Transfer learning and Curriculum learning

Distance and **interactions** between **tasks**

Defining a **geometry** of the space of learning tasks

Antoine Cornuéjols

AgroParisTech – INRAE MIA Paris-Saclay

EKINOCS research group

AgroParisTech

INRAE
la science pour la vie, l'humain, la terre

- Up to now:
  - we were looking at **changes of distribution** between learning and testing
  - Or to **changes of domains** in transfer learning

- Now, we turn to the bias introduced by controlling the **order** of the **examples** or of the **tasks**
  - This supposes that there are **interferences** between examples or tasks

Note: in **I.I.D.** and the **ERM** principle, all examples have the **same weight** and there is **no place for interference or order**

First, are all examples created **equal**?

Outline

1. How to measure the difficulty of a training example
2. Catastrophic forgetting
3. Curriculum building
4. A geometry on the space of tasks
5. Conclusions

How to measure the difficulty

of examples?

Measuring the **difficulty** of examples

- Previously
  - A **statistical** view
    - The probability of predicting the ground truth label **for an example omitted** from the training set
  - A **learning** view
    - The **earliest training iteration** after which the model (e.g. NN) predicts the ground truth class for that example **in all subsequent iterations**

Baldock, R., Maennel, H., & Neyshabur, B. (2021). **Deep learning through the lens of example difficulty**. *Advances in Neural Information Processing Systems*, *34*.

Measuring the **difficulty** of examples

- Proposition
  - The notion of "prediction depth"
  - And three distinct **difficulty types**:
    - Does this example **look mislabeled**?
    - Is classifying this example only easy if the label is given (**ambiguous** example)?
    - Is this **example ambiguous** both with and without its label?

Baldock, R., Maennel, H., & Neyshabur, B. (2021). **Deep learning through the lens of example difficulty**. *Advances in Neural Information Processing Systems*, *34*.

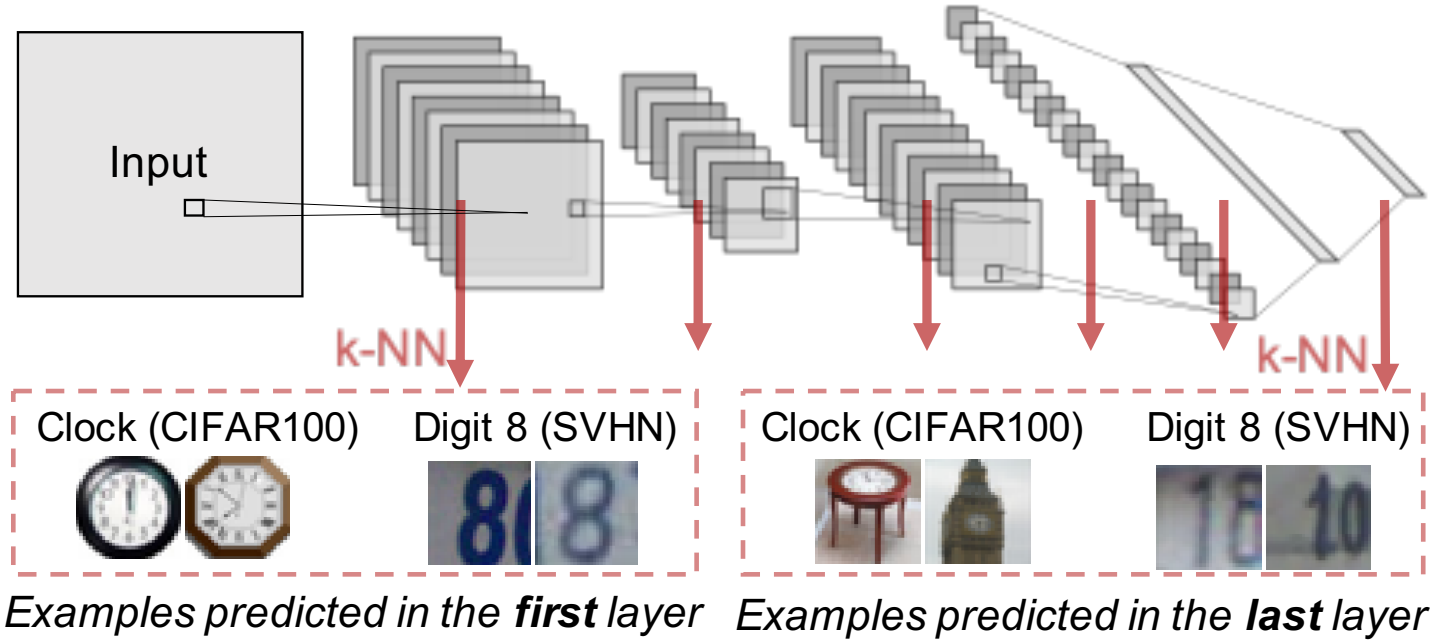
Prediction depth

...

Prediction depth

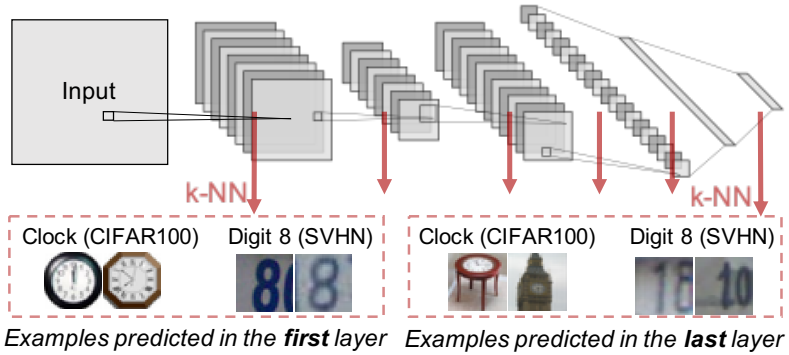
Prediction depth

- The number of hidden layers after which the network's final prediction **is** already **determined**

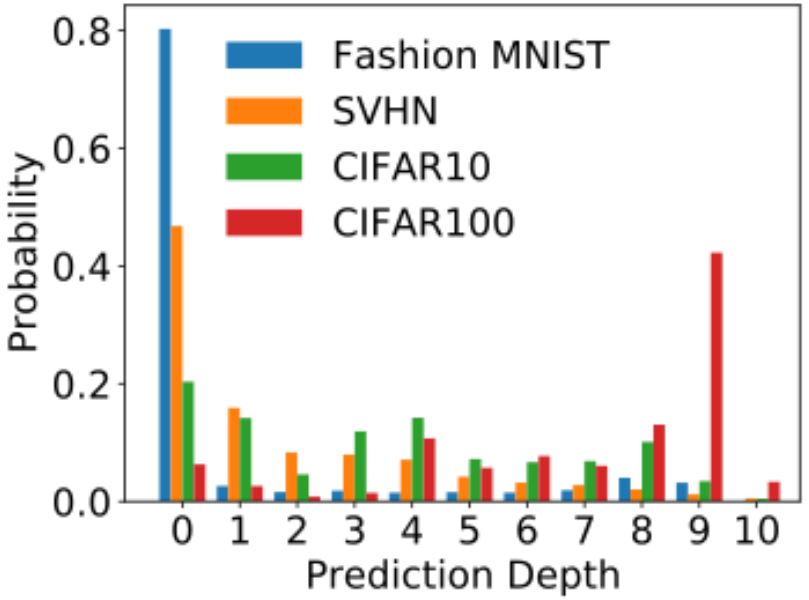


Prediction depth

- The number of hidden layers after which the network's final prediction **is** already **determined**



*Examples predicted in the **first** layer* *Examples predicted in the **last** layer*



How to measure the **prediction depth**?

- k-NN classifier **probes** (with $k = 30$)
  - Compare the **hidden embedding** of an **input**
    to **those of the training set**

(what is the class of the k nearest neighbors in the embedding considered)

How to measure the **prediction depth**?

- k-NN classifier **probes** (with $k = 30$)
  - Compare the **hidden embedding** of an **input**
    to **those of the training set**

  (what is the class of the k nearest neighbors in the embedding considered)

- A prediction is defined to be made at a depth $L = l$ if
  - The k-NN classification **after** layer $L = l - 1$ is **different** from the network's final classification,
  - but the classification of k-NN probes **after** every layer $L \geq l$ are all **equal** to the final classification of the network

What they **claim** to show

1. The **prediction depth is larger** for examples that visually appear to be **more difficult**
   - And this is consistent between NN's architectures and random seeds

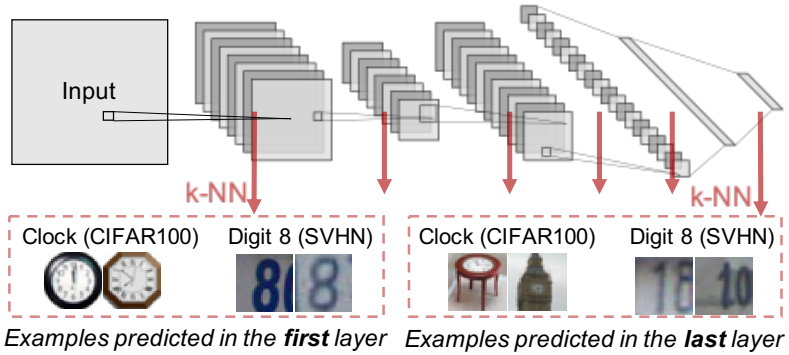
2. Predictions are on average **more accurate** for validation examples with **small prediction depths**

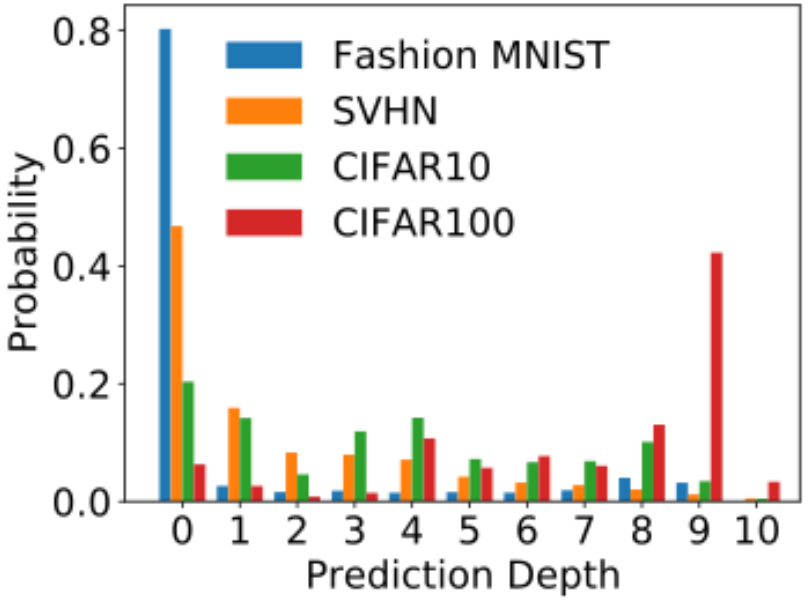
3. Final predictions for examples that **converge earlier** during training are typically determined in **earlier layers** (i.e. small prediction depth)

4. Both the adversarial input margin and output margin are **larger** for examples with **smaller prediction depths**

What they claim to show

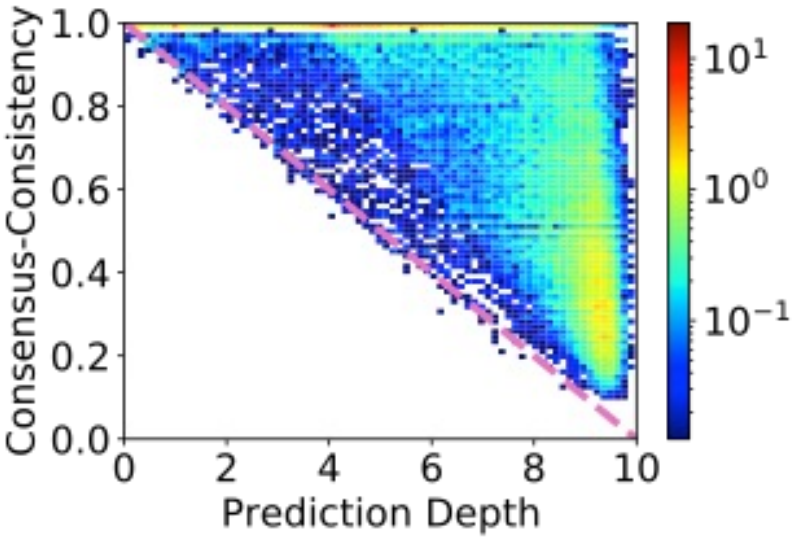
1. The **prediction depth is larger** for examples that visually appear to be **more difficult**





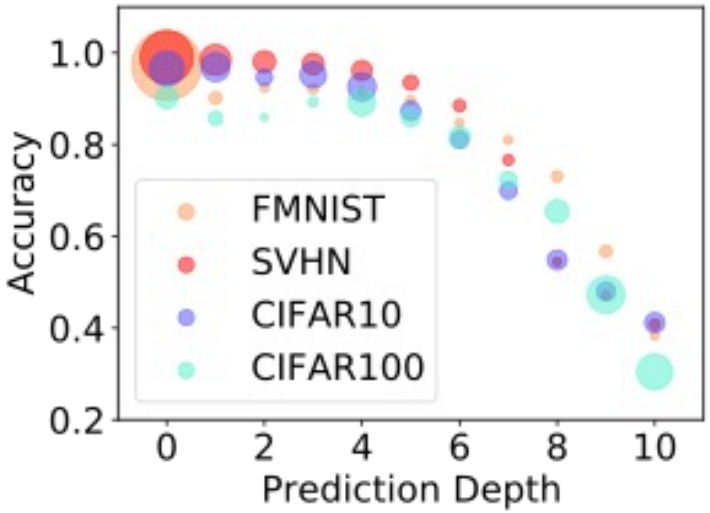
What they claim to show

2. Predictions are on average **more accurate** for validation examples with **small prediction depths**



250 ResNet18 were trained on CIFAR100 (90:10% random train:validation splits). Comparison of the average **prediction depth** of an example to the **consensus-consistency** of the corresponding prediction.

Consensus-consistency: the fraction of NNs that predict the ensemble's consensus class. *The **larger** the fraction, the **higher** the consensus*.

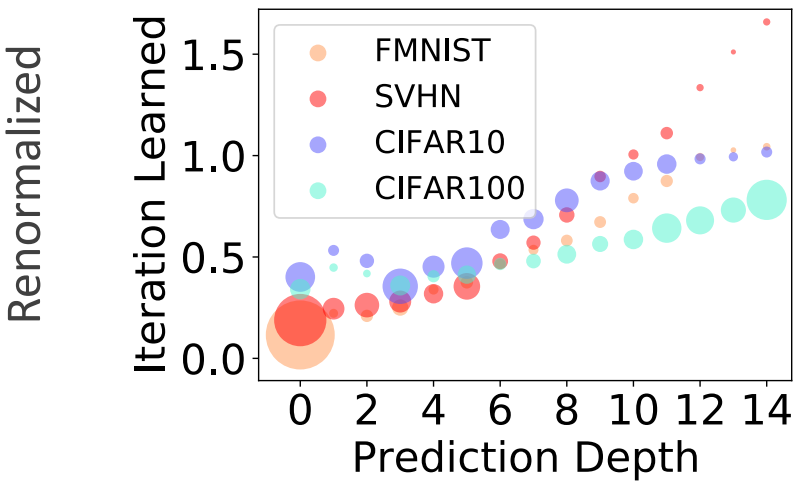


For each dataset, 250 ResNet18 were trained on CIFAR100 (90:10% random train:validation splits). For each point in the validation split, its **prediction depth** and whether the **prediction was correct** was recorded.

What they claim to show

3. Final predictions for data points that **converge earlier** during training are typically determined in **earlier layers**

- Measure the difficulty of learning **an example** by the **speed at which the model's prediction converges** for that input during training
- **Iteration learned**. A data point is said to be learned by a classifier at training iteration $t = \tau$ if the predicted class at iteration $t = \tau - 1$ is different from the final prediction of the converged NN and the predictions at all iterations $t \geq \tau$ are equal to the final prediction of the converged NN.



For each input in the validation split, the **prediction depth** and the **iteration learned** are recorded

Positive correlation between the **prediction depth** and the **iteration learned** appears for all datasets

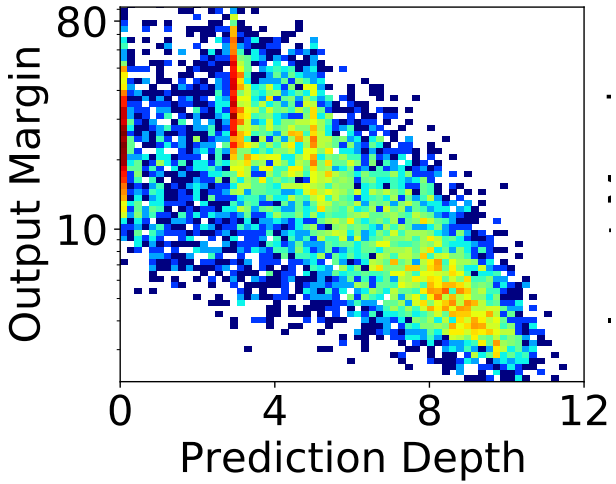
4. Both the adversarial input margin and output margin are **larger** for examples with **smaller prediction depths**

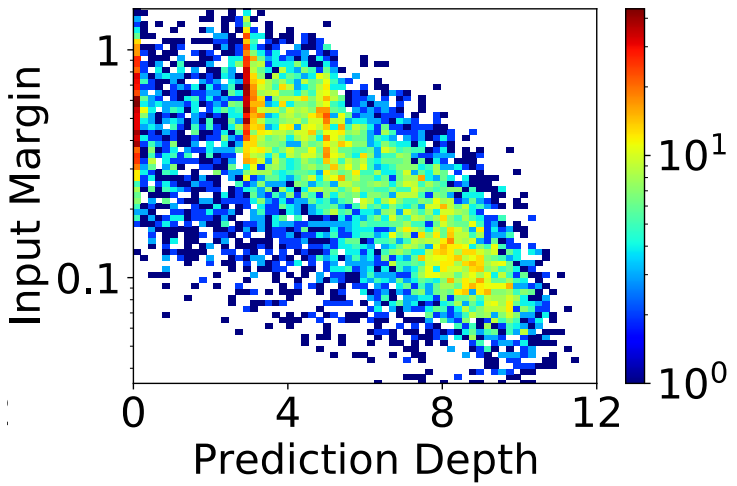
- **Output margin**: difference between the largest and second-largest **output** of the NN (logits).

  *The **smaller** the output margin, the more **uncertain** is the learner.*

- **Input margin**: the smallest norm required for an adversarial perturbation in the **input** to change the NN's class prediction.

  *The **smaller** the input margin, the **easier** it is **to change the prediction** of the learner.*





Shows that data points with **smaller prediction depths** have both **larger** input and output margins on average, and that **variances** of the input and output margins **decrease** as the prediction depth increases

- **Different forms** of example difficulty

  - **Validation**: points with **low** prediction depth are "**clear**"
    high prediction depth are "**ambiguous**"
  - **Training**: idem

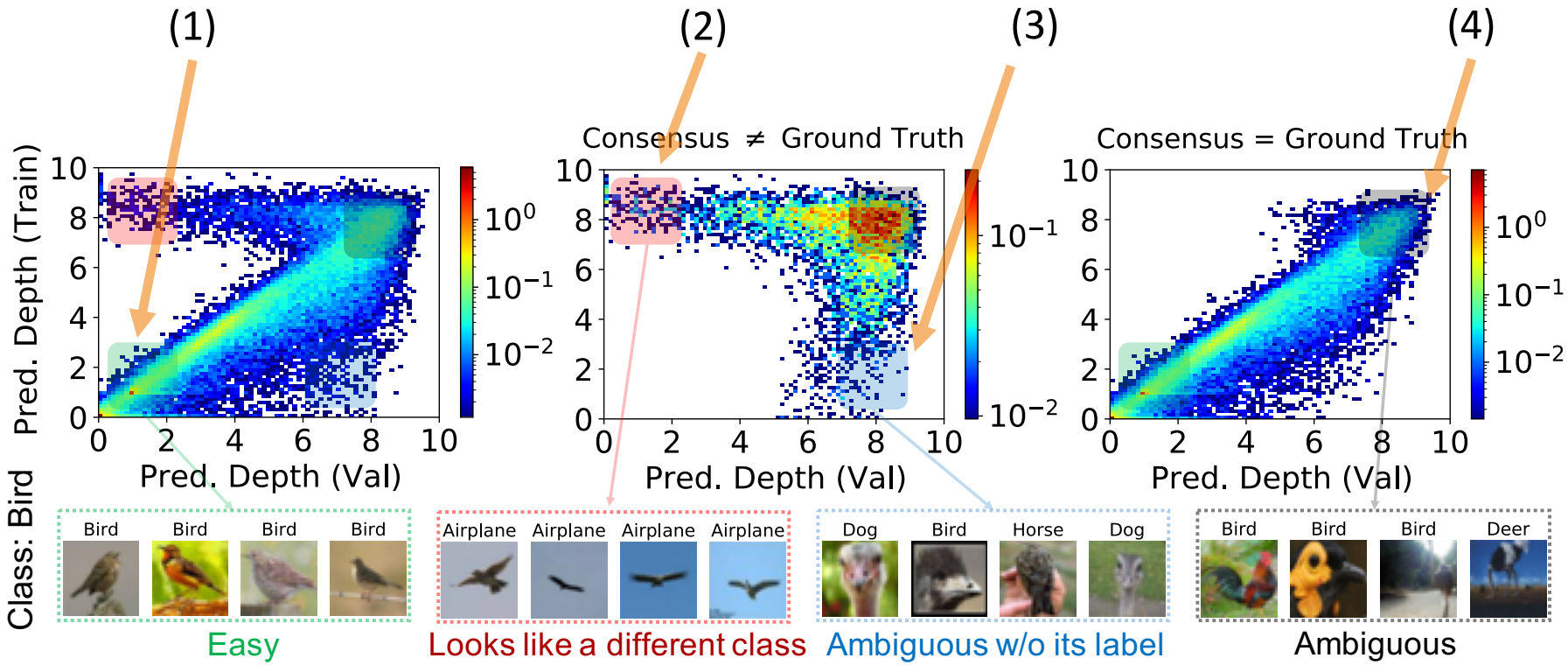
1. (Low PD_{val} and low PD_{train}) : Easy examples

2. (Low PD_{val} and high PD_{train}) : **Look** like a **different class**
   - E.g. **mislabeled** examples

3. (High PD_{val} and low PD_{train}) : **Ambiguous** unless the label is given
   - E.g. resemble both their **own class** and **another** class
     Likely to be **misclassified**

4. (High PD_{val} and high PD_{train}) : **Ambiguous**
   - Examples that may be **corrupted** or of a **rare** sub-class.

