Universität St.Gallen

Data-efficient Deep Learning for Earth Observation

Multi-Task Learning

Michael Mommert, Joëlle Hanna, Linus Scheibenreif, Damian Borth
University of St.Gallen

IGARSS

Concept

- **Conventional approach: one network for one task**

Concept

- Conventional approach: one network for one task
- But: tasks are sometimes related or share common underlying structures

Concept

- Conventional approach: one network for one task
- But: tasks are sometimes related or share common underlying structures
- Example in real-life:

Task A: Learning to play the piano

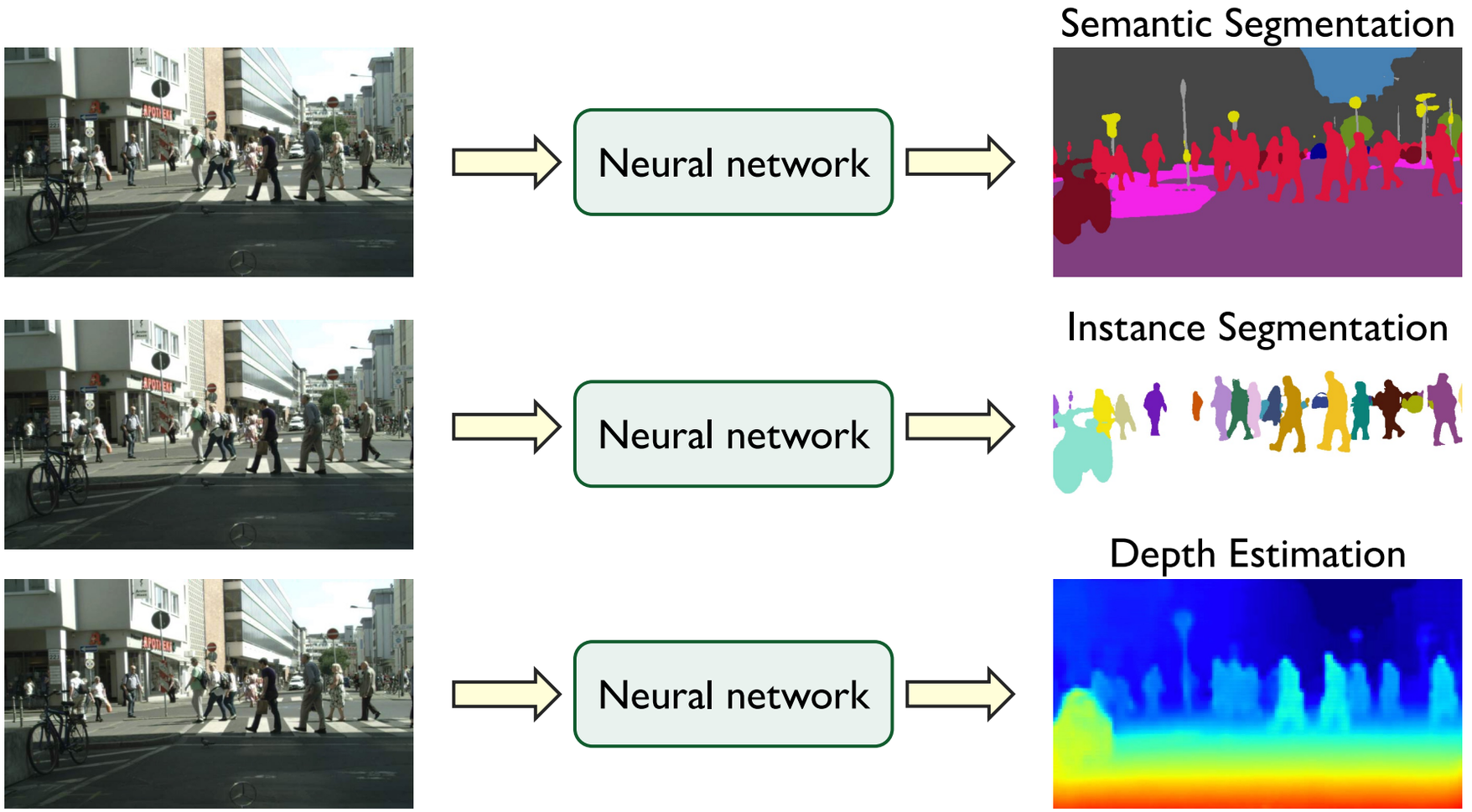
Task B: Learning to play the saxophone

Underlying Structure: reading partitions, coordinating left-right hands, sense of rhythm etc.

Learning multiple tasks simultaneously can results in mutual benefits for each task

