Vision, Language and Reading

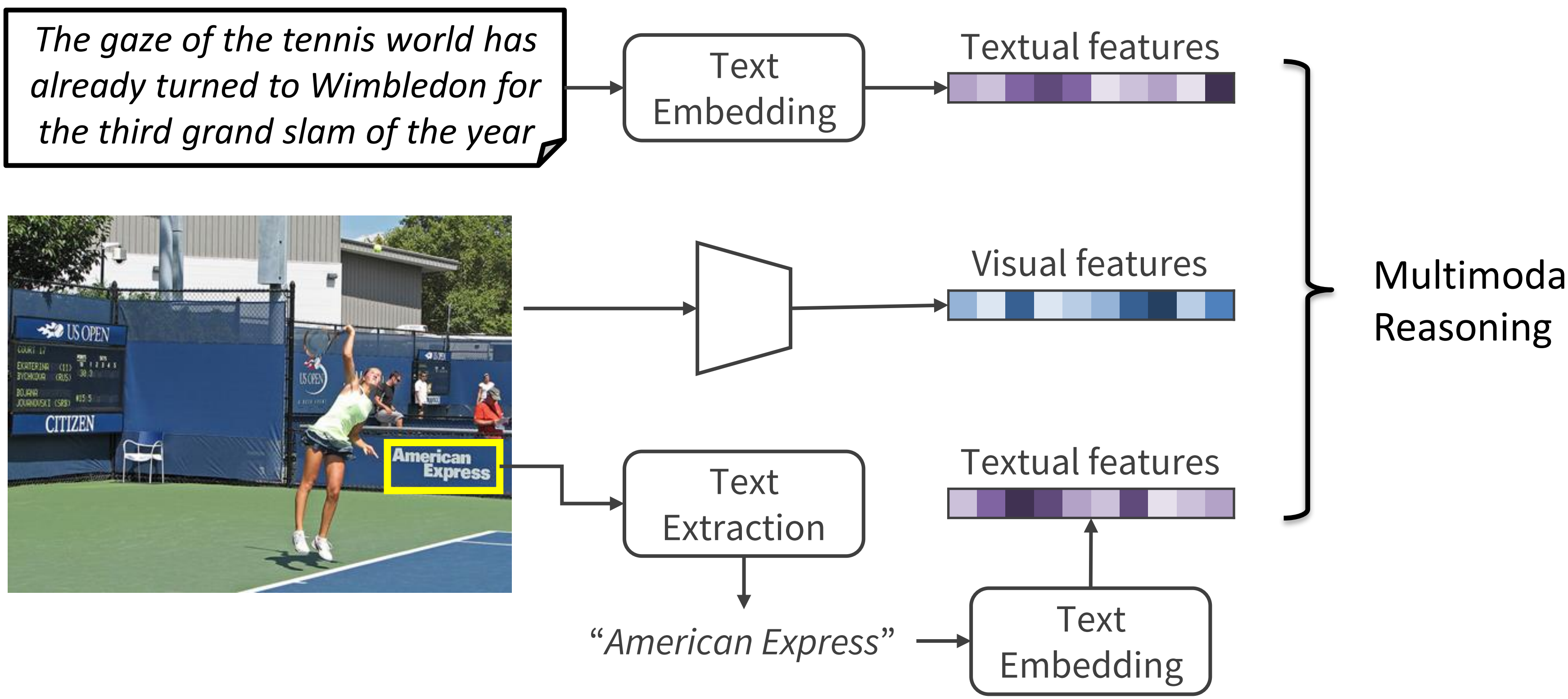
Text Embeddings,
Multimodal Learning

Dimosthenis Karatzas (dimos@cvc.uab.es), Ernest Valveny (ernest@cvc.uab.es)

TEXT EMBEDDINGS

Text Embeddings

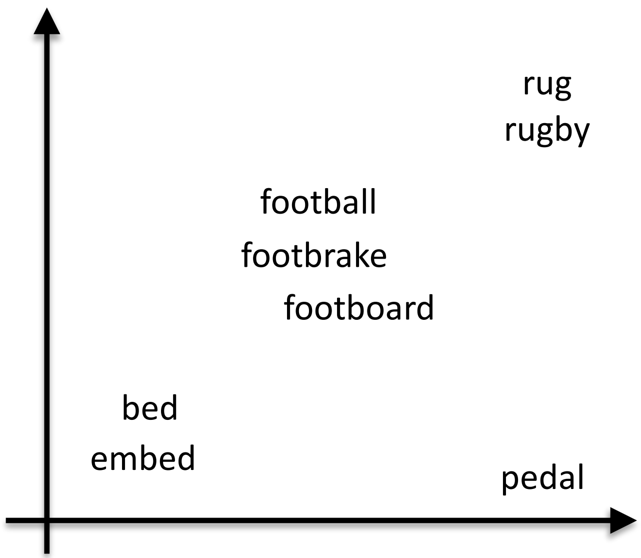
Goal: convert text into a vectorial representation that can be used in further reasoning tasks

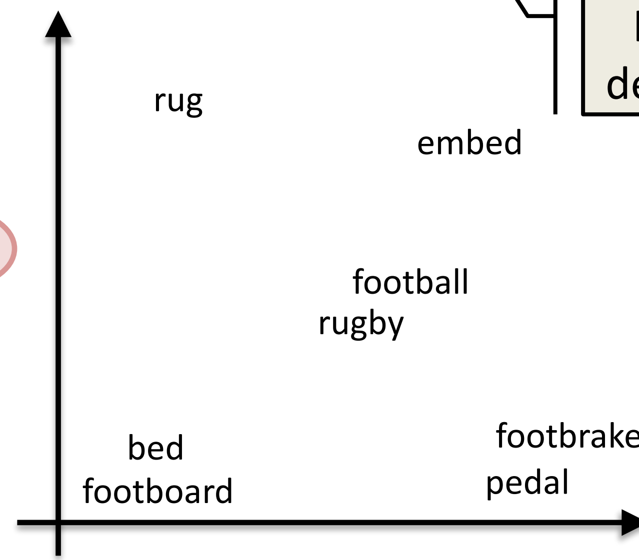


Text Embeddings

Key idea: **similar** words (documents) are projected into **similar** vectorial representations

How do you define *similar*?





Lexical embeddings:	PHOC
Semantic embeddings:	Topic models, word2vec, GloVe, FastText
Contextual embeddings:	BERT, GPT, T5
Multimodal embeddings:	VisualBERT, VilBERT

One-hot encoding

The simplest embedding would be to assign each word to a different one-hot vector

$$\text{The} \rightarrow \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad \text{food} \rightarrow \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad \text{was} \rightarrow \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \qquad \text{amazing} \rightarrow \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Very simple representation. No need for training

Does not scale well: the size of the vectors is as big as the vocabulary

Fixed vocabulary: does not allow to represent out-of-vocabulary words

Adding and removing words changes all representations

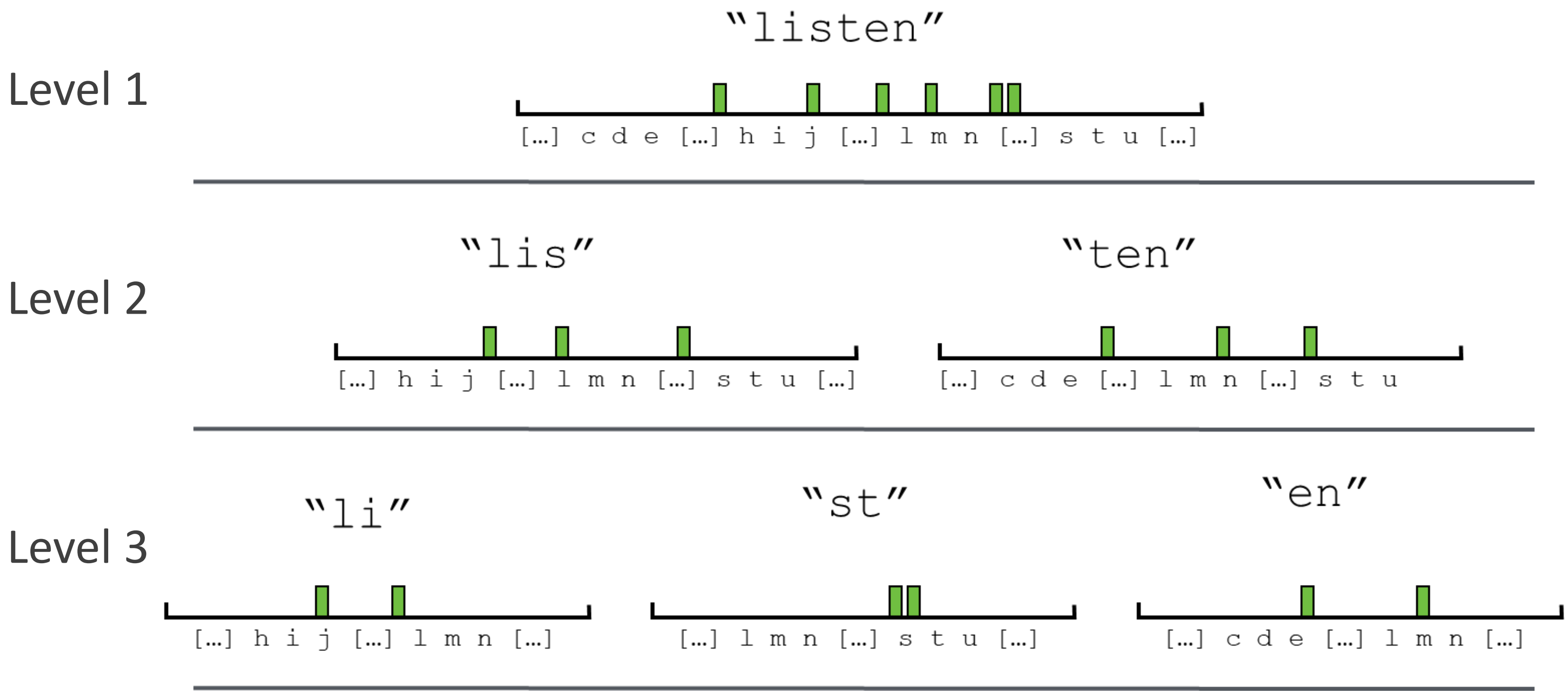
Does not encode any **lexical** or **semantic information**

Text Embeddings

LEXICAL EMBEDDINGS

PHOC
(Pyramidal Histogram of Characters)

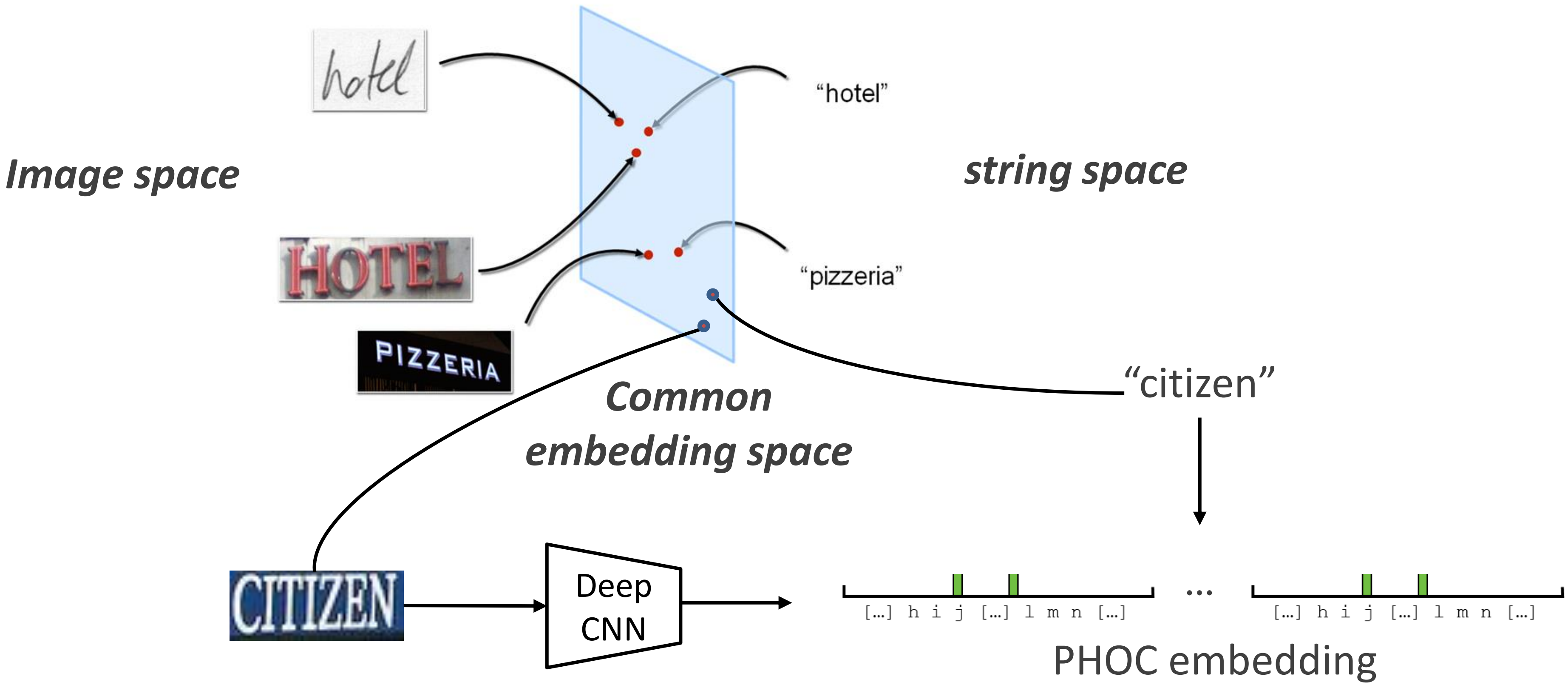
Concatenation of histograms of characters at multiple levels of decomposition



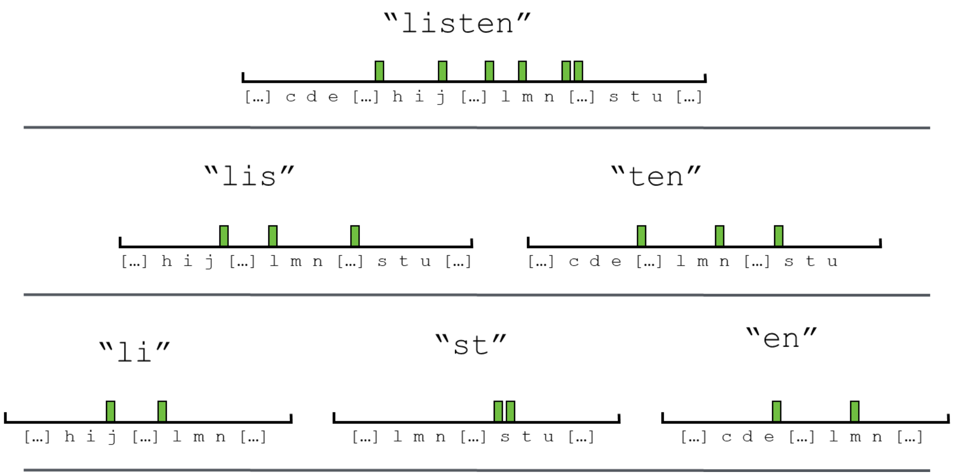
J. Almazán, A. Gordo, A. Fornés, E. Valveny: **Word Spotting and Recognition with Embedded Attributes.** IEEE Transactions on Pattern Analysis and Machine Intelligence, 2014

Word Spotting with PHOC

- The PHOC embedding is directly related with the spatial arrangement of characters in a word image
- We can train a network to predict the PHOC embedding directly from the image without explicitly recognizing the word
- PHOC defines a common embedding space between image and text spaces



PHOC
(Pyramidal Histogram of Characters)



Very simple representation. No need for training in the string space

Low dimensionality, the size of the vectors does not depend on the vocabulary

Allows for out-of-vocabulary words

Can be directly obtained from the word image without explicit recognition

Does not encode any semantic information

Text Embeddings

SEMANTIC EMBEDDINGS

Topic Models

Idea: documents that contain the same words should speak about the same thing, and vice versa, words that appear in the same documents should be related

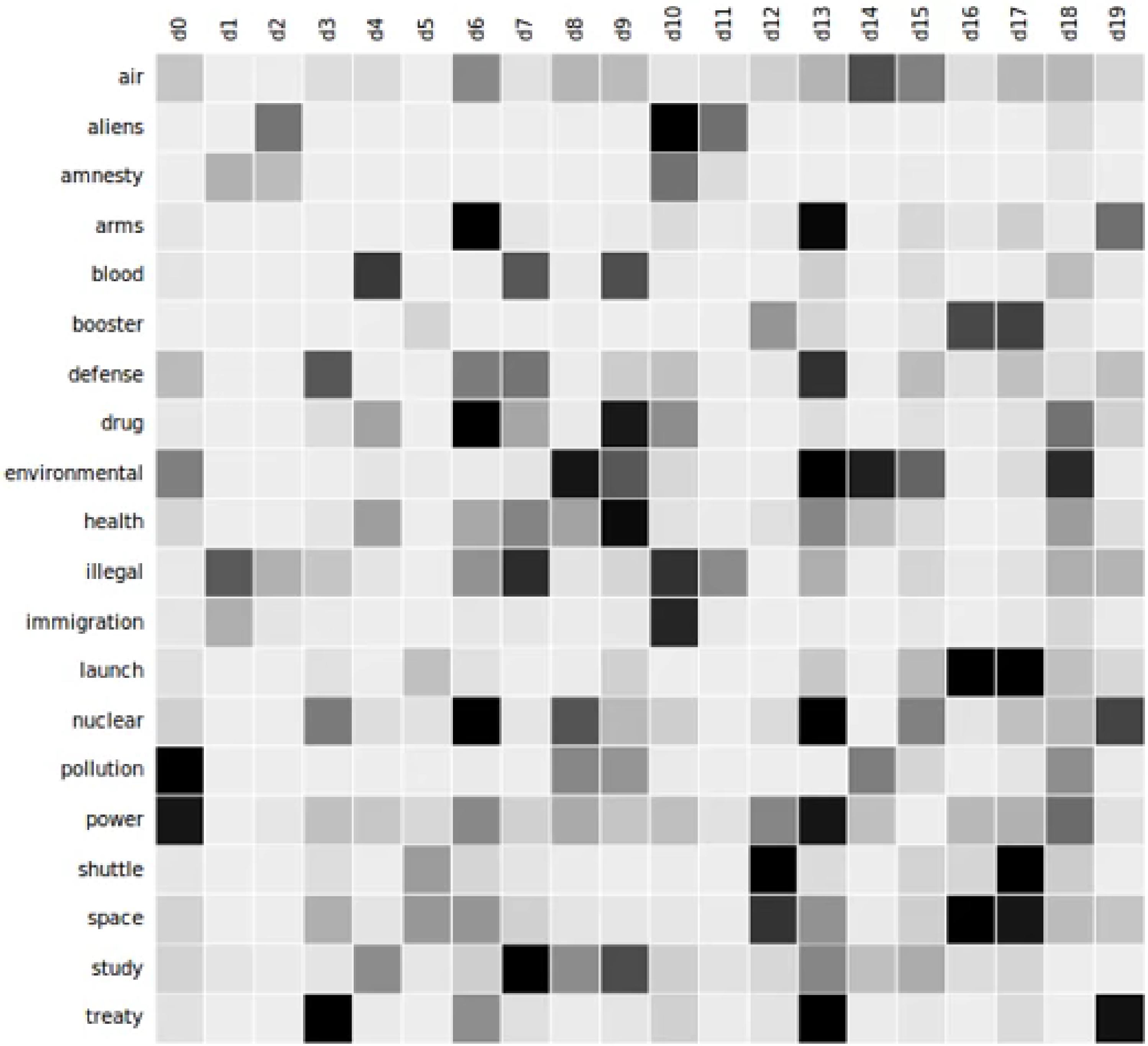
- A word is “similar” to another word if they frequently appear in the same documents (e.g. usually when we speak about “elephants” we also speak about “jungle”)
- A document is “similar” to another document if they have a lot of words in common (e.g. two documents that contain the words: “account”, “transaction”, “interest”, probably both come from your bank)

Topics define **probability distributions** over the words in our vocabulary

A document can be seen as **a mixture of a small number of topics**, while a word has a different probability of having been created by each topic.

