4-6 Wednesday – 210-GD3

Special topics in Computer Science INT3121 20

Lecturer: Nguyen Thi Ngoc Diep, Ph.D.

Email: ngocdiep@vnu.edu.vn

Slide & Code: https://github.com/chupibk/INT3121-20

Image classification with convolutional neural networks

Week	Content	Class hour	Self-study hour
1 28/8/2019	Introduction Image classification problem and its applications A toy problem with CIFAR10	2	1
2 (4/9/2019)	CNN model architectures and visualization	2	1
3 (11/9/2019)	Training and tuning parameters Automatic parameter learning	2	1
4 (18/9/2019)	Data augmentation Data generator	2	2-6
5 (25/9/2019)	Transfer learning	2	2-6
6 (2/10/2019)	Multi-output image classification	2	2-6
7 (9/10/2019)	Building a training dataset How to write a report	1	2-6
8, 9, 10, 11	Seminar: Bag of tricks with CNN (as mid-term tests)	1	2-6
12, 13, 14	Final project presentations	1-3	2-6
15	Class summarization	1	open

Recall week 5: Transfer learning

1. ConvNet as fixed feature extractor
   - Remove (some) last fully-connected layer then run forward pass to extract features
2. Fine-tuning the ConvNet
   - Do backpropagation on some last layers only
3. Pretrained models
   - Do backpropagation on the whole model but with pretrained weights

Multi-output classification

a.k.a: Multi-task learning (MTL)

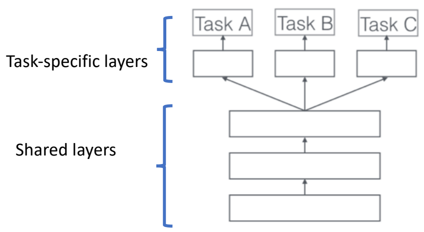
Multi-task learning
or Multi-output classification

- Try to predict different types of labels at the same time
- Multi-output != multi-class
  - Multi-class: number of labels > 2
  - Multi-output: number of types of labels > 1

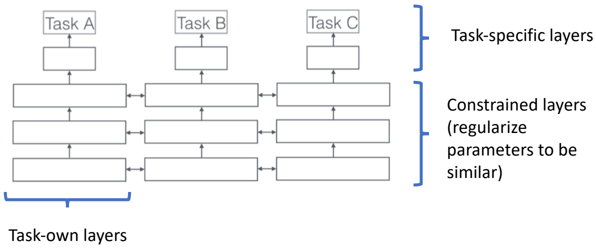
Multi-output classification example

Image credit: pyimagesearch.com

Two MTL methods



Hard parameter sharing



Soft parameter sharing

Image credit: Sebastian Ruder, https://arxiv.org/pdf/1706.05098.pdf

MTL architecture

MTL assumptions

<u>Reused from "Transfer learning"</u>

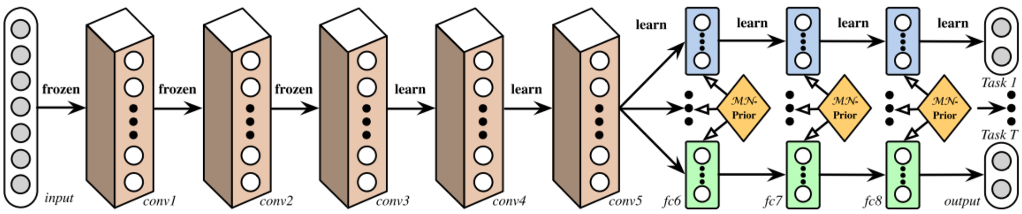
- Implicit data augmentation
- Eavesdropping: learn features from task A for task B
- Representation bias: task A & B likely prefer a same presentation if they are quite similar

<u>MTL's original ideas</u>

- Attention focusing: learn two tasks at the same time → more attentive to relevant features
- Regularization: reducing risk of overfitting

