Topics:

- Masked Language Models
- Embeddings

CS 4803-DL / 7643-A

ZSOLT KIRA

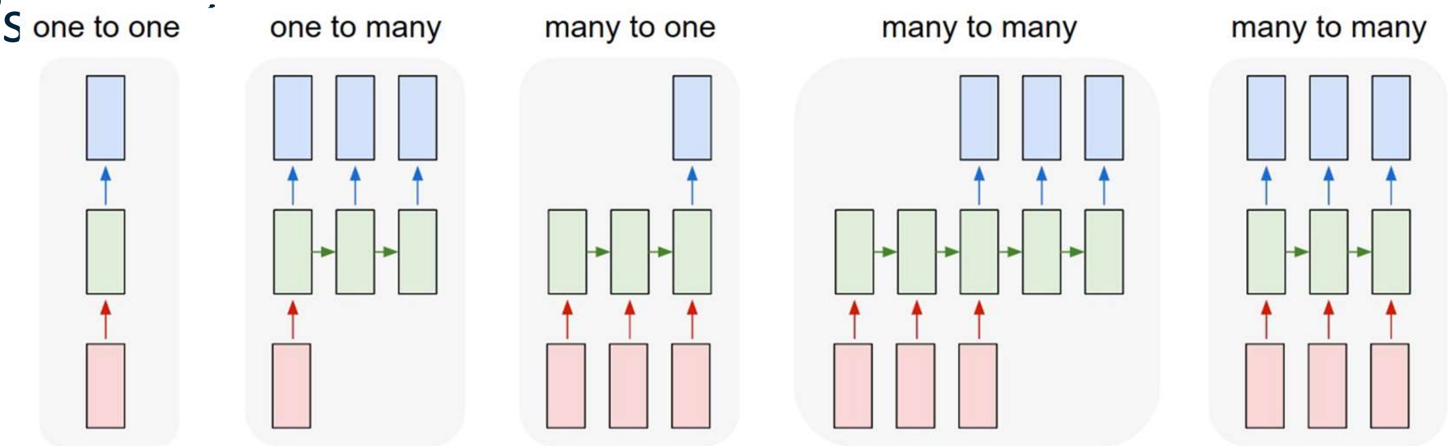
- **Assignment 4 out**
  - Due date **extended** to **April 8th 11:59pm EST.**

- **Projects**
  - Project proposal due **March 22nd 11:59pm EST**
  - FB discourse forum released!

- Outline of rest of course:
  - No class March 24th (“spring break” day)
  - March 26th we start (deep) reinforcement learning
  - Guest lectures/other topics (e.g. self-supervised learning)
    - Ishan Misra (FB) April 9th
  - Generative models (VAEs / GANs)

Sequences in Input or Output?

- It's



Input: No sequence

Output: No sequence

Example: "standard" classification / regression problems

Input: No sequence

Output: Sequence

Example: Im2Caption

Input: Sequence

Output: No sequence

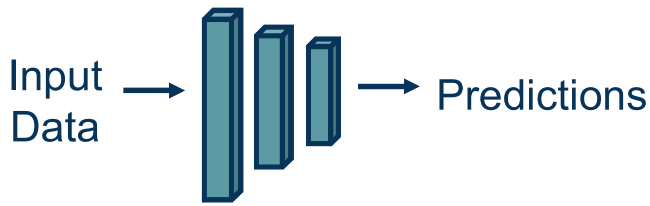
Example: sentence classification, multiple-choice question answering

Input: Sequence

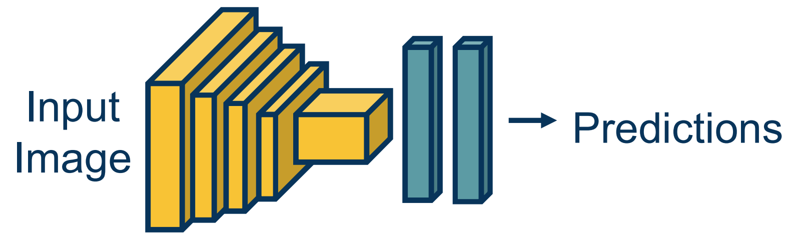
Output: Sequence

Example: machine translation, video classification, video captioning, open-ended question answering

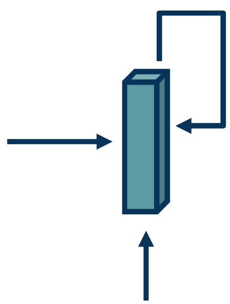
Image Credit: Andrej Karpathy



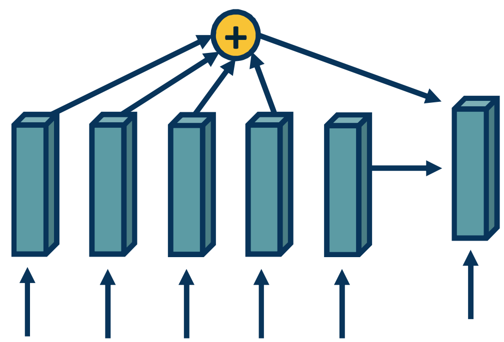
Fully Connected Neural Networks



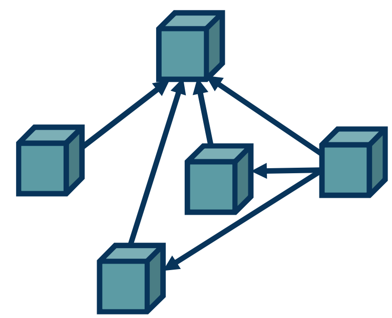
Convolutional Neural Networks



Recurrent Neural Networks

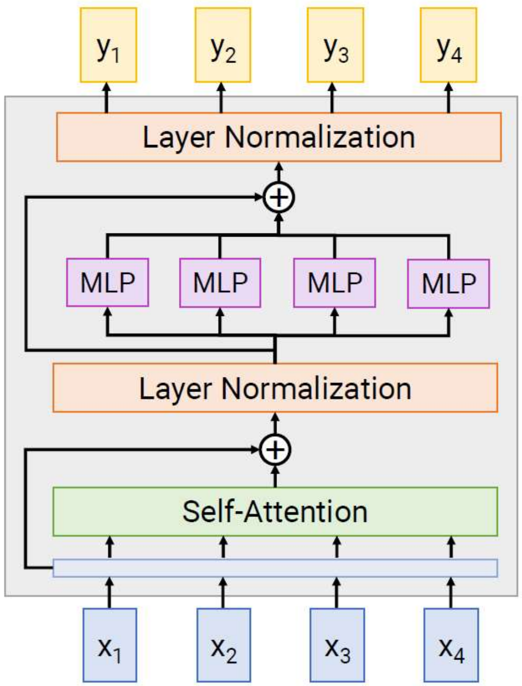


Attention-Based Networks

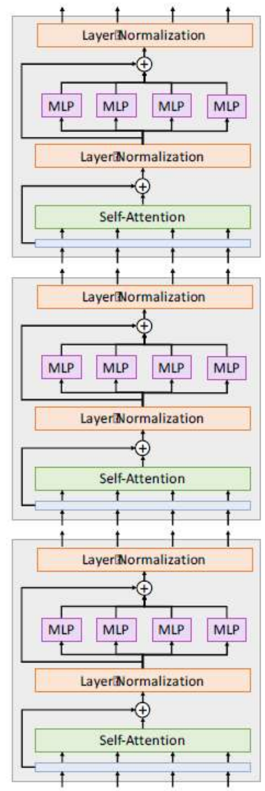


Graph-Based Networks

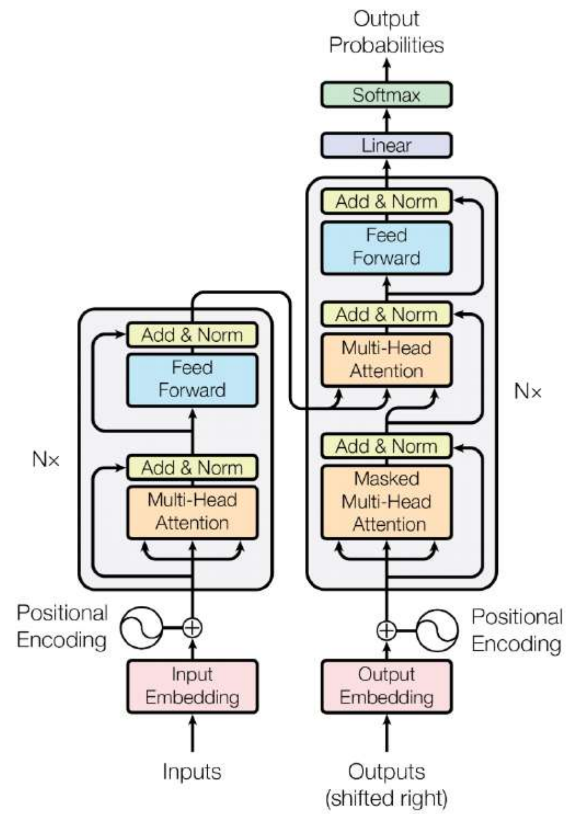
The Space of Architectures



Transformer Block



Multi-Layered



Encoder/Decoder

Recall: Transformers

Georgia Tech

Masked Language Models

FACEBOOK AI Georgia Tech

Jean Maillard

Jean Maillard is a Research Scientist on the Language And Translation Technologies Team (LATTE) at Facebook AI. His research interests within NLP include word- and sentence-level semantics, structured prediction, and low-resource languages. Prior to joining Facebook in 2019, he was a doctoral student with the NLP group at the University of Cambridge, where he researched compositional semantic methods. He received his BSc in Theoretical Physics from Imperial College London.

- **Recall:** language models estimate the probability of sequences of words:

$$\mathrm{p}(\boldsymbol{s}) = \mathrm{p}(w_1, w_2, \ldots, w_n)$$

- ***Masked language modeling*** is a related ***pre-training task*** – an auxiliary task, different from the final task we're really interested in, but which can help us achieve better performance by finding good initial parameters for the model.

- By pre-training on masked language modeling before training on our final task, it is usually possible to obtain higher performance than by simply training on the final task.

