Continual Learning

COMPSCI 760
2024 Semester 1

Olivier Graffeuille
ogra439@aucklanduni.ac.nz

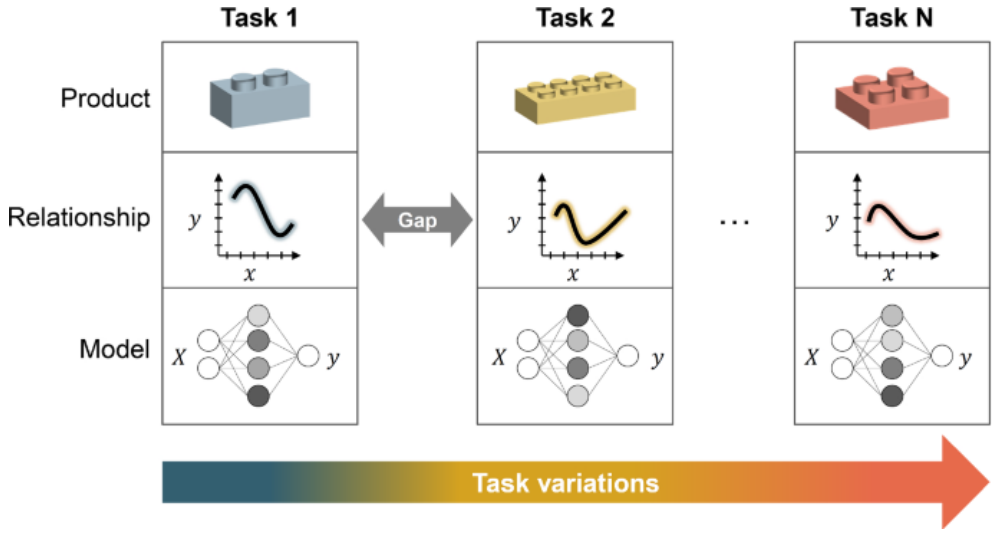
Outline

1. Multi-Task Learning
2. ***Settings for Continual Learning***
3. Techniques for Deep Continual Learning
    1. Replay-based
    2. Regularisation-based
    3. Architecture-based

Continual Learning

A.k.a. Lifelong Learning, Incremental Learning

To learn from a stream of data with changing data distribution, with the goal of transferring knowledge between tasks.



Tercan, H., Deibert, P., & Meisen, T. (2022). Continual learning of neural networks for quality prediction in production using memory aware synapses and weight transfer. *Journal of Intelligent Manufacturing*, *33*(1), 283-292.

Continual Learning

A.k.a. Lifelong Learning, Incremental Learning

To learn from a stream of data with changing data distribution, with the goal of transferring knowledge between tasks.

- Forward transfer
  - Knowledge learnt during one task affects learning of future tasks
- Backward transfer
  - Knowledge learnt during one task affects learning of previous tasks
    - *Catastrophic Forgetting* is negative backwards transfer

Catastrophic Forgetting

- We want to optimise performance on all tasks
- We only have access to the data of the current task
- When optimising for this current task, we will deteriorate performance on old tasks!

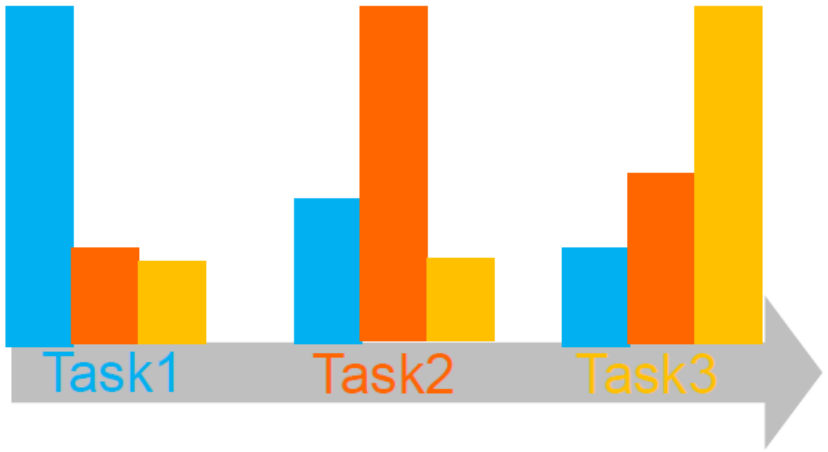
Overall Goal:

$$\sum_{t=1}^{\mathcal{T}} \mathop{\mathbb{E}}_{(\mathcal{X}^{(t)},\mathcal{Y}^{(t)})} [\mathcal{L}(f_t(\mathcal{X}^{(t)};\theta),\mathcal{Y}^{(t)})$$

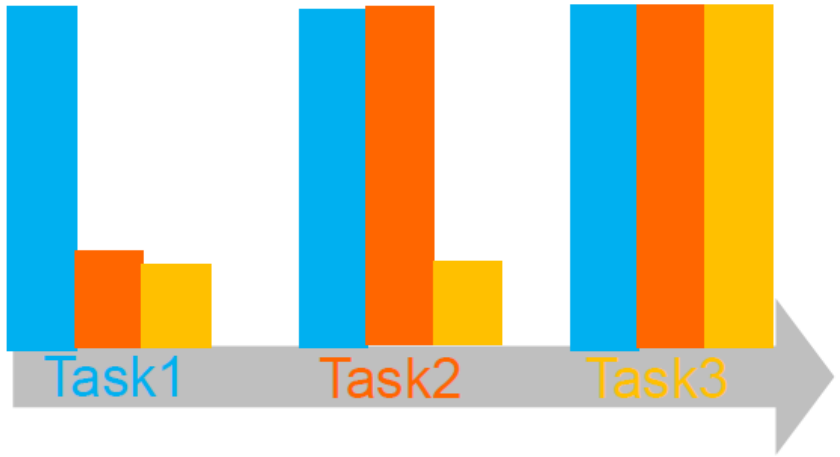
For current task:

$$\frac{1}{N_{\mathcal{T}}} \sum_{i=1}^{N_{\mathcal{T}}} \ell(f(x_i^{(\mathcal{T})};\theta), y_i^{(\mathcal{T})}) \ .$$

