Natural Language Processing

Lecture 24: Contrastive Representation Learning

Natabara Máté Gyöngyössy

Eötvös University, Department of Artificial Intelligence

2023

Acknowledgement

Acknowledgement

The following slides are based on the following review articles (Le-Khac, Healy, and Smeaton 2020; Jaiswal et al. 2020) as well as Yann LeCun's hybrid lecture on Energy-based SSL available online.

Self-supervised learning

Main objective

Self-supervised learning (SSL) aims to obtain supervision from the data itself.

"Predict everything from everything else."
Yann Lecun

The data is partially known, and partially unknown. An underlying structure of the data is utilized (e.g. sequentiality in language modeling).

Main objective

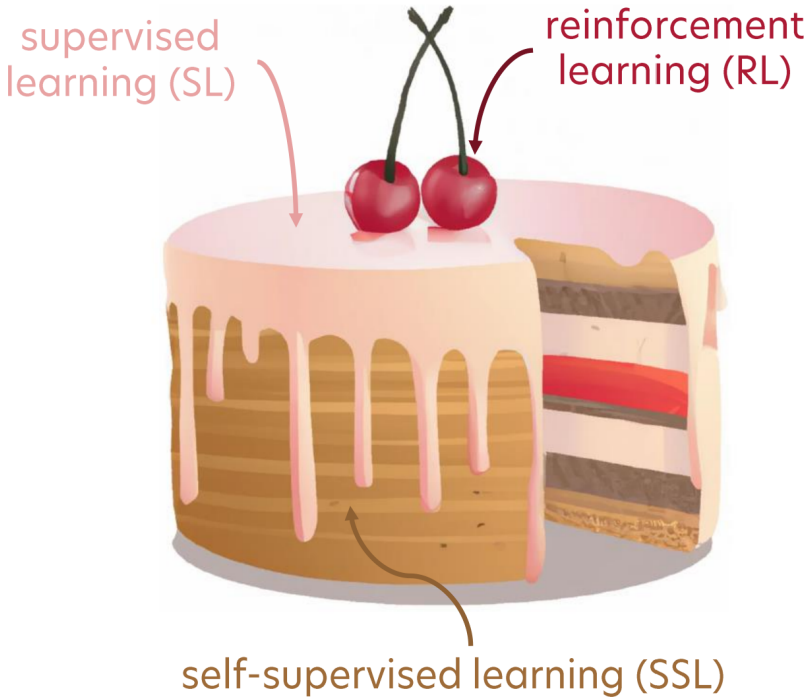


Figure 1: From (Dawid and LeCun 2023)

Why not reinforcement learning?
Trial-and-error is ineffective.

Advantages

Self-supervised learning:

- Reduces the cost and complexity of labeling
- Adds extra generalization capabilities to the system
- Gives control to use the internal structure of the data
- Is able to reconstruct latent variables governing an input set

Energy-based Modeling

Energy-based modeling (EBM) is a unifying principle of most SSL methods.

EBM solves the "averaging problem" of L_2-like losses.

- Imagine a case with multiple viable outputs (such as neighboring words in a Skipgram model)
- The loss will be minimal to the "average" of these individual outputs
- We want a loss function that will be close to minimal for each and every viable solution

Energy function

An energy function $F(x, y)$ over the $x \in X$ input space and $y \in Y$ output space is designed to solve this problem, where low energy means a viable solution.

The inference of such a model could happen by:
$\hat{y} = argmin_y F(x, y)$
It is important to note that multiple $\hat{y}$-s could be viable!

The energy function $F(x, y)$ measures compatibility between x and y.

EBM as a probabilistic model

Using the Gibbs-Boltzmann distribution a generative (joint "distribution") EBM can be converted into a discriminative probabilistic model:

$$P(y|x) = \frac{e^{-\beta F(x,y)}}{\int_{\acute{y}} e^{-\beta F(x,\acute{y})}}$$

Here β is a positive constant, and $\acute{y} \in Y$.

Multimodal EBM architectures I.

EBMs are useful for creating joint multimodal representations.

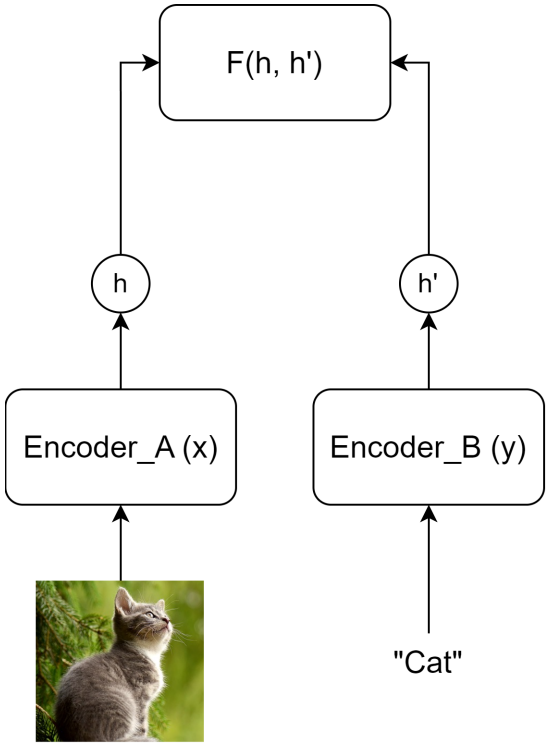


Figure 2: Joint embedding architecture

Multimodal EBM architectures II.

Latent variables could be used for generative processes (e.g. diffusion). z is an independent "explanatory" variable of variation. Inference is possible with joint minimization with respect to y and z.

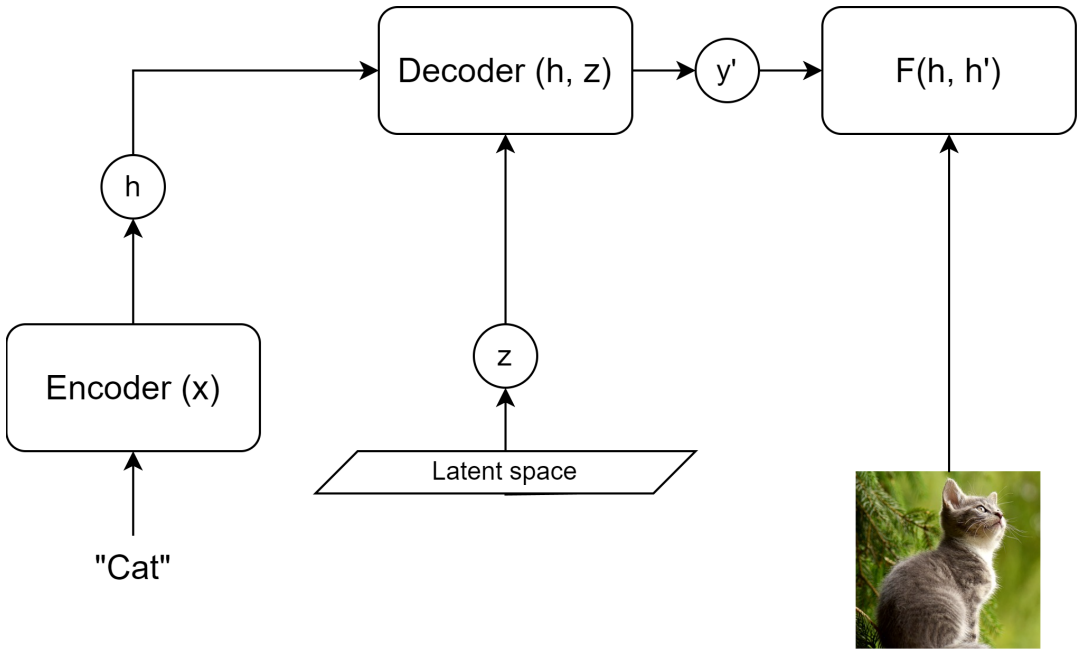


Figure 3: Latent-variable generative architecture

Methods of learning in EBMs

Main objective: Acquire low energy for viable x-y pairs, while maintaining high energy for incompatible pairs.

Contrastive Methods

- Push down $F(x, y)$ for each compatible pair (i.e. for *positive* elements of the dataset).
- Push up $F(x, y')$ for every other possible combination (i.e. for *negative* examples).

Methods of learning in EBMs

Main objective: Acquire low energy for viable x-y pairs, while maintaining high energy for incompatible pairs.

Regularized Methods

- Ensure that the extent of low-energy regions is limited or minimized.
- Regularization, quantization, clustering, etc.

Methods of learning in EBMs

Main objective: Acquire low energy for viable *x*-*y* pairs, while maintaining high energy for incompatible pairs.

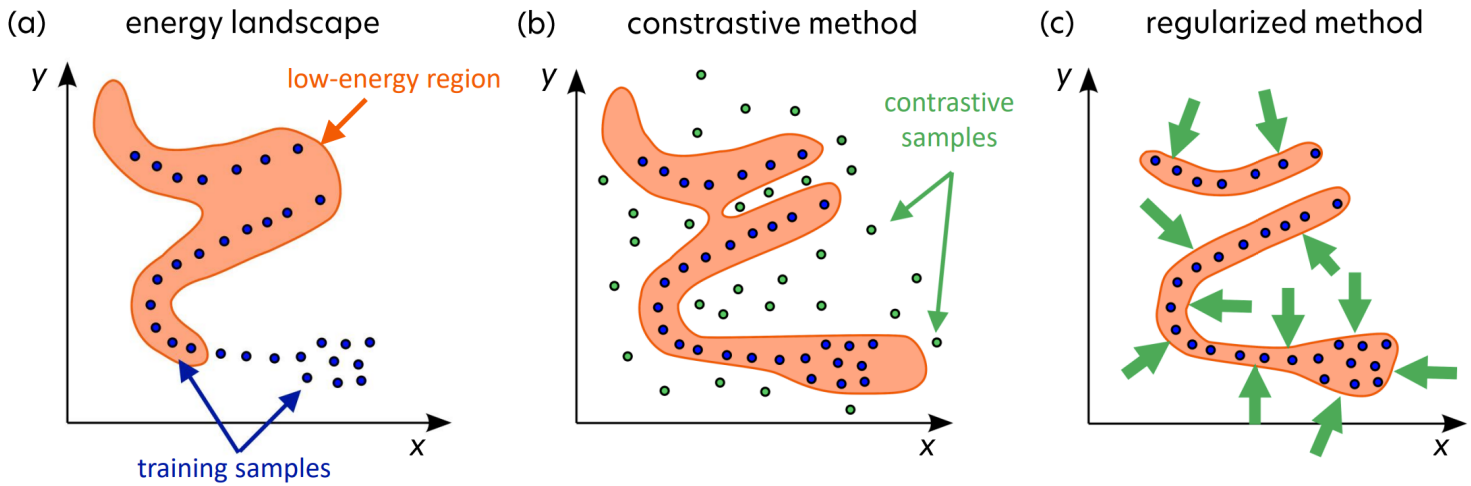


Figure 4: Visualization of learning methods from (Dawid and LeCun 2023)

Contrastive Learning & Variants

Learning method

Contrastive learning generally includes the following main steps:

1. Select a q query and sample the positive key $k^+ \sim p^+(.|q)$ and negative key $k^- \sim p^-(.|q)$ distributions.
2. Apply model transformations that map $\mathcal{X} \to \mathcal{R}^N$ where N is the resulting embedding dimension and $x \in \mathcal{X}|x = (q, k)$
3. Scoring the positive and negative pairs using an energy-based or probabilistic approach.
4. Parameter update

Scoring functions

Scoring functions are the backbone of loss calculation and are determined by the desired embedding space's properties. They are simple functions such as:

- L1 or L2 distance
- Dot-product
- Bi-linear models $S(q, k) = qAk$

Distance and probabilistic loss functions are built on top of these measures.

