IEEE - ISBI 2023
International Symposium on Biomedical Imaging
Cartagena de Indias • Colombia April 18-21 2023

Vision-Language Multi-modal Learning for Biomedical Images Part I

Joonseok Lee
Graduate School of Data Science, Seoul National University
Google Research

Apr 18, 2023

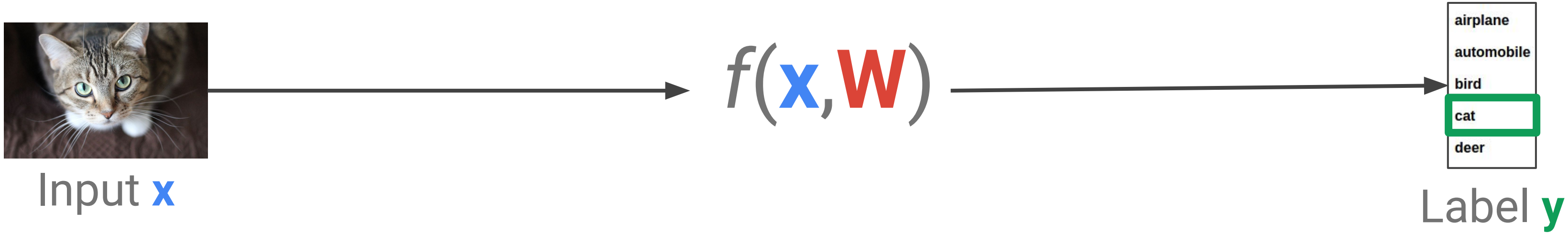
Today's Contents

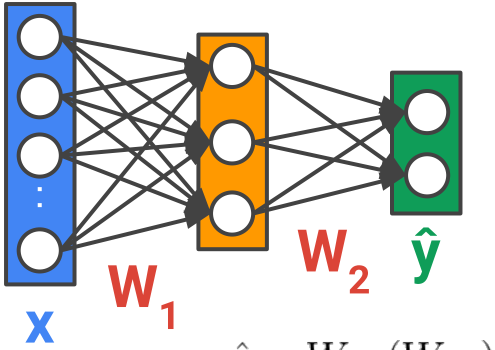
1. Transformers
2. BERT
3. Vision Transformers
   - ViT
   - ViViT
4. Transformer-based Image-Text Models
   - VL-BERT
   - VilBERT
5. Transformer-based Video-Text Models
   - VideoBERT
   - CBT
   - Hammer
6. Large-scale Multimodal Pre-training
   - CLIP
   - MuLan

Transformers

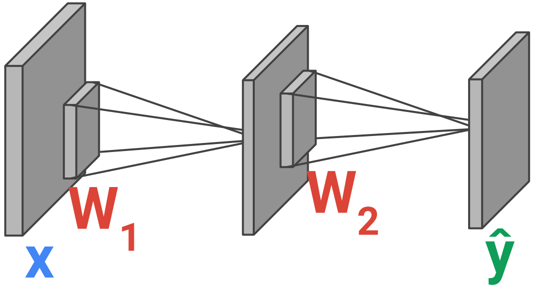
Review: MLPs and CNNs

- What MLPs and CNNs learn:
  - A set of weights that best map from inputs **x** to the labels **y** in the training dataset.
  - Roughly speaking, the output **ŷ** is a weighted sum (+fixed unary operations) of input **x**!





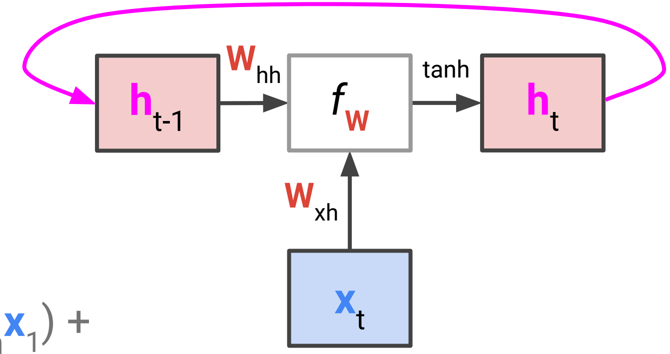
$$\hat{y} = W_2\sigma(W_1\mathbf{x})$$



Review: RNNs

- Then, what about RNNs?

$$\mathbf{h}_t = f_{\mathbf{W}}(\mathbf{h}_{t-1}, \mathbf{x}_t) = \tanh(\mathbf{W}_{hh}\mathbf{h}_{t-1} + \mathbf{W}_{xh}\mathbf{x}_t)$$



$$\mathbf{h}_1 = \tanh(\mathbf{W}_{hh}\mathbf{h}_0 + \mathbf{W}_{xh}\mathbf{x}_1)$$

$$\mathbf{h}_2 = \tanh(\mathbf{W}_{hh}\mathbf{h}_1 + \mathbf{W}_{xh}\mathbf{x}_2) = \tanh(\mathbf{W}_{hh}\tanh(\mathbf{W}_{hh}\mathbf{h}_0 + \mathbf{W}_{xh}\mathbf{x}_1) + \mathbf{W}_{xh}\mathbf{x}_2)$$

$$\mathbf{h}_3 = \tanh(\mathbf{W}_{hh}\mathbf{h}_2 + \mathbf{W}_{xh}\mathbf{x}_3)$$

$$= \tanh(\mathbf{W}_{hh}\tanh(\mathbf{W}_{hh}\tanh(\mathbf{W}_{hh}\mathbf{h}_0 + \mathbf{W}_{xh}\mathbf{x}_1) + \mathbf{W}_{xh}\mathbf{x}_2) + \mathbf{W}_{xh}\mathbf{x}_3)$$

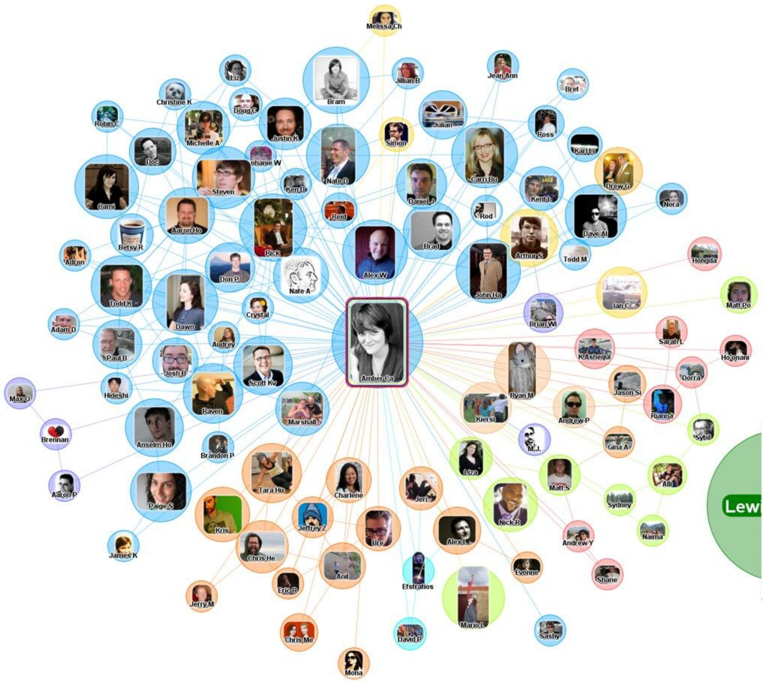
Again, the output $\hat{\mathbf{y}}$ is a weighted sum (+fixed unary operations) of input $\mathbf{x}$!

That is, the $\mathbf{W}$ is optimized to best map the input to the output in the training set, in terms of the loss function.

Transformer: Main Idea

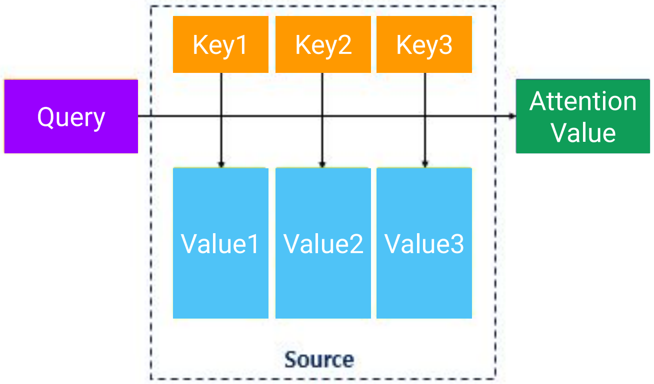
- Basic assumption: the input **x** can be split into **multiple elements** that are **organically related** to each other.
  - People in a society
  - Words in a sentence
  - Frames in a video
  - …

- **Self-attention**: Each element learns to refine its own representation by attending its **context** (other elements in the input).
  - More specifically, as a **weighted sum** of other elements in the sequence.



Review: Attention Idea

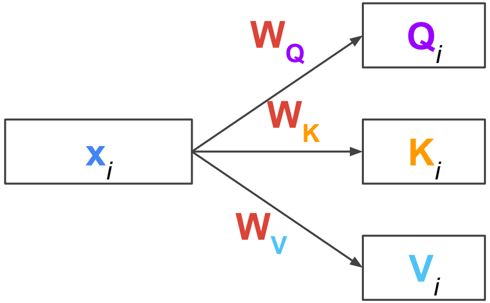
- **Attention function**: Attention (**Q**, **K**, **V**) = Attention value
  For a query (context) and key-value pairs (references), attention value is the weighted average of values, where each weight is proportional to the relevance between the query and the corresponding key.
  - **Q** and **K** must be comparable (usually in the same dimensionality).
  - **V** and Attention value are in the same dimensionality, obviously.
  - In many applications, all four of these are in same dimensionality.



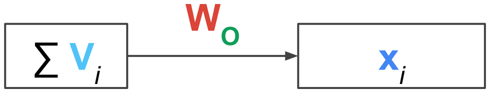
Transformer: Main Idea

- So, what should be the **Query**, **Key**, and **Value**?

- With Transformer, we **make** them!
  - From the input tokens $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$,
  - Each token $\mathbf{x}_i$ is mapped to its own **Query** $\mathbf{Q}_i$, **Key** $\mathbf{K}_i$, **Value** $\mathbf{V}_i$ vectors by a **linear transformation**.
  - The linear weights ($\mathbf{W}_\mathbf{Q}$, $\mathbf{W}_\mathbf{K}$, $\mathbf{W}_\mathbf{V}$) are the **learned parameters, shared by all inputs**.
  - $\mathbf{W}_\mathbf{Q}$ ($\mathbf{W}_\mathbf{K}$, $\mathbf{W}_\mathbf{V}$) **learns how to represent a vector to serve as** a **Query** (**Key**, **Value**) in general.



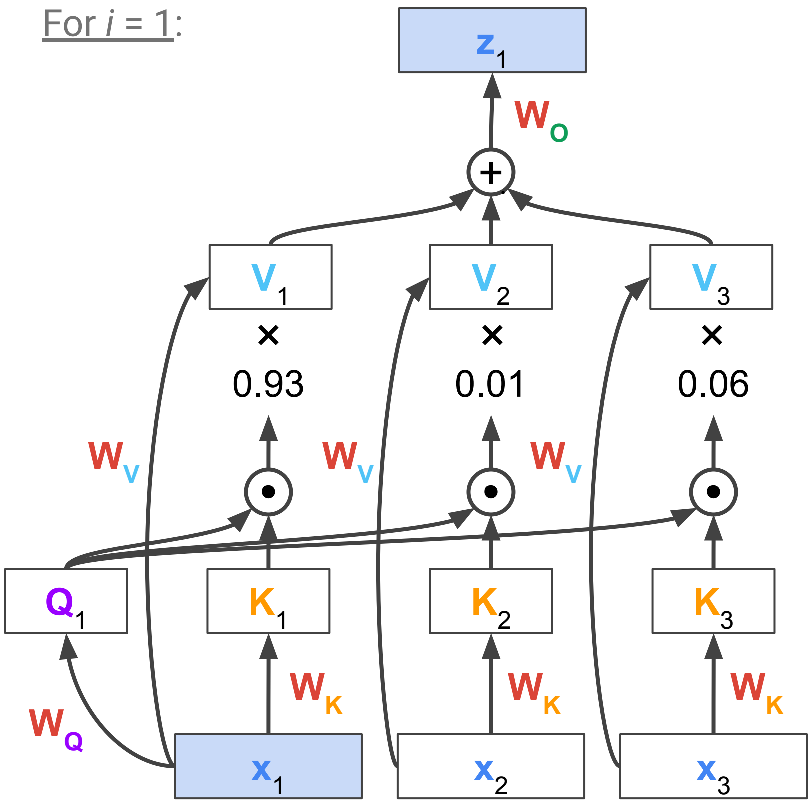
- We need another learnable parameter, $\mathbf{W}_\mathbf{O}$, which maps the **attention value** back to the original space.



Transformer: Main Idea

- Then, how do we perform attention?
  - Each token $\mathbf{x}_i$ becomes the **Query** when we learn about i.
    - *You are the main character in your life!*
  - References are all tokens $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ in the input sequence, including $\mathbf{x}_i$ itself.
    - *Your friends are a mirror that reflects you.*

- From this, we perform the attention:
  - Each element $\mathbf{x}_i$ is represented as a weighted (similarity computed using **Key**) sum of other elements (using **Value**) in $\mathbf{x}$.
  - $\mathbf{z}_i = \mathbf{w}_1\mathbf{V}_1 + \ldots + \mathbf{w}_N\mathbf{V}_N$, where $\mathbf{w}_j = \cos(\mathbf{Q}_i, \mathbf{K}_j)$
  - $\mathbf{W}_O$ maps from the **Value** space back to the original embedding space.



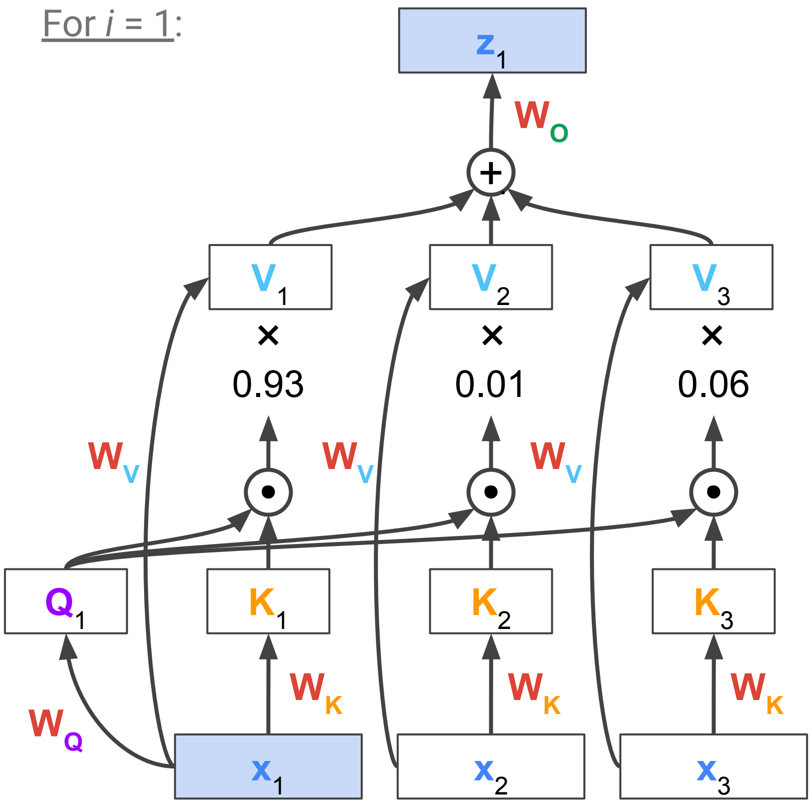
The same procedure is performed for all $i = 1, \ldots, N$.

Transformer: Main Idea

- This resulting embedding $\mathbf{z}_1$ tends to be similar to its original one ($\mathbf{x}_1$), because $\cos(\mathbf{Q}_1, \mathbf{K}_1)$ is likely to be much higher than other $\cos(\mathbf{Q}_1, \mathbf{K}_i)$.