Topics are seen as **latent (hidden) variables** that govern the generation of language

A lot of ways to discover topics (latent variables), by exploiting the above observation: pLSA, LDA (Latent Dirichlet Allocation), …


d0 d1 d2 d3 d4 d5 d6 d7 d8 d9 d10 d11 d12 d13 d14 d15 d16 d17 d18 d19
air
aliens
amnesty
arms
blood
booster
defense
drug
environmental
health
illegal
immigration
launch
nuclear
pollution
power
shuttle
space
study
treaty

Word2Vec

Semantically similar words should have similar word embeddings

Which words are semantically similar?

Word2Vec idea: if two words share the same context (they are always found close to the same words) they should end up having similar word embeddings.

I	wore	my	blue	trousers

Word2Vec

Semantically similar words should have similar word embeddings

Which words are semantically similar?

Word2Vec idea: if two words share the same context (they are always found close to the same words) they should end up having similar embeddings.

I | wore | my | blue

trousers
jeans
T-shirt
leggings
shoes
…

Word2Vec

Semantically similar words should have similar word embeddings

Which words are semantically similar?

Word2Vec idea: if two words share the same context (they are always found close to the same words) they should end up having similar embeddings.

I	wore	my	blue red tight new …	trousers

Word2Vec

Semantically similar words should have similar word embeddings

Which words are semantically similar?

Word2Vec idea: if two words share the same context (they are always found close to the same words) they should end up having similar embeddings.

I | wore / put on / removed / ripped / washed / … | my | blue | trousers

Word2Vec

First, encode everything as a 1-hot vector.

Then set a window-size to define how much “context” we will take into account

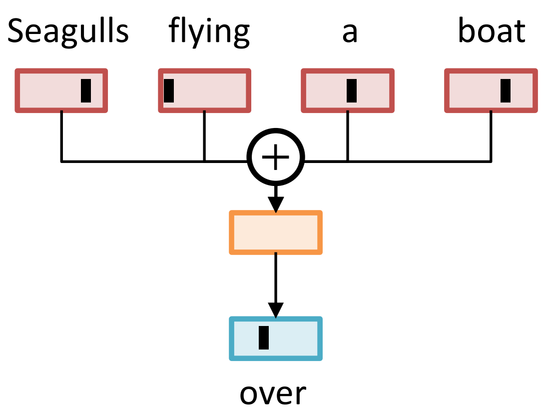
Two ways to learn embeddings by correlating the central word with its context

	Center Word	Context Words
Seagulls flying over a boat in the harbor	[1, 0, 0, 0, 0, 0, 0, 0, ...]	[0, 1, 0, 0, 0, 0, 0, 0, ...] [0, 0, 1, 0, 0, 0, 0, 0, ...]
Seagulls flying over a boat in the harbor	[0, 1, 0, 0, 0, 0, 0, 0, ...]	[1, 0, 0, 0, 0, 0, 0, 0, ...] [0, 0, 1, 0, 0, 0, 0, 0, ...] [0, 0, 0, 1, 0, 0, 0, 0, ...]
Seagulls flying over a boat in the harbor	[0, 0, 1, 0, 0, 0, 0, 0, ...]	[1, 0, 0, 0, 0, 0, 0, 0, ...] [0, 1, 0, 0, 0, 0, 0, 0, ...] [0, 0, 0, 1, 0, 0, 0, 0, ...] [0, 0, 0, 0, 1, 0, 0, 0, ...]
Seagulls flying over a boat in the harbor	[0, 0, 0, 1, 0, 0, 0, 0, ...]	[0, 1, 0, 0, 0, 0, 0, 0, ...] [0, 0, 1, 0, 0, 0, 0, 0, ...] [0, 0, 0, 0, 1, 0, 0, 0, ...] [0, 0, 0, 0, 0, 1, 0, 0, ...]
Seagulls flying over a boat in the harbor	[0, 0, 0, 0, 1, 0, 0, 0, ...]	[0, 0, 1, 0, 0, 0, 0, 0, ...] [0, 0, 0, 1, 0, 0, 0, 0, ...] [0, 0, 0, 0, 0, 1, 0, 0, ...] [0, 0, 0, 0, 0, 0, 1, 0, ...]
Seagulls flying over a boat in the harbor	[0, 0, 0, 0, 0, 1, 0, 0, ...]	[0, 0, 0, 1, 0, 0, 0, 0, ...] [0, 0, 0, 0, 1, 0, 0, 0, ...] [0, 0, 0, 0, 0, 0, 1, 0, ...] [0, 0, 0, 0, 0, 0, 0, 1, ...]
Seagulls flying over a boat in the harbor	[0, 0, 0, 0, 0, 0, 1, 0, ...]	[0, 0, 0, 0, 1, 0, 0, 0, ...] [0, 0, 0, 0, 0, 1, 0, 0, ...] [0, 0, 0, 0, 0, 0, 0, 1, ...]
Seagulls flying over a boat in the harbor	[0, 0, 0, 0, 0, 0, 0, 1, ...]	[0, 0, 0, 0, 0, 1, 0, 0, ...] [0, 0, 0, 0, 0, 0, 1, 0, ...]

CBOW vs Skip-gram

CBOW

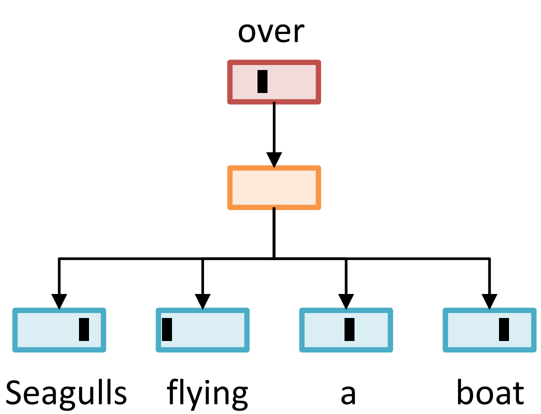
In CBOW, we learn how to **the central word, given the context** reproduce



- trains faster
- better represents more frequent words

Skip-gram

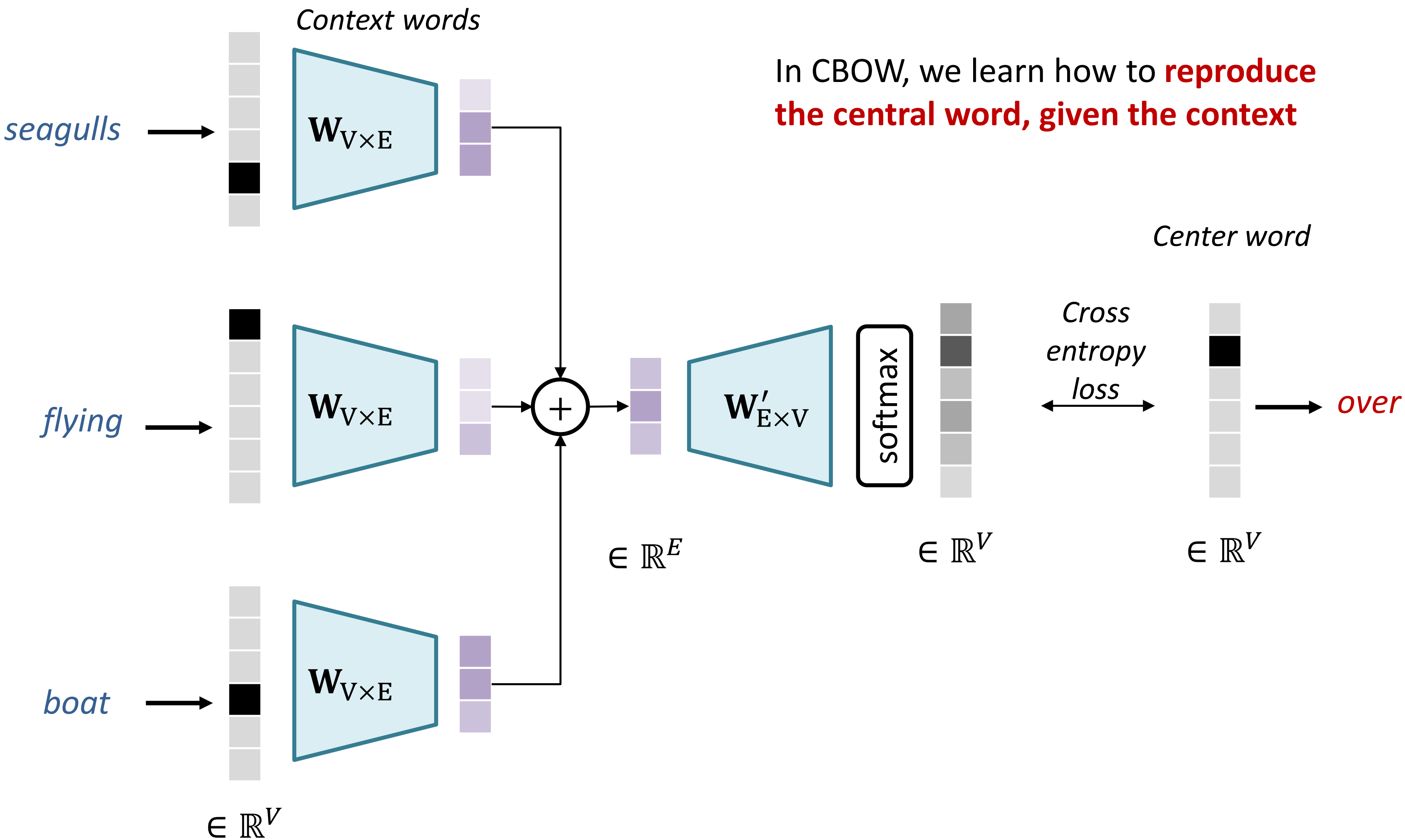
In skip-gram, we learn how to **reproduce the context, given the central word**



- works well with small datasets
- better represents less frequent words

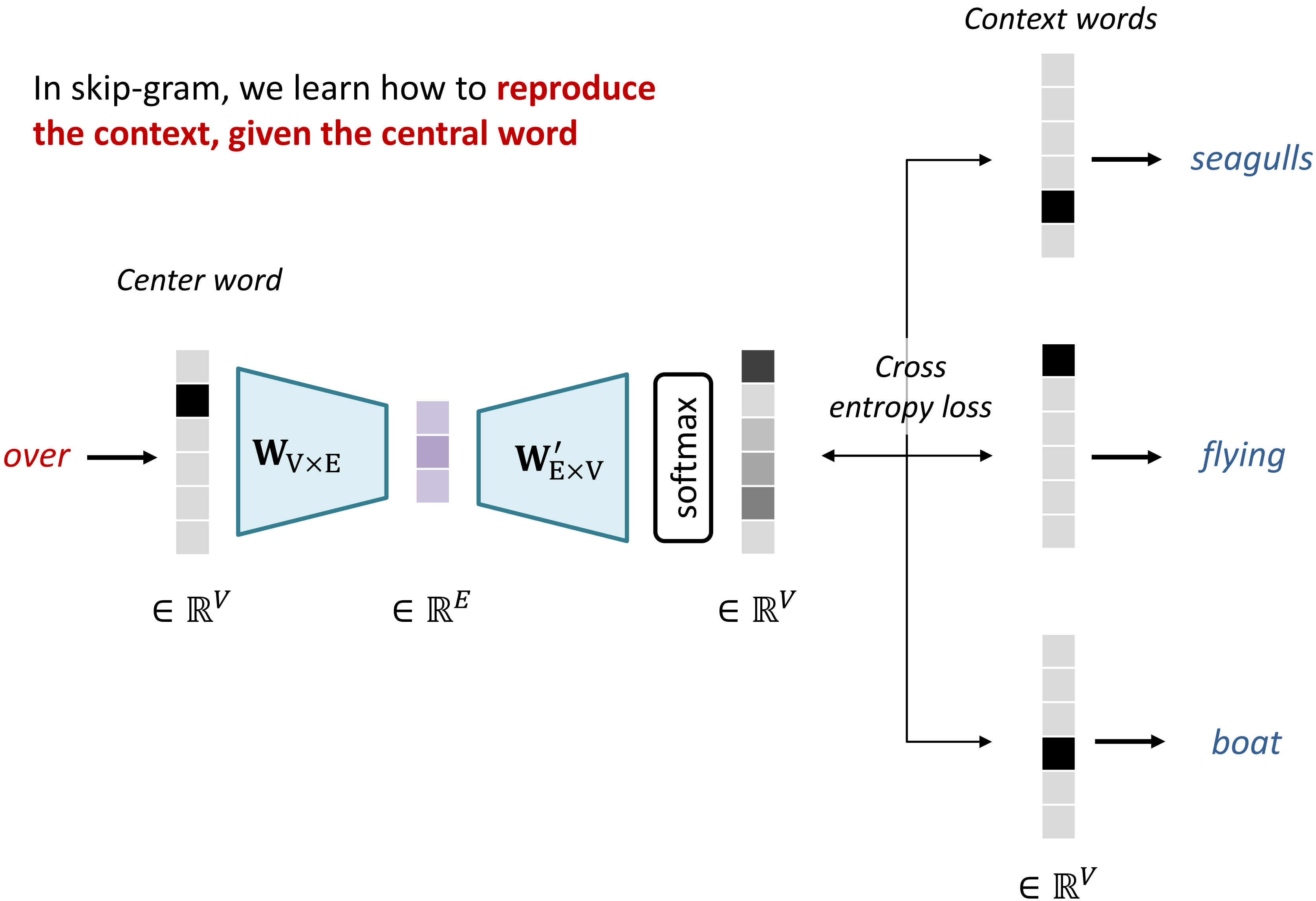
CBOW - Continuous Bag of Words

In CBOW, we learn how to **reproduce the central word, given the context**



Skip-gram

In skip-gram, we learn how to **reproduce the context, given the central word**



GloVe (Global Vectors)

Key idea: the ratios of word-word **co-occurrence** probabilities have the potential to encode some semantics

Probability and Ratio	$k = solid$	$k = gas$	$k = water$	$k = fashion$
$P(k\|ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k\|steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k\|ice)/P(k\|steam)$	8.9	8.5×10^{-2}	1.36	0.96

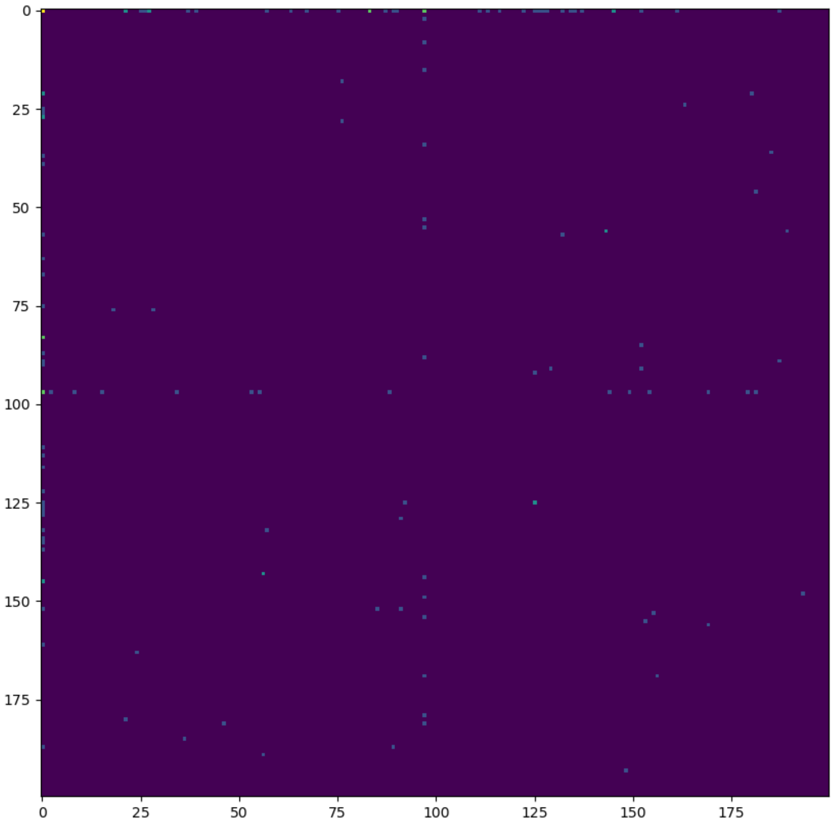
Very small or large:
"*solid*" is related to "*ice*" but not to "*steam*"
"*gas*" is related to "*steam*" but not to "*ice*"

Close to 1:
"*water*" is equally (highly) related to "*ice*" and "*steam*"
"*fashion*" is equally (low) related to "*ice*" and "*steam*"

GloVe (Global Vectors)

The GloVe model is trained on the non-zero entries of a **global** word-word co-occurrence matrix

	quijote	rocinante	sancho	panza	dulcinea	maese	nicolás	ventero	barbero	sabio
a	8.0	4.0	2.0	2.0	3.0	0.0	0.0	1.0	0.0	0.0
de	22.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0
del	0.0	0.0	1.0	1.0	3.0	0.0	1.0	2.0	1.0	0.0
don	40.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
el	2.0	1.0	0.0	0.0	0.0	0.0	0.0	11.0	0.0	1.0
hidalgo	4.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
la	16.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0
panza	0.0	1.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
que	9.0	3.0	1.0	1.0	0.0	0.0	0.0	4.0	0.0	2.0
sancho	0.0	1.0	0.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0
soy	1.0	3.0	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0
y	5.0	4.0	1.0	1.0	0.0	0.0	0.0	4.0	0.0	0.0



word co-occurrence matrix extracted from a portion of Quijote

GloVe (Global Vectors)

The training objective of GloVe is to learn word vectors such that their **dot product** equals the **logarithm** of the words' probability of co-occurrence.

