conv1
Conv1
Input Data
Gate Structure

Fig. 1: Left:Structure of MMoE — Right:Structure of Gate

3 Experiments

3.1 Baseline

U-net is naturally adopted as the baseline to the targeted segmentation task. During the training process, the data loader sequentially loads one batch of data at one time to our U-net model. Cross-Entropy is used as loss function and Adam is used as optimizer with learning rate $lr = 5 \times 10^{-5}$ for our training purpose. We train the model for 100 epochs and after each epoch we evaluate the performance of model using IoU metric.

3.2 MTL: MMoE

In MTL, we apply the same training data as used in baseline to training our model for 100 epochs. We use Cross-Entropy Loss for segmentation, Smooth L1-Loss for object detection and Binary Cross-Entropy Loss for classification. Due to the difference on parameter scale between each task, we use the same optimizer and learning rate as baseline for segmentation and classification part, while setting the learning rate to 1×10^{-4} to update the parameters inside "Expert" and "Tower" of object detection. As each task has a pair of "Expert" and "Tower", it is part of design how to separate a network into "Expert" and "Tower". Our separation is shown in Fig. 1. While we need to sum the outputs of three "Experts" with learnable weights, the output dimension of segmentation is different from that of the other two tasks. To solve this problem, we use a convolutional layer as a "dimension-converter" to convert the output of each "Expert" for compatible summation. Although our model has three outputs, only one loss is obtained by applying a weighted sum of all three "sub-loss" for each task. Each loss has uniformly distributed weight, i.e. in our case, the loss weight is 1/3 across all tasks. Experiments have been conducted to search for the optimal setting of weights for the three losses however no major improvement is observed using different setting. The model has been tested on the test dataset.

3.3 Ablated version of MTL

We perform two experiments on ablation of MTL, i.e. segmentation with classification and segmentation with object detection. Both combinations have two "Experts" and "Towers" for segmentation and another task respectively. Therefore, we only have two "Gates" from two tasks. We use the same loss function, optimizer, and learning rate settings as described in previous section, aiming to control other variables for comparison. Both models are trained for 100 epochs and tested on the test dataset.

3.4 Open-ended Question

How can we identify task relatedness and further improve the performance of related tasks? As shown in section 4, classification and object detection tasks are improved by MTL, while segmentation is impeded. It is intuitive to explain this as a result of sharing parameters. Hence we would like to figure out the impact that a task has on another under MTL framework. Intuitively, more sharing among tasks will improve the performance of tasks that benefit from MTL, while harming those that are "infected" by MTL. In the original MMoE, sharing only happens at a specific stage, e.g. the third layer of ResNet18 for classification, which helps the classification to generalize better. Now we propose sharing twice in MMoE, i.e. increasing the level of sharing to investigate the above conjecture. With more sharing, we can judge the relevance of general tasks by observing changes in task-specific performance. Furthermore, the performance of related tasks may be improved by enhancing the interaction between these tasks, which is the original motivation of multi-task learning. In our implementation, we have additional sharing after the end of the second layer of ResNet18 for classification, the end of the second layer of ResNet34 for object detection, and the first downsampling of U-Net for segmentation.

4 Result

Table 1: Results on test set. IoU is used as the metric for object detection and segmentation task; Accuracy is used for classification. In particular, we focus on the performance of **segmentation** since it is the target task.

		Method	Task	Metric on test set
Baseline		ResNet18	classification	0.737
		ResNet34	object detection	0.652
		U-Net	**segmentation**	0.813
MTL		ResNet18+ ResNet34+ U-Net+MMoE	classification	0.900
			object detection	0.702
			segmentation	0.789
Ablation	-cls	ResNet34+ U-Net+MMoE	object detection	0.647
			segmentation	0.764
	-bbox	ResNet18+ U-Net+MMoE	classification	0.894
			segmentation	0.818
OEQ		ResNet18+ ResNet34+U-Net +MMoE multi-sharing	classification	0.911
			object detection	0.718
			segmentation	0.785

Baseline and MMoE. From Table 1, the high accuracy and IoU show satisfying performance of our baseline models. From Fig. 2, MTL hurts the performance of segmentation, while MTL improves the performance of classification and object detection. This observation implies that segmentation does not benefit from object detection and classification, or at least one of them. However, object detection and classification may help each other, or they both benefit from segmentation. To control variables, we implement ablation experiments to further investigate the task relatedness between segmentation and the other two tasks.

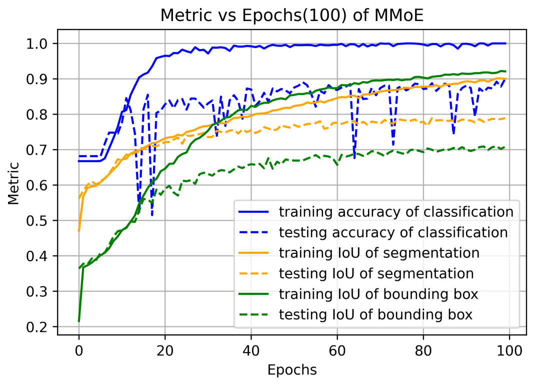
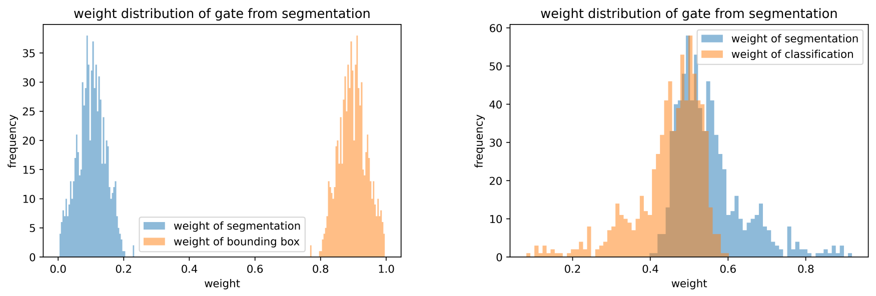