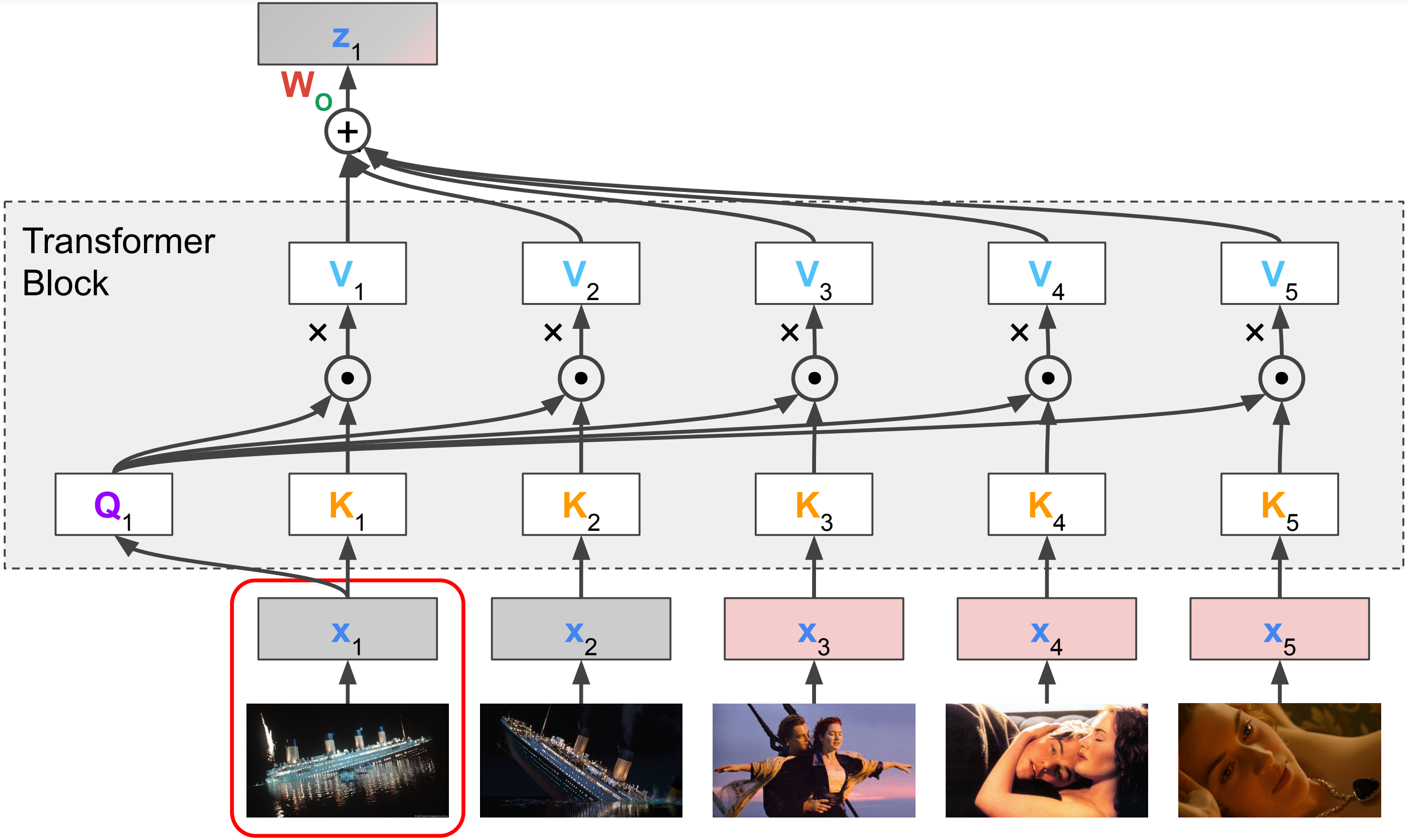
- The resulting $\mathbf{z}_1$ is still not exactly the same as the original one, slightly affected by its context (here, $\mathbf{x}_2$, $\mathbf{x}_3$).

- Usually, this step is repeated multiple times to further **contextualize**.

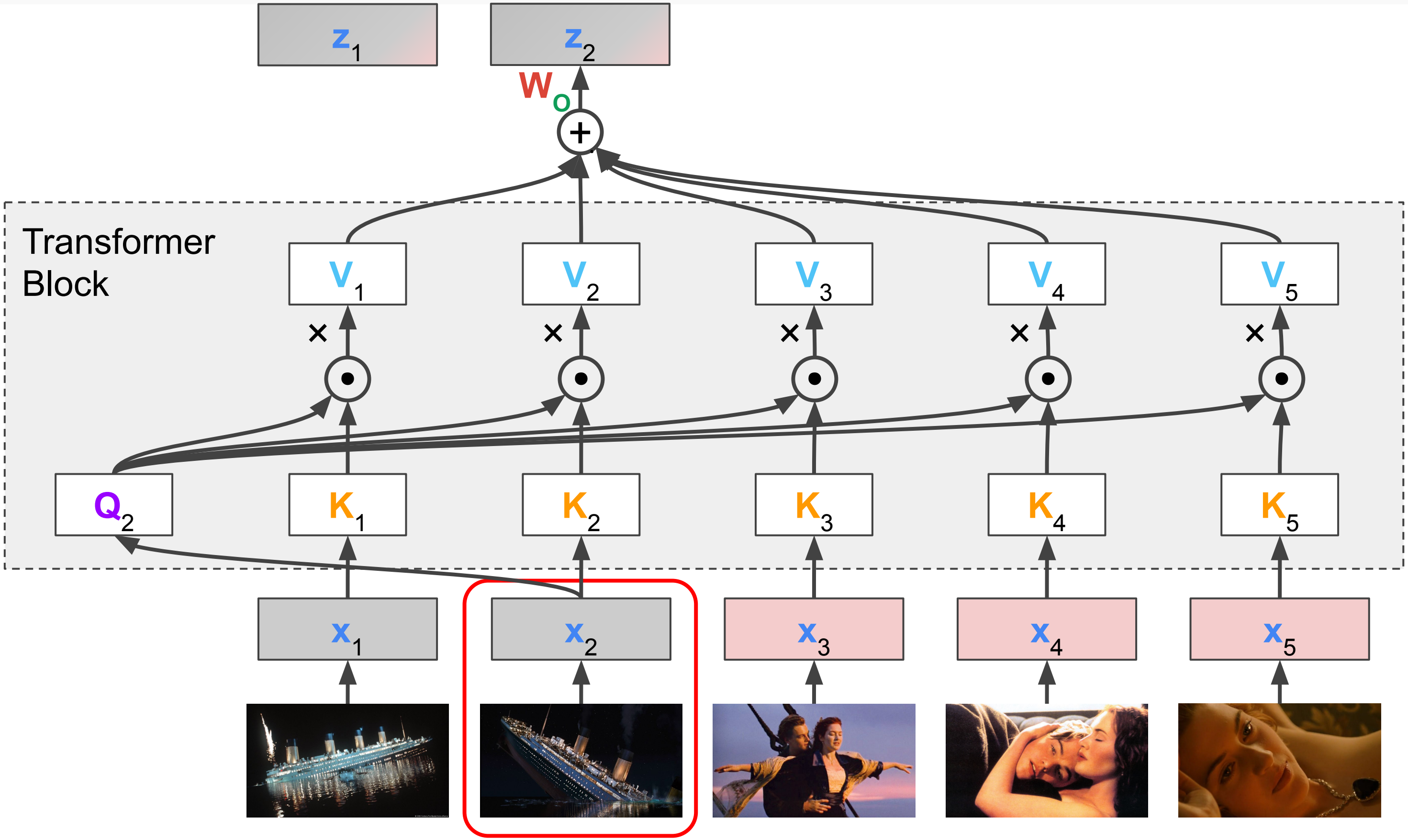


The same procedure is performed for all $i = 1, \ldots, N$.

