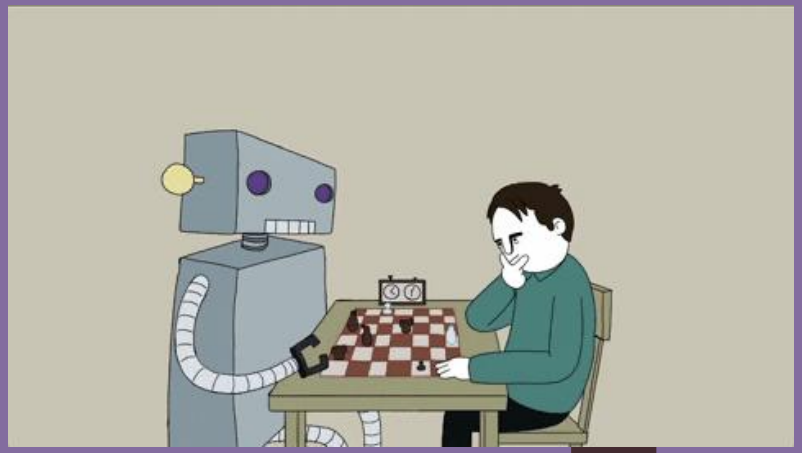
CSE4006
DEEP LEARNING



Dr K G Suma
Associate Professor
School of Computer Science and Engineering

Module No. 6
VAEs and GANS
9 Hours

- Variational Autoencoders
- Generative Adversarial Networks
- Multi-task Deep Learning
- Multi-view Deep Learning

- Various Applications - speech, text, image and video

Multi-Task Learning(MTL) for Deep Learning

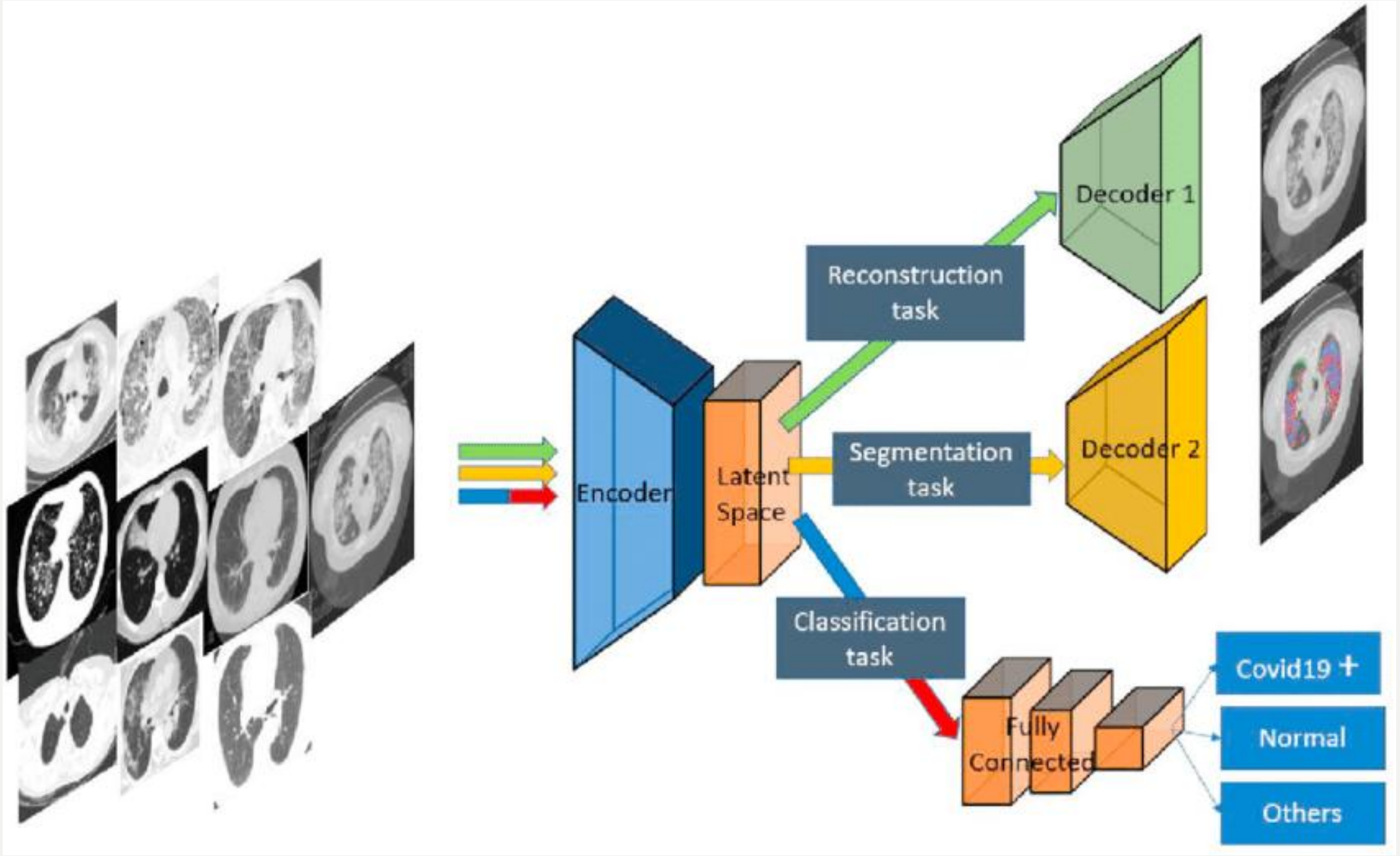
Multi-Task Learning (MTL) is a type of Deep learning technique where a model is trained to perform multiple tasks simultaneously. In deep learning, MTL refers to training a neural network to perform multiple tasks by sharing some of the network's layers and parameters across tasks.