What they claim to show



These examples are difficult to connect to their predicted class in the **validation** split but easy to connect to their ground truth class during **training**. These points may, for example, visually resemble both their own class and another class. They are likely to be misclassified.

Conclusion

Introduces a notion of example difficulty called the prediction depth

- which uses the **processing** of data **inside the network**

  to score the difficulty of **an example**

Conclusion

- **Easy** examples are learned and recognized **early** in the network

What they claim to show

1. **Early** layers generalize while **later** layers memorize
2. Networks converge **from** input layers **towards** output layers
3. **Easy** examples are learned **first**
4. Networks present **simpler** functions **earlier** in the training

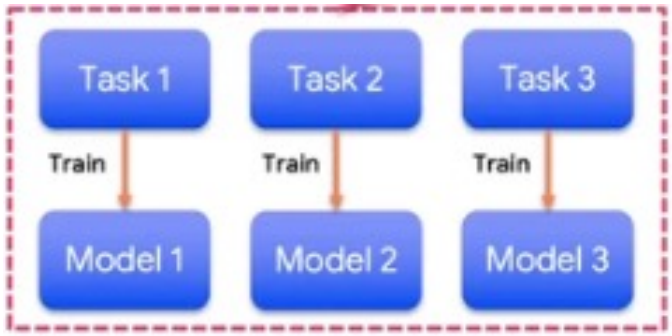
Outline

1. How to measure the difficulty of a training example
2. Catastrophic forgetting
3. Curriculum building
4. A geometry on the space of tasks
5. Conclusions

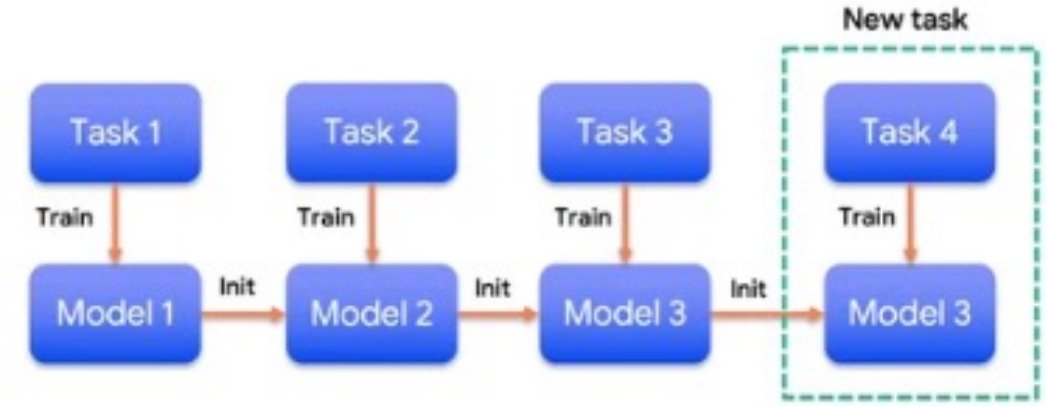
Continual learning of new tasks

Training new tasks

from scratch



Continual learning



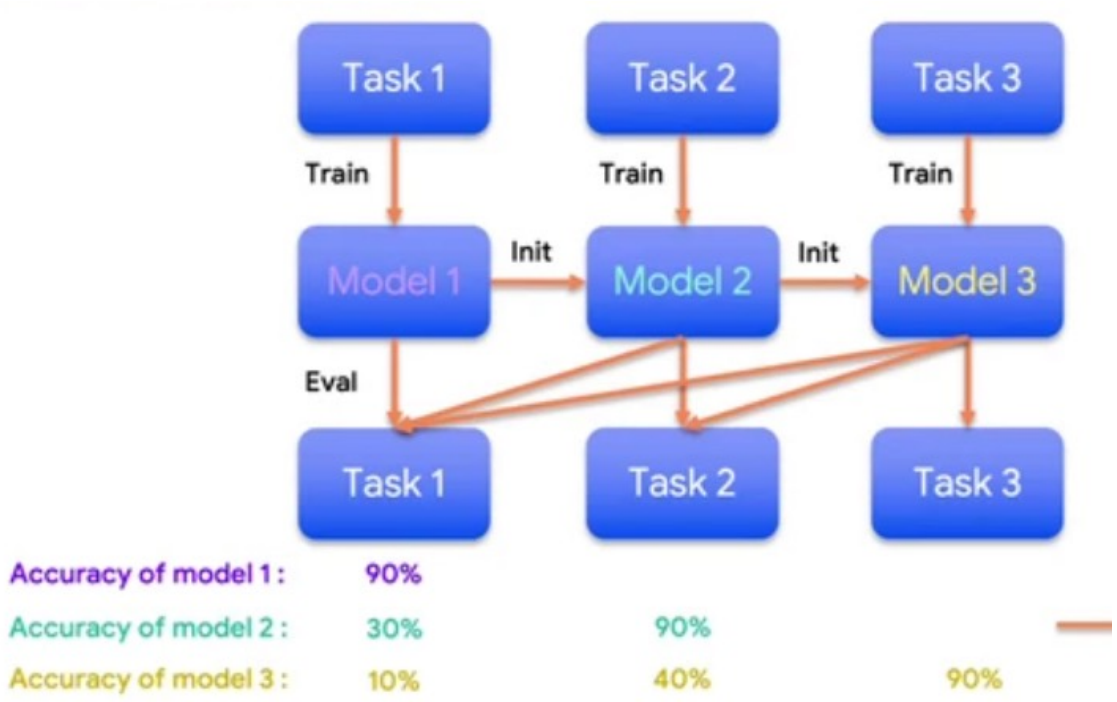
Chen, J., Nguyen, T., Gorur, D., & Chaudhry, A. (2023).

Is forgetting less a good inductive bias for forward transfer?

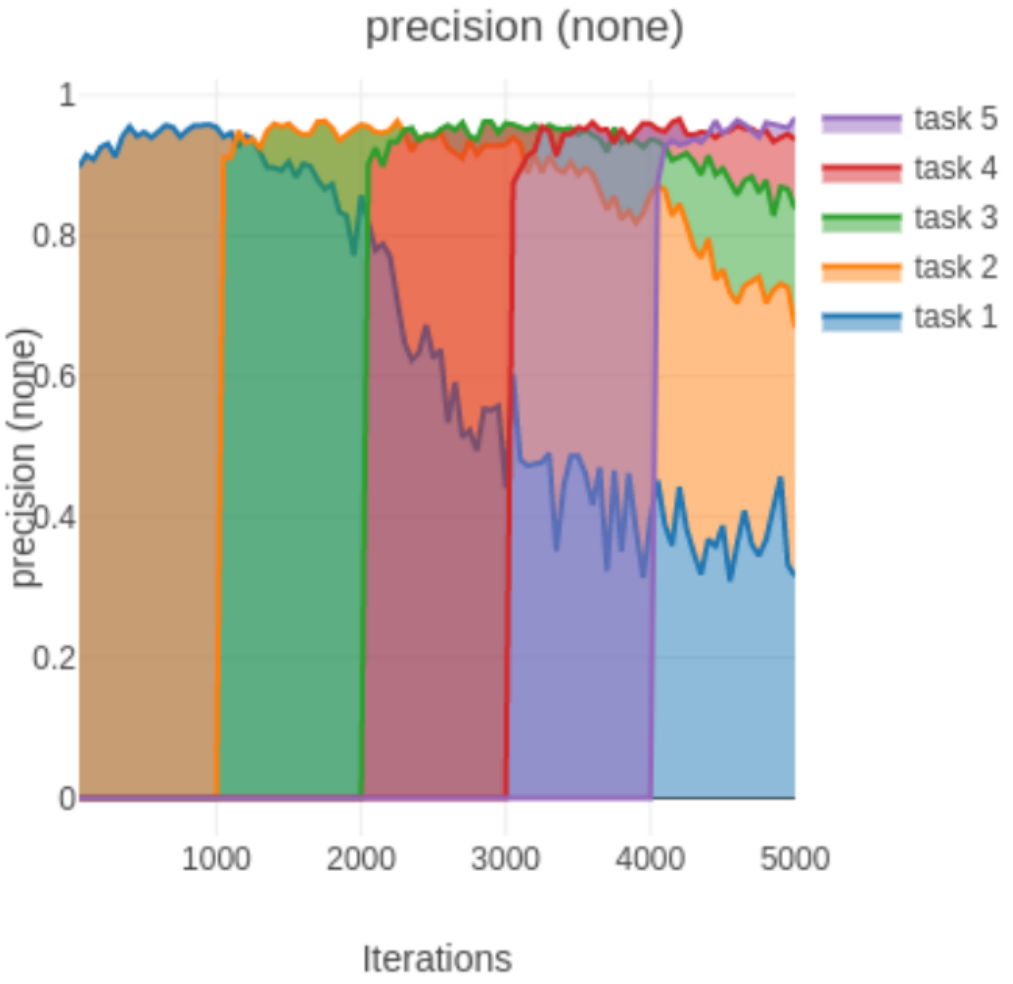
ICLR-2023.

Continual learning of new tasks

Continuously updating the model on new tasks results in severely **degraded** performance on **old tasks**



Catastrophic forgetting



McCaffary, D. (2021).
Towards continual task learning in artificial neural networks: current approaches and insights from neuroscience.
arXiv preprint arXiv:2112.14146.

Catastrophic forgetting

- ANNs have the tendency to completely and **abruptly forget** previous learned information upon learning new information
  - Therefore **ANNs** are unable to learn multiple tasks sequentially
  - Lifelong or continual learning would not be possible for **ANNs**
  - In **humans**, catastrophic forgetting does not happen
    - Learning to **drive a car** does not result in not knowing anymore how to **ride a bike**

Catastrophic forgetting: how to avoid it

- Training for ImageNet typically involves
  - to **break** the training dataset into M **distinct batches**,
  - where for ImageNet each batch typically has about **100,000 instances**
  - from 100 classes that are **not seen in later batches**,
  - and then the algorithm sequentially **loops** over each batch many times.

- **Not** efficient
- **Not** biologically plausible

Catastrophic forgetting

- In the **streaming scenario**
  - Learn with very small batches
  - Performs only a single pass through the dataset
- Large decrease in performance

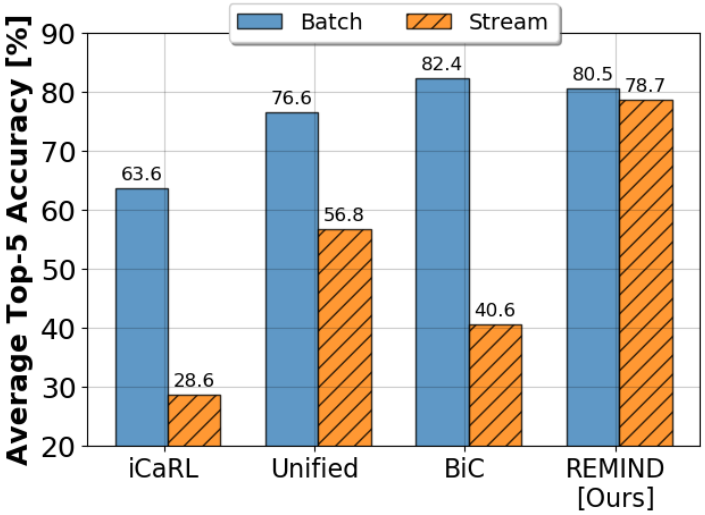


Fig. 1. Average top-5 accuracy results for streaming and incremental batch versions of state-of-the-art models on ImageNet.

Hayes, T. L., Kafle, K., Shrestha, R., Acharya, M., & Kanan, C. (2020, August). **Remind your neural network to prevent catastrophic forgetting**. In *European Conference on Computer Vision* (pp. 466-483). Springer, Cham.

Reasons for catastrophic forgetting

- **Interferences** in the hidden layers
  - When training on the **B task** modifies a lot the weights learnt for **task A**
    - No guarantee that the representation of deeper layers learned for **task A** will be sufficient to losslessly encode novel information, for **task B**, that is not representative of the original training data (i.e. task A)

- The major issue is **balancing**
  - the **stability** of existing representations
  - with the **plasticity** required to efficiently learn new ones

Catastrophic forgetting

- Questions
  - What happens to the **internal representations** of neural networks as they undergo catastrophic forgetting?
  - Does the degree to which a network forgets depend on the ***semantic similarity*** between the successive tasks?

- What do we expect?

Catastrophic forgetting and **hidden representations**

- What role **hidden layers** play in forgetting?
  - Do **all** parameters (and layers) **forget equally**?
  - Or are **specific components** of the network particularly **responsible** for the drop in accuracy in sequential training?

Catastrophic forgetting and **hidden representations**

- What role **hidden layers** play in forgetting?
  - Do **all** parameters (and layers) **forget equally**?
  - Or are **specific components** of the network particularly **responsible** for the drop in accuracy in sequential training?

How to approach these questions?

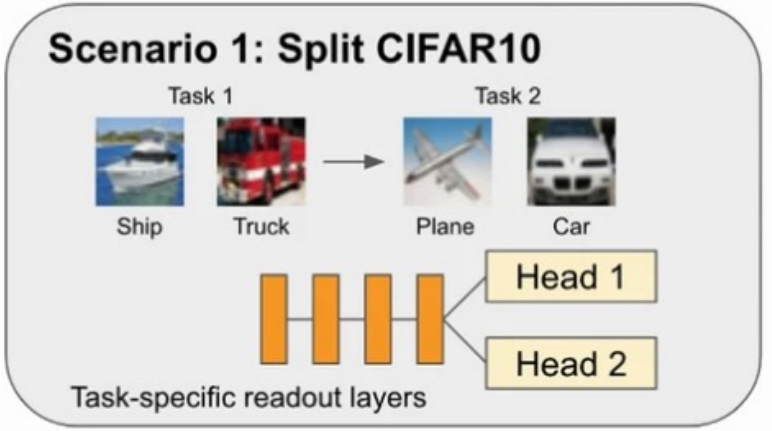
Catastrophic forgetting and **hidden representations**

Study the effect of **individual layers** on forgetting

- Two tasks: task 1 and task 2
  1. First learn task 1
  2. Then **freeze** some layers: starting from the lowest ones
  3. Learn task 2 only **training** the **remaining** layers
     and measure the **performance**

Catastrophic forgetting and **hidden representations**

- What role **hidden layers** play in forgetting?

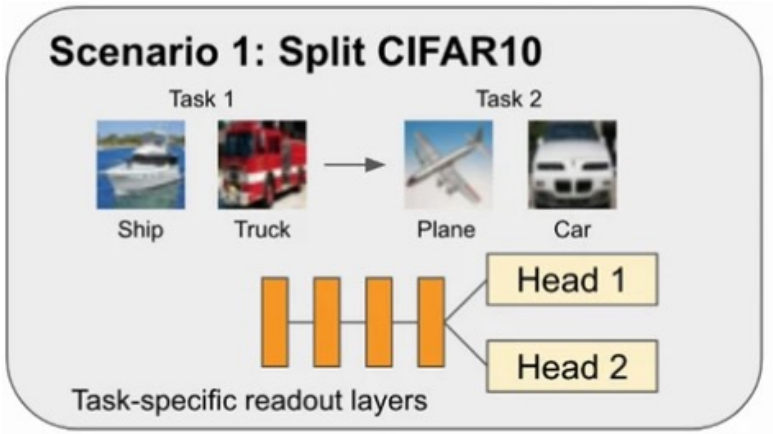


On CIFAR-10:
task 1 (5 classes) then **task 2** (5 ≠ classes)

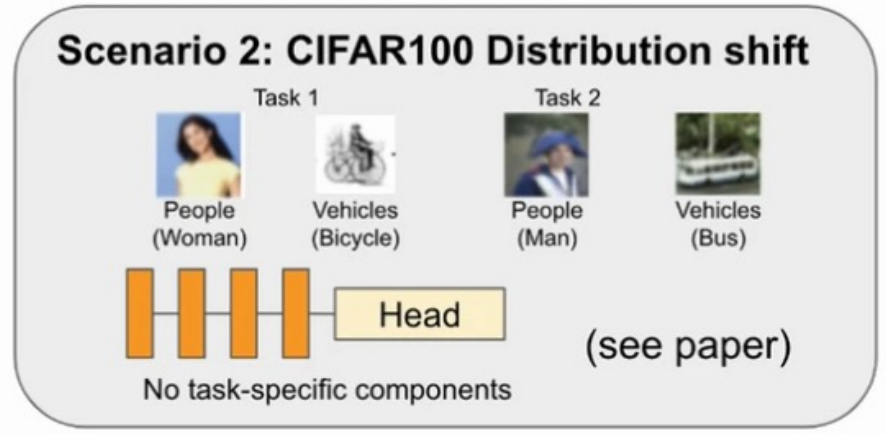
RAMASESH, Vinay V., DYER, Ethan, et RAGHU, Maithra (2021). **Anatomy of catastrophic forgetting: Hidden representations and task semantics**. *ICLR-2021.*

Catastrophic forgetting and **hidden representations**

- What role **hidden layers** play in forgetting?



On CIFAR-10:
task 1 (5 classes) then **task 2** (5 ≠ classes)

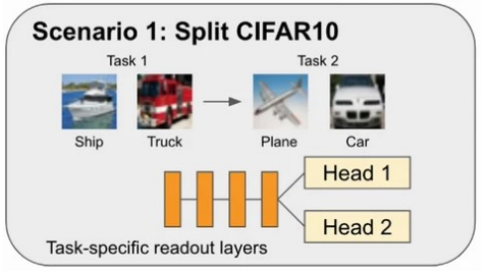


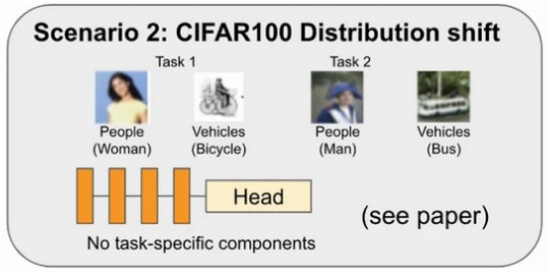
On CIFAR-100:
task 1 (examples of 5 subsets of 5 superclasses) then **task 2** (examples of 5 ≠ subsets of same 5 superclasses).
Each task is to distinguish between the CIFAR-100 superclasses, but input data for each task is **a different subset** of the constituent classes of the superclass.

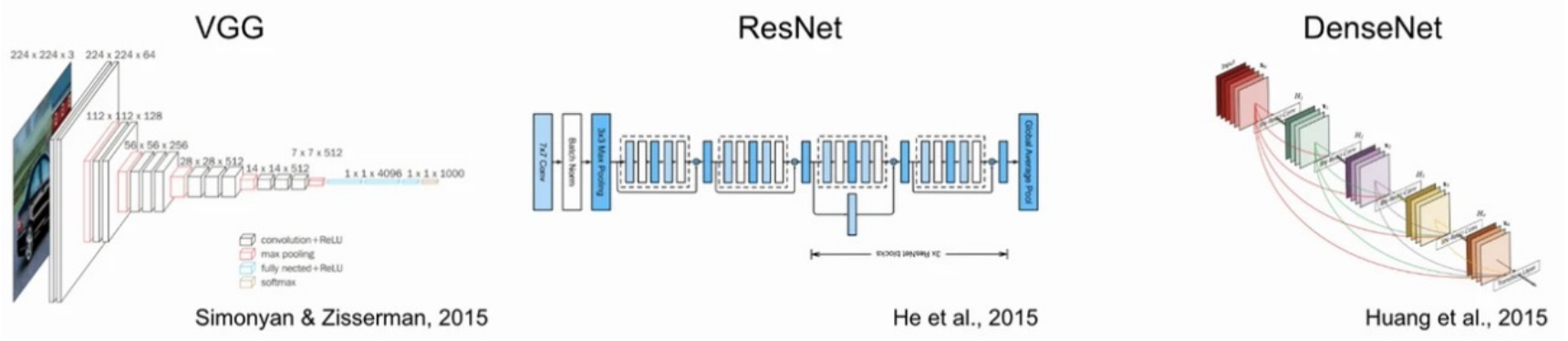
RAMASESH, Vinay V., DYER, Ethan, et RAGHU, Maithra (2021). **Anatomy of catastrophic forgetting: Hidden representations and task semantics**. *ICLR-2021.*

Catastrophic forgetting and **hidden representations**

- What role **hidden layers** play in forgetting?
  - Tested on 3 different **Deep Neural Networks**



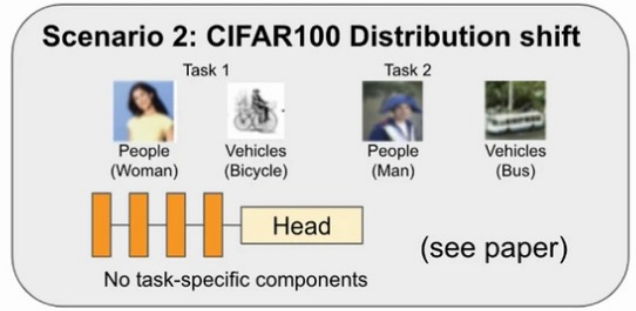




RAMASESH, Vinay V., DYER, Ethan, et RAGHU, Maithra (2021). **Anatomy of catastrophic forgetting: Hidden representations and task semantics**. *ICLR-2021.*

Catastrophic forgetting and **hidden representations**

- Manifestation of catastrophic forgetting?