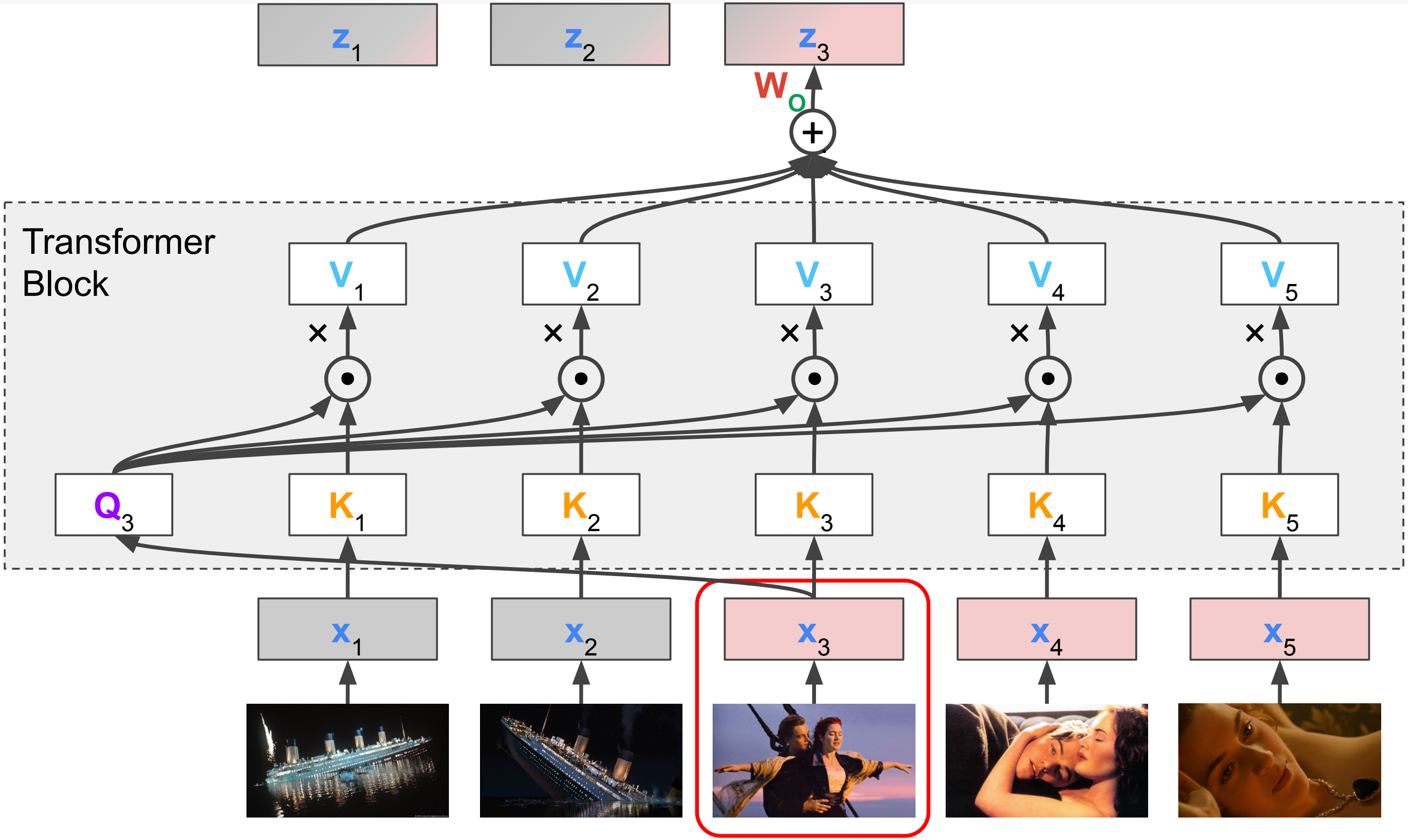
Transformer: Main Idea



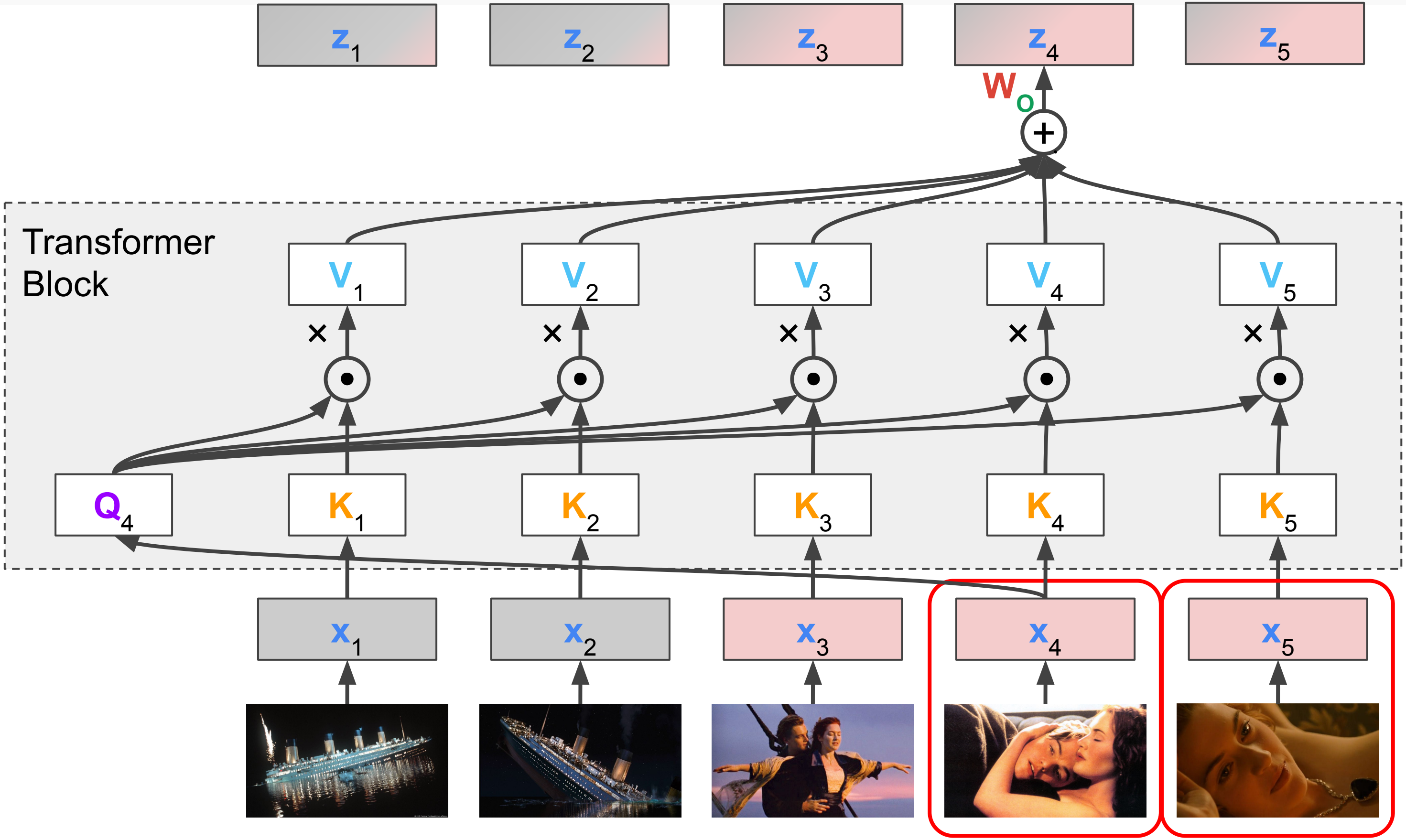
Transformer: Main Idea



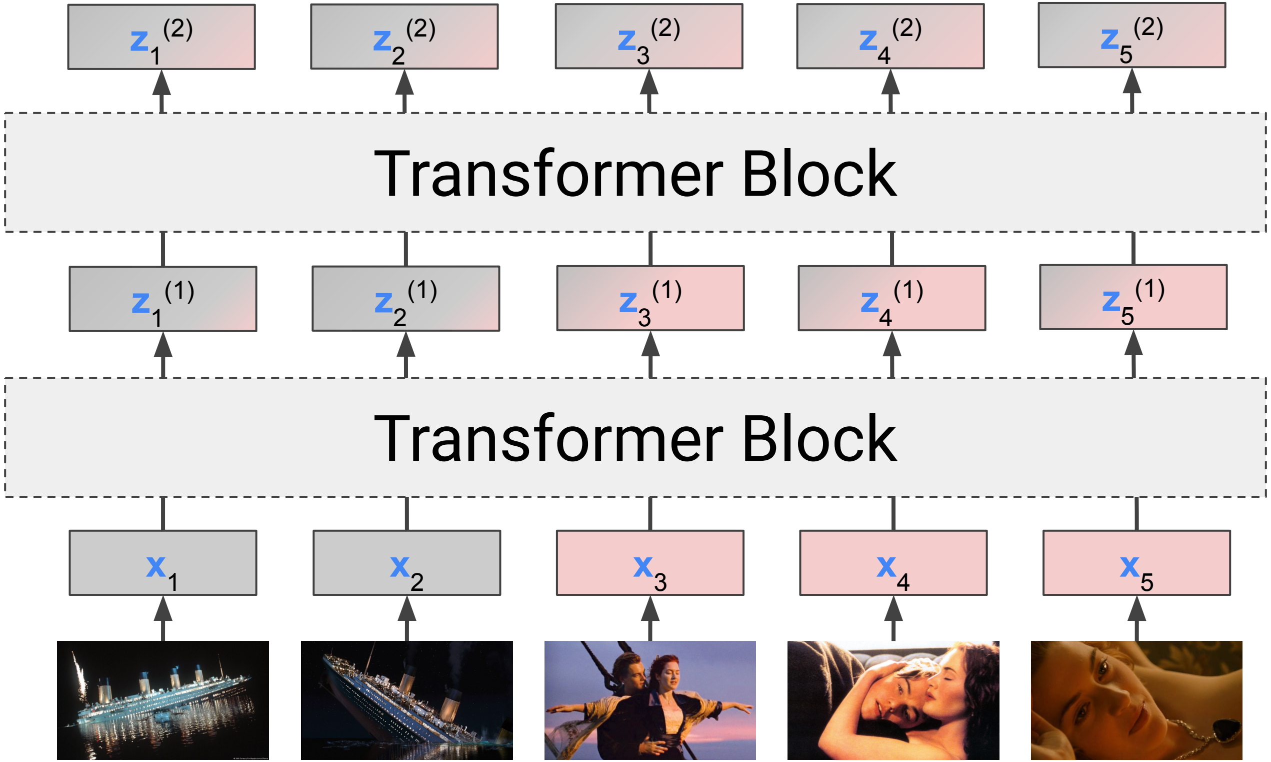
Transformer: Main Idea



Transformer: Main Idea



Transformer: Main Idea

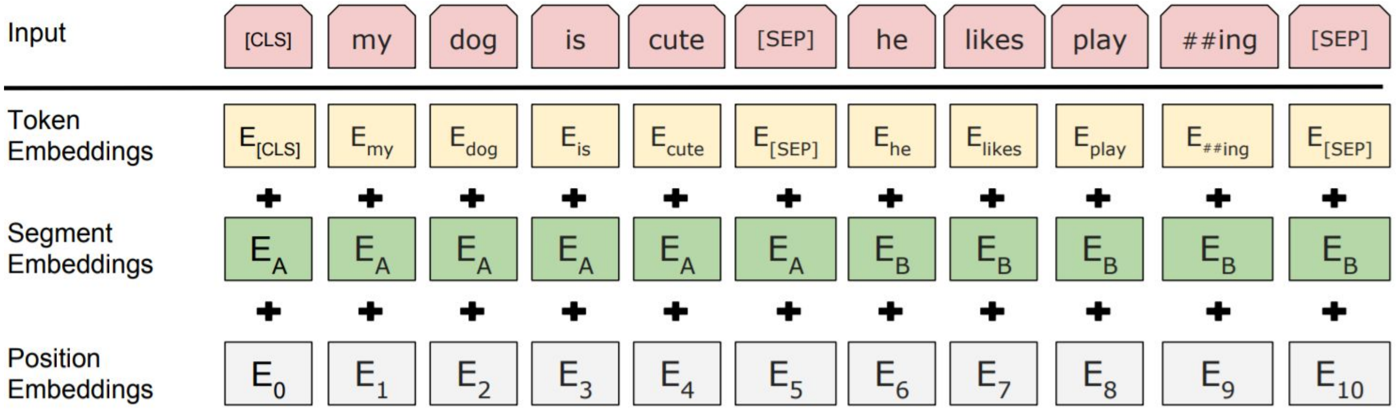


Bidirectional Encoder Representations from Transformers (BERT)

BERT

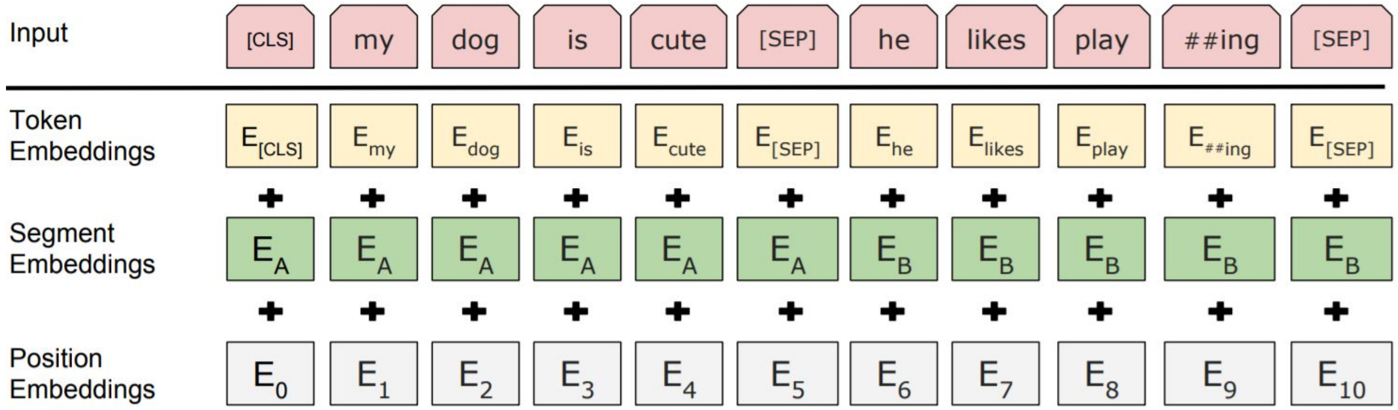
- **B**idirectional **E**ncoder **R**epresentations from **T**ransformers
  - Large-scale **pre-training** of word embeddings using **Transformer encoder**
  - Self-supervised: no human rating required
  - Use the encoder (bi-directional; no masking) only

- https://arxiv.org/pdf/1810.04805.pdf



BERT

- Input sequence consists of **two sentences**, with sum of three things:
  - **Token embedding**: a pre-trained word embedding (WordPiece)
    - **[CLS]**: Classification token, put always at the beginning. Final hidden state for this token is used as the aggregate sequence representation for classification tasks
    - **[SEP]**: Separator token, used to mark the end of a sentence
  - **Segment embedding**: a learned embedding indicating which sentence each token belongs to
  - **Position embedding**: a learned embedding for each position



BERT

- Training task 1: **Masked Language Modeling (MLM)**
  - Similar to **sentence completion** in standard English exams: figuring out the hidden words using the **context**.
  - Masking 15% of tokens randomly (substituting it to a special [MASK] token).
  - Classify the output embedding for these positions across the vocabulary.

4. The old man could not have been accused of ------ his affection; his conduct toward the child betrayed his ------ her.

(A) lavishing. .fondness for
(B) sparing. .tolerance of
(C) rationing. .antipathy for
(D) stinting. .adoration of
(E) promising. .dislike of

BERT

- Training task 2: **Next Sentence Prediction (NSP)**
  - A binary classification problem, predicting if the two sentences in the input are consecutive or not.
  - Half of training data contains two consecutive sentences (B is the actual next sentence of A).
  - The other half contains two sentences randomly chosen from the corpus.

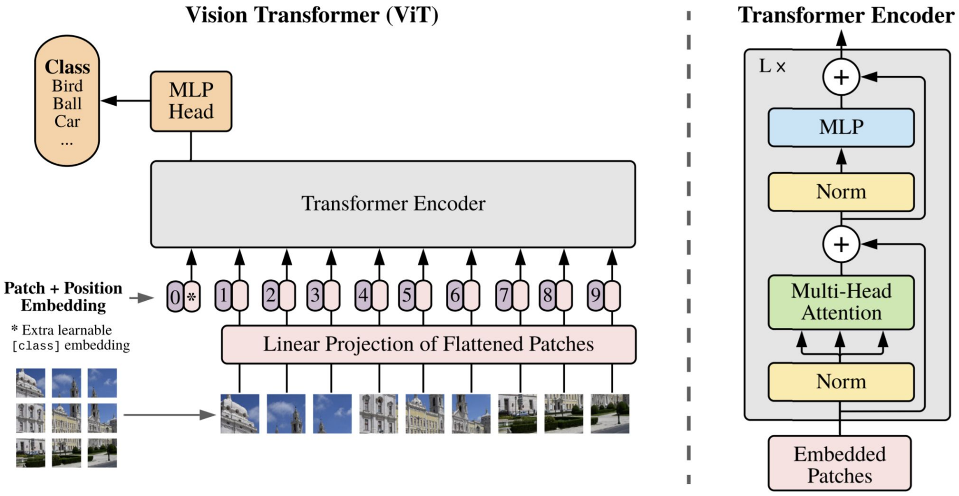
- According to the authors, their model achieved ~98% accuracy on this task, and this was very beneficial to multiple tasks.
  - Later, turns out to be less important than MLM.

- These days, the pre-trained BERT is a default choice for word embeddings.

Vision Transformers

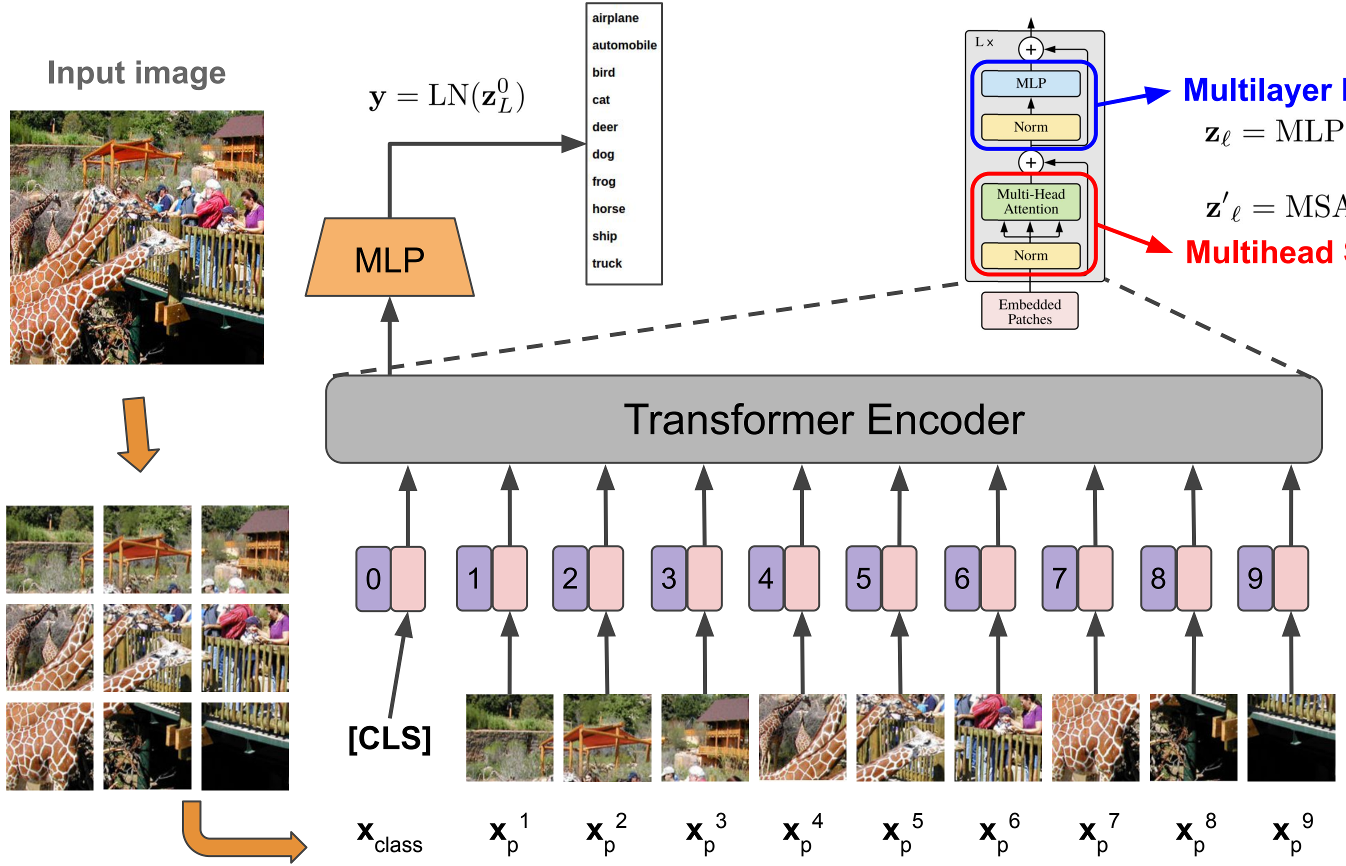
ViT: Vision Transformer

- The standard Transformer model is directly applied to images:
    - An image is split into 16×16 patches. (Each **token** is a **16×16 image patch** instead of a word.)
    - The sequence of linear embeddings of these patches are fed into a **Transformer**.
    - Image patches are treated on the same way as tokens (words).
    - Eventually, an **MLP** is added on top of the `[CLS]` **token** to classify the input image.



https://arxiv.org/pdf/2010.11929.pdf

ViT: Vision Transformer



Multilayer Perceptron (MLP)

$$\mathbf{z}_\ell = \mathrm{MLP}(\mathrm{LN}(\mathbf{z'}_\ell)) + \mathbf{z'}_\ell$$

$$\mathbf{z'}_\ell = \mathrm{MSA}(\mathrm{LN}(\mathbf{z}_{\ell-1})) + \mathbf{z}_{\ell-1}$$

Multihead Self-Attention (MSA)

$$\mathbf{z}_0 = [\mathbf{x}_{\text{class}};\, \mathbf{x}_p^1\mathbf{E};\, \mathbf{x}_p^2\mathbf{E};\cdots;\, \mathbf{x}_p^N\mathbf{E}] + \mathbf{E}_{pos}$$

$$\mathbf{E} \in \mathbb{R}^{(P^2\cdot C)\times D},\ \mathbf{E}_{pos} \in \mathbb{R}^{(N+1)\times D}$$

Positional Encoding E_{pos}

N+1 → D

(N=#patch, D=1024)

Patch embedding

by Linear projection E

P^2C → D (C=3, D=1024)

Input tokens x

P×P (P=16 or 32)

ViT: Experiments and Discussion

- ViT outperforms previous SOTA (ResNet152) 😀

	Ours-JFT (ViT-H/14)	Ours-JFT (ViT-L/16)	Ours-I21k (ViT-L/16)	BiT-L (ResNet152x4)	Noisy Student (EfficientNet-L2)
ImageNet	**88.55** $\pm$ 0.04	87.76 $\pm$ 0.03	85.30 $\pm$ 0.02	87.54 $\pm$ 0.02	88.4/88.5*
ImageNet ReaL	**90.72** $\pm$ 0.05	90.54 $\pm$ 0.03	88.62 $\pm$ 0.05	90.54	90.55
CIFAR-10	**99.50** $\pm$ 0.06	99.42 $\pm$ 0.03	99.15 $\pm$ 0.03	99.37 $\pm$ 0.06	—
CIFAR-100	**94.55** $\pm$ 0.04	93.90 $\pm$ 0.05	93.25 $\pm$ 0.05	93.51 $\pm$ 0.08	—
Oxford-IIIT Pets	**97.56** $\pm$ 0.03	97.32 $\pm$ 0.11	94.67 $\pm$ 0.15	96.62 $\pm$ 0.23	—
Oxford Flowers-102	99.68 $\pm$ 0.02	**99.74** $\pm$ 0.00	99.61 $\pm$ 0.02	99.63 $\pm$ 0.03	—
VTAB (19 tasks)	**77.63** $\pm$ 0.23	76.28 $\pm$ 0.46	72.72 $\pm$ 0.21	76.29 $\pm$ 1.70	—
TPUv3-core-days	2.5k	0.68k	0.23k	9.9k	12.3k

- It takes 300 days with 8 TPUv3 cores 😨

US EUROPE ASIA PACIFIC

Version	On-demand	Preemptible
Cloud TPU v2	\$4.50 / TPU hour	\$1.35 / TPU hour
Cloud TPU v3	\$8.00 / TPU hour	\$2.40 / TPU hour

https://cloud.google.com/tpu#tab1

- TPU v3 costs \$8.00 per hour.
- \$8 × 24 hr/day × 2500 day = **\$480,000** to train this model once!

ViT: Experiments and Discussion

- ViT performs well only when trained on an **extremely large dataset** (*e.g.*, JFT-300M). Why?
  - ViT does **NOT imply any inductive bias** (spatial locality & positional invariance) of CNNs.
  - It needs to learn those **purely from the data**.
    → It requires large amount of examples.
  - Once sufficient training examples provided, however, it can outperform CNN-based models, as it is **capable of modeling hard cases beyond spatial locality**.