Catastrophic Forgetting



Catastrophic Forgetting

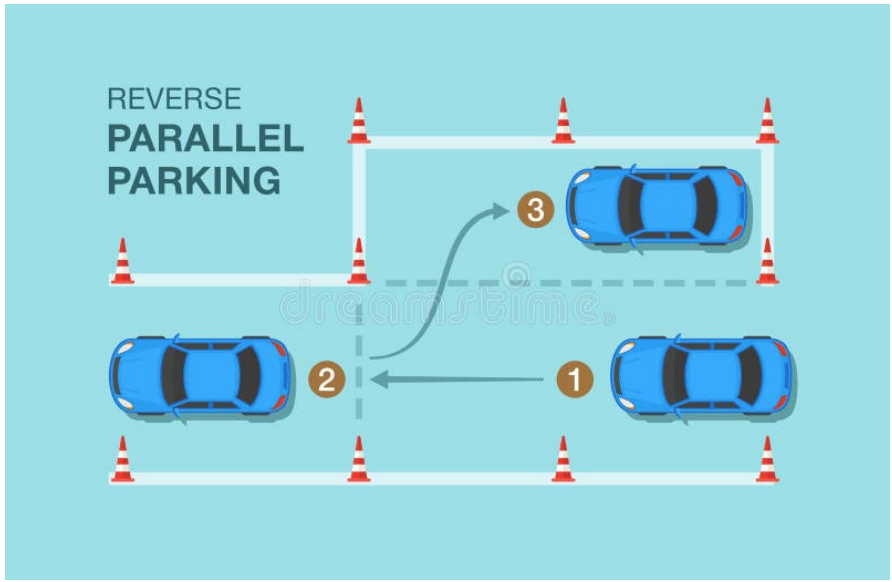


No Forgetting

Catastrophic Forgetting: Intuition

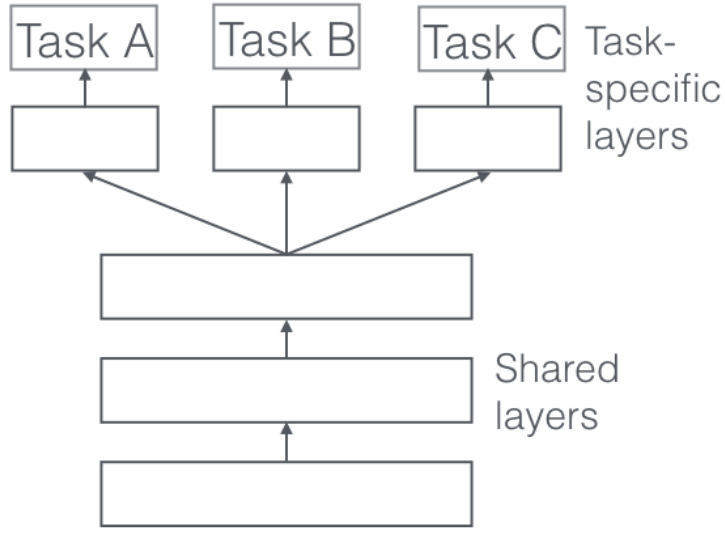
Skills learnt playing one activity may transfer to a future activity

https://www.griffinsgamingguides.com/grand-theft-auto-v-show-off-trophy-guide/



Related areas of research

- Transfer Learning
  - Transfer from source to target domain
- Multi-Task Learning
  - Transfer from multiple domains to multiple other domains
  - Can be though of as the “upper bound” of continual learning
- Domain Generalisation/Adaptation
  - Transfer across different domains
- Similar Goals, Different Settings -> Different Techniques



Multi-Task Learning Architecture

https://www.ruder.io/multi-task/

Related areas of research

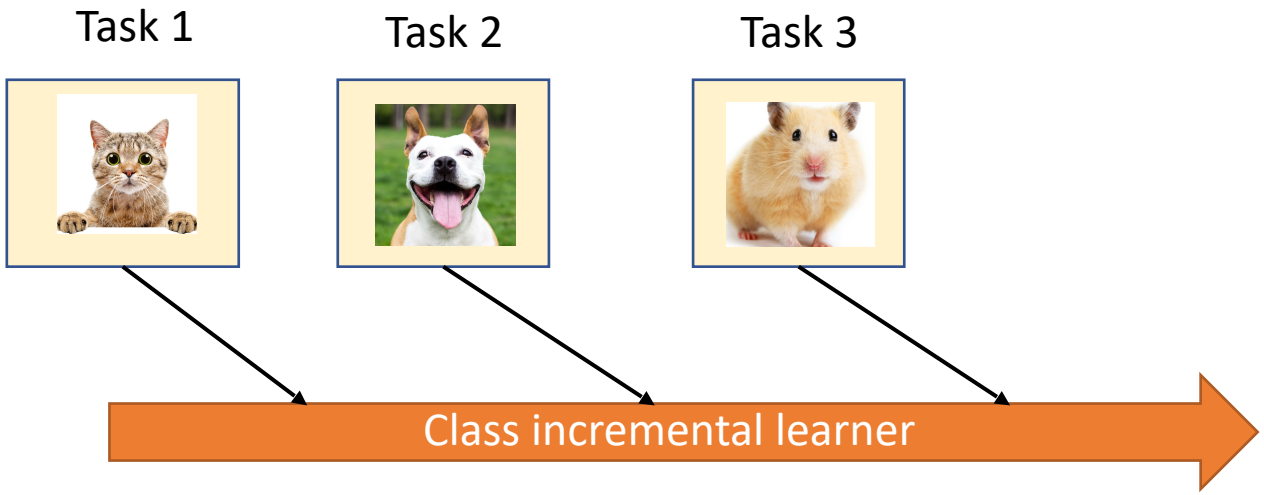
- Continual learning vs. Data Streams
  - Generally more complex data
    - Larger models
  - More of a focus on avoiding forgetting than adapting to changing distributions
  - More of a focus on how knowledge is stored in NNs and how it can be transferred across tasks

Settings of Continual Learning

- Class-Incremental Learning
  - Receive more and more training classes over time
    e.g. medical image classification, where new pathologies are discovered
- Task-Incremental Learning
  - Receive sequence of tasks, one at a time
    e.g. robotics, where a robot must learn a new function
- Domain-Incremental Learning
  - Receive sequence of domains, one at a time. Domain-ID not available in testing.
    e.g. classify objects in new environments, classify reviews of different products

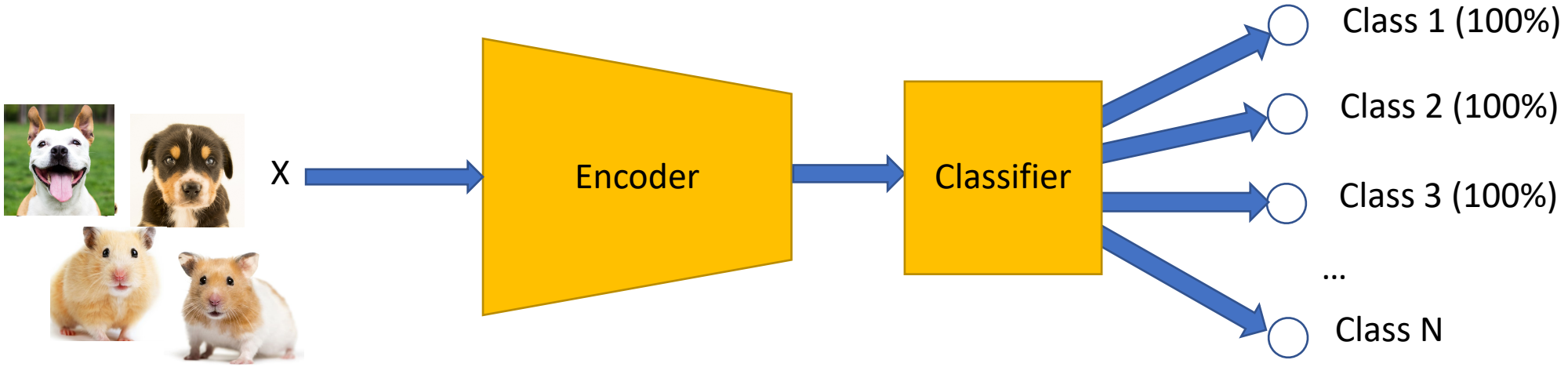
Settings of Continual Learning

- Class-Incremental Learning
  - Receive more and more training classes over time
    e.g. medical image classification, where new pathologies are discovered