Probability and Ratio	$k = solid$	$k = gas$	$k = water$	$k = fashion$
$P(k\|ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k\|steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k\|ice)/P(k\|steam)$	8.9	8.5×10^{-2}	1.36	0.96

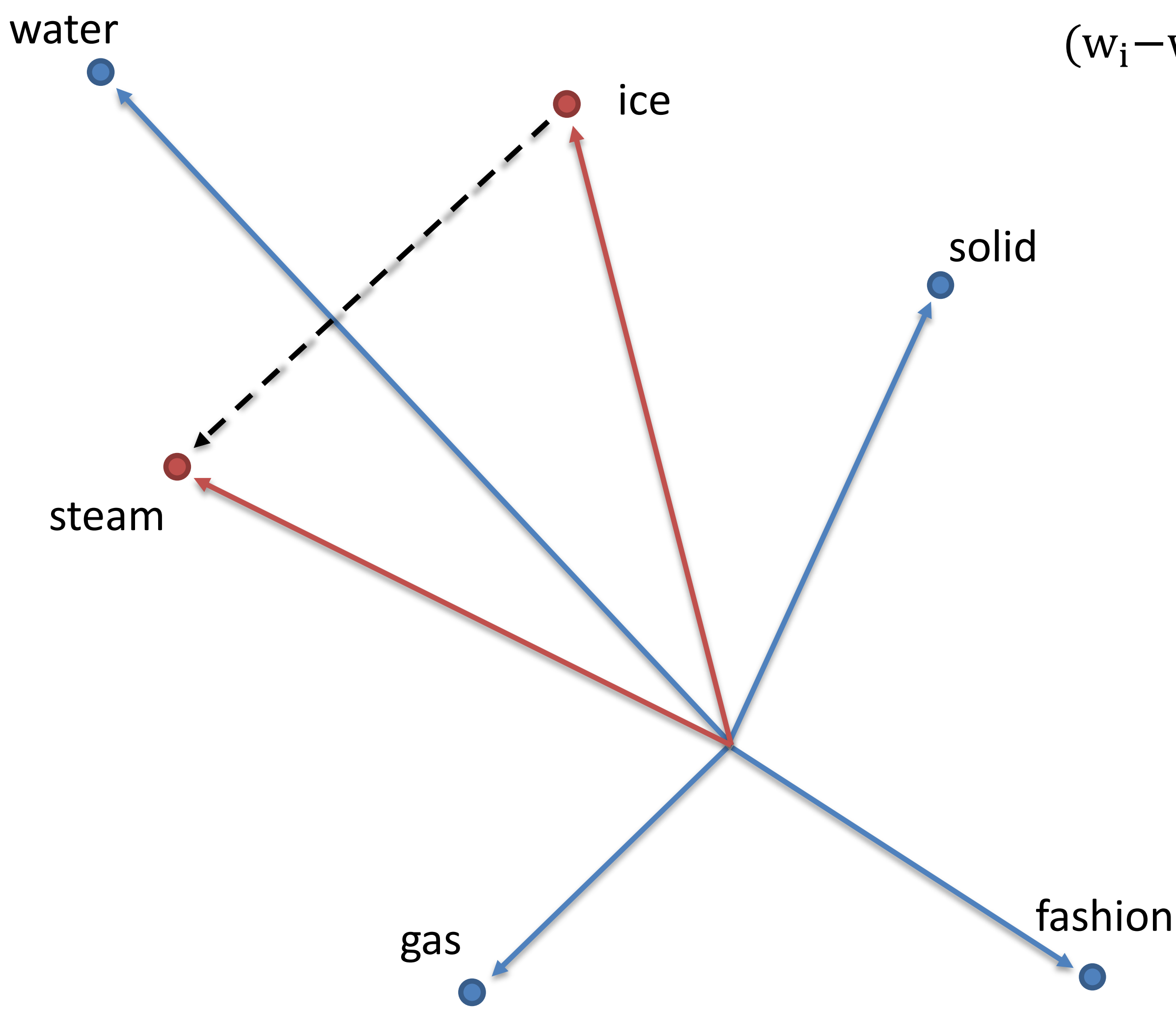
$$w_i^T w_k \approx \log p(k|i)$$

$$w_i^T w_k - w_j^T w_k = \log p(k|i) - \log p(k|j)$$

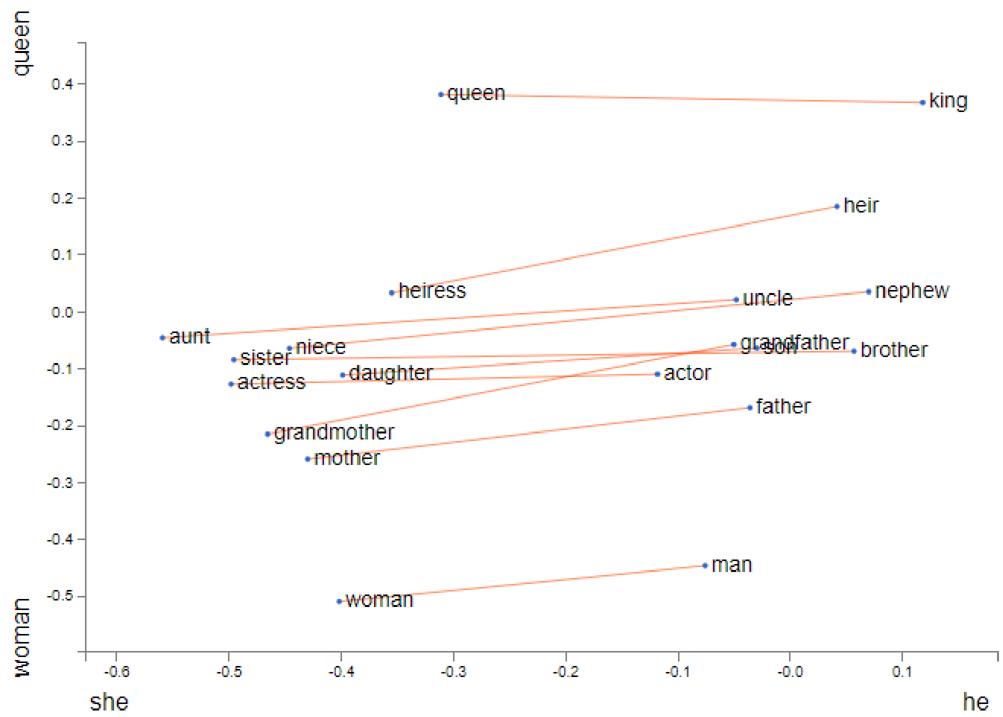
$$(\mathrm{w_i} - \mathrm{w_j})^T w_k = \log \frac{p(k|i)}{p(k|j)}$$

GloVe (Global Vectors)

$$(\mathrm{w_i}-\mathrm{w_j})^T w_k = \log\frac{p(k|i)}{p(k|j)}$$



Demo



Explore word analogies

What do you want to see?

Gender analogies

Modify words

Type a new word... | Add

Type a new word... | Type a new word... | Add pair

X axis: she | he

Y axis: woman | queen

Change axes labels

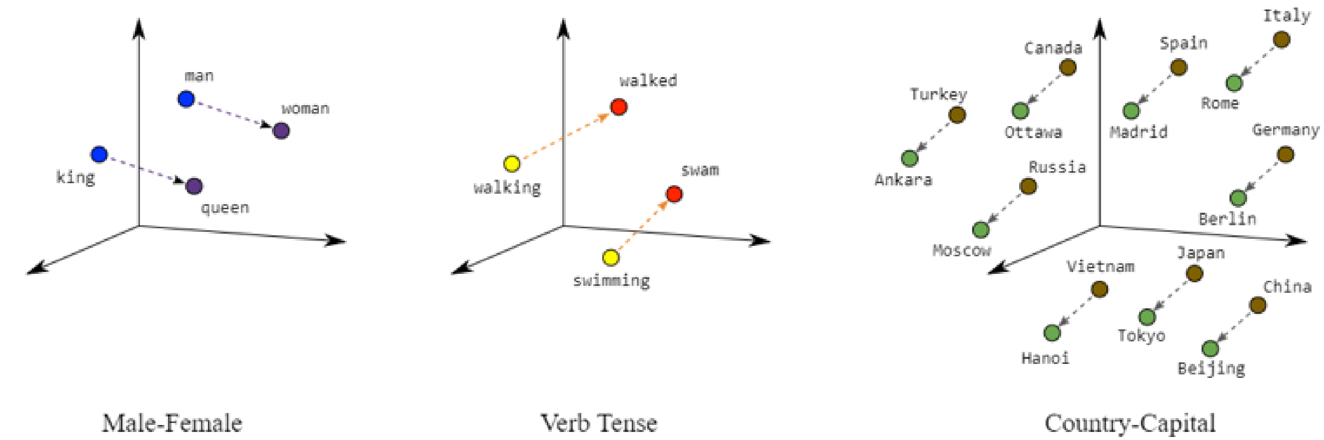
Interactive visualization of word analogies in GloVe. *Hover* to highlight, *double-click* to remove. *Change axes* by specifying word differences, on which you want to project. Uses (compressed) pre-trained word vectors from glove.6B.50d. Made by Julia Bazińska under the mentorship of Piotr Migdał (2017).

Learn more in this blog post!

Explore: https://lamyiowce.github.io/word2viz/

Train on your own text: https://remykarem.github.io/word2vec-demo/

word2vec and Glove



Encodes semantic information

Low dimensionality

It can be pre-trained on an independent corpus

It does not allow for out-of-vocabulary words

fastText

"rain", "rainbow", "raincoat"

Similar spelling implies relationship

"dog" → "dogs"   "car" → "cars"

"boy" → "boyfriend"   "girl" → "girlfriend"

Relationships learned between words can be extended to other words

Word2vec does not directly use morphology information. "fastText" proposes exploiting morphology by using **subword embedding**

<rainbow>

$g \in G$	<ra	rai	ain	inb	nbo	bow	ow>
$\mathbf{z}$							

$$\mathbf{u}_w = \sum_{g \in G_w} \mathbf{z}_g$$

The word vector u_w is the sum of the subword embeddings. The rest is the same as the skip-gram model.

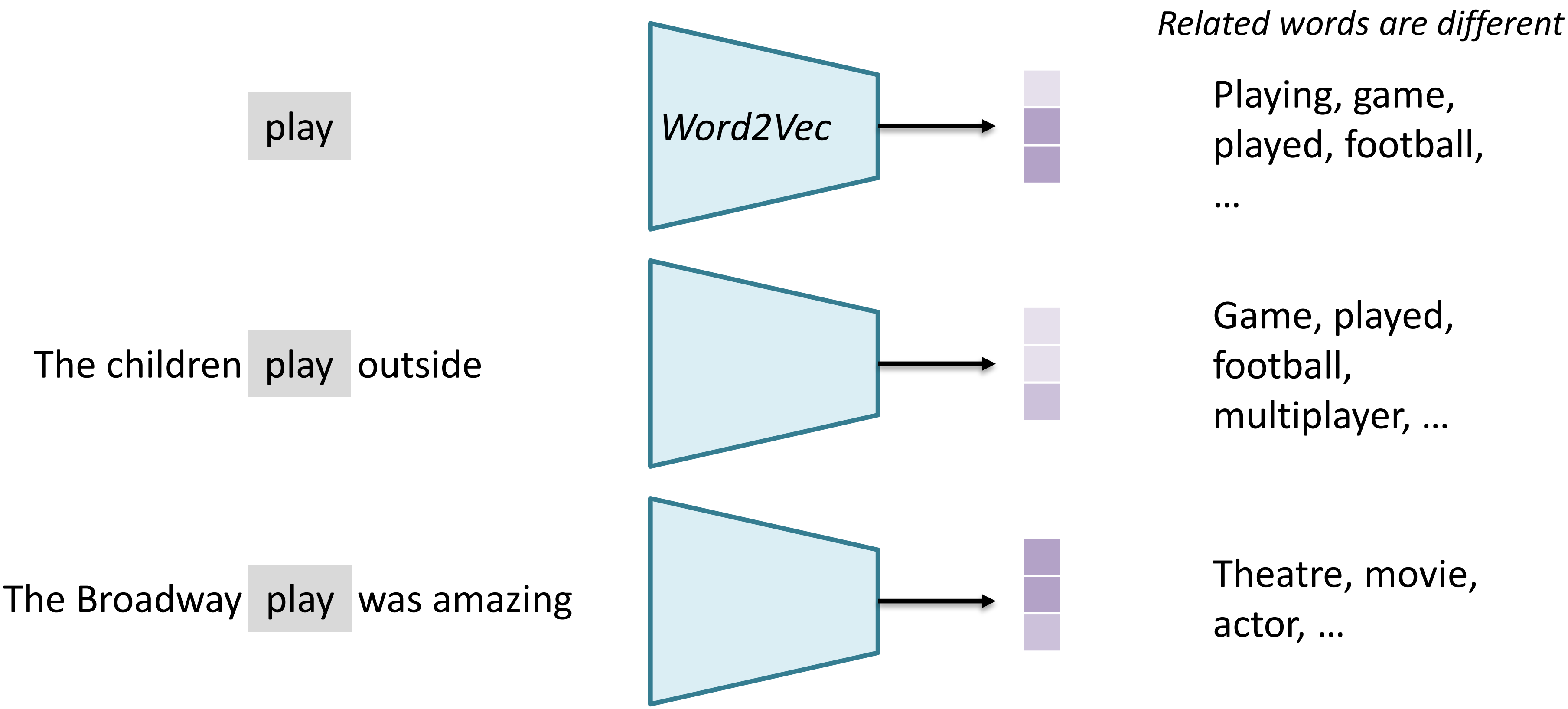
Better vectors for **uncommon** words, deals with **out-of-vocabulary** words!

Text Embeddings

CONTEXTUAL EMBEDDINGS

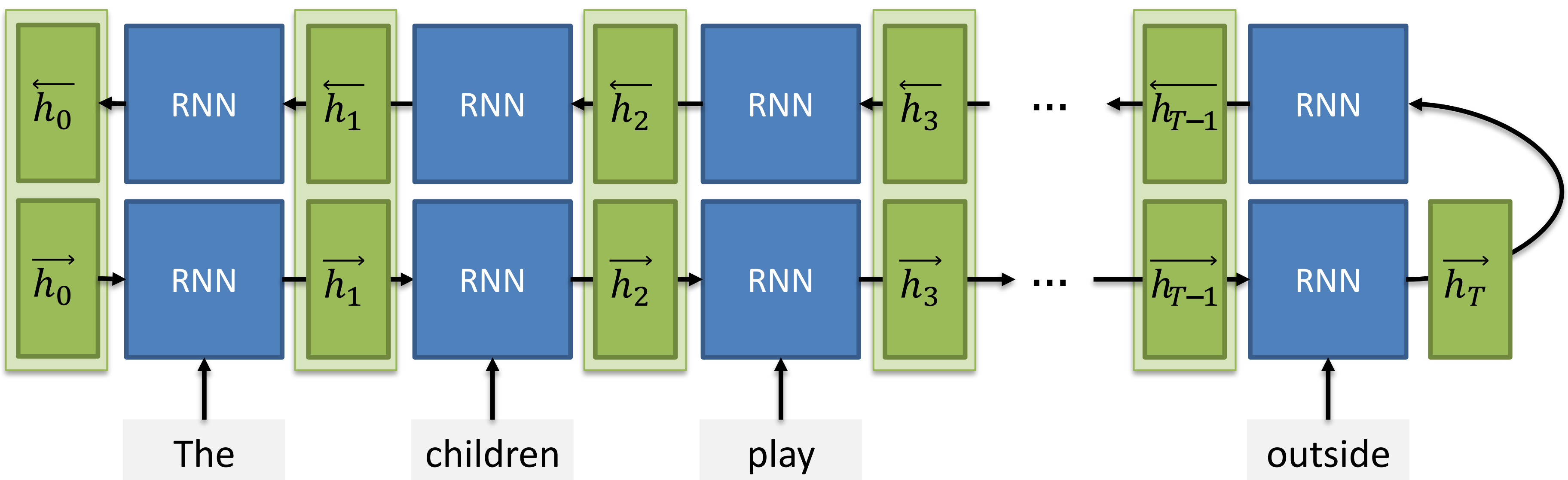
Context (again)