Input Attention

Example:
The model is able to attend wide range of the image even at an early layer. (as opposed to CNNs)

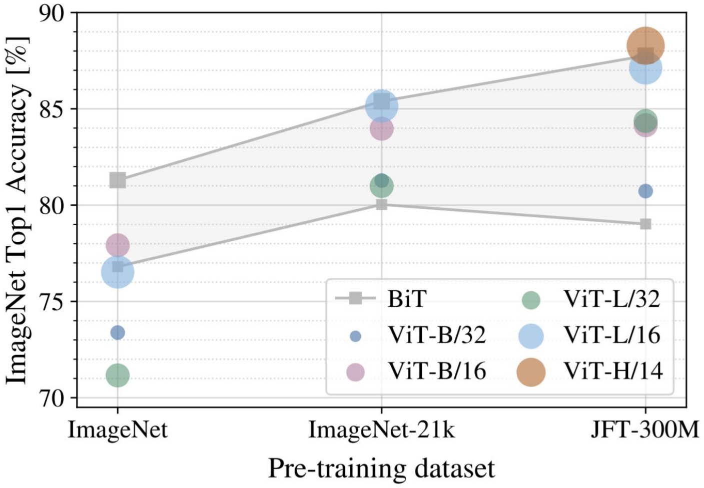
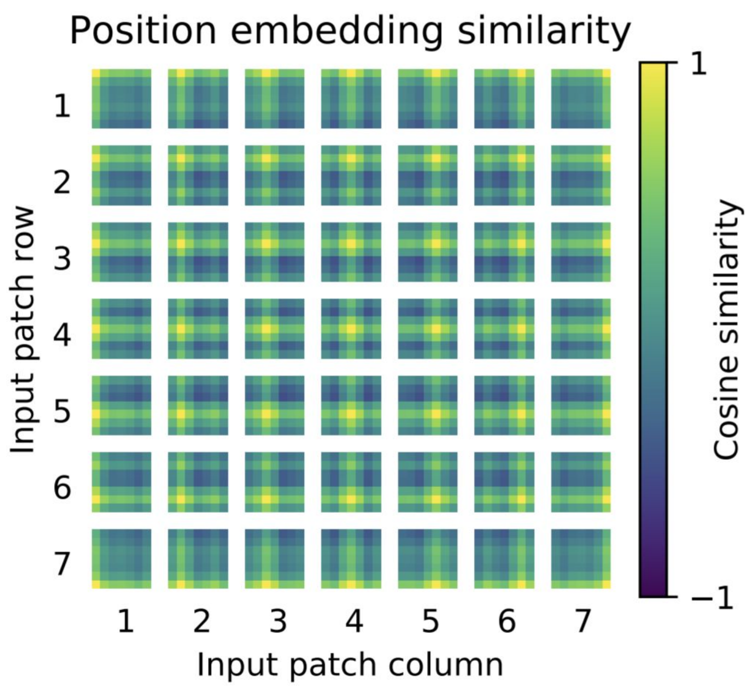


Figure 3: Transfer to ImageNet. While large ViT models perform worse than BiT ResNets (shaded area) when pre-trained on small datasets, they shine when pre-trained on larger datasets. Similarly, larger ViT variants overtake smaller ones as the dataset grows.

ViT: Position Embeddings

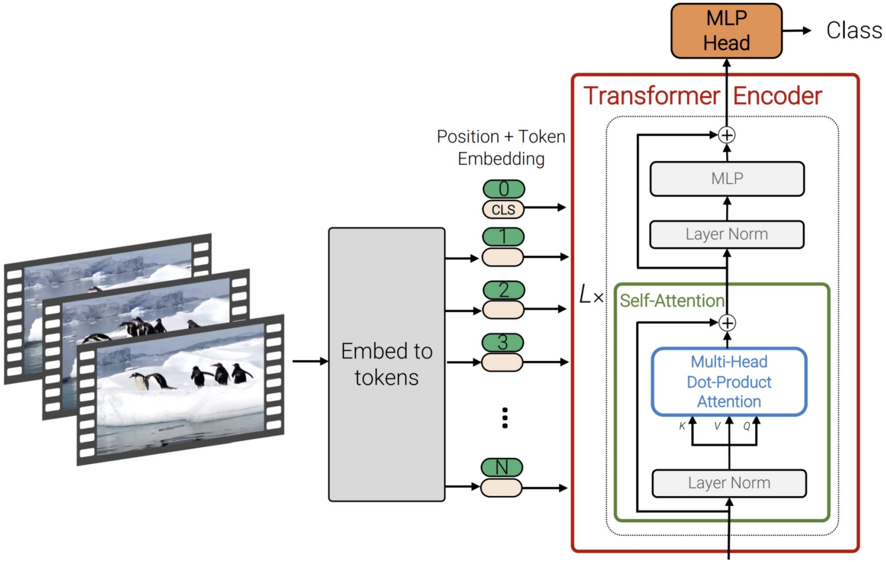
- ViT learns to encode distance within the image in the similarity of position embeddings.
  - Closer patches tend to have more similar position embeddings.
- The row-column structure appears.
  - Patches in the same row/column have similar embeddings, automatically learned from data.
- Hand-crafted 2D-aware embedding variants do not yield improvements for this reason.



ViViT: Video Vision Transformer

- Naturally extending the idea of ViT to **video classification** task. (Model 1)
  - Each frame is split into $n_w \times n_h$ image patches, then total $n_t \times n_w \times n_h$ patches contextualize from each other using Transformer Encoder.
    (n_h: # rows, n_w: # columns, n_t: # frames)

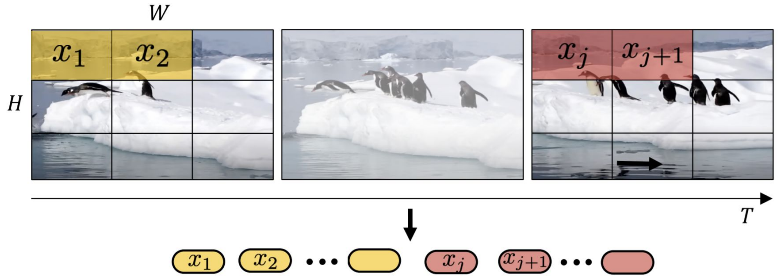
- **Computational overhead** is a serious issue with this naive extension.
  - Attention overhead: $O(n_h^2\ n_w^2\ n_t^2)$ 😱



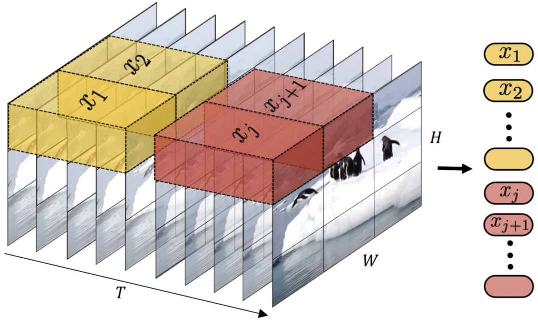
https://arxiv.org/pdf/2103.15691.pdf

ViViT: Video Vision Transformer

- Basic ideas to reduce computational overhead
  - Uniformly sampling some frames across the time domain (**Uniform frame sampling**)
    e.g., one frame per every 2 frames

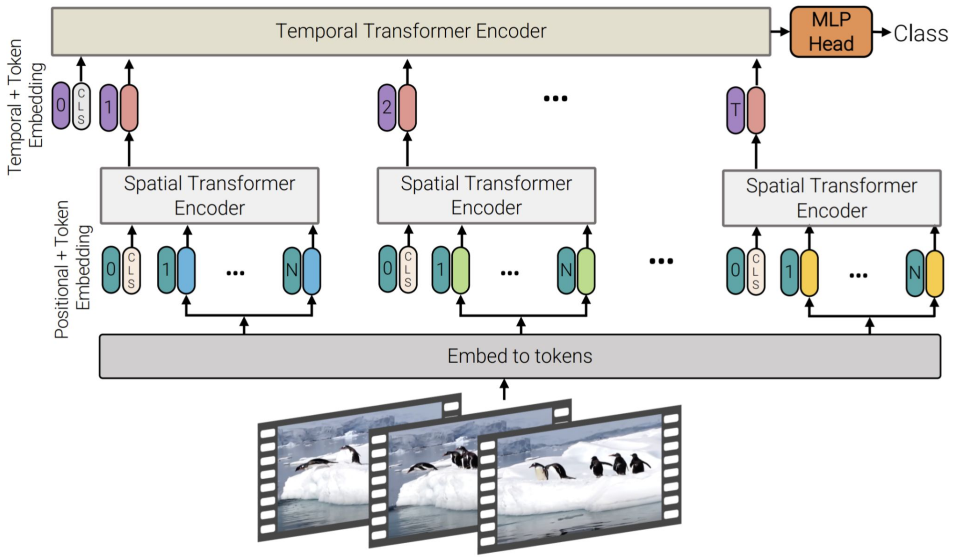


  - Extracting non-overlapping, spatio-temporal tubes from the input volume, and linearly projecting this (**Tubelet embedding**): fuses spatio-temporal information during tokenization.



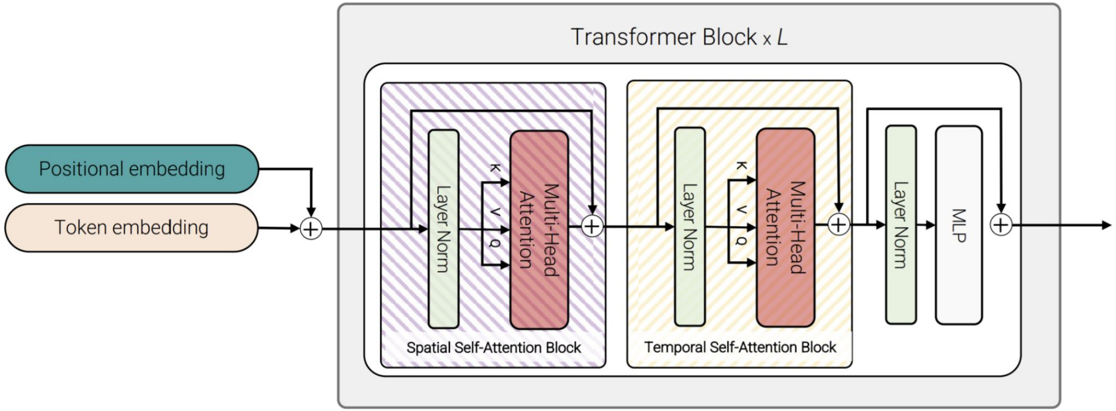
ViViT: Video Vision Transformer

- Factorized Encoder (Model 2):
  - Spatial encoder and temporal encoder are **sequentially separated**.
  - First, only spatial interactions are contextualized through Spatial Transformer Encoder (=ViT).
  - Then, each frame is encoded to a single embedding, fed into the Temporal Transformer Encoder.
  - Complexity: $O(n_h^2 n_w^2 + n_t^2)$



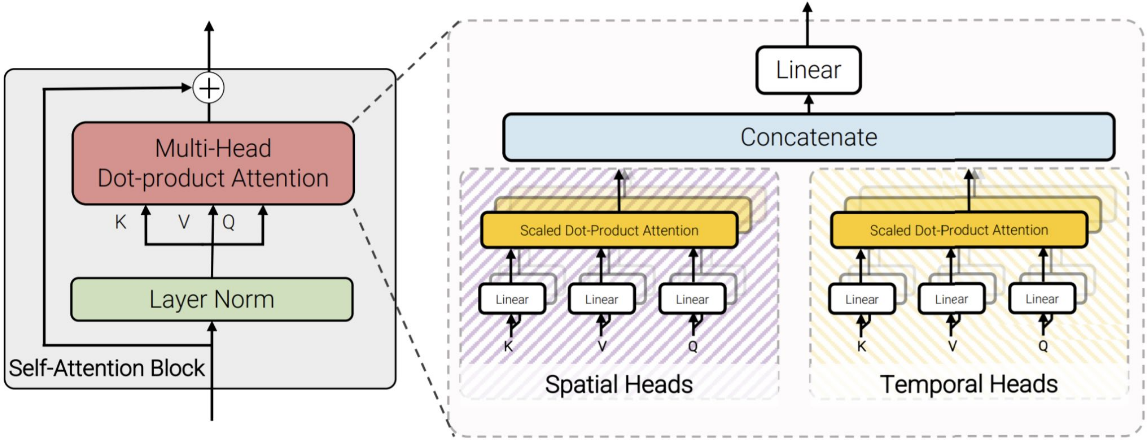
ViViT: Video Vision Transformer

- Factorized Self-Attention (Model 3)
  - Contains the same number of Transformer layers as the naive model (Model 1).
  - Instead of computing multi-headed self-attention across all pairs of tokens,
    - First only compute self-attention **spatially** (among all tokens extracted from the same temporal index),
    - then **temporally** (among all tokens extracted from the same spatial index)
  - No classification token ([CLS]) is used to avoid ambiguities.



ViViT: Video Vision Transformer

- Factorized Dot-Product Attention (Model 4)
  - Recall that the Transformer is based on **multihead** attentions.
  - Half of the attention heads are designed to operate with keys and values from same spatial indices: $\mathbf{K}_s, \mathbf{V}_s \in \mathbb{R}^{n_h \cdot n_w \times d}$ $\quad \mathbf{Y}_s = \text{Attention}(\mathbf{Q}, \mathbf{K}_s, \mathbf{V}_s)$
  - The other half operate with keys and values from same temporal indices: $\mathbf{K}_t, \mathbf{V}_t \in \mathbb{R}^{n_t \times d}$ $\quad \mathbf{Y}_t = \text{Attention}(\mathbf{Q}, \mathbf{K}_t, \mathbf{V}_t)$
  - $\mathbf{Y} = \text{Concat}(\mathbf{Y}_s, \mathbf{Y}_t)\mathbf{W}_O$



ViViT: Experiments and Discussion

- Dataset sparsity problem:
  - Recall that **ViT requires extremely large dataset** to perform well.
  - There's **no video dataset** at such a scale 😟
  - They **initialized with ViT**, pre-trained on large image dataset.
- Comparing Model 1, 2, 3, 4:
  - The naive model (Model 1) performs the best, but most expensive.
  - Model 2 is a good trade-off, with fastest runtime and near-the-best accuracy.

	K400	EK	FLOPs ($\times 10^9$)	Params ($\times 10^6$)	Runtime (ms)	
Model 1: Spatio-temporal	$\boxed{80.0}$	43.1	$\boxed{455.2}$	88.9	$\boxed{58.9}$	**Best performance**
Model 2: Fact. encoder	78.8	$\boxed{43.7}$	$\boxed{284.4}$	115.1	$\boxed{17.4}$	**Most efficient**
Model 3: Fact. self-attention	77.4	39.1	372.3	117.3	31.7	
Model 4: Fact. dot product	76.3	39.5	277.1	88.9	22.9	
Model 2: Ave. pool baseline	75.8	38.8	283.9	86.7	17.3	

Top 1 accuracy

Transformer-based Image-Text Models

Motivating Questions

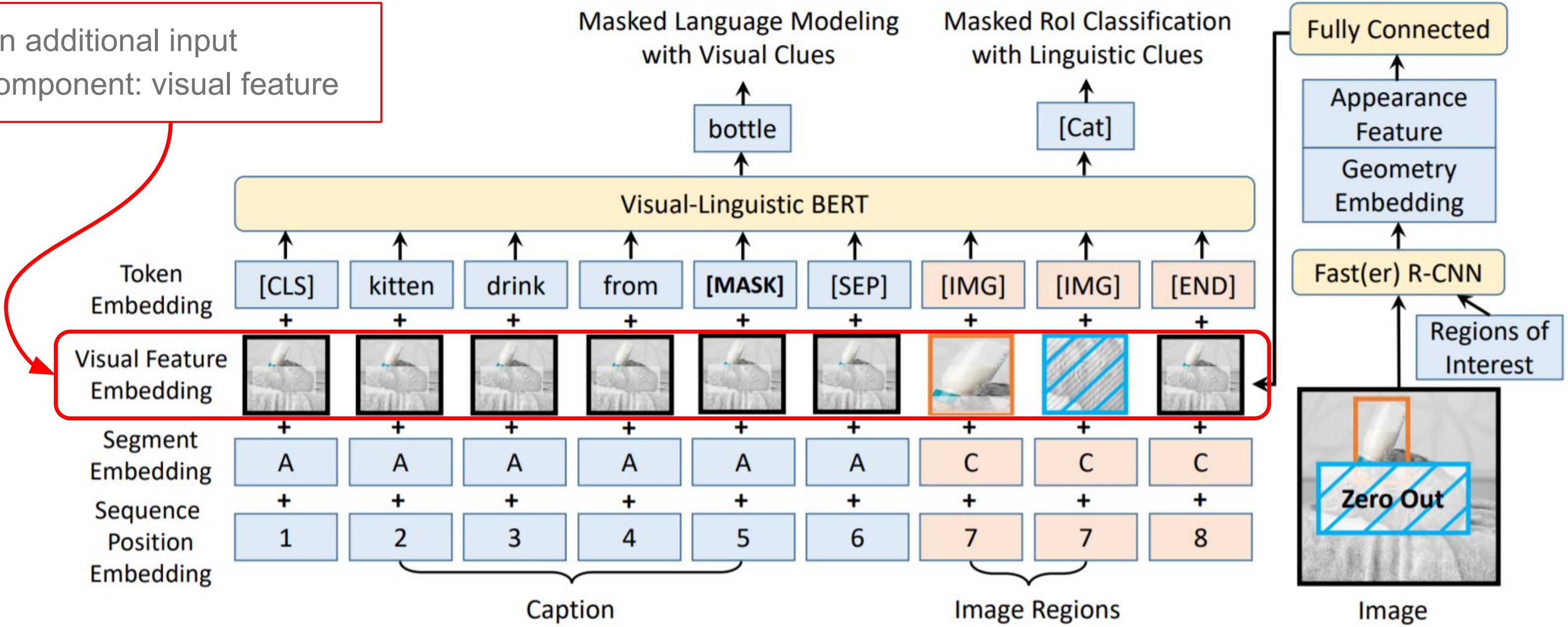
- How can we apply Transformers to learn image-text correspondence?
    - Text is sequential in nature.
    - **Q**: How can we represent an image into a sequence?