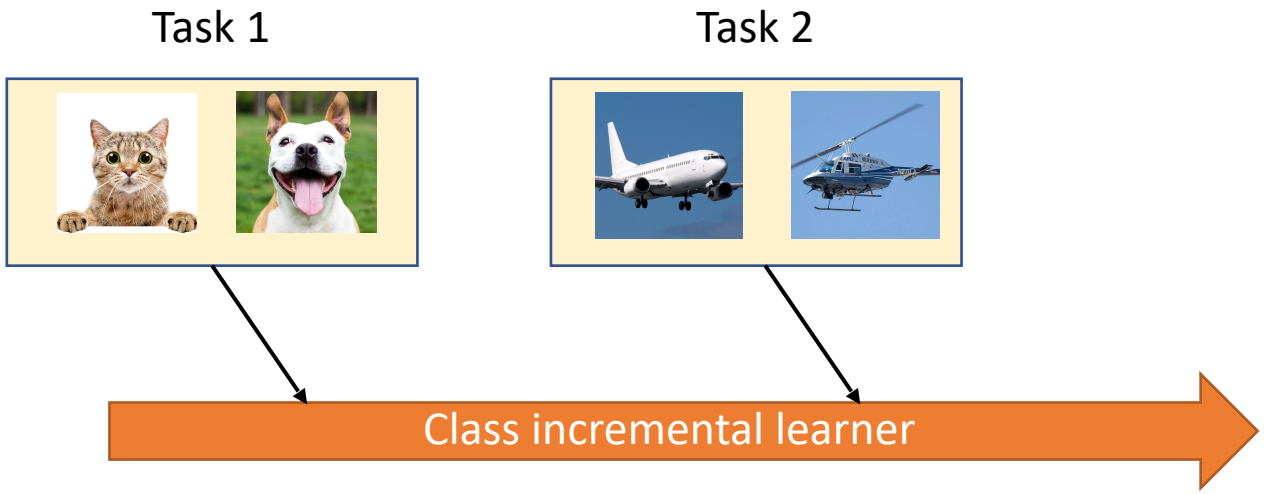
Class-Incremental Learning Model

- What happens if we don't replay?



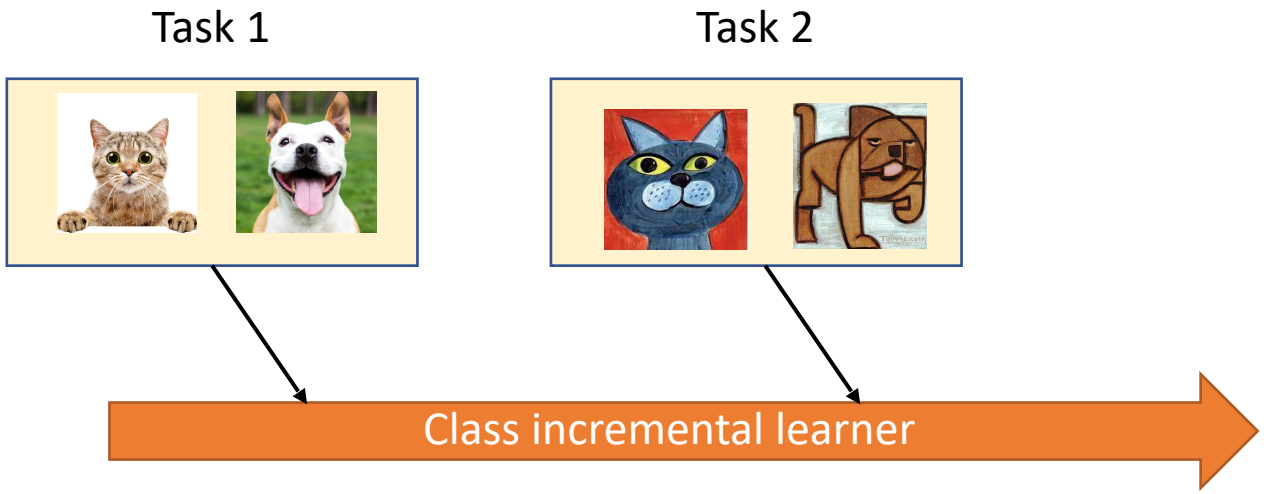
Settings of Continual Learning

- Task-Incremental Learning
  - Receive sequence of tasks, one at a time
    e.g. robotics, where a robot must learn a new function



Settings of Continual Learning

- Domain-Incremental Learning
  - Receive sequence of domains, one at a time. Domain-ID not available in testing.
    e.g. classify pets in new medium, classify reviews of different products



Settings of Continual Learning

- Batch Continual Learning
  - *When a new task arrives, all its training data are available over any number of epochs*
- Online Continual Learning
  - *The data comes in a data stream, examples cannot be revisited*

Outline

1. Multi-Task Learning
2. Settings for Continual Learning
3. ***Techniques for Deep Continual Learning***
    1. Replay-based
    2. Regularisation-based
    3. Architecture-based

Lifelong Sentiment Classification

"The phone case arrived on time, but it's an ugly green colour! Also it's quite slippery."

- Uses a Naïve Bayesian Text Classifier

$$P(w|c_j) = \frac{\lambda + N_{c_j,w}}{\lambda |V| + \sum_{v=1}^{|V|} N_{c_j,v}}$$

- For each task, store word knowledge:
  - Class conditional probability
  - Class counts (priors)

Chen, Z., Ma, N., & Liu, B. (2018). Lifelong learning for sentiment classification. arXiv preprint arXiv:1801.02808.