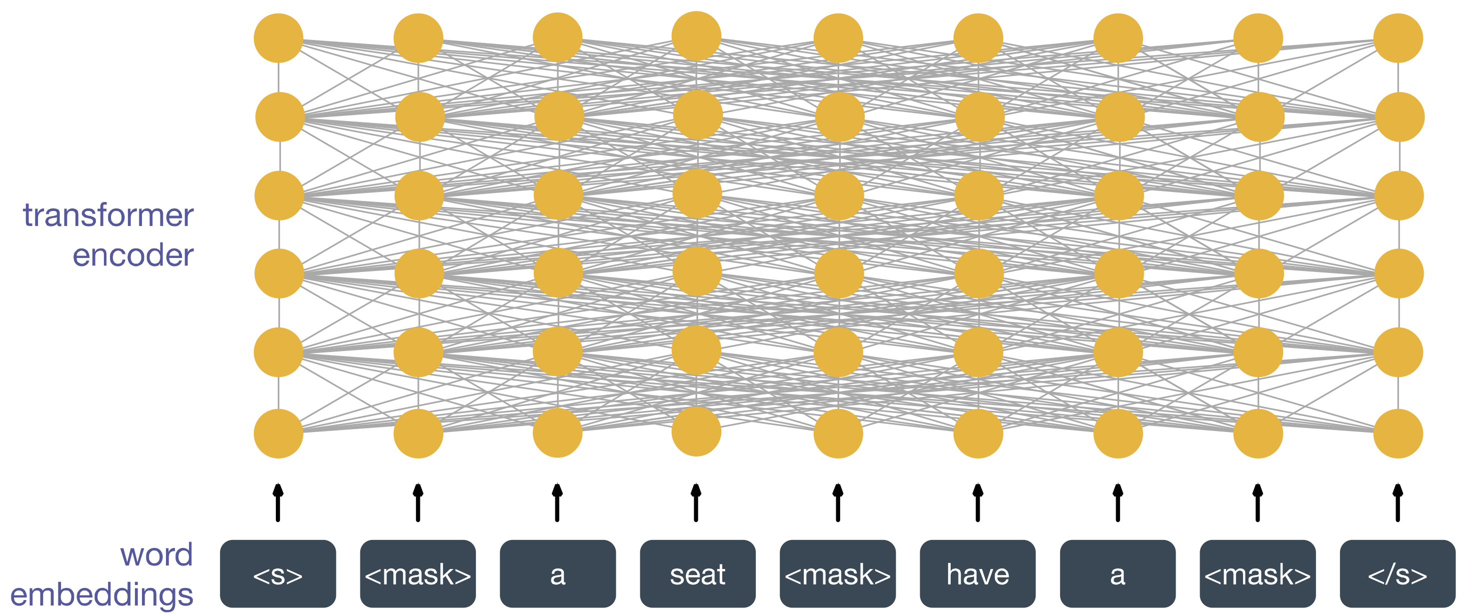
take | a | seat | , | have | a | drink

Masked Language Models

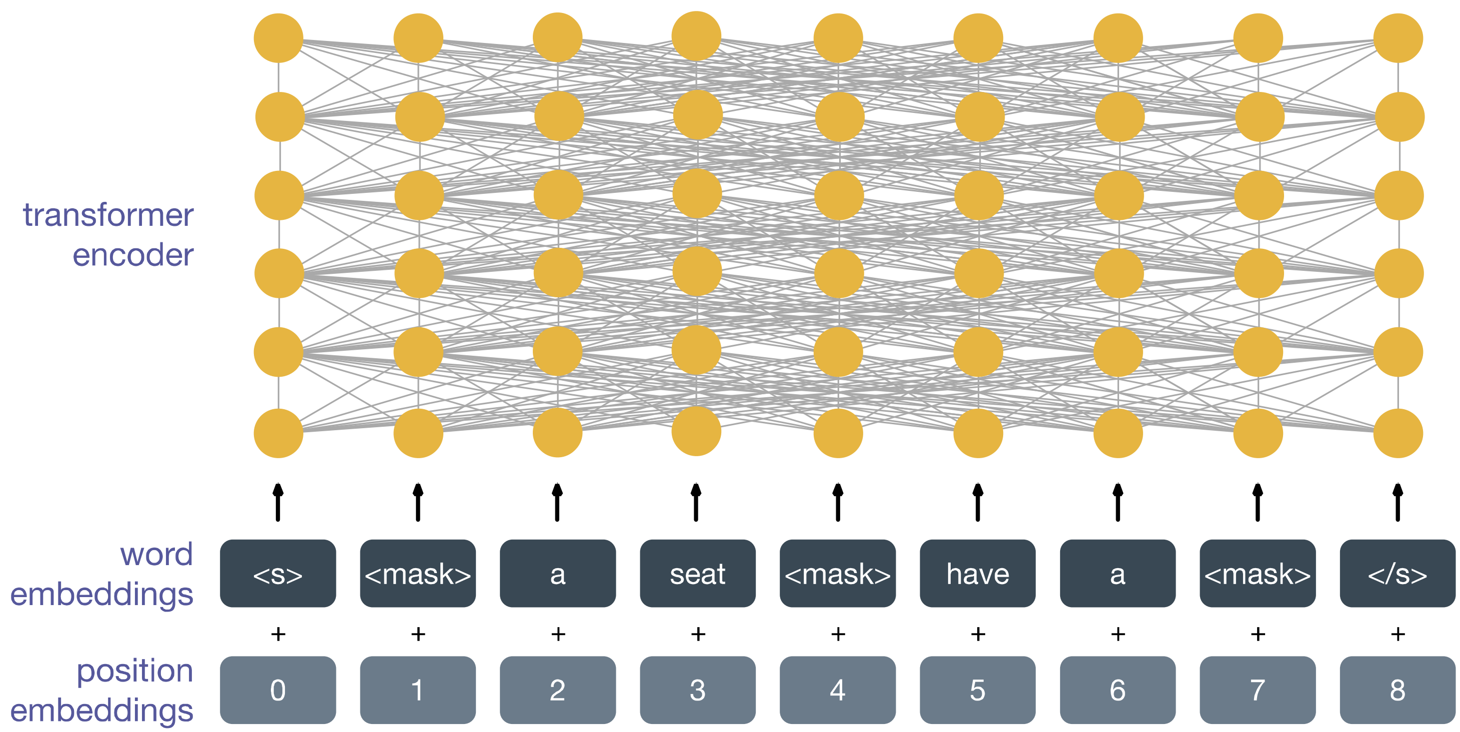
FACEBOOK AI Georgia Tech

<s> <mask> a seat <mask> have a <mask> </s>

Masked Language Models

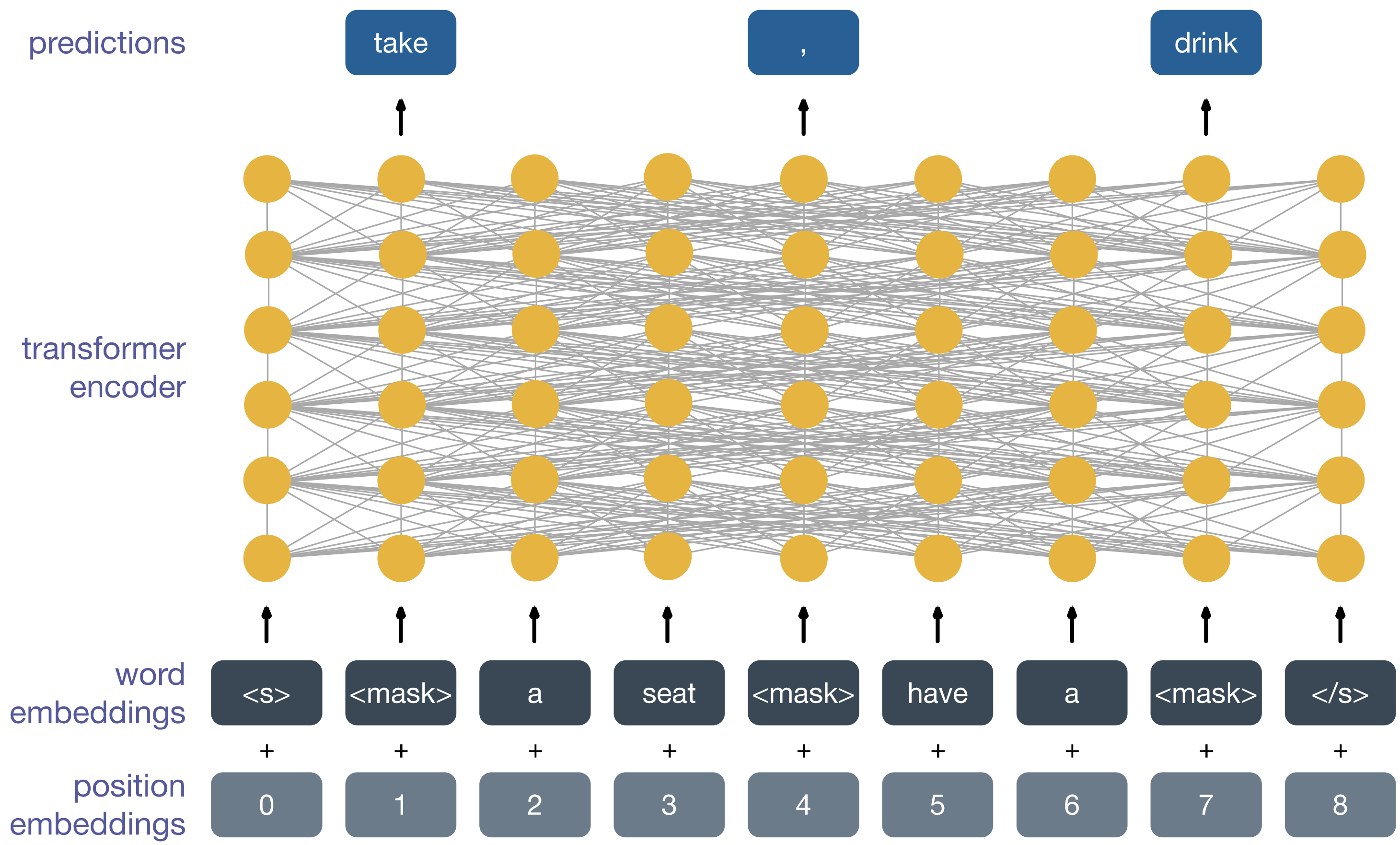


Masked Language Models

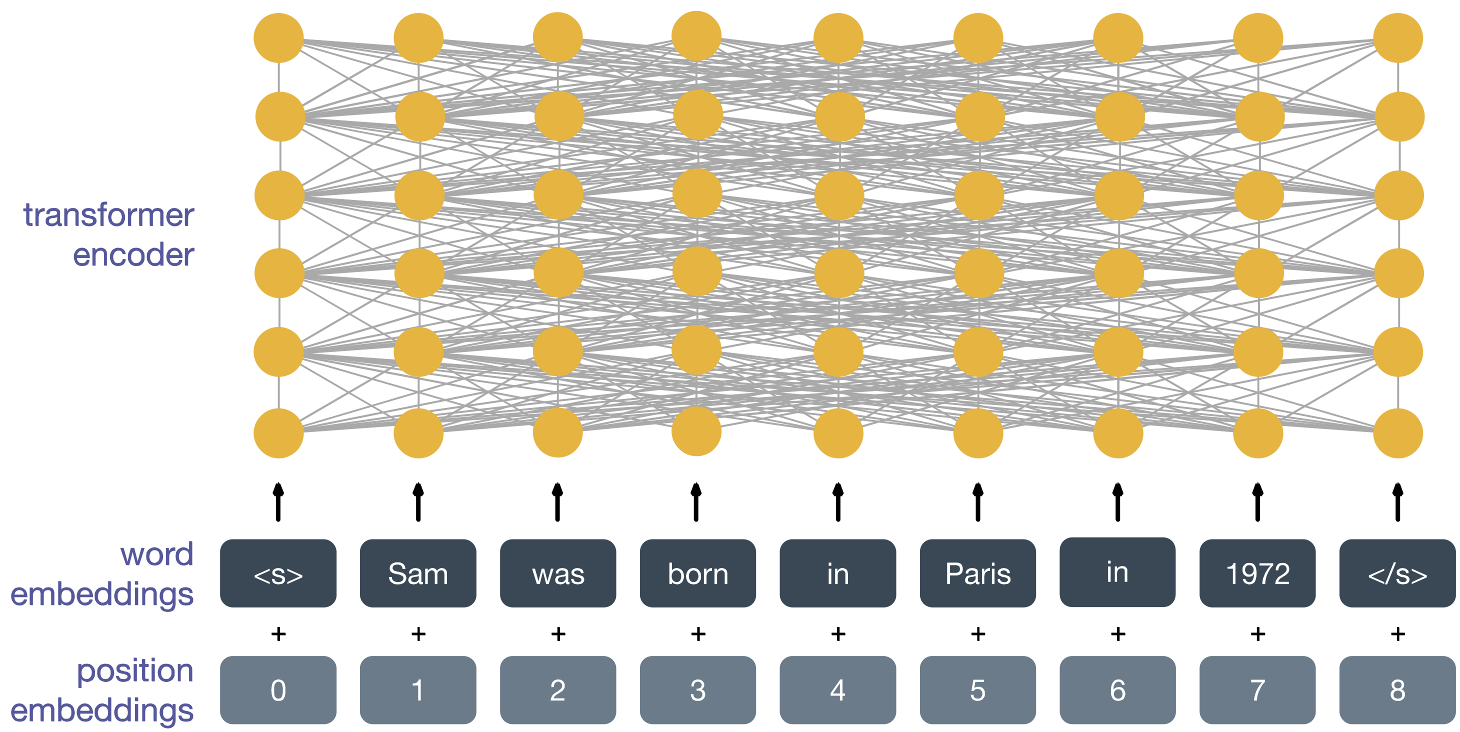


Masked Language Models

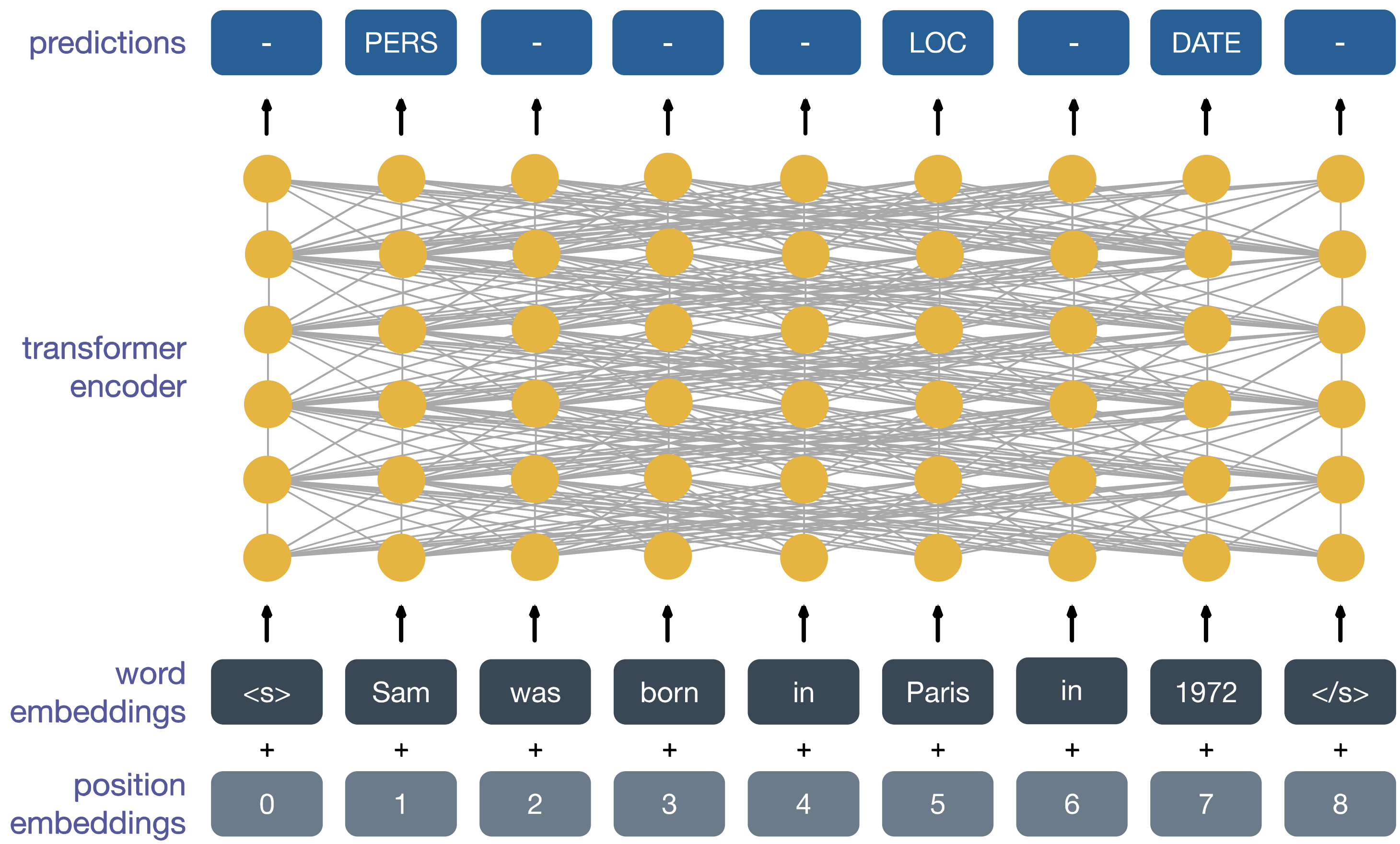
FACEBOOK AI Georgia Tech



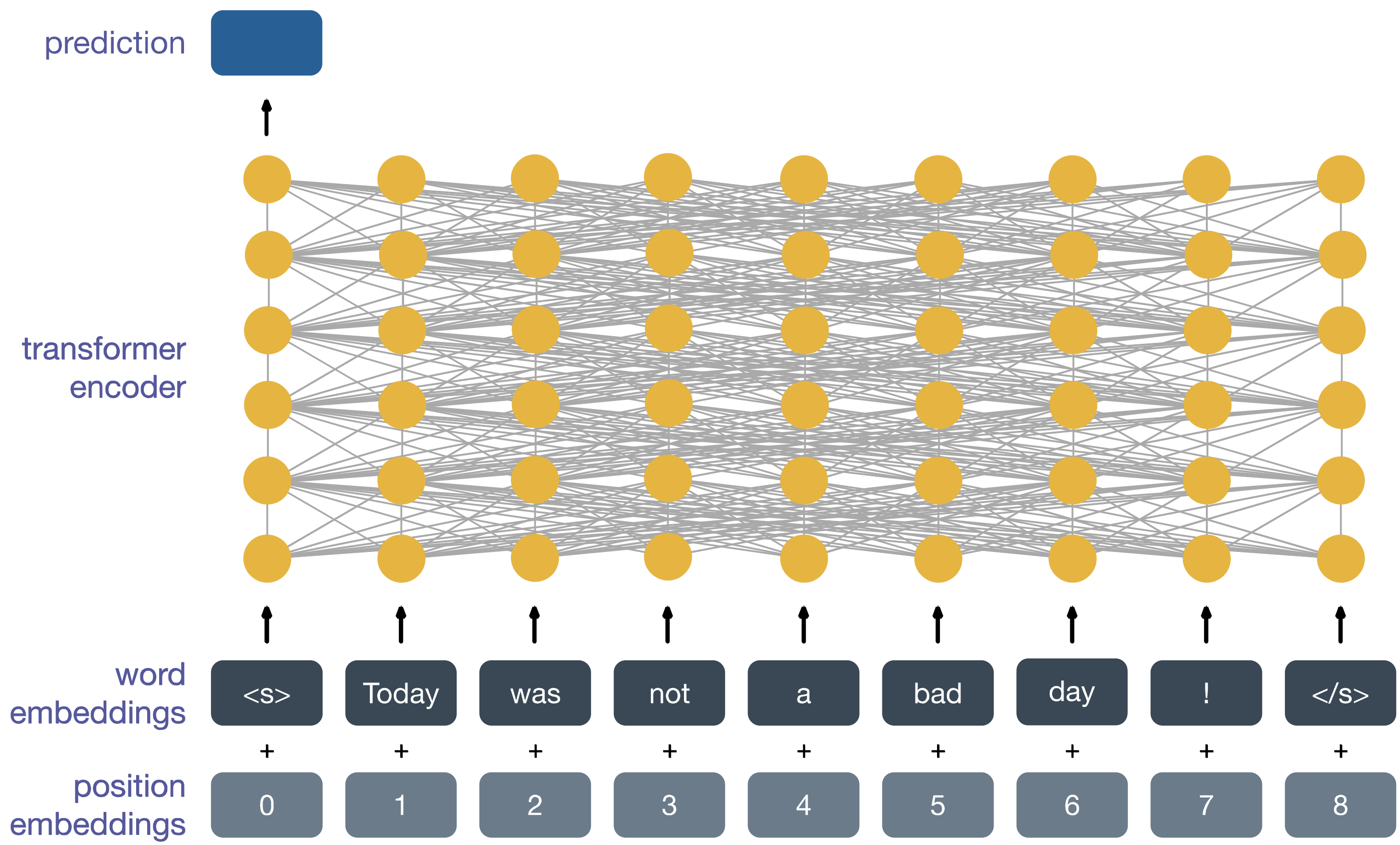
Masked Language Models



Token-level Tasks

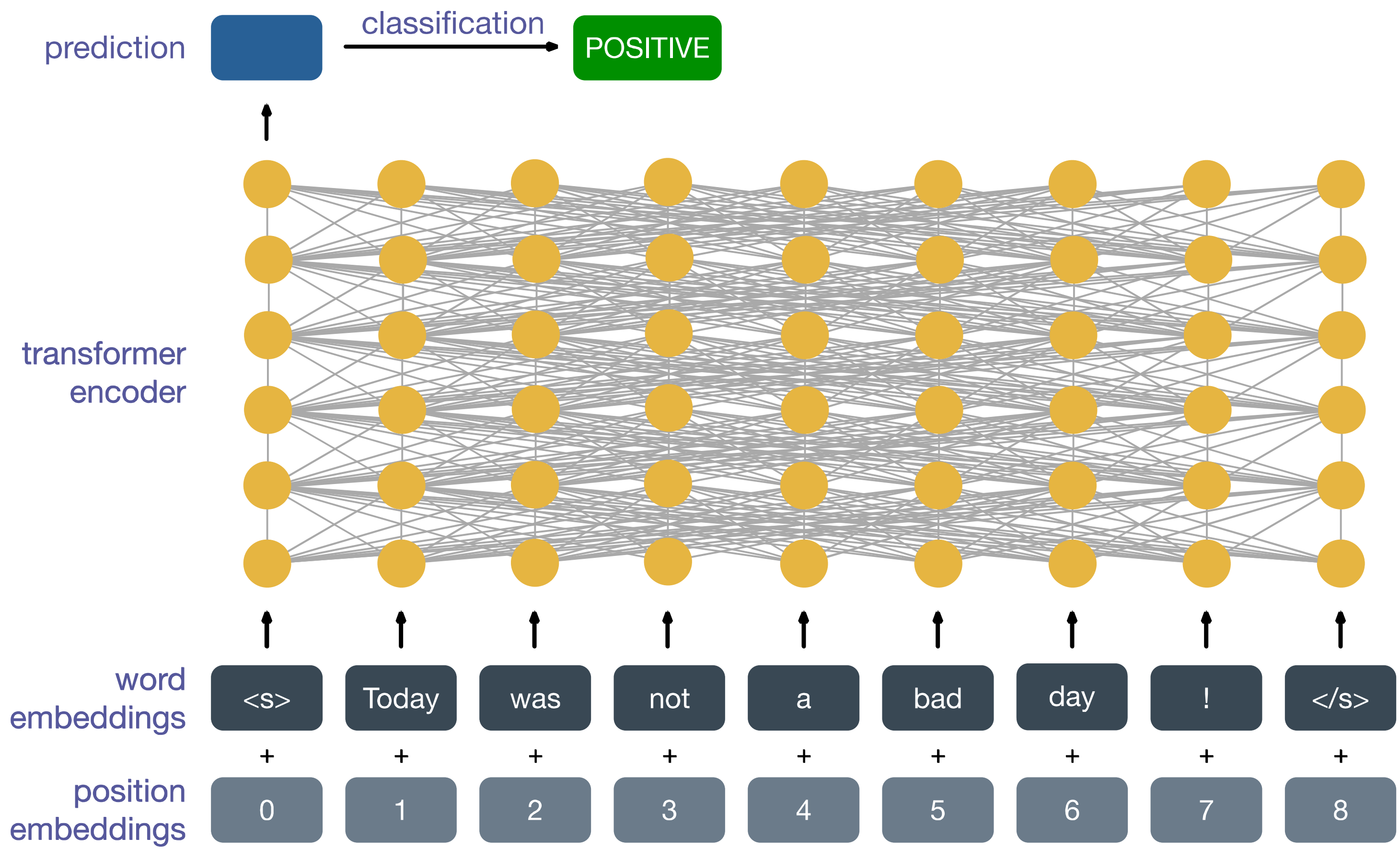


Token-level Tasks



Sentence-level Tasks

FACEBOOK AI Georgia Tech



Sentence-level Tasks

I am hungry J’ ai faim

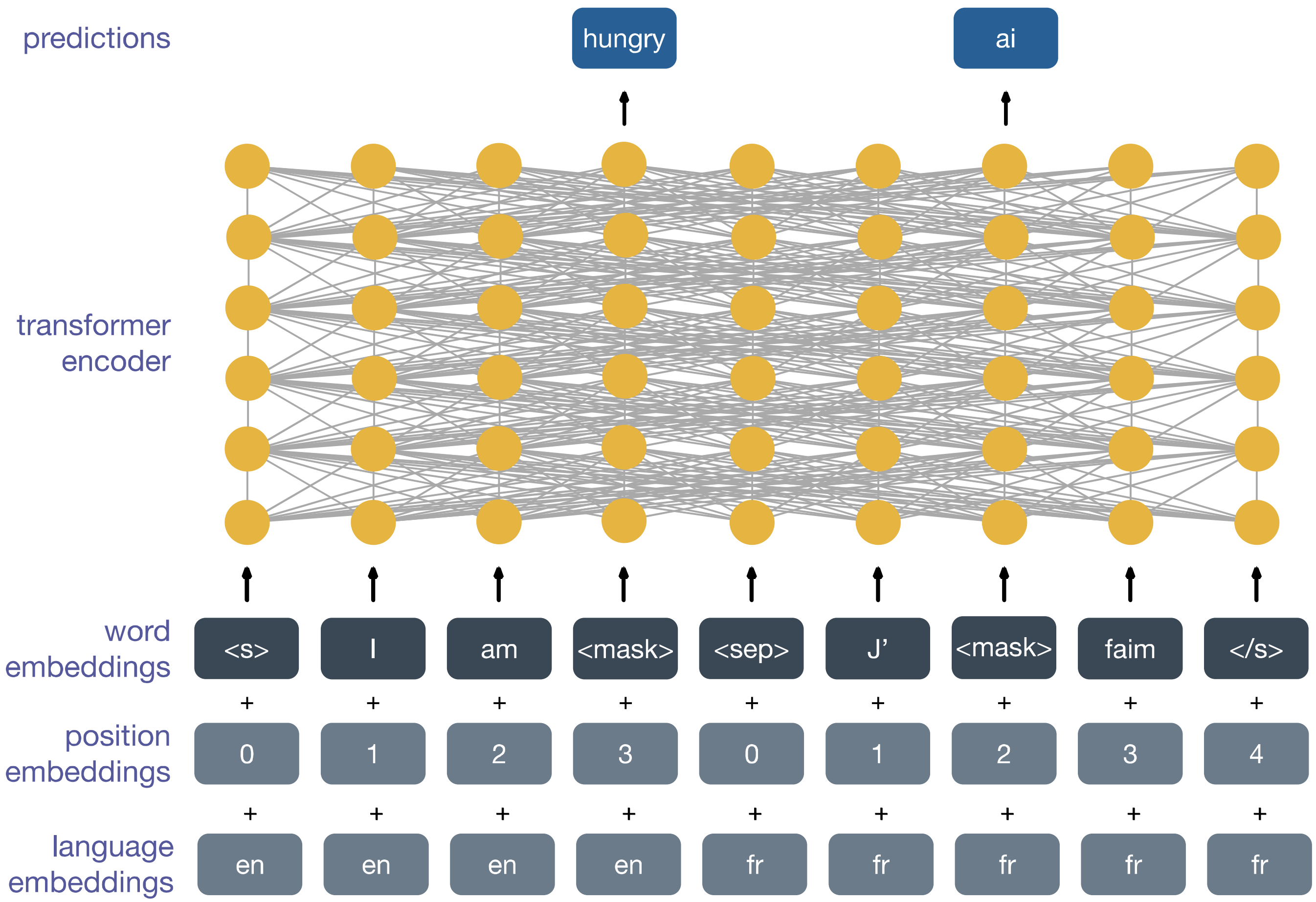
Cross-lingual Masked Language Modeling

FACEBOOK AI Georgia Tech

<s> | I | am | <mask> | <sep> | J’ | <mask> | faim | </s>

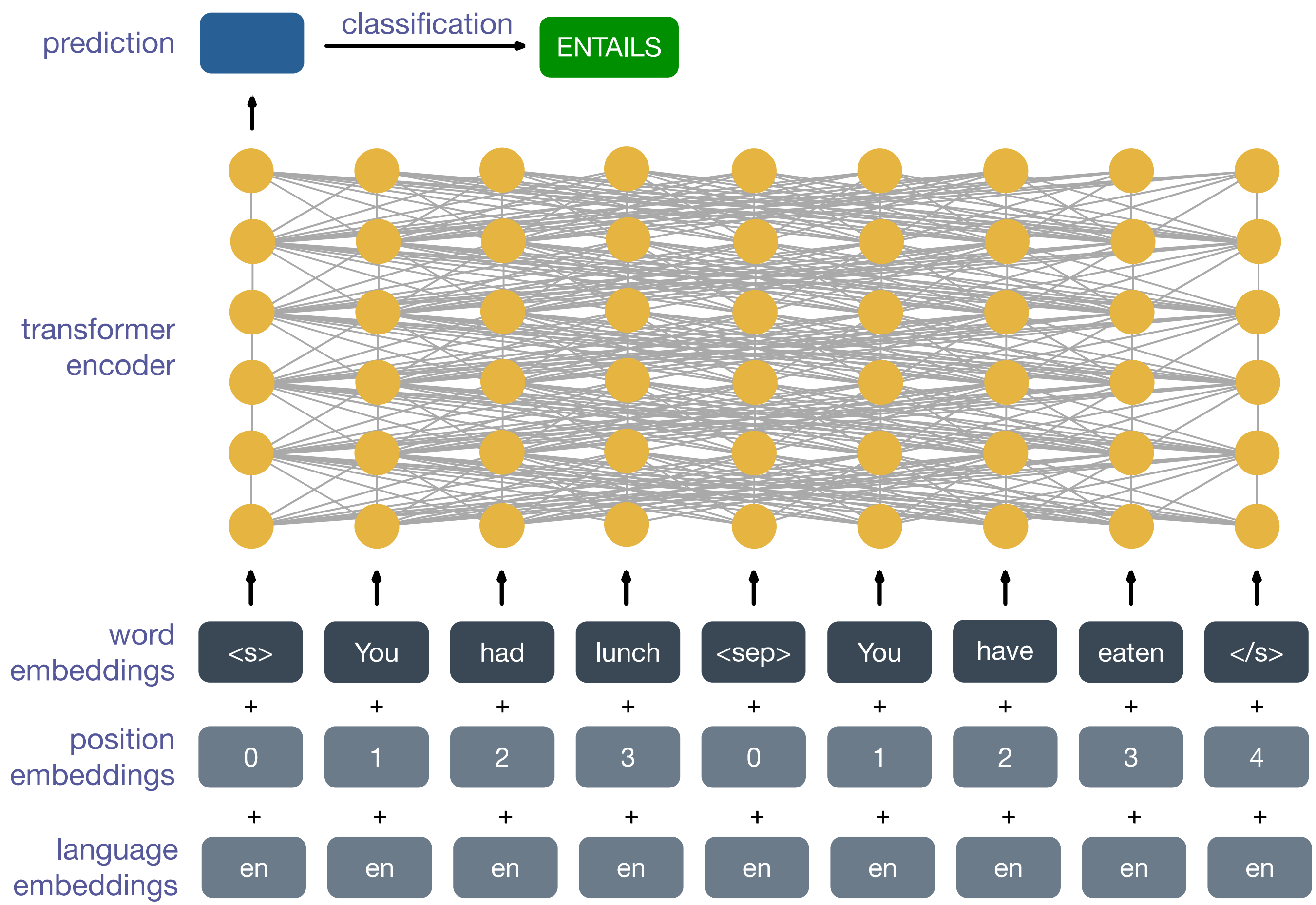
Cross-lingual Masked Language Modeling