- How can we collect (image, text) pairs?
    - Think about what correlation we want to learn from.
    - Direct and explicit description of an image may need human raters.
    - **Q**: Any indirect way to collect noisy labels?
        - Image search query + clicked images
        - An image and text co-existed in a web page
        - Video thumbnail + the video’s title
        - Images and captions posted on SNS
        - ...

VL-BERT

- Applied the Transformer to model image-text (caption) correspondence.
  - Training example: an image and its caption (a sentence) pair, instead of two sentences.
  - For VQA application, an image and two sentences (question, answer).

An additional input component: visual feature

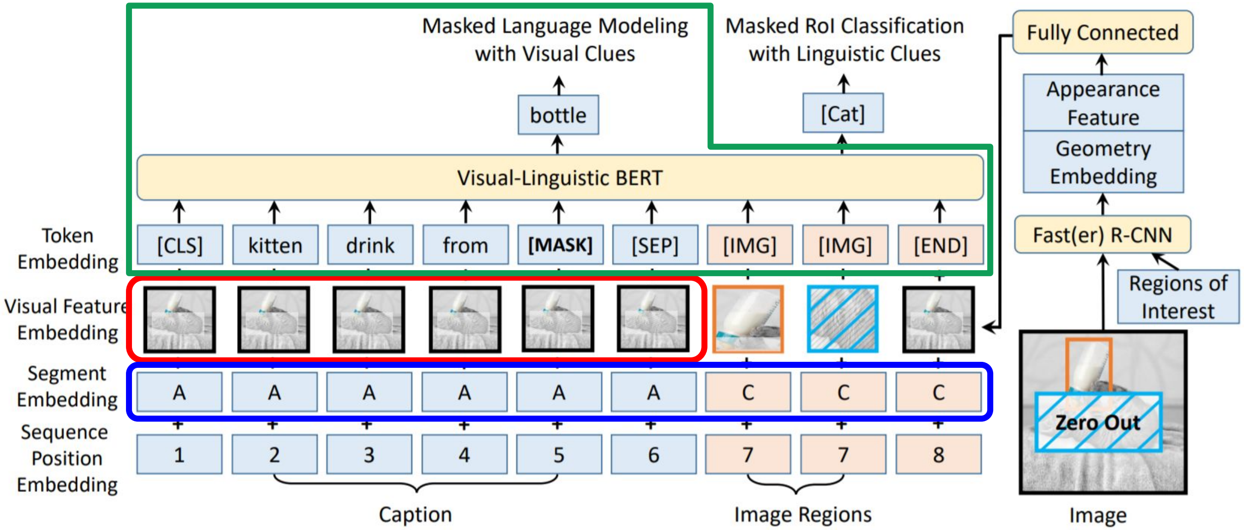


https://arxiv.org/abs/1908.08530

VL-BERT

The **text** part is almost identical to the original BERT, except

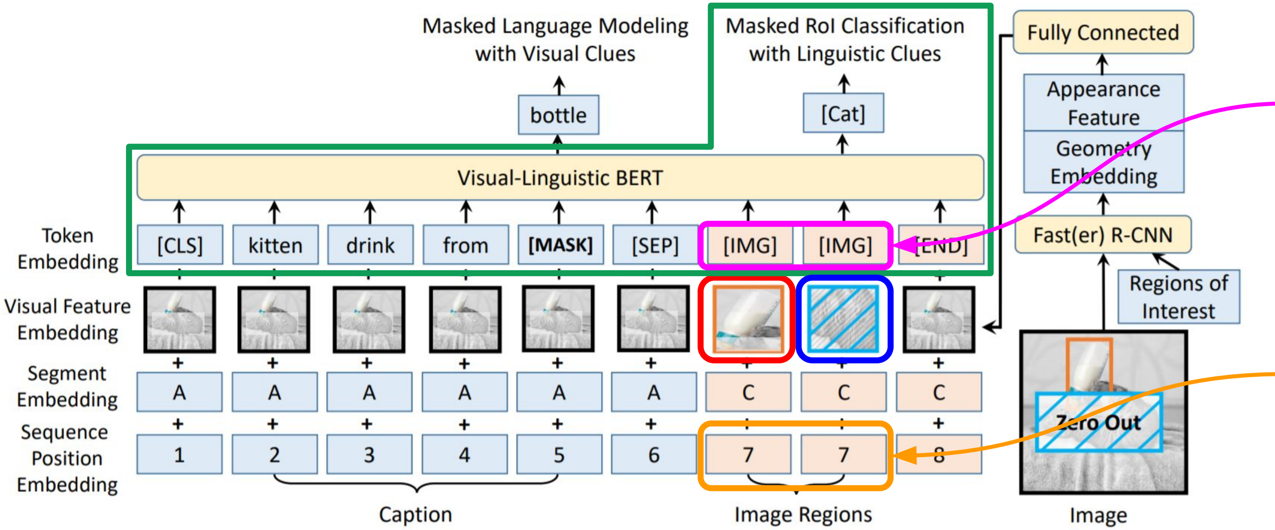
- For the visual feature, the entire image feature is added by default.
- Segment embedding: A is for text, B is for another text (for VQA), C is for image.
- MLM itself is the same, but it now **attends the visual tokens** as well as other words.



VL-BERT

The **image** part is new!

- Using Fast(er) R-CNN, Region of Interests are extracted, and each of them is treated as a token.
- Similarly to MLM, some RoIs are zeroed out.
- **Masked RoI classification**: classify the zeroed out region based on context (visual + linguistic).



For the token embedding, a special token [IMG] is added by default.

Geometry embedding: For the visual positional embedding, the bounding box [left/W, top/H, right/W, bottom/H] is encoded by sinusoidal functions.

VL-BERT: Downstream Tasks

- Visual Question Answering (VQA)
  - Given a natural image, an open-ended or multiple-choice perception question is asked.
  - The model needs to generate the correct answer.

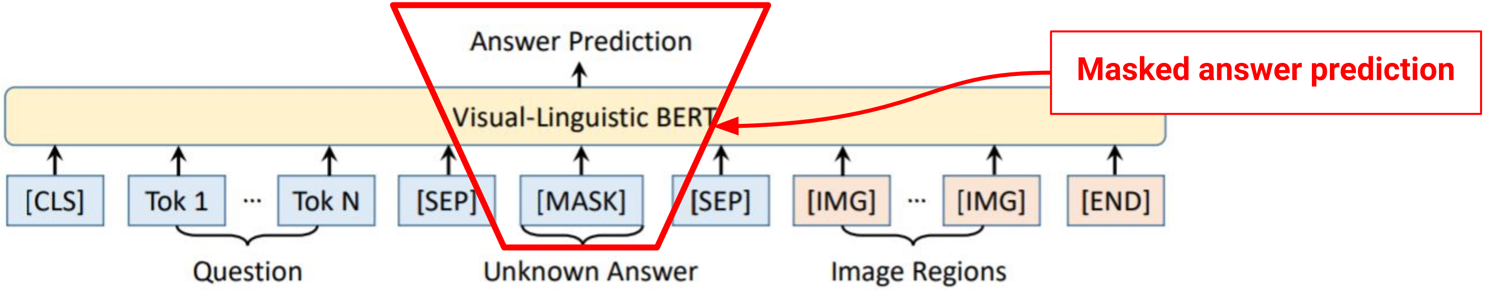
Q: What days might I most commonly go to this building?
A: Sunday

Q: Is this photo from the 50's or the 90's?
A: 50's

Q: What phylum does this animal belong to?
A: chordate, chordata

Q: How many chromosomes do these creatures have?
A: 23

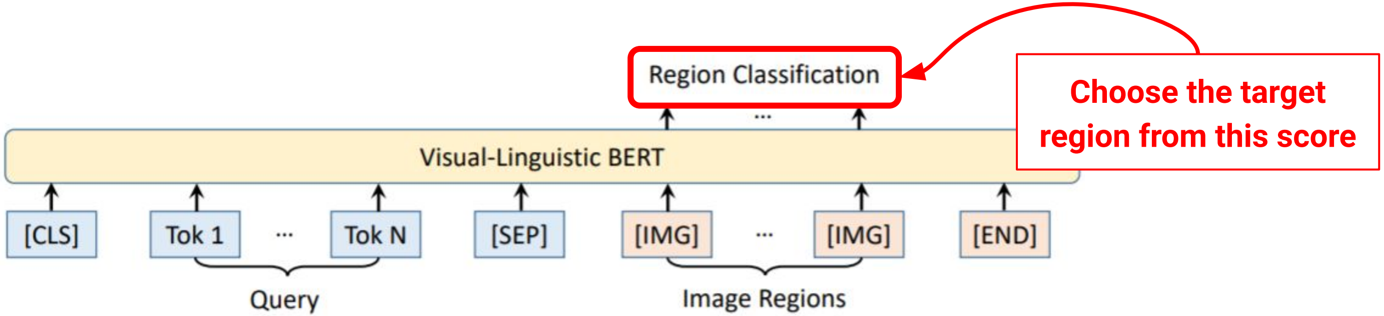
Q: What is the warmest outdoor temperature at which this kind of weather can happen?
A: 32 degrees



VL-BERT: Downstream Tasks

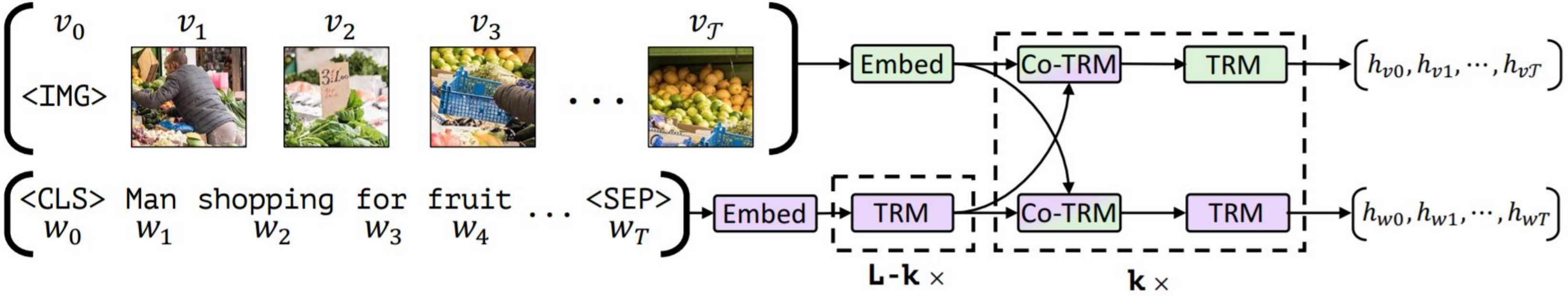
- Referring Expression Comprehension
  - A natural language phrase to an object in an image (referring expression) is given.
  - The model needs to locate the target object.

blurry person with sleeveless and sitting — man in full view in all black — small one grazing — books about bears

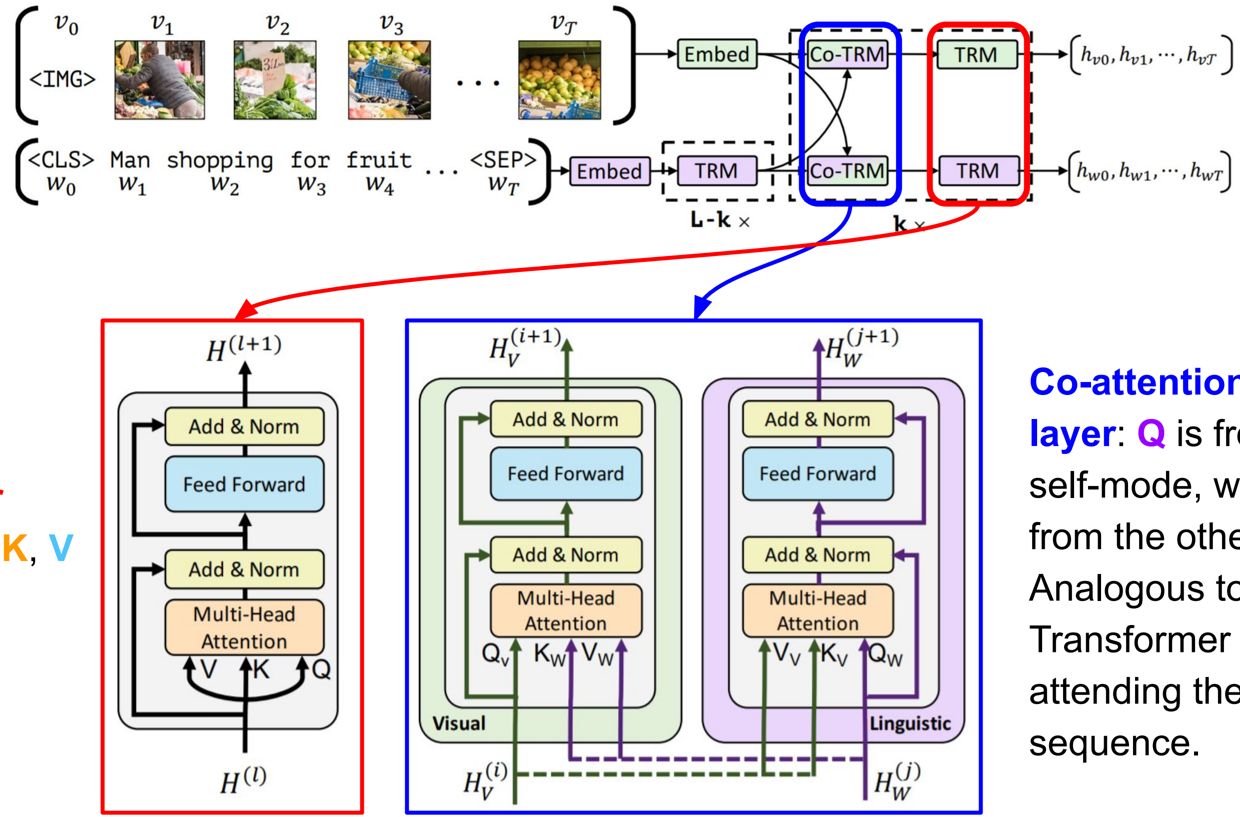


ViLBERT

- Similar to VL-BERT, but added **cross-modal attention** within the BERT.
  - Image region features are extracted using a pre-trained Faster R-CNN model.
  - Text tower is embedded using a pre-trained BERT model, then goes through additional Transformer blocks. No such additional tuning on visual side.
  - Each tower repeatedly attends **cross-modal** and it**self**, similarly to the Transformer decoder.
- https://arxiv.org/pdf/1908.02265.pdf



ViLBERT: Co-attention Transformer

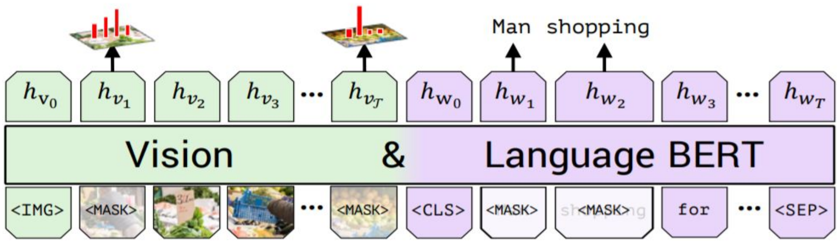


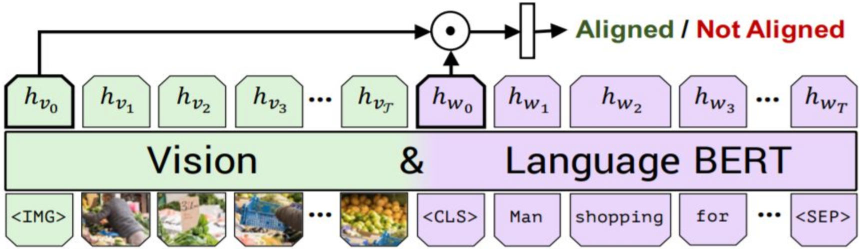
Regular Transformer block: all **Q**, **K**, **V** are from the self-mode.

Co-attention Transformer layer: **Q** is from the self-mode, while **K**, **V** are from the other side. Analogous to the Transformer decoder attending the encoder sequence.