Distance-based loss functions

Pair-loss

$$\mathcal{L}_{pair} = \begin{cases} ||q - k^+||_2^2 \\ max(0, m - ||q - k^-||_2^2) \end{cases}$$

where m is a predefined margin around x. This minimizes positive distance and tries to push the negative distance over the margin.

Triplet-loss

$$\mathcal{L}_{triplet} = max(0, ||q - k^+||_2^2 - ||q - k^-||_2^2 + m)$$

This method enforces that the relative distance between the positive and negative examples.

Softmax-based probabilistic loss functions

Motivation: Classify the pairs correctly. As a classification problem using scoring function $S(.,.)$ we can formulate this as:

$$p(k^+|q) = \frac{exp(S(q,k^+))}{\sum_k exp(S(q,k))}$$

Introducing negative sampling to the process we can avoid calculating the denominator for all k. Instead, we reformulate the calculation as a binary problem.

Noise Contrastive Estimation (NCE)

The probability of a pair being positive (C=1), if we sample negative examples M times more frequently from a uniform distribution, is:

$$p(C = 1|q, k) = \frac{p(k^+|q)}{p(k^+|q)+m \cdot p(k^-|q)}$$

Thus the binary classification loss is (using negative loglikelihoods) over all possible pairs:

$$\mathcal{L}_{bin\_NCE} = -\mathbb{E}_{p^+}[logp(C = 1|q, k)] \\ -\mathbb{E}_{p^-}[log(1 - p(C = 1|q, k))]$$

where $p^-(.|q)$ is the noise (negative sample) distribution and $p^+(.,.)$ is the positive distribution.

InfoNCE

Instead of a binary classification, we could construct a set of several negative examples and a single positive example $K = \{k^+, k_1^-, k_2^-, ..., k_M^-\}$. Then the modified task would be to determine which element is the positive. This results in a softmax-like measure called InfoNCE:

$$\mathcal{L}_{InfoNCE} = -log \frac{exp(S(q,k^+))}{\sum_{i=0}^{M+1} exp(S(q,k[i]))}$$

$$\mathcal{L}_{InfoNCE} = -S(q,k^+) + log \sum_{i=0}^{M+1} e^{S(q,k[i])}$$

Why does it work?

Training a model f with an InfoNCE-like loss function inverts (decodes) the unknown generative process of data generation g. Thus the latent distribution behind our data is reconstructed and made accessible.

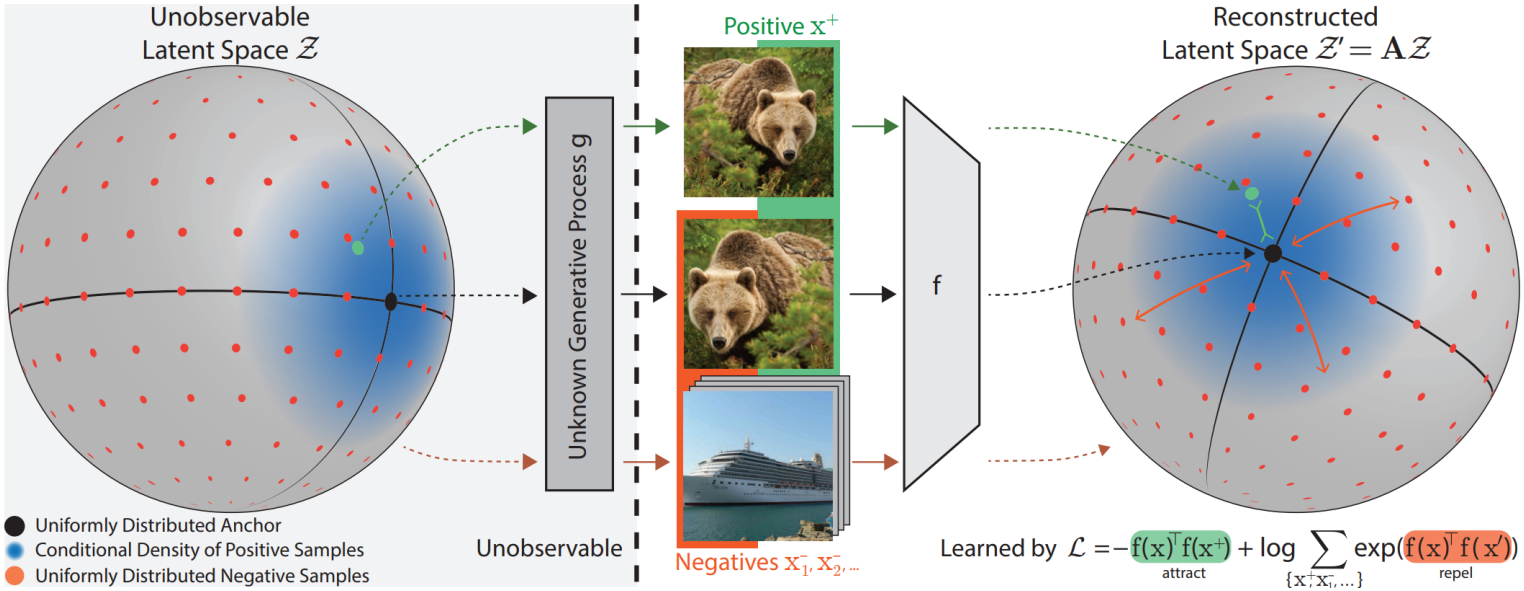


Figure 5: From (Zimmermann et al. 2022)

Examples of sampling

Data generation processes could include a wide range of self-supervised processes, such as:

- Neighborhood information (spatial or temporal)
- Masking
- Various augmentations (visual or audio noise, etc)

Examples of sampling

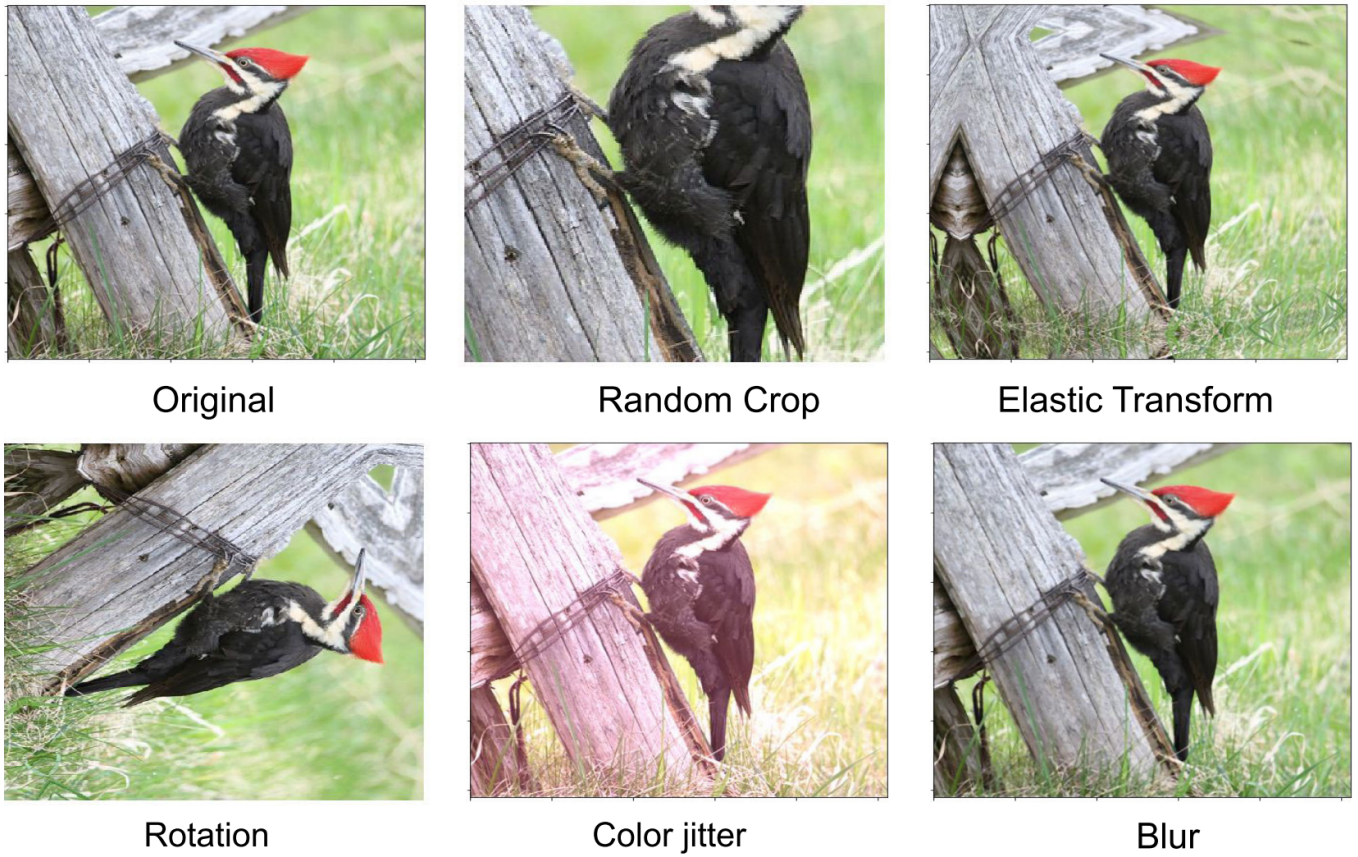


Figure 6: Visual augmentations from (Le-Khac, Healy, and Smeaton 2020)