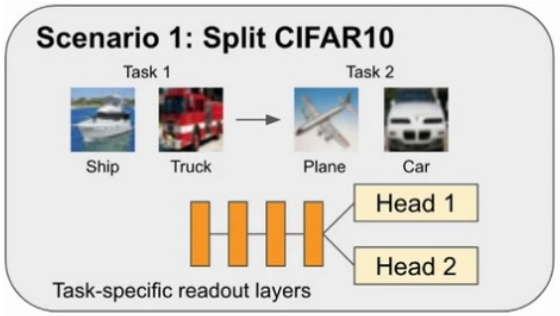


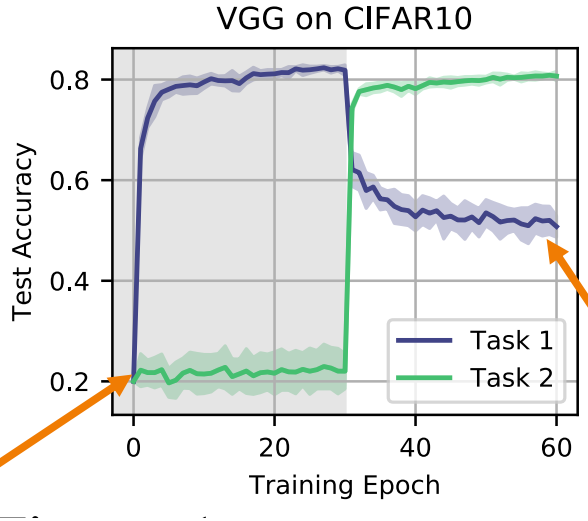
What do we expect?

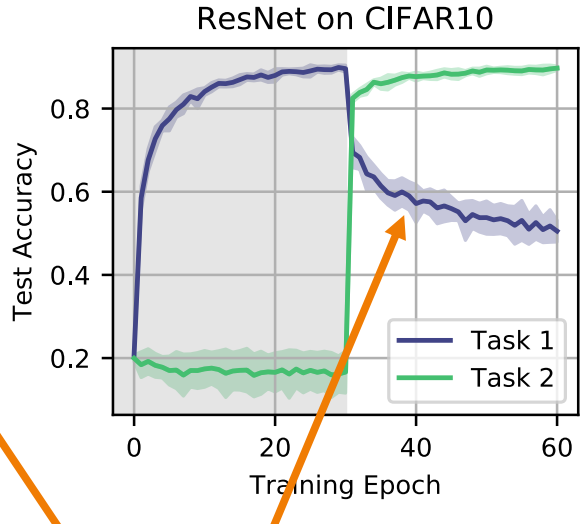
RAMASESH, Vinay V., DYER, Ethan, et RAGHU, Maithra (2021). **Anatomy of catastrophic forgetting: Hidden representations and task semantics**. *ICLR-2021.*

Catastrophic forgetting and **hidden representations**

- Manifestation of catastrophic forgetting?







Starts at ~20% recognition rate: normal

Significant **drop of performance** on task 1 when learning task 2

RAMASESH, Vinay V., DYER, Ethan, et RAGHU, Maithra (2021). **Anatomy of catastrophic forgetting: Hidden representations and task semantics**. *ICLR-2021.*

Catastrophic forgetting and **hidden representations**

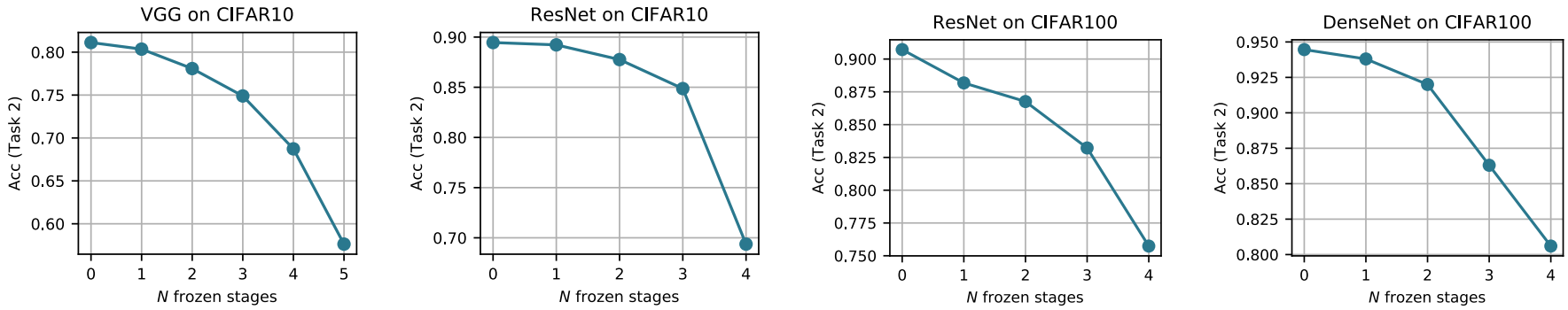
- What role **hidden layers** play in forgetting?

RAMASESH, Vinay V., DYER, Ethan, et RAGHU, Maithra (2021). **Anatomy of catastrophic forgetting: Hidden representations and task semantics**. *ICLR-2021*.

Catastrophic forgetting and **hidden representations**

- What role **hidden layers** play in forgetting?

Performance: accuracy



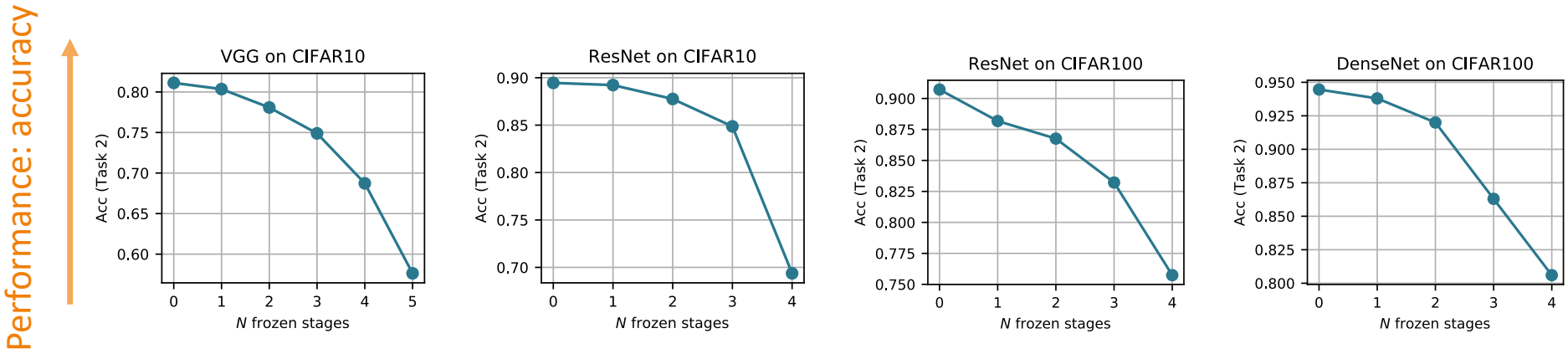
Stages = hidden layers starting from the earliest ones

Which conclusions?

RAMASESH, Vinay V., DYER, Ethan, et RAGHU, Maithra (2021). **Anatomy of catastrophic forgetting: Hidden representations and task semantics**. *ICLR-2021.*

Catastrophic forgetting and **hidden representations**

- What role **hidden layers** play in forgetting?



Stages = hidden layers starting from the earliest ones

- Freezing the **earliest hidden layers** after learning **task 1** has **little impact** on the performance of **task 2**
- **Higher layers** are disproportionately **responsible** for catastrophic forgetting

RAMASESH, Vinay V., DYER, Ethan, et RAGHU, Maithra (2021). **Anatomy of catastrophic forgetting: Hidden representations and task semantics**. *ICLR-2021.*

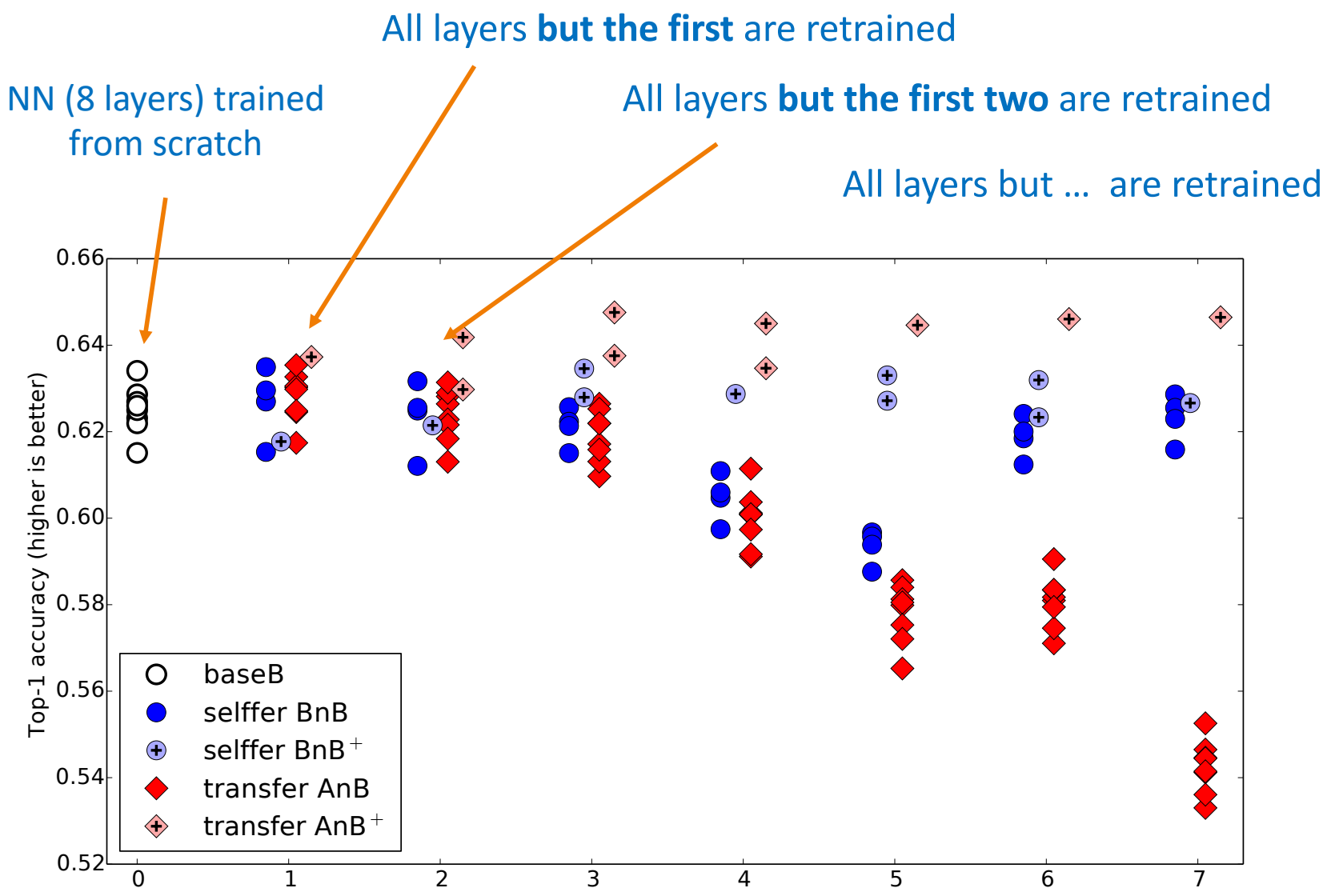
Catastrophic forgetting and **hidden representations**

- What role **hidden layers** play in forgetting?
  - Freezing the **earliest hidden layers** after learning **task 1** has **little impact** on the performance of **task 2**
  - **Higher layers** are disproportionately **responsible** for catastrophic forgetting

Why is it so?

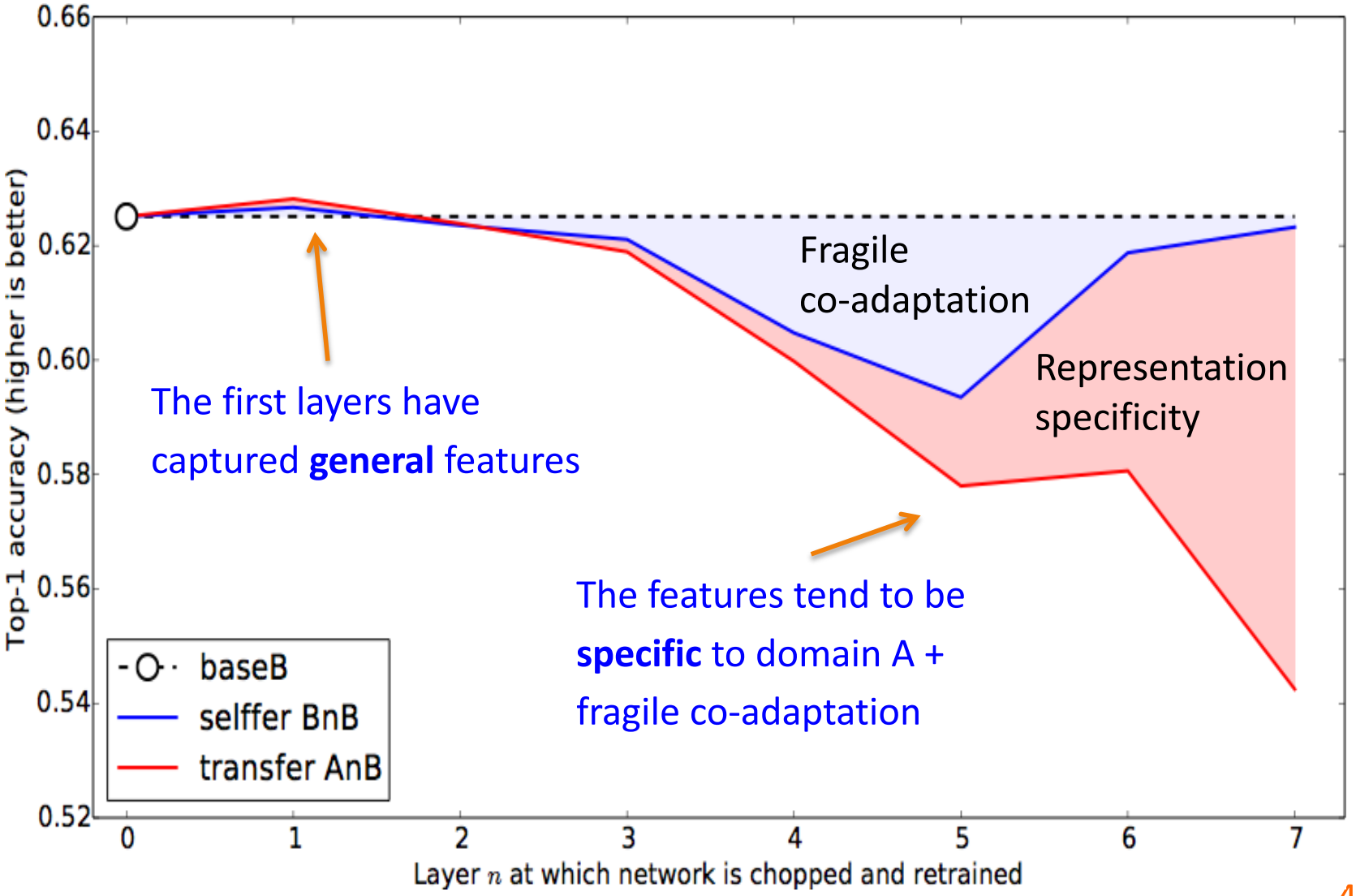
RAMASESH, Vinay V., DYER, Ethan, et RAGHU, Maithra (2021). **Anatomy of catastrophic forgetting: Hidden representations and task semantics**. *ICLR-2021*.

Results: what to think of them?



Yosinski J, Clune J, Bengio Y, and Lipson H. **How transferable are features in deep neural networks?** In *Advances in Neural Information Processing Systems* 27 (NIPS '14), NIPS Foundation, 2014.

Interpretation



Catastrophic forgetting and **hidden representations**

- What role **hidden layers** play in forgetting?
  - Measure *how similar* is each hidden layer **before** and **after** learning **task 2**
    - Use Centered Kernel Alignment (CKA)

Specifically, letting $X \in \mathbb{R}^{n \times p}$ and $Y \in \mathbb{R}^{n \times p}$ be (centered) layer activation matrices of (the same) n datapoints and p neurons, CKA computes

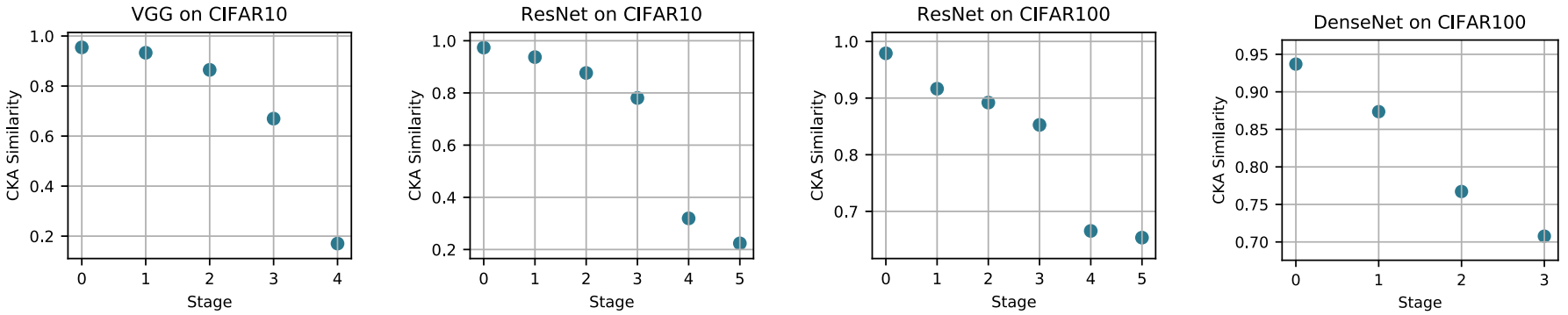
$$\mathrm{CKA}(\mathrm{X}, \mathrm{Y}) = \frac{\mathrm{HSIC}(\mathrm{XX}^{\mathrm{T}}, \mathrm{YY}^{\mathrm{T}})}{\sqrt{\mathrm{HSIC}(\mathrm{XX}^{\mathrm{T}}, \mathrm{XX}^{\mathrm{T}})}\sqrt{\mathrm{HSIC}(\mathrm{YY}^{\mathrm{T}}, \mathrm{YY}^{\mathrm{T}})}} \quad (1)$$

for HSIC Hilbert-Schmidt Independence Criterion (Gretton et al., 2005). We use linear-kernel CKA.

RAMASESH, Vinay V., DYER, Ethan, et RAGHU, Maithra (2021). **Anatomy of catastrophic forgetting: Hidden representations and task semantics**. *ICLR-2021.*

Catastrophic forgetting and **hidden representations**

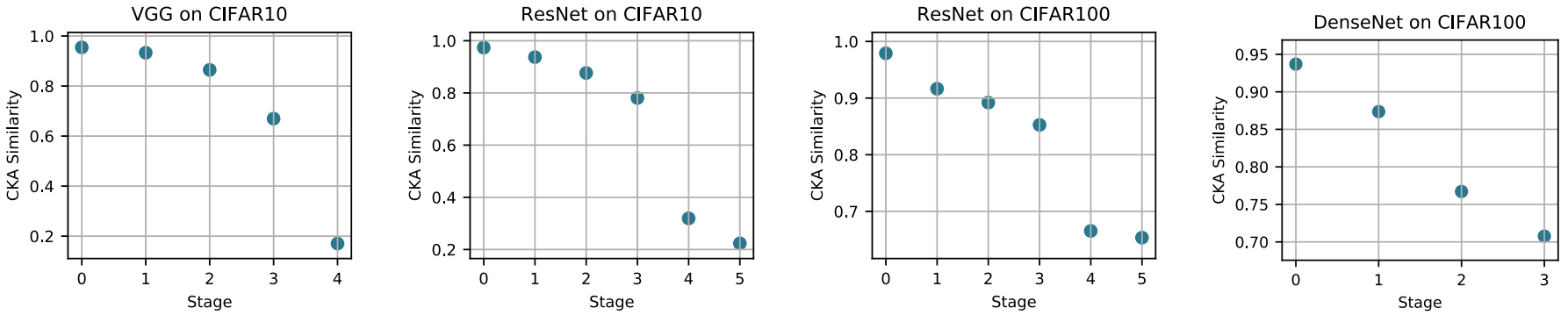
- What role **hidden layers** play in forgetting?
  - Measure how similar is each hidden layer **before** and **after** learning **task 2**
    - Use Centered Kernel Alignment (CKA)



RAMASESH, Vinay V., DYER, Ethan, et RAGHU, Maithra (2021). **Anatomy of catastrophic forgetting: Hidden representations and task semantics**. *ICLR-2021.*

Catastrophic forgetting and **hidden representations**

- What role hidden layers play in forgetting?
  - Measure how similar is each hidden layer **before** and **after** learning **task 2**
    - Use Centered Kernel Alignment (CKA)



Across architectures and tasks, we observe that the **lower layers have high representation similarity**, suggesting that lower layer Task 1 features are reused in Task 2.