Fig. 2: Epochs(100) vs Metric of MMoE

Ablation. After removing the classification task, the IoU of the segmentation task drops from 0.789 for MTL to 0.764. Compared with the baseline performance, the IoU decreases even more, as shown in Table 1. The consistent decrements in the segmentation task in this ablation experiment reveals a conjecture that learned features in object detection may impair the main task, whereas the wider decline in the absence of the classification task may be because it reduces this impairment due to the object detection task. On the other hand, after removing the object detection task, the IoU of the segmentation task increases to 0.818, which is even higher than that of the baseline. Hence, this result to some extend validates our conjecture that the classification task helps the segmentation task, while the object detection task harms the segmentation task. In addition, the target segmentation task also affects two auxiliary tasks.

From the above observation, we reason that there may be a relatedness between different tasks, as described in [2]. Tasks with strong relatedness may boost performance for each other, while tasks with weak relatedness may weaken that for each other. To further verify the relatedness, we plot distributions of gates in ablation experiments. From Fig. 3a, in the ablation experiment deleting classification task, the object detection weight distribution of gate from the segmentation barely overlaps with the segmentation weight distribution. On the other hand, as shown in Fig. 3b, after removing the object detection task, two weight distributions overlap a lot. This relatedness of weight distributions may reflect the relatedness of different tasks to some extent.

MMoE multi-sharing. Previously we state that object detection cannot benefit from segmentation and vice versa. Classification and segmentation are helping each other to generalize better. From the results of MMoE and ablation studies, as shown in Table 1, we can see further improvements in classification and object detection performance after we increase the level of sharing between three tasks. While the performance of segmentation drops from 0.789 (single-sharing MMoE) to 0.784, which further verifies the above reasoning that, classification



(a) Without classification (b) Without object detection

Fig. 3: Weight distributions of gate in two ablation experiments

benefits from segmentation. Also, it helps the object detection to generalize better. The segmentation task is further "infected" by sharing at the beginning of the network, which can cause performance degradation. Thus, we can conclude that more sharing between related tasks boosts the performance of both tasks, while it may be detrimental for unrelated tasks.

5 Discussion

Note that more identical experiments can be implemented to derive more solid results. However, based on our GPU (Tesla P100), one epoch spends around 10 minutes to complete. Although we can verify the task relatedness by doing experiments as we did in this paper and further choose related tasks to apply MTL structure, to repeat it multiple times is time and computational consuming. Besides, MMoE uses gate to control the level of "transfer" between tasks, it is not enough since harm will be brought about more or less by the non-zero value provided by gate. For future work, we are interested in adding the intermediate output(s) of pre-trained main-task network to the sharing(s) of MMoE, aiming to provide an "Attention" to MMoE that, whenever there are harms happening on specific task, MTL structure should pay attention to the original parameters in single task network instead of MTL learned parameters, thus improving the performance of main-task without having any prior knowledge on task relatedness of a specific scenario.

6 Conclusion

We adopt MMoE to learn classification and object detection simultaneously with the target task segmentation. Model quality is close-related to the relatedness of sub-tasks. We initially prove that the segmentation task performance is not well supported by auxiliary tasks because of low-correlation between auxiliary tasks and the target task.

References

1. Ruder, S. (2017). An overview of multi-task learning in deep neural networks. arXiv preprint arXiv:1706.05098.
2. Ma, J., Zhao, Z., Yi, X., Chen, J., Hong, L., and Chi, E. (2018). Modeling Task Relationships in Multi-task Learning with Multi-gate Mixture-of-Experts. Proceedings Of The 24Th ACM SIGKDD International Conference On Knowledge Discovery and Data Mining. https://doi.org/10.1145/3219819.3220007
3. He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).
4. Ren, S., He, K., Girshick, R. and Sun, J. (2017). Faster R-CNN: Towards Real-Time Object object detection with Region Proposal Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(6), pp.1137–1149.
5. Ronneberger, O., Fischer, P., Brox, T. (2015, October). U-net: Convolutional networks for biomedical image segmentation. In International Conference on Medical image computing and computer-assisted intervention (pp. 234-241). Springer, Cham.
6. Jordan, J. (2018). Evaluating image segmentation models. Jeremy Jordan, 30. https://www.jeremyjordan.me/evaluating-image-segmentation-models/
7. Miech, A., Laptev, I., and Sivic, J. (2017). Learnable pooling with context gating for video classification. arXiv preprint arXiv:1706.06905.
8. Parkhi, O. M., Vedaldi, A., Zisserman, A., and Jawahar, C. V. (2012, June). Cats and dogs. In 2012 IEEE conference on computer vision and pattern recognition (pp. 3498-3505). IEEE.