Examples of sampling

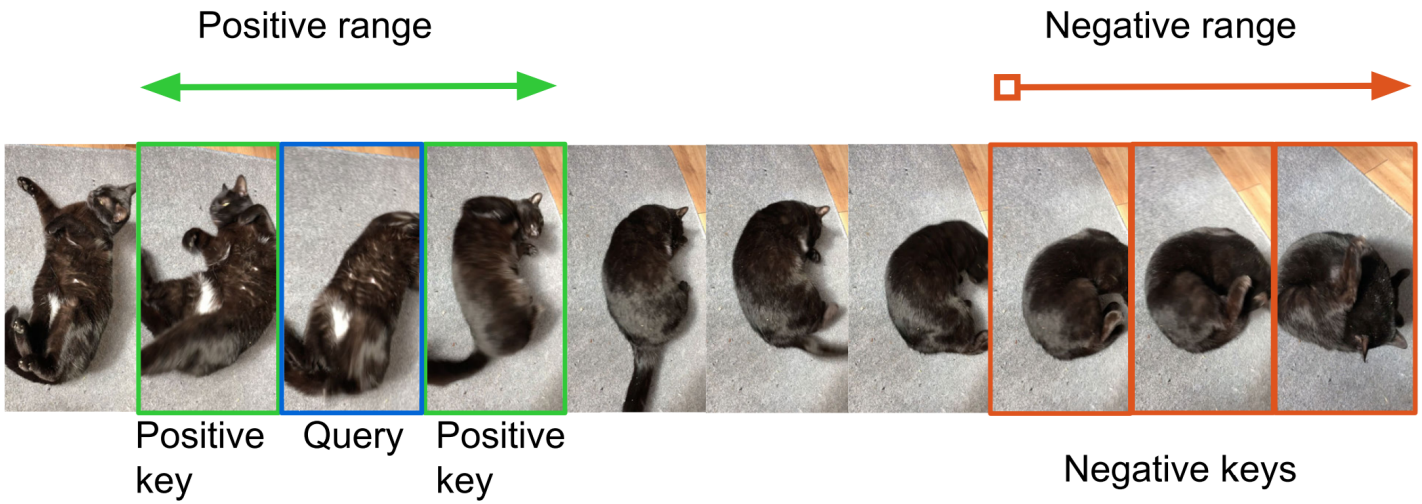


Figure 7: Data generation from temporal streams from (Le-Khac, Healy, and Smeaton 2020)

Adding label supervision

Data generation is possible via incorporating label information as well (adding classical supervision). In this case the normal InfoNCE equation will change, as multiple positive examples are present. Resulting in a sum over InfoNCE terms. There are two variants present with the sum inside and outside of the log.

$$\mathcal{L}_{in}^{sup} = \sum_{q \in J} -log \left(\frac{1}{|P(q)|} \sum_{k^p \in P(q)} \frac{exp(S(q,k^p))}{\sum_{i \in I} exp(S(q,k[i]))} \right)$$

where J is the set of batch elements, q is the selected query element, I is the set of batch elements excluding q, $P(q)$ is the set of elements with the same label as q.

Adding label supervision

$$\mathcal{L}_{out}^{sup} = \sum_{q \in J} \frac{-1}{|P(q)|} log \sum_{k^p \in P(q)} \frac{exp(S(q,k^p))}{\sum_{i \in I} exp(S(q,k[i]))}$$

where J is the set of batch elements, q is the selected query element, I is the set of batch elements excluding q, $P(q)$ is the set of elements with the same label as q.

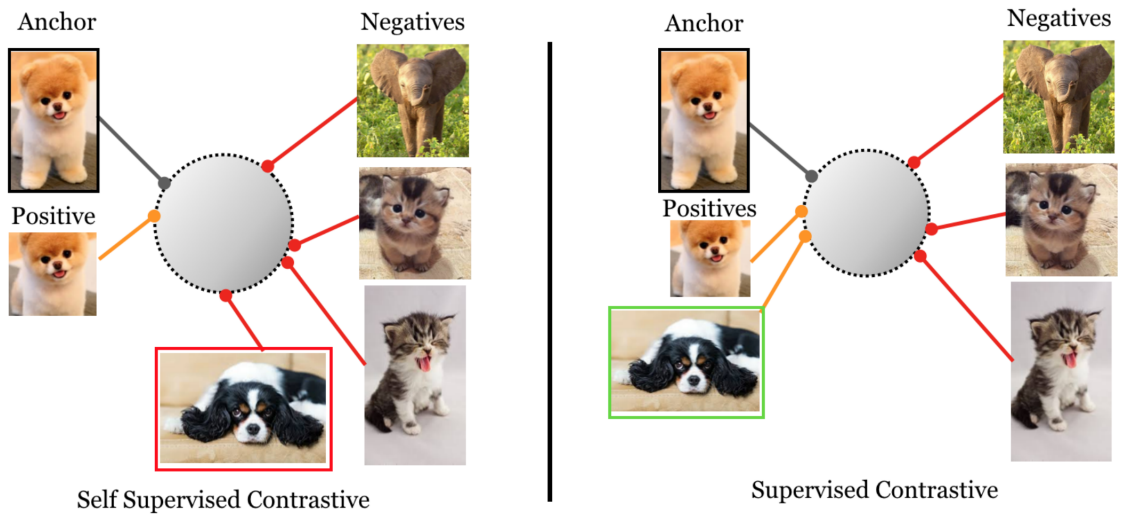


Figure 8: From (Khosla et al. 2020)

Invariant, Equivariant traits

In standard contrastive learning, the positive pairs have a required invariancy. $S(q, k)$ should be high. Standard similarity metrics yield this behavior best when $q = k$. This behavior will negate the effect of certain differences between the two original inputs x_q and x_k

Let $T(.)$ transform represent this difference and $f(.)$ represent our function (or network) trained with CL. In the invariant optimal case:

$$x_k = T(x_q) \rightarrow k = q$$

Invariant, Equivariant traits

There are some cases where we would like to keep this transformation in the embedding space as well. Meaning that we would require that the same, or a similar transformation ($\acute{T}(.)$) be present in the embedding space as in the input space.

$$x_k = T(x_q) \rightarrow k = \acute{T}(q)$$

Invariant, Equivariant traits

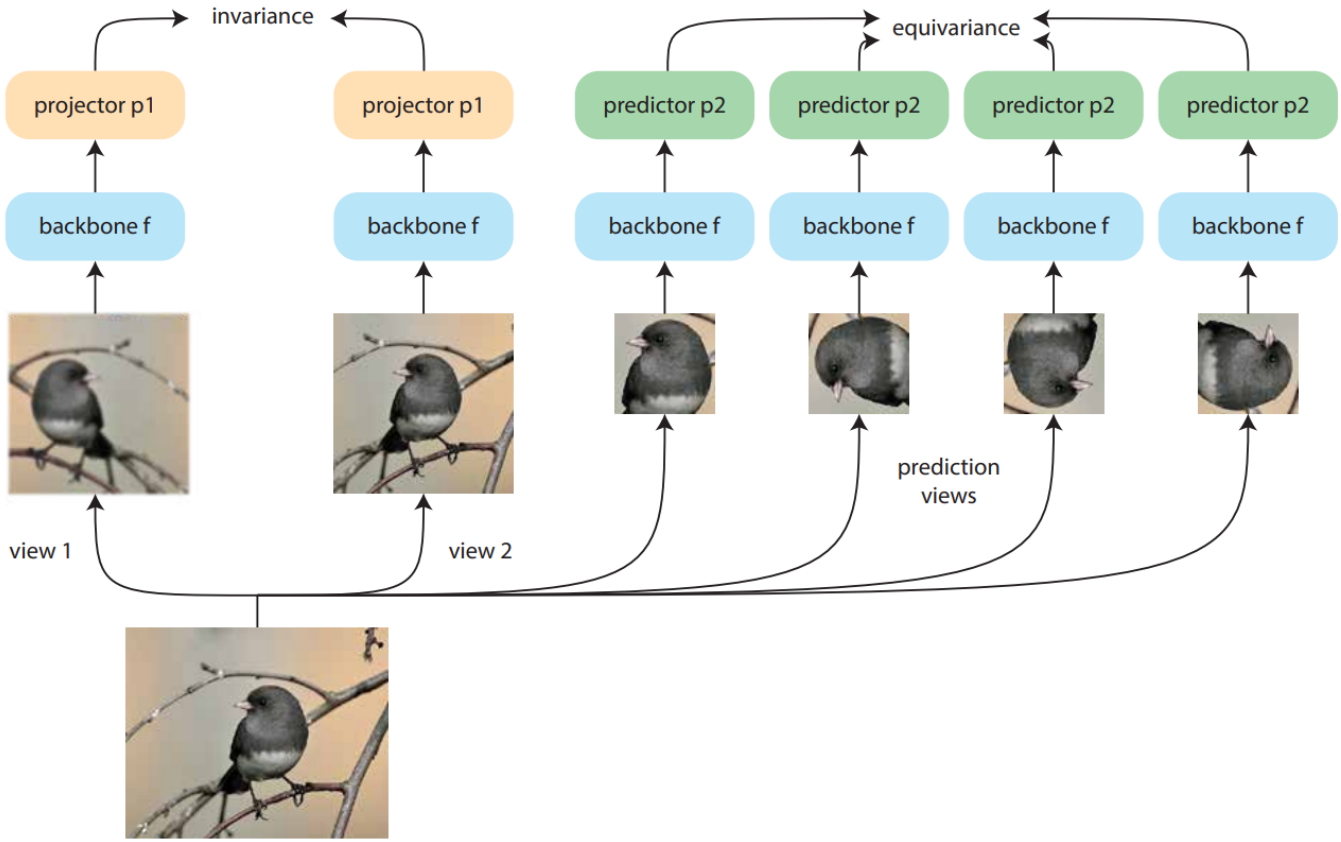
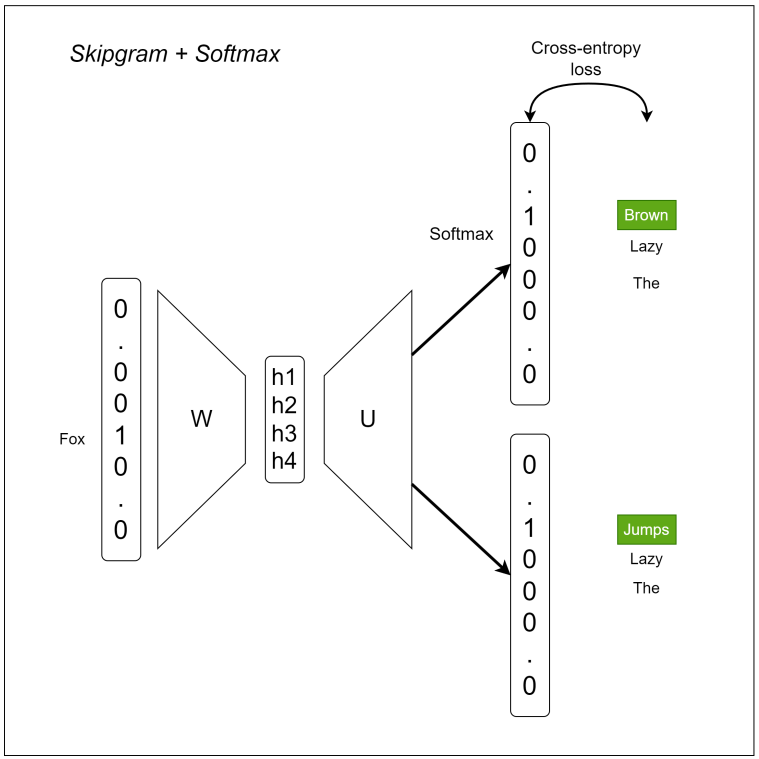