For all **tasks** and all **NNs**

RAMASESH, Vinay V., DYER, Ethan, et RAGHU, Maithra (2021). **Anatomy of catastrophic forgetting: Hidden representations and task semantics**. *ICLR-2021.*

Catastrophic forgetting and **hidden representations**

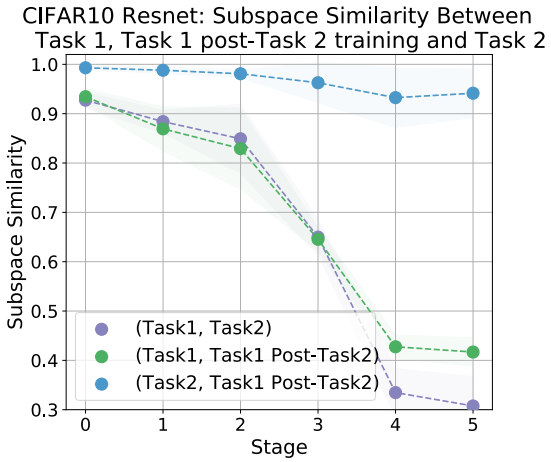
- What role **hidden layers** play in forgetting?
  - Measure how similar is each subspace (PCA of activations) of the hidden layers **before** and **after** learning **task 2**

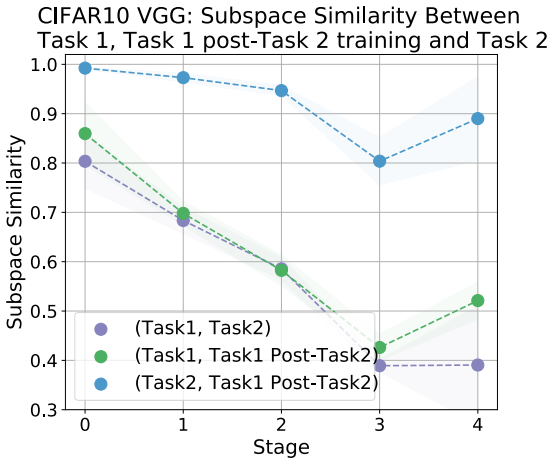
Letting $X \in \mathbb{R}^{n \times p}$ be the (centered) layer activation matrix of n examples by p neurons, we compute the PCA decomposition of X, i.e. the eigenvectors $(v_1, v_2, \ldots)$ and eigenvalues $(\lambda_1, \lambda_2, \ldots)$ of $X^\top X$. Letting V_k be the matrix formed from the top k principal directions, $v_1, \ldots, v_k$ as columns, and U_k the corresponding matrix for a different activation matric Y, we compute

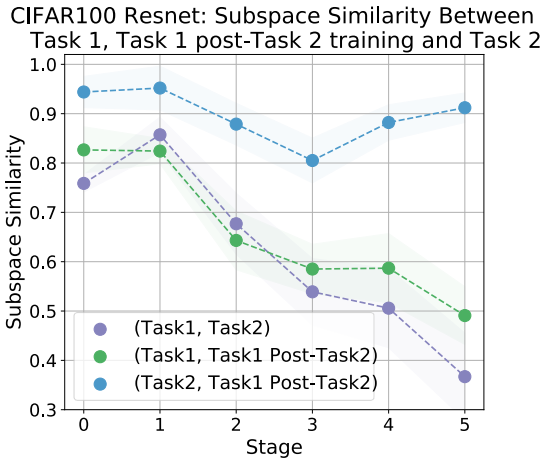
$$\text{SubspaceSim}_k(X, Y) = \frac{1}{k} ||V_k^\top U_k||_F^2$$

This measures the **overlap** in the subspaces spanned by $(v_1, ..., v_k)$ and $(u_1, ..., u_k)$. Concretely, if X and Y correspond to layer activation matrices for two different tasks, $\text{SubspaceSim}_k(X, Y)$ measures **how similarly the top** k **representations** for those tasks are stored in the network.

- What role **hidden layers** play in forgetting?
  - Measure how similar is each subspace (PCA of activations) of the hidden layers **before** and **after** learning **task 2**







- (Task 1, task 2) : low similarity for higher hidden layers
- (Task 1, task 2 and again on task 1) : much has been lost
- (Task 2, task 1 **after** training on task 2): higher hidden layers are more similar to task 2 than to task 1! The Task 1 subspace in higher layers is erased through Task 2 training

All comparisons show **high similarity in lower layers**, indicative of feature reuse

Catastrophic forgetting and **hidden representations**

- During sequential training,
  - effective **feature reuse happens** in the **lower layers**,
  - but in **the higher layers**, after Task 2 training, **Task 1 representations are mapped into the same subspace as Task 2**.

Specifically, **Task 2** training **causes subspace erasure** of **Task 1** in the **higher layers**.

Catastrophic forgetting and **hidden representations**

- During sequential training,
  - effective **feature reuse happens** in the **lower layers**,
  - but in **the higher layers**, after Task 2 training, **Task 1 representations are mapped into the same subspace as Task 2**.

Specifically, **Task 2** training **causes subspace erasure of Task 1** in the **higher layers**.

→ Do popularly used mitigation methods act to **stabilize higher layers**?

Mitigation strategies and hidden representations

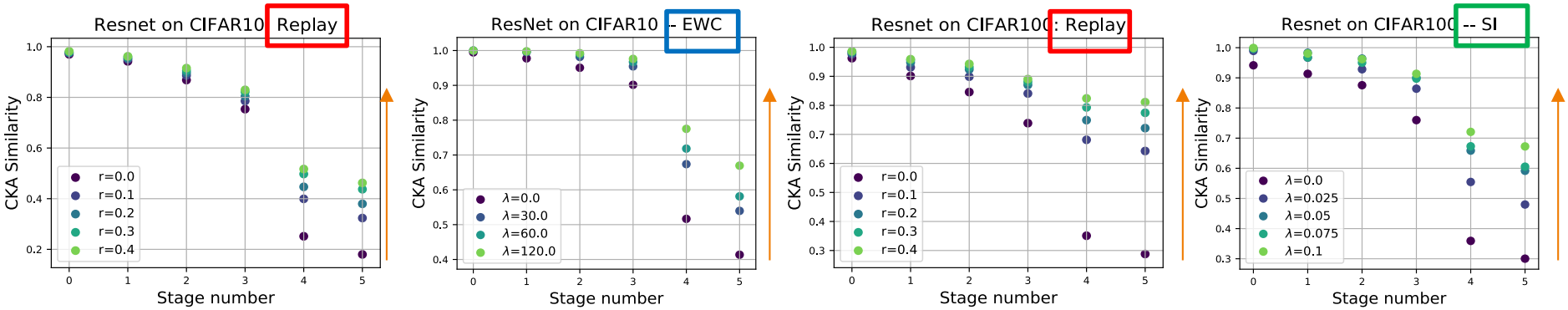
Types of mitigation **strategies**

- **Regularization**-based approaches
- **Replay**-based approaches

- What are their **impact**? Are they successful?
- **How do they act** on the hidden layers?
  - Do they **stabilize higher** hidden layers?
    - If yes, by keeping **non interfering** orthogonal **representations**?
    - Or by promoting **high features reuse**?

Mitigation strategies and hidden representations

- CKA analysis
  - Measures how similar a pair of hidden layer representations are

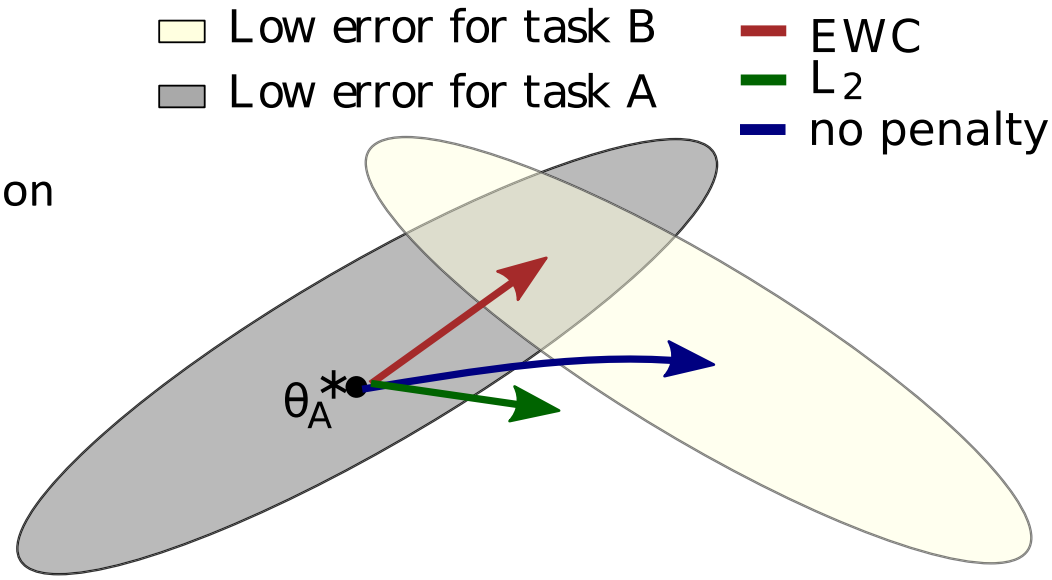


- Compute CKA between layer representations of Task 1 **before** and **after** Task 2 training
- With varying amounts and types of mitigation.

→ Mitigation methods **stabilize** the **higher layer** representations

EWC

Elastic Weight consolidation



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.

While learning task B, **EWC** protects the performance in task A by constraining the parameters to **stay in a region of low error**

for task A centered around θ_A^*

Mitigation strategies - (1) **Regularization** approaches

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.

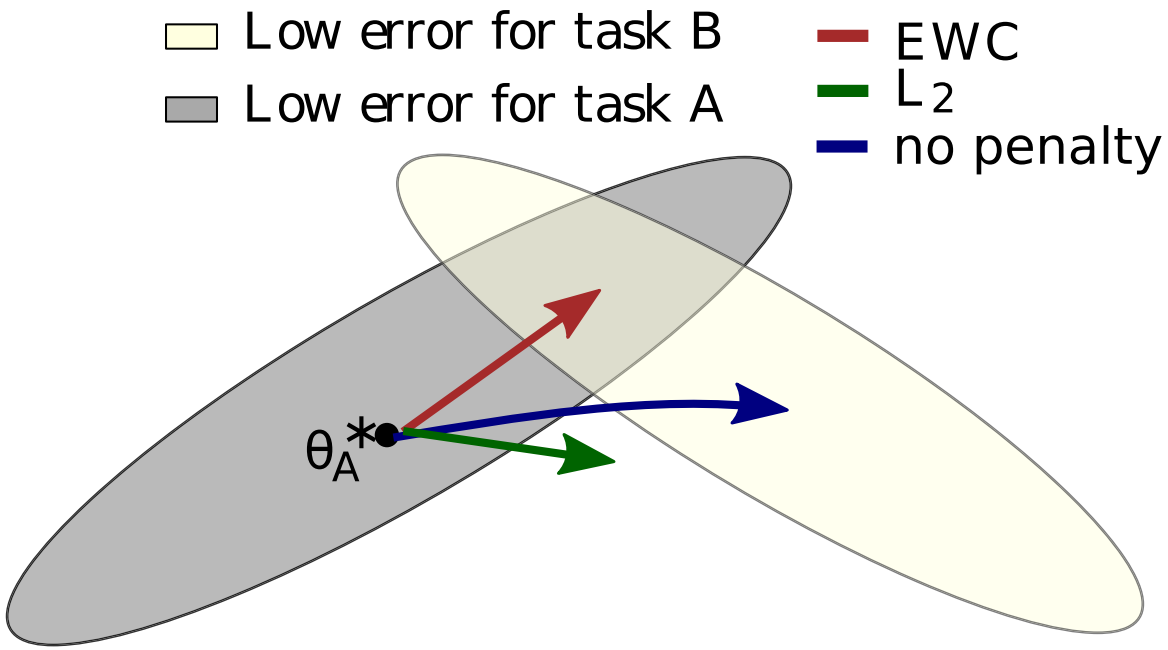


Fig. 1. EWC ensures task *A* is remembered while training on task *B*. Training trajectories are illustrated in a schematic parameter space, with parameter regions leading to good performance on task *A* (gray) and on task *B* (cream color). After learning the first task, the parameters are at θ_A^*. If we take gradient steps according to task *B* alone (blue arrow), we will minimize the loss of task *B* but destroy what we have learned for task *A*. On the other hand, if we constrain each weight with the same coefficient (green arrow), the restriction imposed is too severe and we can remember task *A* only at the expense of not learning task *B*. EWC, conversely, finds a solution for task *B* without incurring a significant loss on task *A* (red arrow) by explicitly computing how important weights are for task *A*.

Mitigation strategies - (1) **Regularization** approaches

- "Elastic Weight consolidation" (EWC)
  - **EWC** works by slowing learning of the network weights which are most relevant for solving **previously encountered tasks**

Hyperparameter indicating the relative importance of previously encountered tasks compared to new ones

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

Parameters of the network

Loss for **task B**

Fisher information matrix

Parameters important for solving **task A**

Mitigation strategies - (1) **Regularization** approaches

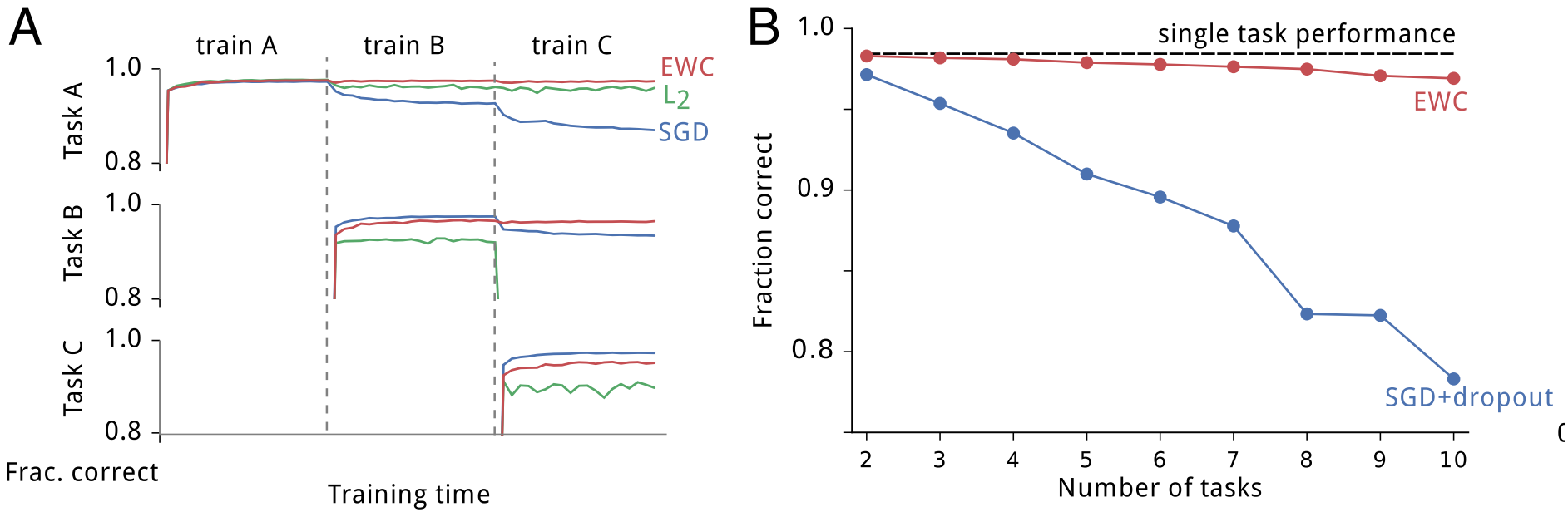
- Elastic Weight consolidation (**EWC**)
  - the Fisher information matrix is used to give an estimation of the **importance of weights** for solving tasks
    - The **importance weighting** is proportional to the diagonal of the Fisher information metric over the old parameters for the previous task

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

Parameters important for solving task A

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.

Mitigation strategies - (1) **Regularization** approaches



- (A) **Training curves** for three random permutations A, B, and C, using **EWC** (red), L_2 **regularization** (green), and **plain SGD** (blue).
  Note that only EWC is capable of maintaining a high performance on old tasks, while retaining the ability to learn new tasks.
- (B) **Average performance across all tasks**, using **EWC** (red) or **SGD** with dropout regularization (blue). The dashed line shows the performance on a single task only.

EWC is **efficient**

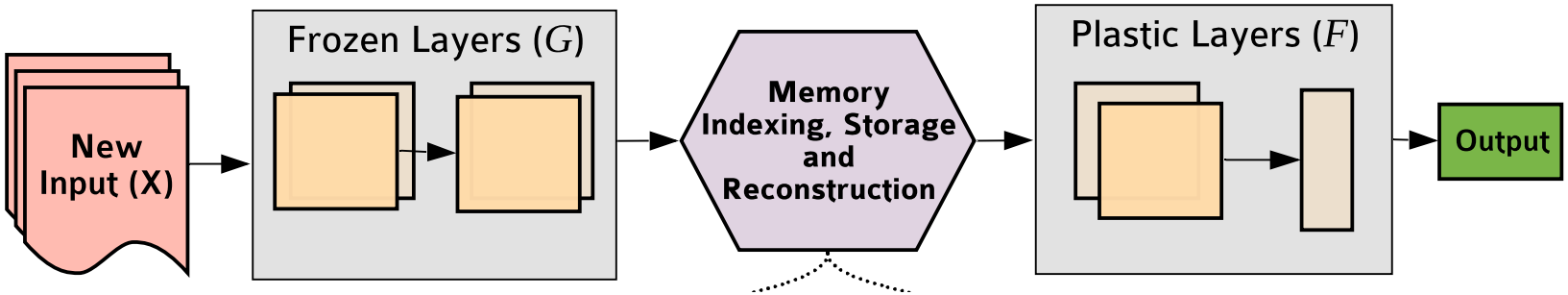
- **How** does it do it?
  - High features **reuse**?
  - Promoting **non interfering** higher subspaces?

Mitigation strategies – (2) **Replay-based** approaches

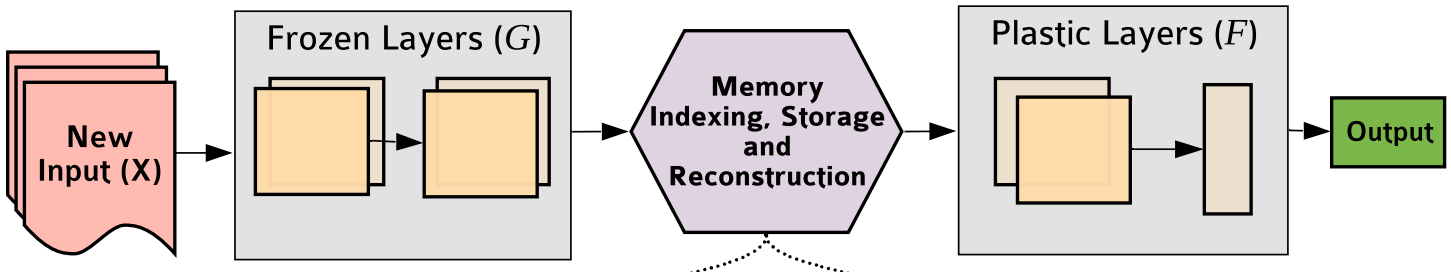
- Deep generative replay
  - A generative model is used to **generate representative data** from **previous** tasks
  - From which a sample is **selected** and **interspersed** with the dataset of the new task
  - Example: REMIND (Replay using Memory Indexing)
    - Replays a compressed representation of previously encountered training data
    - Using hidden layers (e.g. a feature map)

Hayes, T. L., Kafle, K., Shrestha, R., Acharya, M., & Kanan, C. (2020, August). **Remind your neural network to prevent catastrophic forgetting**. In *European Conference on Computer Vision* (pp. 466-483). Springer, Cham.