Word2Vec, Glove and fastText produce specific vectors for specific words



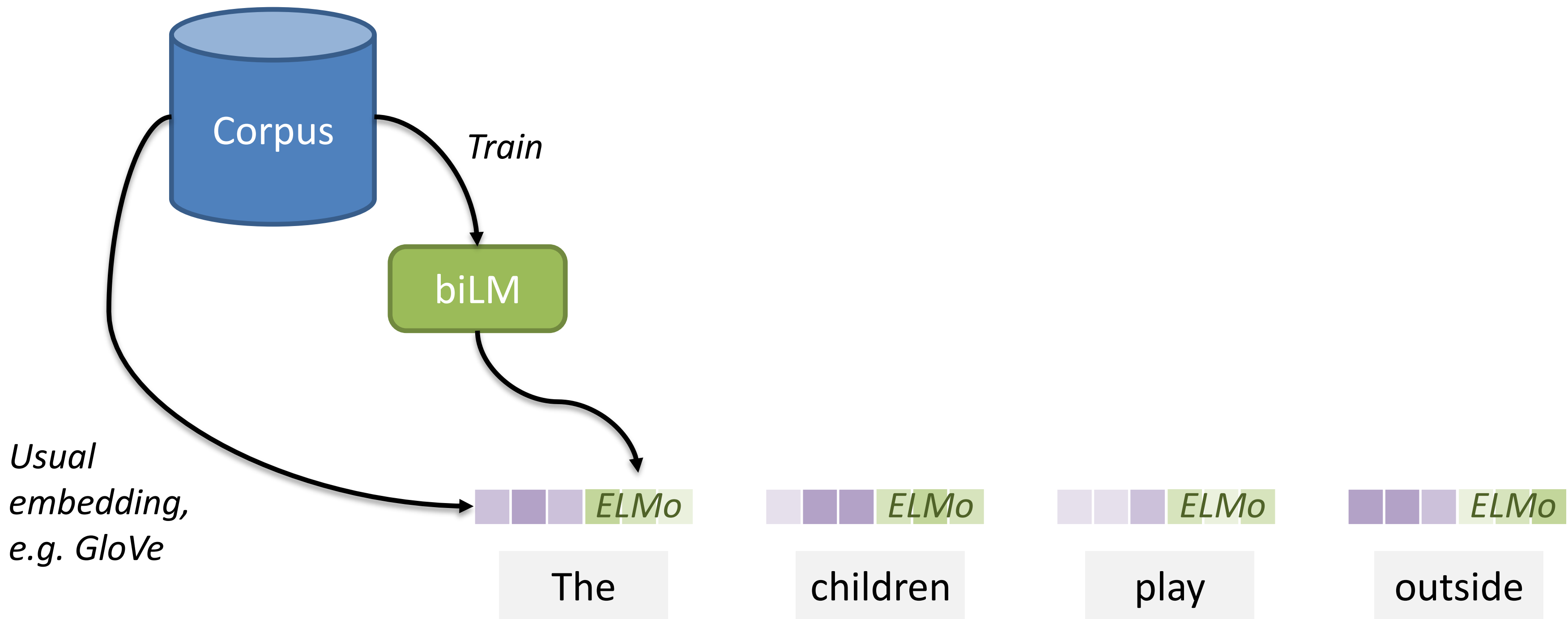
ELMo
(Embeddings from Language Models)

Idea: learn embeddings from **building bidirectional language models** (biLM)



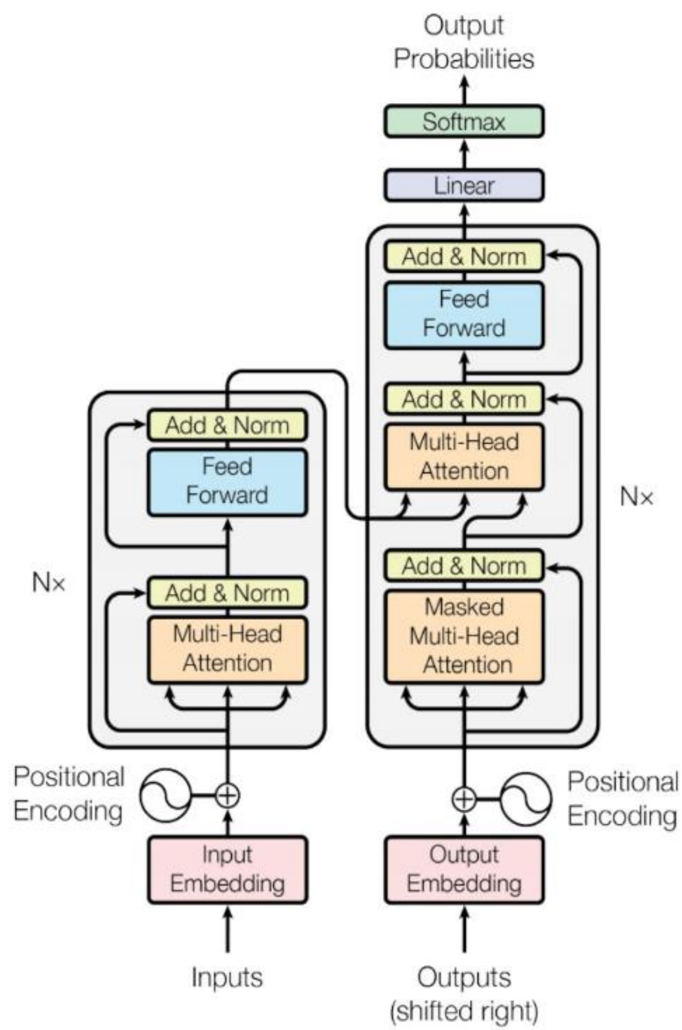
ELMo uses an LSTM to predict the words in both directions to build a biLM. Each word is represented as a linear combination of corresponding hidden layers

ELMo
(Embeddings from Language Models)



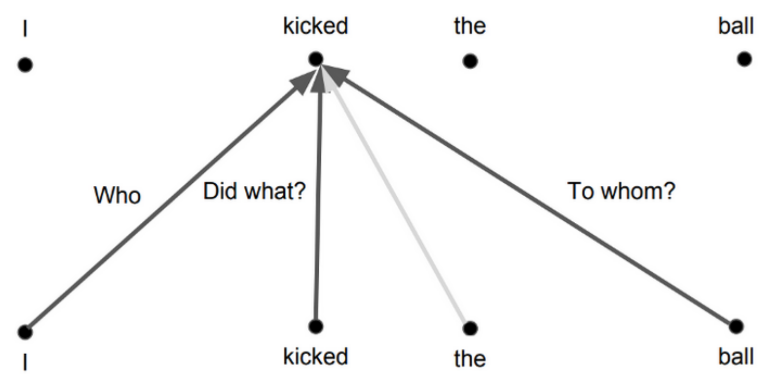
ELMo uses an LSTM to predict the words in both directions to build a biLM. Each word is represented as a linear combination of corresponding hidden layers

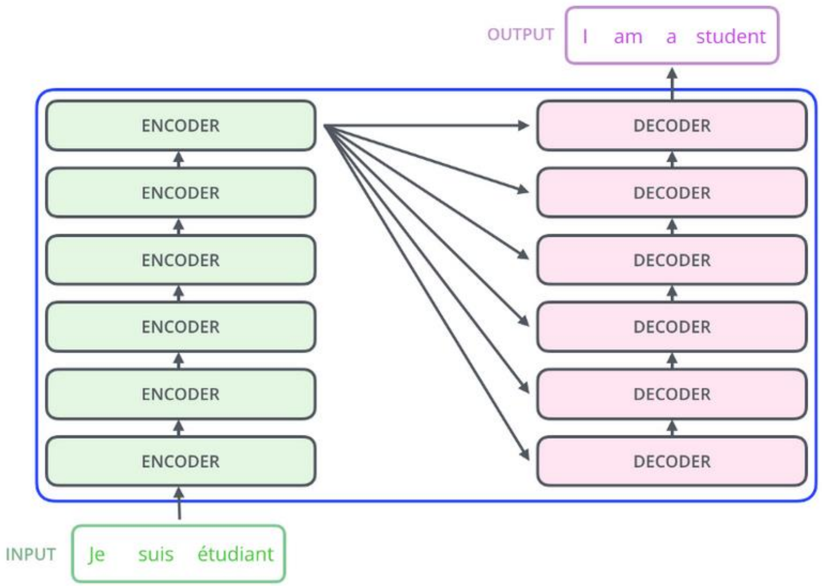
(Interlude: Transformers



A. Vaswani et al.: **Attention is all you need**. NIPS, 2017.

Interlude: Transformers



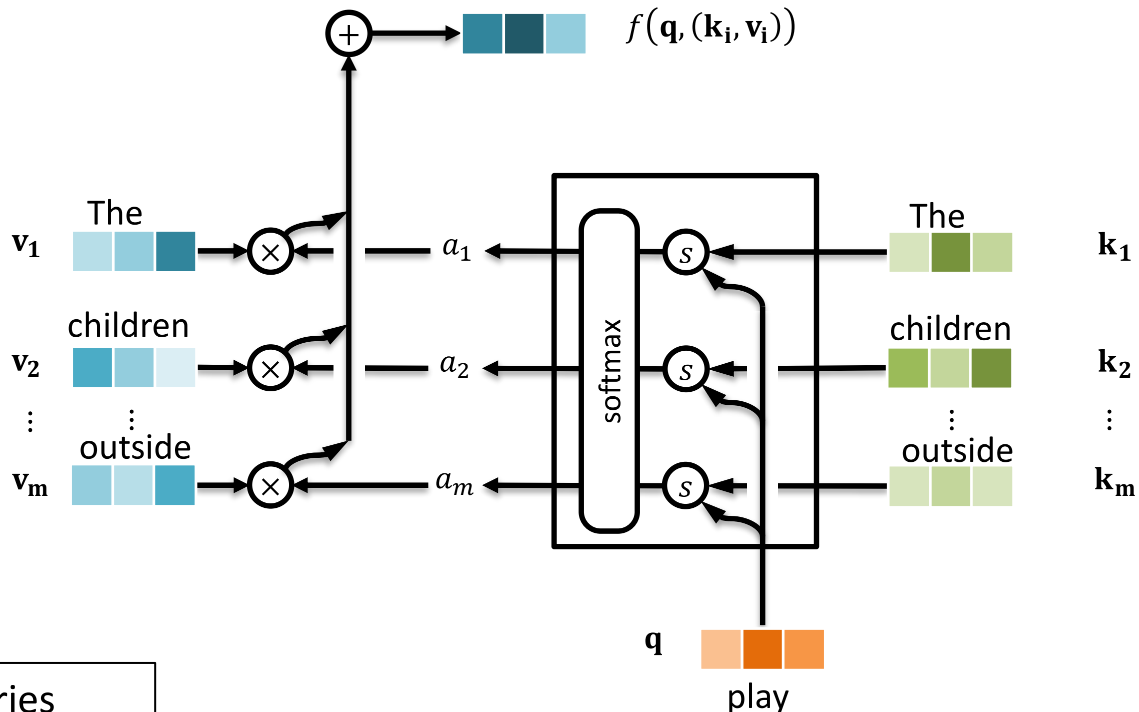


Encoder. Passing through several transformer units, the representation of a word is modified according to the context of neighbourhood words.

Decoder. Using the encoded representation, each output word is generated according to the context of neighbourhood words.

A. Vaswani et al.: **Attention is all you need**. NIPS, 2017.

Attention Mechanism



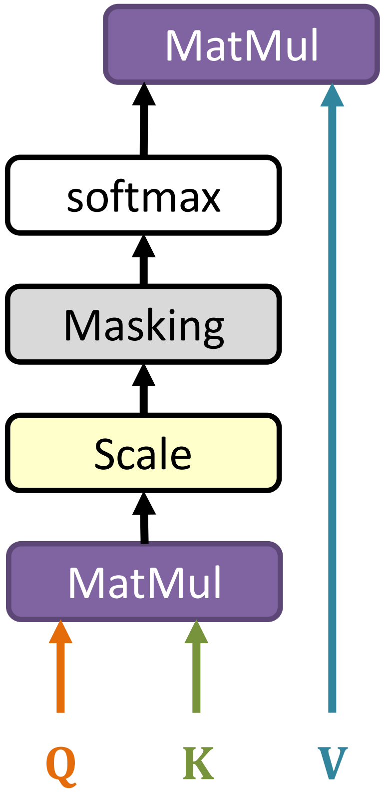
n: # of queries
m: # of keys
$\mathbf{q}, \mathbf{k} \in \mathbb{R}^d$
$\mathbf{Q} \in \mathbb{R}^{n \times d}$
$\mathbf{K} \in \mathbb{R}^{m \times d}$
$\mathbf{V} \in \mathbb{R}^{m \times v}$

$$s(\mathbf{q}, \mathbf{k}) = \frac{\mathbf{q}^T \mathbf{k}}{\sqrt{d}}$$

Scaled Dot-Product Attention

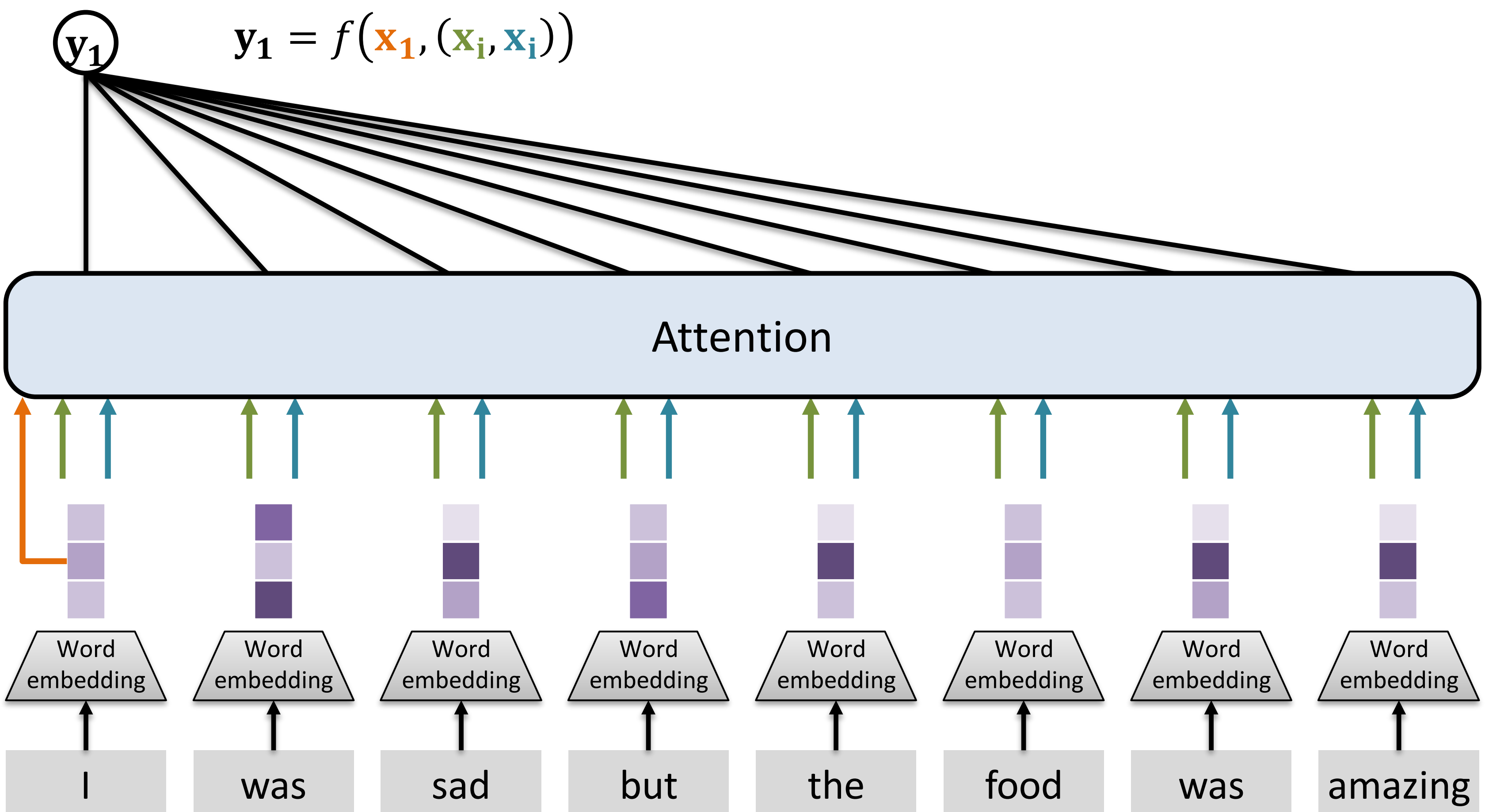
Efficient parallel implementation
for multiple keys/queries:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}}\right)\mathbf{V}$$



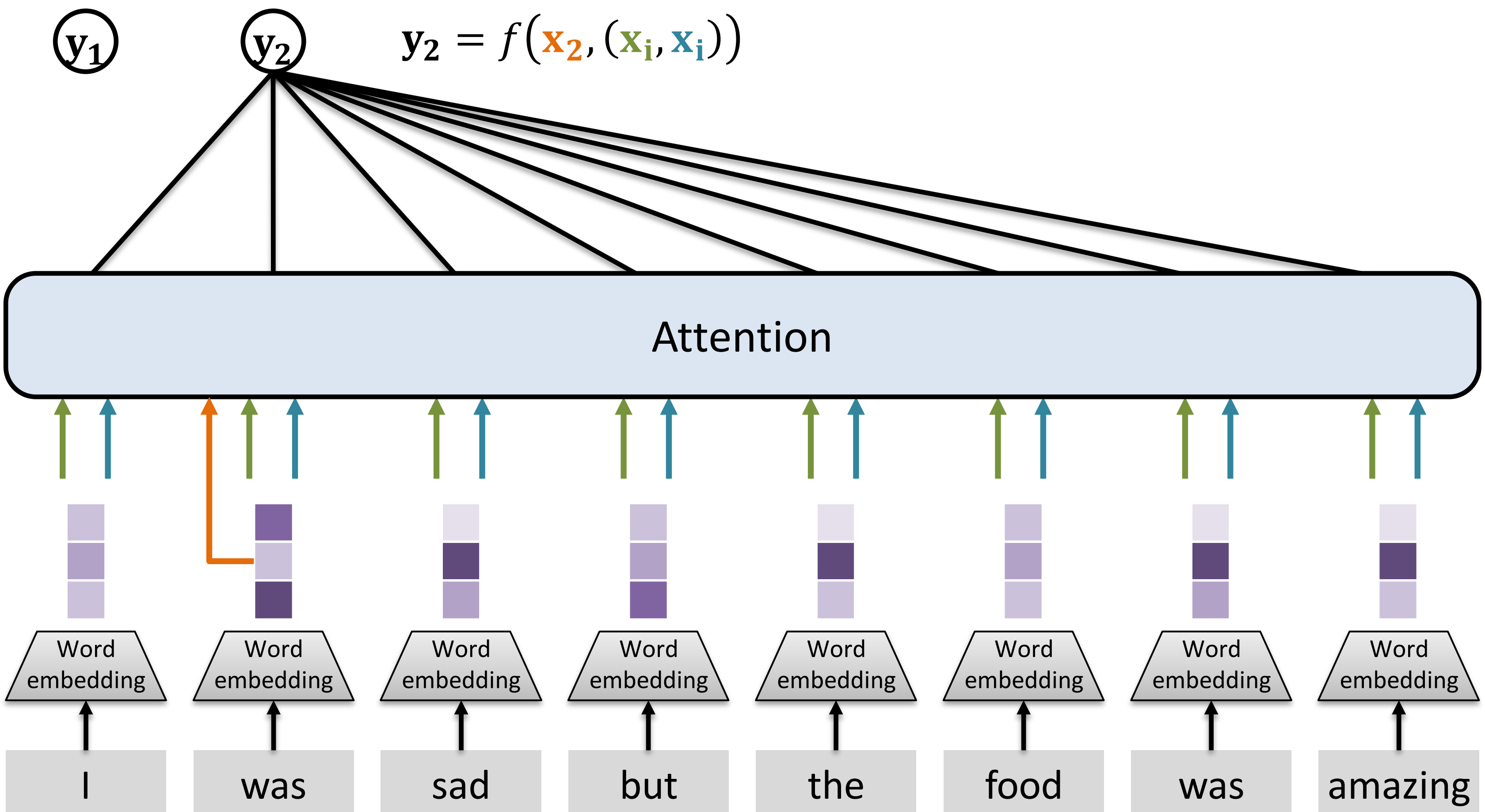
Self attention

In the case of self-attention, **all three** queries, keys and values **come from the same tokens**