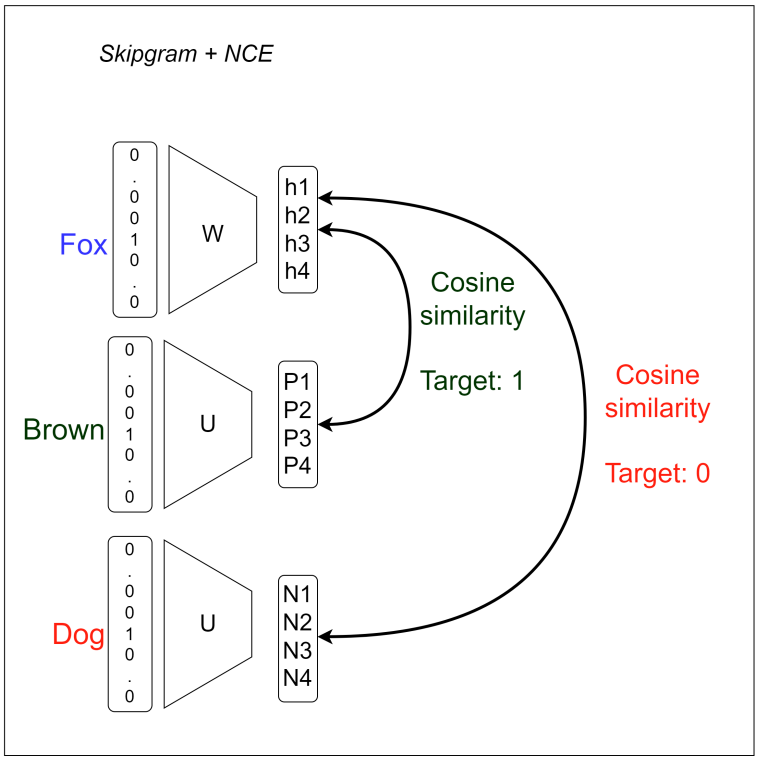
Figure 9: Rotation equivariant and flip invariant contrastive training. From (Dangovski et al. 2021)

Contrastive methods in NLP

Word2Vec as Contrastive Learning

The quick **brown fox jumps** over the lazy dog.





Word2Vec as Contrastive Learning

Reformulating skipgram, to a multi-encoder joint embedding-type self-supervised problem.

Instead of Softmax we use the Noise Contrastive Estimation loss (SGNS).

Positive pairs maximize similarity (minimize energy according to EBM modeling).

Negative pairs minimize similarity (maximize energy according to EBM modeling).

BERT Next Sentence Prediction

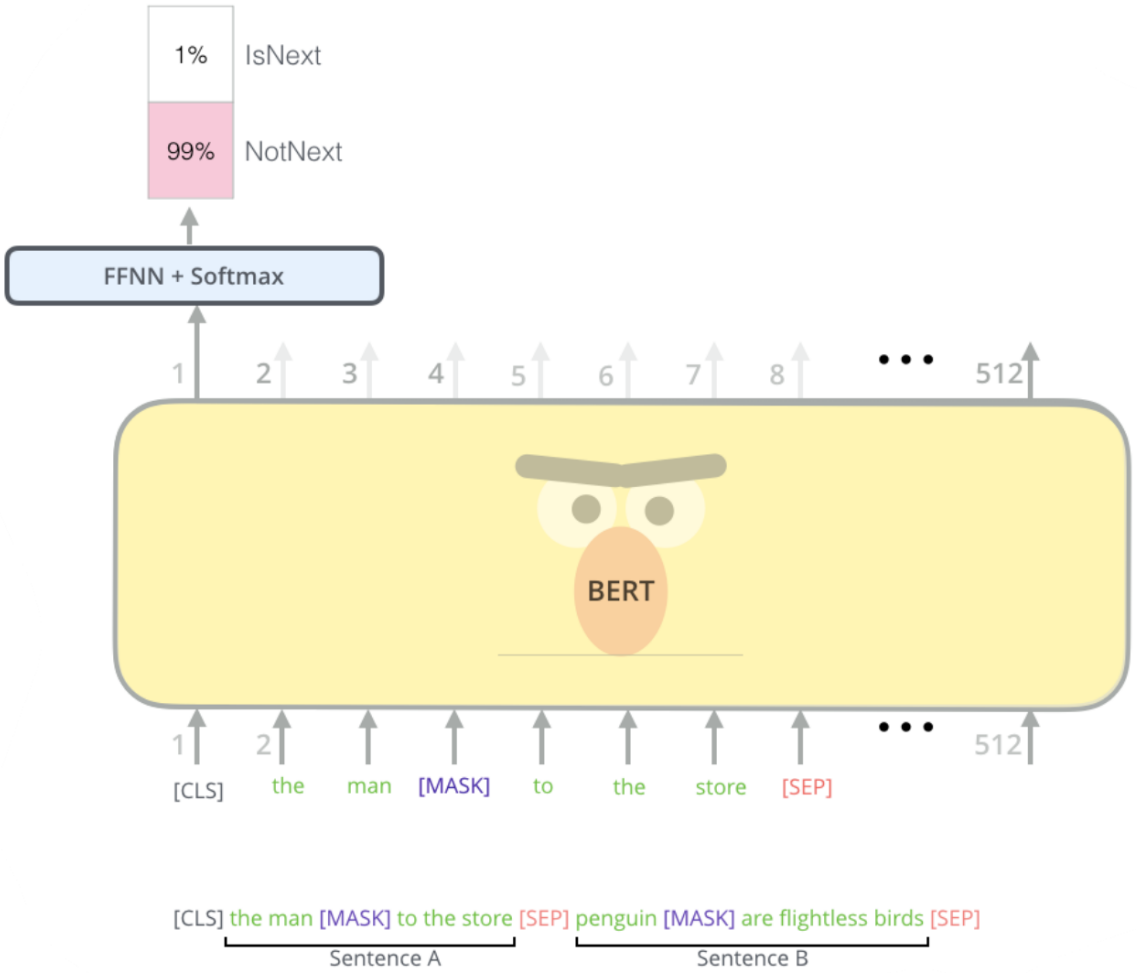
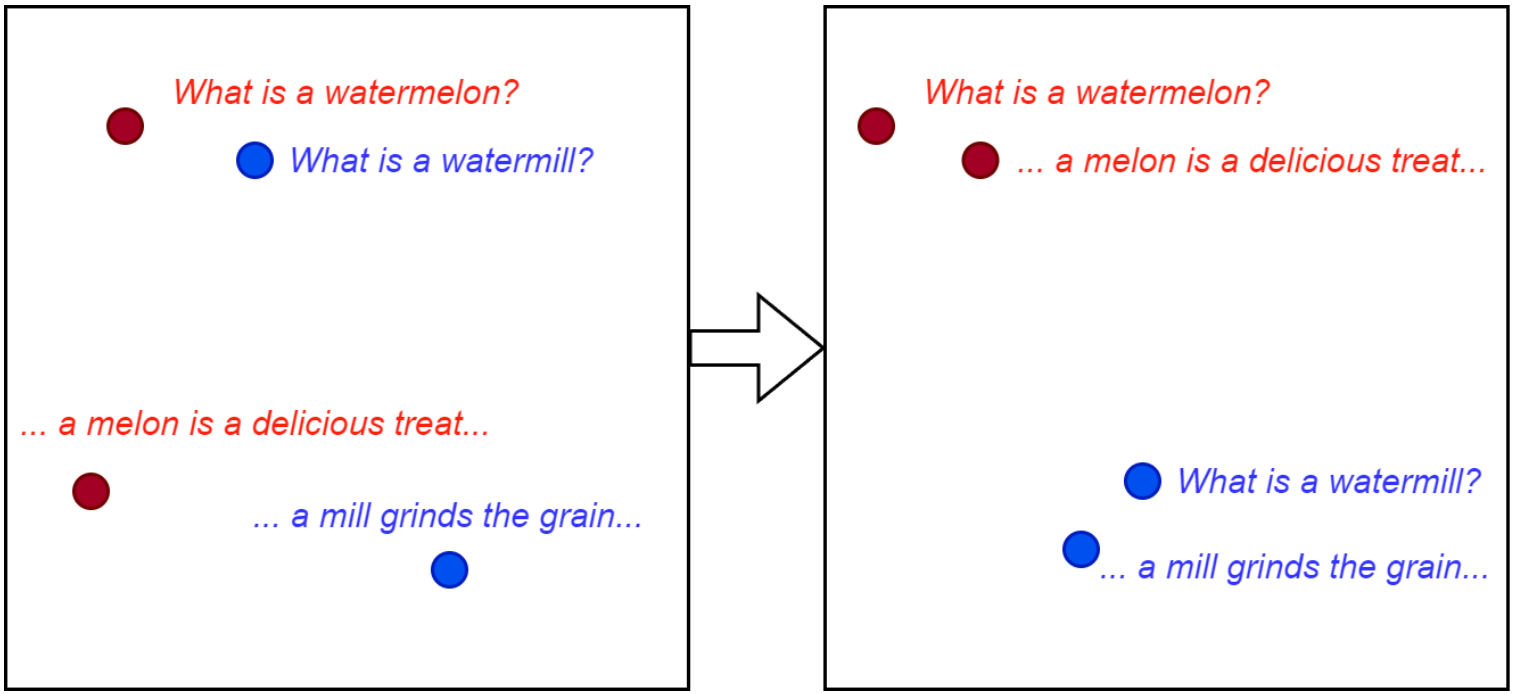


Figure 10: From: Alammar, J (2018). The Illustrated Transformer

Text-embedding models

Pre-trained and fine-tuned LMs could be used to produce semantic embeddings of text.

- This is good in terms of general language semantics only



Text-embedding models

Contrastive fine-tuning on additional SSL tasks comes in handy in the case of domain-dependent embeddings or multi-task embedders. Such tasks could include (Su et al. 2022):

- Retrieval, reranking (find/rank documents based on query)
- Clustering (creating clusters in the embedding space)
- Text classification
- Summarization
- Deduplication

Contrastive Multimodal Methods

CLIP

Contrastive Language-Image Pre-training (Radford et al. 2021)

Problem: Visual classifiers are bound to a finite set of supervised labels.