(2) REMIND


New Input (X)
Frozen Layers (G)
Memory Indexing, Storage and Reconstruction
Plastic Layers (F)
Output

(2)

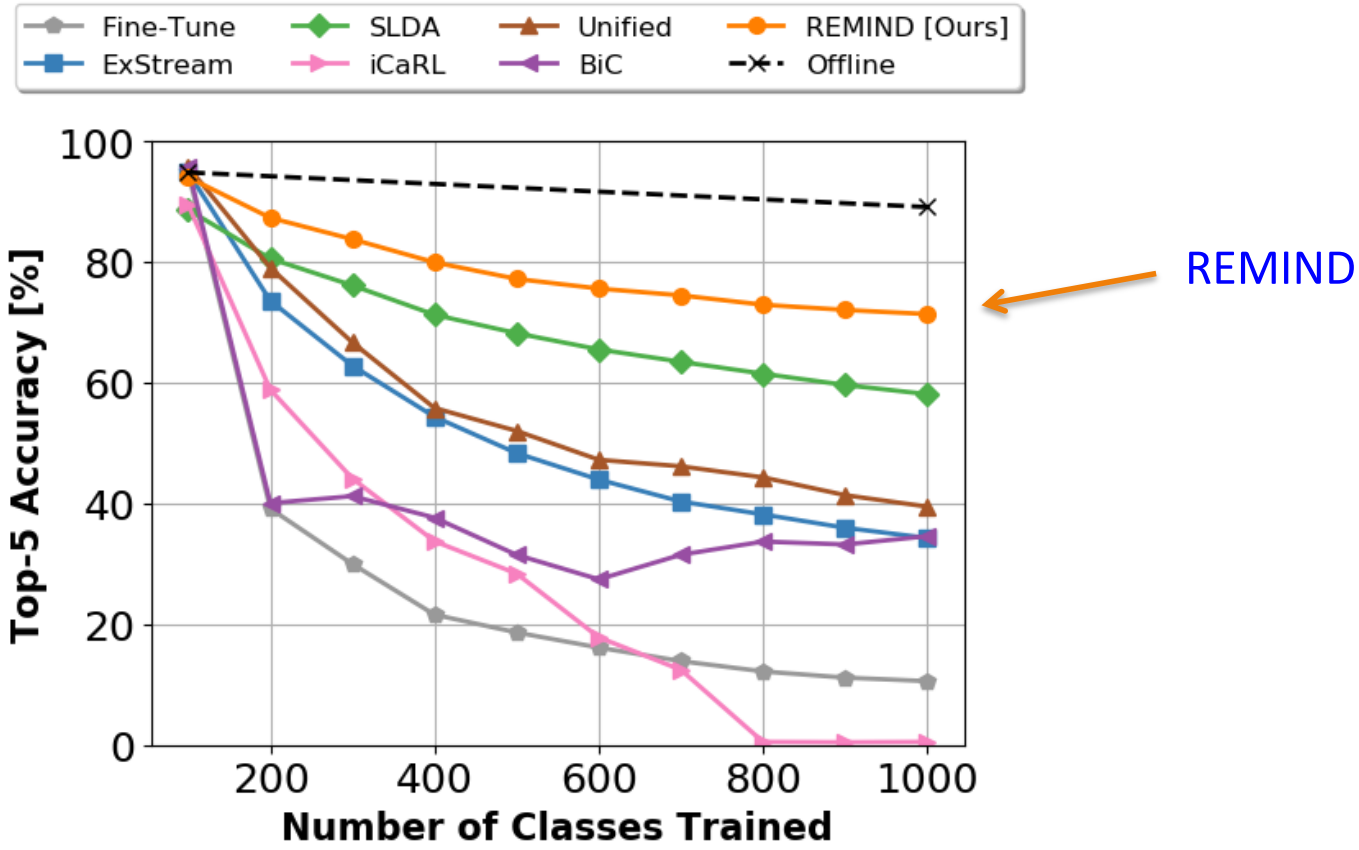


G F

- First, **train the complete network**: G + F layers on the training set
- Froze (G) and **store** sort of **prototype features** of the training examples
- Later, during training of **new tasks**, use the stored prototype features to **generate training instances** before (F) related to the previous tasks together with new training examples and train only (F)

(2) REMIND

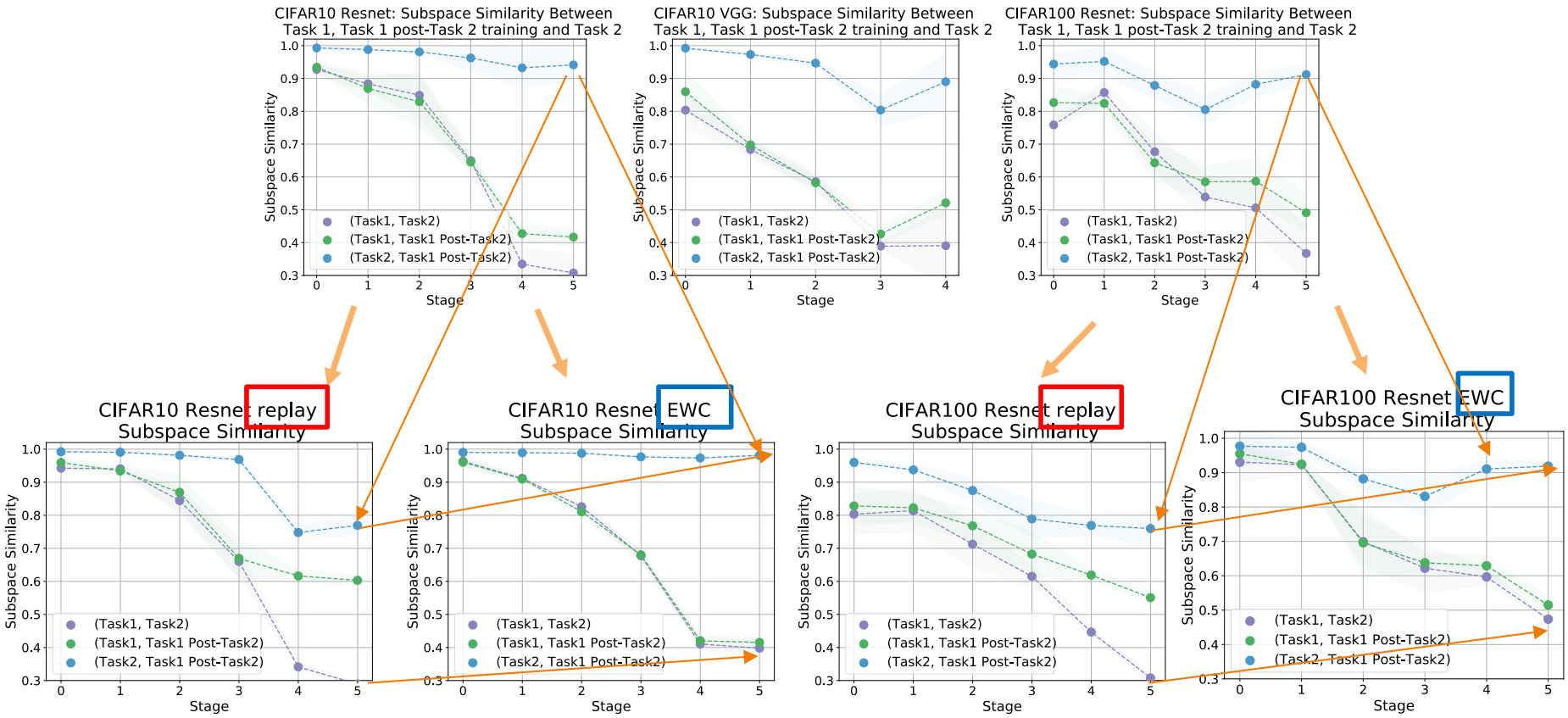
- Performances when learning additional classes of ImageNet



Now the **analysis**

Mitigation strategies and hidden representations

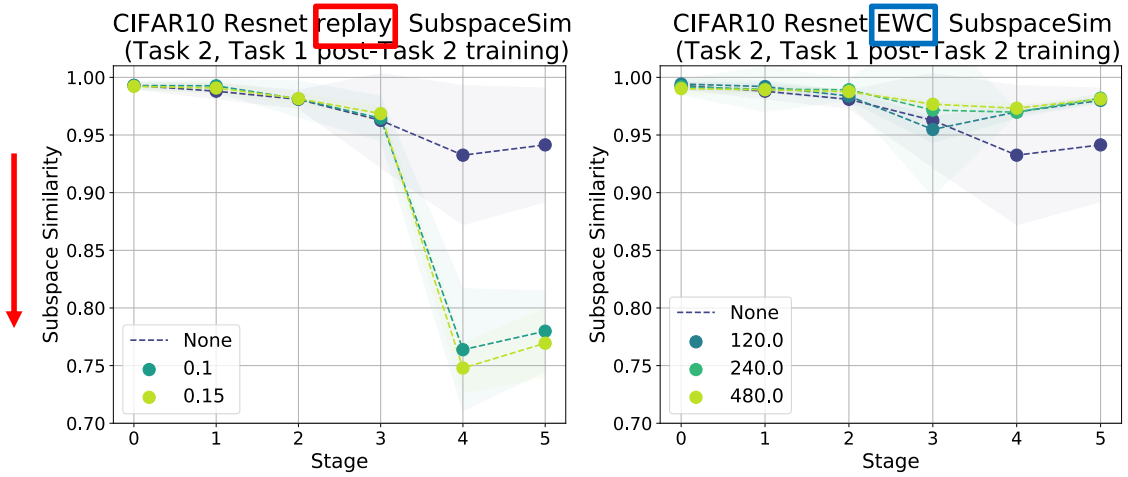
- But what about **subspace** similarity?



- (**Task 2, Task 1** post-Task 2 training) similarity is **lower** in **replay** compared to EWC **regularization-based methods**
- As is (**Task1, task 2**)

Mitigation strategies and hidden representations

- But what about **subspace** similarity?
  - With varying **degree** of mitigation



- Again (Task 2, Task 1 post-Task 2 training) is **much lower** in **replay** compared to no mitigation
- When **EWC** maintains **similar** subspaces for (Task2, Task 1 post Task 2 training)

Conclusion

- REPLAY adds **non interfering** additional features in higher layers
- EWC promotes the **reuse** of features in higher layers

Catastrophic forgetting

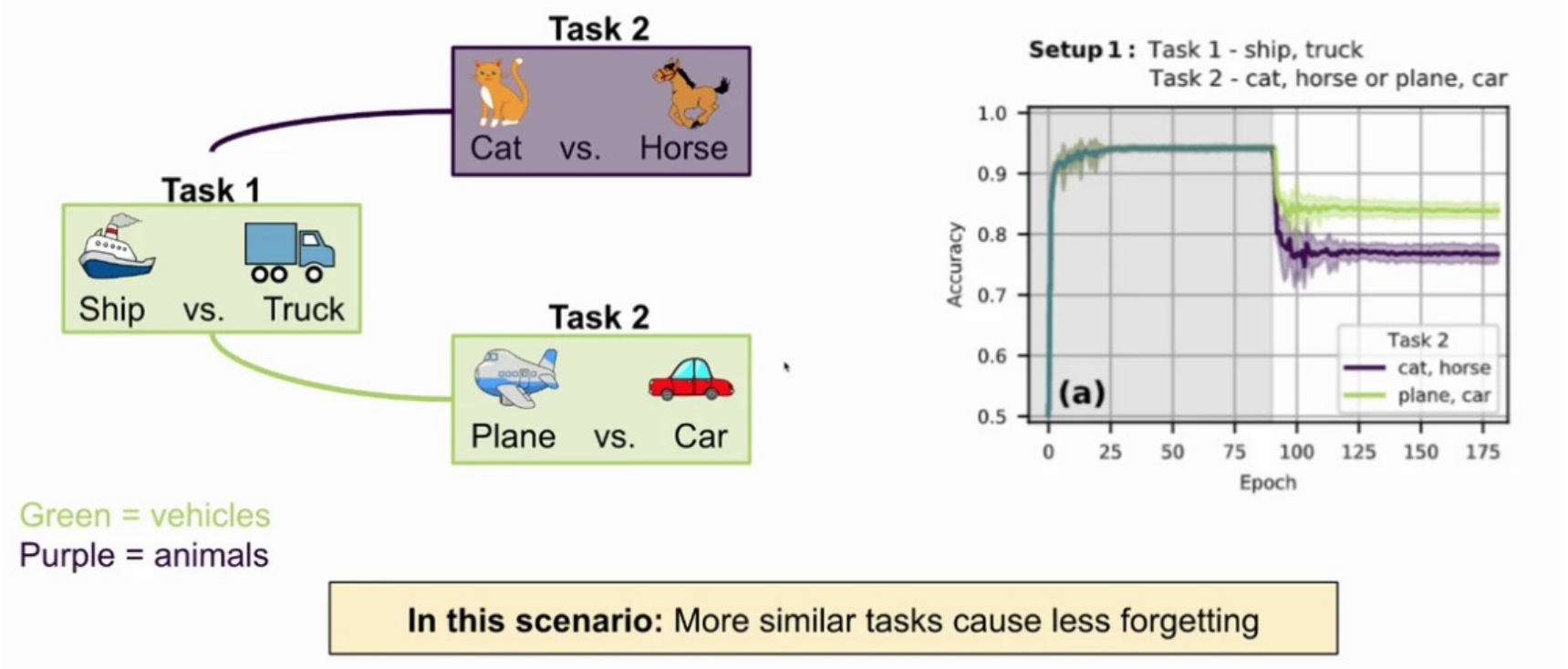
- Questions

  - What happens to the **internal representations** of neural networks as they undergo catastrophic forgetting?

  - Does the degree to which a network forgets depend on the ***semantic similarity*** between the successive tasks?

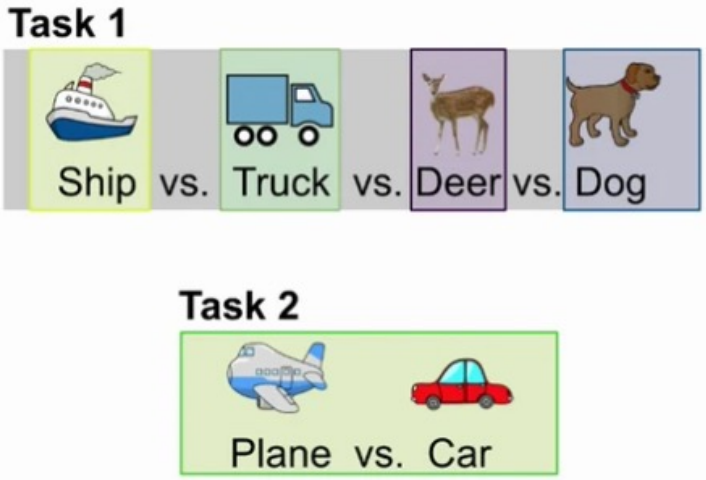
Catastrophic forgetting and **semantic similarity** between tasks

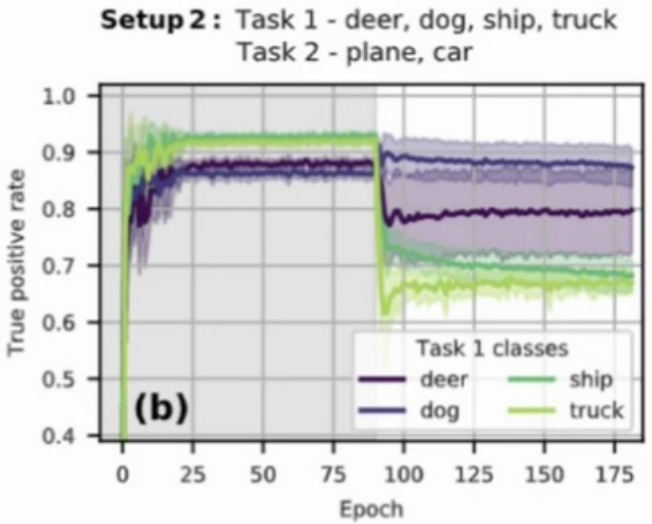
...



Catastrophic forgetting and **semantic similarity** between tasks

- But …

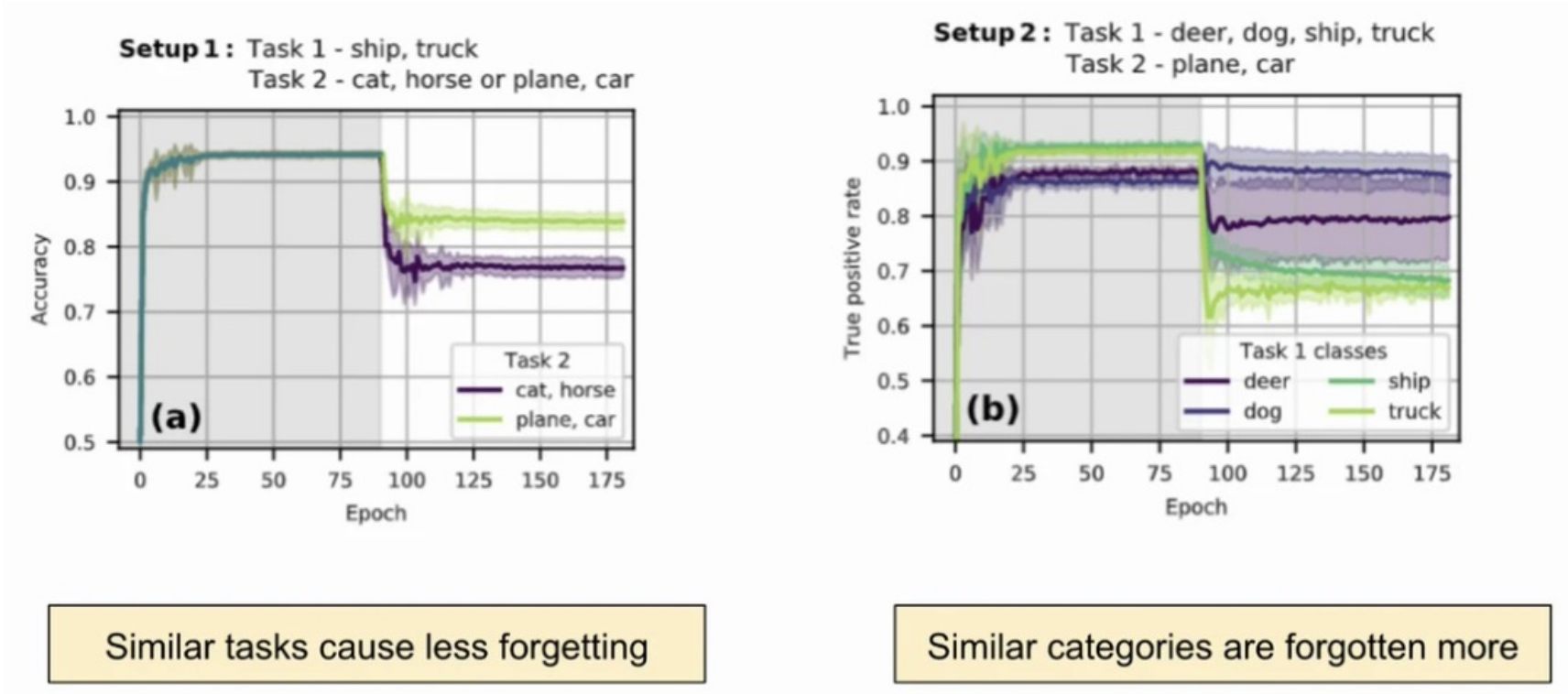




In this scenario: similar categories are forgotten more

Catastrophic forgetting and **semantic similarity** between tasks

- A **contradiction**?

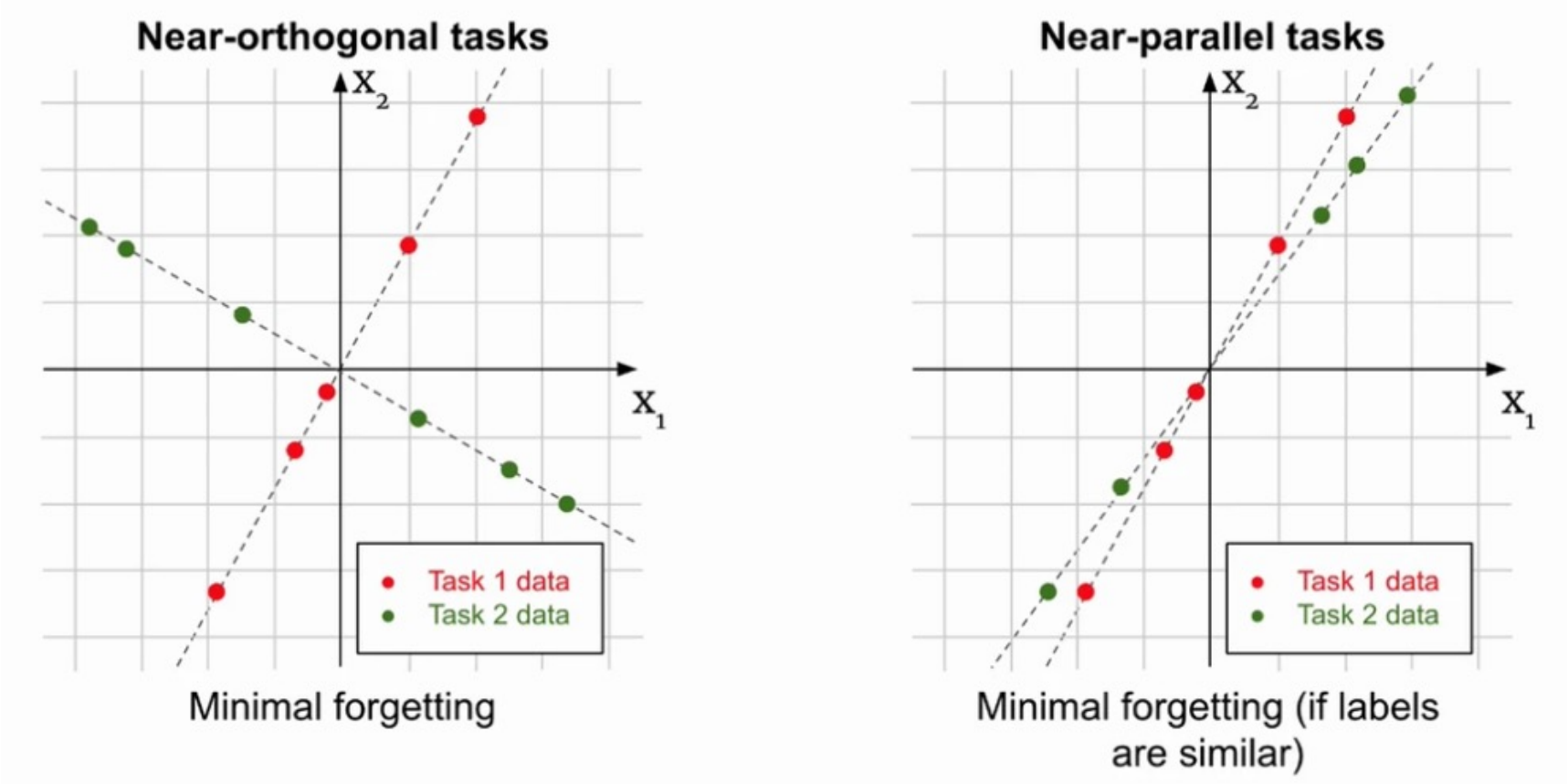


Similar tasks cause less forgetting

Similar categories are forgotten more

Catastrophic forgetting and **semantic similarity** between tasks

- Alignment of subspaces



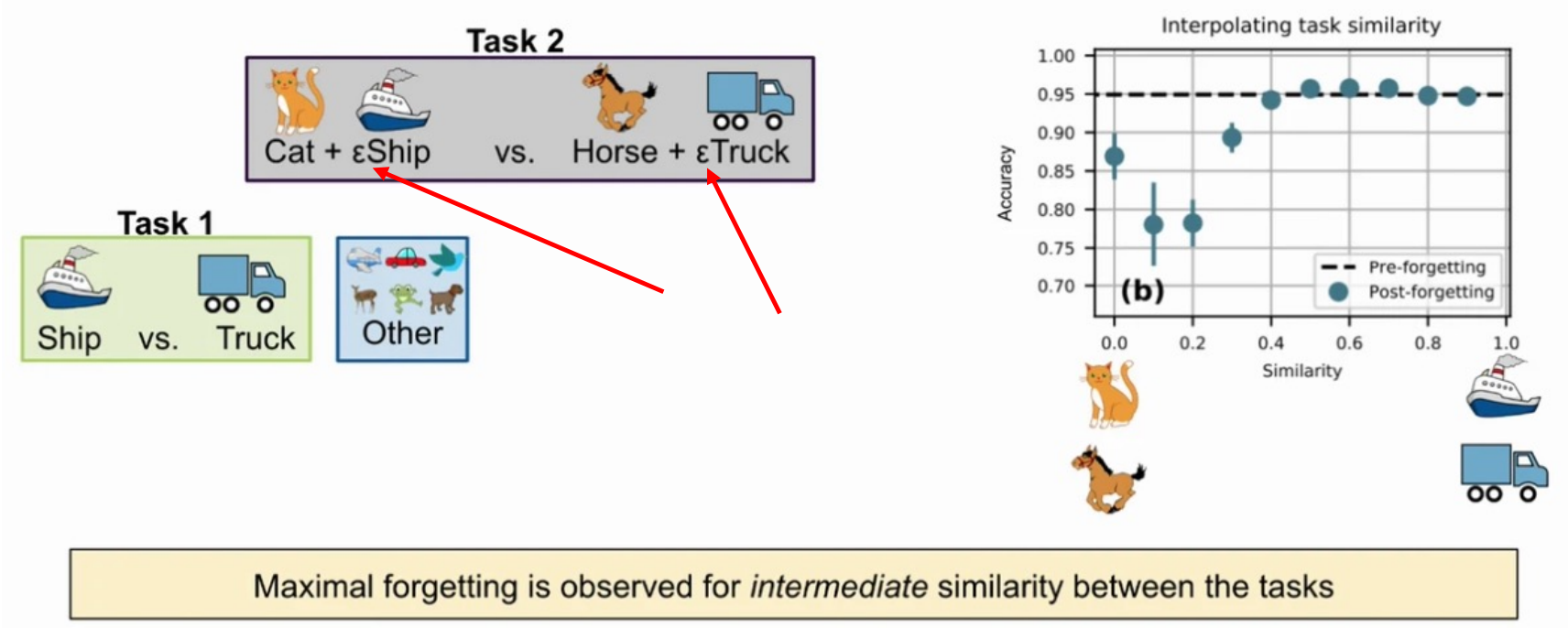
Near **orthogonal** model representations