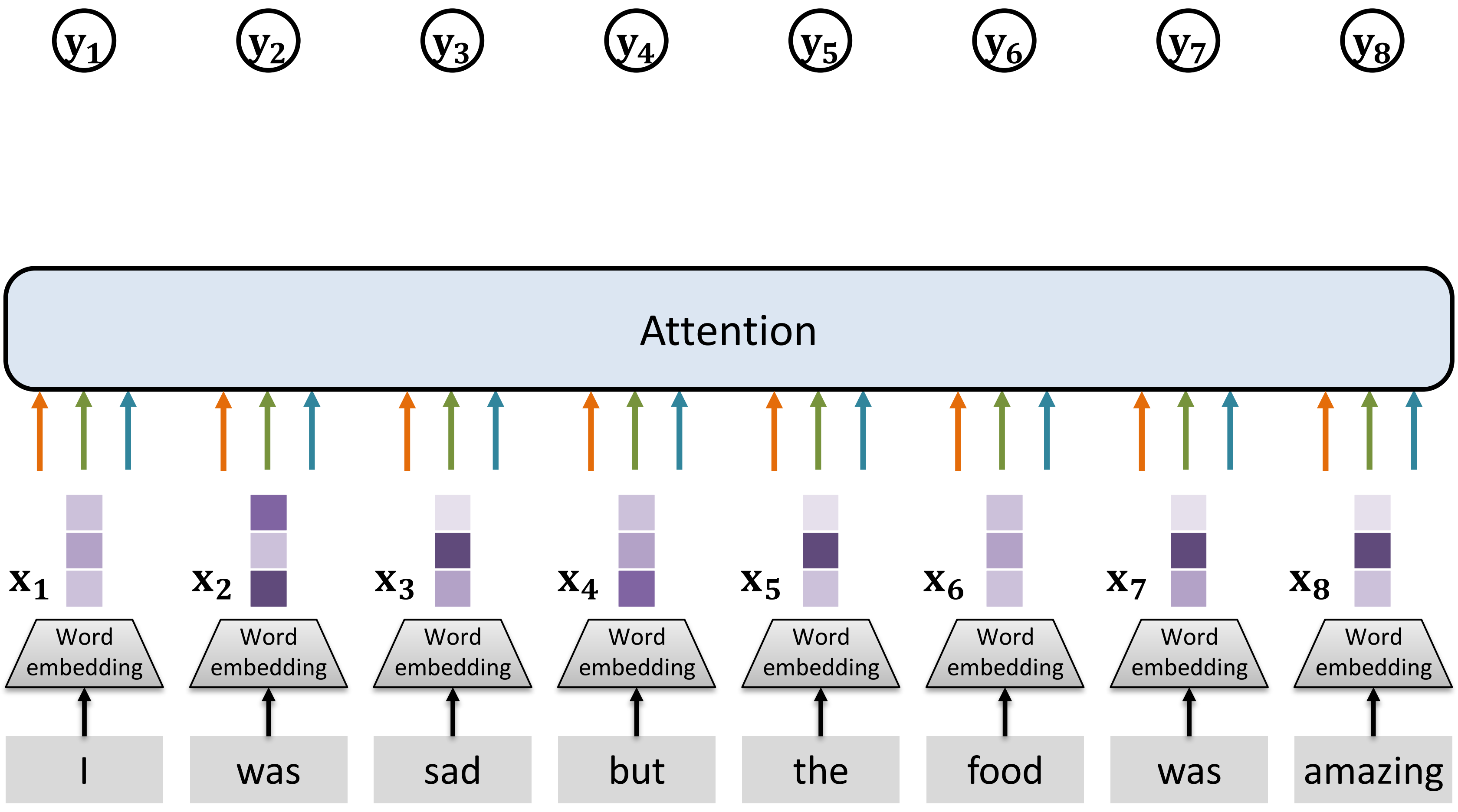
Self attention

In the case of self-attention, **all three** queries, keys and values **are the same**

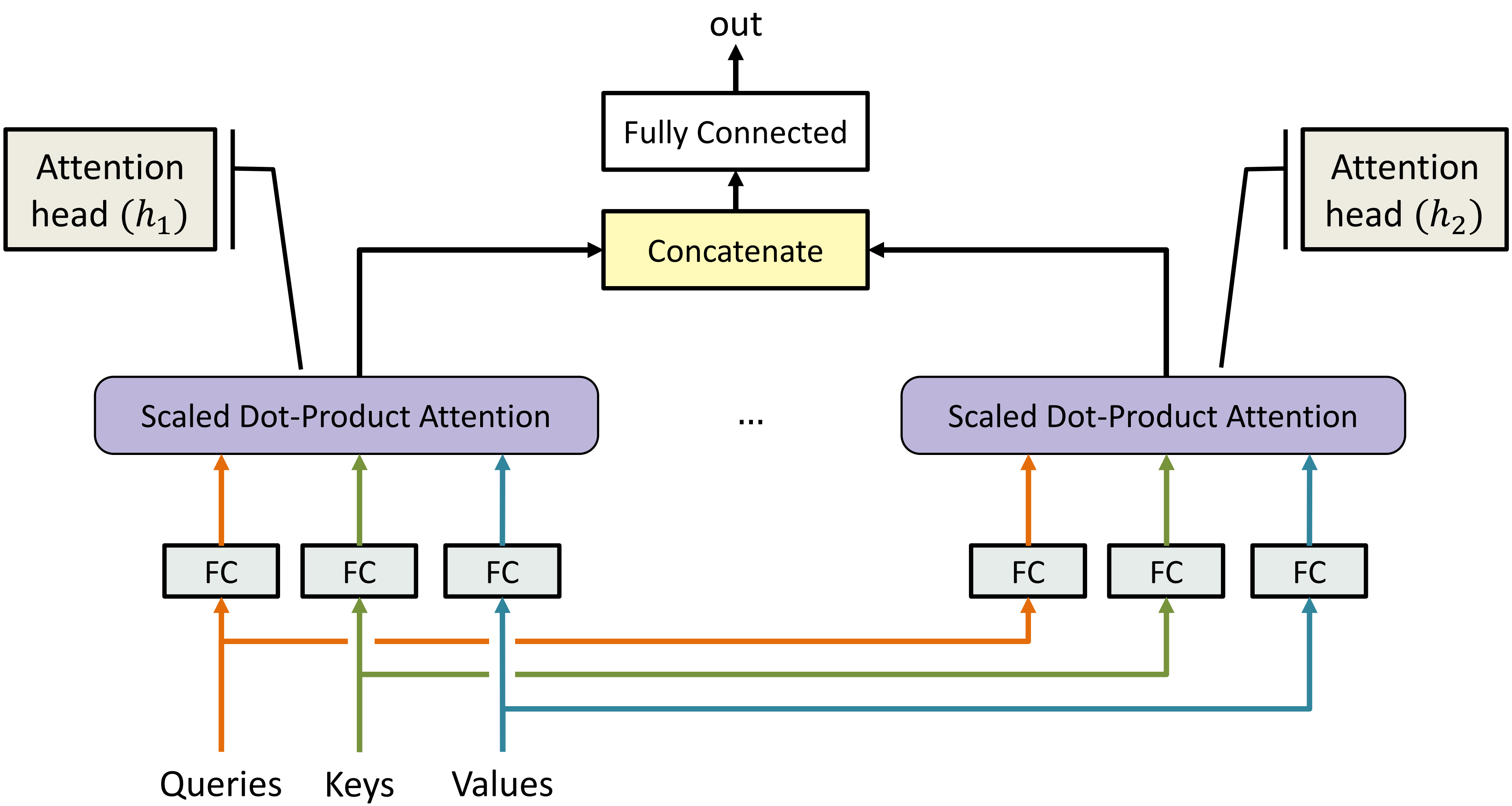


Self attention

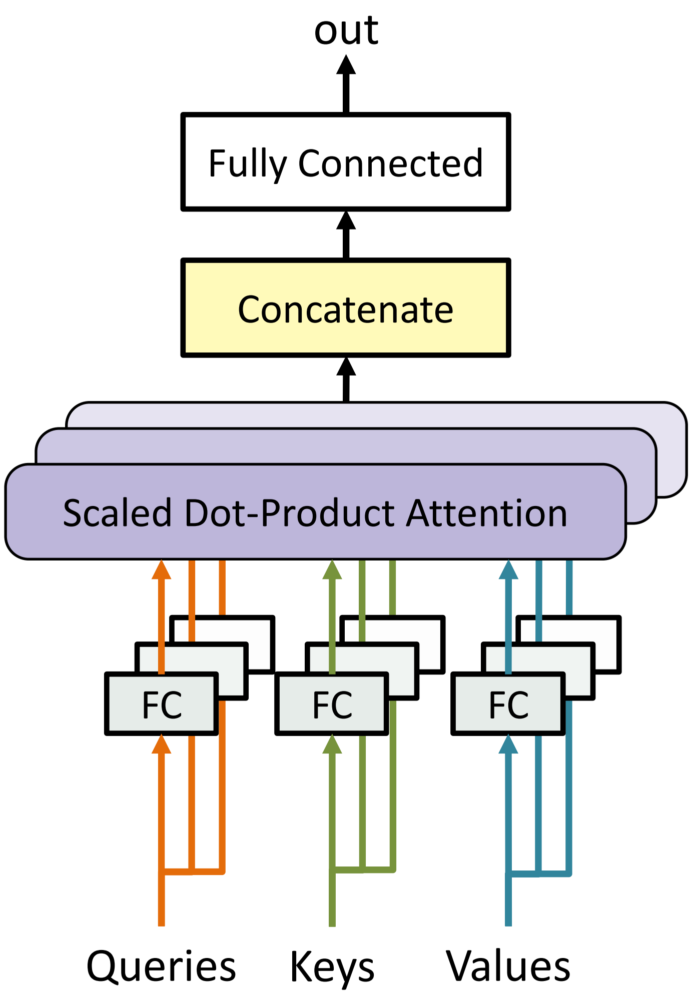
$$\mathbf{y_i} = f\left(\mathbf{x_i}, (\mathbf{x_1}, \mathbf{x_1}), (\mathbf{x_2}, \mathbf{x_2}), \ldots, (\mathbf{x_n}, \mathbf{x_n})\right) \in \mathbb{R}^d$$



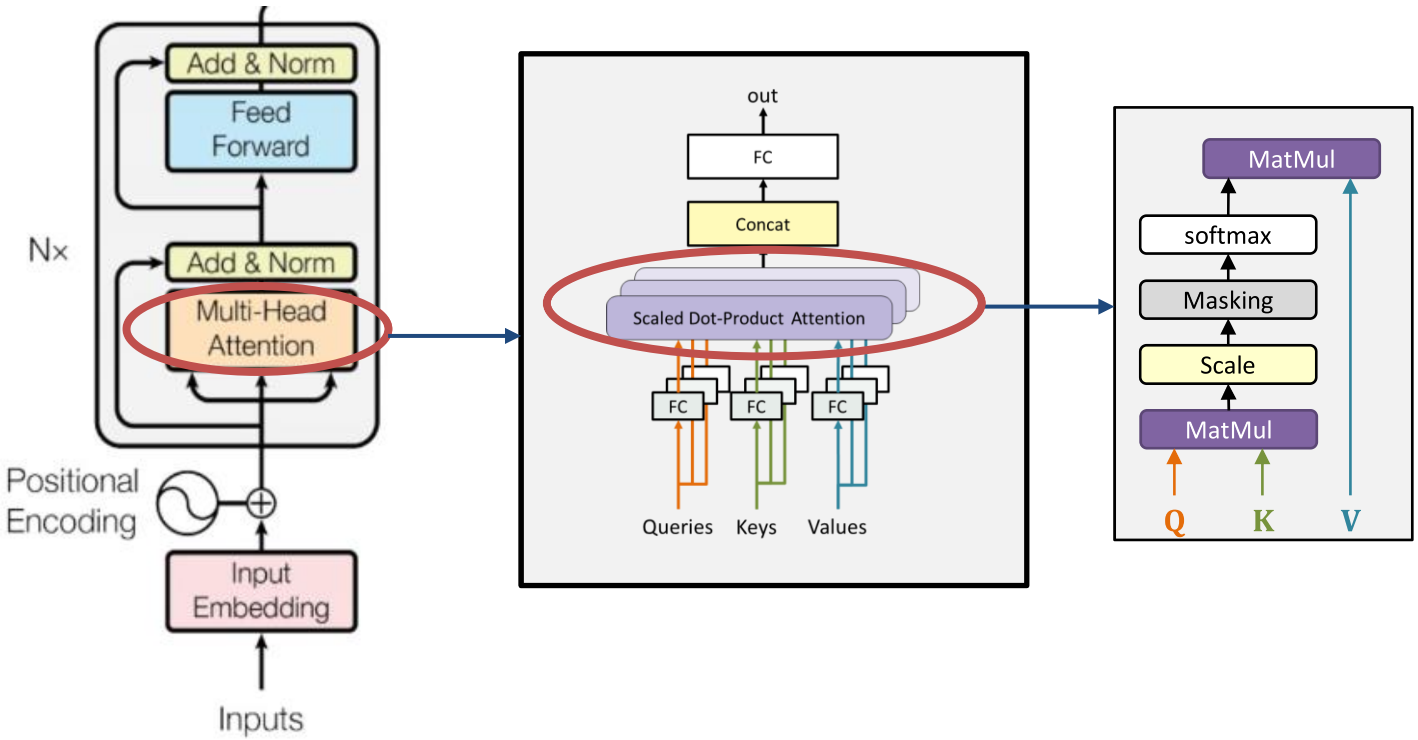
Multi-head Attention



Multi-head Attention



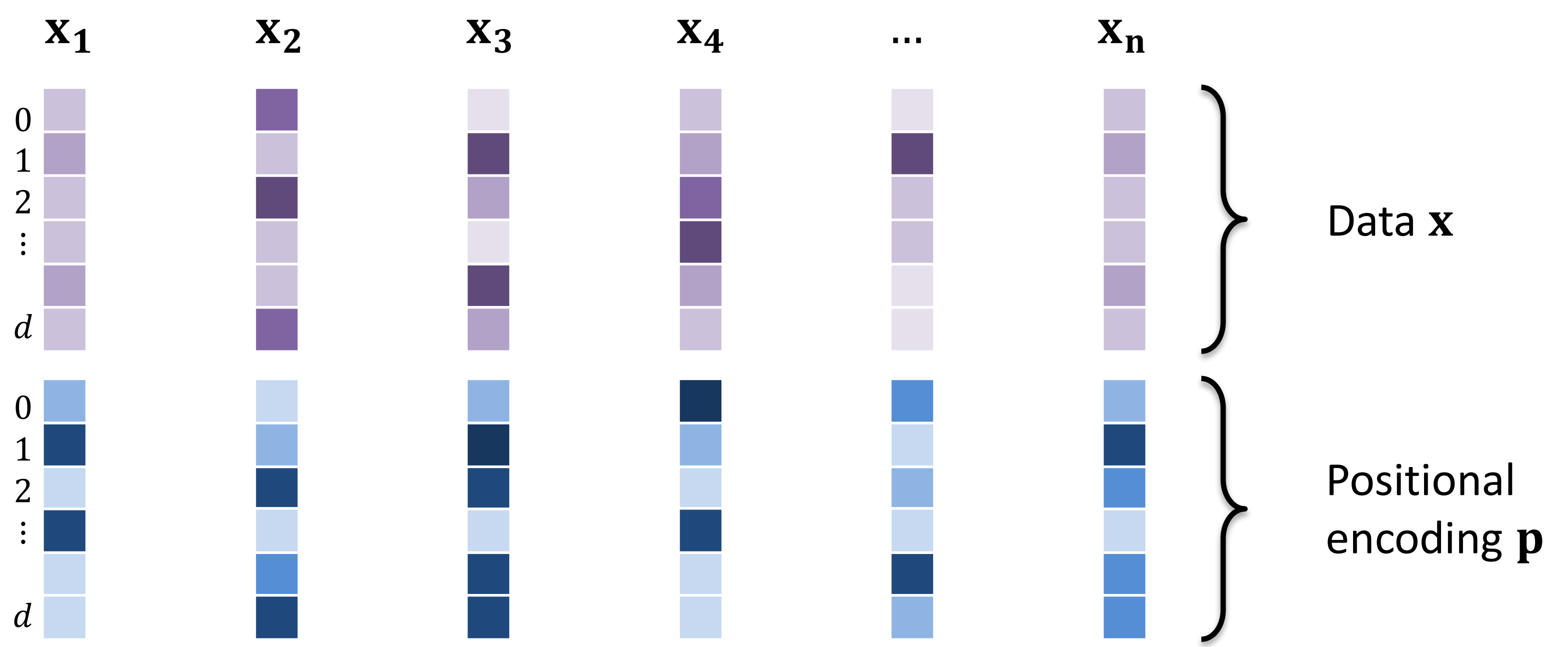
Putting Everything Together



Positional Encoding

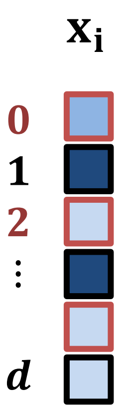
Self-attention ditches sequential operations in favor of parallel computation.