Lifelong Sentiment Classification

- Use historical knowledge to determine reliable words to classify current task

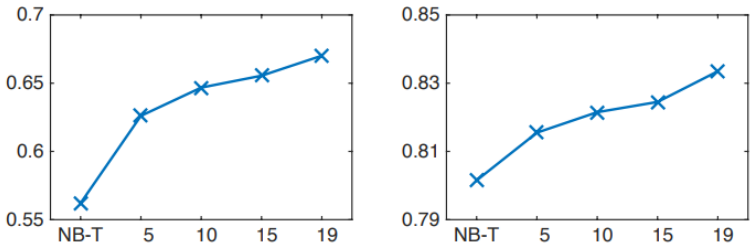
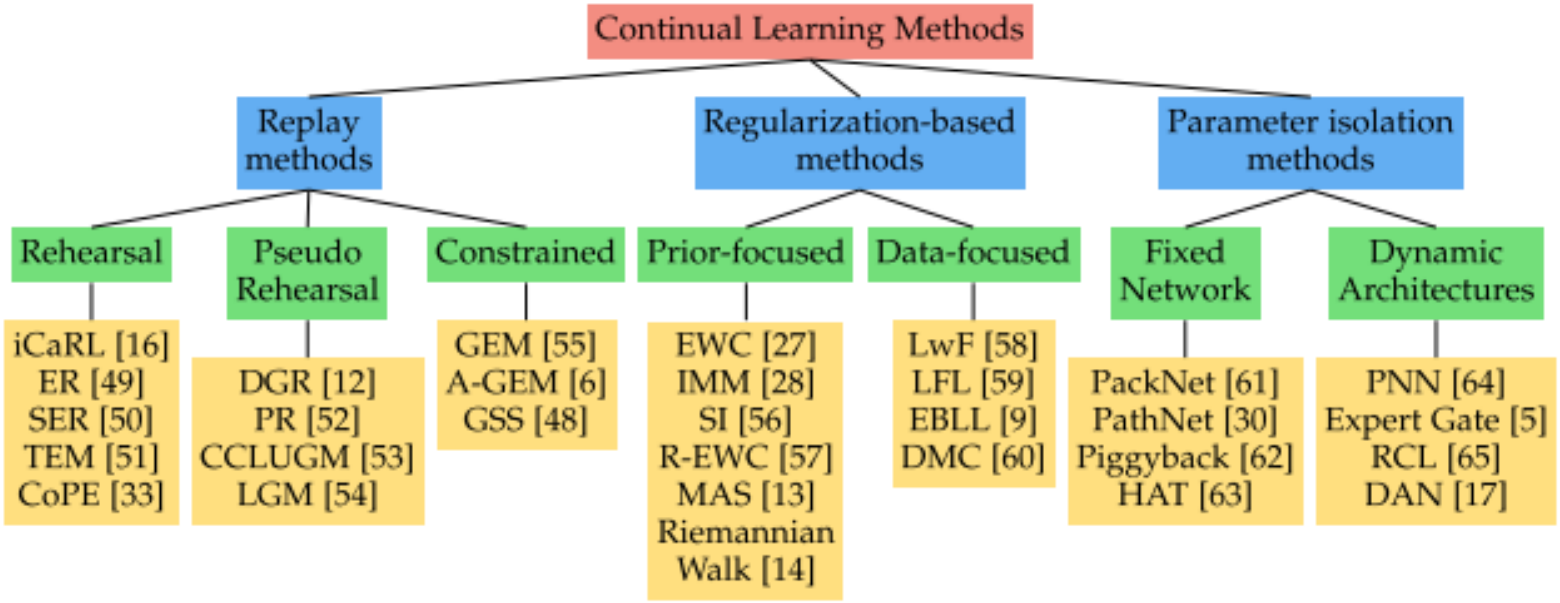


Figure 1: (Left): Negative class F1-score of LSC with #past domains in natural class distribution. (Right): Accuracy of LSC with #past domains in balanced class distribution.

Chen, Z., Ma, N., & Liu, B. (2018). Lifelong learning for sentiment classification. arXiv preprint arXiv:1801.02808.

Overview of Deep Continual Learning



De Lange, M., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., ... & Tuytelaars, T. (2021). A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence*, *44*(7), 3366-3385.

Outline

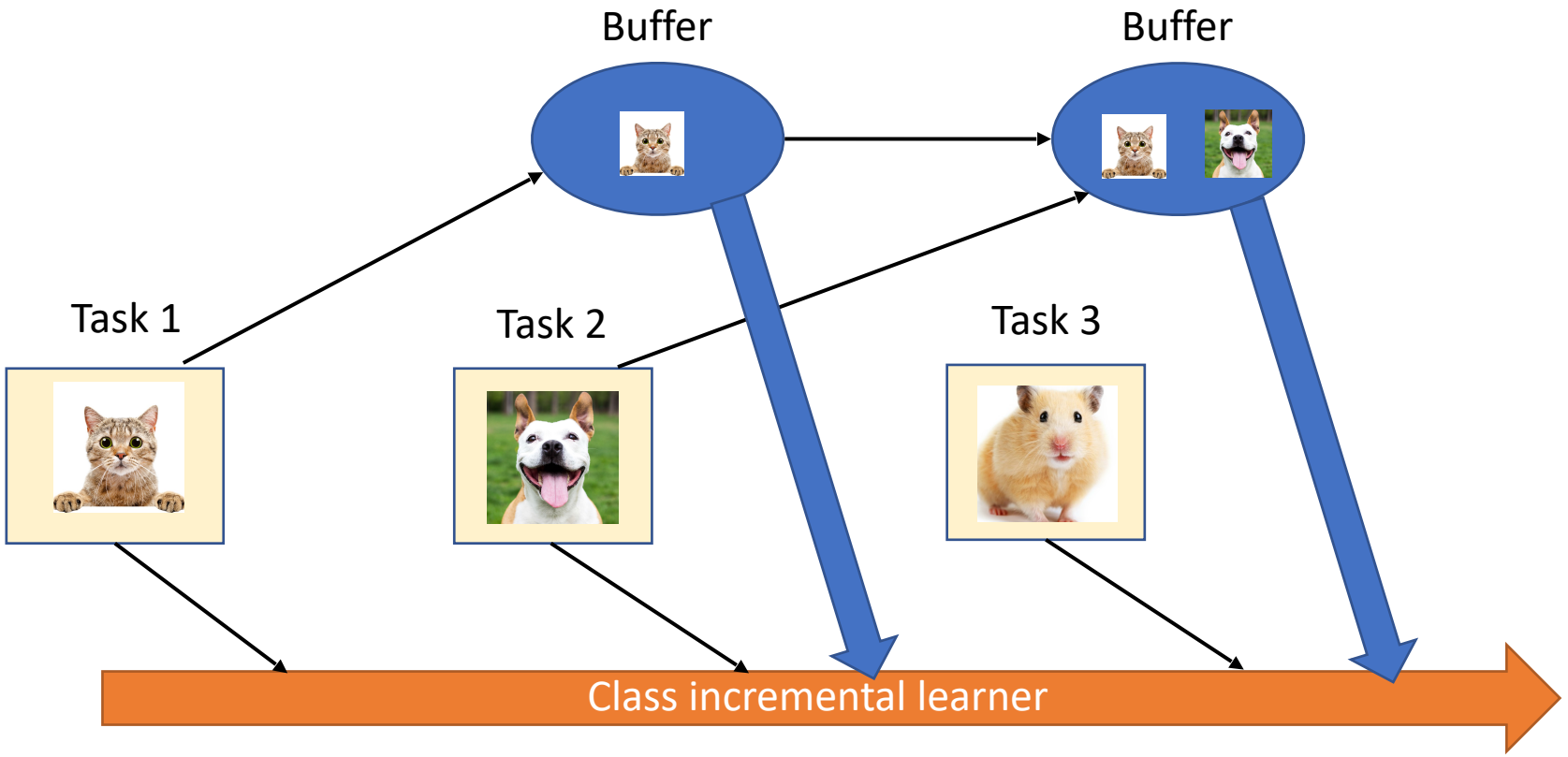
1. Multi-Task Learning
2. Settings for Continual Learning
3. Techniques for Deep Continual Learning
    1. ***Replay-based***
    2. Regularisation-based
    3. Architecture-based