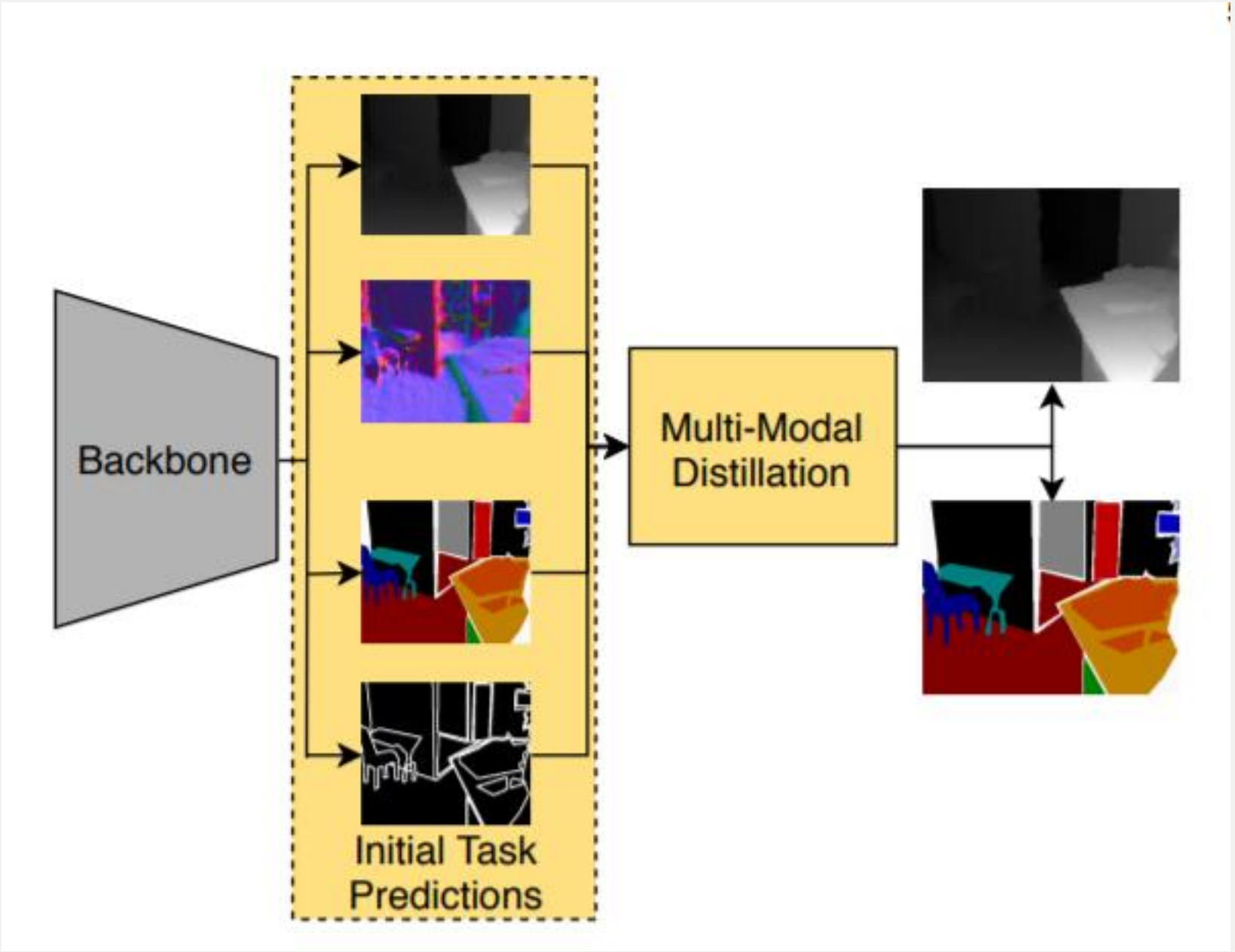
In MTL, the goal is to improve the generalization performance of the model by leveraging the information shared across tasks. By sharing some of the network's parameters, the model can learn a more efficient and compact representation of the data, which can be beneficial when the tasks are related or have some commonalities.

