An Exploration of Multi-task Learning Approaches for Animal Segmentation

Xifong Christian, Daniel Dosti, Zizhen Fu, Aleksandar Milosavljevic, Joan Rossello Bover, Yicheng Wang, and Xiyan Yang

University College London, London, UK

1 Introduction

Multi-task learning (MTL) is a sub-field of machine learning, which aims to improve the performance of the target tasks by learning multiple tasks either sequentially or simultaneously [1][2].

In the sequential learning model, the parameters of the target task and the auxiliary tasks are shared. During the training, each task is completed one after the other[2]. Similar to the principle of sequential learning, simultaneous learning can be achieved by completing multiple tasks in a single training set. The difference is that each task runs in parallel[1], and it usually applies the hard parameter sharing in the auxiliary tasks such that the parameters in auxiliary tasks are partly the same.

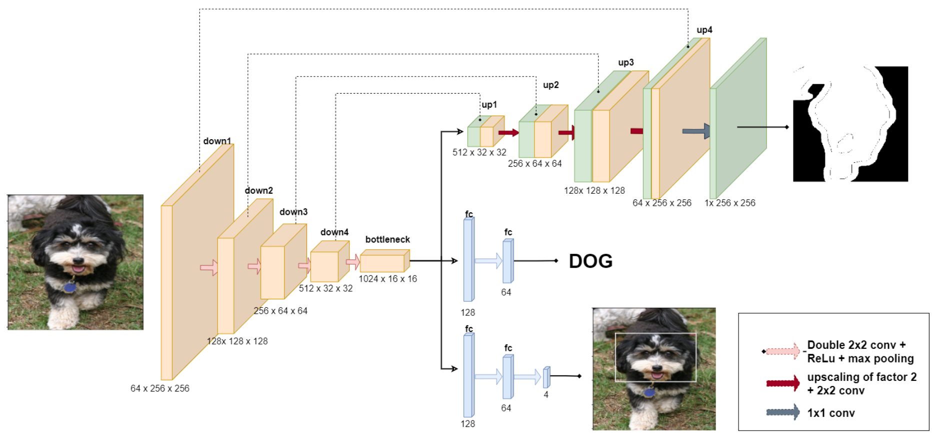
The purpose of this project is to implement and evaluate MTL models for the target task of animal segmentation. Our group has decided to take the simultaneous learning approach. Simultaneous learning can use both hard and soft parameter sharing, which can provide us with the flexibility to design the network according to different purposes. Furthermore, comparing sequential learning to simultaneous learning, the former has potential for negative transfer[3], while the latter reduces the chance of over-fitting[4].

Table 1: Summary of the models

		Model Name	Model Architecture	Model Description
Baseline Model		base	U-net	Target task (segmentation)
MTL studies		mtl	U-net	Target task, Classification, Bounding Boxes
Ablation studies	Exp 1	mtl-lessdepth	U-net with less depth	Same as mtl
	Exp 2	mtl-class	U-net	Target task, Classification
		mtl-bbox	U-net	Target task, Bounding Boxes
Open-ended question		mtl-rec	U-net	Target task, Classification, Reconstruction

In our study, we based our models on the U-net architecture. Table 1 shows the summary of our different models. Together with the target task, we built our MTL model with classification and bounding boxes. For ablation studies, we conducted two experiments as shown. For further studies, we investigated and built our model based on this research question: How does adding reconstruction task influence our MTL model?