Solution: Use natural language to describe visual features and try to achieve zero/few-shot learning.

Data: (image, text) pairs from web crawls (even filenames), including Instagram, Wikipedia-based Image Text, YFCC100M and MS-COCO. Open-source large-scale datasets include Laion5B (Schuhmann et al. 2022).

CLIP Structure

Image embedding (E_I) ResNet or **ViT** $[n \times d_I]$

Text embedding (E_T) Transformer LM $[n \times d_T]$

Linear projections (W_I, W_T) $[d_I \times d_E]$, $[d_T \times d_E]$

t temperature parameter for classification

L labels of similarity (usually one-hot) $[n,]$

$CE_{col|row}$ cross-entropy loss by columns (text) or rows (image) of the first argument.

$$S_{scaled} = ||E_I \cdot W_I||_{L2} \cdot ||E_T \cdot W_T||_{L2}^T \cdot exp(t)$$
$[n \times n]$

$$loss = 0.5 CE_{col}(S_{scaled}, L) + 0.5 CE_{row}(S_{scaled}, L)$$

CLIP Encoder details

- Modified global pooling: attentional pooling (Lee et al. 2019)
  Cross-attention where the image features are K, V and Q is defined by a learned constant vector (or a set of vectors).
- ViT (Vision Transformer): Transformer that uses small patches (rectangular parts) of the image as tokens. (Covered in upcoming lectures.)
- The text encoder is a GPT-2 style model.

CLIP Training

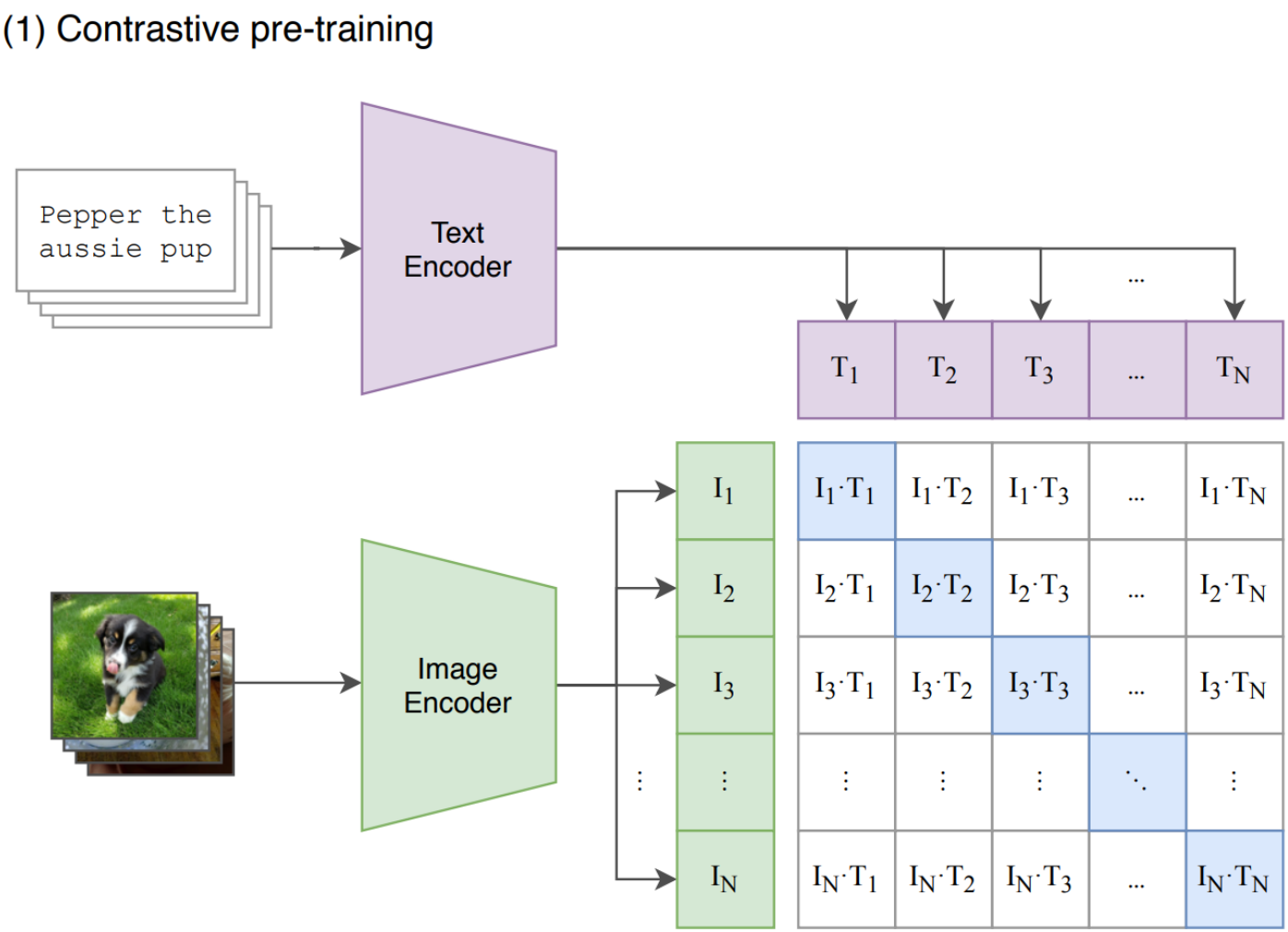


Figure 11: CLIP training by (Radford et al. 2021)

CLIP Zero-shot inference

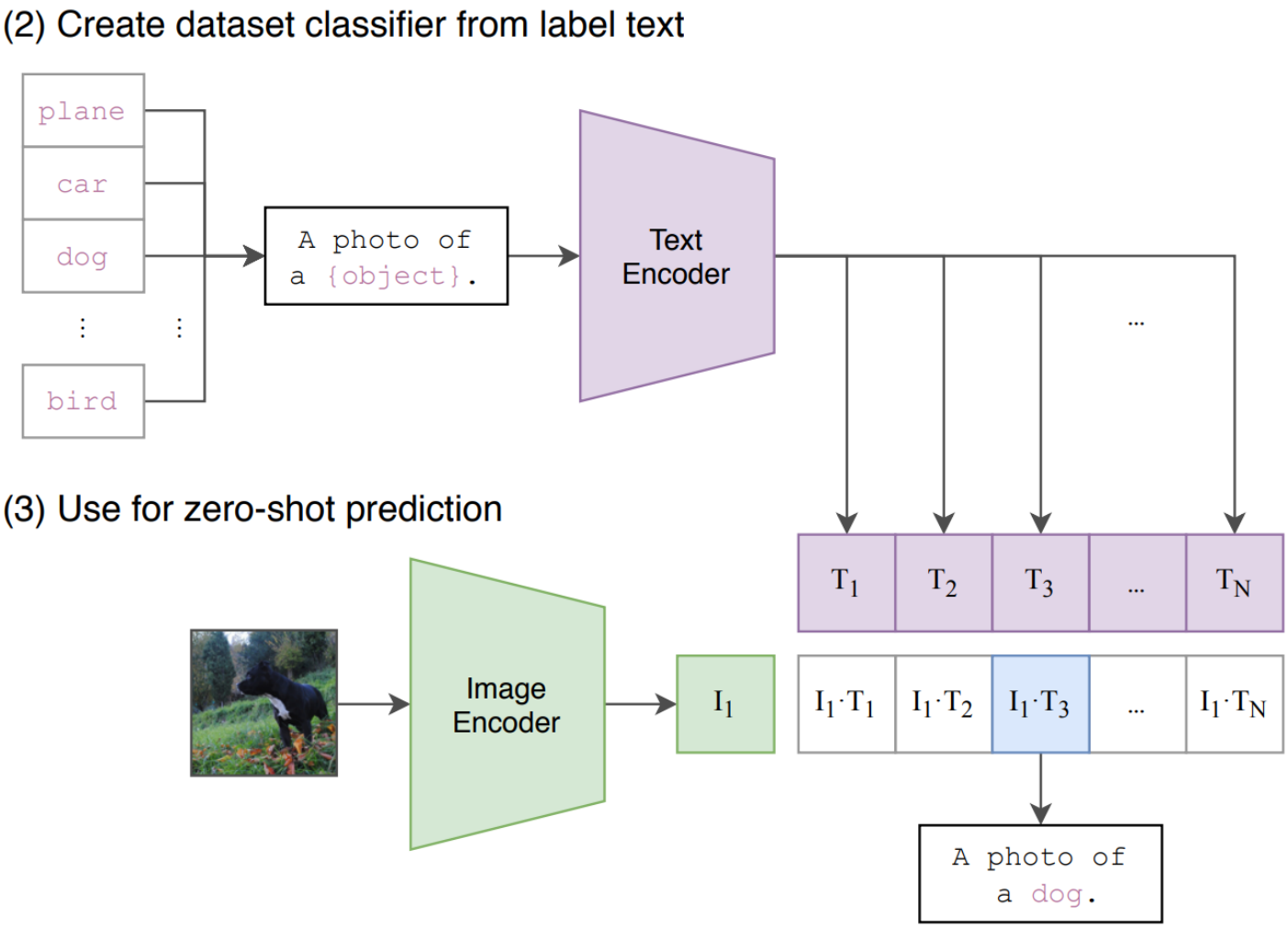


Figure 12: CLIP inference by (Radford et al. 2021)

CLIP Zero-shot inference

CLIP can classify images based on a corresponding text definition of classes.

Selection is done by finding the most similar class definition.

Other use-cases include:

- Base-model for custom classifiers
- Base-model for transfer-learning (outperforms previous ImageNet models)
- Image retrieval (search-engine)
- Condition vectors for image generation
- Multi-modal semantics

ImageBind

CLIP demonstrated that additional generalization capabilities can originate from incorporating multiple modalities in one representation space. ImageBind (Girdhar et al. 2023) takes it one step further and joins 7 modalities in one embedding space.