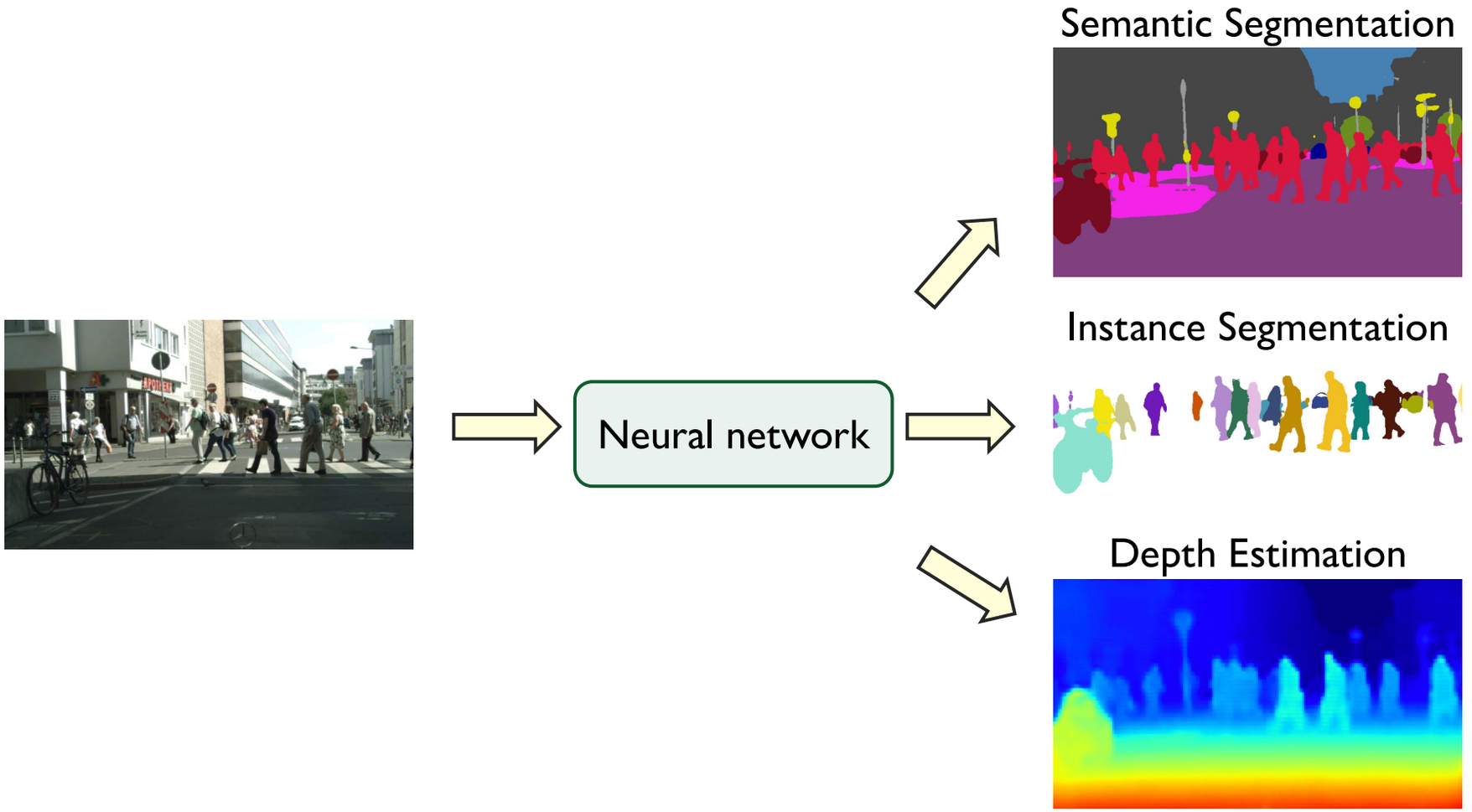
Concept

- From three single task models …



Concept

- … to one multi-task model



Advantages of Multi-Task Learning

- **Some advantages of Multi-Task Learning (MTL):**
  - Improved data efficiency
  - Reduced overfitting
  - Better generalization

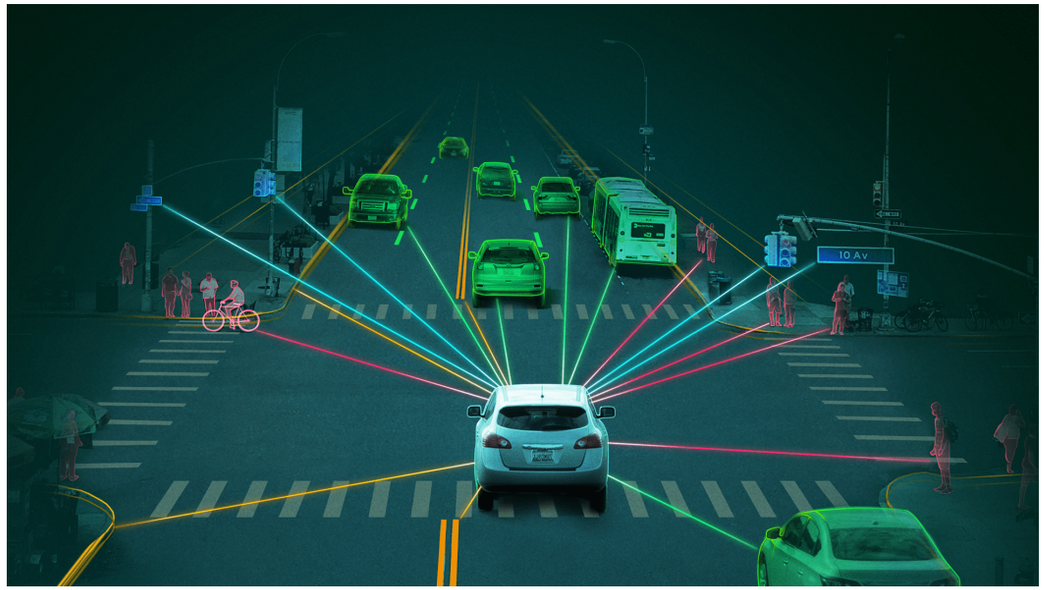
Advantages of Multi-Task Learning

- **Some advantages of Multi-Task Learning (MTL):**
  - Improved data efficiency
  - Reduced overfitting
  - Better generalization

All these advantages can be attributed to the learning of shared representations

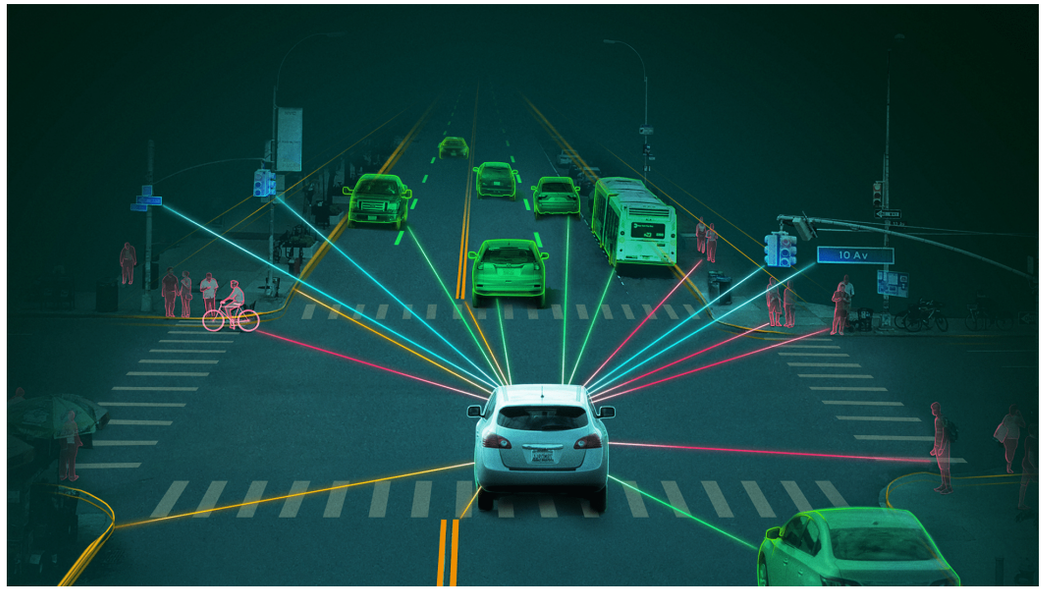
Real World Application: Autonomous driving

- Object detection and semantic segmentation of surrounding environment
- Lane detection and trajectory prediction



Real World Application: Autonomous driving

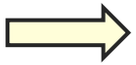
- Object detection and semantic segmentation of surrounding environment
- Lane detection and trajectory prediction



Leading to safer and more reliable driving systems, with a better understanding of road scenes, enabling better prediction of the trajectory of other vehicles or pedestrians

Task Relationships

- **Task Similarity**
  - Example: emotion classification and sentiment analysis



Closely related tasks often share common underlying features and patterns. Exploiting the similarities between them can lead to improved performance. On the other hand, tasks with little or no similarity may benefit from separate models.

Task Relationships

- Task Similarity
  - Example: emotion classification and sentiment analysis

⇒ Closely related tasks often share common underlying features and patterns. Exploiting the similarities between them can lead to improved performance. On the other hand, tasks with little or no similarity may benefit from separate models.

- Task Hierarchy
  - Example: semantic segmentation and instance segmentation

⇒ Leveraging task hierarchies can facilitate knowledge transfer and enable learning more complex tasks through the shared representation of lower-level tasks.