Fig. 1: UNet-Multi Visualization



2 Methodology

a) Architecture of Multitask learning
In this report we use a network structure for simultaneous learning inspired from Psi-net[5]. It is based on the U-net. The main difference between them is that U-net[6] has one encode for one decode, whereas Psi-net has one encode for three decodes, with each decode process corresponding to one task. They are similar in structure and are juxtaposed. See Figure 1 for a visualisation.
U-Net style decoders retain the global spatial context of our images, which makes them ideal for the reconstruction and segmentation tasks. The encoding (a batch_size x 1024 x 16 x 16 tensor) retains local spatial information, so the decoders for box detection/classification are dense layers with 128, 64 and 4/1 neurons, using ReLU activations.

b) Training strategy and hyperparameters of MLT
We use a training strategy called early stopping[7]. We divide the dataset into a training set and a validation set. The rationale is that the training and validation errors are synchronized and in most cases the training and validation errors can be minimized simultaneously. However, our training loss may fluctuate, so that we cannot accurately determine whether to stop training.
To solve this problem, we use a patience of 5, which means we stop training if there is no improvement of validation loss during 5 consecutive epochs.
In addition to patience, we use a learning rate of 10^{-4} and a batch size of 5.

c) Regularization of MLT
The loss equation for our MTL model is a linear combination of three losses:
$\mathcal{L}_{total} = \lambda_1 \mathcal{L}_{mask} + \lambda_2 \mathcal{L}_{binary} + \lambda_3 \mathcal{L}_{bounding}$
where λ_1, λ_2, λ_3 are scaling factors, $\mathcal{L}_{mask}, \mathcal{L}_{binary}$ are binary cross-entropy loss for segmentation and binary classification tasks respectively, while $\mathcal{L}_{bounding}$ is

the Mean Square Error loss for the box bounding task.

d) Network of baseline model
The architecture of our baseline model is U-net, which has only one encode and decode process, which corresponds to the task of segmentation. The structure and parameters of the encoder and decoder are the same as those of the MTL network. The similarity between them helps us to focus on the impact of different network structures on the learning target task, and their structures are also distinctly different, which makes for an interesting comparison.

e) Network of ablation model
The architecture of the ablation model differs from that of psi-net in that it has one less layer of encoding and decoding than psi-net. This design allows us to explore the effect of the depth and complexity of the deep learning network on the learning results.
In addition to this, another set of ablation models was developed to explore the impact of the auxiliary tasks. The two models included in this study discarded the binary classification and bounding box auxiliary tasks respectively. However, their construction and depth were kept constant.

f) Further studies
To measure the effects of adding a reconstruction task to the MTL model, we also recur to filter visualisation and feature maps to identify in what way the network's learning process changes to yield such results.
Neural networks are often referred to as a 'black box', since their learning process towards predicting an output is usually not interpretable. However, the filters that are applied to the image in each convolutional layer can be extracted and used to generate feature maps that show the information that the network tries to identify in each layer, such as edges, patterns, or entire shapes.
We compare the 10 feature maps that result from applying the convolutional filters in the encoder part of the psi-net, between the *mtl-class* and *mtl-rec* models.

g) Learning loss weights
Until now, we've only considered total loss to be the simple sum of losses for each task with fixed weights. As indicated in [8], we can adopt a more flexible approach, by trying to learn optimal weight for the losses. The loss function that we are going to use is:
$\mathcal{L}_{total} = \frac{1}{\sigma_1^2}\mathcal{L}_{mask} + \frac{1}{\sigma_2^2}\mathcal{L}_{binary} + \frac{1}{2\sigma_3^2}\mathcal{L}_{reconstruction} + log(\sigma_1\sigma_2\sigma_3)$
Here, the σ parameters represent observation noise parameters. In essence, the contribution of $\mathcal{L}_i$ will be decreased by large values of σ_i, and will be increased by small values of σ_i. The last term plays the role of regularization for the σ parameters. We train our network to predict $s_i = log(\sigma_i)$, in addition to other network parameters. This is more numerically stable than trying to predict σ_i directly as it's not necessary for s_i to be positive, unlike σ_i. The mtl-rec model was run with the following initialization $s_1 = s_2 = 0, s_3 = 4$.

3 Experiments

Because our dataset was of HDF5 format, we passed data to our models using a generator that iterates through the indices of the HDF5 files, yielding batches of images alongside their corresponding labels, masks, and bounding boxes.

