LECTURE 15

Pretraining, Transfer Learning, and Fine-tuning

07-280

Aviral Kumar & Nihar Shah

Representations

An example: recognizing flowers

- **Suppose we have 200 labeled photos of daisies and tulips.**
- We want to classify new photos taken by our camera.
- An existing image model has already learned from many other images.
- What can we reuse, and what still needs to be learned?

What is a representation?

- **A representation describes an input in a form a model can use.**

- In a neural network, it is often the activations produced by hidden layers.

- Image pixels are one representation. Learned features are another.

$$h = f_\theta(x)$$

What is representation learning?

- **Representation learning uses data to learn how to compute useful features.**
- Training changes the network weights, which changes the features it computes.
- For flowers, useful features might capture shape, texture, or combinations of visual patterns.
- “Useful” always depends on the task and learning objective.

Two parts of a prediction model

Part	What it does	Flower example
Backbone / encoder	Converts an input into a feature vector	Extract visual features from a photo
Prediction head	Converts features into task outputs	Produce daisy and tulip scores

- The backbone supplies features. The head uses them to make a prediction.

Pretraining and transfer

Pretraining: the first learning stage

- **Pretraining is an initial training stage that produces weights for later use.**
- Example: train an image model on a large collection of labeled images.
- The saved weights are a checkpoint: a starting point we can load later.
- Pre-training can use labels or a task constructed from the data.

Transfer learning: reusing prior learning

- **Transfer learning uses knowledge learned in one setting to help in another.**
- Source setting: the data and task used to learn the existing model.
- Target setting: the data and task we want to solve now.
- Our example transfers visual features to daisy-versus-tulip classification.

Adaptation: making a model useful for the target

- **Adaptation is the process of adjusting a model or its use for a target problem setting.**

- We might train a new head, update existing weights, or add a small trainable module.

- We can also change how we use a frozen model, for example with class descriptions in CLIP.

Fine-tuning: further training of a pretrained model

- **Fine-tuning continues training a pretrained model for the target task.**
- For flowers, we use labeled target photos to update some or all of the backbone.
- The features can change to better support the new prediction task.

How the different paradigms fit together

Term	Main idea	In the flower example
Pre-training	Learn an initial set of weights	Train on source images
Transfer learning	Reuse prior learning in a target setting	Reuse visual features
Adaptation	Choose how the model serves the target	Train a head or update weights
Fine-tuning	Further train the pretrained model	Update the backbone

Learning objectives

Supervised (“pre”)training: cross-entropy

- **An image x has a correct class y. The model has weights θ.**

$$\ell(x, y) = -\log p_\theta(y \mid x)$$

$$L_{\mathrm{CE}} = \frac{1}{N} \sum_{i=1}^{N} \ell(x_i, y_i)$$

- N is the number of labeled examples. Average their individual losses.

- We have seen this before!

Unsupervised and self-supervised learning

Term	Where the learning signal comes from	Example
Unsupervised learning	Structure in data without external task labels	Group similar examples
Self-supervised learning	A prediction target constructed from the data itself	Predict a missing word

- Self-supervised learning is often considered a form of unsupervised learning.

Language-model pre-training

Observed context	Prediction target
The cat sat on the	mat
The cat sat on the mat	.

$$L_{\mathrm{LM}}(\theta) = -\frac{1}{T}\sum_{t=1}^{T} \log p_\theta(x_t \mid x_{<t})$$

- x_t is the observed next token. $x_{<t}$ means all earlier tokens.
- T counts total predicted predicted tokens. θ denotes the model weights.
- For “The cat sat on the”, p(mat) = 0.8 gives token loss ≈ 0.22.