Task Relationships

- Task Similarity
  - Example: emotion classification and sentiment analysis

⇨ Closely related tasks often share common underlying features and patterns. Exploiting the similarities between them can lead to improved performance. On the other hand, tasks with little or no similarity may benefit from separate models.

- Task Hierarchy
  - Example: semantic segmentation and instance segmentation

⇨ Leveraging task hierarchies can facilitate knowledge transfer and enable learning more complex tasks through the shared representation of lower-level tasks.

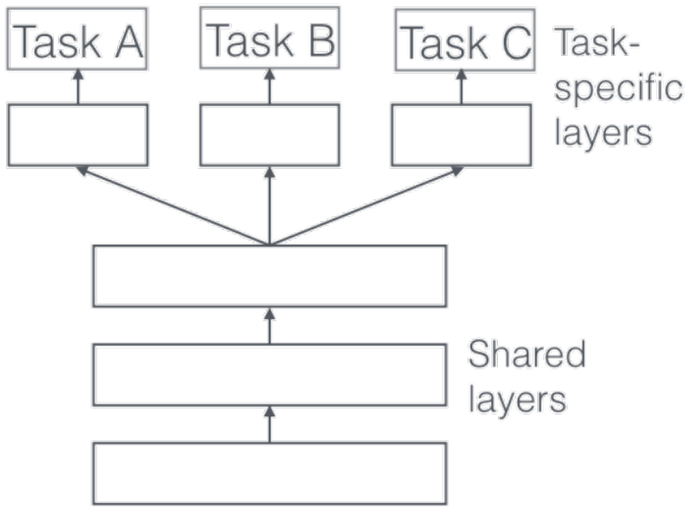
- Task Complementarity
  - Example: object recognition and scene classification

⇨ Identifying and exploiting such complementarities can enhance the overall learning performance.