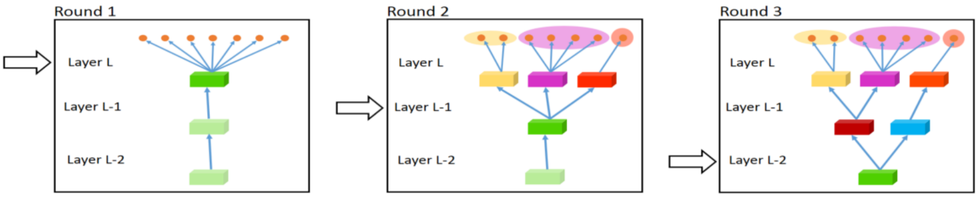
Reference: Sebastian Ruder, "An overview of Multi-task Learning in Deep Neural Networks," arxiv preprint, 2017

MTL with matrix priors



Long, M. and Wang, J. (2015). Learning Multiple Tasks with Deep Relationship Networks. *arXiv preprint arXiv:1506.02117*.

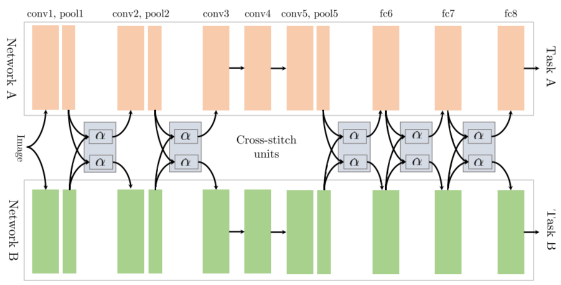
MTL with fully-adaptive feature sharing



Iteratively widen the network using a criterion that promotes grouping of similar tasks

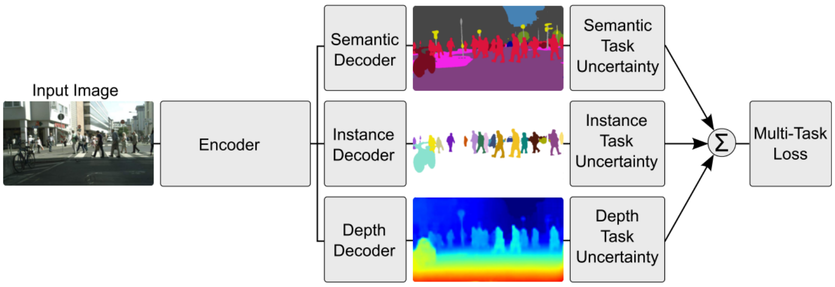
Lu, Y., Kumar, A., Zhai, S., Cheng, Y., Javidi, T., and Feris, R. (2016). Fully-adaptive Feature Sharing in Multi-Task Networks with Applications in Person Attribute Classification.

MTL with cross-stitch network



Misra, I., Shrivastava, A., Gupta, A., and Hebert, M., Cross-stitch Networks for Multi-task Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. , 2016

MTL with uncertainty-based loss function weighting



Kendall, Alex, Yarin Gal, and Roberto Cipolla. "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018.

MTL training

Model output

- All tasks are associated with a single output

```
model = Model(
    inputs=inputs,
    outputs=[outBranch1, outBranch2]
)
```

Combining losses

- Sum of individual losses
- Weighted sum of individual losses

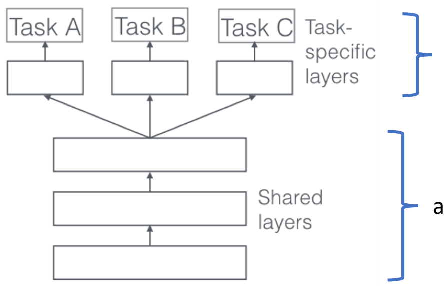
$$L_{total} = \sum_i w_i L_i.$$

```
model.compile(loss={'branch1': 'categorical_crossentropy',
                    'branch2': 'binary_crossentropy'},
              loss_weights={'branch1': 1.0,
                            'branch2': 0.5},
              optimizer='adam',
              metrics={'branch1': 'accuracy',
                       'branch2': ['binary_crossentropy', 'mse']}
)
```

Training data

```
model.fit(x_train,
          {'branch1': y_train, 'branch2': x_train},
          batch_size=batch_size,
          epochs=epochs,
          validation_data= (x_test,
                               {'branch1': y_test, 'branch2': x_test}
                           ),
          verbose=1
)
```