Near **equal** model representations

Little forgetting

Catastrophic forgetting and **semantic similarity** between tasks

- …



General conclusions

- **Higher layers** are disproportionately **responsible** for catastrophic forgetting
- Different methods for **mitigating** forgetting exist
  - all **stabilize higher layer** representations,
    - But some methods encourage **greater feature reuse** in higher layers, (e.g. EWC)
    - Others **store task** representations **as orthogonal subspaces**, preventing interference (e.g. REPLAY)
- **Semantic similarity** between subsequent tasks consistently **controls** the degree of forgetting
  - forgetting is **most severe** for tasks with **intermediate similarity**

Can **forgetting** be **useful** in transfer learning?

- we want our models to capture generalizable concepts rather than purely statistical regularities

- Forgetting = any process that results in a decrease in training accuracy.

Zhou, H., Vani, A., Larochelle, H., & Courville, A. (2022). **Fortuitous forgetting in connectionist networks**. *ICLR-2022*.

Can **forgetting** be **useful** in transfer learning?

…

Zhou, H., Vani, A., Larochelle, H., & Courville, A. (2022). **Fortuitous forgetting in connectionist networks**. *ICLR-2022*.

Can **forgetting** be **useful** in transfer learning?

Forgetting = any process that results in a **decrease in training accuracy**.

Zhou, H., Vani, A., Larochelle, H., & Courville, A. (2022). **Fortuitous forgetting in connectionist networks**. *ICLR-2022*.

Can **forgetting** be **useful** in transfer learning?

- **Forgetting** is an inescapable component of **human memory**.
- It is **often thought** to be an **undesirable** characteristic of the human mind
- However, substantial evidence in psychology and neuroscience have shown that **forgetting** and **learning** have a **symbiotic** relationship

Could that be the **same** for **machine learning**?

- "Forgetting"

$$h_t \in \mathcal{H} \quad \text{trained on} \quad \mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{1 \leq i \leq n}$$

$$\text{Acc}(h) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}\{h(\mathbf{x}_i) = y_i\}$$

Forgetting operation — Noise — Random guessing

$$P\big[\text{Acc}\big(f(h_t, U)\big) < \text{Acc}(h_t) \;\&\; \text{Acc}(h_t) > C\big] = 1$$

Adding noise **decreases the accuracy** — The accuracy was better than random

$$I\big(f(N_t, U), \mathcal{D}\big) > 0$$

Adding noise equates to a **partial removal of information** (still "aligned")

Zhou, H., Vani, A., Larochelle, H., & Courville, A. (2022). **Fortuitous forgetting in connectionist networks**. *ICLR-2022*.

Can **forgetting** be **useful** in transfer learning?

- The forget-and-relearn hypothesis
  - Given an appropriate forgetting operation, iterative **re-training AFTER forgetting** will **amplify unforgotten features** that are **consistently useful** under different learning conditions induced by the forgetting step.
  - A forgetting operation that favors the **preservation** of **desirable features** can thus be used to steer the model towards those desirable characteristics.

Many existing **algorithms** which have successfully demonstrated improved generalization **have a forgetting step** that disproportionately **affects undesirable information** for the given task.

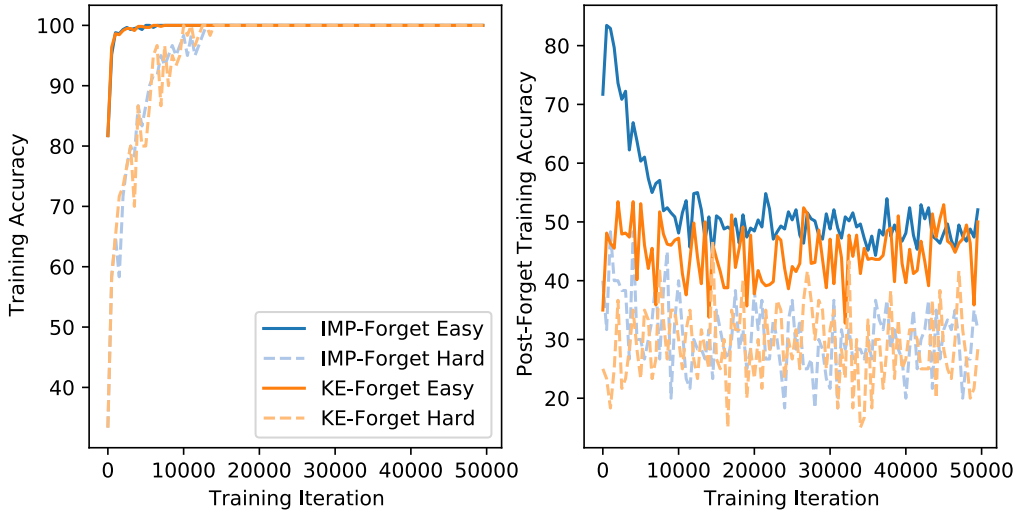
Can **forgetting** be **useful** in transfer learning?

- Easy vs. Hard examples
  - Use the **output margin** between the **largest** and **second-largest** logits (outputs) for each example

KE-style perturbation: the **reinitialization** of a random subset of the weights

IMP-style perturbation: the **zeroing** of **small magnitude weights** and rewinding of the remaining weights

Hard examples are **more adversely affected** than easy ones by weight perturbations



Normal training

Training with **weight perturbations** at each iteration

Targeted forgetting

- Later-Layer Forgetting (LLF)
  - **Reinitialization** of **later layers** at each learning iteration

Method	Flower	CUB	Aircraft	MIT	Dog
Smth (N1)	51.02 ±0.09	58.92 ±0.24	57.16 ±0.91	56.04 ±0.39	63.64 ±0.16
Smth long (N3)	59.51 ±0.17	66.03 ±0.13	62.55 ±0.25	59.53 ±0.60	65.39 ±0.55
Smth + KE (N3)	57.95 ±0.65	63.49 ±0.39	60.56 ±0.36	58.78 ±0.54	64.23 ±0.05
Smth + LLF (N3) **(Ours)**	**63.52** ±0.13	**70.76** ±0.24	**68.88** ±0.11	**63.28** ±0.69	**67.54** ±0.12

- Importance of having **variable conditions** for **refining first** layers
- Keeps and **amplifies** the **useful** features of the **first layers**

Lesson

- Forgetting is **useful**
  - If it promotes the **amplification** of **useful** features in the **first layers**

Outline

1. How to measure the difficulty of a training example
2. Catastrophic forgetting
3. Curriculum building
4. A geometry on the space of tasks
5. Conclusions

- Out of Distribution learning (OOD)

How the past can be used to solve the present task

An **intelligent** agent must have the capacity to **accumulate** and

maintain task proficiency across experiences

An **intelligent** agent must have the capacity to **accumulate** and

maintain task proficiency across experiences

The sequence of tasks may **not be explicitly labeled**,

tasks may **switch unpredictably**,

and any individual task may **not recur for long time** intervals

- **Curriculum** learning
  - **Teacher** – student
  - **Interferences** between examples and/or learning tasks
  - Mostly interested in the performance on the **final task**

- **Curriculum** learning
  - **Teacher** – student
  - **Interferences** between examples and/or learning tasks
  - Mostly interested in the performance on the **final task**
- ≠ **Continual** learning
  - The learner is expected to **be good** on each task
  - and to **retain** its performance on all tasks

How to use the past

- This question is already present in I.I.D. learning (classical inductive learning)

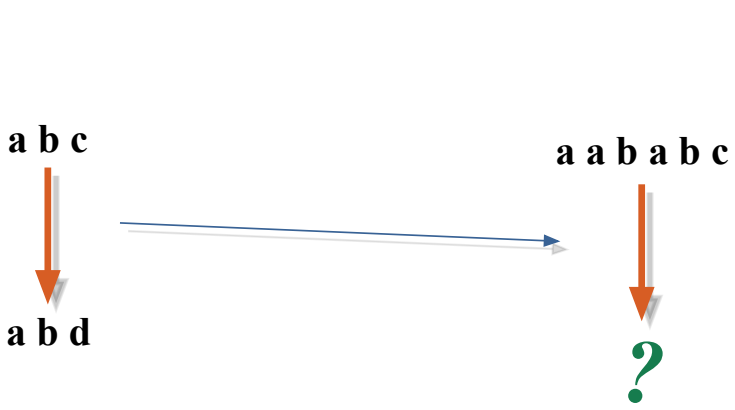
Training examples are useful only insofar as **there exists a bias**

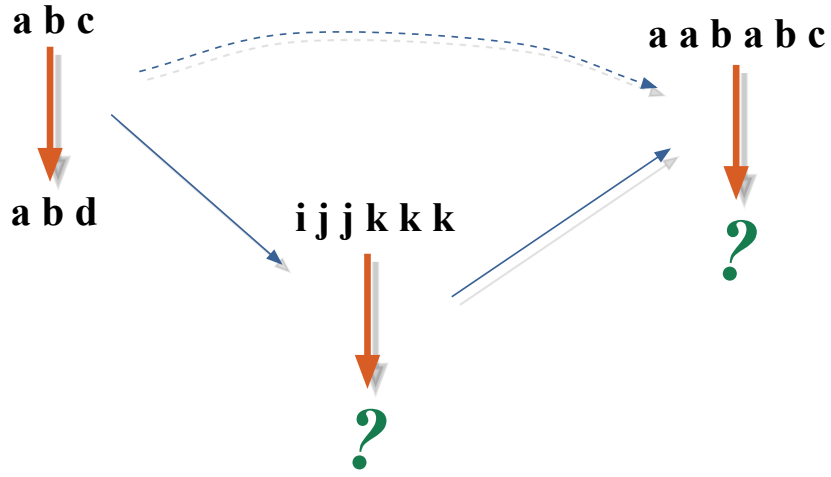
Otherwise, only **rote learning** is possible

Sequencing effects

An example of ANTI-**curriculum**

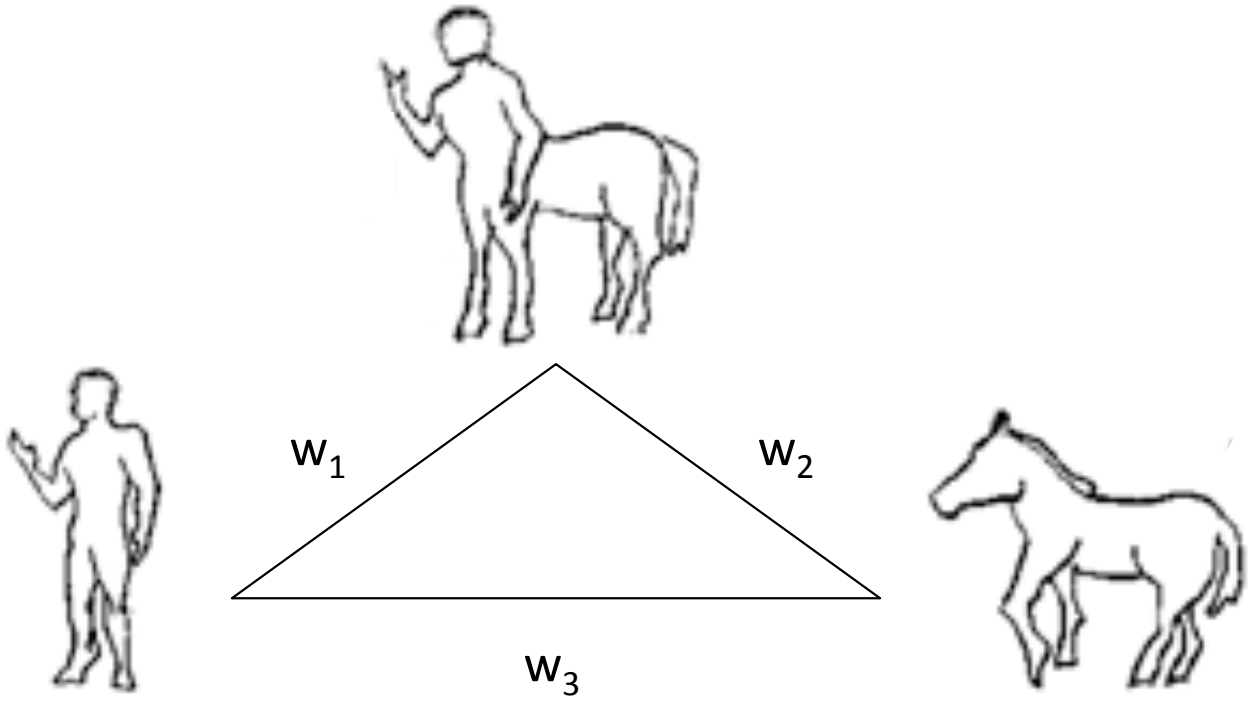
Transfer and **path dependence**





...

Need for non-symetrical similarity



$$w_3 > w_1 + w_2$$

Adapted from: D.W. Jacobs, D. Weinshall, and Y. Gdalyahu. Classication with non-metric distances: Image retrieval and class representation. PAMI 2000.

The **order** of the examples (learning tasks) is in itself **a bias**

Curriculum building

Curriculum learning

- Curriculum learning is a biologically-inspired learning paradigm. It is motivated by the observation that humans and animals **learn better** when they are exposed to **examples that are organized in a meaningful way**.
  - For example, the **visual system** of human infants develops gradually, focusing first on simple geometric patterns or objects with prominent contours before processing more complex patterns [Anderson et al., 2024].
  - **Works in ANNs** have also shown that the presence of an ordered set of **progressively more difficult tasks** can help a model to learn better and faster [Elman, 1993; Krueger and Dayan, 2009; Bengio et al., 2009|.
    - In particular, the experiments of Bengio et al. |2009| suggest that a curriculum, defined as an ordered set of experiences, makes the loss function of the model **converge towards better local minima** and **reduces the number of optimization steps** needed to reach a given performance.

Curriculum learning

- Curriculum strategies can be considered a special case of transfer learning, in which knowledge from early experiences guides the learning process for more challenging tasks.

Incremental learning

- Usually when there is no teacher to organize the order of the training examples or tasks
- It is a particular form of continual learning

- "… Unlike (statistical) machine learning, in human learning supervision is often accompanied by a curriculum. Thus the order of presented examples is rarely random when a human teacher teaches another human.
- Likewise, the task may be divided by the teacher into smaller sub-tasks, a process sometimes called shaping (Krueger & Dayan, 2009) and typically studied in the context of reinforcement learning (e.g. Graves et al., 2017).
- Although it remained for the most part in the fringes of machine learning research, **curriculum learning** has been identified as **a key challenge** for machine learning throughout."

[Daphna Weinshall et al. (2018) « **Curriculum Learning by Transfer Learning: Theory and Experiments with Deep Networks** ». ICML-2018.]

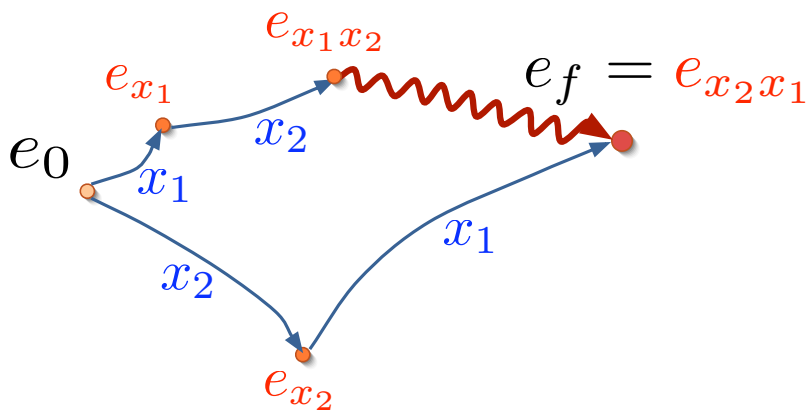
Curriculum learning

- “Humans need about two decades to be trained as fully functional adults of our society.
- That training is highly organized, based on **an education system** and a curriculum which introduces different concepts at different times, exploiting previously learned concepts to ease the learning of new abstractions.
- By choosing which examples to present and in which order to present them to the learning system, one can guide training and remarkably increase the speed at which learning can occur.”

[Joshua Bengio (2018) « **Learning deep architectures for AI** ». Now Publishers Inc, 2009.]

Order effects

- How to **predict** them?
- How to **quantify** them?
- How to **formalize** them?
- How to **control** them?



Sequencing effects

- How to eliminate them?

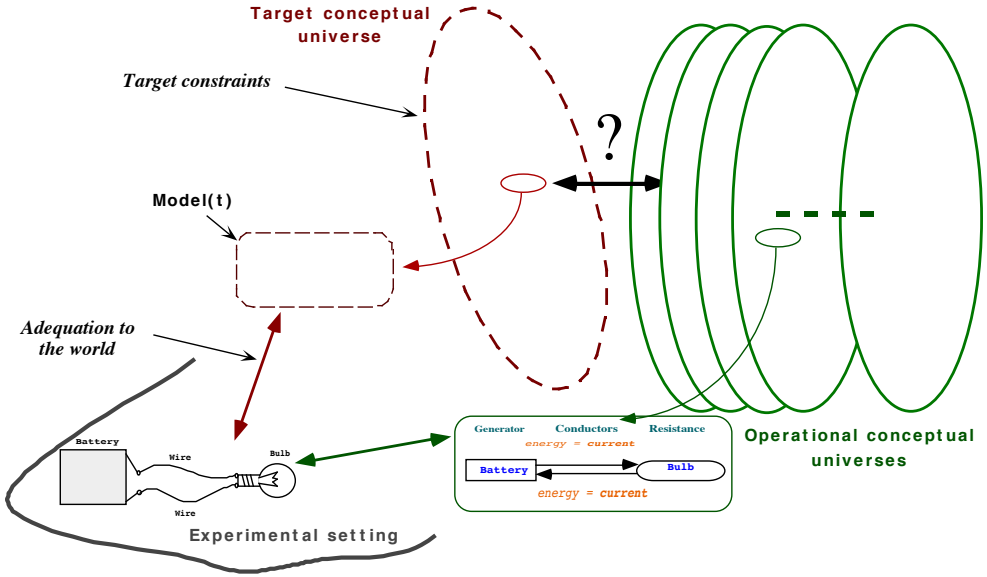
NO!

- How to organize them and **guide** learning?
- How to build a **curriculum** for machines?

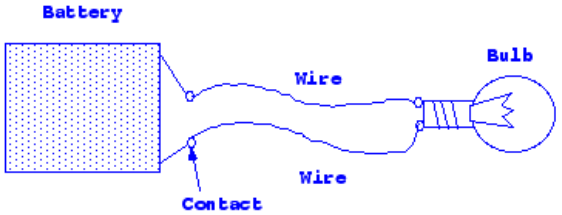
YES!