Replay Methods

The idea:

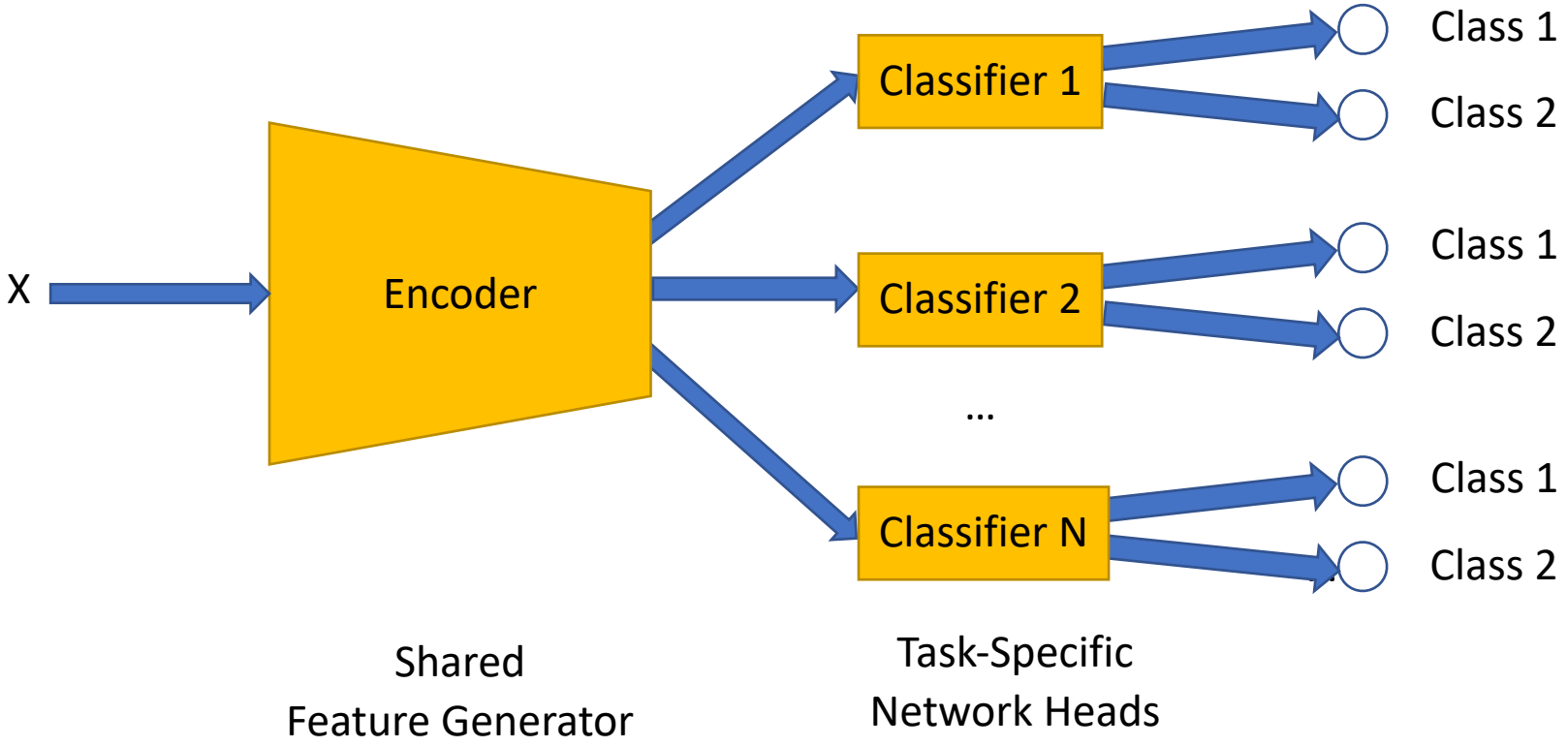
Replay samples of old tasks when learning new tasks,
to defy forgetting.

Replay Methods



Task-Incremental Learning Model

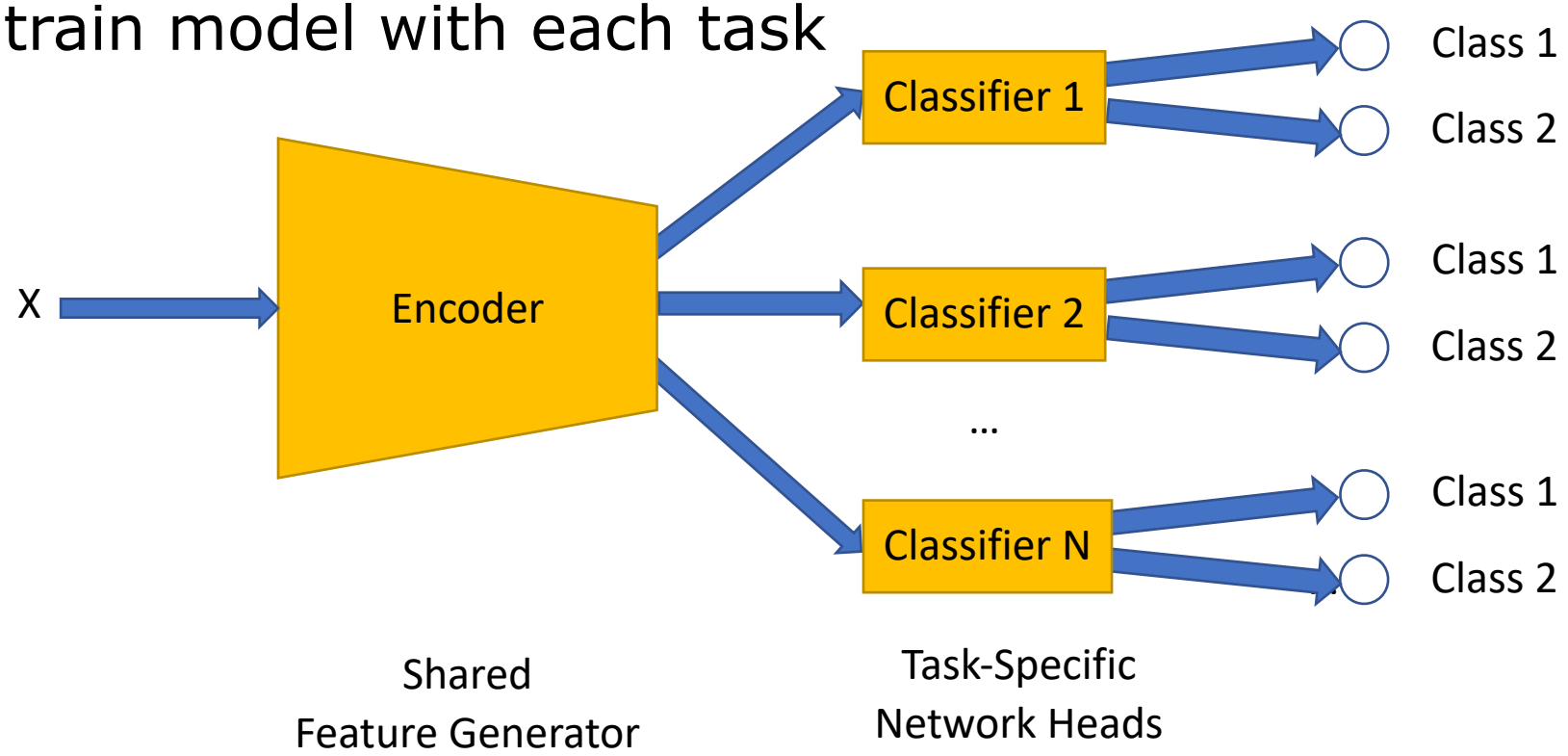
Designed for the Task Incremental Learning setting:



Lopez-Paz, D., & Ranzato, M. A. (2017). Gradient episodic memory for continual learning. *Advances in neural information processing systems*, *30*.

Can we think of a Trivial Solution?

- Store all previous data
- Re-train model with each task



Lopez-Paz, D., & Ranzato, M. A. (2017). Gradient episodic memory for continual learning. *Advances in neural information processing systems*, *30*.

Gradient Episodic Memory (GEM)

The idea:

We can’t re-train on stored instances from previous task, otherwise we would overfit

So, we constrain the performance on these instances instead!

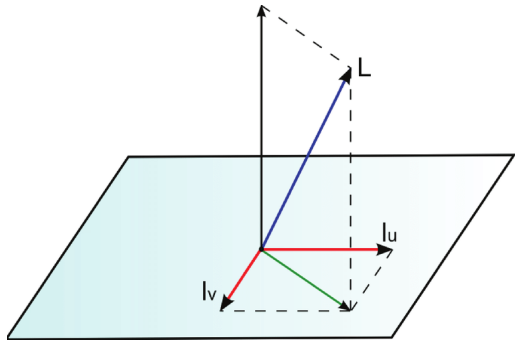
Lopez-Paz, D., & Ranzato, M. A. (2017). Gradient episodic memory for continual learning. *Advances in neural information processing systems*, *30*.

Gradient Episodic Memory (GEM)

- Solve the following constrained optimisation problem

$$\begin{aligned} \text{minimize}_\theta \quad & \ell(f_\theta(x,t),y) \\ \text{subject to} \quad & \ell(f_\theta,\mathcal{M}_k) \leq \ell(f_\theta^{t-1},\mathcal{M}_k) \text{ for all } k<t, \end{aligned}$$

- We can't directly use SGD algorithms (unconstrained)
  - We project the gradient on the feasible region of previous task gradients



Lopez-Paz, D., & Ranzato, M. A. (2017). Gradient episodic memory for continual learning. *Advances in neural information processing systems*, *30*.

Gradient Episodic Memory (GEM)

- This method can allow for backwards transfer?

$$\textbf{Backward Transfer: BWT} = \frac{1}{T-1}\sum_{i=1}^{T-1} R_{T,i} - R_{i,i}$$

$$\textbf{Forward Transfer: FWT} = \frac{1}{T-1}\sum_{i=2}^{T} R_{i-1,i} - \bar{b}_i.$$

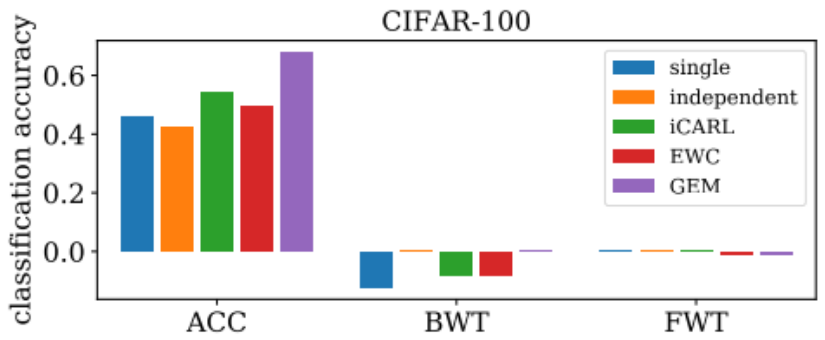
Where $R_{T,i}$ is the accuracy of task i after the model has trained T tasks

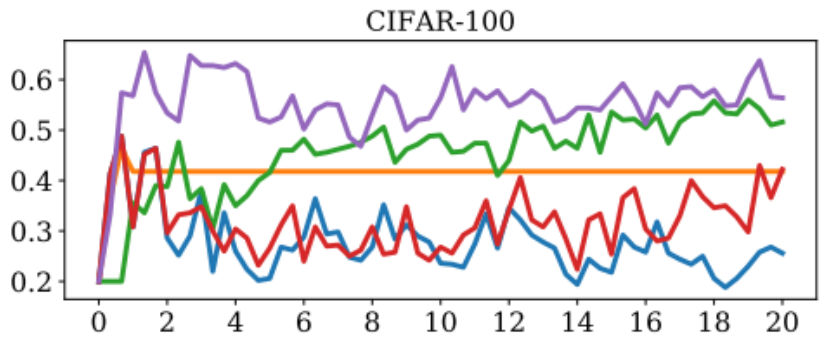
Lopez-Paz, D., & Ranzato, M. A. (2017). Gradient episodic memory for continual learning. *Advances in neural information processing systems*, *30*.

Gradient Episodic Memory (GEM)

$$\textbf{Backward Transfer: BWT} = \frac{1}{T-1}\sum_{i=1}^{T-1} R_{T,i} - R_{i,i}$$

$$\textbf{Forward Transfer: FWT} = \frac{1}{T-1}\sum_{i=2}^{T} R_{i-1,i} - \bar{b}_i.$$





Lopez-Paz, D., & Ranzato, M. A. (2017). Gradient episodic memory for continual learning. *Advances in neural information processing systems*, *30*.