FACEBOOK AI Georgia Tech

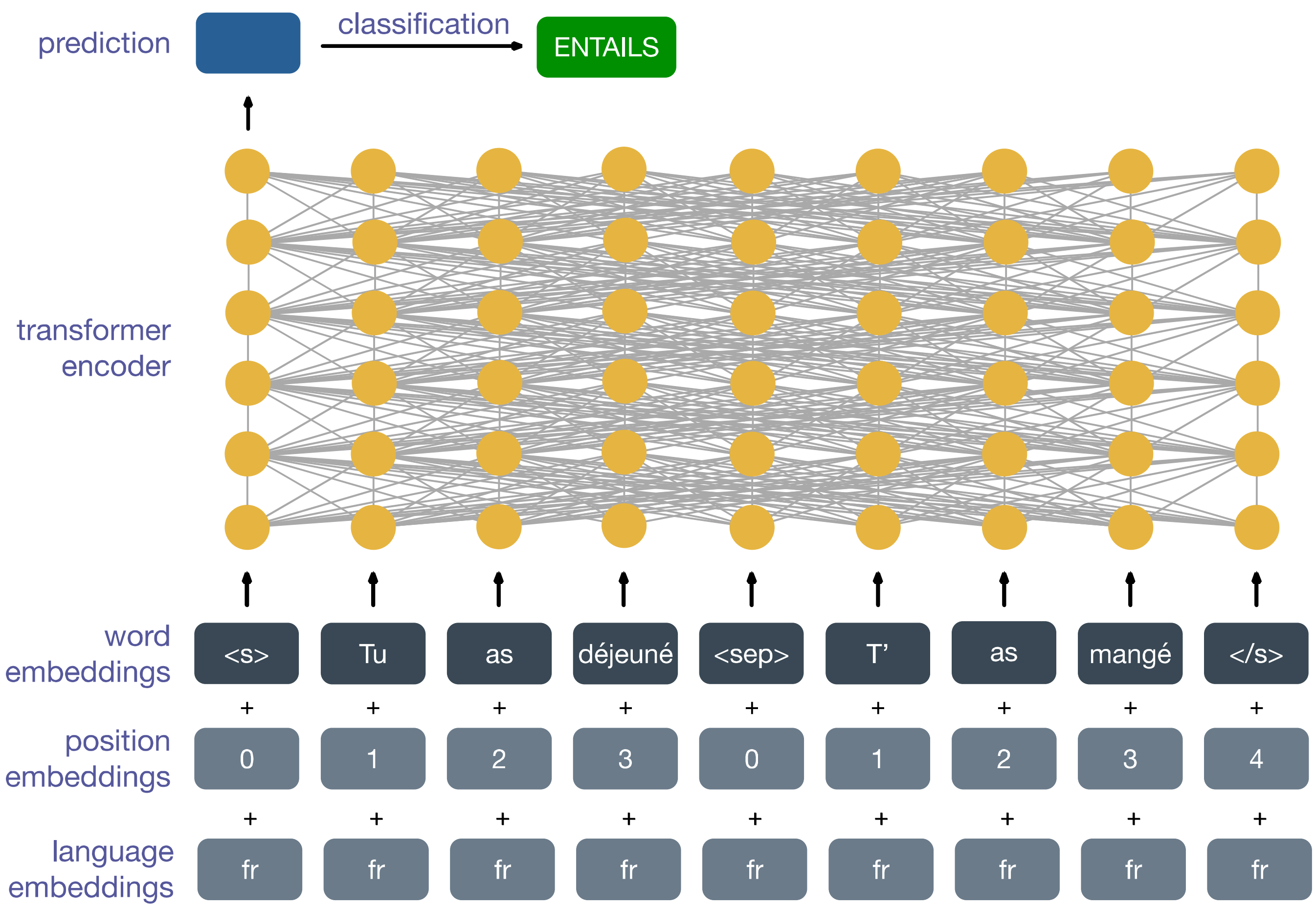


Cross-lingual Masked Language Modeling

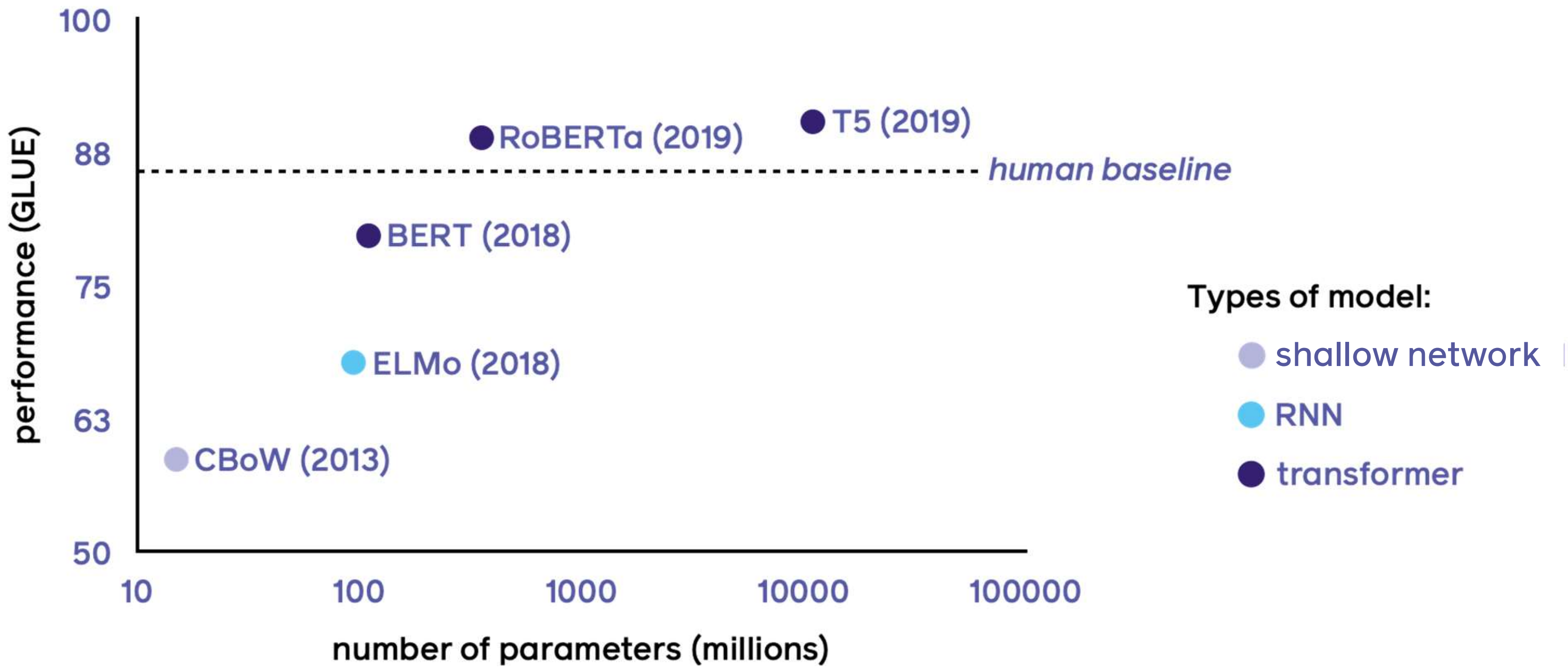
FACEBOOK AI Georgia Tech



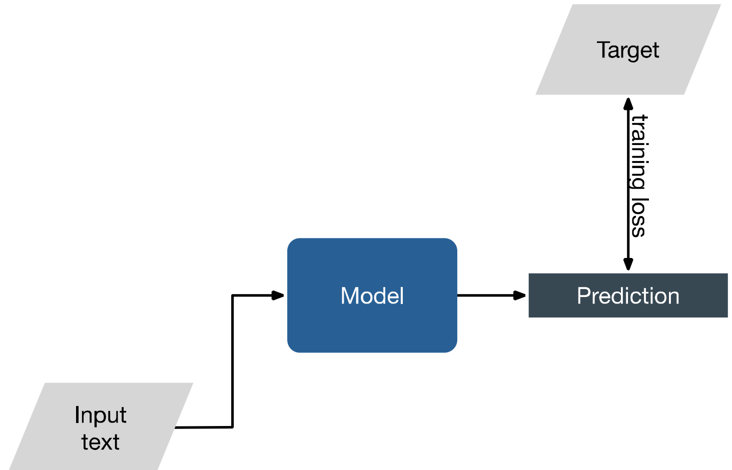
Cross-lingual Task: Natural Language Inference

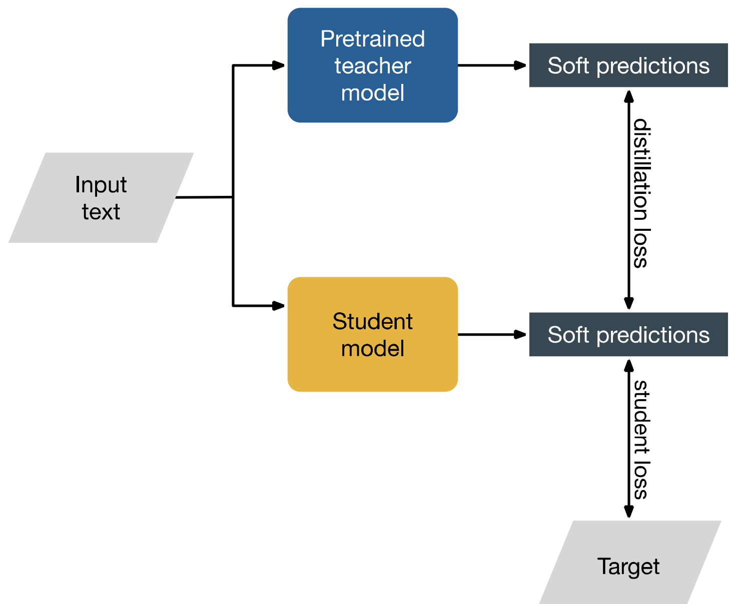


Cross-lingual Task: Natural Language Inference

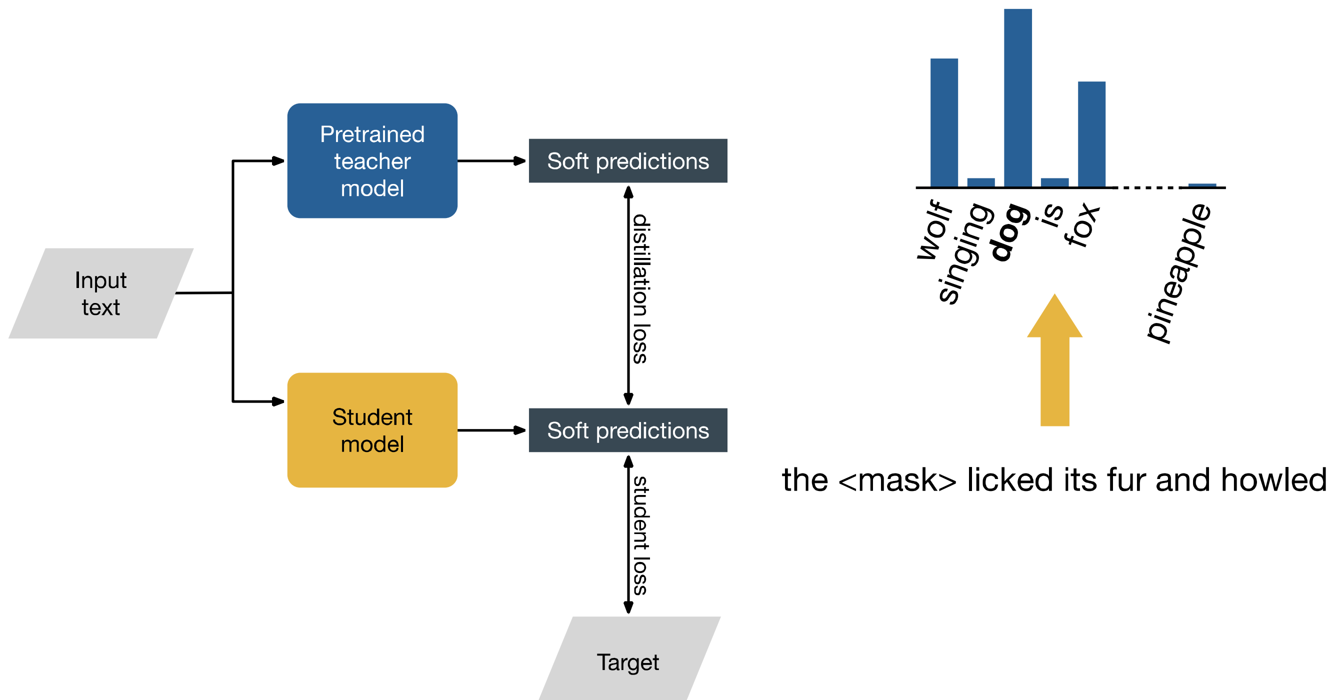


Model Size in Perspective


Target
training loss
Model
Prediction
Input
text



Knowledge Distillation to Reduce Model Sizes



Knowledge Distillation to Reduce Model Sizes

cross-entropy $$\mathrm{H}(p^*, p) = -\sum_{x \in \mathcal{X}} p^*(x) \log p(x)$$

reference distribution

$$\mathcal{L}_{\text{dist}} = \mathrm{H}(t, s) = -\sum_i t_i \log s_i \quad \text{or} \quad \mathrm{D}_{\mathrm{KL}}(t \| s)$$

$$\mathcal{L}_{\text{student}} = \mathrm{H}(y, s) = -\sum_i y_i \log s_i$$

$$\mathcal{L} = \alpha \mathcal{L}_{\text{dist}} + \beta \mathcal{L}_{\text{student}}$$

- Vaswani et al. (2017). "Attention is all you need", in *NIPS 2017*.
- Devlin et al. (2018). "BERT: pre-training of deep bidirectional transformers for language understanding".
- Liu, Ott, Goyal, Du, et al. (2019). "RoBERTa: a robustly optimized BERT pretraining approach".
- Lample & Conneau (2019). "Cross-lingual language model pretraining", in NeurIPS 2019.
- Conneau, Khandelwal, et al. (2020). "Unsupervised cross-lingual representation learning at scale", in *ACL 2020*.
- Lewis, Liu, Goyal, et al. (2019). "BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension", in *ACL 2020*.
- Raffel, Shazeer, Roberts, Lee, et al. (2020), "Exploring the limits of transfer learning with a unified text-to-text transformer", in *JMLR* 21(2020): 1-67.
- Hinton, Vinyals, Dean (2015). "Distilling the knowledge in a neural network", in *NIPS 2014 deep learning workshop*.