Cognitive tunnel effect

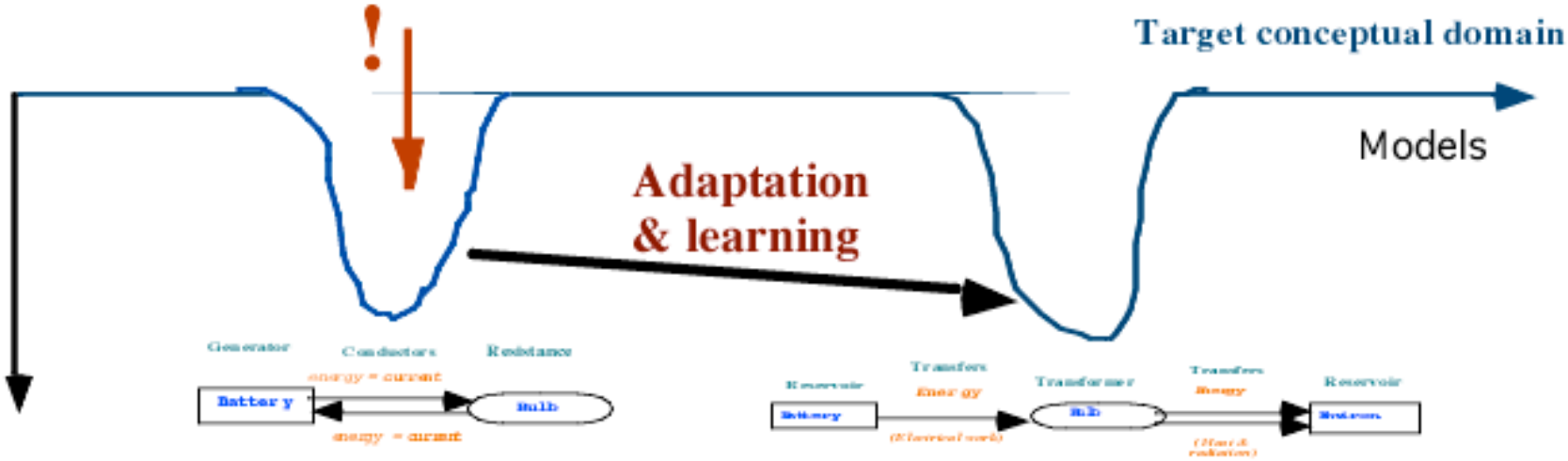
[A. Cornuéjols, A. Tiberghien, G. Collet. *Tunnel Effects in Cognition: A new Mechanism for Scientific Discovery and Education*. Arxiv-1707.04903- Tue, 18 Jul 2017 00:00:00 GMT]

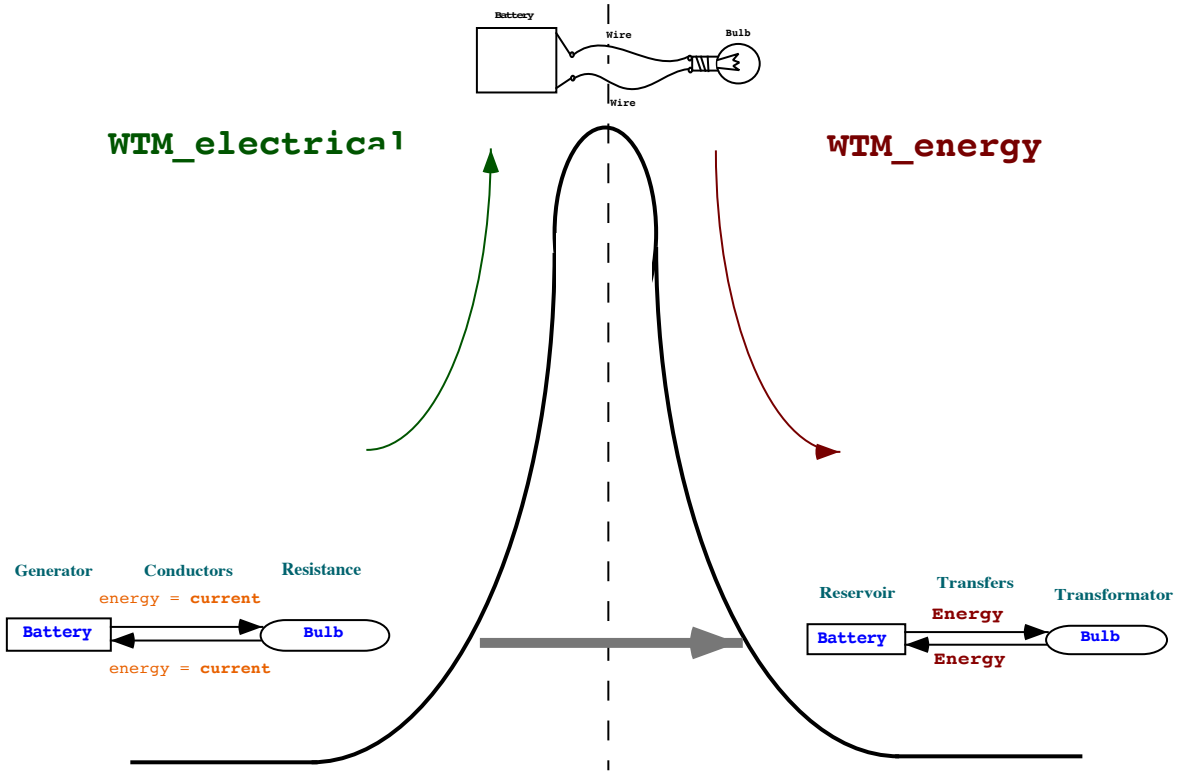


Experimental setting



Conceptual interpretation in terms of **energy chain**


Target conceptual domain
Models
Adaptation
& learning
Generator
Conductors
Resistance
energy = current
Battery
Bulb
energy = current
Transfers
Energy
Transformer
Transfers
Energy
Reservoir


Battery
Wire
Bulb
Wire
WTM_electrical
WTM_energy
Generator
Conductors
Resistance
energy = current
Battery
Bulb
energy = current
Reservoir
Transfers
Energy
Transformator
Battery
Bulb
Energy

...

What did we learn so far?

on **curriculum learning**

Teacher - student

- E.g. **Distillation**
  - Change gradually the **targets**
  - Change gradually the **source data points**
  - Change gradually the **learning task** (real curriculum learning)
    - With a teacher aware of the student

Continual Learning

- the ability to **learn continually from a stream of data**
  - **building on** what was learned previously
  - and being able to **remember those learnt tasks**.

- What humans are capable of, and what is also **the end goal for an artificially intelligent machine**.

Outline

1. How to measure the difficulty of a training example
2. Catastrophic forgetting
3. Curriculum building
4. A geometry on the space of tasks
5. Conclusions

A geometry on the space of tasks

- We expect that transfer is easy when source and target tasks are “close”
- And it may be difficult to transfer across tasks that are “far away”

But **how to measure** “*closeness*”

and “*far away*” for learning tasks?

Define a geometry over the space of tasks

- How to measure the difficulty of a new example?
- Curriculum learning
- How to best help learning?

Outline

1. A reminder about O.O.D.: fundamental questions
2. Curriculum building: a geometry on the space of tasks
3. How to measure the difficulty of a training example
4. Defining a distance between tasks
5. Conclusions

Curriculum building

And the **geometry** of the space of **learning tasks**

Geometry of the space of tasks

- Desiderata

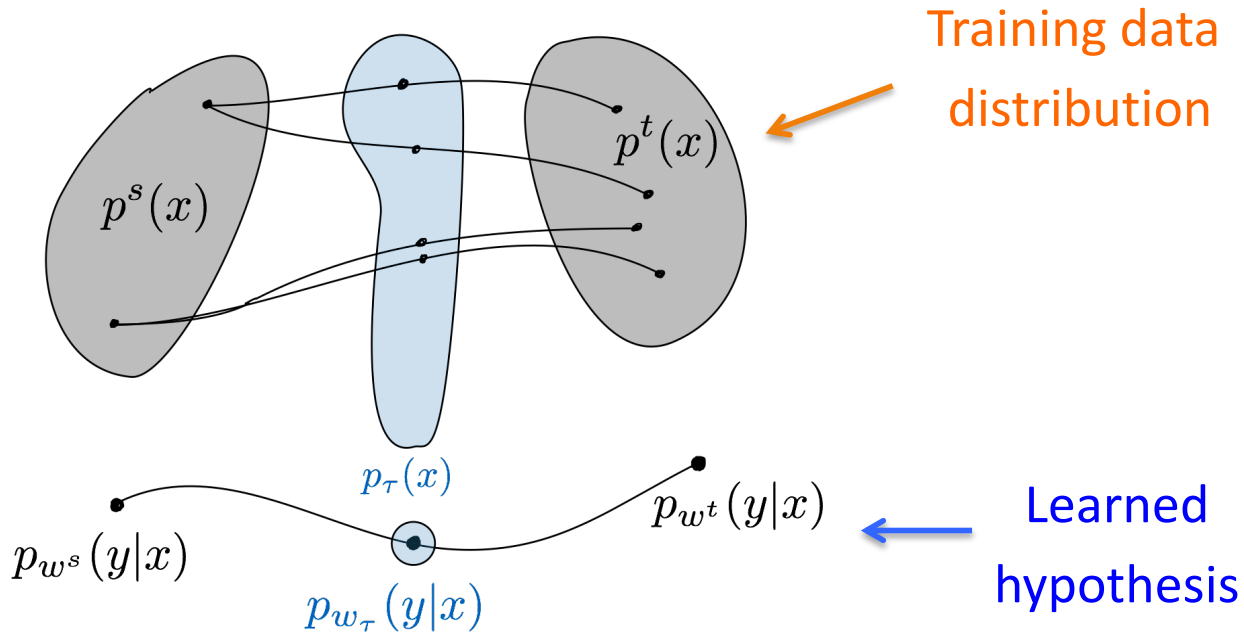
  1. Should **incorporate the hypothesis space**,
     and not only the "distance" between the inputs (as is usually done)

     - For instance, it is often observed that *transferring larger models is easier*.
       The **geometry** should **reflect** this.

  2. The distance between tasks is **not symmetrical**

Gao, Y., & Chaudhari, P. (2021, July). **An information-geometric distance on the space of tasks**.
In *International Conference on Machine Learning* (pp. 3553-3563). PMLR.

Idea



Training data distribution

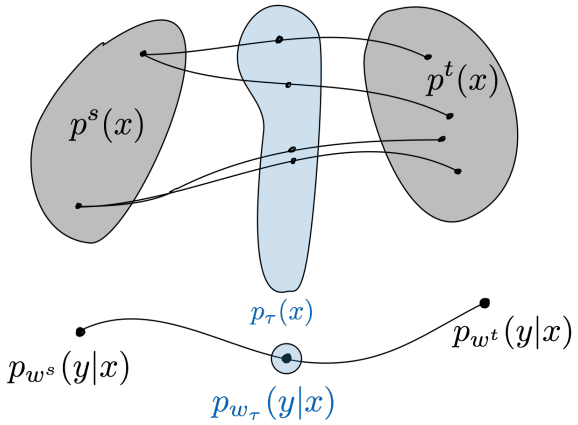
Learned hypothesis

Modify **conjointly** the training data distribution and the **learned hypothesis**

Compute iteratively the intermediate training sets such that

- at each step τ the new task is close to
- what can be learned by **the current learner** (characterized by its **current hypothesis**)

Contributions

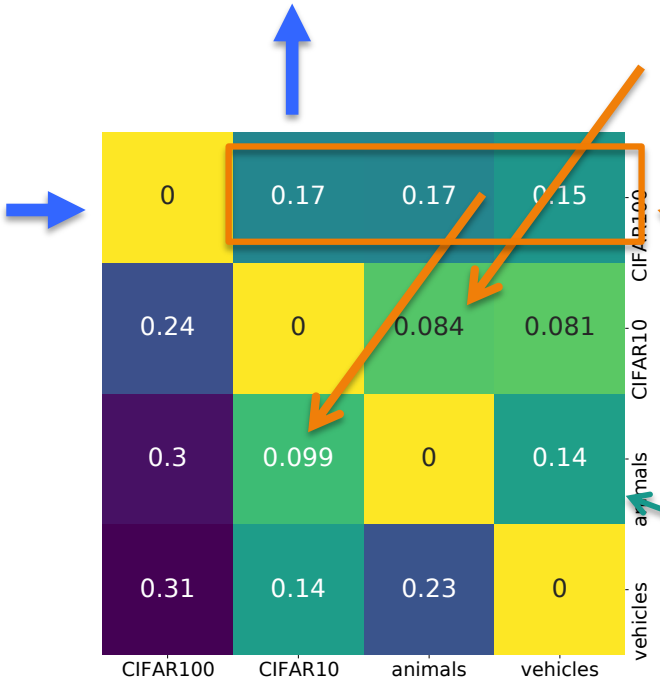


- They formalize a "coupled transfer distance" between learning tasks
  - as the **length of the shortest trajectory** on a Riemannian manifold that **the weights** of a classifier travel on when they are adapted **from the source** task **to the target** task.
  - At each instant during this transfer, weighs are fitted on an interpolating task that evolves along the **optimal transportation** (OT) trajectory between source and target tasks.

- Given an optimal transport between tasks, the Fisher-Rao distance between the initial and final weights, which the coupled transfer distance computes, corresponds to finding a weight trajectory that **keeps the generalization gap small** on the interpolated tasks

Experimental results

- Using an **8-layer convolutional NN** (ReLU, dropout, batch-normalization) with a final fully connected layer



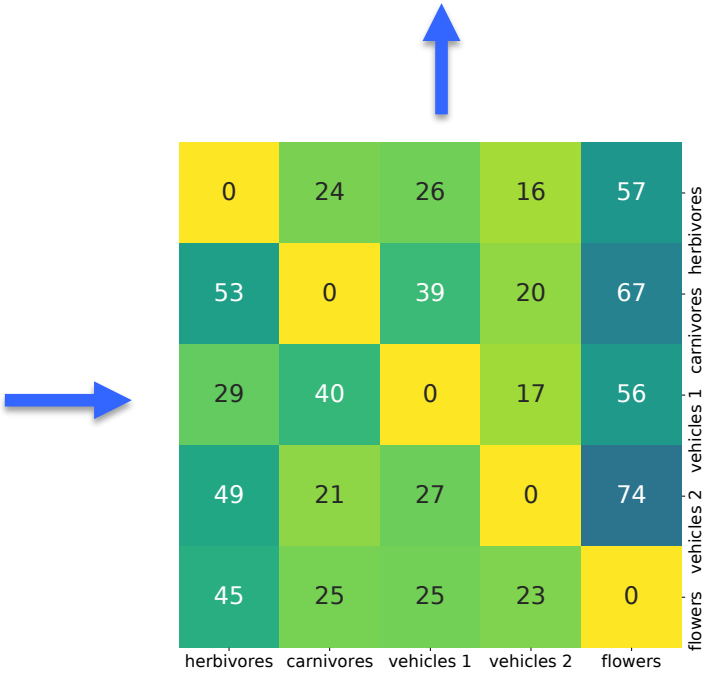
Distance is **asymmetrical**

- CIFAR-10 to animals < animals to CIFAR-10
- CIFAR-100 to any other is much easier than the reverse

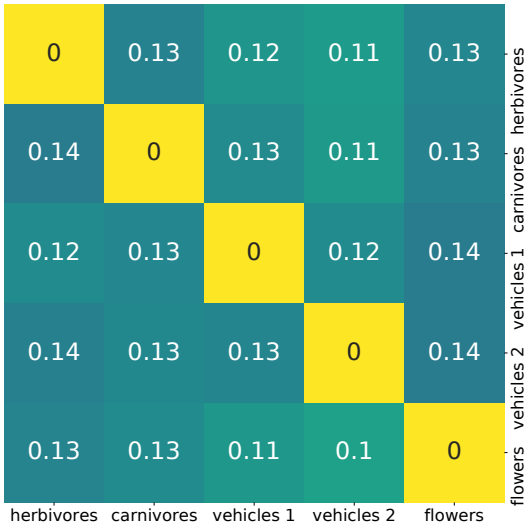
Estimated task distances

Experimental results

- Using an **8-layer convolutional NN**



- And a **wide residual network** (WRN-16-4): larger capacity



Distance is much **reduced**

using a **larger capacity** model

Conclusions

- **Interesting** work
  - New definition of **distance** between tasks
    - **Asymmetrical**
    - Depends on the **capacity** of the learning system
  - New way to build a **curriculum**

Conclusions

- Interesting work
  - New definition of **distance** between tasks
    - **Asymmetrical**
    - Depends on the **capacity** of the learning system
  - New way to build a **curriculum**

- **Limits**
  - Still a **crude** way to build intermediate tasks
  - **Same** input-output source and target domains!!!
    - Because the distance is based on the Kullback-Leibler divergence
  - **Same hypothesis space** in both source and target domains!!!

Conclusions

- Interesting work
  - New definition of **distance** between tasks
    - **Asymmetrical**
    - Depends on the **capacity** of the learning system
  - New way to build a **curriculum**

Not **general** enough for transfer learning

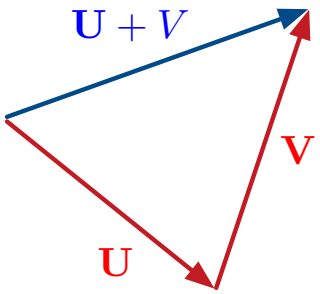
- **Limits**
  - Still a **crude** way to build intermediate tasks
  - **Same** input-output source and target domains!!!
    - Because the distance is based on the Kullback-Leibler divergence
  - **Same hypothesis space** in both source and target domains!!!

Parallel Transport

and Covariant Derivative

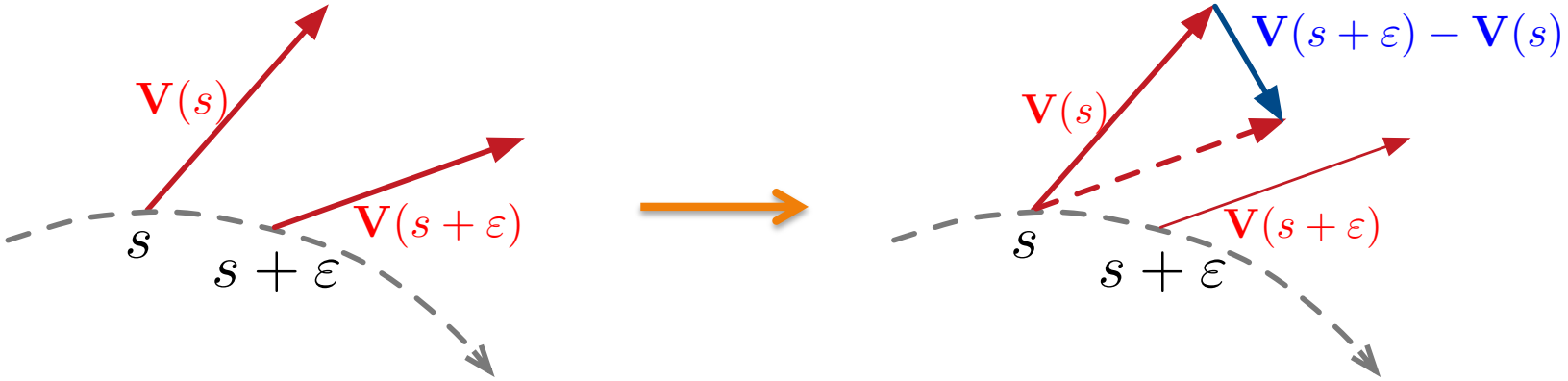
Euclidian geometry

- **Addition** of vectors



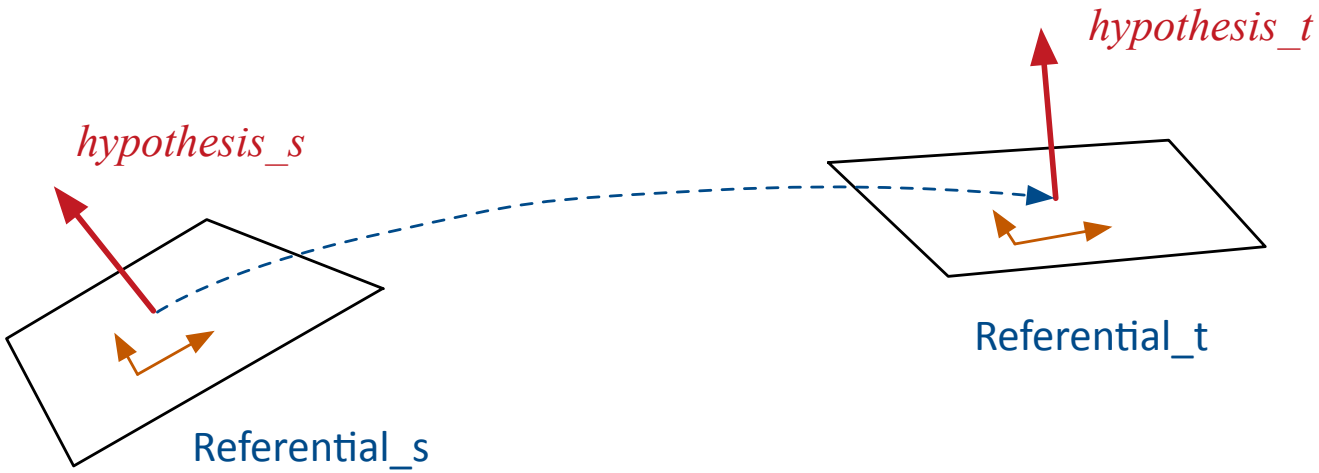
- **Substraction** of vectors and **derivative**

$$\frac{\mathrm{d}\mathbf{V}}{\mathrm{d}s} = \lim_{\varepsilon \to 0} \frac{\mathbf{V}(s+\varepsilon) - \mathbf{V}(s)}{\varepsilon}$$



Non Euclidian geometry

- Substraction of vectors and **derivative**



We can **no** longer **directly compare** vectors (or tensors)

