COMP0090 Group Work (CW2) - Multi-task learning

1 Introduction

Multi-task learning (MTL) is a field of machine learning which focuses on learning multiple tasks either simultaneously or sequentially [1, 2, 3]. The aim of MTL is to leverage information from multiple tasks in order to improve performance for either all the tasks being learnt, or for a single separate 'target task'. In the case that target task(s) exist, auxiliary task(s) may be learnt prior to or simultaneously with the target task(s). Under this definition, 'transfer learning' may be considered as a special case of MTL with one auxiliary task and one target task, where the learning is sequential rather than simultaneous.

2 The group project

In this project you will implement and evaluate a MTL framework for a target task of animal segmentation. Learning auxiliary task(s) simultaneously with or prior to learning this target task may improve segmentation performance. The choice and number of auxiliary tasks, as well as the decision of whether learning is simultaneous or sequential, can all impact the target task performance in MTL. You will be implementing a MTL framework and evaluating this with respect to a non-MTL baseline as well as to a MTL framework that omits certain components (ablation study).

The goal of this project is to assess your ability to reason about design choices in deep learning experiments, to implement deep learning frameworks, and to evaluate the developed deep learning models, akin to conducting a research study. Your findings are to be compiled in a report which will follow a similar structure to a research paper.

2.1 The data set

The Oxford-IIIT Pet Data-set will be used in this project where a pre-processed version has been made available in an open-source repository. The pre-processed data contains 3686 samples with three kinds of task labels available for each sample: 1) binary classification of dog vs cat ('binary.h5'); 2) animal bounding boxes ('bboxes.h5'); 3) animal segmentation masks ('masks.h5'). In addition to these tasks, other tasks that utilise self-supervision such as self-reconstruction of the images using auto-encoding, predicting random rotations for images, predicting random flips for images etc. may be used, if suitable.

2.2 The "minimum required project"

The so-called minimum required project (MRP) (70%) for this group work consists of the following points, which are expected to be found in your submission. Some of these points are discussed in more detail to get you started.

1. Deciding and justifying which MTL formulation to use
2. Implementing a data loading pipeline
3. Implementing a MTL framework
4. Implementing a baseline network and an ablated version of MTL
5. Experimenting for network comparison
6. Describing implemented methods and conducted experiments
7. Summarising obtained results and drawing conclusions

2.2.1 Deciding and justifying which MTL formulation to use

In this project we only focus on a MTL formulation with a single target task with potentially multiple ($\geq$1) auxiliary tasks. It should be noted that formulations with multiple target tasks or all tasks being the target task are also valid MTL formulations. As introduced above, MTL may be formulated in one of two ways: 1) simultaneous learning: auxiliary task(s) are learnt together with the target task; 2) sequential/ transfer learning: auxiliary task(s) are learnt prior to the target task. In this project you must decide which formulation you want to use as well as present a brief survey of both kinds of MTL formulations (by citing previous work). The choice must be justified in the report.

Hint: some keywords that you can search for, are: 1) 'multi-task leanring'; 2) 'transfer learning'; 3) 'parameter sharing'; 4) 'deep learning fine-tuning'; 5) 'shared weights'.

2.2.2 Implementing a data loading pipeline

For this working point you must implement one data generator/ loader to sequentially load batches of data from h5 files, for the purpose of your project. For the loading pipeline in MTL, the output must be compatible with the deep learning framework you use for this project (either TensorFlow or PyTorch).

Hint: some keywords that you can search for, are: 1) 'multi-task learning'; 2) 'multi-output models'; 3) 'python yield statement'; 4) 'python generator'; 5) 'tensorflow data generator'; 6) 'pytorch data loader'.

```python
import h5py
import os
import numpy as np

# number of samples to load from h5 file
num_samples = 256

# define the path for the h5 files (in this case we use the train set)
h5_save_path = r'COMP0090_CW/train'

img_path = r'images.h5'
mask_path = r'masks.h5'
bbox_path = r'bboxes.h5'
bin_path = r'binary.h5'

total_samples = 2210

# generate random indices which are to be sampled from the dataset
inds_to_sample = np.sort(np.random.choice(total_samples, size=num_samples, replace=False
    ))

# define a function to load only samples at the specified indices
def load_data_from_h5(path, inds_to_sample):
    with h5py.File(path, 'r') as file:
        key = list(file.keys())[0]
        elems = file[key][inds_to_sample]
    return elems

# load data at the randomly generated indices
masks = load_data_from_h5(os.path.join(h5_save_path, mask_path), inds_to_sample)
imgs = load_data_from_h5(os.path.join(h5_save_path, img_path), inds_to_sample)
bboxes = load_data_from_h5(os.path.join(h5_save_path, bbox_path), inds_to_sample)
binarys = load_data_from_h5(os.path.join(h5_save_path, bin_path), inds_to_sample)
```

Listing 1: Simple loading

Note: if you fail to implement a suitable loading scheme, you can still proceed with the remainder of this project by loading all or part of the data directly into memory using the code snippet above, however, you will loose marks for this working point for doing so.

2.2.3 Implementing a MTL framework

Your MTL framework must follow one of the two schemes:

1. Simultaneous learning: The auxiliary task(s) and target task are learnt simultaneously, where your network 'shares parameters' in some layers whereas some layers are specific to the task and non-shared (usually output layers). Your model may then be considered to have multiple output branches, each learning a separate task. Once trained, you will only be required to evaluate your

model for the target task. You have freedom to implement any network that you deem to be suitable, however, please note that you will not be marked for the complexity of your network, but simply for the correctness of the parameter sharing and multi-output behaviour.