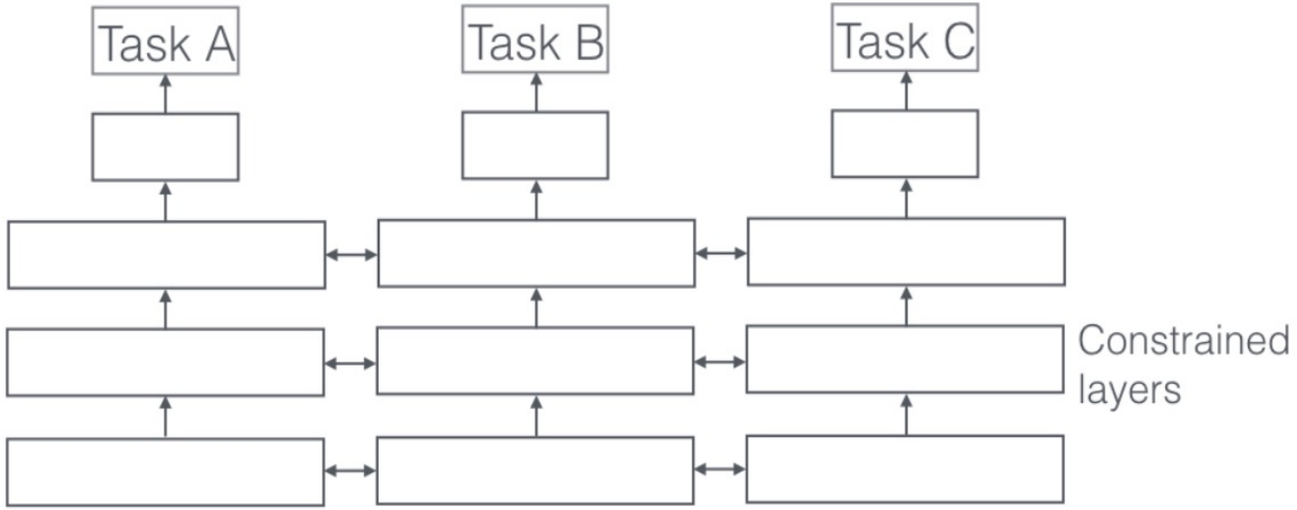
Main Approaches

- **They can be divided in two parts:**

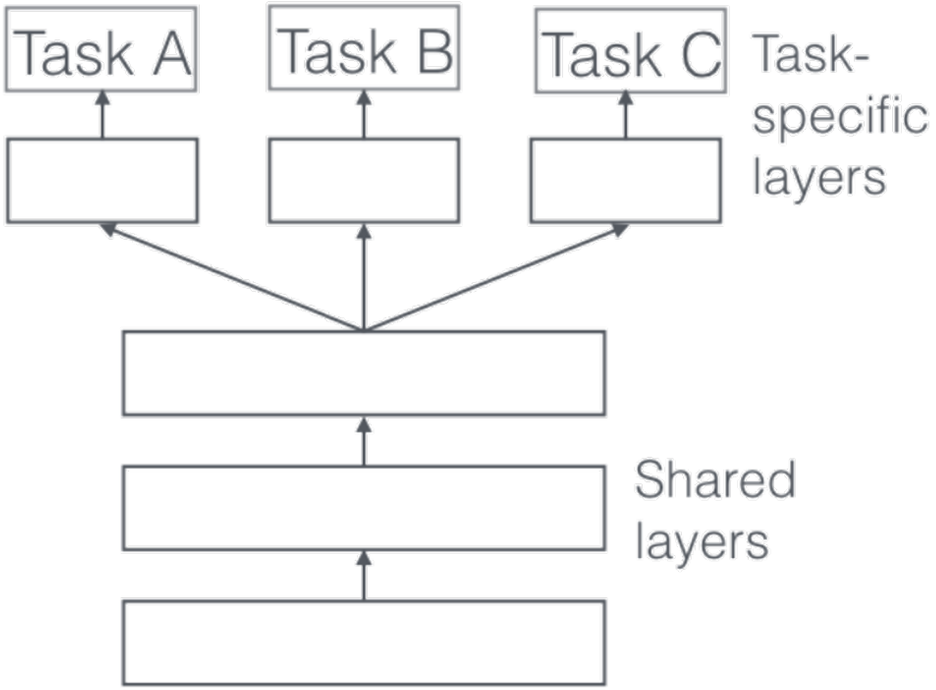
Hard parameter sharing



Soft parameter sharing



Hard Parameter Sharing

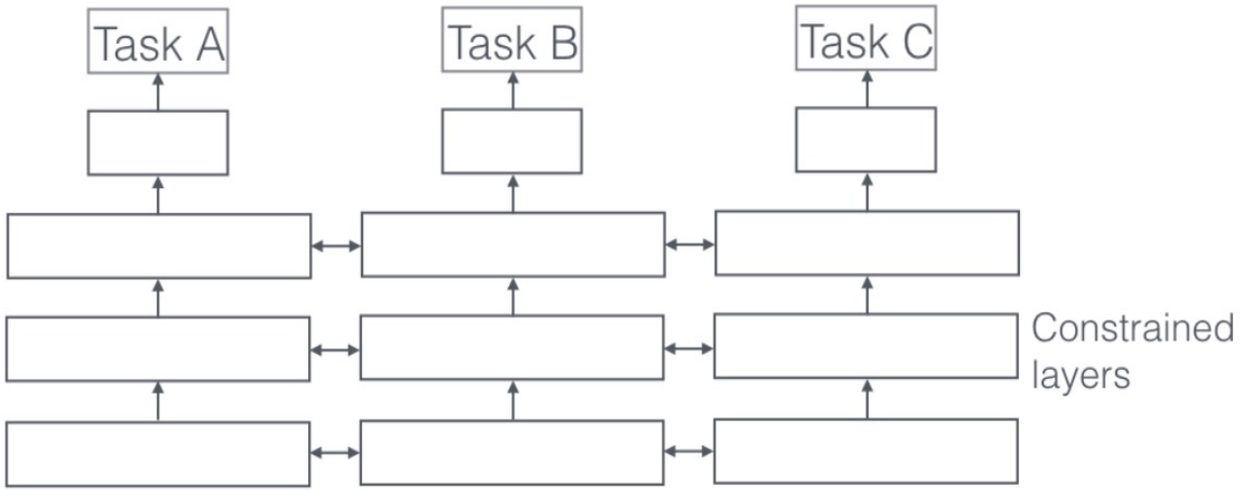


Main characteristics:

- All tasks share several hidden layers
- Additional task-specific output layers

If the tasks are related, a feature that's important for Task A is probably important for Task C as well. Similarly, unimportant features are likely to be unimportant for all the tasks.

Soft Parameter Sharing

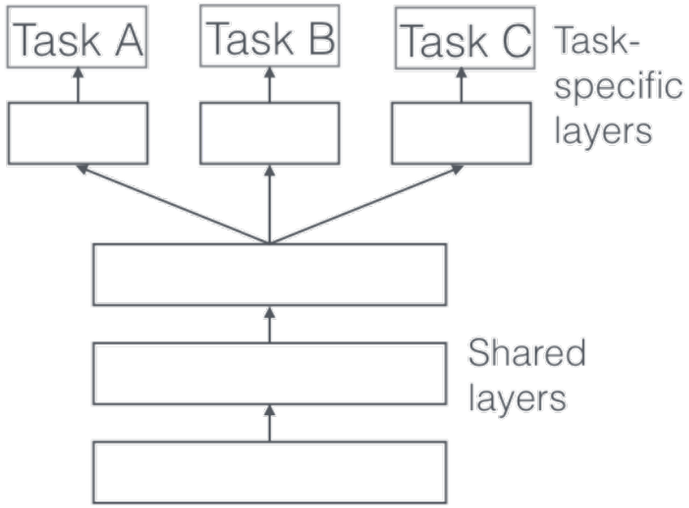


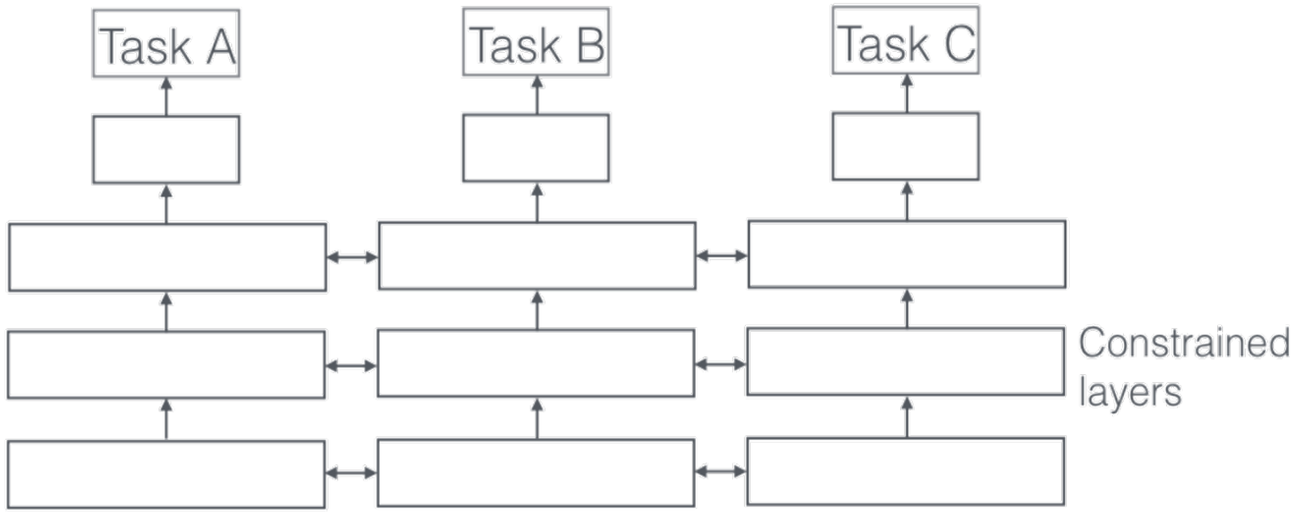
Main characteristics:

- Each model has its own parameters
- Distance between parameters is regularized as part of the loss function

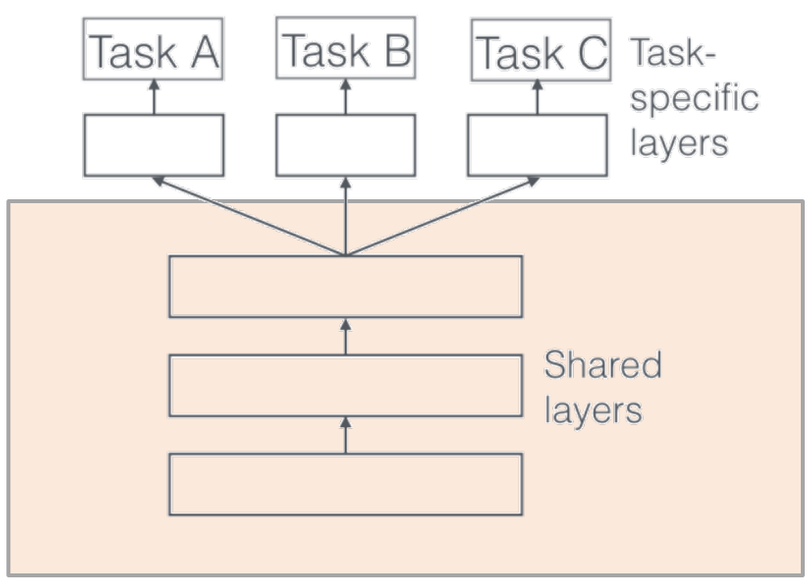
It introduces an additional term in the loss function. It constrains the activations in the respective layers to be similar (but not forcing equality)