Necessity of the covariant derivative

Parallel transport

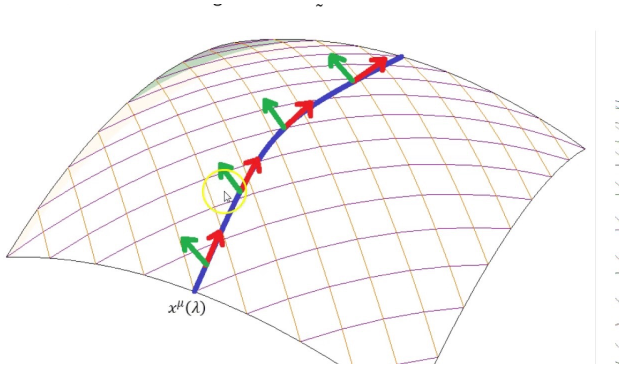
- **Transport** a vector (or a tensor) **parallel to itself** along a curve

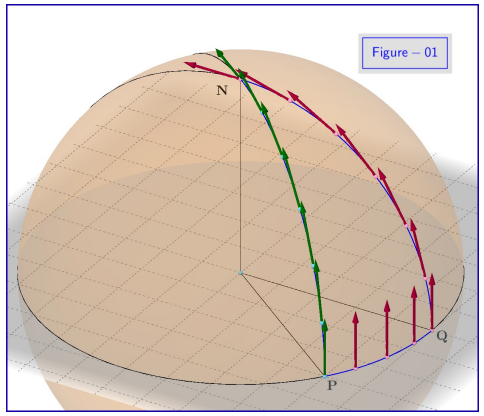
Covariant derivative = 0

Kronecker symbol

$$(\partial_k V^i)^{\text{covariant}} = \partial_k V^i + \Gamma^i_{jk} V^j$$

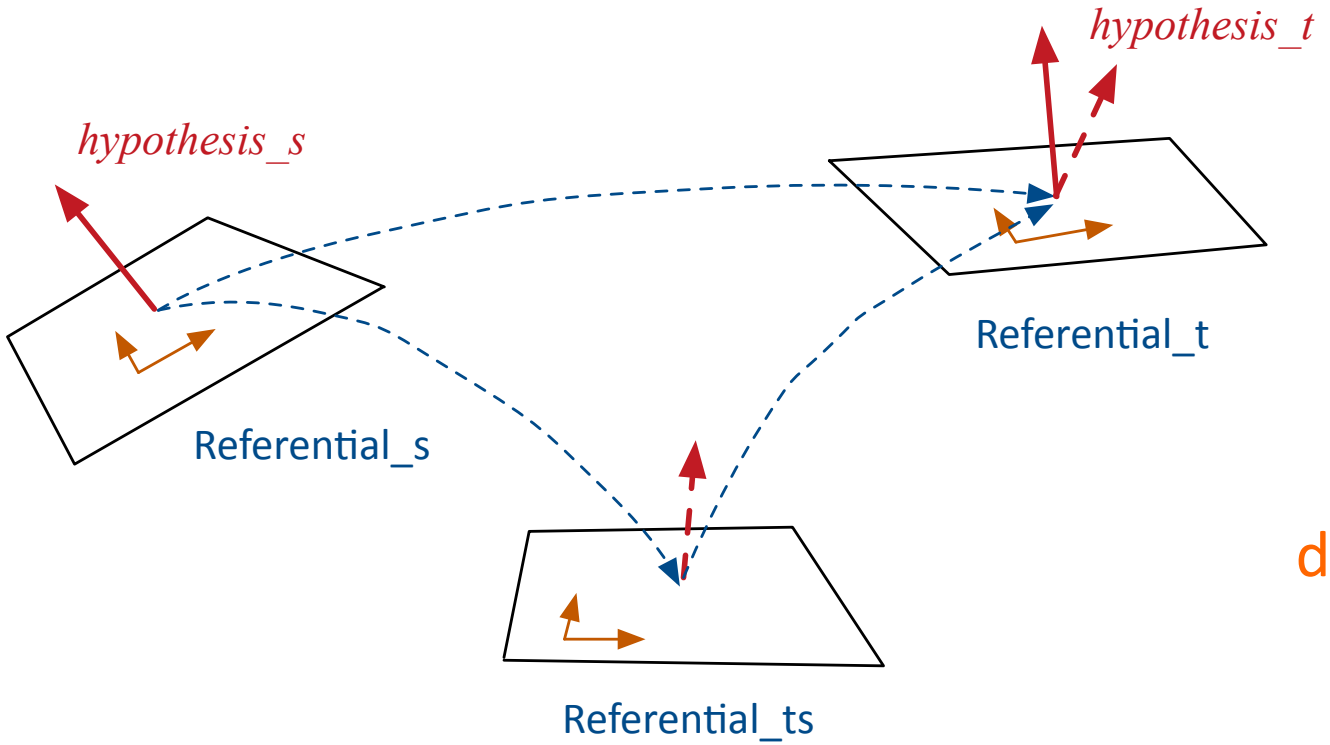
$$V^i(x^k)^{\text{parallel transported}} = V^i(x^k) + \Gamma^i_{jk} V^j \Delta x^k$$





Path **dependent**!

Transfer and **path dependence**

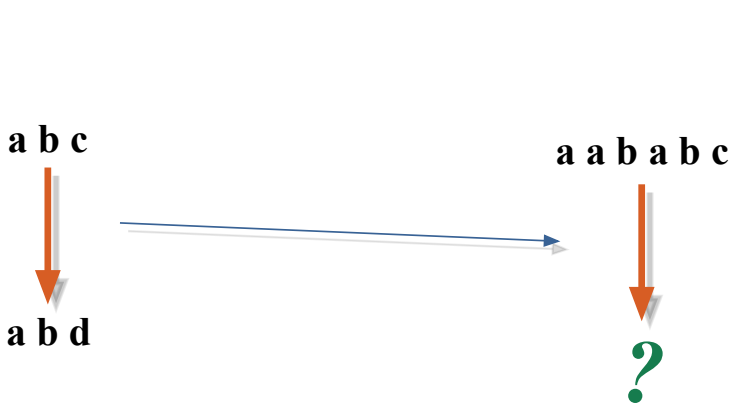


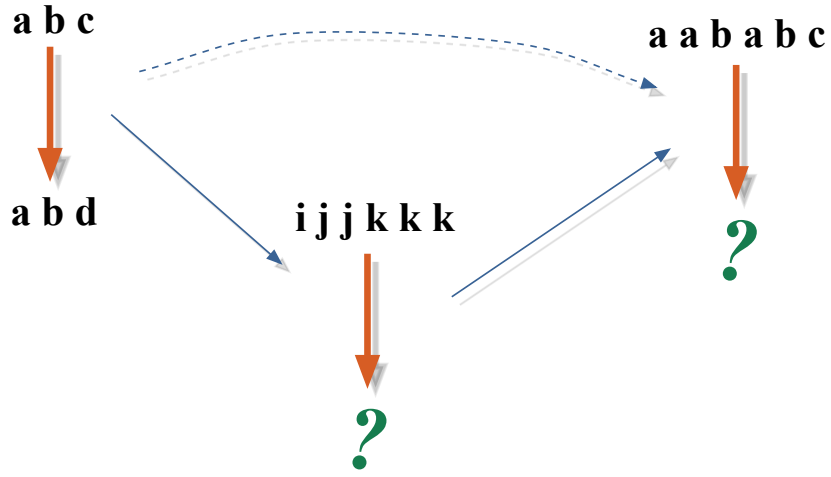
Path dependence

Transfer $\stackrel{?}{=}$ **Parallel transport** of hypothesis from source to target

But, **how** to compute it between different **source** and **target domains**?

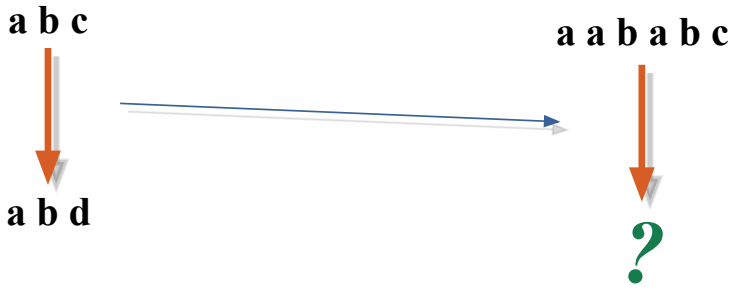
Transfer and **path dependence**

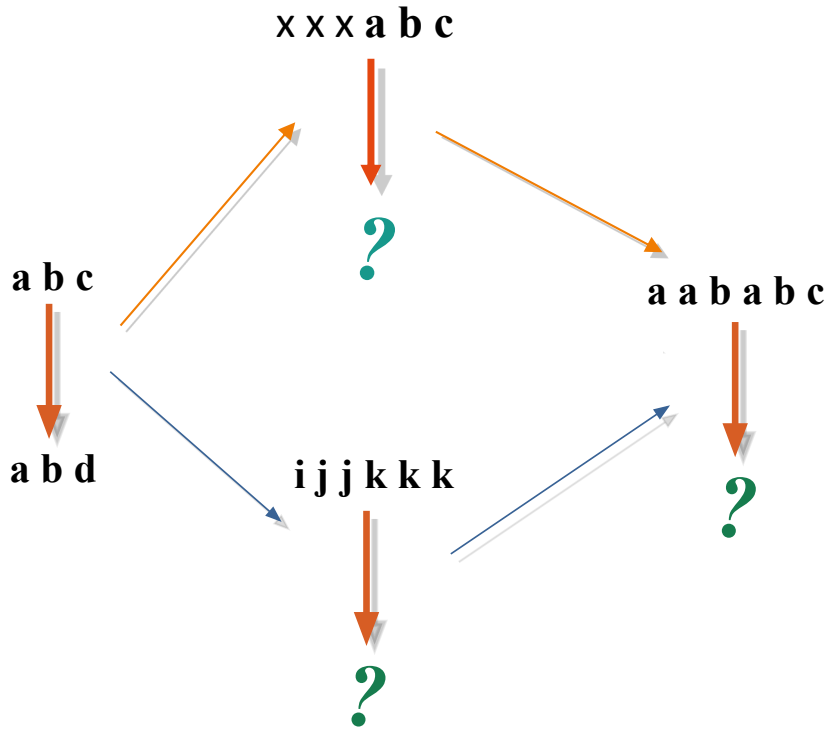




...

Transfer and **path dependence**





...

Outline

1. How to measure the difficulty of a training example
2. Catastrophic forgetting
3. Curriculum building
4. A geometry on the space of tasks
5. Conclusions

- **Transfer** learning
  - ability to **use** what has been learned **from a previous task** on a **new task**.

    The **difference with continual learning** is that transfer learning is not concerned about keeping the ability to solve previous tasks.

- **Curriculum** learning
  - a training process that proposes a **sequence of more and more difficult tasks** to a learning algorithm in order to make it able to **learn, at last**, a generally **harder task**.

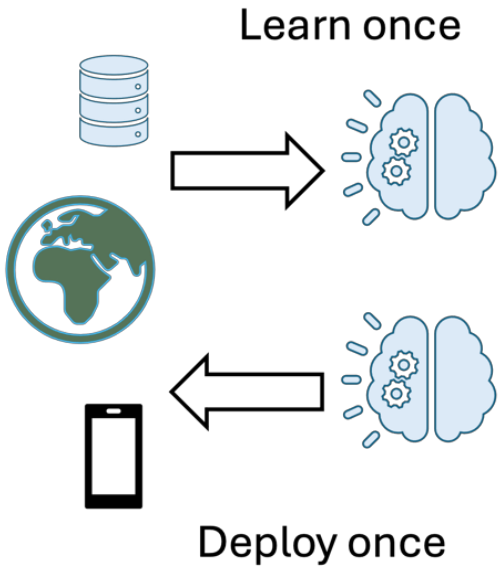
    The sequence of tasks is designed in order to be able to learn the last one.

Continual Learning

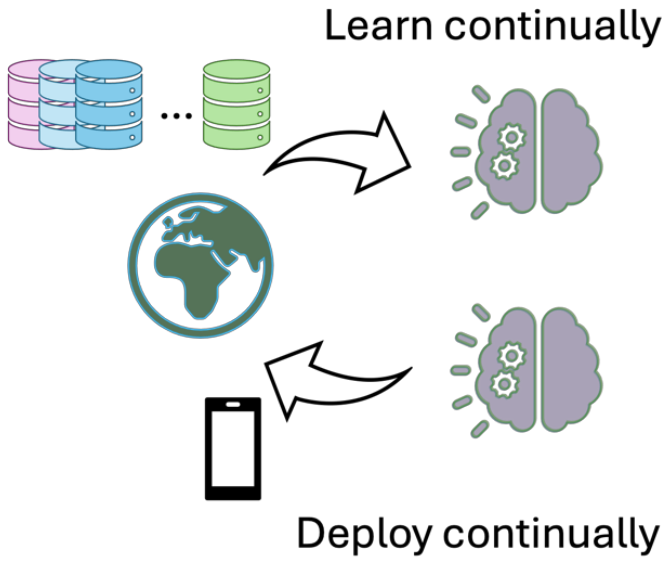
- the ability to **learn continually from a stream of data**
  - **building on** what was learned previously
  - and being able to **remember those learnt tasks**.

- What humans are capable of, and what is also **the end goal for an artificially intelligent machine**.

LLMs and **Continual** learning

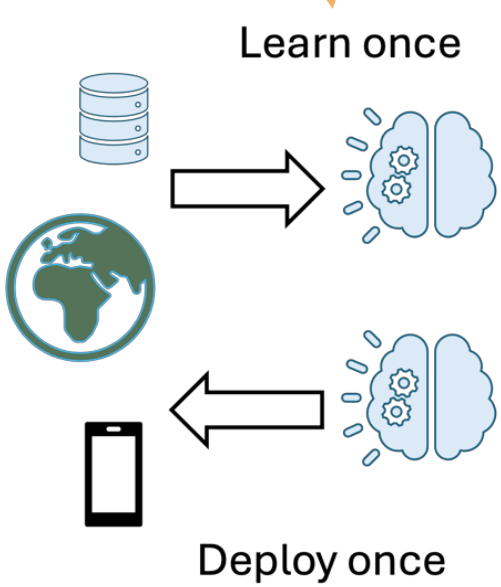


Static Machine Learning

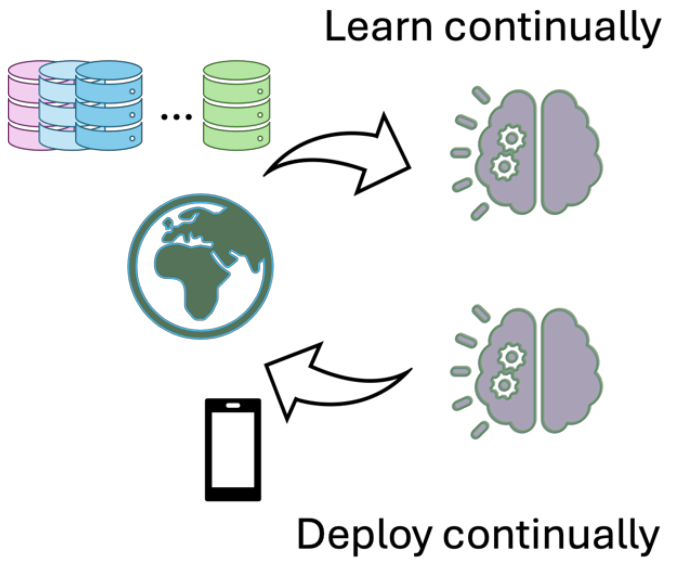


Continual Machine Learning

LLMs and **Continual** learning: not solved yet



Static Machine Learning



Continual Machine Learning

Issues

- **In numerous cases**, transfer learning works well

- **But** in other cases, it does not
  - A pretrained model on ImageNet leads to poor performance on MRI images [Merkow, et al. 2017]

- And **we still cannot** predict how transfer will fare from one learning task to another and the reasons for success or failure

Transfer learning and **curriculum** learning

- An active and constructive viewpoint:
  - Training a system for a target task through **successive intermediate** learning **tasks**
  - Necessitates
    - To **identify** relevant intermediate subtasks
    - To **order** them

Curriculum learning

Conclusions (1)

Transfer learning → mostly heuristical approaches so far

1. **Parallel transport** is a natural way for looking at **transfer** learning
   - The **covariant derivative** is then a measure of difference
     - **How** to compute it?
       - Pioneering works in **computer vision**
     - What about when the **source** and **target** domains are **different**?
       - TransBoost: a **proposal**

2. Transfer learning is **path dependent** in general
   - The study of these path dependencies is **important** ...
     - Curriculum learning
     - Longlife learning
   - ... and a wide **open research question**

Conclusions (2)

- The **theoretical guarantees** for transfer learning:
  - Do not necessarily depend on the **performance of the source hypothesis** h_S

    But depend on the **bias** that h_S determines
  - Involve the **capacity** of the space of **transformations**

    (and **the path** followed between source and target)

Still to be explored

Bibliography

- Baldock, R., Maennel, H., & Neyshabur, B. (2021). **Deep learning through the lens of example difficulty**. *Advances in Neural Information Processing Systems*, *34*.
- Bauer, M., Klassen, E., Preston, S. C., & Su, Z. (2018). **A diffeomorphism-invariant metric on the space of vector-valued one-forms**. arXiv preprint arXiv:1812.10867.
- Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., & Vaughan, J. W. (2010). A theory of learning from different domains. *Machine learning*, *79*(1-2), 151-175.
- Cornuéjols A., Murena P-A. & Olivier R. "*Transfer Learning by Learning Projections from Target to Source*". Symposium on Intelligent Data Analysis (IDA-2020), April 27-29 2020, Bodenseeforum, Lake Constance, Germany.
- Cornuéjols, A. (2024). Some thoughts about transfer learning. What role for the source domain? , *International Journal of Approximate Reasoning (IJAR)* 166, 109107.
- Gao, Y., & Chaudhari, P. (2021, July). An information-geometric distance on the space of tasks. In *International Conference on Machine Learning* (pp. 3553-3563). PMLR.
- Kuzborskij, I., & Orabona, F. (2013, February). Stability and hypothesis transfer learning. In *International Conference on Machine Learning* (pp. 942-950).
- Mansour, Y., Mohri, M., & Rostamizadeh, A. (2009). Domain adaptation: Learning bounds and algorithms. *arXiv preprint arXiv:0902.3430*.
- Redko, I., Morvant, E., Habrard, A., Sebban, M., & Bennani, Y. (2019). *Advances in Domain Adaptation Theory*. Elsevier.
- Schonsheck, S. C., Dong, B., & Lai, R. (2018). **Parallel transport convolution: A new tool for convolutional neural networks on manifolds**. arXiv preprint arXiv:1805.07857.
- V. Vapnik and A. Vashist (2009) "A new learning paradigm: Learning using privileged information". *Neural Networks*, vol. 22, no. 5, pp. 544–557, 2009
- H. Venkateswara, S. Chakraborty, and S. Panchanathan, "Deep-learning systems for domain adaptation in computer vision: Learning transferable feature representations," *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 117–129, 2017.
- Yosinski, J., Clune, J., Bengio, Y., & Lipson, H. (2014). How transferable are features in deep neural networks?. In *Advances in neural information processing systems* (pp. 3320-3328).
- Zhang, C., Zhang, L., & Ye, J. (2012). Generalization bounds for domain adaptation. In *Advances in neural information processing systems* (pp. 3320-3328).

What did we learn so far?

- Domain adaptation

  - $X_S = X_T$ & $Y_S = Y_T$ & $P_{Y|X} = C^{te}$

  - **Reweighting** the training examples

    - So as to **put more weight** on the training examples "close" to the target distribution

      - But **few test examples** → Estimate target distribution?

      - **How** to estimate the amount of reweighting?

Translation

- Domain adaptation / Transfer learning
  - $(X_S = \text{or} \neq X_T)$ & $(Y_S = \text{or} \neq Y_T)$ & $(P_{Y|X} = \text{or} \neq C^{te})$
  - Assumption: **something** is **shared** between the *source* and the *target*
    - A **representation**
      - E.g. First layers of NNs
    - The **decision** function
      - E.g. TransBoost (there are theoretical guarantees! Same as boosting)
    - **Invariant** representation between environments
      - E.g. Invariant Risk Minimization (IRM)
    - **Causality**

- **Analogy** making

Maximum relevance of the source

  - $(X_S = \text{or} \neq X_T)$ & $(Y_S = \text{or} \neq Y_T)$ & $(P_{Y|X} \neq C^{te})$

  - Identify both:
    - **Good latent spaces** (local representations) of the source and of the (incomplete) target
    - And an **economical translation** between these representations

E.g. using MDLp: Minimum Description Length principle

- In
  - **Domain** adaptation
    - $X_S = X_T$ & $Y_S = Y_T$ & $P_{Y|X} = C^{te}$
  - **Transfer** learning
    - $X_S \neq X_T$ and/or $Y_S \neq Y_T$ & $P_{Y|X} \neq C^{te}$
    - The goal is to **be good on the target**, not to stay good on the source
  - **Continual** learning (incremental learning)
    - $X_S = X_T$ & $Y_S = Y_T$ & $P_{Y|X} \neq C^{te}$
    - The goal is to **be good on the target** AND to **stay good on the source**
  - **On line** learning
    - $X_S = X_T$ & $Y_S = Y_T$ & $P_{Y|X} \neq C^{te}$
    - The goal is to be **good at each time step**

The source is part of **the bias**

Questions

- In which cases is the **source useful**?
  - What are the conditions
    - On the **source**?
    - On the **relationships** between the source and the target?
      - Kullback-Leibler ; MDLp ; boosting of weak translators ; …
- Can we obtain **theoretical guarantees**?
  - As in PAC learning
- **What should be transmitted** from the source to the target?
  - A **representation**?
    - **Common** to source and target (e.g. as in deep NNs)
    - **Changed** (e.g. as in analogy making)
  - A **decision** function? (e.g. as in TransBoost)
  - …
- **How** should the transmission be done?
  - Reweighting
  - translation
  - …

h_S does **not** have to be good on the source

h_S must act as a **good bias**

How to measure this before hand?

(Same problem already in I.I.D. learning!)