2. Sequential/ transfer learning: The auxiliary task is to be learnt prior to learning the target task. Parameters are again shared between the two learning stages (e.g. you can learn the auxiliary task, then freeze parameters in all layers except the last after which you can edit the output layer according to your target task and train). After training, you will only need to evaluate your model for the target task. You have freedom to implement any network that you deem to be suitable, however, please note that you will not be marked for the complexity of your network, but simply for the correctness of the parameter fixing and layer editing behaviour.

Other things to consider include, but are not limited to, network architecture, training strategies, regularisers, hyperparamters for both types of tasks. Describe and justify these details in the report, if considered relevant.

2.2.4 Implementing a baseline network and an ablated version of MTL

The baseline network must be a network capable of performing only your target task. Discuss what should be considered a fair comparison, in aspects such as network architecture and/or training strategies, compared with the MTL network.

For the ablation study, you must perform at least one ablation study by removing at least one component of the MTL network. Discuss why this is an interesting ablation study to do. Examples of components that can be ablated include 1) removing one auxiliary task from the simultaneous learning scheme, if you have multiple auxiliary tasks; 2) removing parameter fixing from transfer learning scheme.

2.2.5 Experimenting for network comparison

Design experiments which 1) compare the baseline to the MTL, and 2) compare the ablated version to the MTL. Things to consider may include, but not limited to, planning data sets and model development, metrics to use, how to quantitatively/statistically compare these metrics. Describe and justify these options in the report, if relevant.

2.3 An "open-ended question"

To be awarded the remaining 30% in this project, you need to come up with a new research/study question to answer, an open-ended question (OEQ). Novelty is encouraged in this part. The group should clearly identify and generate a hypothesis (i.e. the research/study question), design experiments that produce results to answer this question and analyse the obtained experimental results for quantitative conclusion. This part needs to be built on the MRP and relevant to MTL. Describe the question, experiment and results clearly and cohesively with the MRP in the report.

Some example research/study questions are given as follows.

- What other auxiliary tasks can be used
- What exactly in the auxiliary task is used to help the target task
- Is there any other things that can help the target task
- What can be used to control the "transfer" between the target and auxiliary tasks

  .....

3 Submission and Marking Criteria

Each group will submit 1) one single joint report and 2) one huddle of code in a single zip file (50 MB file size limit also applies). Individuals will also submit a peer assessment form. More details of submission will be available on Moodle before the (updated) deadline on 17^{th} January 2022.

3.1 Report

Your report must be structured as a research paper. Each group only needs to submit one joint report. Unlike the first individual coursework, the marking is primarily based on the report. The submitted report must contain the following sections (followed with the suggested content). The percentages indicate the maximum marks for each section, each with a 7:3 split for the MRP and OEQ, respectively.

Note: your report must not exceed 6 pages in the LNCS template, excluding references.

- **Title**: an interesting and informative title of the study (0%)
- **Introduction**: background, literature, motivation, the research/study questions to address (20%)
- **Methods**: the methodological details for the implemented MTL and other networks, optional novel methodology for the OEQ (25%)
- **Experiments**: two MRP comparisons, other experiments for the OEQ (25%)
- **Results**: a summary of results, comparison, quantitative analysis (20%)
- **Discussion**: interesting findings, unanswered questions, limitations, future directions of this study (5%)
- **Conclusion**: summarising the study and the results (5%)

The marking criteria used is adapted using the departmental project marking criteria (both UG and PG) as reference. To summarise, in descending order of importance:

- Scientific soundness: reasoning and justification of problems, methods and experiments, background reading and understanding, experiment design and quality
- Technical accuracy: appropriate and correct use of terminology, methodology, data, code and other tools
- Completeness: objective achieved, conclusion of study question, completeness of report
- Presentation: writing organisation, clarity, illustrative report, structure, code readability
- Critical appraisal: conclusive results, informative analysis, future outlook
- Novelty (only relevant to OEQ): this point can be subjective, and can be particularly related to the background and existing literature

3.2 Code

In this project, you must only use one of the PyTorch or TensorFlow for the entirety of the project. However, you can use up to three additional pip-installable packages within the 'comp0090-cw1' conda environment, in addition to any other available tools in the conda environment. If you do choose to use the optional Python packages, please also include a "requirements.txt" file detailing these additional packages, in the submitted code bundle. See the official document for further guidance. It is your responsibility to ensure the compatibility of the packages you added and that they can run on a marking environment on Ubuntu 20.04.

There is no specific format or style required for how you design, structure and implement your code. Include a brief "instruction.txt" file to list detailed steps to run your code, reproducing all the results included in the report.

3.3 Peer assessment

Your final mark for this project will be assigned based on peer assessment. A mark will be awarded to the group and this group mark will be scaled/ modified based on scores given to you by peers in your group. UCL's recommended algorithm will be used to scale/ modify the awarded marks based on peer assessment. A link to the form will be available on Moodle for individuals to give a Likert score (between 1 to 5) to each group member, based on your judgement of her/his contribution to the group work. This peer assessment is a mechanism to encourage engagement and contribution to the group work only, not for other purposes such as ranking group members or encouraging internal competition, so there will be no limit on sum of scores you can give, i.e. all fives are completely acceptable for a functioning collaborating group - in this case, all members will be awarded the group mark.

References

[1] Ruder, S., 2017. An overview of multi-task learning in deep neural networks. arXiv preprint arXiv:1706.05098.

[2] Vafaeikia, P., Namdar, K. and Khalvati, F., 2020. A Brief Review of Deep Multi-task Learning and Auxiliary Task Learning. arXiv preprint arXiv:2007.01126.

[3] Zhang, Y. and Yang, Q., 2018. An overview of multi-task learning. National Science Review, 5(1), pp.30-43.