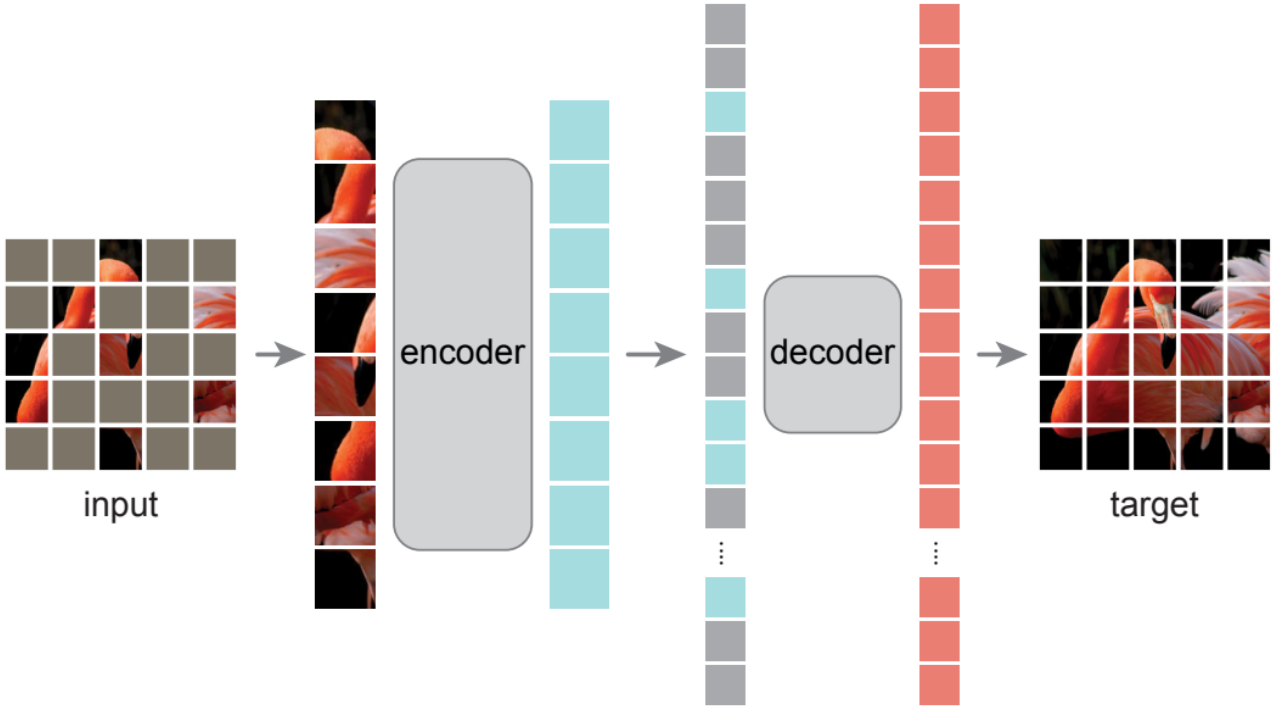
Brown et al. (2020) / Devlin et al. (2019)

Masked image prediction: MAE

- **Main idea:** Hide image patches. Predict their pixels.

- Visible patches provide context. Later, reuse the encoder for recognition.