In our experiments, we built six models according to table 1. The network structures are described in the methodology section. For training, we implemented our loss function respectively according to 2c and 2g. Furthermore, we introduced our training strategies of early stopping and dividing dataset into a training and validation set, which are justified in 2b. For standardization, each model was trained with the same hyper-parameters: patience of 5, learning rate of 10^{-4}, batch size of 5, and epochs of 20, as described in 2b.

To quantitatively and statistically compare the performance and results between our models, we introduced three different kinds of metrics - pixel accuracy[9], Intersection-Over-Union (IoU) score[10] and Dice Coefficients[11]. Although pixel accuracy measures the percentage of correctly classified pixels, one potential concern with it is class imbalance[10]. To avoid biased results, we also used the IoU score and the Dice Coefficients.

Last but not least, to visualize the performance of our models, we have also plotted graphs of the metrics against epoch number, shown in the appendix.

4 Results

Overall, our experiments have shown a strong, difficult to beat baseline performance; *base* was out-performed robustly on segmentation only by the *mtl-rec* model. For this MTL setup, weighting segmentation and classification losses by 0.8 and 0.2 respectively produced significant improvements across all three segmentation metrics, whereas if using uniform weighting, *base* still performs better. This is not surprising as it would be expected that reducing primary task losses should be prioritised. The same effect is observed for the *mtl-class* models, though in this case, the lower original performance of the uniform weighted model means the re-weighted model only improves upon the baseline marginally.

Despite the theory that encoding features which could allow pet classification and head bounding box detection into the same latent space that is then used for segmentation would allow beneficial information to pass between the tasks, surprisingly *mtl* does not beat the baseline across any metric. This gap is particularly noticeable for IoU. The range between the performance of the strongest and weakest models is larger when measured in terms of IoU, indicating that it provides a fuller picture of segmentation ability.

Reducing model depth predictably makes for significantly poorer segmentation. Clearly this shows that going from 3 up and down sampling blocks to 4 neither leads to gradient instabilities nor diminishing returns on increased model capacity. This is to be expected since the original UNet architecture has 4 up and down sampling blocks and state-of-the-art segmentation networks tend to be deeper.

Table 2: Segmentation performance by model

Task	Model Name	Seg	Class	Bbox	Recon	Epochs	Pixel Accuracy	IoU	Dice Score
Baseline	base	1				14	90.75	0.801	0.889
MTL	mtl	1	1	1		27	88.00	0.754	0.861
		1	1	1/1000		18	89.02	0.773	0.872
		0.8	0.2	1/1000		20*	90.47	0.801	0.893
Depth Ablation	mtl-lessdepth	1	1	1		20*	82.26	0.637	0.780
Task Ablation	mtl-class	1	1	0		16	86.68	0.719	0.835
		0.8	0.2	0		22	90.57	0.802	0.891
	mtl-bbox	1	0	1		24	88.28	0.757	0.862
		1	0	1/1000		21	90.42	0.795	0.888
Open-ended question	mtl-rec	1	1		1/100000	20*	88.06	0.756	0.861
		0.8	0.2		1/100000	25	**91.73**	**0.822**	**0.903**
	mtl-rec with learned weights	1.657	2.542		$\frac{1}{17676}$	15*	87.00	0.721	0.840

Note that * indicates the choice of a custom number of training epochs, otherwise early stopping is used.

Task ablation has lead to less clear results - while the removal of the bounding boxes task has a detrimental effect in the uniformly weighted case, removing it when segmentation is properly prioritised leads to a moderate improvement, suggesting that previously, bounding boxes may merely have been making up for the over-weighting of classification loss. Ablating classification leads to improved performance when bboxes are scaled and otherwise no change. In general, this suggests that the removal of the auxiliary tasks leads to improvement on segmentation, though there is a hint that the bounding box task is surprisingly useful given that the baseline is generally stronger than the MTL models.