ViLBERT: Training

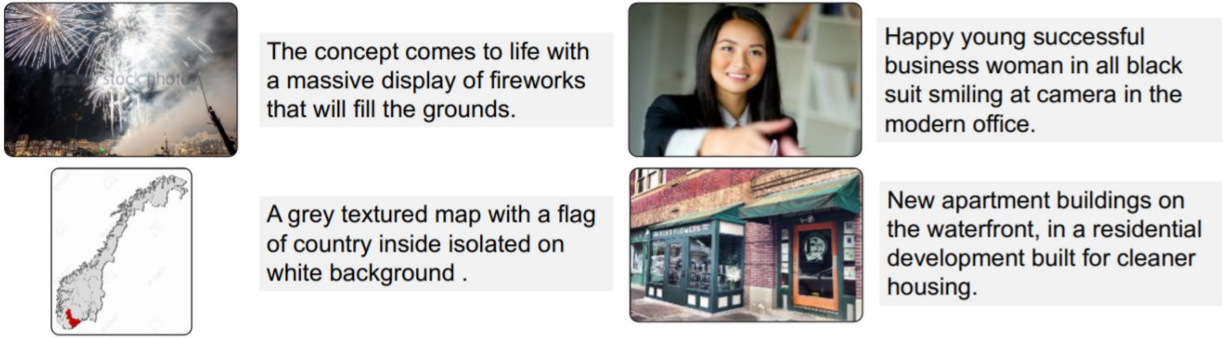
- Masked multi-modal modelling task
  - Analogous to **MLM** in BERT.
  - For the image regions, the **distribution over semantic classes** is predicted.
  - **Prediction of the Faster R-CNN** model is used as the ground truth.
  - The text part is same as original MLM; **attending on visual** signals as well.
- Multi-modal alignment task
  - Analogous to **NSP** in BERT.
  - The model takes (image, text) pair as an input, and multiple image patches are extracted and fed.
  - The output embeddings corresponding to [IMG] and [CLS] are trained to represent the **entire image and sentence**.
  - Trained to classify if these are **aligned or not**.

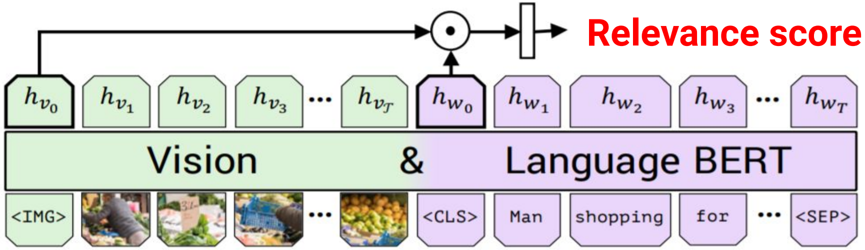




ViLBERT: Downstream Tasks

- Caption-Based Image Retrieval
  - Given a text describing an image, retrieve the most relevant image from a corpus.
  - Similar to image search on web search engine, but the query tends to be more descriptive.





Relevance score

With the text query fixed, compute this score for all images in the corpus, and return the top-*k*.

Transformer-based Video-Text Models

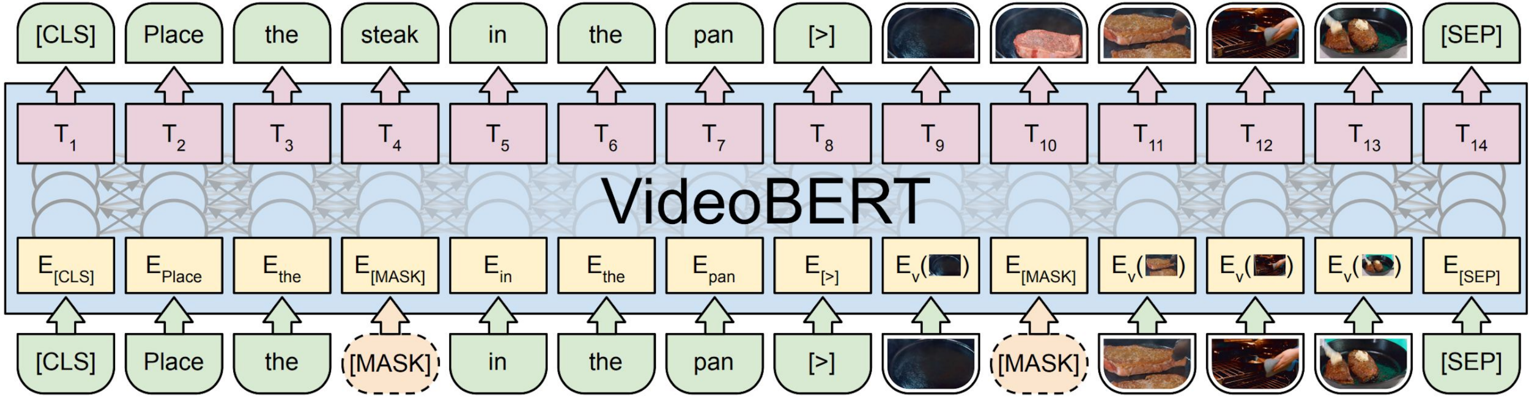
Motivating Questions: Revisited

- How can we apply Transformers to learn **video**-text correspondence?
    - Text is sequential in nature.
    - **And, video is also sequential in nature!**

- How can we collect (**video**, text) pairs?
    - Think about what correlation we want to learn from.
    - Human raters on videos are **much more costly (time, money, …)** than on images.
    - **Q**: Any other way to collect noisy labels?
        - Video search query + clicked videos
        - A video and its title/descriptions on YouTube
        - Video and its own **ASR** (Automatic Speech Recognition) or uploader-provided **captions**
        - ...

VideoBERT

- Instead of spatial sampling in image models, frames are sampled temporally.
    - Sampled frames for every 1.5 sec
    - S3D features are extracted to represent each frame (1024-D).
- Both visual (sampled frames) and text (ASR) are from a part of a video.
    - Then, the main training task is **temporal correspondence** between the frames and ASR.
- Focused on cooking/recipe videos.
- https://arxiv.org/pdf/1904.01766.pdf



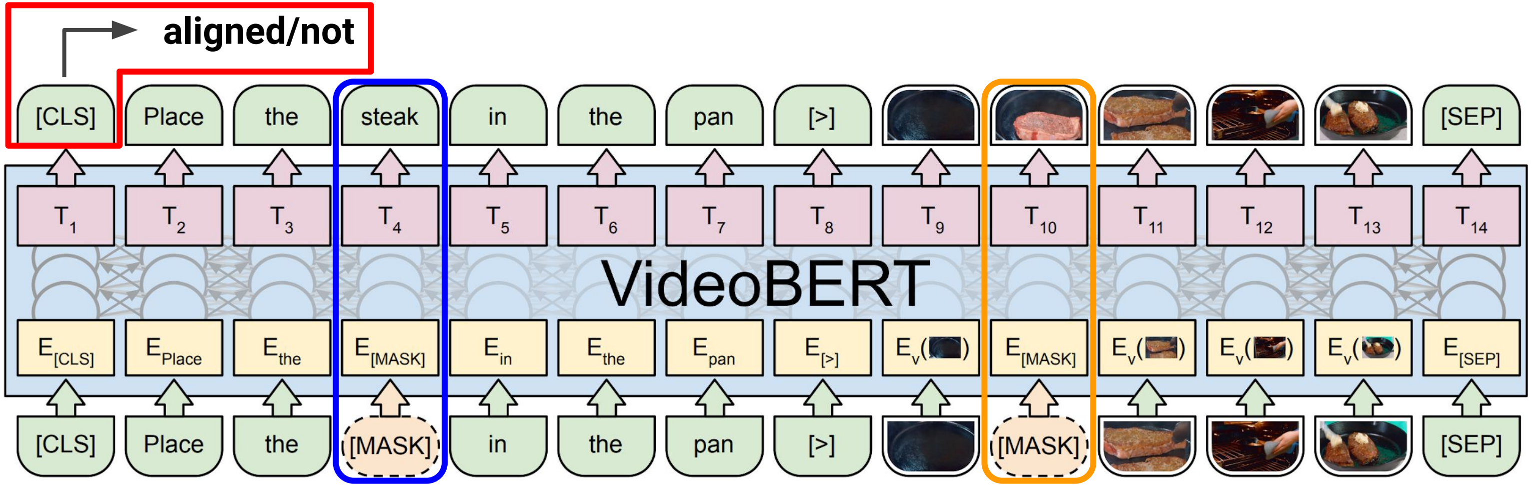
VideoBERT: Video Tokenization

- As we do not use a detection model (*e.g.*, Faster R-CNN) anymore, no labels are available for the sampled frames.
- Solution: **video tokenization** (similar to ‘visual words’)
  - **Clusters** the frames in the dataset (they used hierarchical k-means).
  - Each frame is now represented as another frame **closest to the centroid** of its cluster.
  - The representative may look different visually, but **preserve semantics**.

Original: *Their representative:*

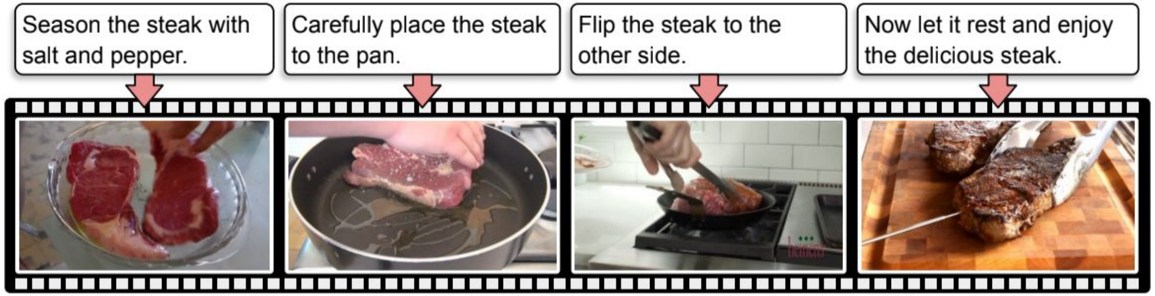
VideoBERT: Training

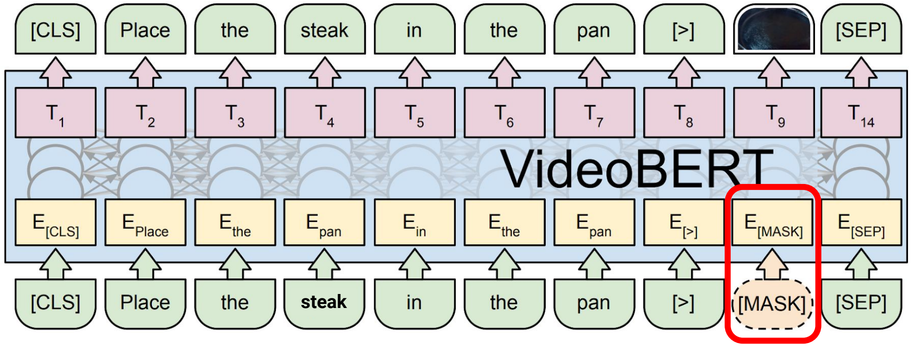
- **Linguistic-visual alignment task**: from the final hidden state of [CLS] token, the model classifies if the input video clip and text is temporally aligned or not.
- **Masked language modeling (MLM)**: same as BERT
- **Masked frame modeling (MFM)**: similar to MLM; learning to **classify the image cluster**.



VideoBERT: Downstream Tasks

- Recipe Illustration
  - Conditioned on an input sentence, **generate a video token for visualization** (w/ centroids).

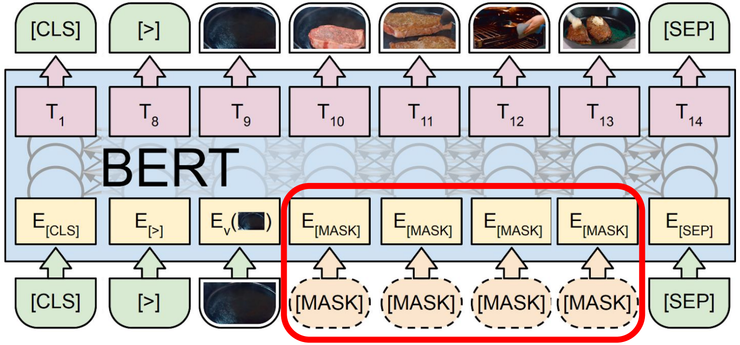




VideoBERT: Downstream Tasks

- Future frame prediction
  - Providing one (or a few more) input video token(s) and add some more masked future tokens.

Input: Possible futures: put into an oven, turned into brownie or cupcake...

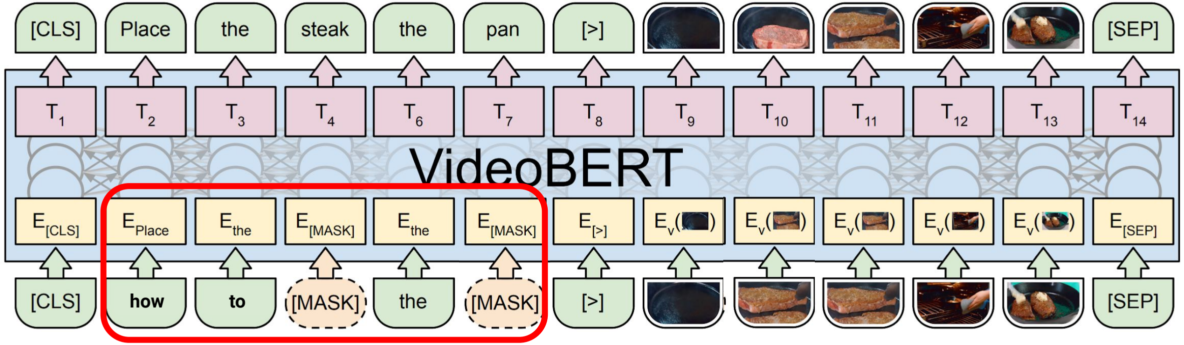


VideoBERT: Downstream Tasks

- Zero-shot action classification
  - Force VideoBERT to “speak” by providing a template sentence but mask out verbs and nouns: “Now let me show you how to [MASK] the [MASK].”

Top verbs: make, assemble, prepare
Top nouns: pizza, sauce, pasta

Top verbs: make, do, pour
Top nouns: cocktail, drink, glass



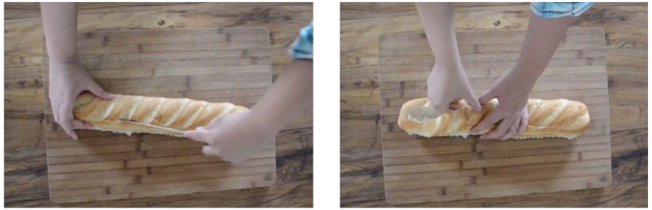
VideoBERT: Downstream Tasks

- Video captioning
  - Mask all the words in the text part. The model will generate a sentence based on the visual signal.