Multi-Task Learning(MTL) for Deep Learning

There are different ways to implement MTL in deep learning, but the most common approach is to use a **shared feature extractor** and multiple **task-specific heads**. The shared feature extractor is a part of the network that is shared across tasks and is used to extract features from the input data. The task-specific heads are used to make predictions for each task and are typically connected to the shared feature extractor.

Another approach is to use a **shared decision-making layer**, where the decision-making layer is shared across tasks, and the task-specific layers are connected to the shared decision-making layer.


Decoder 1
Reconstruction task
Encoder
Latent Space
Segmentation task
Decoder 2
Classification task
Fully Connected
Covid19 +
Normal
Others


Backbone
Initial Task Predictions
Multi-Modal Distillation

Applications - MTL for Deep Learning

MTL can be useful in many applications such as natural language processing, computer vision, and healthcare, where multiple tasks are related or have some commonalities. It is also useful when the data is limited, MTL can help to improve the generalization performance of the model by leveraging the information shared across tasks.

Applications:

- Object detection and Facial recognition
- Self Driving Cars: Pedestrians, stop signs and other obstacles can be detected together
- Multi-domain collaborative filtering for web applications
- Stock Prediction
- Language Modelling and other NLP applications

However, MTL also has its own limitations, such as when the tasks are very different

Multi-Task Learning(MTL) for Deep Learning

- Formally, if there are **n** tasks (conventional deep learning approaches aim to solve just 1 task using 1 particular model), where these **n** tasks or a subset of them are related to each other but not exactly identical, Multi-Task Learning **(MTL)** will help in improving the learning of a particular model by using the knowledge contained in all the n tasks.

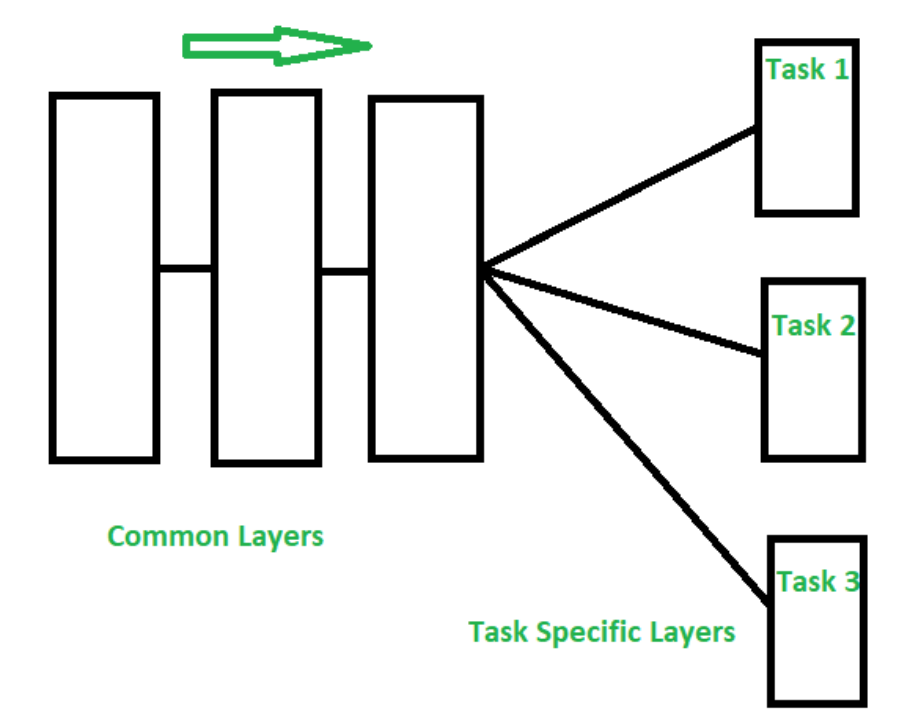
- **Intuition behind Multi-Task Learning (MTL):** By using Deep learning models, we usually aim to learn a good representation of the features or attributes of the input data to predict a specific value. Formally, we aim to optimize for a particular function by training a model and fine-tuning the hyperparameters till the performance can't be increased further. By using MTL, it might be possible to increase performance even further by forcing the model to learn a more generalized representation as it learns (updates its weights) not just for one specific task but a **bunch of tasks**. Biologically, humans learn in the same way. We learn better if we learn multiple related tasks instead of focusing on one specific task for a long time.