References

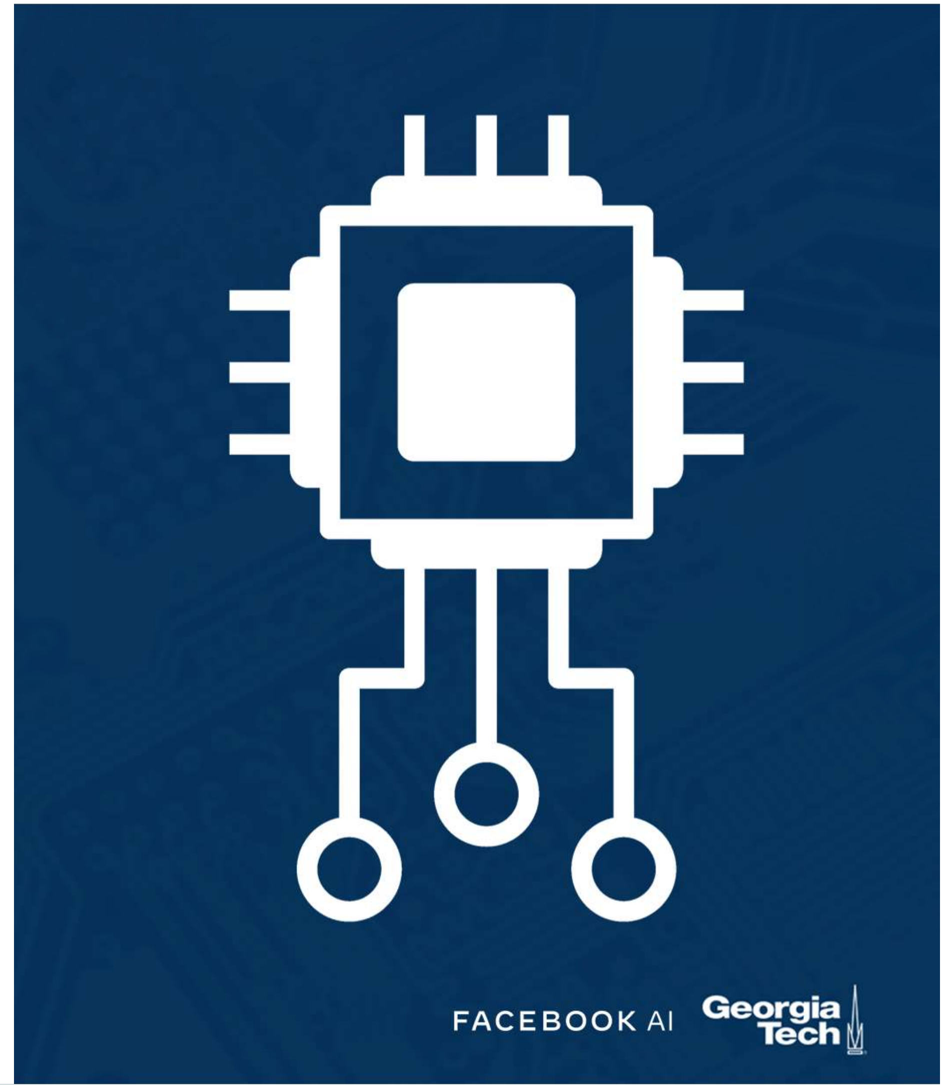
Ledell Wu

Ledell Wu is a research engineer at Facebook AI Research. Ledell joined Facebook in 2013 after graduating from University of Toronto. She worked on Newsfeed ranking as a machine learning engineer. After joining Facebook AI, Ledell worked on general purpose and large-scale embedding systems. She collaborated with teams including page recommendations, video recommendations, ads interest suggestion, people search and feed integrity, to use embeddings to better serve products. She is one of the main contributors in open source projects including StarSpace (general purpose embedding system), PyTorch Big-Graph (large-scale graph embedding system) and BLINK (entity linking). Ledell also studies fairness and biases in machine learning models.

Lecturer Introduction

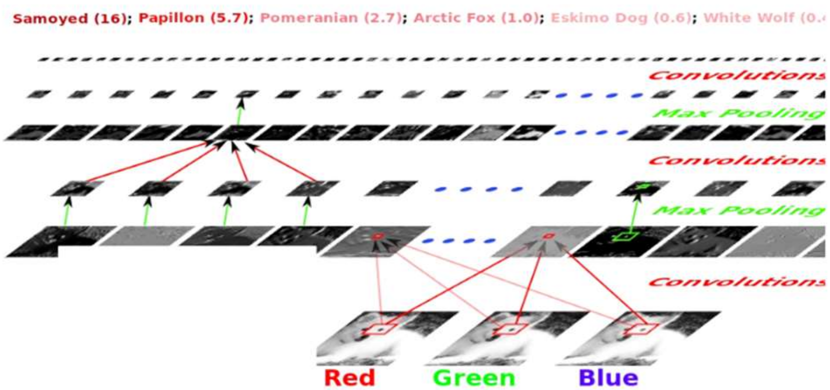
Embeddings

- Word Embeddings
- Graph Embeddings
- Applications, world2vec
- Additional Topics



- **Mapping Objects to Vectors through a trainable function**

[0.4, -1.3, 2.5, -0.7, …]

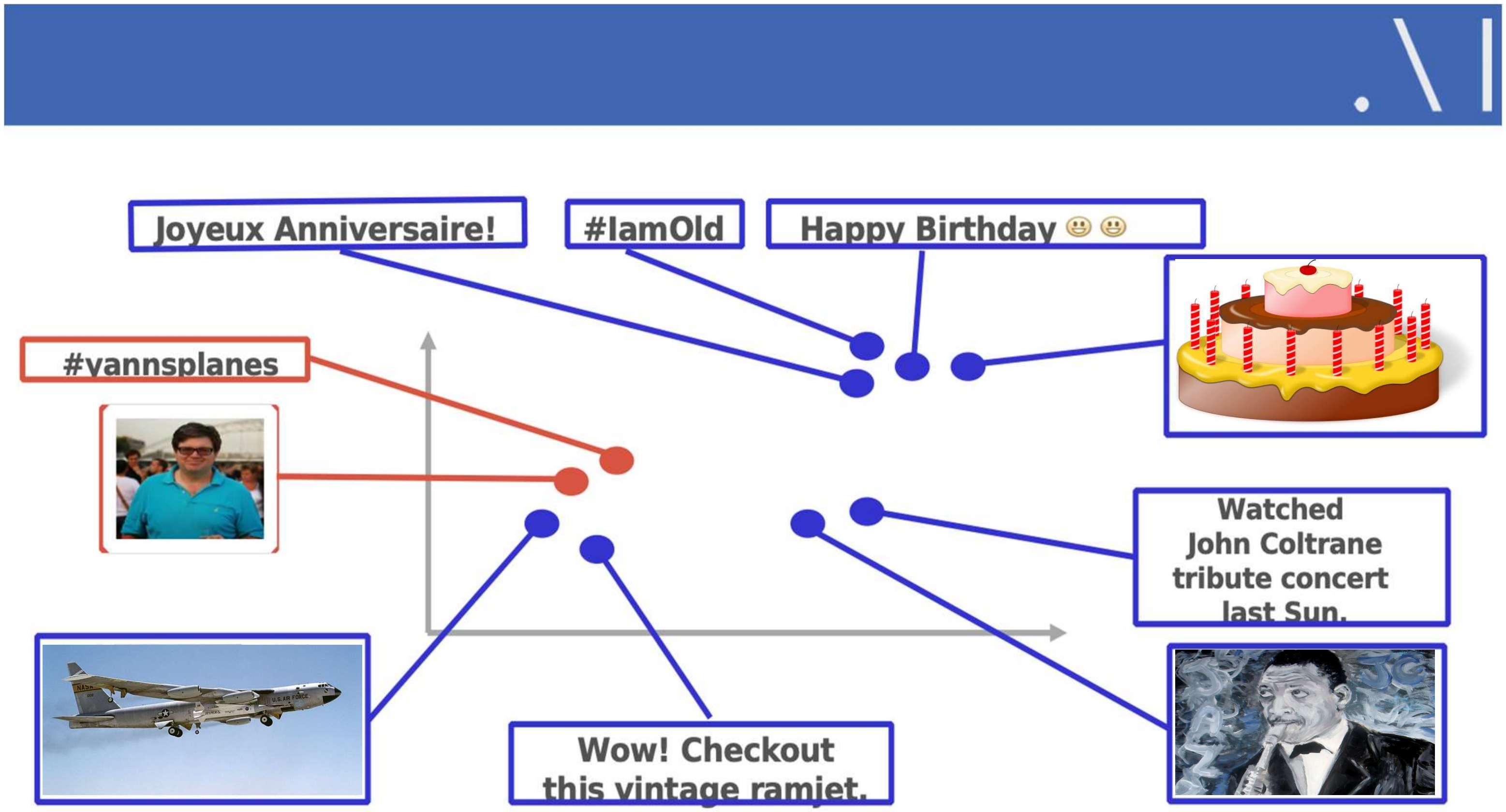


[0.2, -2.1, 0.4, -0.5, …]

Neural Net

"The neighbors' dog was a Samoyed, which looks a lot like a Siberian husky"

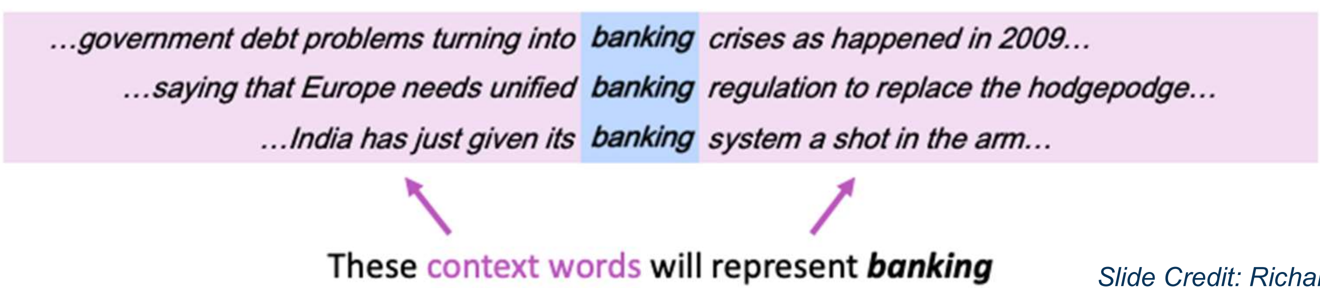
Slide Credit: Yann LeCun



Slide Credit: Yann LeCun

Representing words by their context

- Distributional semantics: **A word’s meaning is given by the words that frequently appear close-by**
  - *“You shall know a word by the company it keeps”* (J.R.Firth 1957:11)
  - One of the most successful ideas of modern statistical NLP!
- When a word w appears in a text, its **context** is the set of words that appear nearby (within a fixed-size window).
- Use the many contexts of ***w*** to build up a representation of ***w***



Slide Credit: Richard Socher, Christopher Manning

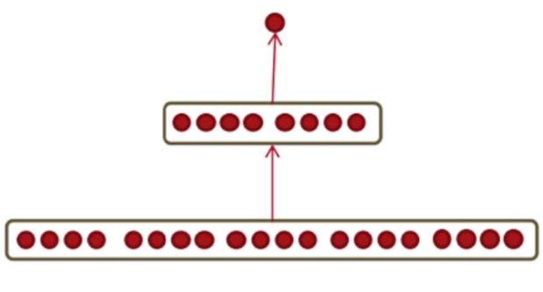
Collobert & Weston vectors

- **Idea:** a word and its context is a positive training sample; a random word in that sample context gives a negative training sample:

cat chills **on** a mat   cat chills Ohio a mat

score(cat chills on a mat) > score(cat chills Ohio a mat)

$$\boxed{J = \max(0, 1 - s + s_c)}$$



$$\begin{aligned} s &= U^T a \\ a &= f(z) \\ z &= Wx + b \end{aligned}$$

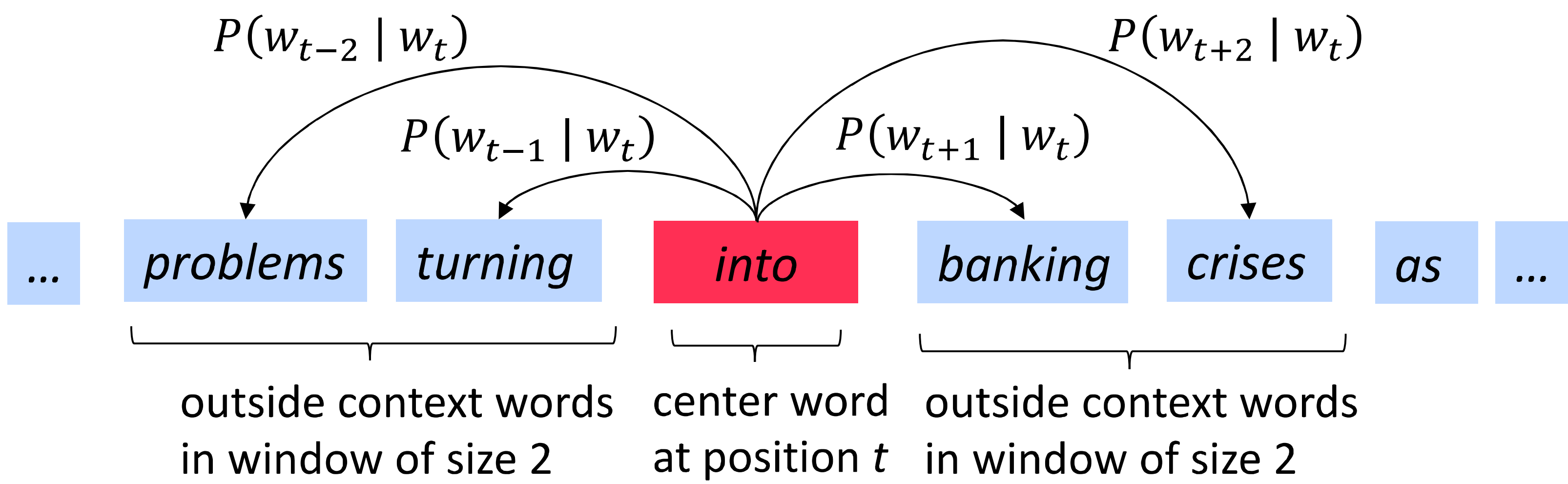
$$x = [x_{cat} \quad x_{chills} \quad x_{on} \quad x_a \quad x_{mat}]$$

$$L \in \mathbb{R}^{n \times |V|}$$

Slide Credit: Danqi Chen

Word2vec: the Skip-gram model

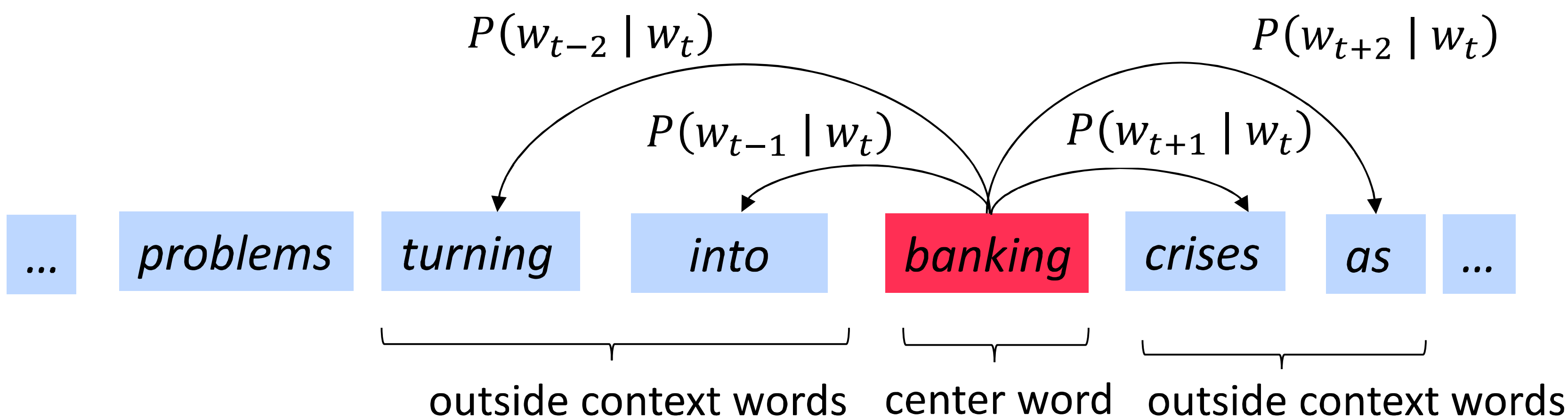
- The idea: use words to **predict** their context words
- Context: a fixed window of size ***2m***



Slide Credit: Richard Socher, Christopher Manning

Word2vec: the Skip-gram model

- The idea: use words to **predict** their context words
- Context: a fixed window of size ***2m***



Slide Credit: Richard Socher, Christopher Manning

Skip-gram Objective function

- For each position t = 1,...,T, predict context words within a window of fixed size m, given center word w_j:

all the parameters to be optimized

$$\mathcal{L}(\theta) = \prod_{t=1}^{T} \prod_{-m \leq j \leq m, j \neq 0} P(w_{t+j} \mid w_t; \theta)$$

- The objective function is the (average) negative loglikelihood:

$$J(\theta) = -\frac{1}{T} \log \mathcal{L}(\theta) = -\frac{1}{T} \sum_{t=1}^{T} \sum_{-m \leq j \leq m, j \neq 0} \log P(w_{t+j} \mid w_t; \theta)$$

Slide Credit: Richard Socher, Christopher Manning

How to define $P(w_{t+j} / w_t ; \theta)$?