If the order is important, we need to **explicitly inject** absolute or relative **positional information** by adding *positional encoding* to the input representations.



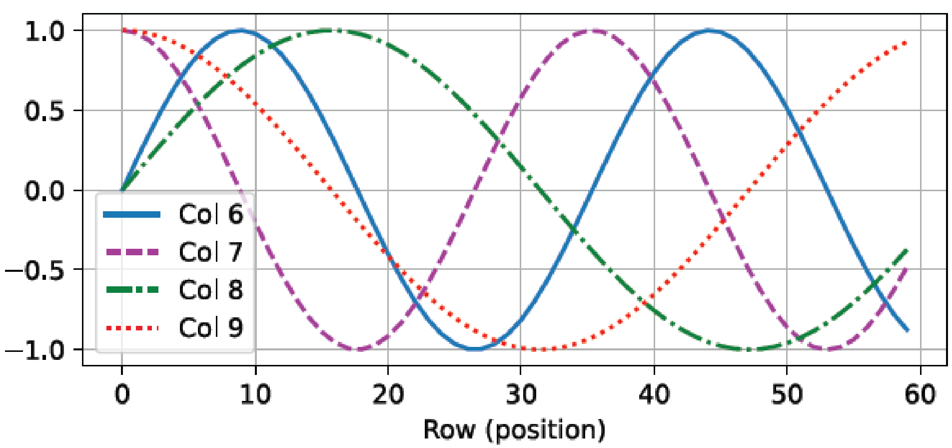
Positional Encoding)

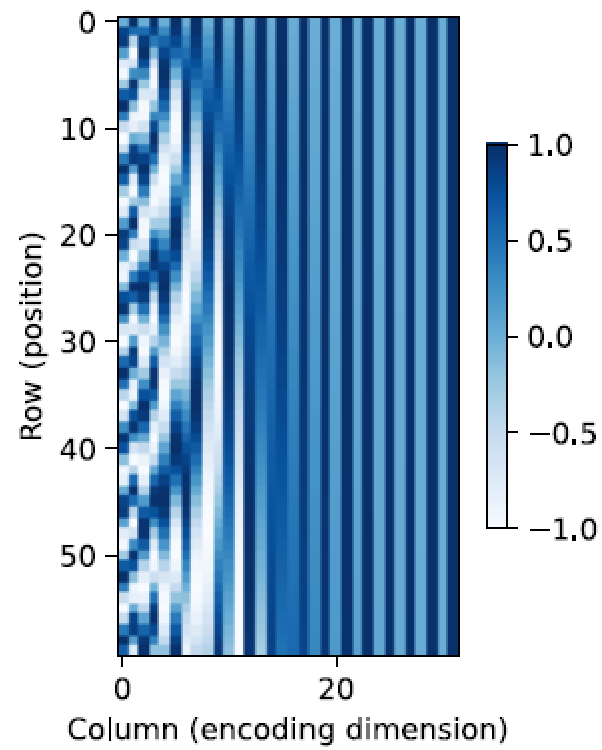
Resembles a binary representation, but continuous (more space efficient)



$$p_{i,\mathbf{2j}} = \sin\left(\frac{i}{10000^{2j/d}}\right)$$

$$p_{i,\mathbf{2j+1}} = \cos\left(\frac{i}{10000^{2j/d}}\right)$$



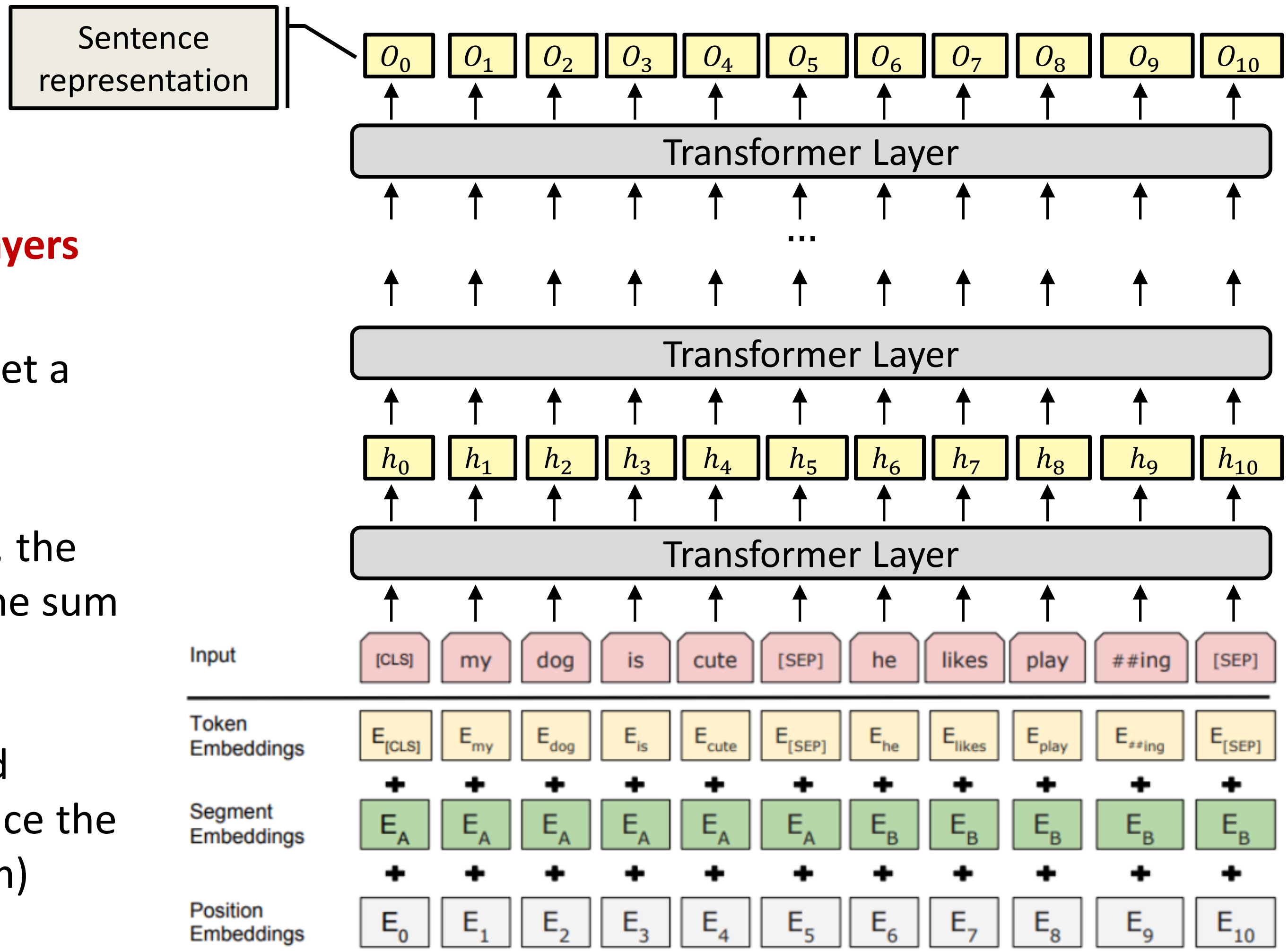


BERT: Bidirectional Encoder Representations from Transformers

Stacked Transformer layers (only encoder)

Adds a **[CLS] token** to get a global sentence representation

For every token (word), the input to the model is the sum of **three learned embeddings**: token semantics, position and segment (which sentence the input token comes from)



J. Devlin, et al: **BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding**. NAACL-HLT, 2019

BERT: Bidirectional Encoder Representations from Transformers

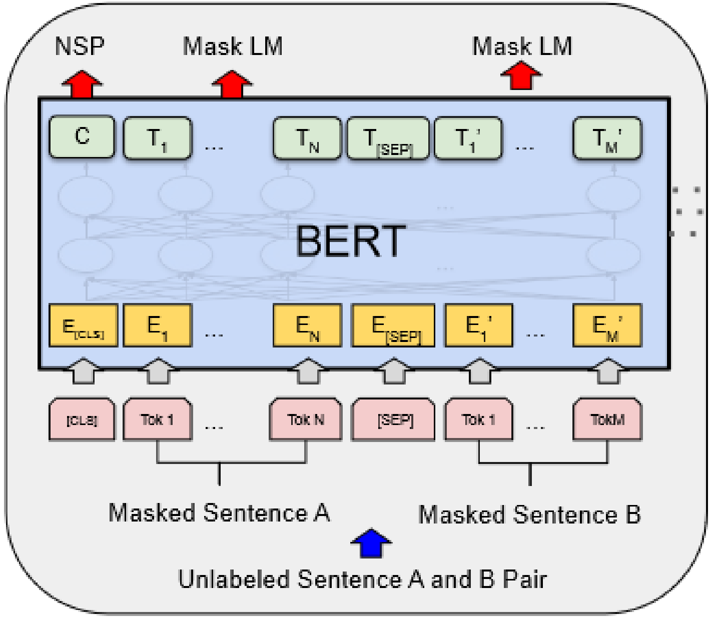
Pre-training: the model is trained on pairs of sentences with two pre-training tasks:

Masked Language Model (MLM)

- Randomly replace some tokens (15%) with a `[MASK]` token or with another token.
- The model is trained to predict the masked tokens using a softmax layer on top of the final token representations.

Next Sentence Prediction (NSP)

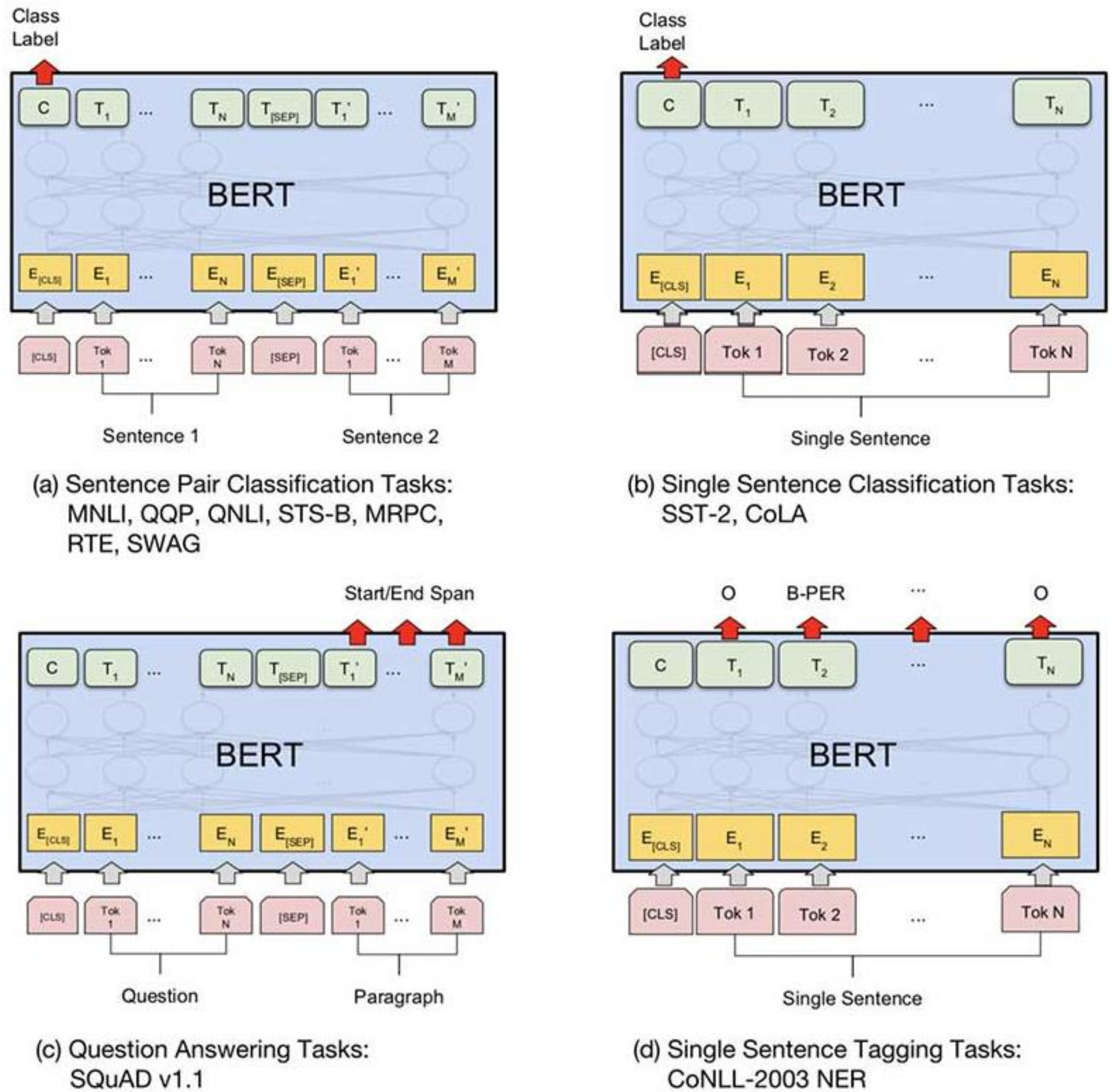
- Randomly break the logical sequential order between input sentences A and B. 50% of the time, B is the actual next sentence that follows A in the training corpus and 50% of the time, it is a random sentence.
- The model is trained to predict whether the two sentences follow the correct order or not, using a softmax layer on top of the final representation of the `[CLS]` token



BERT: Bidirectional Encoder Representations from Transformers

Once pre-trained the model can be used in different ways:

- The final representation of every token in a sentence can be taken as a **contextual semantic word embedding**
- The final representation of the [CLS] token can be taken as a **global semantic representation** of the whole sentence
- The **whole model can be fine-tuned** for several downstream tasks: question answering, sentence classification, tagging, etc.



(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG

(b) Single Sentence Classification Tasks: SST-2, CoLA

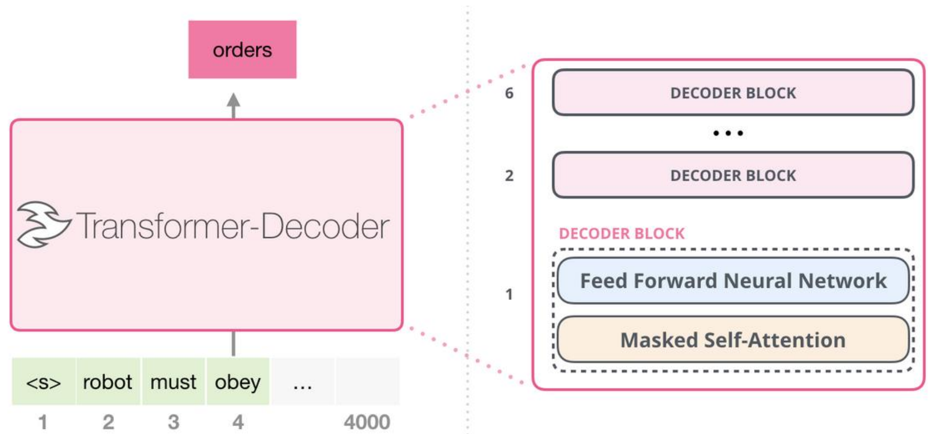
(c) Question Answering Tasks: SQuAD v1.1

(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

Other Language Models

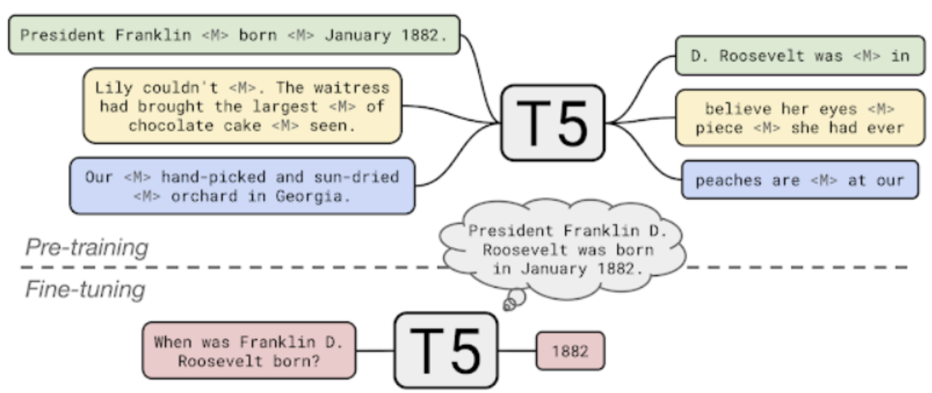
GPT

- Autoregressive model. Pre-training predicts the next token in the input sequence, one at each generation step
- Unidirectional, left-to-right. Only previous tokens in the sequence are considered when predicting the next token
- Only uses the decoder segment of the Transformer model



T5

- Text-to-text framework. All tasks are converted into a text generation format
- MLM pre-training generates the sequence of masked tokens
- Uses the complete encoder-decoder Transformer model