GT: add some chopped basil leaves into it

VideoBERT: chop the basil and add to the bowl

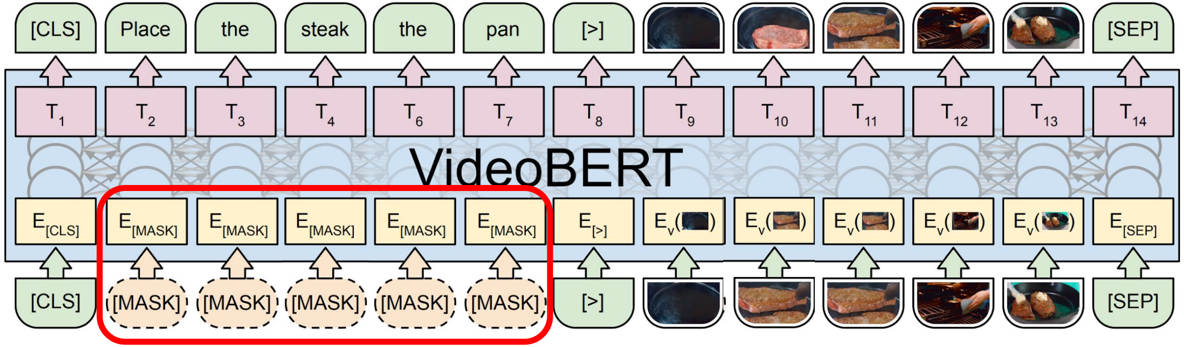
S3D: cut the tomatoes into thin slices



GT: cut the top off of a french loaf

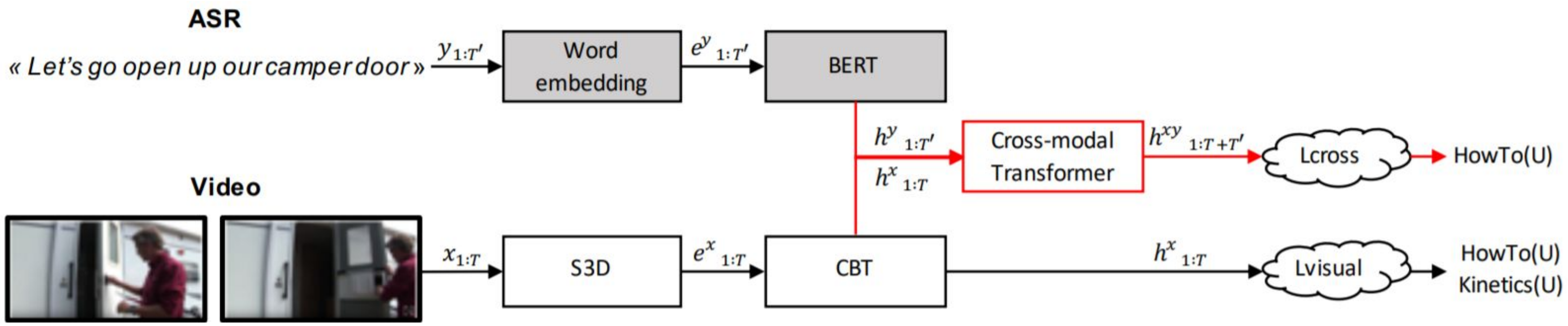
VideoBERT: cut the bread into thin slices

S3D: place the bread on the pan



CBT

- **C**ontrastive **B**idirectional **T**ransformer
- Visual and text towers are trained **separately with unimodal BERTs**, followed by **cross-modal Transformer** to learn multimodal correspondence.
  - *c.f.*, VilBERT mixed cross-modal and unimodal attention repeatedly.
- https://arxiv.org/pdf/1906.05743.pdf

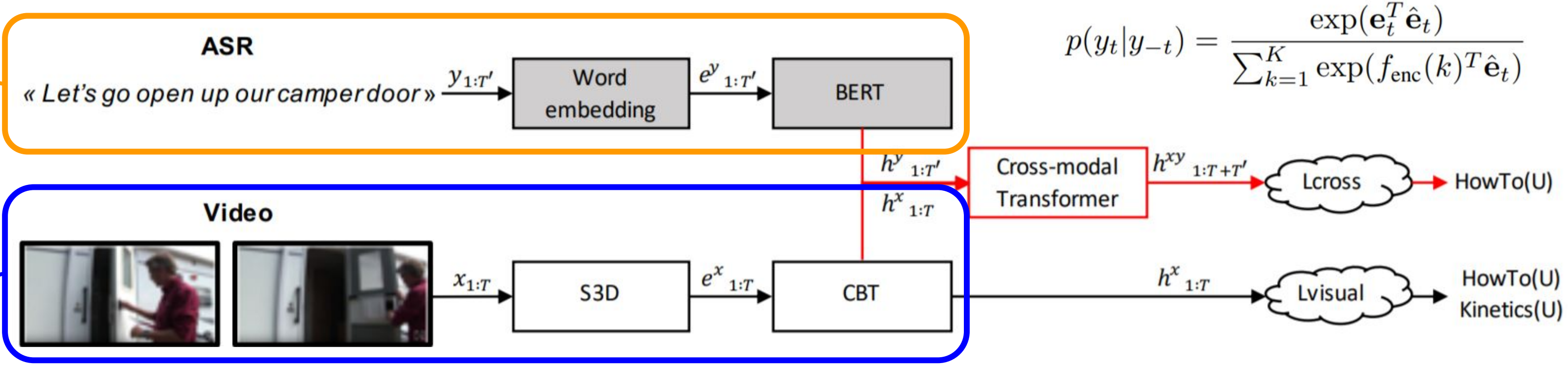


CBT: Model Structure

The text tower is same as original BERT.

$$L_{\text{bert}} = -E_{\mathbf{y}\sim\mathcal{D}} \sum_{t=1}^{T} \log p(y_t | y_{-t})$$

$$p(y_t|y_{-t}) = \frac{\exp(\mathbf{e}_t^T \hat{\mathbf{e}}_t)}{\sum_{k=1}^{K} \exp(f_{\text{enc}}(k)^T \hat{\mathbf{e}}_t)}$$



Sampled frames are first encoded by S3D, then goes through a Transformer called CBT. As image is a real-valued vector (not a token), **contrastive learning** is adopted.

$$L_{\text{visual}} = -E_{\mathbf{x}\sim\mathcal{D}} \sum_{t} \log \text{NCE}(\mathbf{x}_t | \mathbf{x}_{-t})$$

$$\text{NCE}(\mathbf{x}_t|\mathbf{x}_{-t}) = \frac{\exp(\mathbf{e}_t^T \hat{\mathbf{e}}_t)}{\exp(\mathbf{e}_t^T \hat{\mathbf{e}}_t) + \sum_{j\in\text{neg}(t)} \exp(\mathbf{e}_j^T \hat{\mathbf{e}}_t)}$$

Positive: frames from same video

Negative: frames from other videos in the same minibatch

CBT: Loss Functions

BERT (Softmax): encourages the model to learn to identify the correct token (given the context) compared to all vocabs.

BERT input: Actual embedding for word *t*

BERT output: Predicted embedding for word *t*

$$p(y_t|y_{-t}) = \frac{\exp(\mathbf{e}_t^T \hat{\mathbf{e}}_t)}{\boxed{\sum_{k=1}^{K} \exp(f_{\text{enc}}(k)^T \hat{\mathbf{e}}_t)}}$$

Actual embedding for word *k*

Divided by sum of scores for all vocabs

CBT (NCE): encourages the model to learn to identify the correct frame (given the context) compared to a set of negative distractors.

CBT input: S3D embedding for frame *t*

CBT output: Predicted embedding for frame *t*

$$\text{NCE}(\mathbf{x}_t|\mathbf{x}_{-t}) = \frac{\exp(\mathbf{e}_t^T \hat{\mathbf{e}}_t)}{\boxed{\exp(\mathbf{e}_t^T \hat{\mathbf{e}}_t) + \sum_{j \in \text{neg}(t)} \exp(\mathbf{e}_j^T \hat{\mathbf{e}}_t)}}$$

Divided by sum of scores for the positive and some negative samples

CBT: Summary

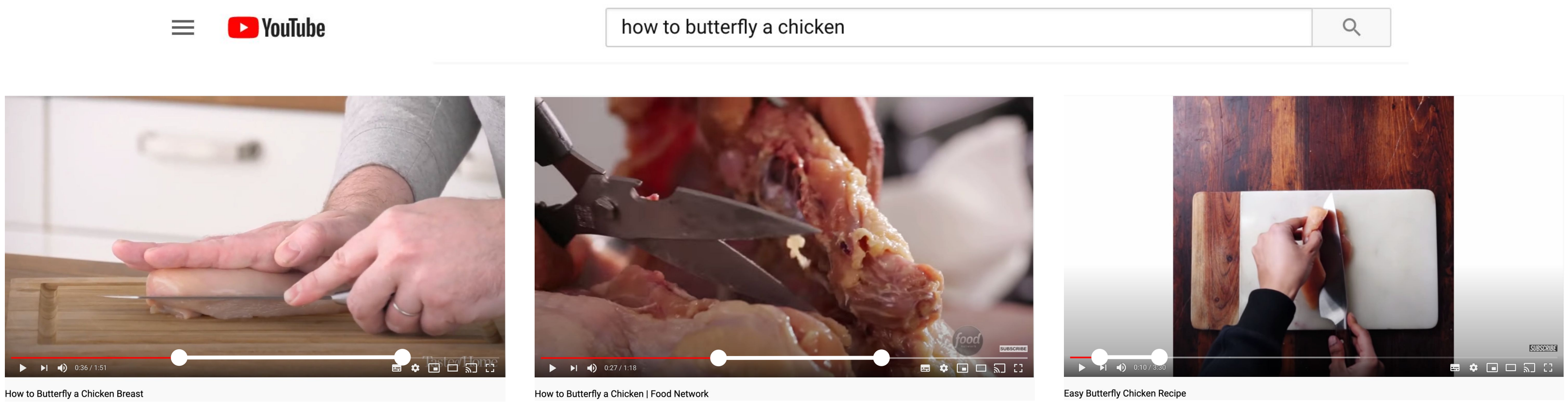
- Given a sequence of frames **x** = $\{x_1, \ldots, x_m\}$ and of ASR tokens **y** = $\{y_1, \ldots, y_n\}$, the Cross-modal Transformer learns their correspondence/relevance/alignment.
  - **x** and **y** are not necessarily aligned at frame/token level.
  - Thus, we try to maximize mutual information (MI) at the **sequence level**.

- **x** and **y** are concatenated and fed into a cross-modal Transformer (*e.g.*, VideoBERT), producing output embedding sequence **h** = $\{h_1, \ldots, h_{m+n}\}$.
  - This Transformer is also trained using the **NCE loss**.
  - Lastly, **h** goes through a shallow MLP to compute correspondence (MI) score.

- The entire model is trained end-to-end, weighted-summing all three losses (BERT, CBT, Cross-modal).
  - End-to-end training was not possible with VideoBERT, due to the frame clustering.
  - Now, with NCE loss, the entire training can be done end-to-end.

Hammer

- Target task: Moment localization in Video Corpus (MLVC)
  - Moment: A short clip (or segment) in a video that contains a semantically meaningful sequence.
  - **Moment Localization in Single Video (MLSV)** task: Given a video, find the **time window of an event** that is described by the given **natural language query**.
  - **Moment Localization in Video Corpus (MLVC)** task: Find a video segment that corresponds to a **text query** from a **corpus** of **untrimmed and unsegmented videos**.
- https://arxiv.org/pdf/2011.09046.pdf



Hammer: Two-Stage Approach

Ranking all moments in all the corpus' videos?
Intractable!

$$p(moment \mid query)$$

$$\Sigma_{\color{red}{top\text{-}k}} \; p(moment \mid video, query) \times p(video \mid query)$$

Moment Localization in Single Video

- Higher Resolution Task
- Apply to *top-k* videos during inference

Video Retrieval

- Lower Resolution Task
- Filters videos during inference

Hammer: Training

- **Video retrieval task**: Contrastive Learning of Video & Text Matching

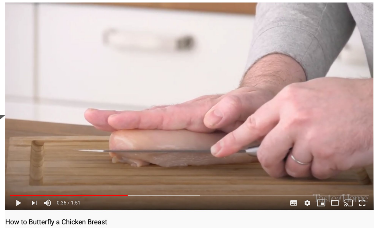
$$Likelihood \propto \log\left[\frac{p(video^{+} \mid query^{+})}{p(video^{+} \mid query^{+}) + p(video^{-} \mid query^{+}) + p(video^{+} \mid query^{-})}\right]$$

butterfly a chicken

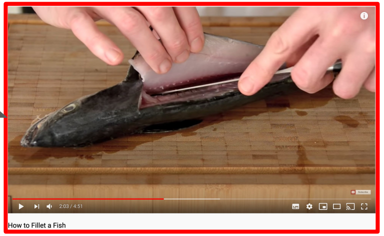
query⁻

video⁻

line the parchment paper

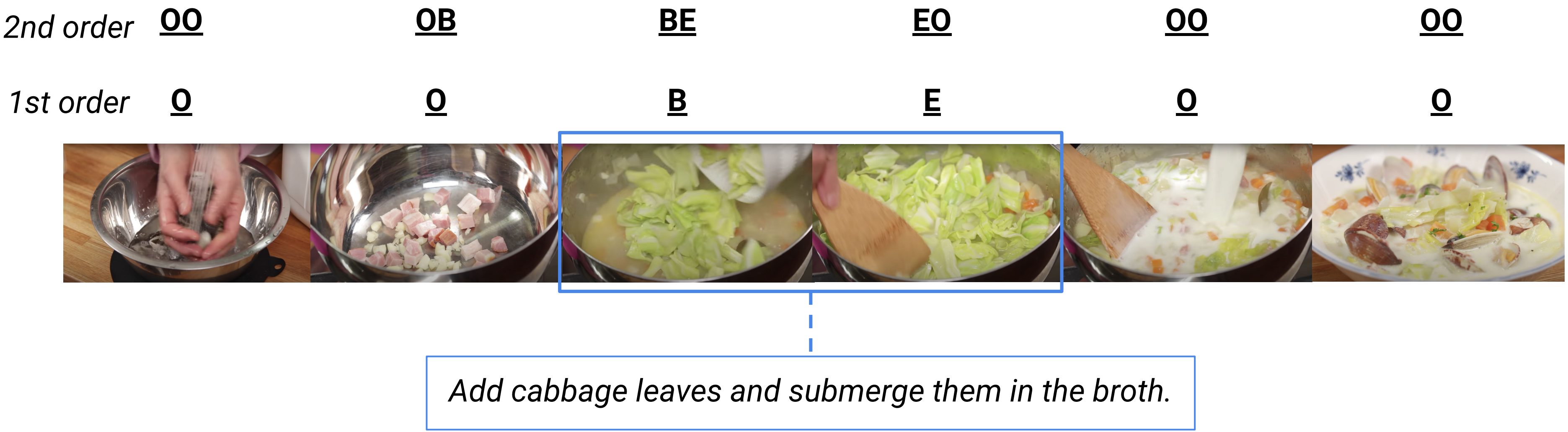


butterfly a chicken



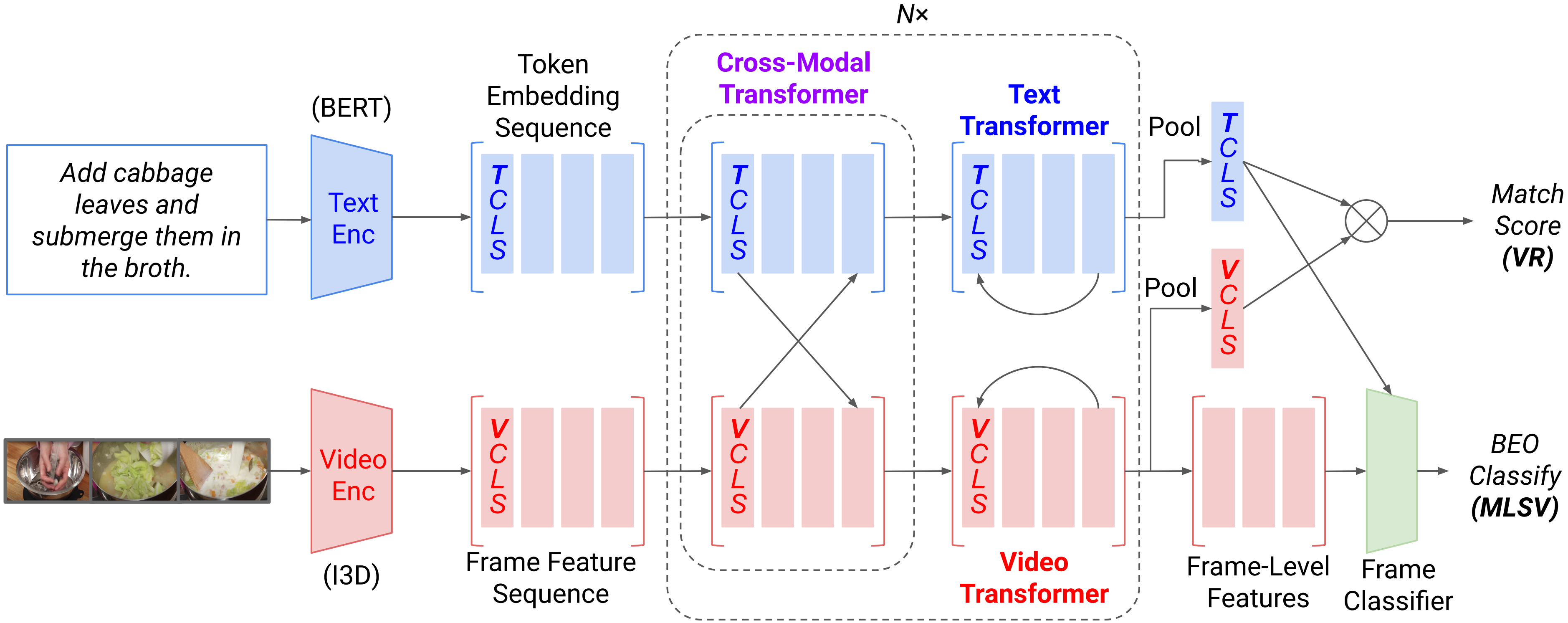
Hammer: Training

- **Temporal localization task**:
    - 3-way classification at frame-level (<u>B</u>egin <u>E</u>nd <u>O</u>ther)
    - Higher-order *n*-grams work slightly better.