Multi-Task Learning(MTL) for Deep Learning

- **MTL as a regularizer:** In the lingo of Machine Learning, MTL can also be looked at as a way of inducing bias. It is a form of inductive transfer, using multiple tasks induces a bias that prefers hypotheses that can explain all the **n** tasks. MTL acts as a regularizer by introducing inductive bias as stated above. It significantly reduces the risk of overfitting and also reduces the model's ability to accommodate random noise during training.

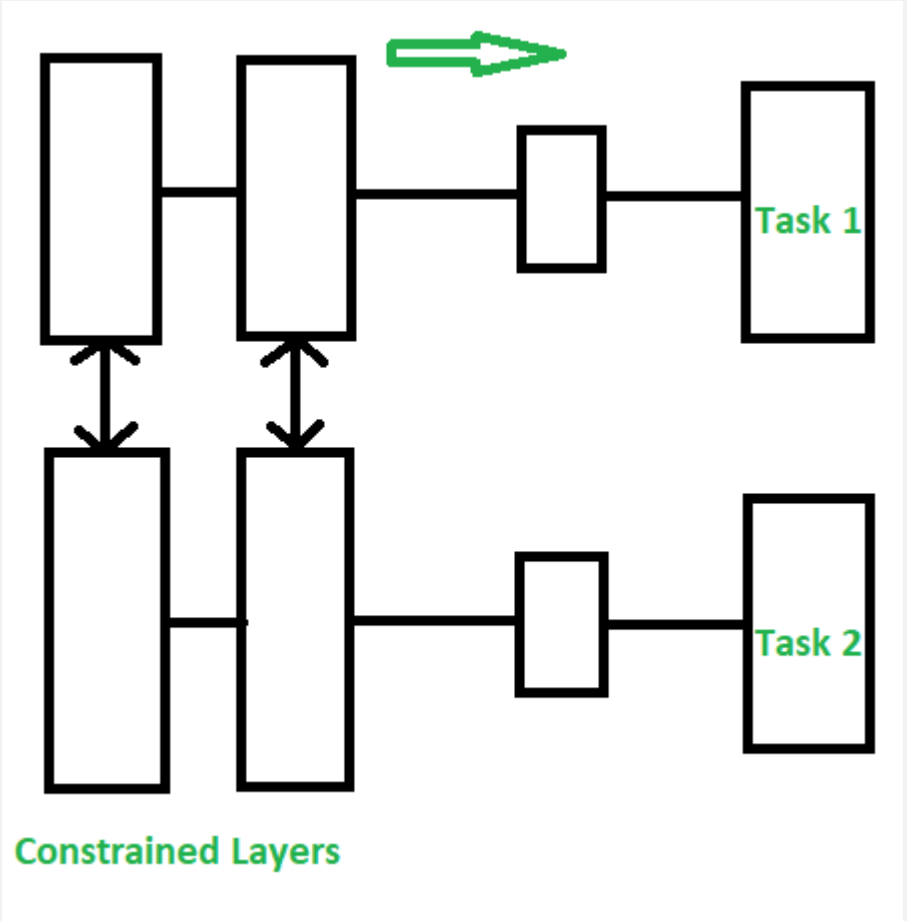
Multi-Task Learning(MTL) for Deep Learning

Hard Parameter Sharing – A common hidden layer is used for all tasks but several task specific layers are kept intact towards the end of the model. This technique is very useful as by learning a representation for various tasks by a common hidden layer, we reduce the risk of overfitting.