Training learning rates



Different learning rates for each task head

a learning rate

Keras:

```
multipliers = {'dense_1': 0.5, 'dense_2': 0.4}
opt = LearningRateMultiplier(SGD, lr_multiplier=multipliers, lr=0.001, momentum=0.9)
```

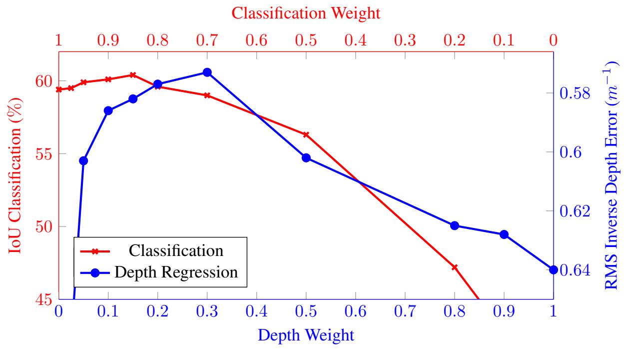
Per layer learning rate

How to weigh losses

Reference:

Kendall, Alex, Yarin Gal, and Roberto Cipolla. "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018.

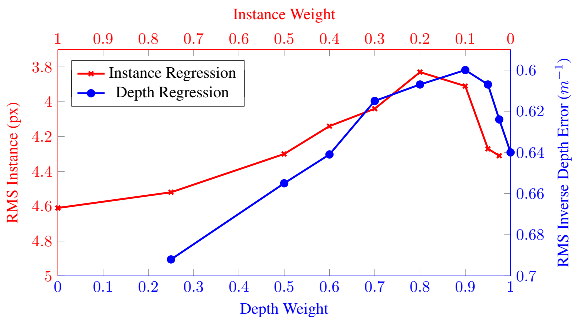
Multi-task vs single task learning



Task Weights Class	Task Weights Depth	Class IoU [%]	Depth Err. [px]
1.0	0.0	59.4	-
0.975	0.025	59.5	0.664
0.95	0.05	59.9	0.603
0.9	0.1	60.1	0.586
0.85	0.15	60.4	0.582
0.8	0.2	59.6	0.577
0.7	0.3	59.0	0.573
0.5	0.5	56.3	0.602
0.2	0.8	47.2	0.625
0.1	0.9	42.7	0.628
0.0	1.0	-	0.640
Learned weights with task uncertainty (this work, Section 3.2)		**62.7**	**0.533**

(a) Comparing loss weightings when learning **semantic classification and depth regression**

Multi-task vs single task learning



Task Weights		Instance	Depth
Instance	Depth	Err. [px]	Err. [px]
1.0	0.0	4.61	
0.75	0.25	4.52	0.692
0.5	0.5	4.30	0.655
0.4	0.6	4.14	0.641
0.3	0.7	4.04	0.615
0.2	0.8	3.83	0.607
0.1	0.9	3.91	0.600
0.05	0.95	4.27	0.607
0.025	0.975	4.31	0.624
0.0	1.0		0.640
Learned weights with task uncertainty (this work, Section 3.2)		**3.54**	**0.539**

(b) Comparing loss weightings when learning **instance regression and depth regression**

Figure 2: **Learning multiple tasks improves the model's representation and individual task performance**. These figures and tabl

Type of uncertainty

- Epistemic
  - Can be explained away with increased training data
- Aleatoric
  - Can be explained away with the ability to observe all explanatory variables with increasing precision
- Data-dependent or Heteroscedastic
  - Depends on the input data and is predicted as a model output
- Task-dependent or Homoscedastic
  - Is independent on the input data, varies between different tasks