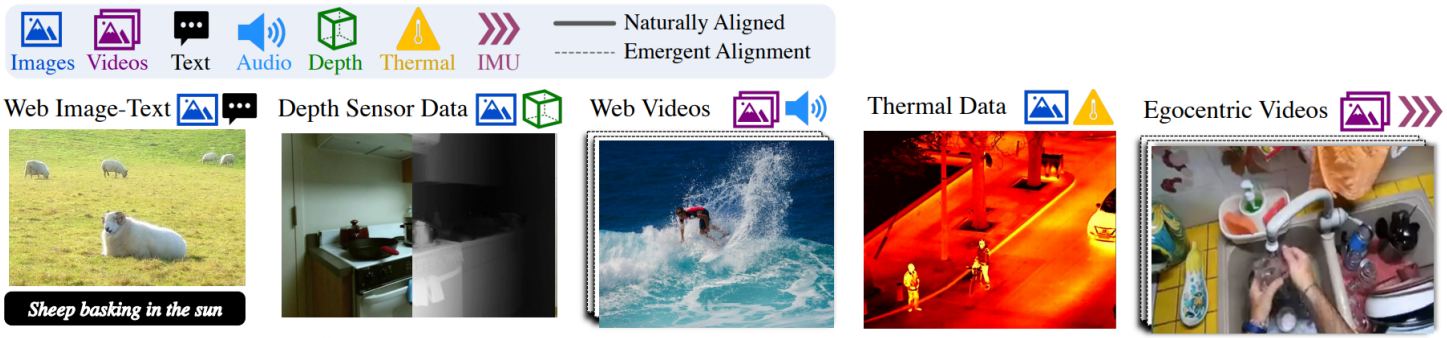


Figure 13: Modalities and data sources of ImageBind (Girdhar et al. 2023)

Emergent Alignment

Using InfoNCE again we can construct alignments of $(\mathcal{I}, \mathcal{M}_1)$ and $(\mathcal{I}, \mathcal{M}_2)$. It is observed that this alignment is transitive and results in a partial $(\mathcal{M}_1, \mathcal{M}_2)$ alignment. Encoders are now initialized from pre-trained models (e.g.: CLIP)

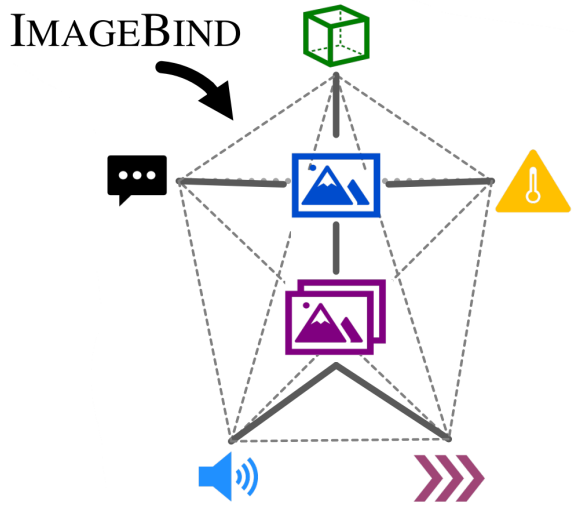


Figure 14: Natural and emergent alignment in ImageBind (Girdhar et al. 2023)

ImageBind Results

Multimodal contrastive embeddings outperform supervised modality converters in the absence of naturally present multimodal signals (e.g.: text-to-audio).

ImageBind use-case examples include: - Cross-modal retrieval - Embedding-space arithmetics - Cross-modal decoder re-utilization

Cross-modal retrieval

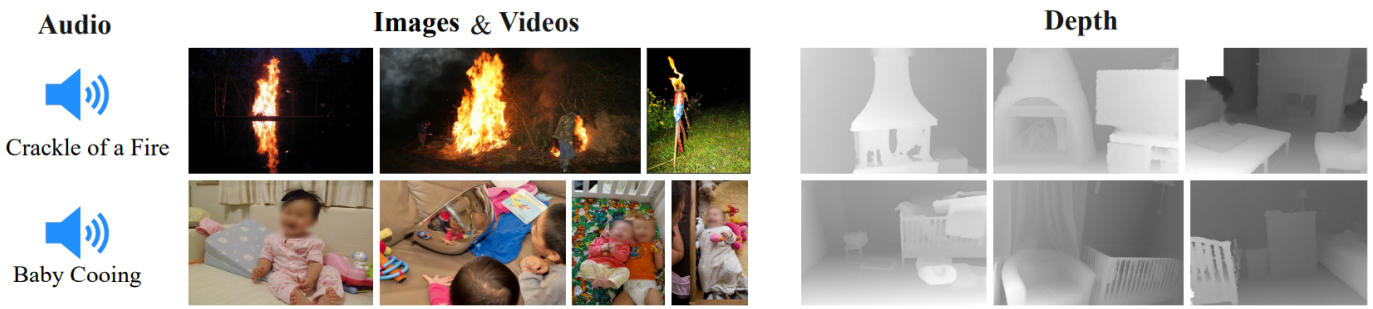


Figure 15: ImageBind retrievals of non-trivial modality pairs (Girdhar et al. 2023)

Cross-modal retrieval

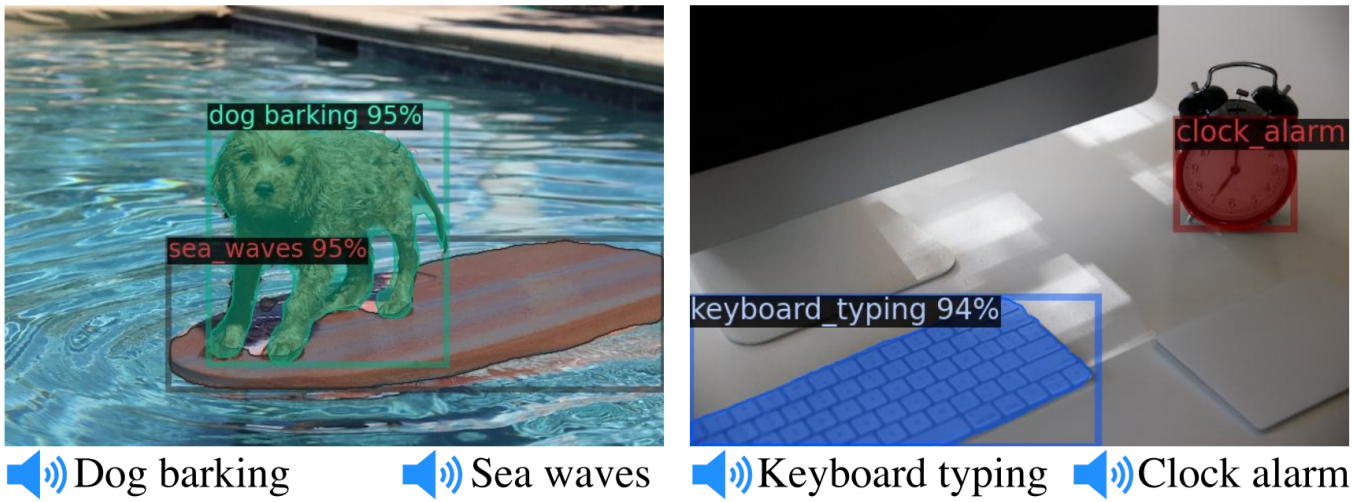


Figure 16: ImageBind retrievals of non-trivial modality pairs (with object detection in the visual modality) (Girdhar et al. 2023)

Cross-modal retrieval

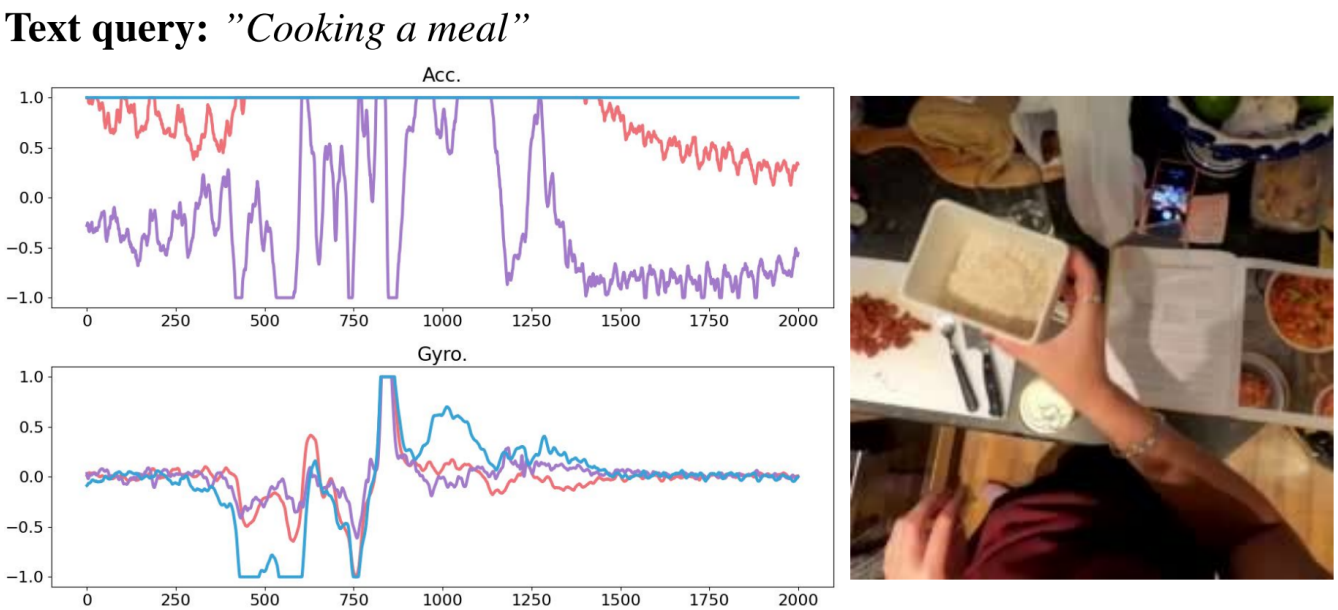


Figure 17: ImageBind retrievals of non-trivial modality pairs (Girdhar et al. 2023)