- We have two sets of vectors for each word in the vocabulary:
  - u_w when w is a center word
  - v_o when o is a context word
- Use inner product (u_w, v_o) to measure how likely word w appears with context word o:

$$P(w_{t+j} \mid w_t) = \frac{\exp(\mathbf{u}_{w_t} \cdot \mathbf{v}_{w_{t+j}})}{\sum_{k \in V} \exp(\mathbf{u}_{w_t} \cdot \mathbf{v}_k)}$$

- $\theta = \{u_k\}, \{v_k\}$ are all the parameters in the model!

Expensive to compute!

Solution:

- Hierarchical Softmax
- Negative Sampling

Slide Credit: Richard Socher, Christopher Manning

Negative Sampling

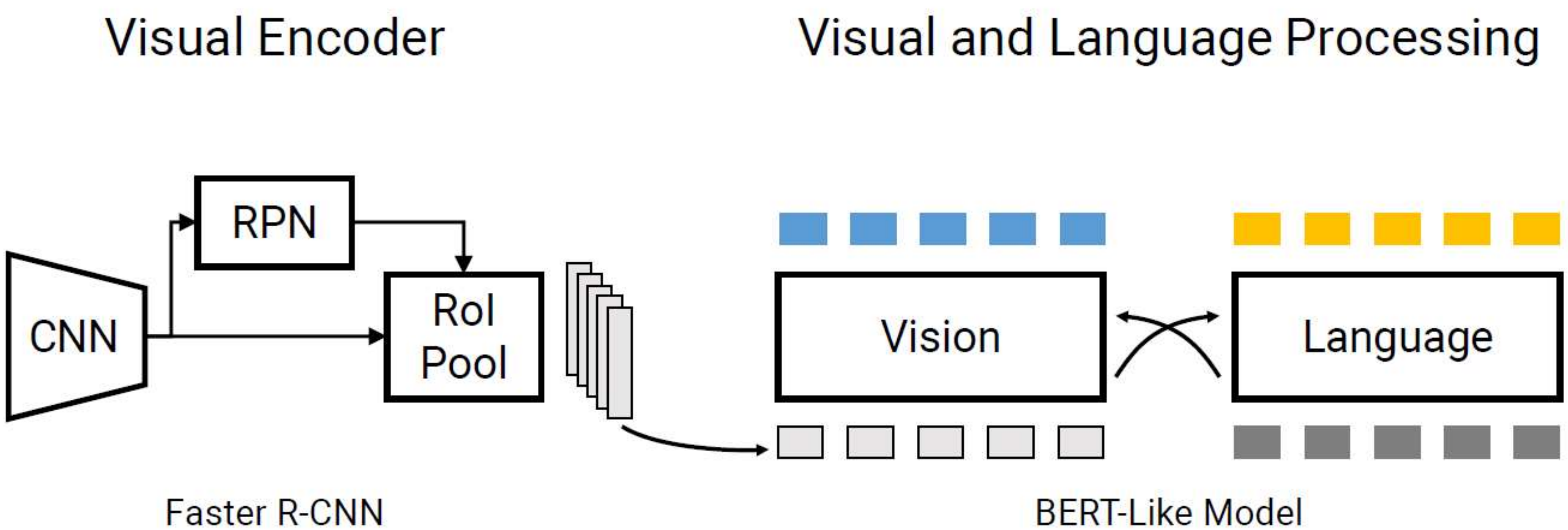
$$P(w_{t+j} \mid w_t) = \frac{\exp(\mathbf{u}_{w_t} \cdot \mathbf{v}_{w_{t+j}})}{\sum_{k \in V} \exp(\mathbf{u}_{w_t} \cdot \mathbf{v}_k)}$$

Expensive to compute!

Intuition:

- For each ***(w, c)*** pair, we sample ***k*** negative pairs ***(w, c')***: (***k*** = 5, 10, .., 20)
- Maximize probability that real outside word appears, minimize prob. that random words appear around center word.
- Distribution makes less frequent words be sampled more often.

Slide Credit: Danqi Chen, Christopher Manning



Multi-Model Embeddings

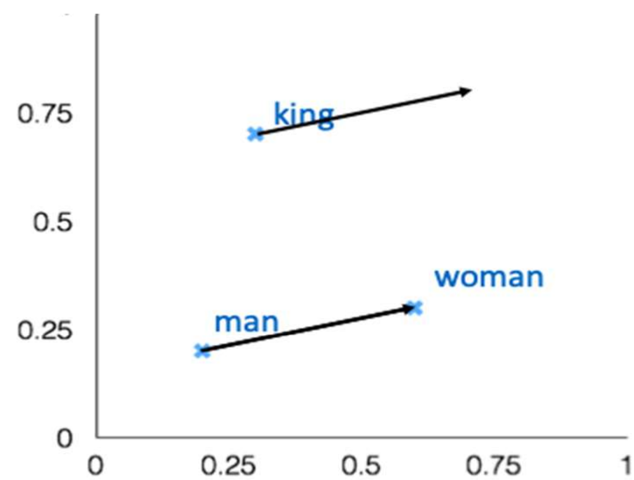
Intrinsic word embedding evaluation

a:b :: c:?

man:woman :: king:?

$$d = \arg\max_i \frac{(x_b - x_a + x_c)^T x_i}{||x_b - x_a + x_c||}$$

- Evaluate word vectors by how well their cosine distance after addition captures intuitive semantic and syntactic analogy questions
- **More examples:** http://download.tensorflow.org/data/questions-words.txt



Slide Credit: Richard Socher, Christopher Manning

Graph Embeddings

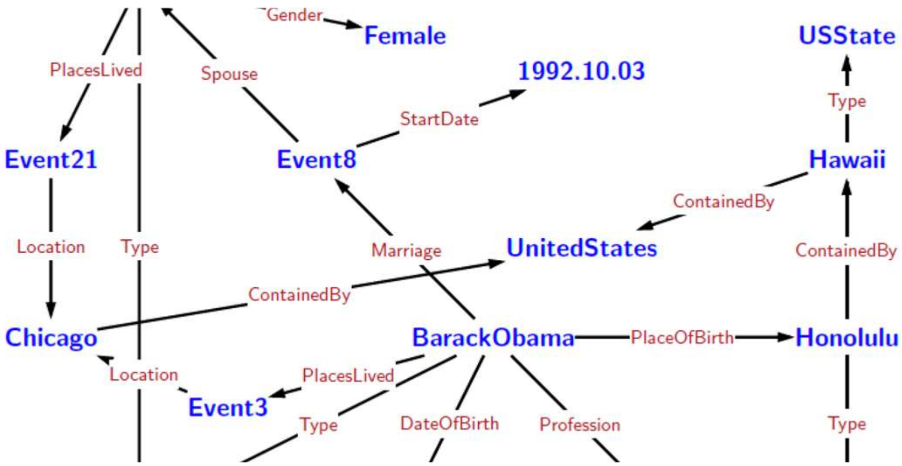
FACEBOOK AI

Georgia Tech

(Big) Graph Data is Everywhere

Knowledge Graphs

Standard domain for studying graph embeddings *(Freebase, …)*



Wang, Zhenghao & Yan, Shengquan & Wang, Huaming & Huang, Xuedong. (2014). An Overview of Microsoft Deep QA System on Stanford WebQuestions Benchmark.

Recommender Systems

Deals with graph-like data, but supervised

	user_id	movie_id	rating
0	196	242	3

Social Graphs

Predict attributes based on homophily or structural similarity *(Twitter, Yelp, …)*

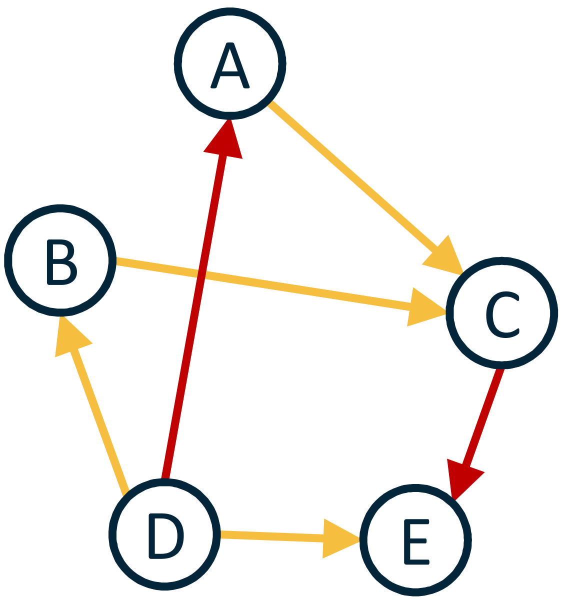
Slide Credit: Adam Lerer

Graph Embedding & Matrix Completion

	item1	item2	…	itemN
person1	-	+		+
person2	+	?		
…				
personP	+	-		?

- Relations between items (and people)
- Items in {people, movies, page, articles, products, word sequences…}
- Predict if someone will like an item, if a word will follow a word sequence

Slide Credit: Yann LeCun



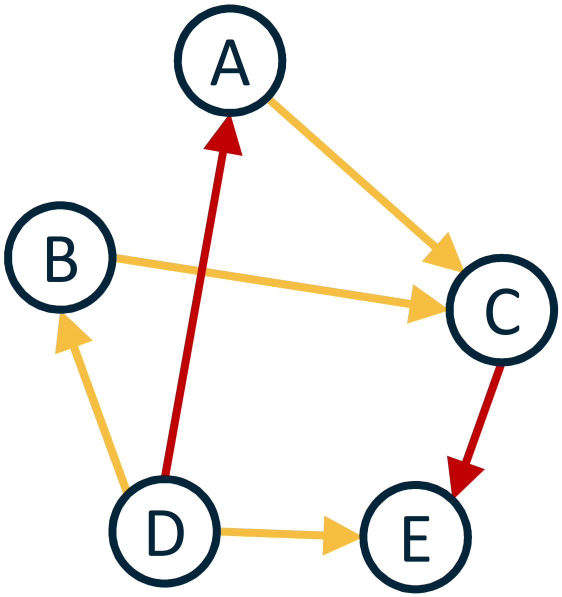
A multi-relation graph

Embedding: A learned map from entities to vectors of numbers that encodes similarity

- Word embeddings: word → vector
- Graph embeddings: node → vector

Graph Embedding: Optimize the objective that **connected nodes have more similar embeddings** than unconnected nodes via gradient descent.

Slide Credit: Adam Lerer



A multi-relation graph

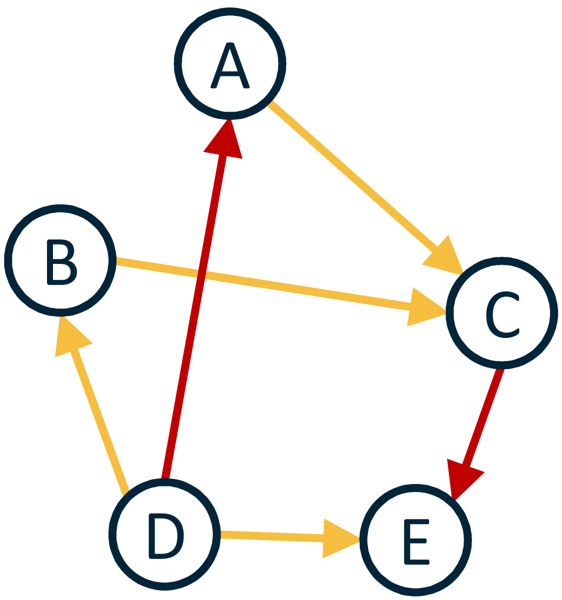
Why Graph Embeddings?

Graph embeddings are a form of **unsupervised learning** on graphs.

- **Task-agnostic** entity representations
- Features are useful on downstream tasks without much data
- Nearest neighbors are semantically meaningful

Slide Credit: Adam Lerer

PyTorch BigGraph



A multi-relation graph

Margin loss between the score for an edge $f(e)$ and a negative sampled edge $f(e')$

$$\mathcal{L} = \sum_{e \in G} \sum_{e' \in S'_e} \max(f(e) - f(e') + \lambda, 0))$$

The score for an edge is a similarity (e.g. dot product) between the source embedding and a transformed version of the destination embedding, e.g.

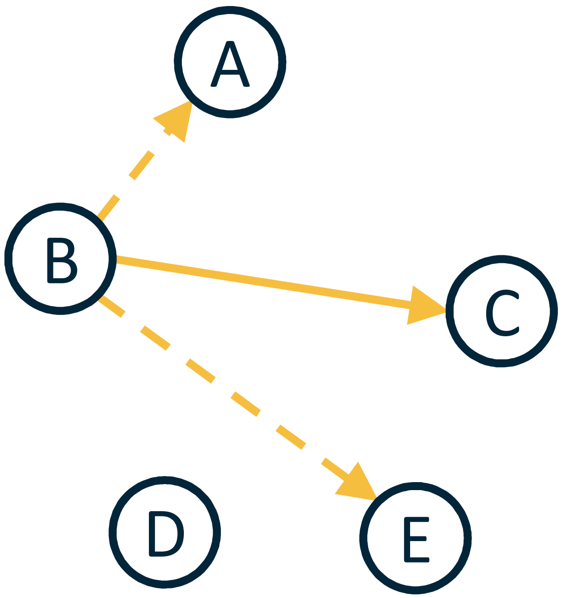
$$f(e) = \cos(\theta_s, \theta_r + \theta_d)$$

Negative samples are constructed by taking a real edge and replacing the source or destination with a random node.

$$S'_e = \{(s', r, d) | s' \in V\} \cup \{(s, r, d' | d' \in V\}$$

Slide Credit: Adam Lerer

PyTorch BigGraph



A multi-relation graph

Margin loss between the score for an edge $f(e)$ and a negative sampled edge $f(e')$

$$\mathcal{L} = \sum_{e \in G} \sum_{e' \in S'_e} \max(f(e) - f(e') + \lambda, 0))$$

The score for an edge is a similarity (e.g. dot product) between the source embedding and a transformed version of the destination embedding, e.g.