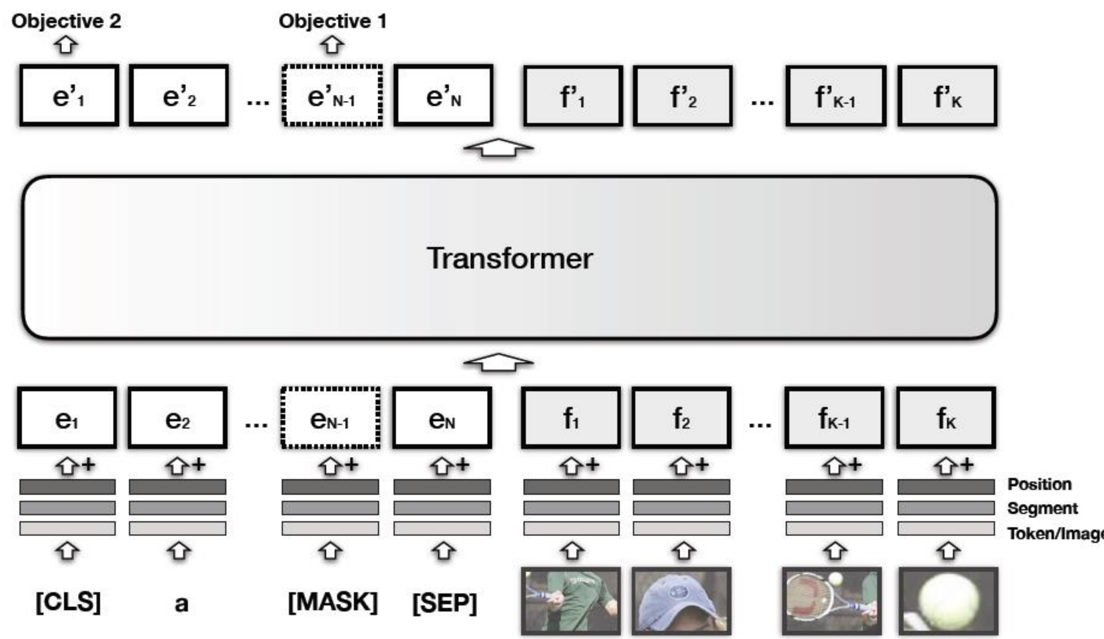
MULTIMODAL EMBEDDINGS

VisualBERT

Extends BERT to include visual tokens using image features extracted from region proposals using Faster R-CNN

By modelling the interaction among words and object proposals the model captures the intricate associations between text and image

A person hits a ball with a tennis racket



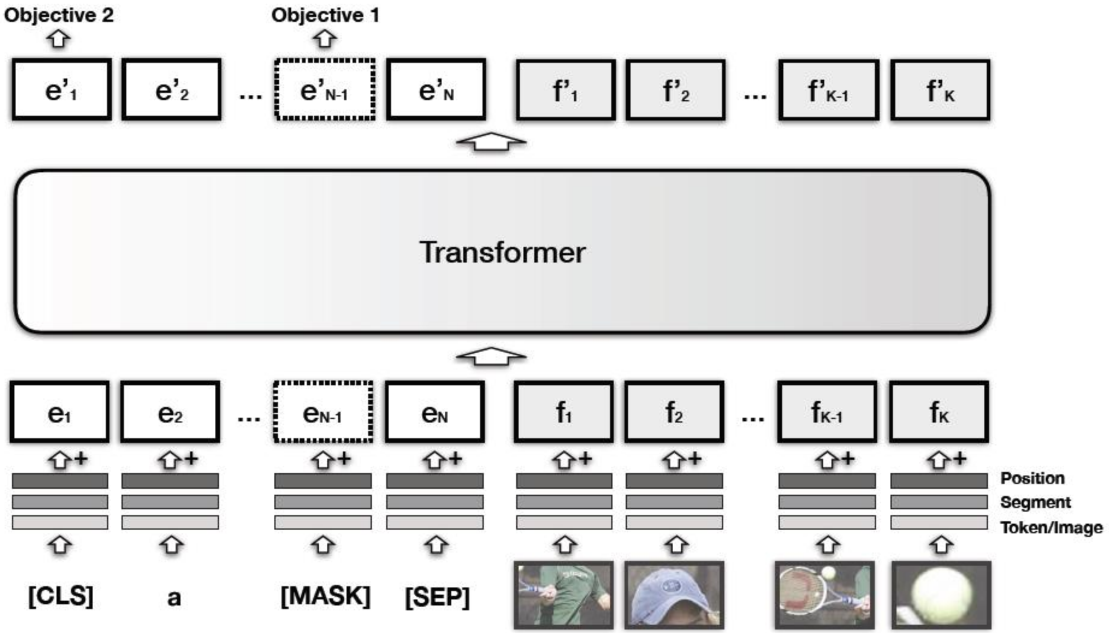
L. H. Li, et al: **VisualBERT: A Simple and Performant Baseline for Vision and Language**. ACL 2020

VisualBERT - Input

Every visual token is the sum of three learned embeddings:

- **Visual feature representation** of a region proposal
- **Segment** embedding: text/image
- **Position embedding**: sum of position embeddings of the words aligned with the regions proposal (when these alignments are provided)

A person hits a ball with a tennis racket



L. H. Li, et al: **VisualBERT: A Simple and Performant Baseline for Vision and Language**. ACL 2020

VisualBERT - Pretraining

The model is trained on pairs of paired sentences and images with two pre-training tasks:

Masked Language Model (MLM)

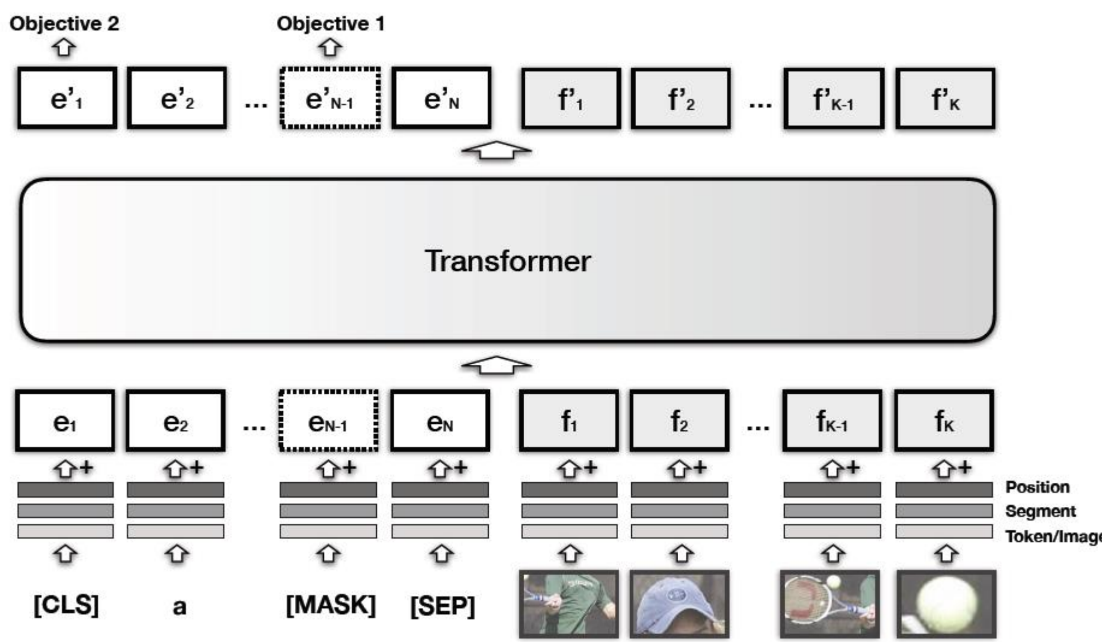
Random textual tokens are masked (as in BERT) and must be predicted with the context of the rest of textual tokens and all visual tokens

Sentence-image prediction

Every image is associated with a text segment consisting of two captions.

One of the two captions always correspond to the image while for the other caption 50% of the time is a corresponding caption and 50% of the time is a random caption.

The model is trained to predict whether the two captions correspond to the image or not, using the final representation of the [CLS] token.



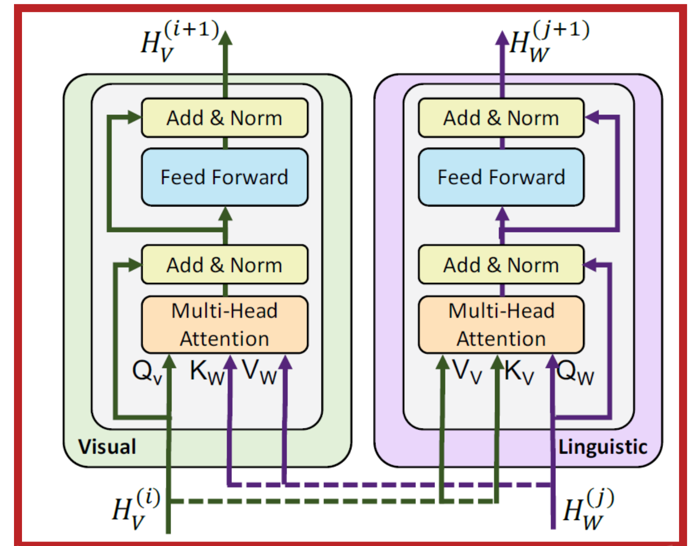
L. H. Li, et al: **VisualBERT: A Simple and Performant Baseline for Vision and Language**. ACL 2020

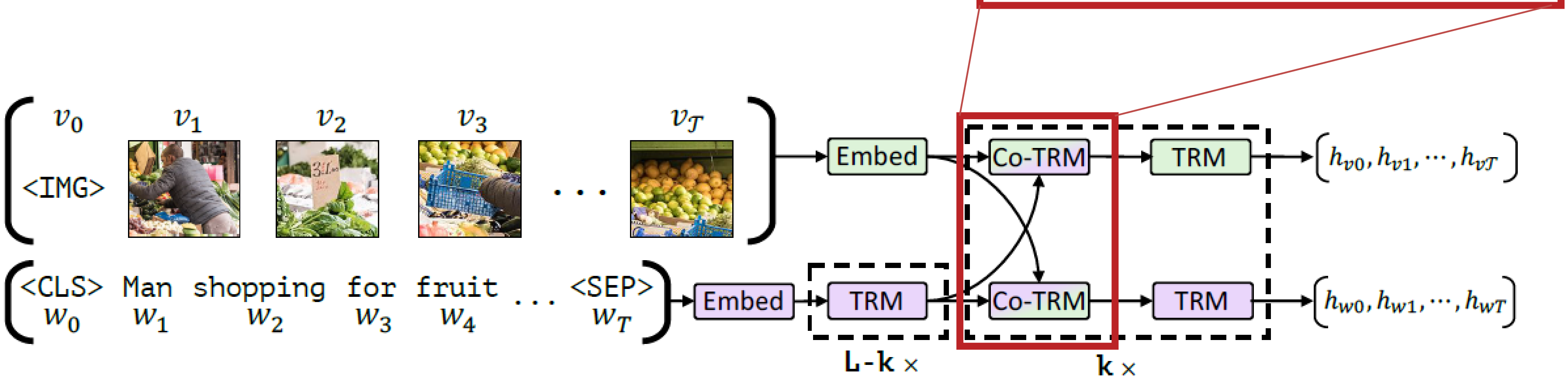
ViLBERT

Two **separate input streams** for visual and textual tokens

Introduces a **co-attention transformer** layer to capture interactions between both modalities:

- The keys and values from each modality are passed as input to the other modality's multi-headed attention block
- It produces attention-pooled features for each modality conditioned on the other





J. Lu, et al: **ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks**. *NIPS* 2019.

ViLBERT

Textual representation

- Initialized with **pre-trained BERT**

Visual representation

- Visual features of image regions extracted with a **pre-trained object detector**
- Position embedding: learned embedding of a 5-d vector from **bounding box coordinates** and **fraction of image area covered**
- A global **<IMG>** token is added at the beginning of the image representation (similar to <CLS> token for text)

Pre-training

- **Multi-modal mask modelling task**: similar to MLM in BERT, but random visual tokens are also masked. The model must predict the original distribution over semantic classes of the image region obtained with the detection model.
- **Multi-model alignment task**: using the representation of <IMG> and <CLS> tokens the model must predict whether image and text are aligned (*i.e.* the text is a corresponding description of the image)

J. Lu, et al: **ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks**. *NIPS* 2019.

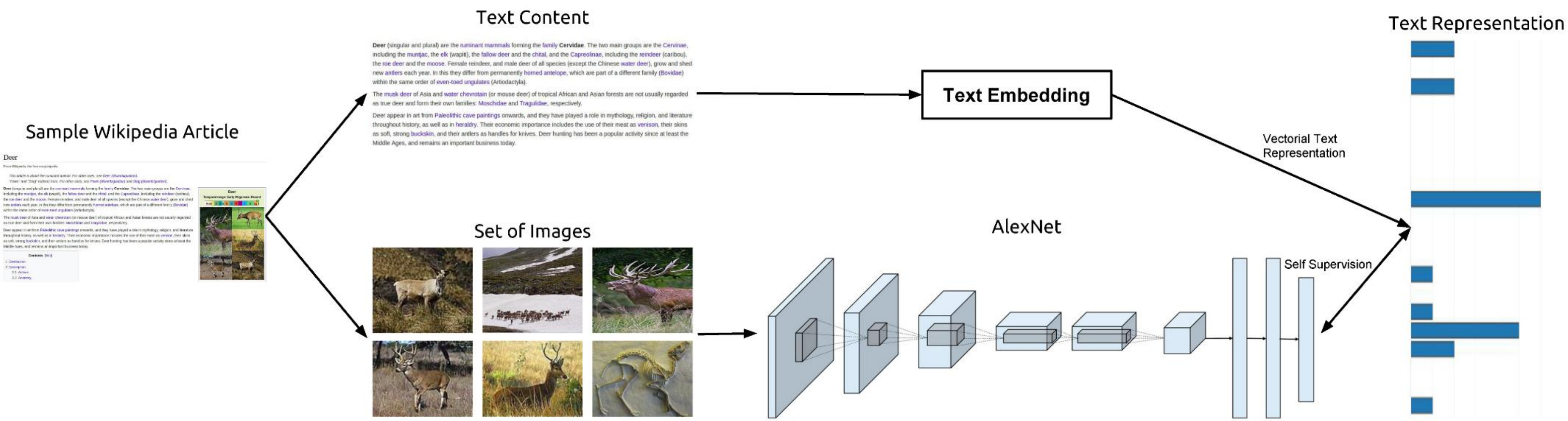
Multimodal Learning

ALIGNING MODALITIES


THE
SNUGGL

Learning to understand images by reading the Wikipedia

Task: Look at the image and predict what kind of article (topic) it illustrates



L. Gomez, Y. Patel, M. Rosinyol, D. Karatzas, "Self-supervised learning of visual features through embedding images into text topic spaces", CVPR 2017

Self-supervised learning from Web Data

Wikipedia:
1.7M articles in English with 4.2M associated illusrtative images.

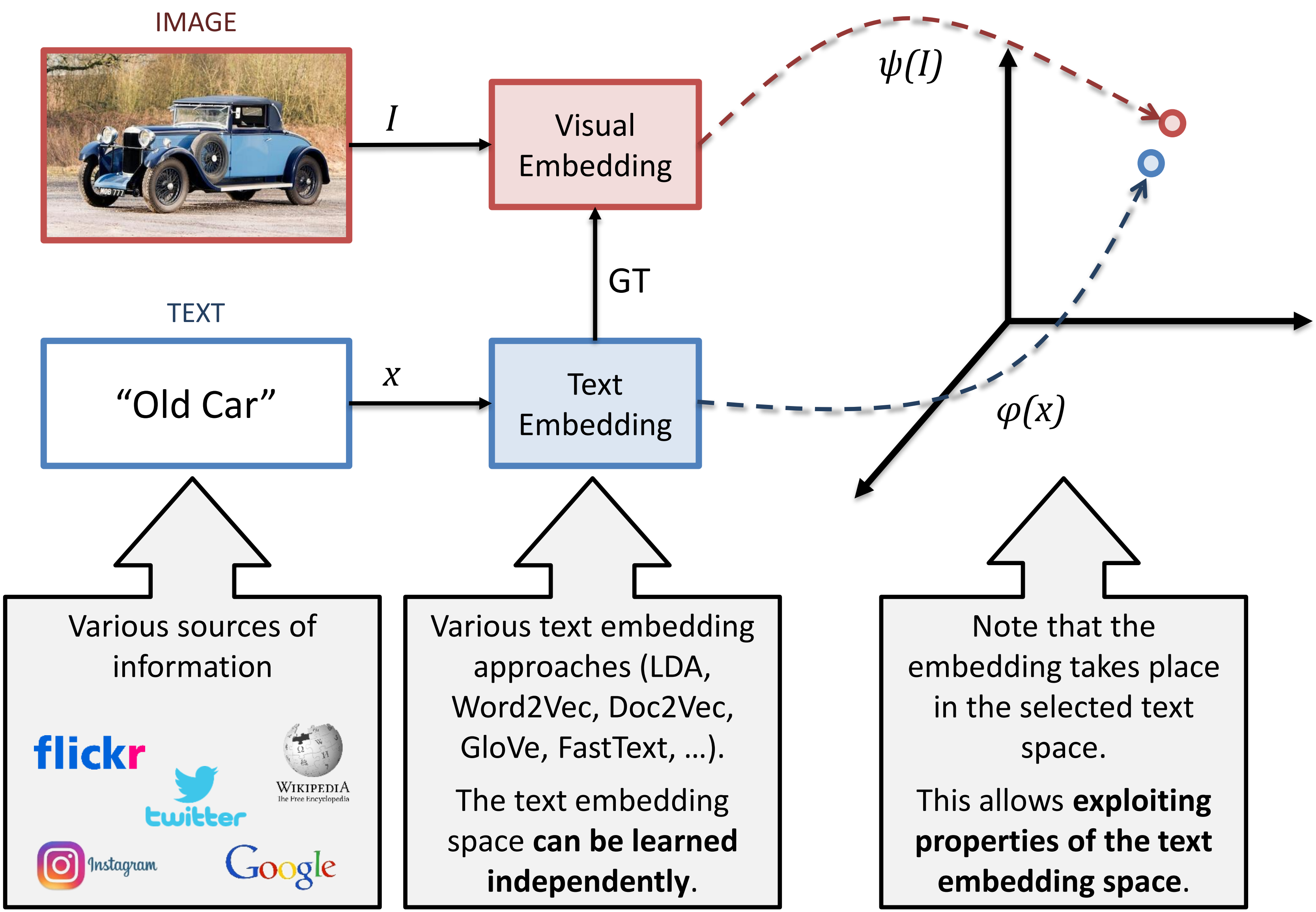
WebVision:
2.4M Flickr and Google images associated to ImageNet classes.

InstaCities1M:
1M Instagram images associated with one of the 10 most populated English speaking cities.