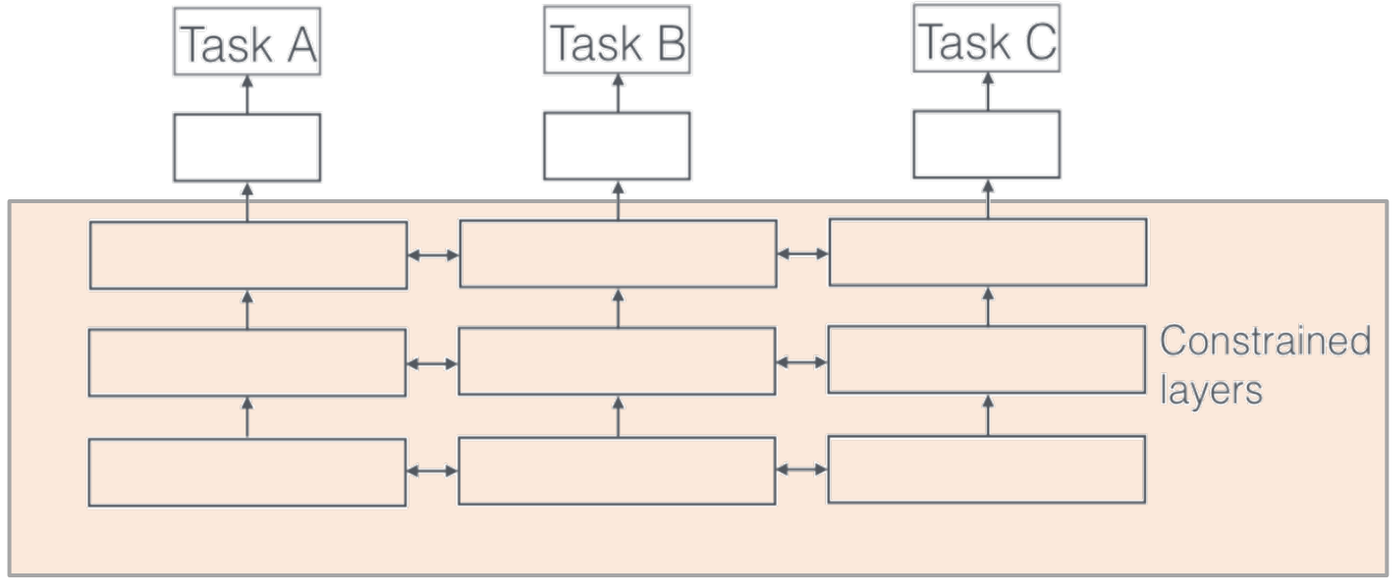
Joint Optimization



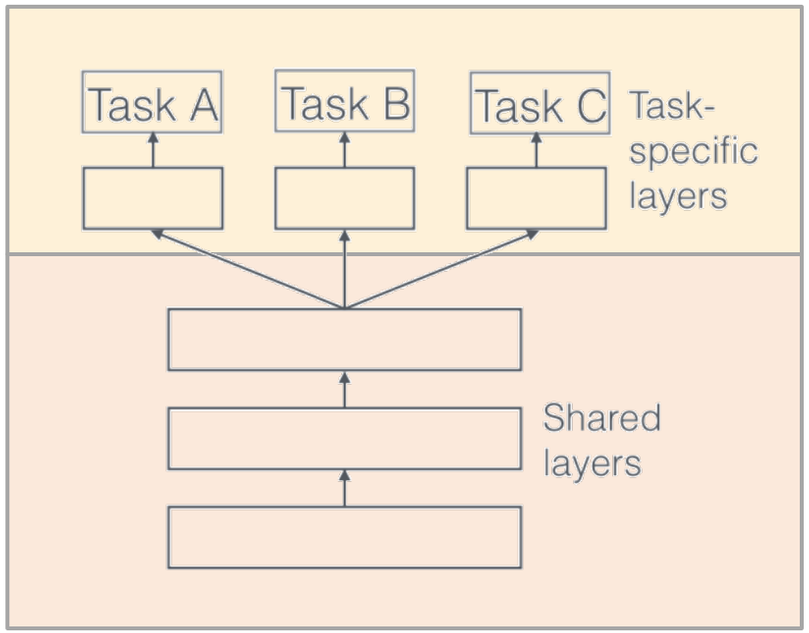


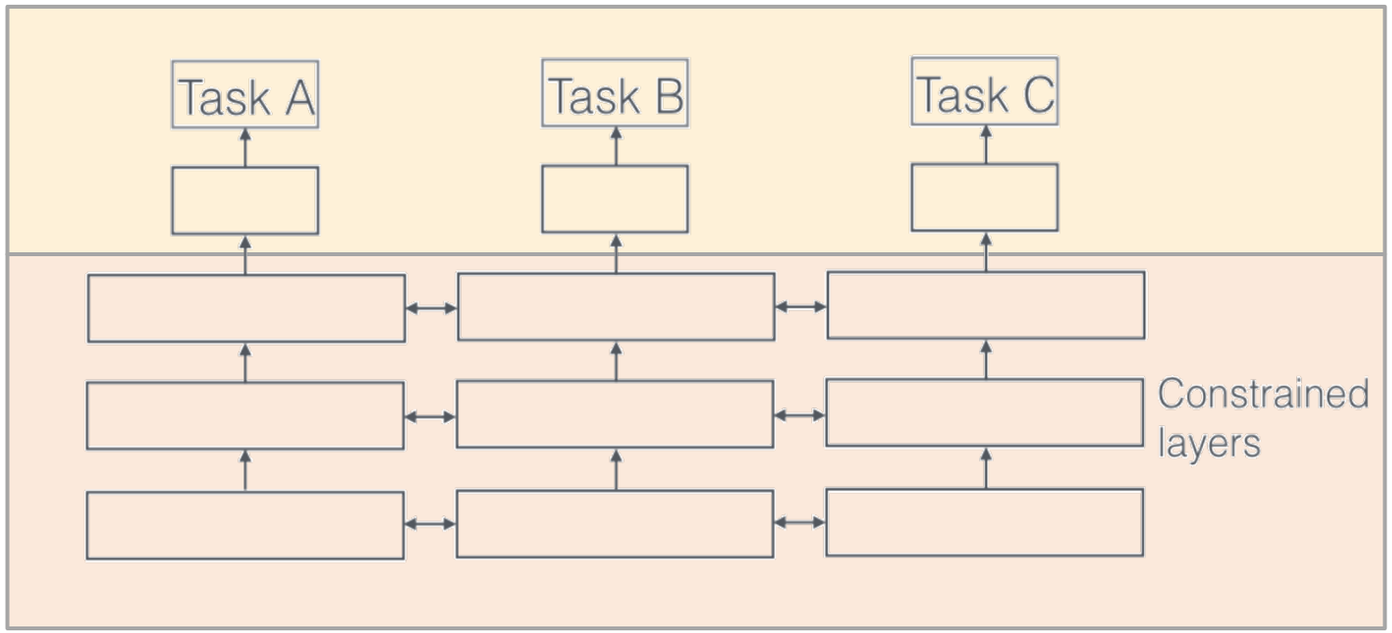
Joint Optimization



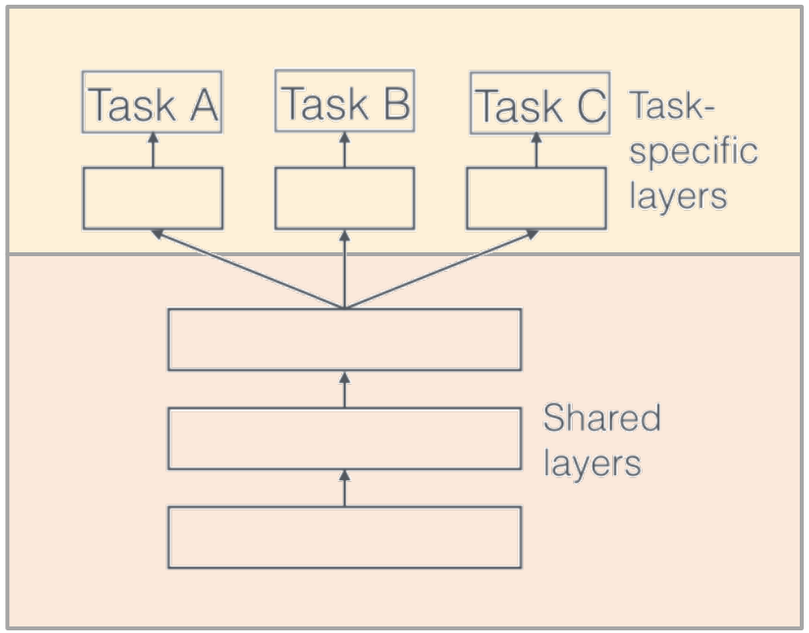


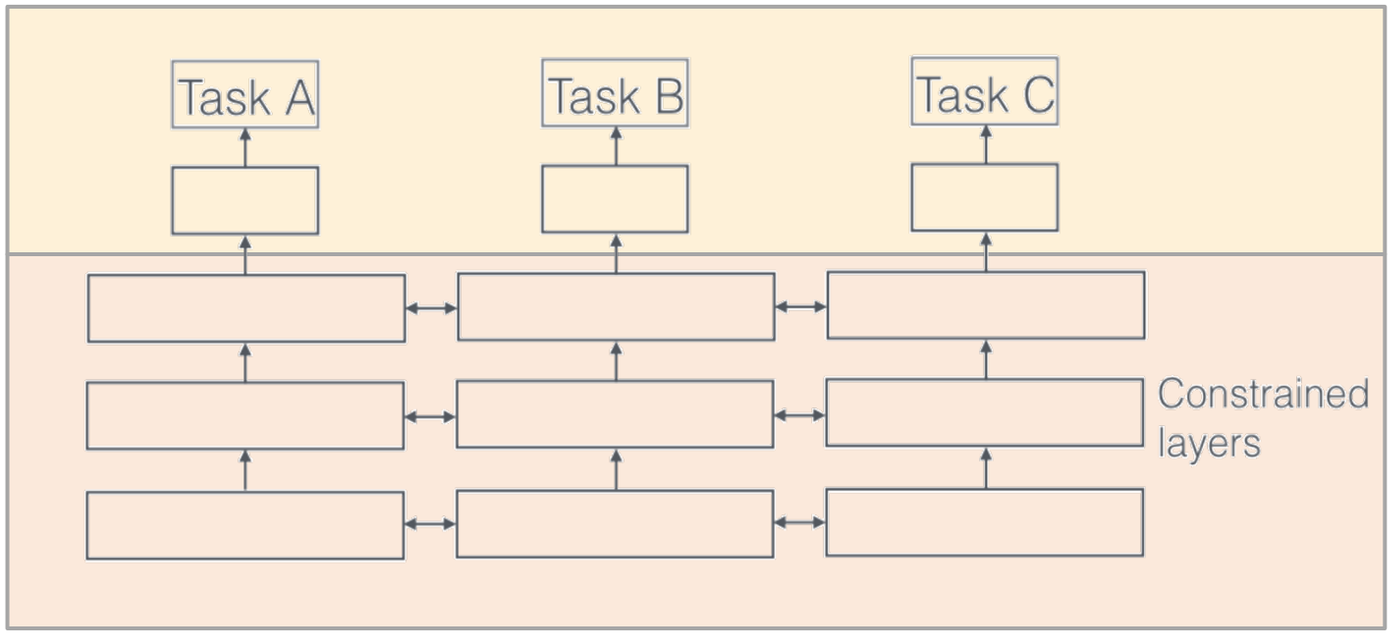
Joint Optimization





Joint Optimization





The optimization process aims to strike a balance between exploiting information shared between tasks and taking into account the specific characteristics of each task.

Loss Functions

- Joint optimization can be achieved through minimization of the sum of task-specific losses. For T tasks, the loss function is as follows:

$$L_{total} = \sum_{t=1}^{T} w_t L_t$$

Loss Functions

- Joint optimization can be achieved through minimization of the sum of task-specific losses. For T tasks, the loss function is as follows:

$$L_{total} = \sum_{t=1}^{T} w_t L_t$$