$$L_{\text{MAE}} = \frac{1}{|M|} \sum_{r \in M} (\hat{x}_r - x_r)^2$$

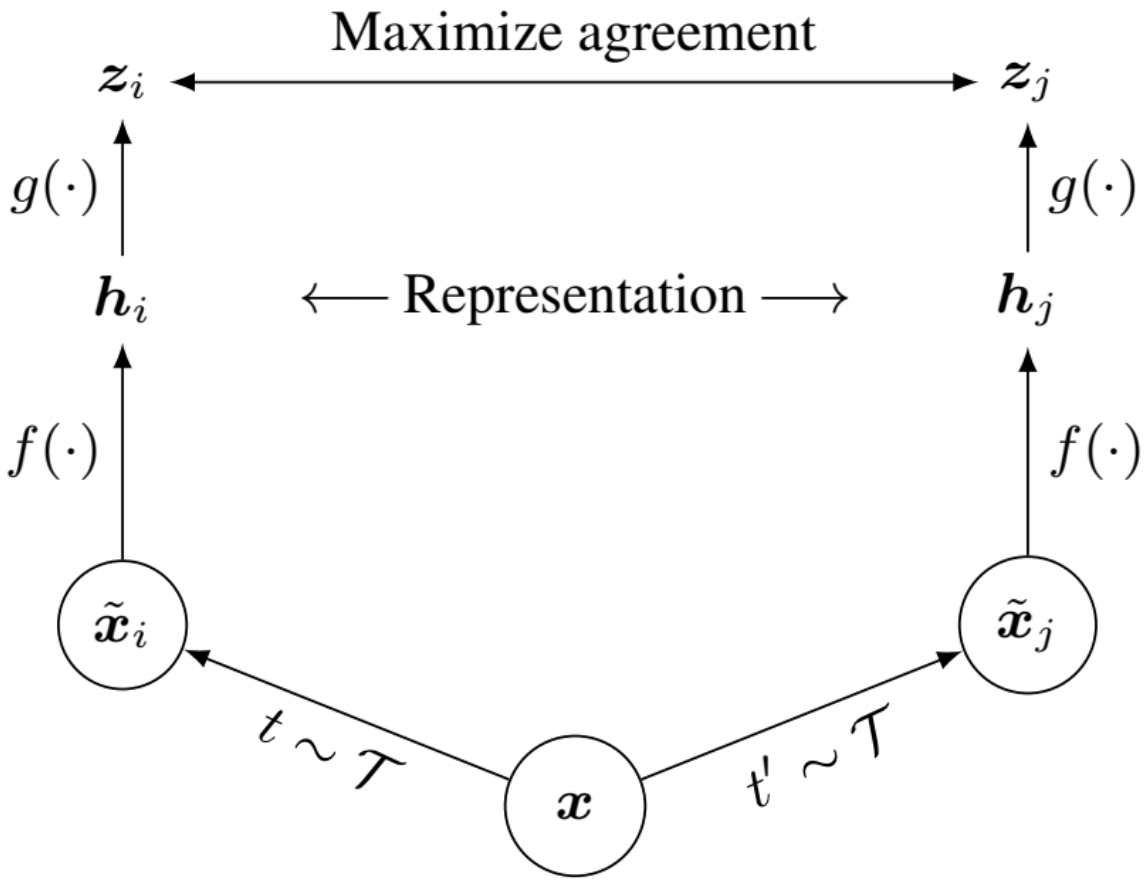


- M indexes hidden pixel values. |M| denotes the total count of x. $\hat{x}_r$ predicts x_r.

He et al., 2022, Fig. 1

Contrastive image learning: SimCLR

- **Make two transformed views of one image.**
- Train their representations to match relative to other images.
- The chosen transformations teach which augmentations to images to ignore.
- “Data augmentation”



Chen et al., 2020, Fig. 2

SimCLR: the contrastive loss

- **N images give 2N views. For anchor i, j = j(i) is its matching view.**

$$s_{ik} = \frac{\mathrm{sim}(z_i, z_k)}{\tau}$$

- z is a projected feature vector. sim is cosine similarity. τ > 0 is temperature.

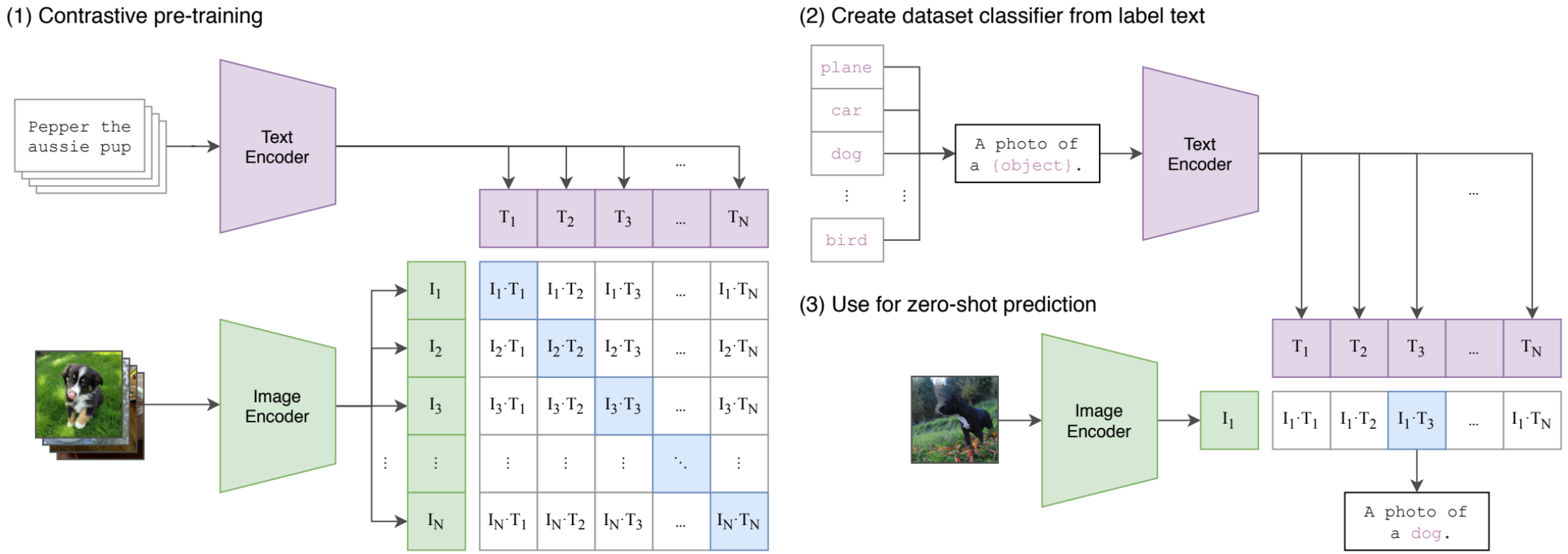
$$\ell_{i,j} = -\log \frac{\exp(s_{ij})}{\sum_{k \neq i} \exp(s_{ik})}$$