Outline

1. Multi-Task Learning
2. Settings for Continual Learning
3. Techniques for Deep Continual Learning
    1. Replay-based
    2. ***Regularisation-based***
    3. Architecture-based

Regularisation-Based Methods

- Use a regularisation term in the loss function to consolidate previous learning.
    - Loss + penalty term
- Pros
    - Doesn’t need to store previous instances – privacy

Weight Prior Regularisation

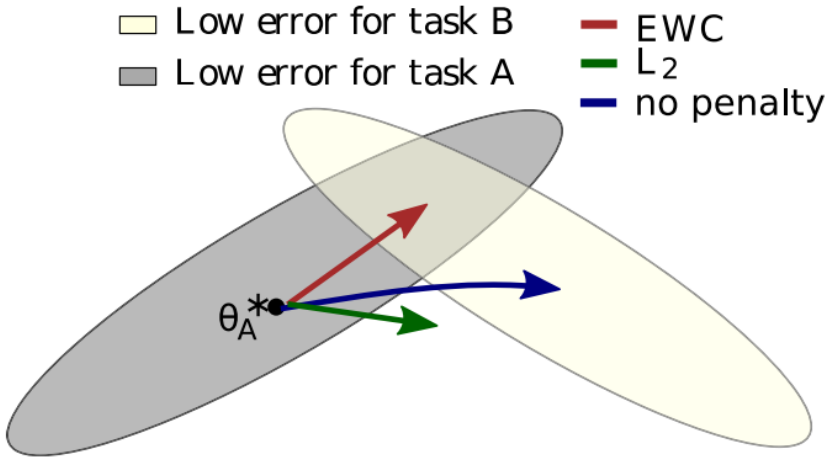
- One approach is to regularise network parameters based on a prior, to maintain consistency.
- Can we think of a naïve approach?
  - L2 Regularisation?

$$\mathcal{L}(\theta) = \mathcal{L}_B(\theta) + \sum_i (\theta_i - \theta^*_{A,i})^2,$$

Elastic Weight Consolidation (EWC)

The idea:

Find parameters important to the performance of previous tasks, and stop these from being changed much



Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., ... & Hadsell, R. (2017). Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, *114*(13), 3521-3526.

Elastic Weight Consolidation (EWC)

- How do we identify important weights?
  - Parameter gradients
    - This logic is applied in other ML fields, including network pruning
  - After training a task, pass data through to identify average gradient magnitude

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., ... & Hadsell, R. (2017). Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, *114*(13), 3521-3526.

Elastic Weight Consolidation (EWC)

- How do we prevent these weights from changing?
  - Penalise change of model parameters, proportional to their importance

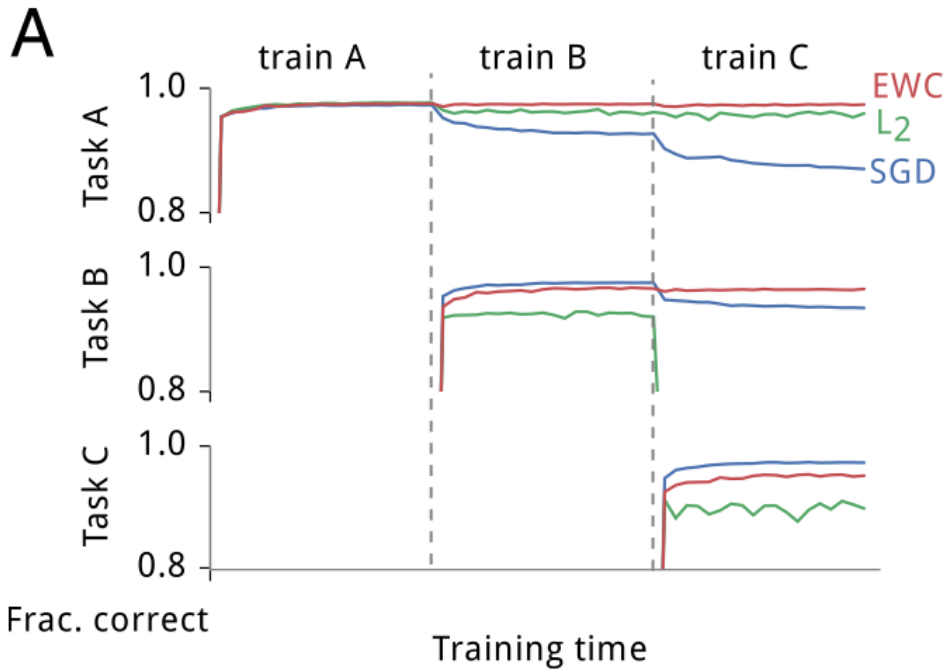
$$\mathcal{L}(\theta) = \mathcal{L}_B(\theta) + \sum_i \frac{\lambda}{2} F_i (\theta_i - \theta^*_{A,i})^2,$$

  - We could alternatively identify parameter importance by sensitivity

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., ... & Hadsell, R. (2017). Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, *114*(13), 3521-3526.

Elastic Weight Consolidation (EWC)

Does it work?



Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., ... & Hadsell, R. (2017). Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, *114*(13), 3521-3526.

Elastic Weight Consolidation (EWC)

Issues with this approach:

- Inefficient with increasing number of tasks
- Soft penalty typically not sufficient to prevent forgetting
    - Imagine randomly changing every parameter in a NN by a tiny amount

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., ... & Hadsell, R. (2017). Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, *114*(13), 3521-3526.

Outline

1. Multi-Task Learning
2. Settings for Continual Learning
3. Techniques for Deep Continual Learning
    1. Replay-based
    2. Regularisation-based
    3. ***Architecture-based***