Multi-Task Learning(MTL) for Deep Learning

- **Soft Parameter Sharing** – Each model has their own sets of weights and biases and the distance between these parameters in different models is regularized so that the parameters become similar and can represent all the tasks.



Multi-Task Learning(MTL) for Deep Learning

LIMITATION:

Assumptions and Considerations – Using MTL to share knowledge among tasks are very useful only when the tasks are very similar, but when this assumption is violated, the performance will significantly decline.

Multi-Task Learning(MTL) - Implementation

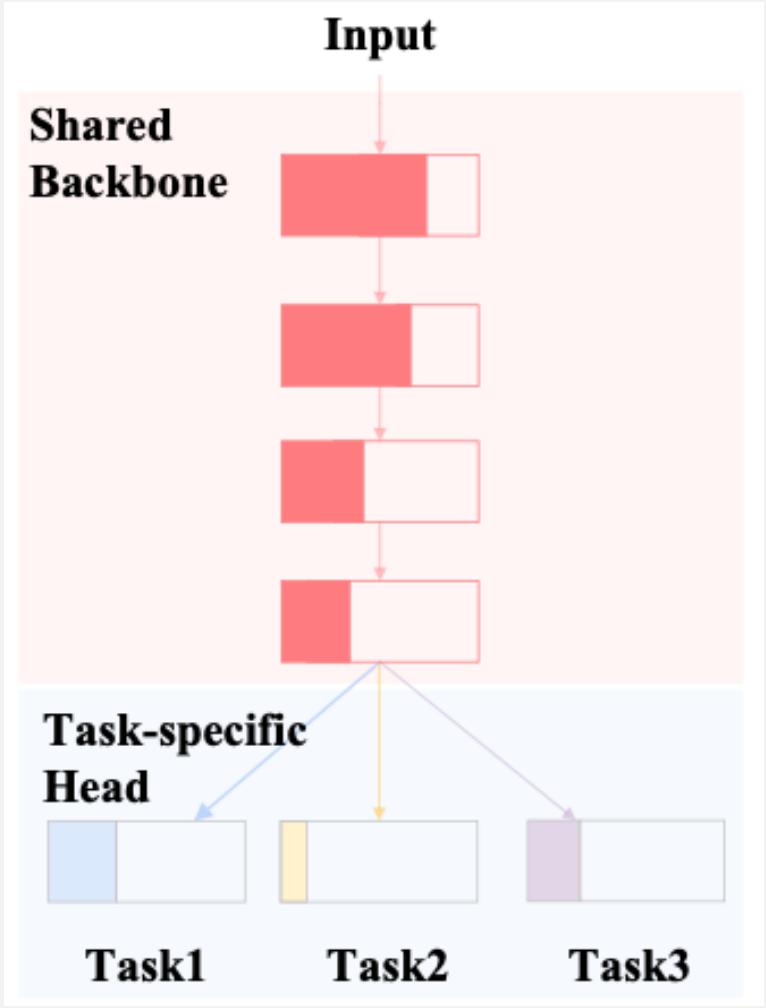
Here are some important points to consider when implementing Multi-Task Learning (MTL) for deep learning:

1. **Task relatedness:** MTL is most effective when the tasks are related or have some commonalities, such as natural language processing, computer vision, and healthcare.

2. **Data limitation:** MTL can be useful when the data is limited, as it allows the model to leverage the information shared across tasks to improve the generalization performance.

3. **Shared feature extractor:** A common approach in MTL is to use a shared feature extractor, which is a part of the network that is shared across tasks and is used to extract features from the input data.

4. **Task-specific heads:** Task-specific heads are used to make predictions for each task and are typically connected to the shared feature extractor.


Input
Shared
Backbone
Task-specific
Head
Task1
Task2
Task3

Multi-Task Learning(MTL) - Implementation

6. **Shared decision-making layer:** another approach is to use a shared decision-making layer, where the decision-making layer is shared across tasks, and the task-specific layers are connected to the shared decision-making layer.

7. **Careful architecture design:** The architecture of MTL should be carefully designed to accommodate the different tasks and to make sure that the shared features are useful for all tasks.

8. **Overfitting:** MTL models can be prone to overfitting if the model is not regularized properly.

9. **Avoiding negative transfer:** when the tasks are very different or independent, MTL can lead to suboptimal performance compared to training a single-task model. Therefore, it is important to make sure that the shared features are useful for all tasks to avoid negative transfer.