- The denominator uses every other view, including the matching view j.

$$L_{\mathrm{SimCLR}} = \frac{1}{2N} \sum_{i=1}^{2N} \ell_{i,j(i)}$$

Chen et al., 2020, Eq. 1 and Algorithm 1

CLIP: learning from image–text pairs



- Image and text encoders learn to match corresponding images and captions.

Radford et al., 2021, Fig. 1

CLIP: matching images and captions

- **N paired examples: image i belongs with caption i.**

$$s_{ij} = \frac{u_i \cdot v_j}{\tau}$$

- u and v are unit-length image and text vectors. τ > 0 sets the score scale.

$$L_{\mathrm{T}\to\mathrm{I}} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{\exp(s_{ii})}{\sum_{j=1}^{N}\exp(s_{ji})} \qquad L_{\mathrm{I}\to\mathrm{T}} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{\exp(s_{ii})}{\sum_{j=1}^{N}\exp(s_{ij})}$$

$$L_{\mathrm{CLIP}} = \frac{L_{\mathrm{I}\to\mathrm{T}} + L_{\mathrm{T}\to\mathrm{I}}}{2}$$

Radford et al., 2021, §2.2 and Fig. 3

Adaptation methods

A transfer problem needs a source and a target

Question	Source setting	Target setting
What data?	A broad image collection	Our flower photos
What prediction?	Source categories or a self-supervised task	Daisy or tulip
What do we reuse?	A learned backbone checkpoint	Its features or initial weights

- Success means useful performance on new target examples.

Linear probing: a classifier on fixed features

- **Freeze the pretrained backbone: do not update its weights.**
- Train a new linear prediction head using labeled target examples.
- A linear head computes a weighted sum of features plus a bias for each class.

$$\text{class scores} = W\,h + b$$

What a linear probe tells us

- **It asks whether a simple classifier can use the fixed representation.**
- Good performance means useful class information is accessible to that classifier.
- Poor performance can mean the needed information is missing or difficult to read out linearly.
- **Key limitation:** A nonlinear head or a changed backbone can give a different answer.

Partial and full fine-tuning

Method	Backbone weights	Prediction head
Linear probe	All frozen	Train a new linear head
Partial fine-tuning	Update selected layers	Train the head
Full fine-tuning	Update all layers	Train the head

- More trainable weights give more flexibility and more opportunities to overfit.

Adaptation architectures

Where adaptation changes the model

Approach	Where learning happens	Original backbone weights
New prediction head	After the backbone features	Frozen
Partial / full tuning	Inside selected original layers	Updated
Adapter modules	Small added modules inside the network	Mostly frozen
LoRA	Small learned updates to selected matrices	Stored base weights frozen

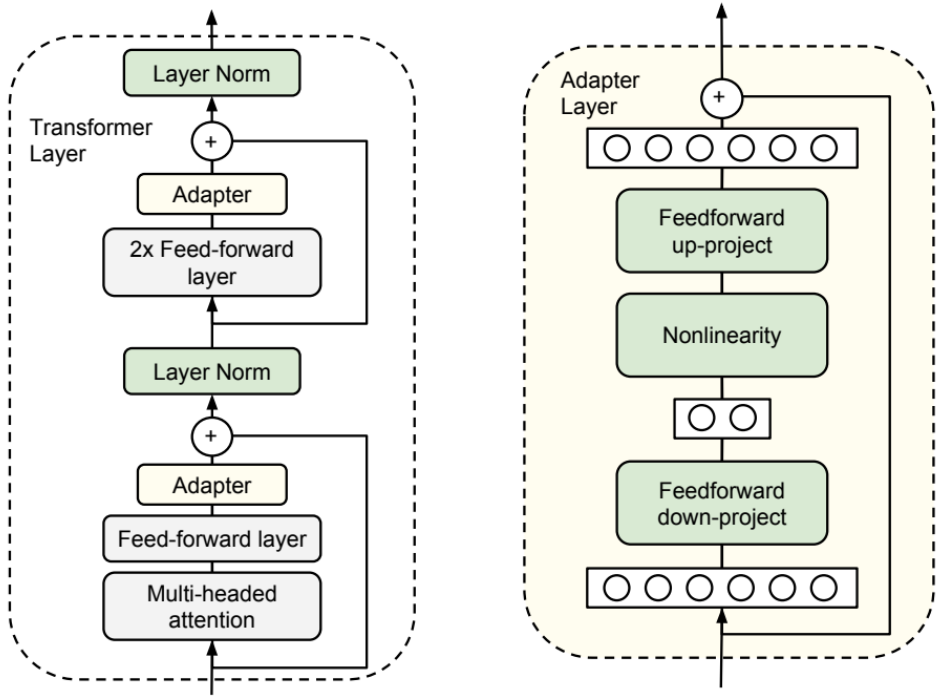
- The training objective and the update architecture are separate choices.