The *mtl-rec* models are best compared against *mtl-class* models; from this we can see that the addition of the reconstruction task causes moderate improvements in all metrics, leading to the baseline-beating model when combined with task re-weighting. This demonstrates that models can in fact extract beneficial knowledge about segmentation from the auxiliary tasks classification and reconstruction. In addition, both reconstruction models with set task weights perform as well as the equivalent full *mtl* model, indicating that reconstruction is an adequate replacement for bounding boxes.

The improvement from the *mtl-rec* model over *mtl-class* in performing the segmentation task can be visualised in the feature maps, where the convolutional filters seem to recognise more abstract features - apparent in sharper, more defined edges in the feature maps -, especially in the second feature, which is the one used in the skip connection for the final segmentation layer.

The most complex model trained in this investigation: *mtl-rec* with weights learnt using homoscedastic uncertainties, performed more poorly than the equivalent models with fixed loss weights. This is especially surprising given the importance of these weights shown by other comparisons. Perhaps the sub-optimal learnt weights were due to the limited training epochs - 15 epochs is less than any other MTL model.

Across the models trained with early stopping, *base* required the least epochs of training (as expected), while the task ablation models seem to need less than the full task, though this cannot be firmly concluded given an unknown interaction with task weight settings. This too could be reasonably expected as the result of a less complex optimization.

5 Discussion

The main limitation of this investigation became apparent as the empirical results were produced: at least for simultaneous MTL models, the loss weightings of the tasks are extremely important, often more significant that the addition or removal of tasks, yet the choice of these weightings was either naive or merely an educated guess. We attempted to rectify this by training a network to also learn these weights, though the learnt weights did not lead to improvements as expected. Future investigations in hyper-parameter tuning could be conducted to determine optimal tasks weights and to understand why the currently implemented mechanism has not worked.

The principal finding of this investigation was that knowledge about segmentation can indeed be extracted from neighbouring tasks on the same data, though the exact reason is to be determined. Future work identifying and tracing the latent features most used by the auxiliary tasks through the decoding layers unique to the target segmentation task, could add further support to this finding, while analysis of the test examples segmented well by the MTL models and less well by the baseline could shed light on why this is the case. The choice of auxiliary task which led to this result: image reconstruction, is shown to be particularly good when compared to the more obvious choice of classification and bounding boxes, both of which neither lead to baseline beating results nor showed up as being vitally important in the task ablation tests.

Another interesting result was that the bounding box task may be more relevant than the classification task to segmentation; perhaps models with the reconstruction and bounding box tasks trained together would beat the best existing model from this experiment. Additionally, future study into the interaction between the auxiliary tasks would answer whether they are well-suited to being used together.

6 Conclusion

In this investigation, we conducted tests to evaluate the impact of the addition of related tasks learnt simultaneously from the same data to a UNet-style segmentation model; there was not a performance improvement. Ablated models with less depth and less auxiliary tasks were also tested, demonstrating the necessity for depth. Next the addition of an image reconstruction task was shown to, under the right task weightings, produce models with better test performance than the baseline. Finally, an experiment into also having the task weightings learnt showed that further study on task trade-offs could be of use.

References

1. Yu Zhang and Qiang Yang. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*, PP, 07 2017.
2. Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, PP:1–34, 07 2020.
3. Lisa Torrey and Jude Shavlik. Transfer learning. `https://ftp.cs.wisc.edu/machine-learning/shavlik-group/torrey.handbook09.pdf`.
4. Sebastian Ruder. An overview of multi-task learning in deep neural networks. 06 2017.
5. Kaushik Sarveswaran Balamurali Murugesan. Psi-net: Shape and boundary aware joint multi-task deep network for medical image segmentation. 08 2019.
6. T. Brox O. Ronneberger, P. Fischer. U-net: Convolutional networks for biomedical image segmentation. 08 2015.
7. Shubham.jain Jain. An overview of regularization techniques in deep learning. `https://www.analyticsvidhya.com/blog/2018/04/fundamentals-deep-learning-regularization-techniques/`, April 2018.
8. Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7482–7491, 2018.
9. Jeremy Jordan. Evaluating image segmentation models. `https://www.jeremyjordan.me/evaluating-image-segmentation-models/`, Dec 2018.
10. Ekin Tiu. Metrics to evaluate your semantic segmentation model. `https://towardsdatascience.com/metrics-to-evaluate-your-semantic-segmentation-model-6bcb99639aa2`, Aug 2019.
11. Dr Daniel J Bell. Dice similarity coefficient. `https://radiopaedia.org/articles/dice-similarity-coefficient?lang=gb`, Aug 2021.