- L_t can be one of the following:
  - L1 Loss
  - Mean Squared Error (MSE)
  - Cross-entropy Loss
  - Custom Loss
  - …

Loss Functions

- Joint optimization can be achieved through minimization of the sum of task-specific losses. For T tasks, the loss function is as follows:

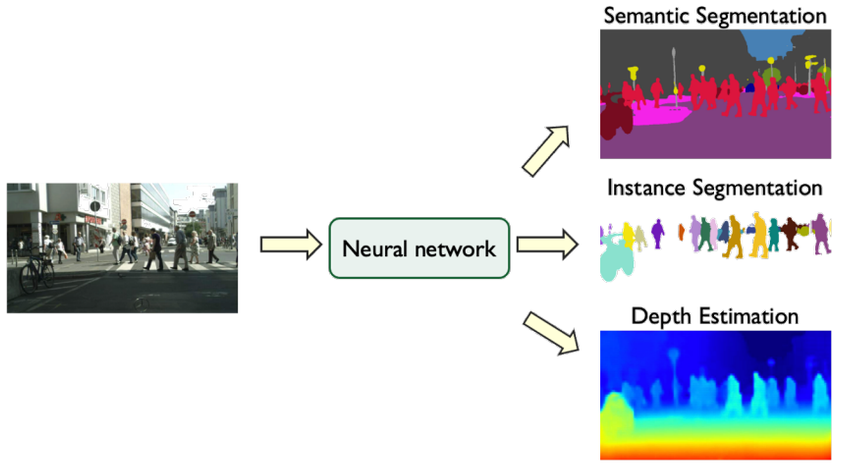
$$L_{total} = \sum_{t=1}^{T} w_t L_t$$

- L_t can be one of the following:
  - L1 Loss
  - Mean Squared Error (MSE)
  - Cross-entropy Loss
  - Custom Loss
  - …
- Weights (w_t) can be:
  - All equal
  - Task-specific
  - Based on uncertainty
  - Dynamic
  - …

Data Sampling Approaches

- The number of labelled instances may varies across tasks.
- Solution: “mask” labelled data points when computing the loss function, meaning that we do not consider unlabelled outputs when calculating the loss function.