Assumption

- A multi-task loss function is derived on maximising the Gaussian likelihood with homoscedastic uncertainty

$$p(\mathbf{y}|\mathbf{f}^{\mathbf{W}}(\mathbf{x})) = \mathcal{N}(\mathbf{f}^{\mathbf{W}}(\mathbf{x}), \sigma^2)$$

$$\begin{aligned} p(\mathbf{y}_1, \mathbf{y}_2|\mathbf{f}^{\mathbf{W}}(\mathbf{x})) &= p(\mathbf{y}_1|\mathbf{f}^{\mathbf{W}}(\mathbf{x})) \cdot p(\mathbf{y}_2|\mathbf{f}^{\mathbf{W}}(\mathbf{x})) \\ &= \mathcal{N}(\mathbf{y}_1; \mathbf{f}^{\mathbf{W}}(\mathbf{x}), \sigma_1^2) \cdot \mathcal{N}(\mathbf{y}_2; \mathbf{f}^{\mathbf{W}}(\mathbf{x}), \sigma_2^2). \end{aligned}$$

$$\begin{aligned} \log p(\mathbf{y} = c|\mathbf{f}^{\mathbf{W}}(\mathbf{x}), \sigma) = &\frac{1}{\sigma^2} f_c^{\mathbf{W}}(\mathbf{x}) \\ &- \log \sum_{c'} \exp\left(\frac{1}{\sigma^2} f_{c'}^{\mathbf{W}}(\mathbf{x})\right) \end{aligned}$$

$$\begin{aligned} \mathcal{L}(\mathbf{W}, \sigma_1, \sigma_2) &= -\log p(\mathbf{y}_1, \mathbf{y}_2 = c|\mathbf{f}^{\mathbf{W}}(\mathbf{x})) \\ &= -\log \mathcal{N}(\mathbf{y}_1; \mathbf{f}^{\mathbf{W}}(\mathbf{x}), \sigma_1^2) \cdot \mathrm{Softmax}(\mathbf{y}_2 = c; \mathbf{f}^{\mathbf{W}}(\mathbf{x}), \sigma_2) \\ &= \frac{1}{2\sigma_1^2}||\mathbf{y}_1 - \mathbf{f}^{\mathbf{W}}(\mathbf{x})||^2 + \log \sigma_1 - \log p(\mathbf{y}_2 = c|\mathbf{f}^{\mathbf{W}}(\mathbf{x}), \sigma_2) \\ &= \frac{1}{2\sigma_1^2}\mathcal{L}_1(\mathbf{W}) + \frac{1}{\sigma_2^2}\mathcal{L}_2(\mathbf{W}) + \log \sigma_1 \\ &\quad + \log \frac{\sum_{c'} \exp\left(\frac{1}{\sigma_2^2} f_{c'}^{\mathbf{W}}(\mathbf{x})\right)}{\left(\sum_{c'} \exp\left(f_{c'}^{\mathbf{W}}(\mathbf{x})\right)\right)^{\frac{1}{\sigma_2^2}}} \\ &\approx \frac{1}{2\sigma_1^2}\mathcal{L}_1(\mathbf{W}) + \frac{1}{\sigma_2^2}\mathcal{L}_2(\mathbf{W}) + \log \sigma_1 + \log \sigma_2, \end{aligned}$$

Experiment results

Loss	Task Weights Seg.	Inst.	Depth	Segmentation IoU [%]	Instance Mean Error [px]	Inverse Depth Mean Error [px]
Segmentation only	1	0	0	59.4%	-	-
Instance only	0	1	0	-	4.61	-
Depth only	0	0	1	-	-	0.640
Unweighted sum of losses	0.333	0.333	0.333	50.1%	3.79	0.592
Approx. optimal weights	0.89	0.01	0.1	62.8%	3.61	0.549
2 task uncertainty weighting	✓	✓		61.0%	**3.42**	-
2 task uncertainty weighting	✓		✓	62.7%	-	0.533
2 task uncertainty weighting		✓	✓	-	3.54	0.539
3 task uncertainty weighting	✓	✓	✓	**63.4%**	3.50	**0.522**

Mid-term registration

- Form to register groups: https://forms.gle/BgAWdsCjHgD1nwoA6
- Spreadsheet to change datasets, submit references, etc.
  - Edit with care!!!! (don't change things that do not belong to you)
  - https://docs.google.com/spreadsheets/d/1HtpFzZUsacJqzrSXr66nnN8V1TB8ErnvCgrB3V07mqo/edit?usp=sharing