7 Appendix.A

Fig. 2: UNet Baseline model training result

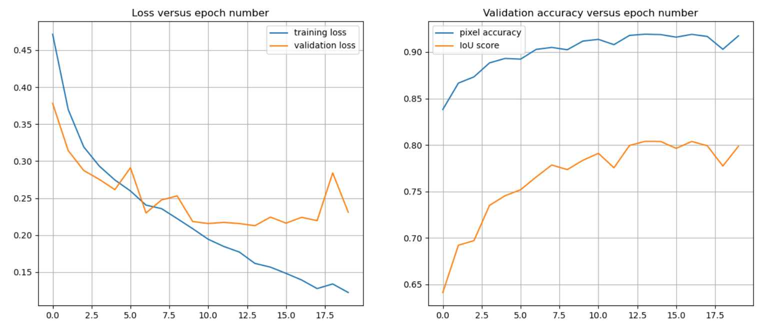


Fig. 3: UNet-Multi model training result

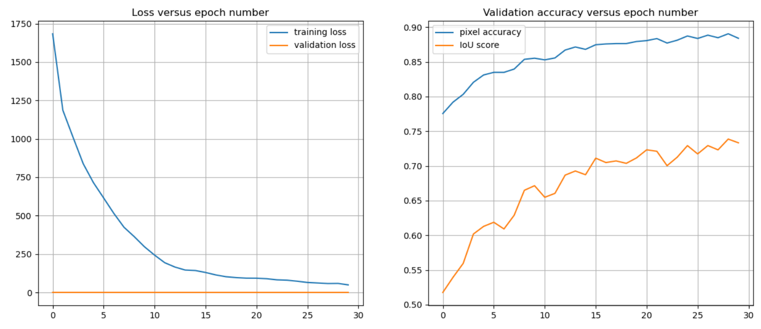


Fig. 4: UNet-Multi model using different weight training result

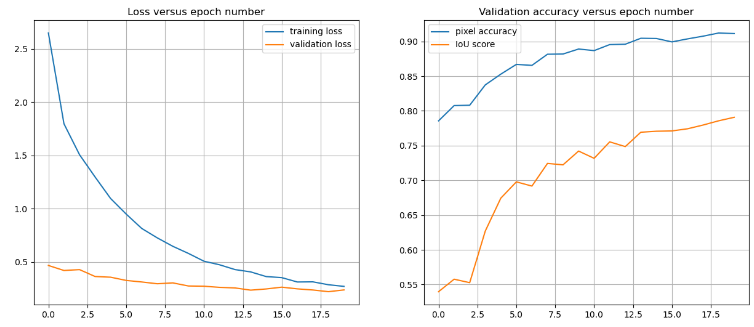


Fig. 5: UNet-Multi without Classification task model training result

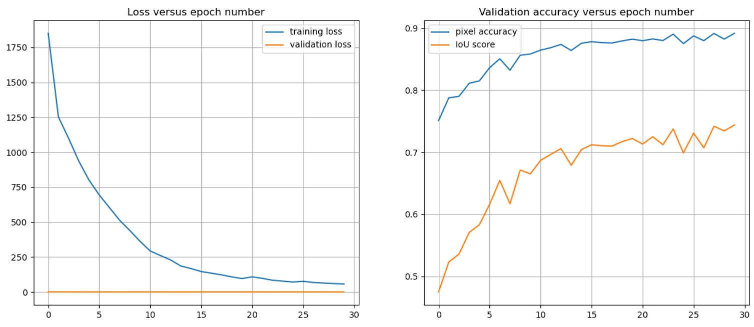


Fig. 6: UNet-Multi without Classification task model using different weights training result