$$L_{total} = \sum_{t=1}^{T} w_t L_t$$



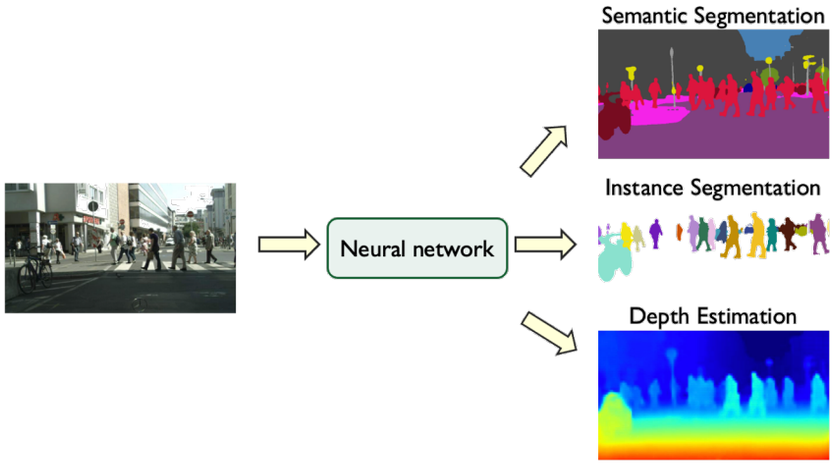
Data Sampling Approaches

- The number of labelled instances may varies across tasks.
- Solution: "mask" labelled data points when computing the loss function, meaning that we do not consider unlabelled outputs when calculating the loss function.

$$L_{total} = \sum_{t=1}^{T} w_t \underbrace{L_t}$$

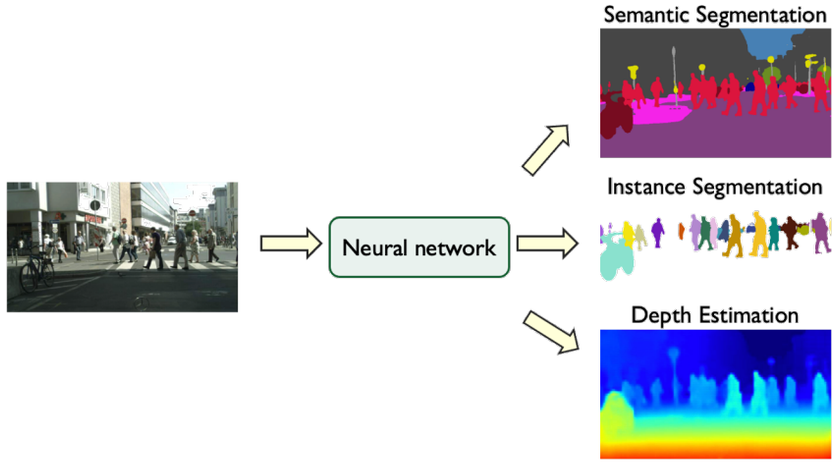
$$L_t = \frac{1}{N} \sum_{i=1}^{N} \underbrace{\delta_{t,i}} l_t(\hat{y}_t^i, y_t^i)$$

$\delta_{t,i} \in \{0, 1\}$ denotes the availability of labelled data for example i and task t

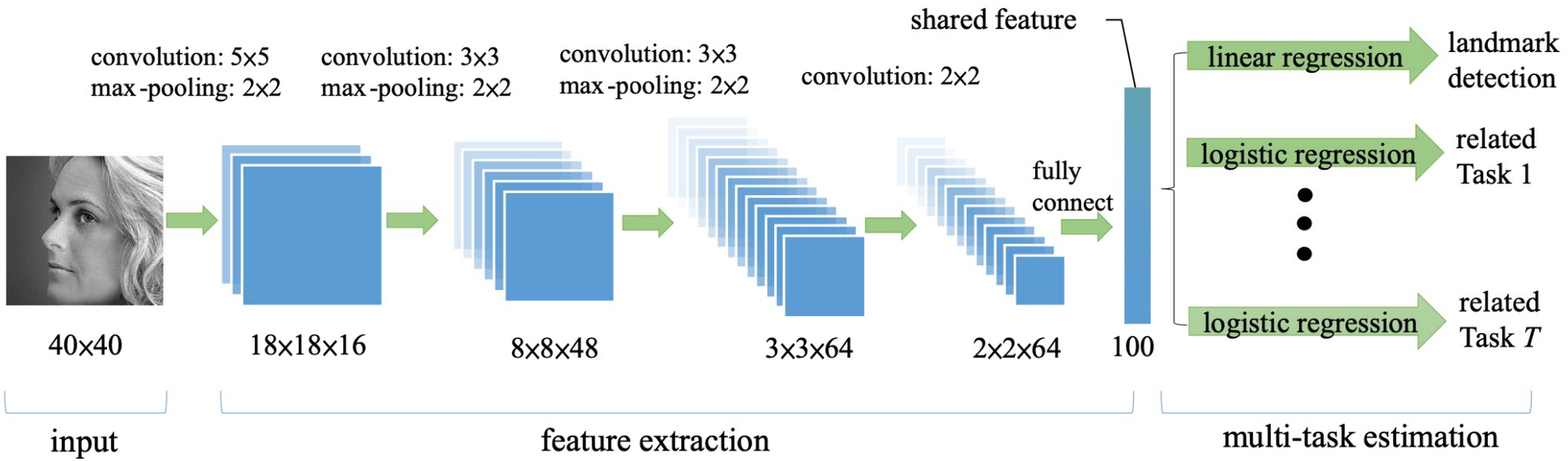


Data Sampling Approaches

- To determine the number of instances to draw from each dataset:
  - Uniform Sampling:
    - The number of instances sampled uniformly for a task is limited by the task with the smallest number of labels.
  - Size-dependent Sampling:
    - The number of instances sampled is proportional to the number of labelled data in each task.
  - Dynamic Sampling:
    - The number of instances sampled is higher for tasks that need more training and smaller for tasks that have converged