Embedding-space Arithmetics

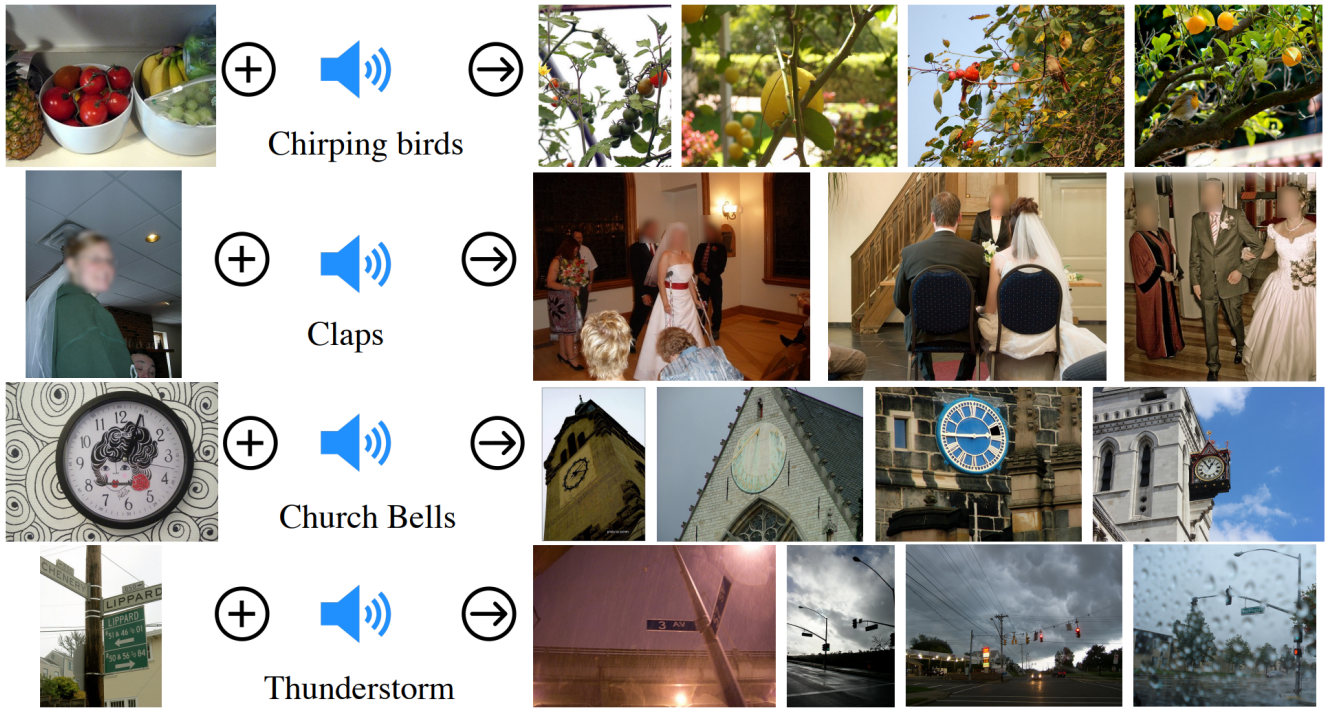


Figure 18: ImageBind multi-modal embedding arithmetics (Girdhar et al. 2023)

Cross-modal decoder re-utilization

Figure 19: ImageBind re-utilizing text-to-image decoder as audio-to-image using the text-to-audio alignment (Girdhar et al. 2023)

Decoding Methods

How to invert a joint embedding?

- Iterative method
- Prefix decoder
- Zero-shot decoder
- Contrastive Captioners (CoCa)
- *Diffusion processes (detailed later in upcoming lectures)*

Our examples focus on the visual-language modality pair (mainly captioning), but these methods are adaptable for other pairs as well.

Iterative decoder

Simplest solution, no training involved.

The method relies on a language model. During generation intermediate text outputs are iteratively encoded to the joint CLIP space, where the ones with the best similarities to the encoded image representation are selected. New candidate captions (or continuations) are then generated based on these.

Problems:

- Inaccurate (no proper guiding)
- Inefficient (scales with vocabulary size / caption length)

Prefix decoders

Prefix-decoders use classical seq2seq decoding methods. By joining CLIP and a LM (typically GPT) the data needed for such a captioner decreases.

A small mapping network is enough to make the CLIP image embedding space and the LM compatible. Fine-tuning the LM as well usually results in a slight performance increase.

Let's imagine that the mapper is a small MLP or Transformer generating $[p_1^i, ..., p_k^i] = MAP(CLIP(x^i))$ prefix from input image x^i.

Mapping in Prefix decoders

Why do we need mapping?

- Contrastive loss does not ensure the exact match of positive text-image pair embeddings.
- Domain-dependent captioning could need a slightly different alignment/structure in the embedding space.

Training of Prefix decoders

The model is finetuned on captioned images. Using the following loss function:

$$L = -\sum_{i=1}^{N} \sum_{j=1}^{M} log p_\theta(c_j^i | p_1^i, ..., p_k^i, c_1^i, ..., c_{j-1}^i)$$

Where $c_1^i, ..., c_{j-1}^i$ are the previous caption tokens, and θ represents the trainable params.

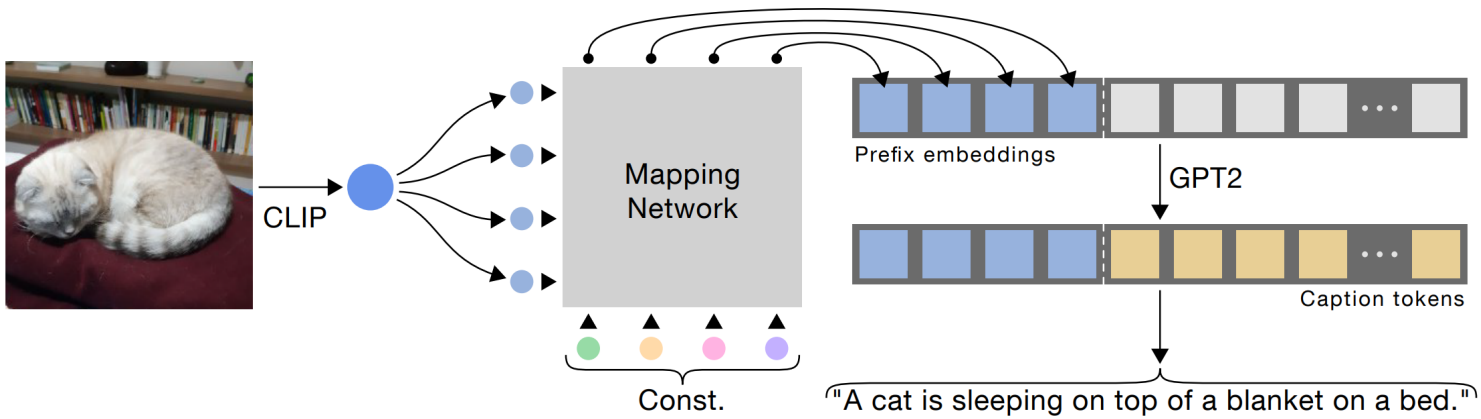


Figure 20: ClipCap architecture with frozen CLIP and GPT. (Mokady, Hertz, and Bermano 2021)

Zero-shot decoders

While prefix decoders are effective and have acceptable performance, they still need domain-dependent (image, caption) training data.

Most popular solutions use text-only prefix-finetuned decoders with different tricks to replace CLIP space mapping:

- Non-trained projection based on previously encoded text embeddings (Li et al. 2023)
- Noise injection to train a robust decoder (Nukrai, Mokady, and Globerson 2022)

DeCap

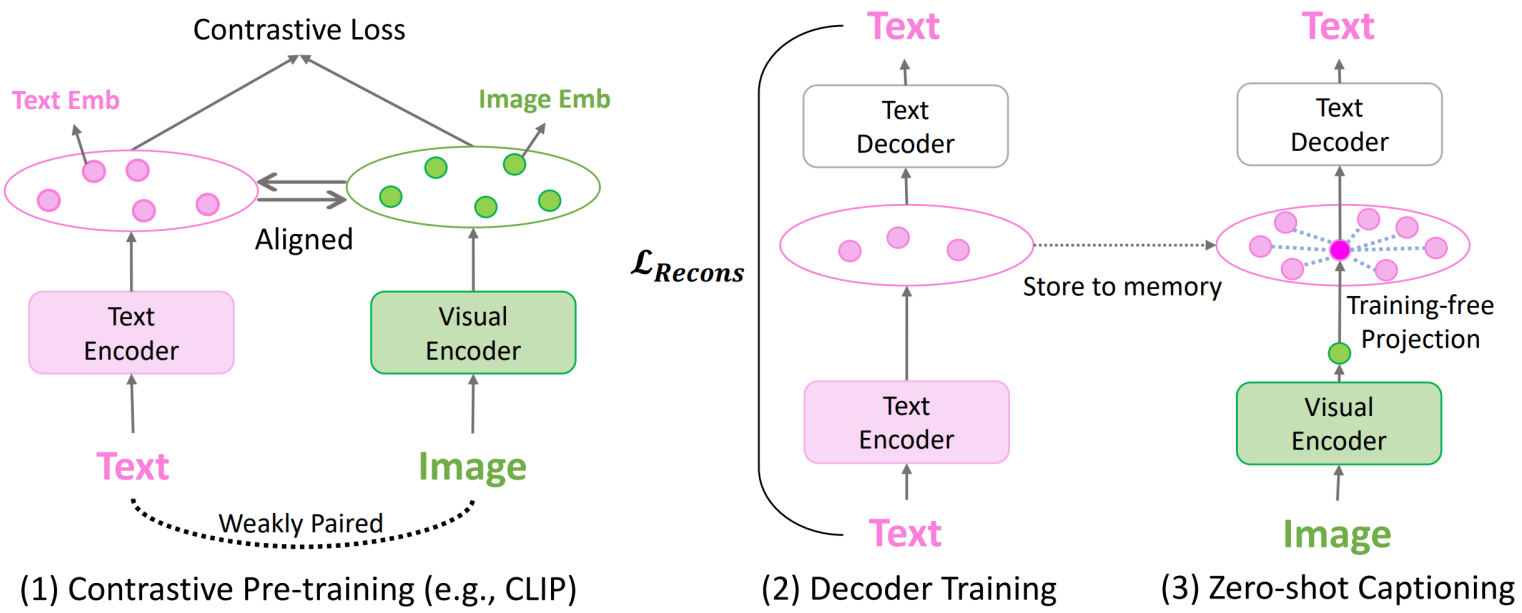


Figure 21: DeCap with a text-only finetuned decoder (reconstruction loss) and training-free projection (Li et al. 2023)