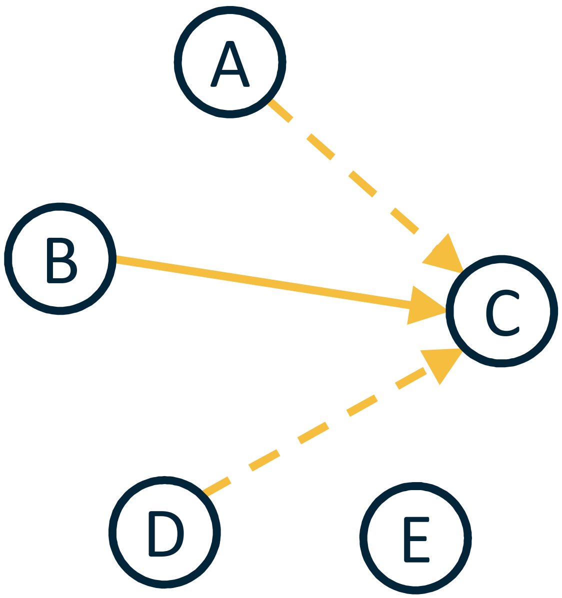
$$f(e) = \cos(\theta_s, \theta_r + \theta_d)$$

Negative samples are constructed by taking a real edge and replacing the source or destination with a random node.

$$S'_e = \{(s', r, d) | s' \in V\} \cup \{(s, r, d' | d' \in V\}$$

Slide Credit: Adam Lerer

PyTorch BigGraph



A multi-relation graph

Margin loss between the score for an edge $f(e)$ and a negative sampled edge $f(e')$

$$\mathcal{L} = \sum_{e \in G} \sum_{e' \in S'_e} \max(\boxed{f(e)} - f(e') + \lambda, 0))$$

The score for an edge is a similarity (e.g. dot product) between the source embedding and a transformed version of the destination embedding, e.g.

$$f(e) = \cos(\theta_s, \theta_r + \theta_d)$$

Negative samples are constructed by taking a real edge and replacing the source or destination with a random node.

$$S'_e = \{(s', r, d) | s' \in V\} \cup \{(s, r, d' | d' \in V\}$$

Slide Credit: Adam Lerer

Multiple Relations in Graphs

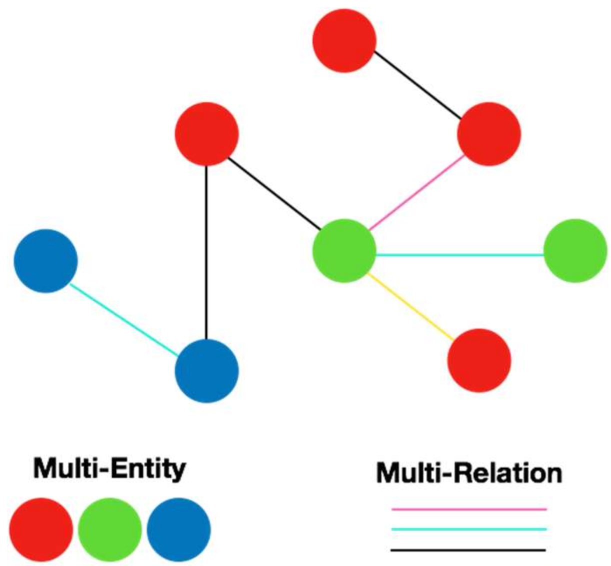
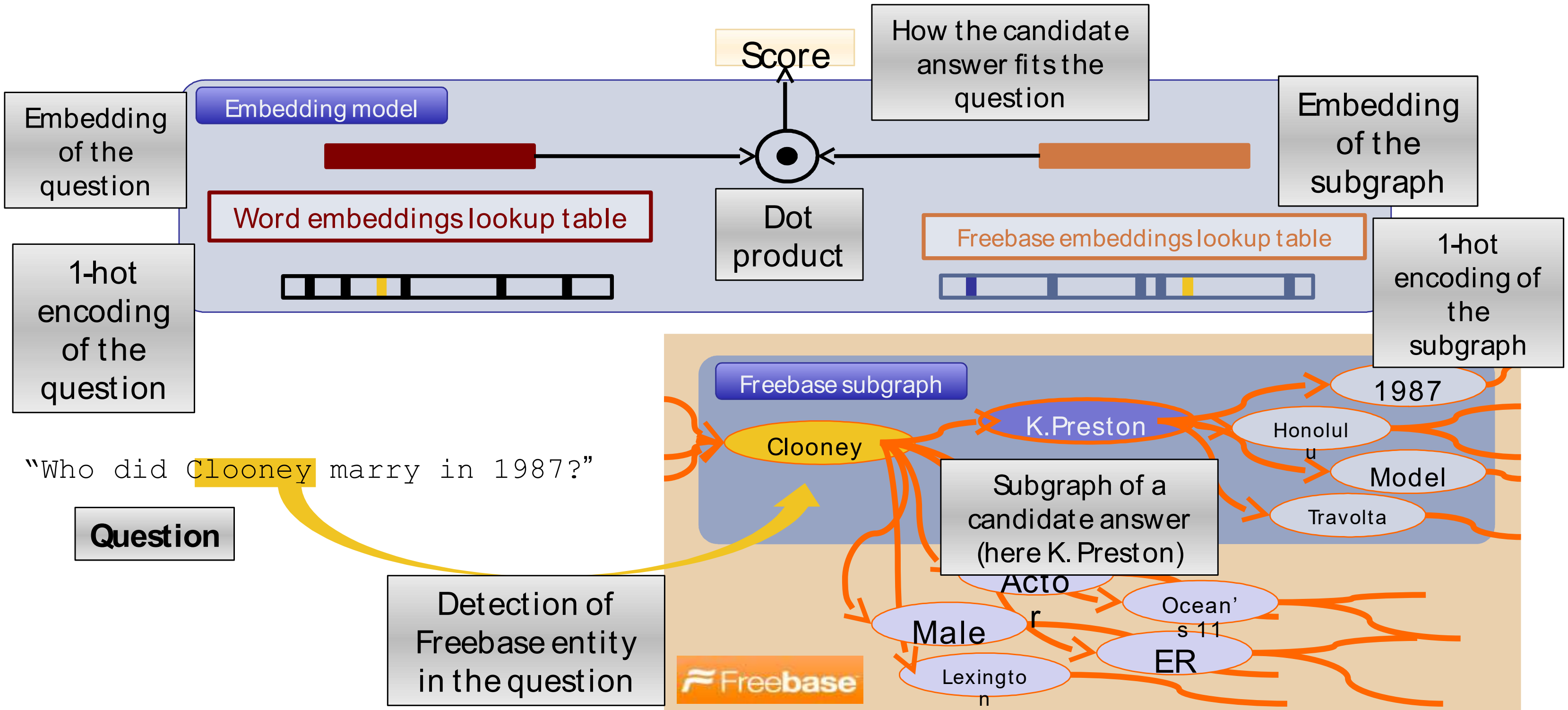


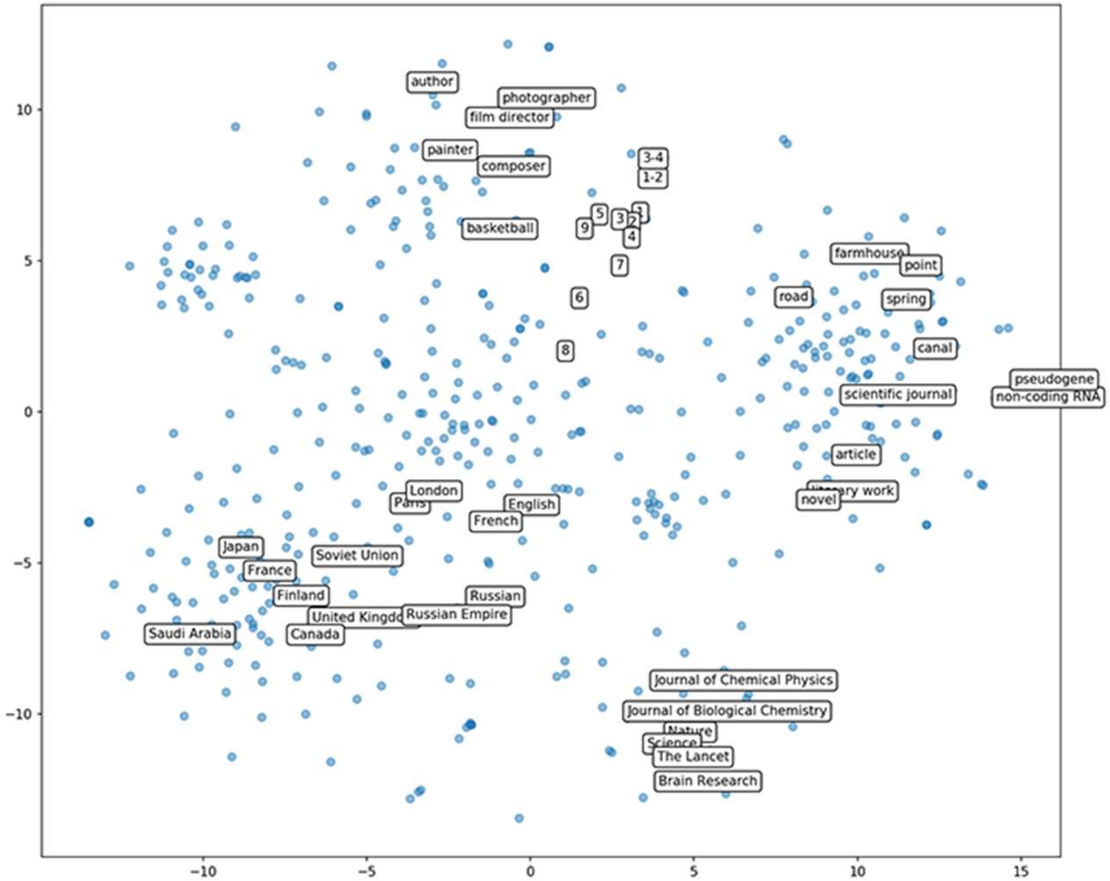
Figure Credit: Alex Peysakhovich

- **Identity:** $g(x) = x$
- **Translator:** $g(x|\Delta) = x + \Delta$
  [Bordes et al. 13']
- **Affine:** $g(x|A, \Delta) = Ax + \Delta$
  [Nickel et al., 11']
- **Diagonal:** $g(x|b) = b \odot x$
  [Yang et al., 15']

Embedding a Knowledge Base [Bordes et al. 2013]



Slide Credit: Yann LeCun



Embedding Wikidata Graph
[Lerer et al. 19']

PyTorch BigGraph

https://github.com/facebookresearch/PyTorch-BigGraph

Applications, world2vec

Application: TagSpace, PageSpace

TagSpace

Input: restaurant has great food

Label: #yum, #restaurant

Use-cases:

- Labeling posts
- Clustering of hashtags

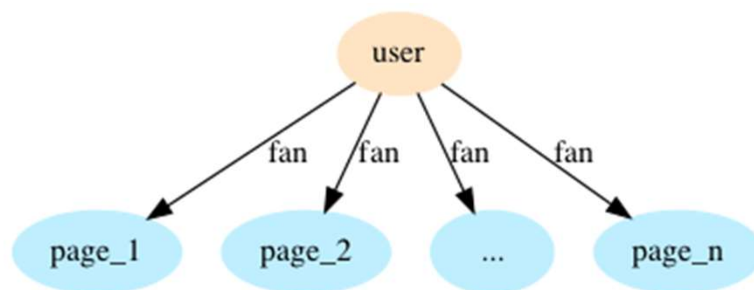
Reference: [Weston et al. 14'], [Wu et al. 18']
https://github.com/facebookresearch/StarSpace

PageSpace

Input: (user, page) pairs

Use-cases:

- Clustering of pages
- Recommending pages to users



PageSpace

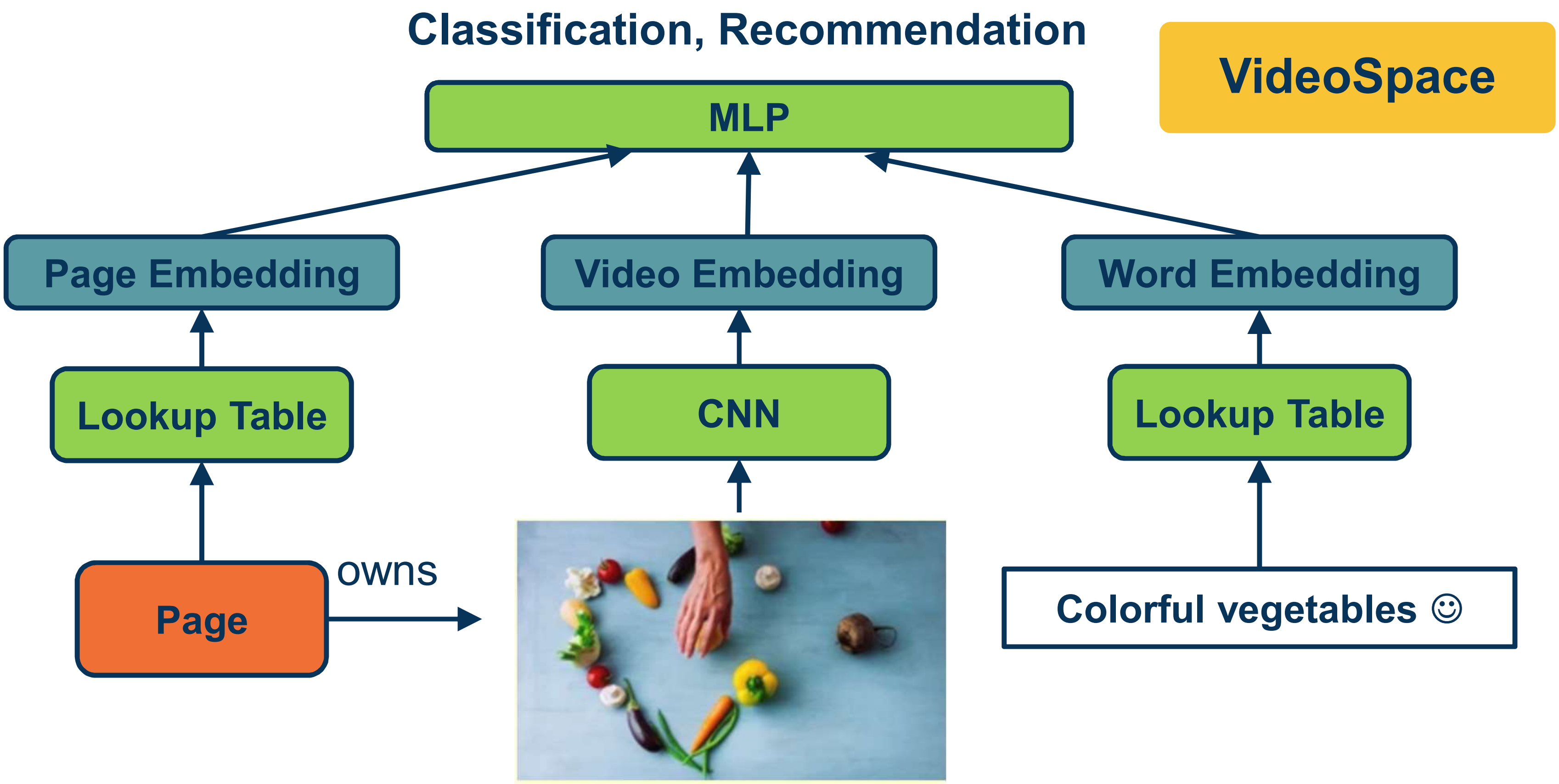
Search nearest neighbor for page *The New York Times:*