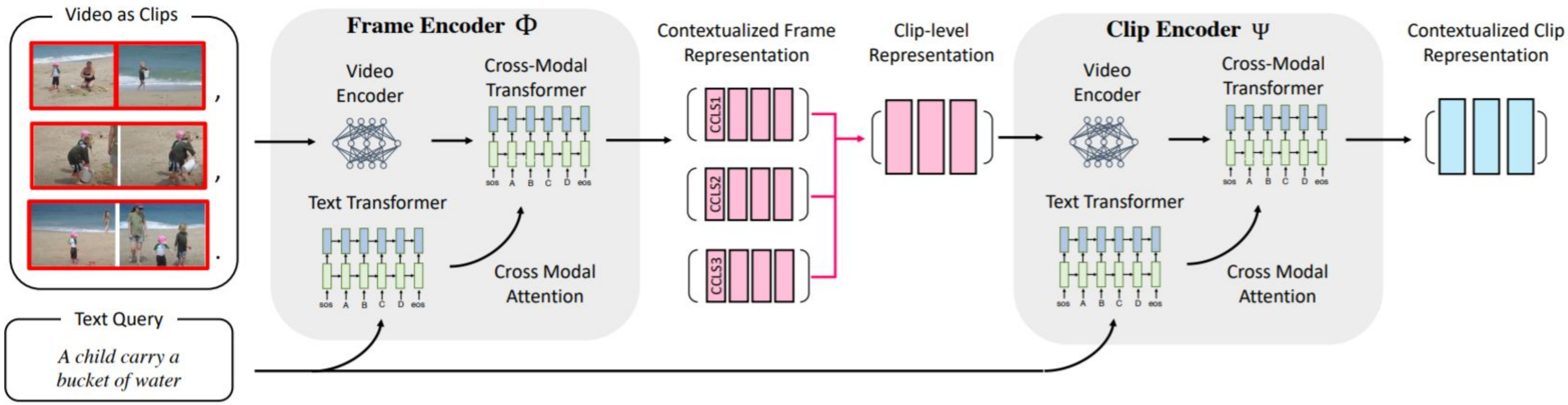
Hammer: Architecture

- Backbone model: Cross-modal Transformer (similar to VilBERT)

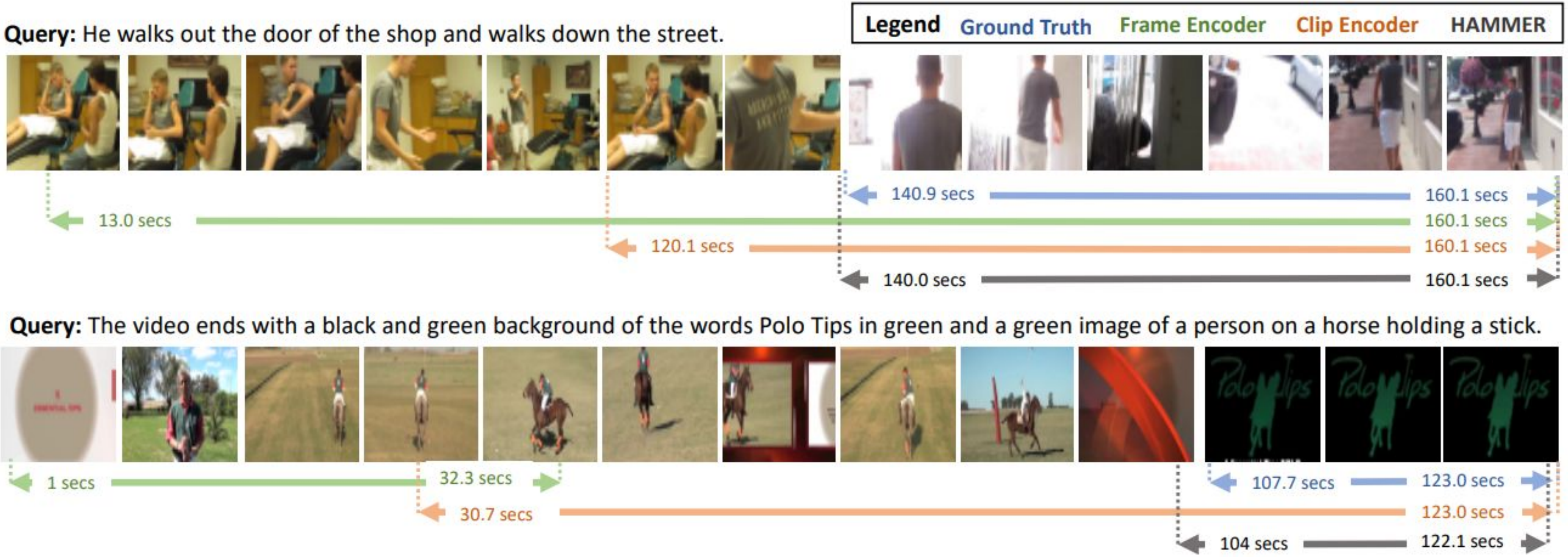


Hammer: Architecture

- Hierarchical visual encoders:
  - Aligning text and video segments requires **fine-grained** spatio-temporal understanding **at different time scales**.
  - **Frame encoder**: sequence of frames + text query → clip representation
  - **Clip encoder**: sequence of clips + text query → video representation
  - Extendable to 3rd or higher levels, if we want to deal with longer videos.



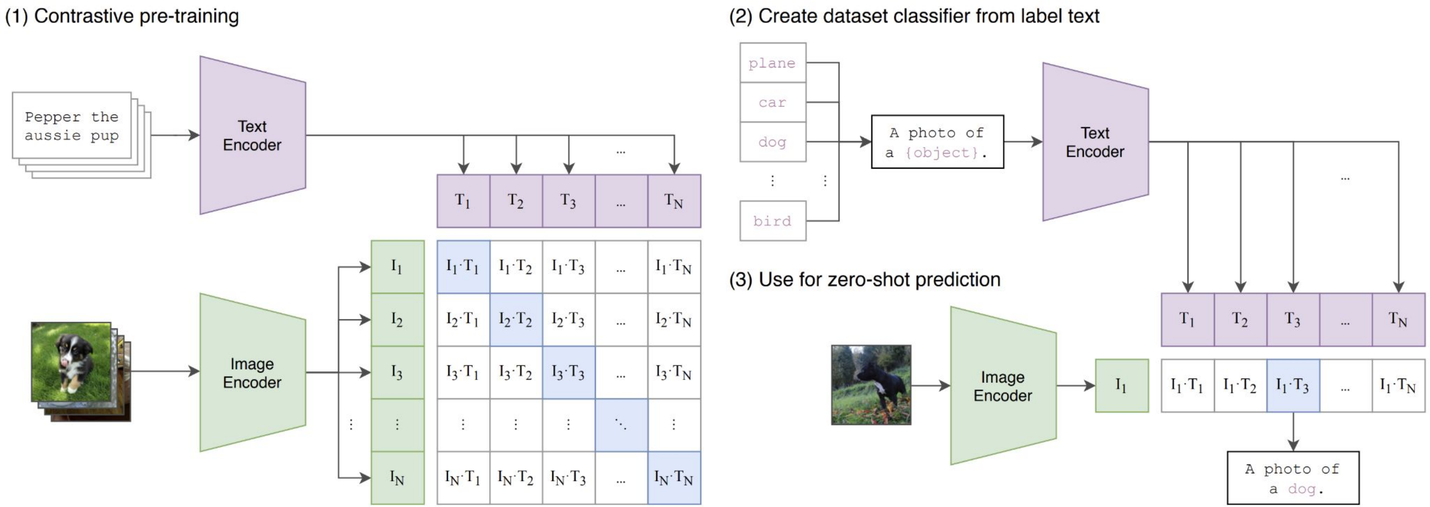
Hammer: Examples



Large-scale Multimodal Pre-training

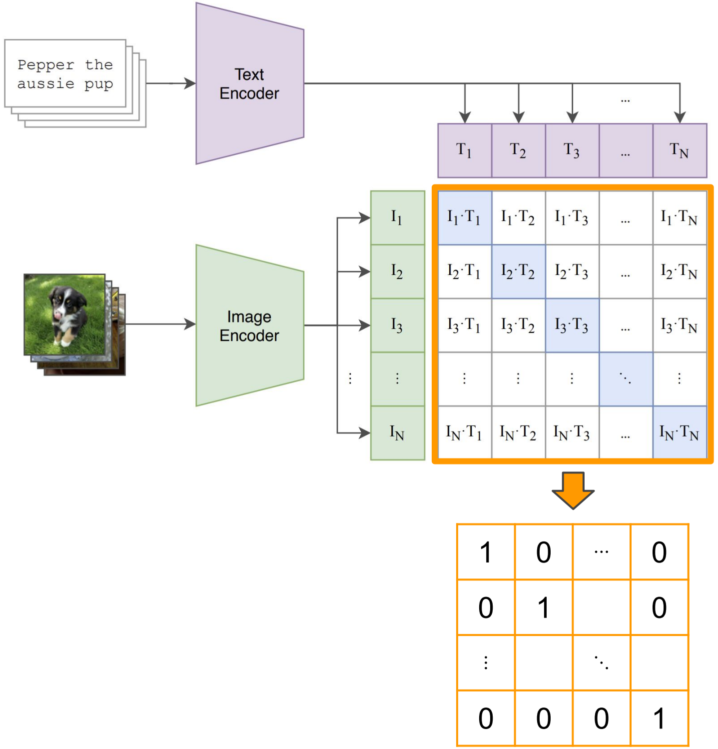
CLIP

- “A multimodal metric learning using large-scale paired dataset”
  - At training: Jointly trains an image encoder and a text encoder to predict the correct pairings of a batch of (image, text) training examples.
  - At testing: the learned text encoder synthesizes a zero-shot linear classifier by embedding the names or descriptions of the target dataset’s classes.
- https://arxiv.org/pdf/2103.00020.pdf



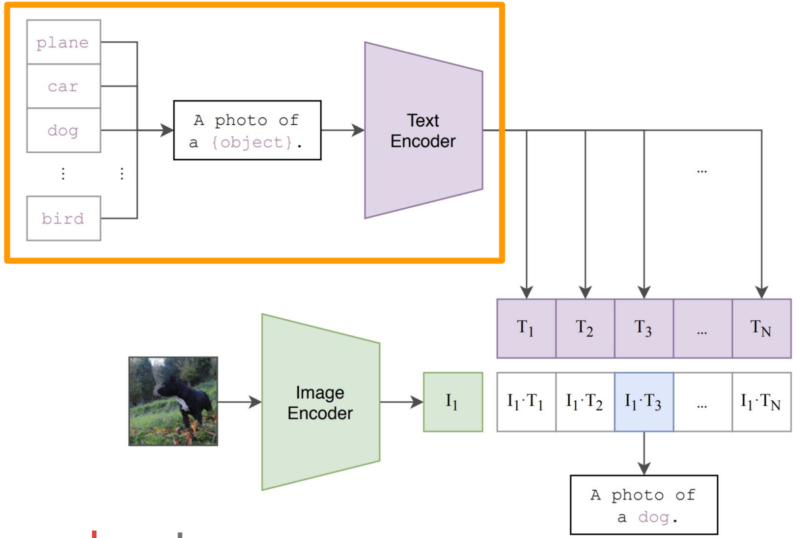
CLIP: Training

- Given *N* (batch size) image-text pairs, the training procedure
  - **Maximizes** similarity (dot-product) between the true pairs $(I_1, T_1), (I_2, T_2), \ldots, (I_N, T_N)$,
  - while **minimizes** similarity (dot-product) between all other pairs $(I_1, T_2), (I_2, T_1), \ldots$ in the batch.
- Mathematically, identical to making the outer-product of image and text matrices closer to an identity matrix.
- Same as the contrastive learning, or InfoNCE loss we learned in the last lecture.



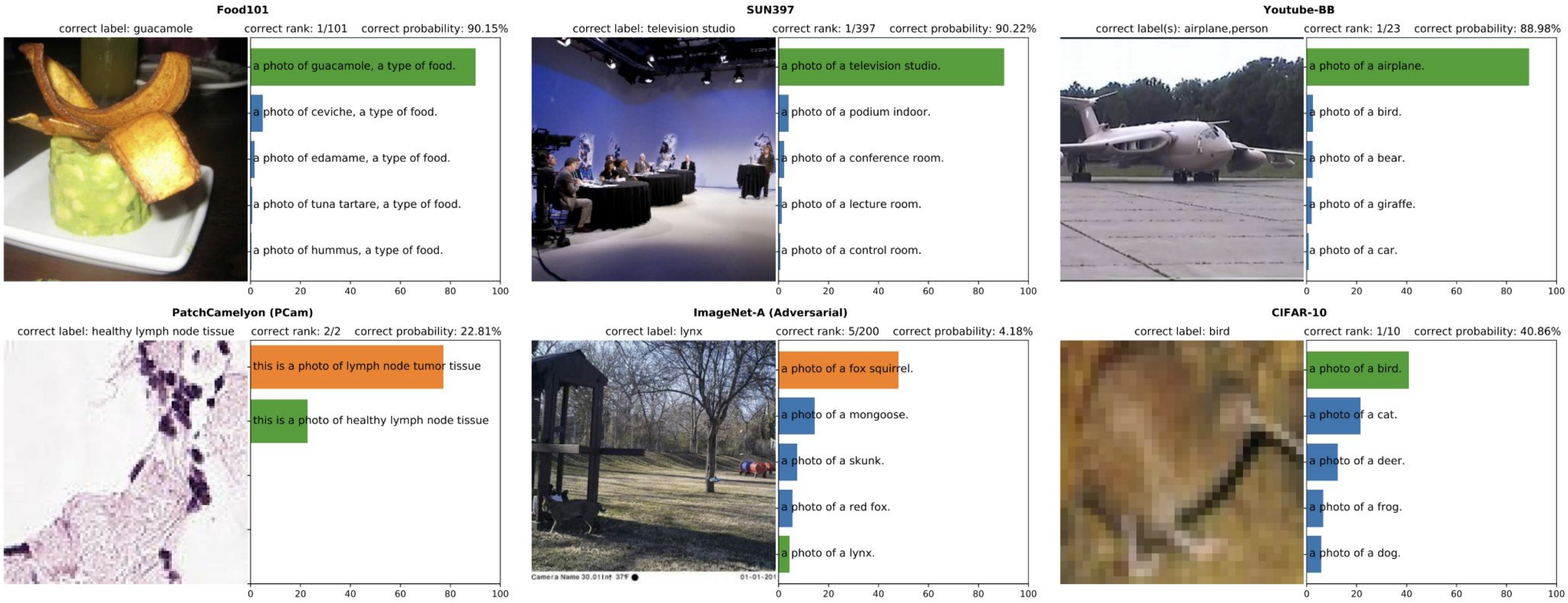
CLIP: Inference

- To make this as a classifier, we use a text prompt: “A photo of a ________”
  - Because the model is knowledgeable of natural language (stronger than class terms), it can easily adapt to such a prompt.
  - The image encoding is also already aligned with the semantics represented by the text, so an inner-product with the image and corresponding prompt will be larger.



- The text and image encoders are useful themselves!
  - Image embedding is semantically powered by language pairs.
  - Text embedding is also powered by visual cues.
  - Common use cases:
    - Embed a text, then retrieve the closest *k* images / videos.
    - Embed an image, then select / generate a sentence describing it.

CLIP: Text Retrieval Examples

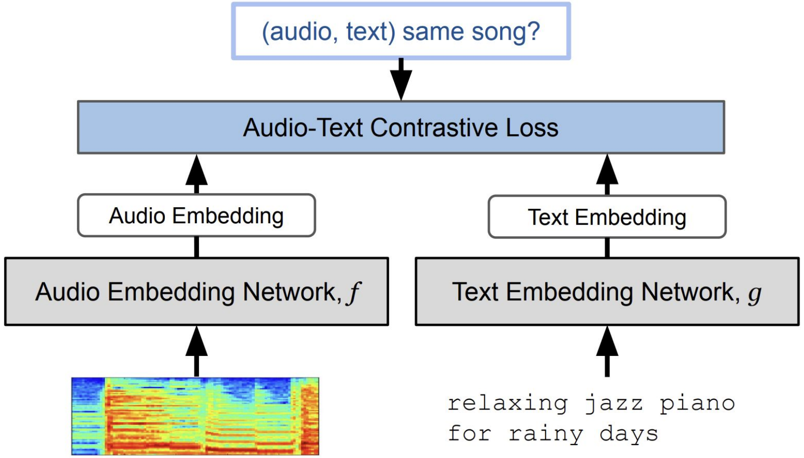


MuLan

- **Music-Language** matching: Audio / Music version of CLIP
- Training: a batch-wise contrastive learning similar to CLIP

$$\sum_{i=1}^{B} -\log\left[\frac{h[f(\mathbf{x}^{(i)}), g(\mathbf{t}^{(i)})]}{\sum_{j\neq i} h[f(\mathbf{x}^{(i)}), g(\mathbf{t}^{(j)})] + h[f(\mathbf{x}^{(j)}), g(\mathbf{t}^{(i)})]}\right]$$

  - Text is collected from a webpage containing music content.
  - E.g., music title, description, general web pages, …



https://arxiv.org/pdf/2208.12415.pdf