CapDec

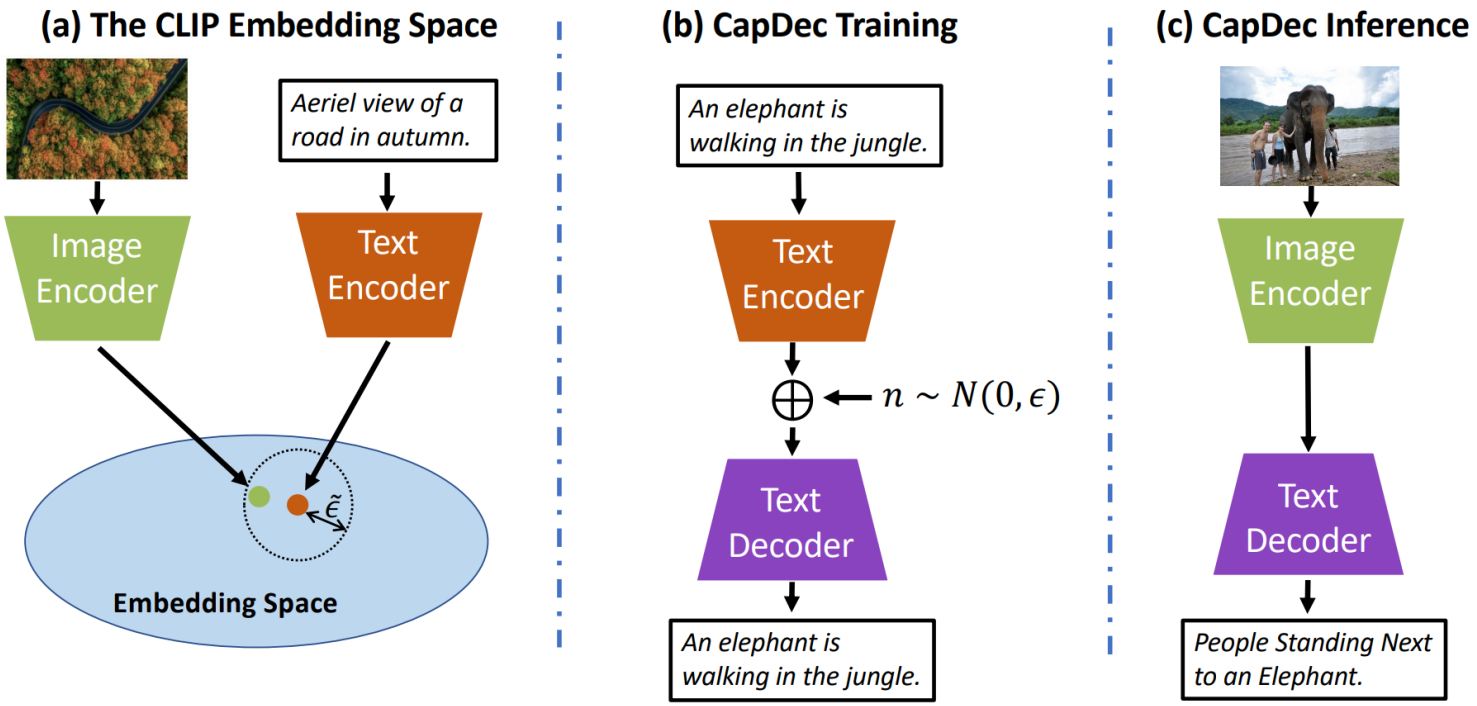


Figure 22: CapDec with a noise-robust decoder (step b) is similar to a denoising VAE) (Nukrai, Mokady, and Globerson 2022)

Contrastive Captioners (CoCa)

Performance and efficiency concerns related prefix decoders:

- Do we need a prefix when we have cross-attention?
- Why not design the original model with decoding capabilities by training a decoder parallel to the contrastive training phase?
- Encoders should be transfer-learned.

CoCa Architecture

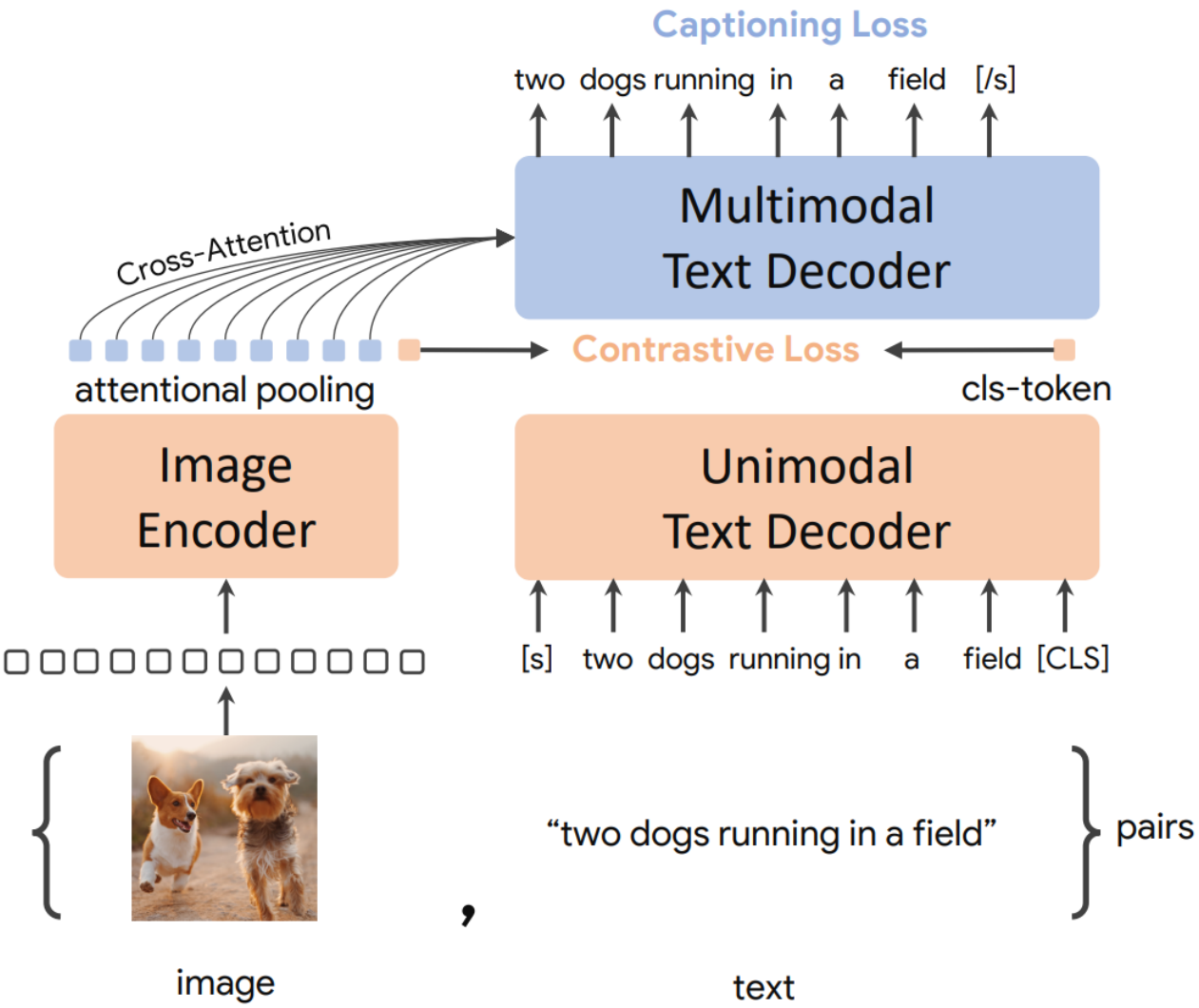


Figure 23: From (Yu et al. 2022)

CoCa Training

1. Initialize models from single-modality pre-trained models
2. Change vision heads (different attentive pooling for captioning and contrastive learning)
3. Split the text omitting cross-attention from the first half
4. Perform simultaneous contrastive and reconstruction (captioning) training. Image-only datasets could also be used in the reconstruction task if the vocabulary is exactly the set of possible classes.

CoCa Inference

Contrastive Captioner models can be used with further fine-tuning or in a zero-shot manner as any combination of its building blocks.

CoCa-s are not limited to the visual-language modalities. [CoCa use cases from Yu et al. (2022)

Summary

Summary

Self-supervised learning (SSL) is a strong and cost-efficient training method that can capture the underlying latent distribution of a given dataset. A widespread neural formulation is via Contrastive Learning (defined by InfoNCE-like losses).

Contrastive methods produce joint embeddings of multiple modalities, which create powerful semantic representations by cross-modality alignment. These methods are useful for retrieval and zero-shot classification tasks. Decoders (e.g.: captioners) can also be constructed to perform inverse tasks.

References

References I

Dangovski, Rumen, Li Jing, Charlotte Loh, Seungwook Han, Akash Srivastava, Brian Cheung, Pulkit Agrawal, and Marin Soljačić. 2021. "Equivariant Contrastive Learning." *arXiv Preprint arXiv:2111.00899*.

Dawid, Anna, and Yann LeCun. 2023. "Introduction to Latent Variable Energy-Based Models: A Path Towards Autonomous Machine Intelligence." https://arxiv.org/abs/2306.02572.

Girdhar, Rohit, Alaaeldin El-Nouby, Zhuang Liu, Mannat Singh, Kalyan Vasudev Alwala, Armand Joulin, and Ishan Misra. 2023. "Imagebind: One Embedding Space to Bind Them All." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 15180–90.

Jaiswal, Ashish, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. 2020. "A Survey on Contrastive Self-Supervised Learning." *Technologies* 9 (1): 2.

References II

Khosla, Prannay, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. "Supervised Contrastive Learning." *Advances in Neural Information Processing Systems* 33: 18661–73.

Lee, Juho, Yoonho Lee, Jungtaek Kim, Adam R. Kosiorek, Seungjin Choi, and Yee Whye Teh. 2019. "Set Transformer: A Framework for Attention-Based Permutation-Invariant Neural Networks." https://arxiv.org/abs/1810.00825.

Le-Khac, Phuc H, Graham Healy, and Alan F Smeaton. 2020. "Contrastive Representation Learning: A Framework and Review." *Ieee Access* 8: 193907–34.

Li, Wei, Linchao Zhu, Longyin Wen, and Yi Yang. 2023. "DeCap: Decoding CLIP Latents for Zero-Shot Captioning via Text-Only Training." *arXiv Preprint arXiv:2303.03032*.

Mokady, Ron, Amir Hertz, and Amit H. Bermano. 2021. "ClipCap: CLIP Prefix for Image Captioning." https://arxiv.org/abs/2111.09734.

References III

Nukrai, David, Ron Mokady, and Amir Globerson. 2022. "Text-Only Training for Image Captioning Using Noise-Injected Clip." *arXiv Preprint arXiv:2211.00575*.

Radford, Alec, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, et al. 2021. "Learning Transferable Visual Models from Natural Language Supervision." https://arxiv.org/abs/2103.00020.

Schuhmann, Christoph, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, et al. 2022. "LAION-5B: An Open Large-Scale Dataset for Training Next Generation Image-Text Models." https://arxiv.org/abs/2210.08402.

Su, Hongjin, Jungo Kasai, Yizhong Wang, Yushi Hu, Mari Ostendorf, Wen-tau Yih, Noah A Smith, Luke Zettlemoyer, Tao Yu, et al. 2022. "One Embedder, Any Task: Instruction-Finetuned Text Embeddings." *arXiv Preprint arXiv:2212.09741*.

References IV

Yu, Jiahui, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. 2022. "Coca: Contrastive Captioners Are Image-Text Foundation Models." *arXiv Preprint arXiv:2205.01917*.

Zimmermann, Roland S., Yash Sharma, Steffen Schneider, Matthias Bethge, and Wieland Brendel. 2022. "Contrastive Learning Inverts the Data Generating Process." https://arxiv.org/abs/2102.08850.