Gomez, R., Gomez, L., Gibert, J., & Karatzas, D. "Learning to Learn from Web Data through Deep Semantic Embeddings" MULA 2018

Gomez, R., Gomez, L., Gibert, J., & Karatzas, D. "Self-Supervised Learning from Web Data for Multimodal Retrieval", arXiv:1901.02004, 2019

Self-supervised learning from Web Data



Text-based semantic retrieval

"haircut"

Gomez, R., Gomez, L., Gibert, J., & Karatzas, D. "Learning to Learn from Web Data through Deep Semantic Embeddings" MULA 2018

Gomez, R., Gomez, L., Gibert, J., & Karatzas, D. "Self-Supervised Learning from Web Data for Multimodal Retrieval", arXiv:1901.02004, 2019

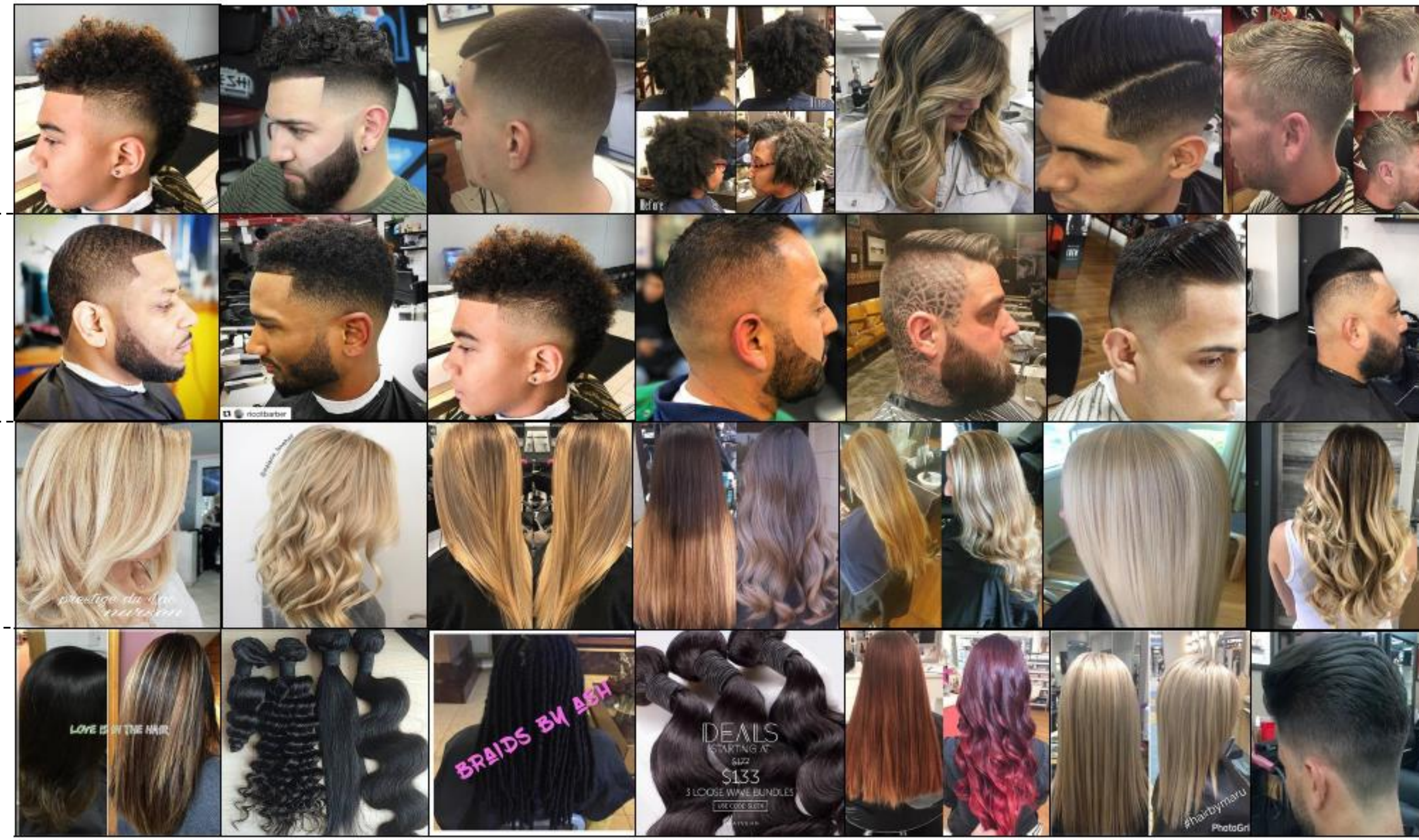
Text-based semantic retrieval

“haircut”

“haircut” + “man”

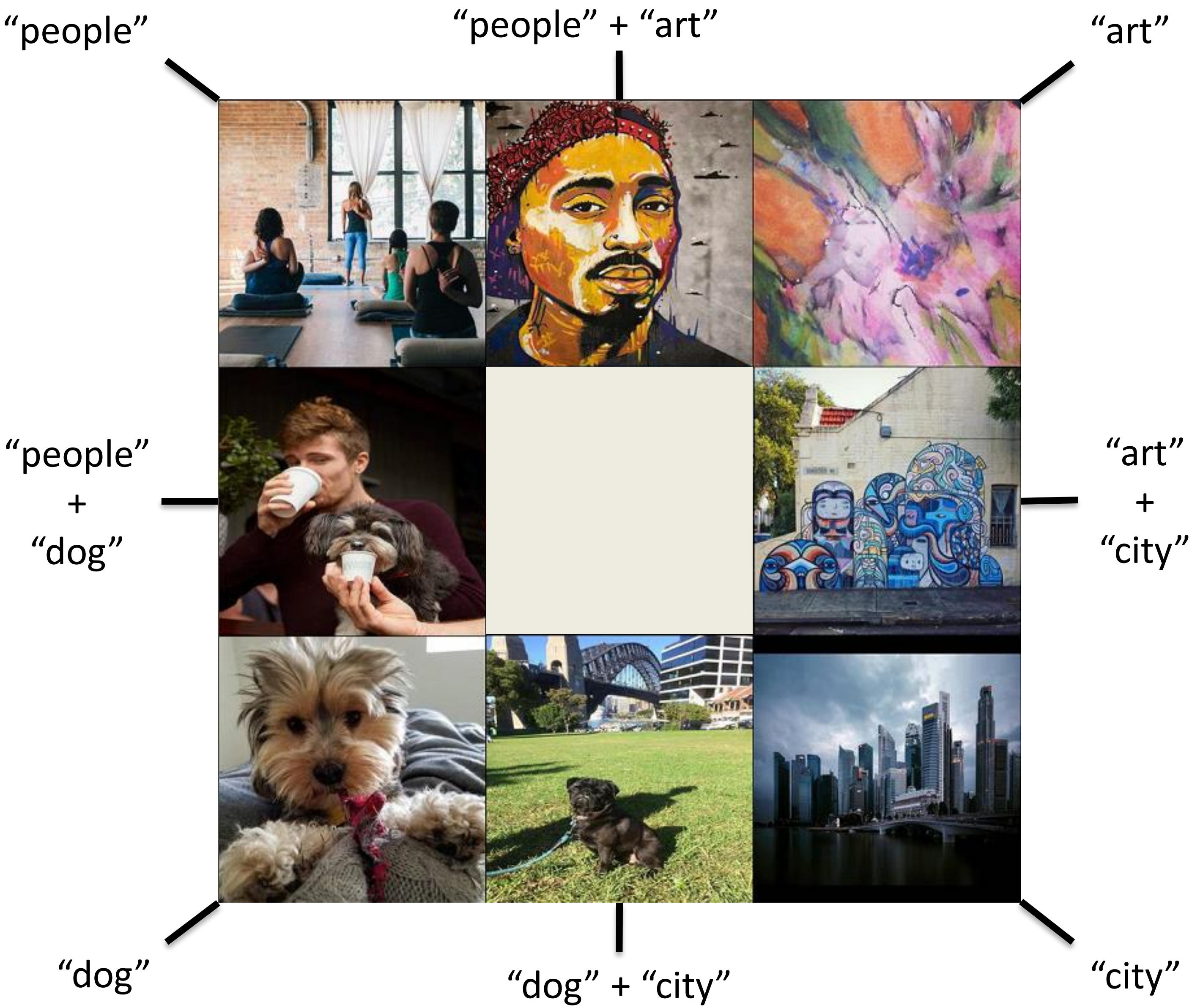
“haircut” + “woman”

“haircut” - “man”



Gomez, R., Gomez, L., Gibert, J., & Karatzas, D. “Learning to Learn from Web Data through Deep Semantic Embeddings” MULA 2018

Gomez, R., Gomez, L., Gibert, J., & Karatzas, D. “Self-Supervised Learning from Web Data for Multimodal Retrieval”, arXiv:1901.02004, 2019



Gomez, R., Gomez, L., Gibert, J., & Karatzas, D. "Learning to Learn from Web Data through Deep Semantic Embeddings" MULA 2018

Gomez, R., Gomez, L., Gibert, J., & Karatzas, D. "Self-Supervised Learning from Web Data for Multimodal Retrieval", arXiv:1901.02004, 2019

Gomez, R., Gomez, L., Gibert, J., & Karatzas, D. "Learning to Learn from Web Data through Deep Semantic Embeddings" MULA 2018

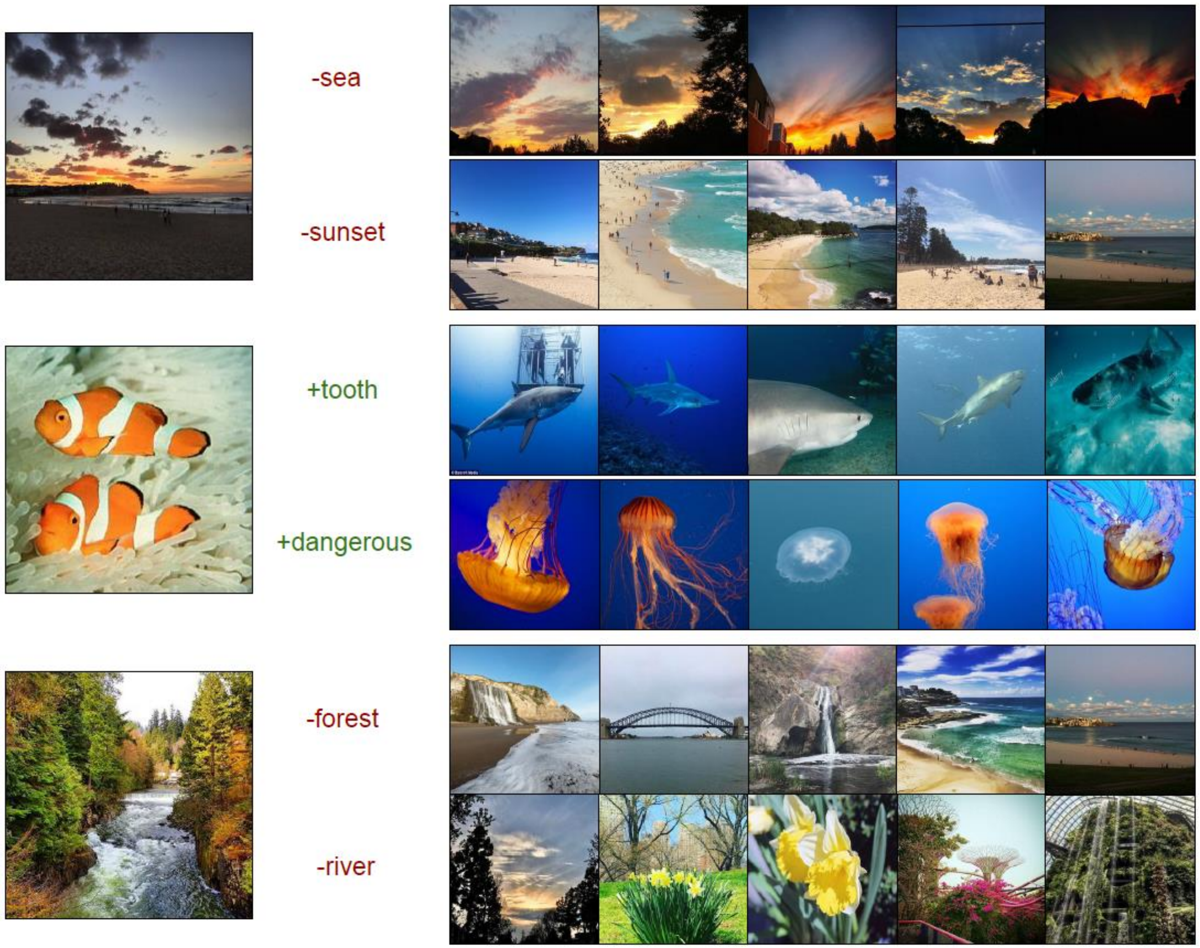
Gomez, R., Gomez, L., Gibert, J., & Karatzas, D. "Self-Supervised Learning from Web Data for Multimodal Retrieval", arXiv:1901.02004, 2019

-wedding

+animal

Gomez, R., Gomez, L., Gibert, J., & Karatzas, D. “Learning to Learn from Web Data through Deep Semantic Embeddings” MULA 2018

Gomez, R., Gomez, L., Gibert, J., & Karatzas, D. “Self-Supervised Learning from Web Data for Multimodal Retrieval”, arXiv:1901.02004, 2019

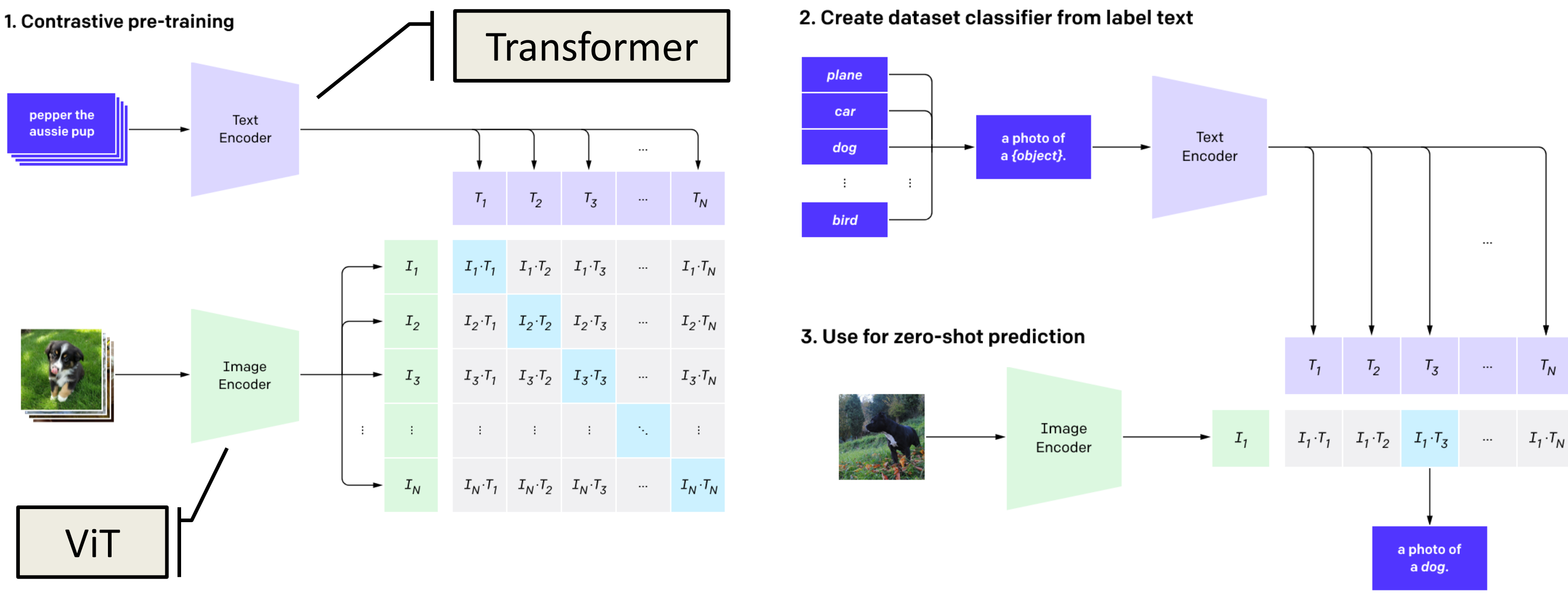


Gomez, R., Gomez, L., Gibert, J., & Karatzas, D. "Learning to Learn from Web Data through Deep Semantic Embeddings" MULA 2018

Gomez, R., Gomez, L., Gibert, J., & Karatzas, D. "Self-Supervised Learning from Web Data for Multimodal Retrieval", arXiv:1901.02004, 2019

CLIP: Contrastive Language-Image Pre-training

Key take away: (1) Contrastive objectives can learn better representations than their equivalent predictive objective, (2) scale matters



Training: 10 GPU years (256 GPUs for 2 weeks, Batch size: of 32,768)

A. Radford et al, "Learning Transferable Visual Models From Natural Language Supervision", 2021

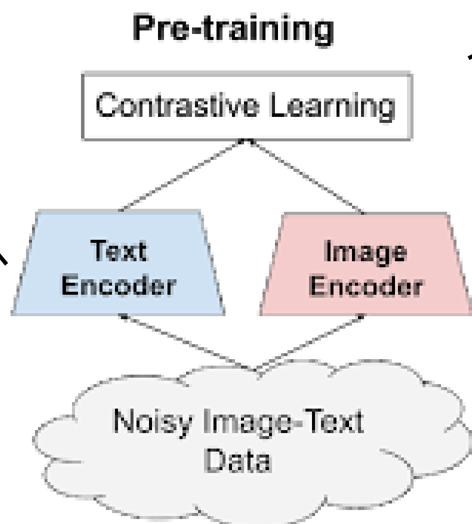
ALIGN

Nothing special… just taking advantage of large-scale data. Showed they can train from scratch and align two state of the art encoders

$$L_{i2t} = -\frac{1}{N}\sum_{i}^{N} log \frac{exp\left(x_i^T y_i/\sigma\right)}{\sum_{j=1}^{N} exp\left(x_i^T y_j/\sigma\right)}$$

$$L_{t2i} = -\frac{1}{N}\sum_{i}^{N} log \frac{exp\left(y_i^T x_i/\sigma\right)}{\sum_{j=1}^{N} exp\left(y_i^T x_j/\sigma\right)}$$

BERT



EfficientNet-L2

1.8B image – alt text pairs from the Web

Training: on 1024 Cloud TPUv3 (batch size: 16,384)

C. Jia et al, "Scaling Up Visual and Vision-Language Representation Learning With Noisy Text Supervision", ICML 2021