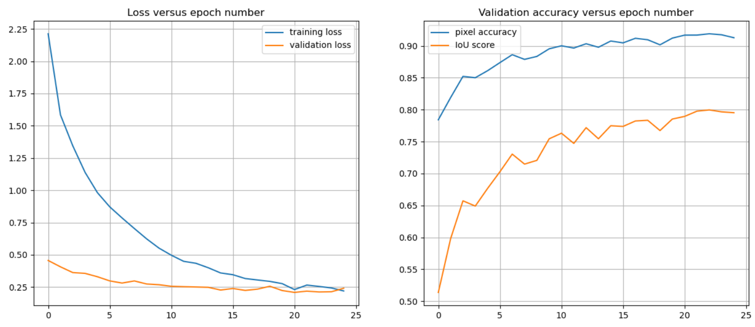


Fig. 7: UNet-Multi without Bounding Boxes task model training result

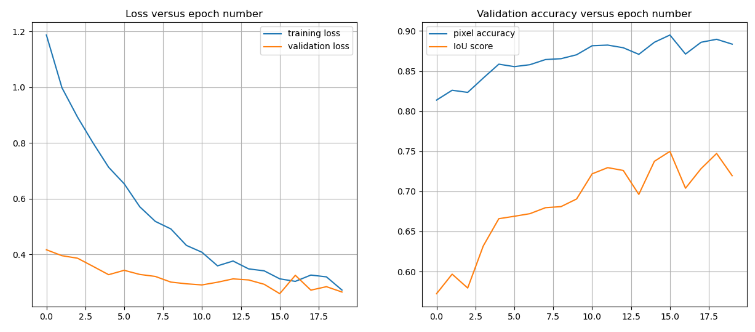


Fig. 8: UNet-Multi without Bounding Boxes task model using different weights training result

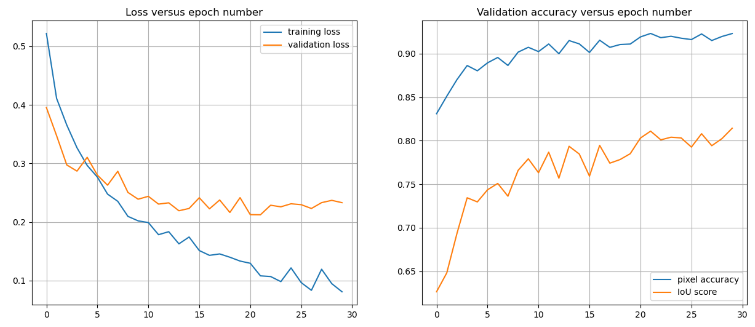
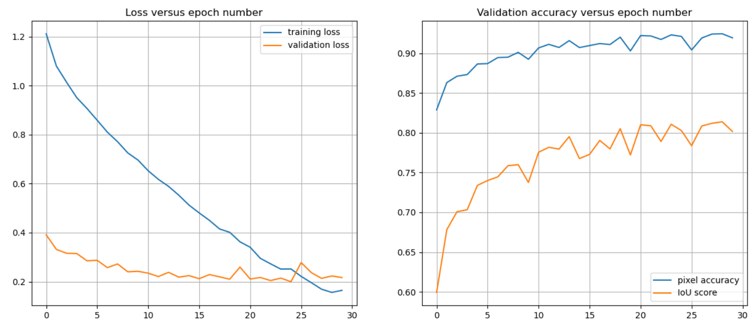


Fig. 9: UNet-Multi with Reconstruction model training result


Loss versus epoch number
training loss
validation loss
Validation accuracy versus epoch number
pixel accuracy
IoU score