Houlsby et al. (2019), Parameter-Efficient Transfer Learning for NLP. / Hu et al. (2022)

Adapters: small modules inside a frozen network

- **An adapter learns a correction to an intermediate representation.**

- A bottleneck maps features to fewer dimensions, applies a nonlinearity, then maps back.



- Train the small modules and target head while retaining most pretrained weights.

Houlsby et al., 2019, Fig. 2

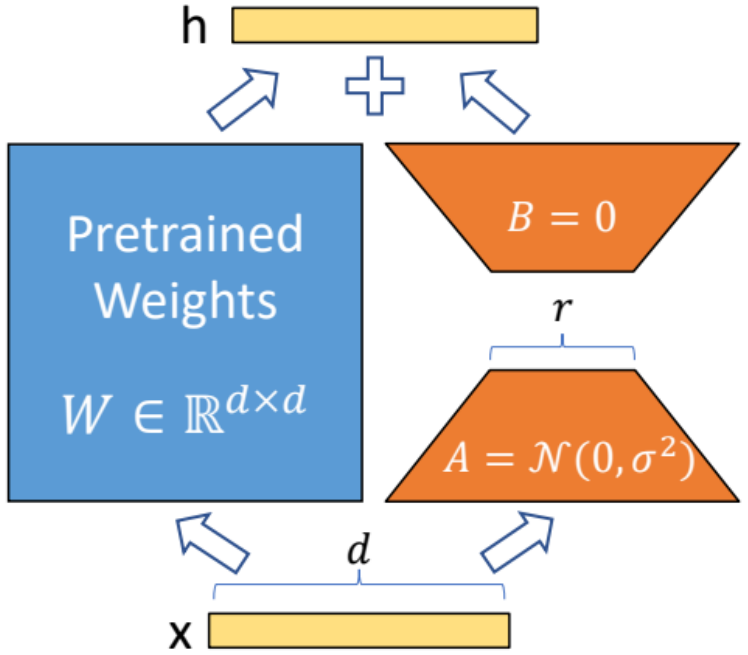
LoRA: Low-Rank Adaptation

- **Keep the base matrix W_0 fixed.**

- Learn two smaller matrices, A and B, whose product changes the transformation.

$$W = W_0 + BA$$

- Train A and B. The figure labels their initial values.



Hu et al., 2022, Fig. 1

Adaptation for a language model

Procedure	Learning signal or input	Weights change?
Continued pre-training	More domain text and next-token targets	Yes
Supervised fine-tuning	Examples of desired prompt–response behavior	Yes
Prompting	Instructions or examples supplied as context	No

- Full updates, adapters, or LoRA can support a training objective.

Brown et al. (2020) / Gururangan et al. (2020) / Ouyang et al. (2022)