Washington Post, score: 0.80
Bloomberg Politics, score: 0.77
VICE News, score: 0.71
Bloomberg: 0.69
Financial Times: 0.68

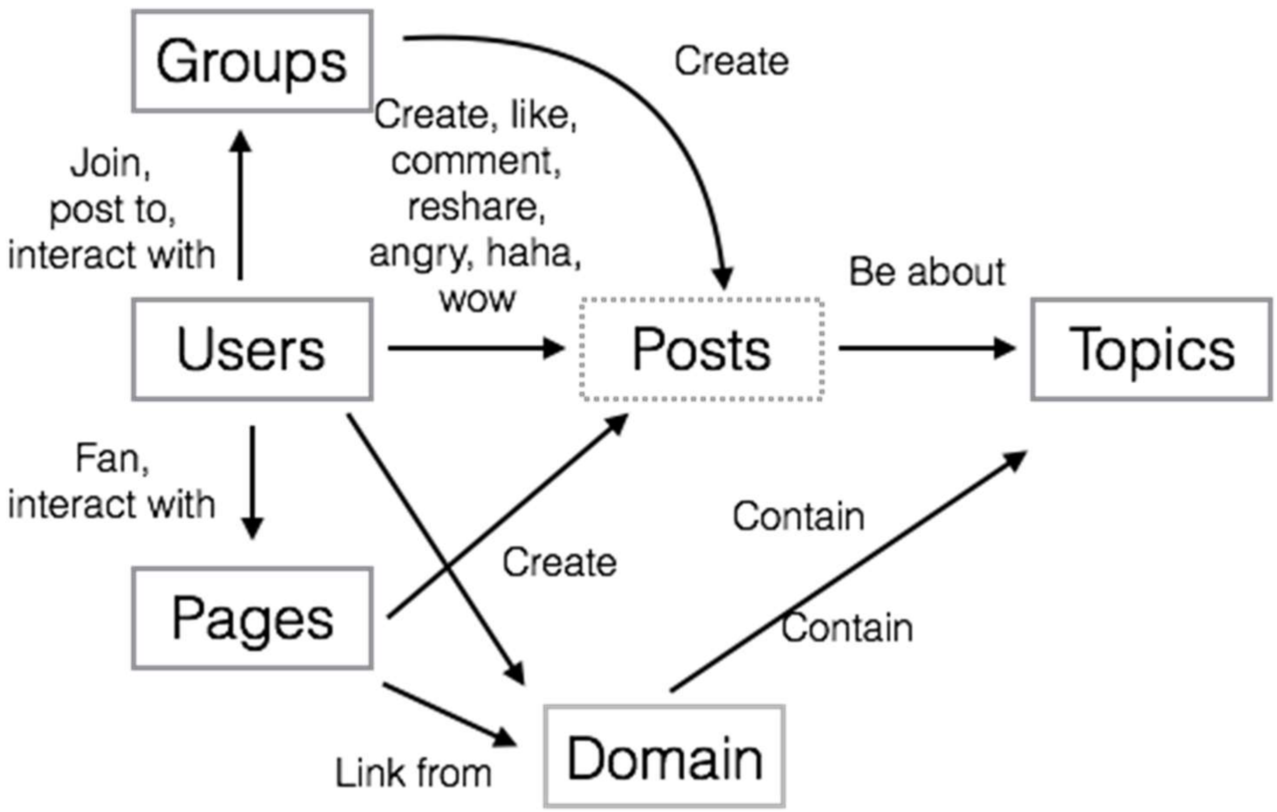
Other information:

- Title and description (words)
- Images
- Videos

Faiss: a library for efficient similarity search and clustering of dense vectors.
https://github.com/facebookresearch/faiss



Application: VideoSpace

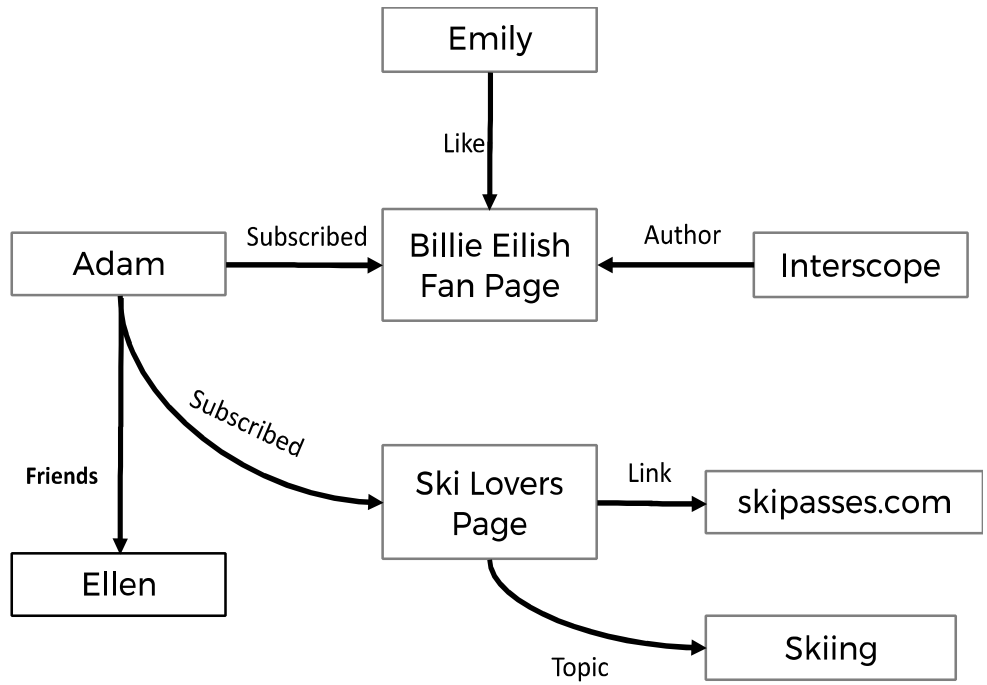


Slide Credit: Alex Peysakhovich

Application: world2vec

The Power of Universal Behavioral Features

- What pages or topics might you be interested in?
- Which posts contain misinformation, hate speech, election interference, …?
- Is a person's account fake / hijacked?
- What songs might you like? (even if you've never provided any song info)



Slide Credit: Adam Lerer

Users

- Bad Actor Cluster

Groups

- 'For Sale' Group prediction

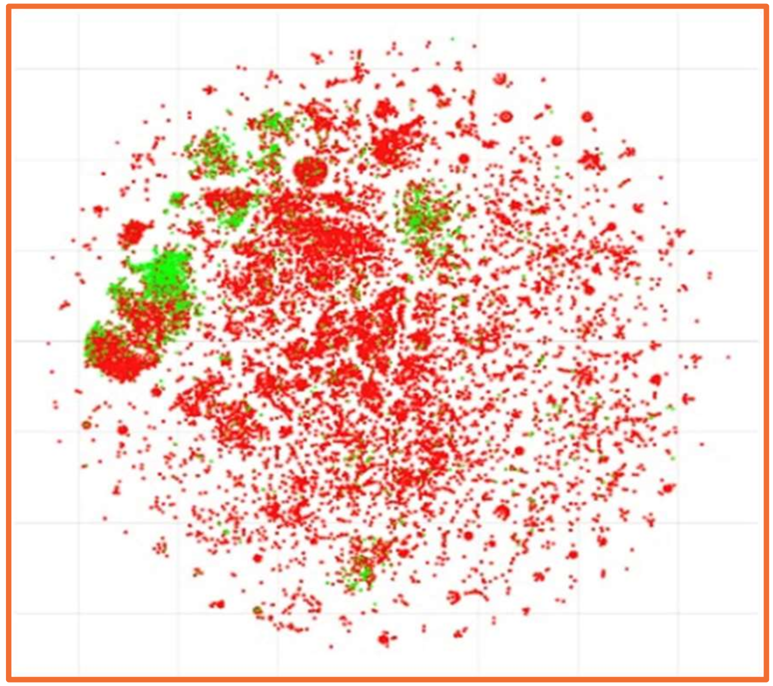
Pages

- Recommendation
- Page category prediction
- Identify spam / hateful pages

Domains

- Domain type prediction
- Identify mis-Information

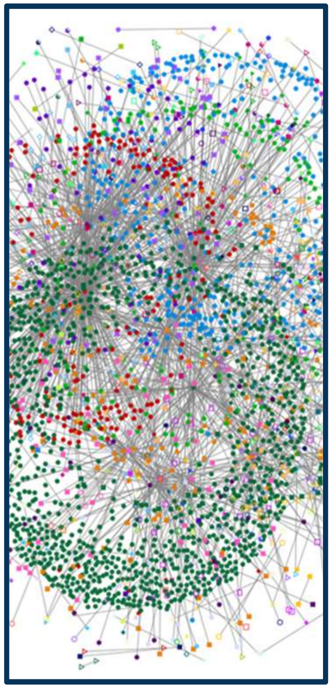
T-SNE plot of page embeddings. Pages labeled as misinformation marked in green.



Slide Credit: Alex Peysakhovich

Application: world2vec

Learning Node Features



BEHAVIOR GRAPH

Unsupervised Graph Embedding Method

OTHER STUFF: CONTENT FEATURES, COUNTERS, ETC

NODE EMBEDDINGS

RANKING

CLASSIFICATION

VISUALIZATION

CLUSTERING

NEAREST NEIGHBOR

Slide Credit: Adam Lerer

PREPROCESSING

- COLLECT DATA
- CONSTRUCT EDGELIST
- PRUNE / WEIGHT EDGES

PYTORCH-BIGGGRAPH WORKFLOW

- PREPROCESSING
- PYTORCH-BIGGRAPH
- POSTPROCESSING

DOWNSTREAM TASKS

- NEAREST NEIGHBORS

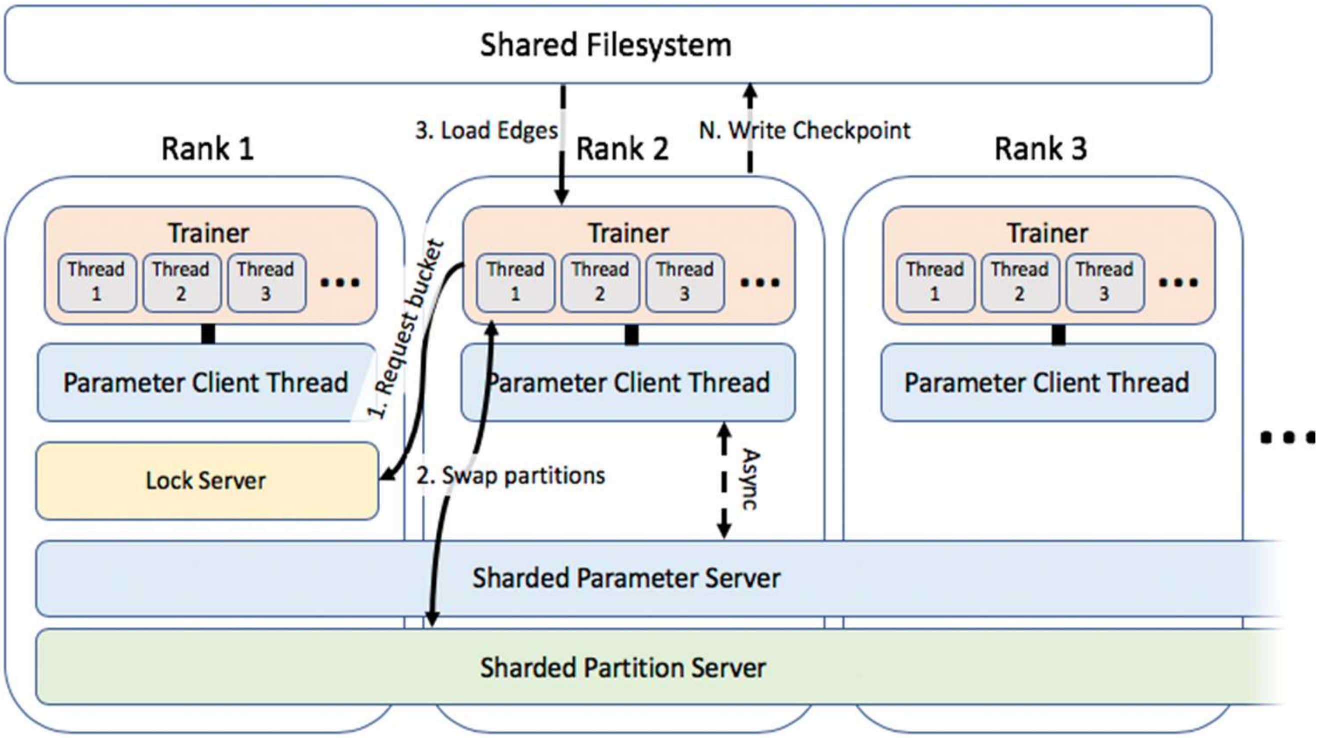
  RETRIEVAL / RECOMMENDATION
- VISUALIZATION

  ANALYSIS CLUSTERING
- FEATURES → DB FOR MODEL FEATURE

  CLASSIFICATION RANKING

Slide Credit: Adam Lerer

Application: world2vec



Slide Credit: Adam Lerer

Application: world2vec