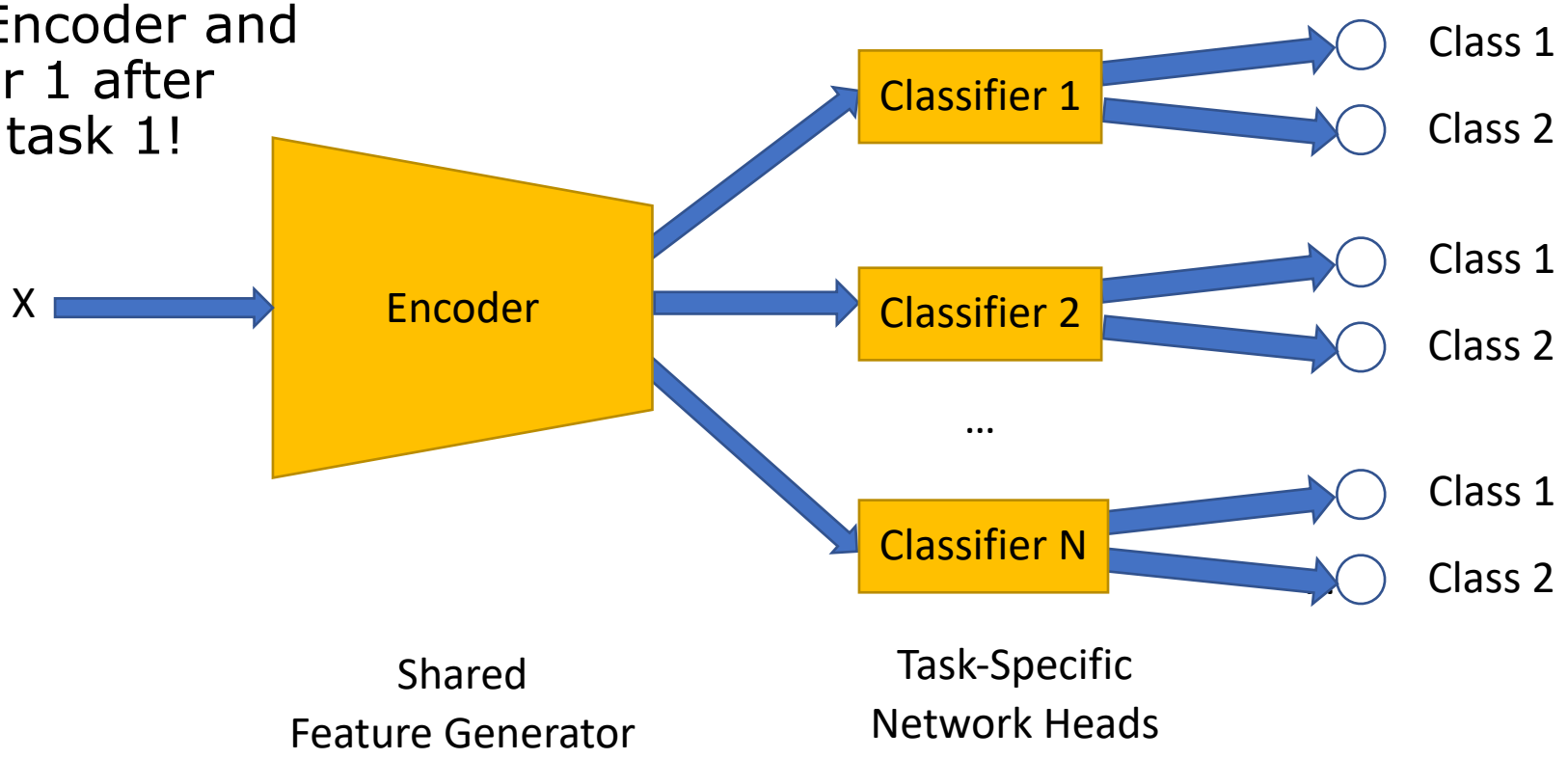
Architecture Based Approaches

- Use different sub-networks for each task.
    - Typically: dedicates different model parameters to each task
- Prevent future tasks from changing previous task weights.
- *This can prevent any possible forgetting*

Architecture Based Approaches

Can we think of a naïve approach?

Freeze Encoder and Classifier 1 after training task 1!



PackNet

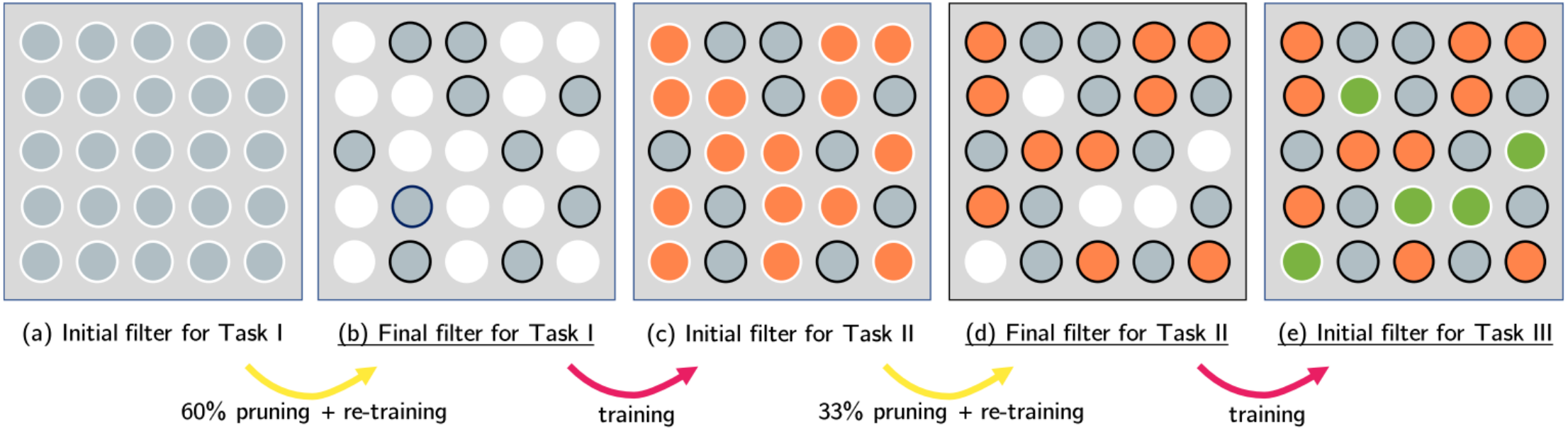
The idea:

For each task, we prune the network weights,
then re-train the model on the new task,
then freeze those task-specific weights.

Mallya, A., & Lazebnik, S. (2018). Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition* (pp. 7765-7773).

PackNet

Colour indicates task
White border indicates not frozen
Black border indicates frozen



Mallya, A., & Lazebnik, S. (2018). Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition* (pp. 7765-7773).

PackNet

What are some problems with this method?
How can we improve this?

Mallya, A., & Lazebnik, S. (2018). Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition* (pp. 7765-7773).

PackNet

- Problem: will run out of model weights!
  - Static architectures:
    - Fixed network size, shared between tasks
  - Dynamic architectures:
    - Grow new parameters, e.g. branches, for new tasks.

Mallya, A., & Lazebnik, S. (2018). Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition* (pp. 7765-7773).

PackNet

- Problem: the way we're selecting sub-networks is random
  - Identify sensible parameters to store

Mallya, A., & Lazebnik, S. (2018). Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition* (pp. 7765-7773).

PackNet

- Problem: memory constraints!
  - Have to store mask for each parameter…
- Solution: store these masks in an intelligent way
- Only $\log_2$(num_tasks) bits per model parameter
  - Empirically, 6% increase memory requirement in VGG-16 experiments

Mallya, A., & Lazebnik, S. (2018). Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition* (pp. 7765-7773).

Conclusions

- Lots of progress to be made…
  - Catastrophic Forgetting is not a solved problem
  - Positive backwards transfer and positive forward transfer are even bigger challenges
  - Lots of practical applications as ML is used in more real-life situations
    - Deployed ML models dealing with changing data
    - Large models we don’t want to re-train
    - Data privacy concerns