Supervised fine-tuning: response-token loss

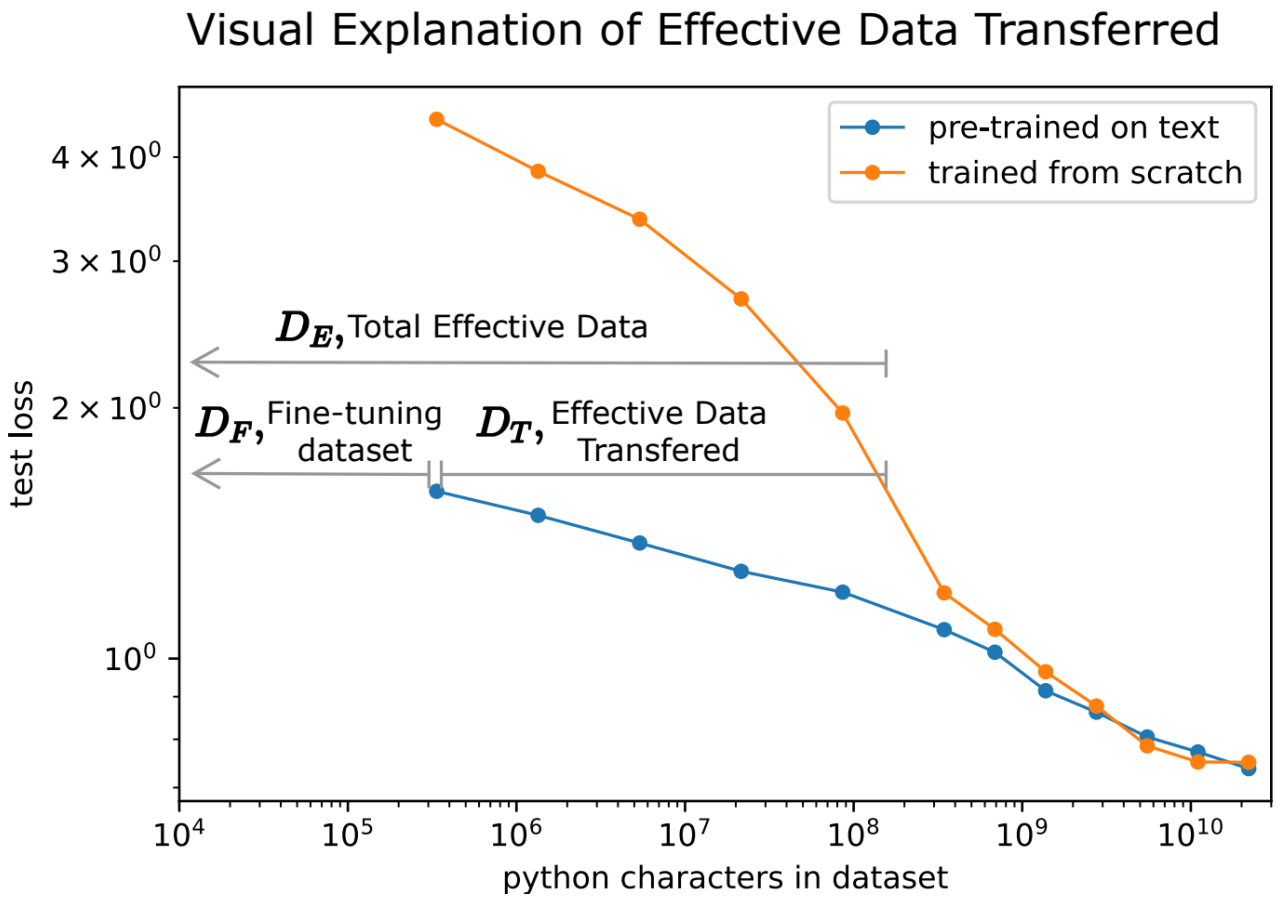
- **A prompt q comes with a desired response $y_1, \ldots, y_m$.**
- Example prompt: “Explain photosynthesis.”
  Example response: “Plants use light to …”

$$L_{\mathrm{SFT}}(\theta) = -\frac{1}{m}\sum_{t=1}^{m} \log p_\theta(y_t \mid q, y_{<t})$$

- m counts response tokens. The prompt and earlier response tokens supply context.

Evidence for transfer

Pre-training can reduce the target data needed



- **Source: text**
  Target: Python code
- Lower test loss is better. Farther right means more target data.
- Across much of this plot, pre-training reaches a given loss with less target data.

Hernandez et al., 2021, Fig. 1. Same model size, Python target.

Putting it together

An adaptation experiment

- **Define the target:** Choose the data, prediction, and held-out evaluation.
- **Load a checkpoint:** Use the expected preprocessing and input format.
- **Start with a baseline:** Try fixed features or a suitable zero-shot method.
- **Compare updates:** Use validation data to compare a head, selected layers, or a small update.
- **Test new examples:** Measure the conditions in which the model must work.

The core ideas

- **Representation learning learns useful features from data.**
- Pre-training supplies a starting point for later use.
- Transfer reuses prior learning. Adaptation specifies how.
- Fine-tuning further trains the model. A linear probe trains a new head on fixed features.