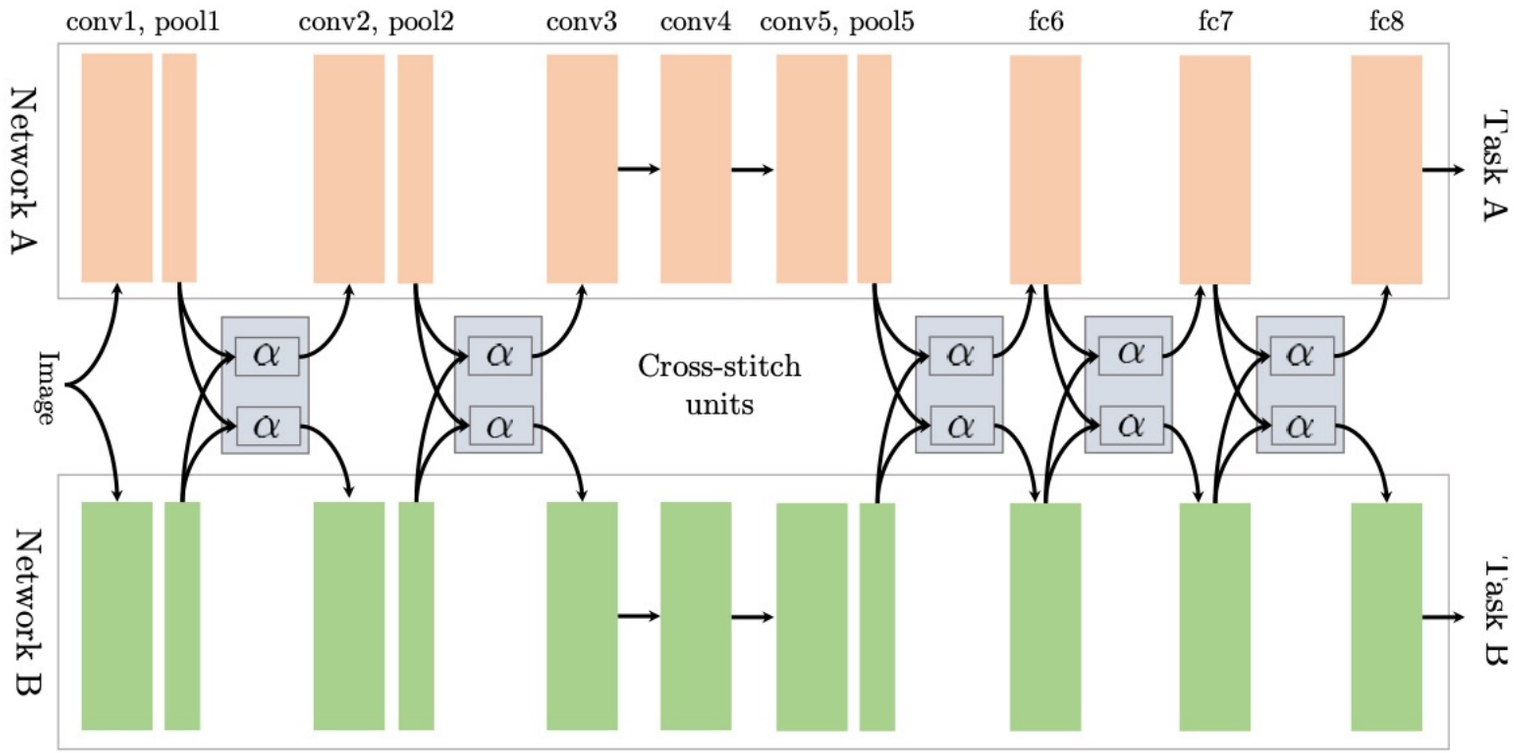


Some MTL Architectures – Shared Trunk



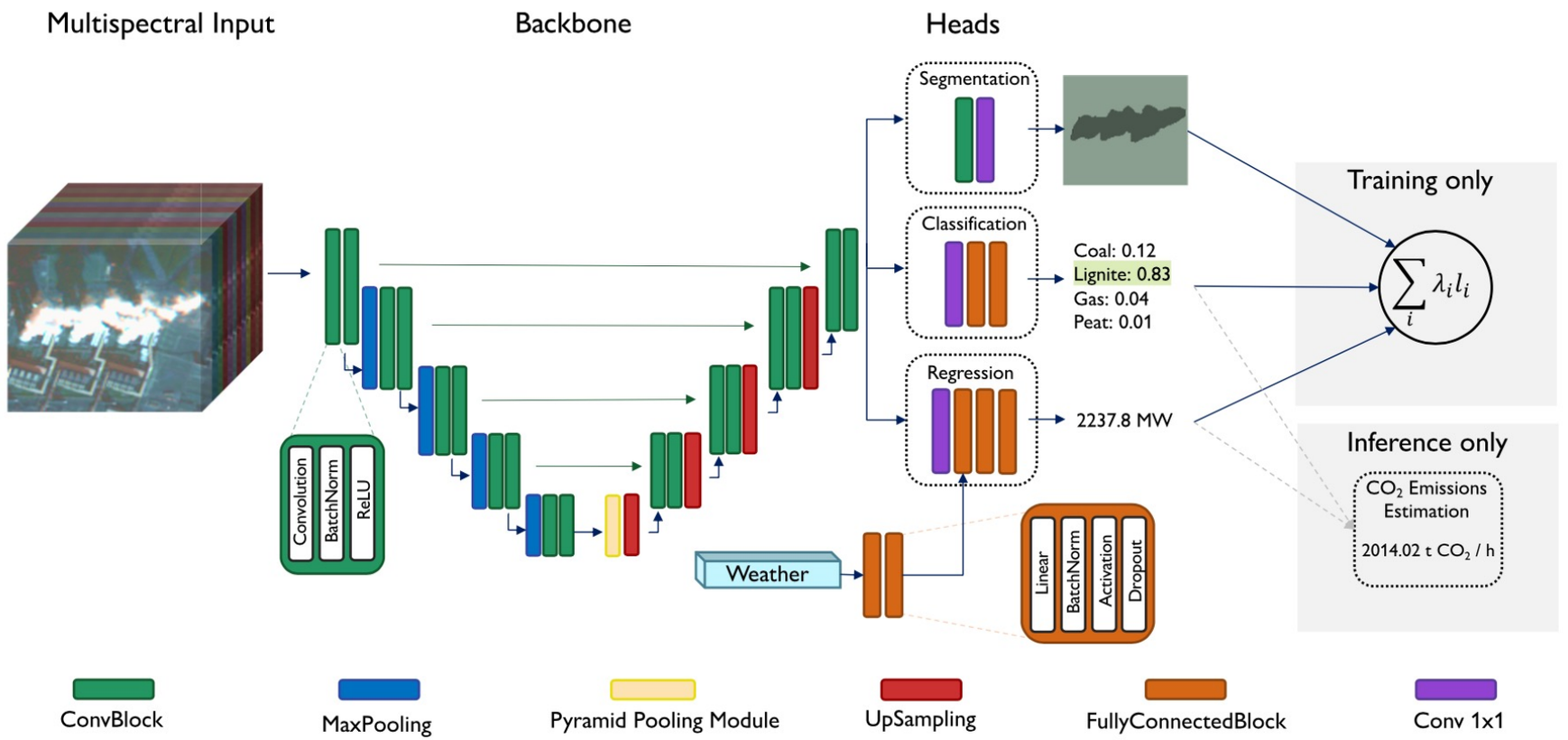
Zhang, Zhanpeng, Ping Luo, Chen Change Loy and Xiaoou Tang. "Facial Landmark Detection by Deep Multi-task Learning." *European Conference on Computer Vision* (2014).

Some MTL Architectures – Cross-Talk



Misra, Ishan, Abhinav Shrivastava, Abhinav Kumar Gupta and Martial Hebert. "Cross-Stitch Networks for Multi-task Learning." *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016): 3994-4003.

Some MTL Architectures – Earth Observation



Hanna, Joelle, Damian Borth and Michael Mommert. “Physics-Guided Multitask Learning for Estimating Power Generation and CO2 Emissions From Satellite Imagery.” *IEEE Transactions on Geoscience and Remote Sensing* 61 (2023): 1-12.

And now, let's build and train a multi-task model !