Neuro-Symbolic Representations for IR
2.1 – Neural Text and Graph Representations

Laura Dietz

ECIR

2023

Opportunities for Query-Specific Entity Representations

Context-aware representations of entities in text based on a query

- Query Understanding
  - Captures both inherent properties of entities
  - Models entity relevance to the query
- Retrieval and Ranking
  - Overcomes limitations of simple keyword matching
  - Uncovers deeper connections between entities, documents, and the query
- Answer Generation
  - Generates more coherent, contextually appropriate responses
  - Tailors output to the specific information need

Opportunities for Query-Specific Entity Representations

Context-aware representations of entities in text based on a query

- Query Understanding
  - Captures both inherent properties of entities
  - Models entity relevance to the query
- Retrieval and Ranking
  - Overcomes limitations of simple keyword matching
  - Uncovers deeper connections between entities, documents, and the query
- Answer Generation
  - Generates more coherent, contextually appropriate responses
  - Tailors output to the specific information need

Lots of magical hand waving. Let's look at that!

Deep Learning Techniques

update

Recap Part 1: Symbolic AI representations and tasks

- Latent embedding space
- BERT for Classification and Tagging
- Dense Retrieval Models

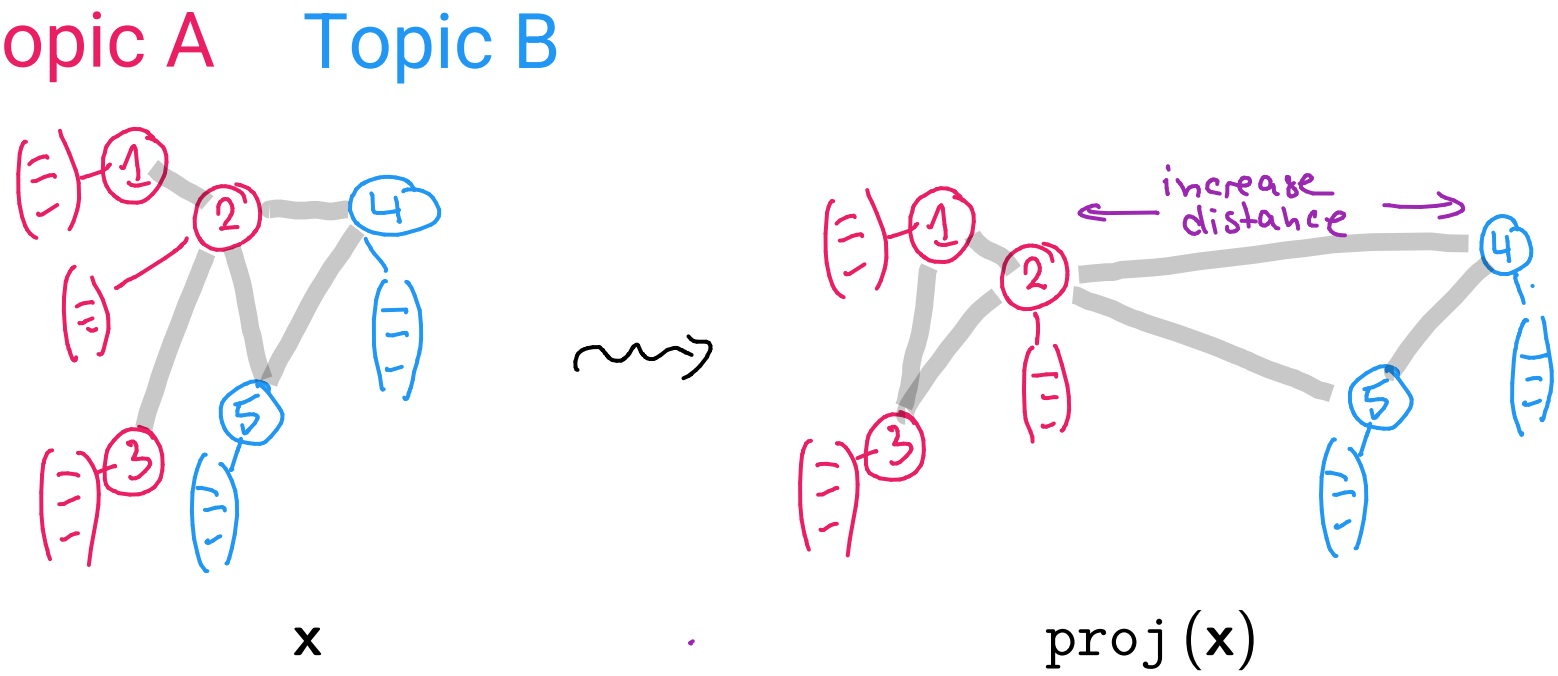
Part 2: Text-to-symbols and Ranking Neural Text Representations ←You Are Here

- Metric Learning in Latent Space
- Attention + Transformers
- Dense retrieval: Bi-Encoders and Cross-Encoders
- Graph Neural Networks
- Retrieval-augmented generation models

Part 3: Neural approaches for Neuro-Symbolic Reasoning

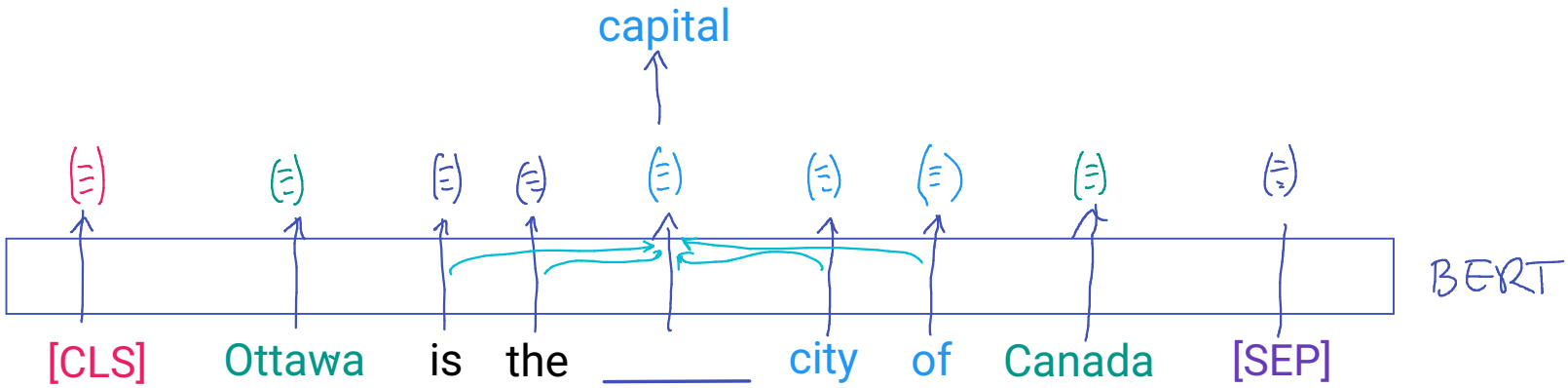
Metric Learning

- Project each point $\mathbf{x}$ with a function `proj`.
  - $\mathbf{y} = \texttt{proj}(\mathbf{x}) = \mathbf{W}\mathbf{x} + \mathbf{b}$
  - where $\mathbf{W}$ and $\mathbf{b}$ are trainable parameters.



Train: latent space has "useful" representation of proximity.

BERT Training with CLOZE "fill the blanks"



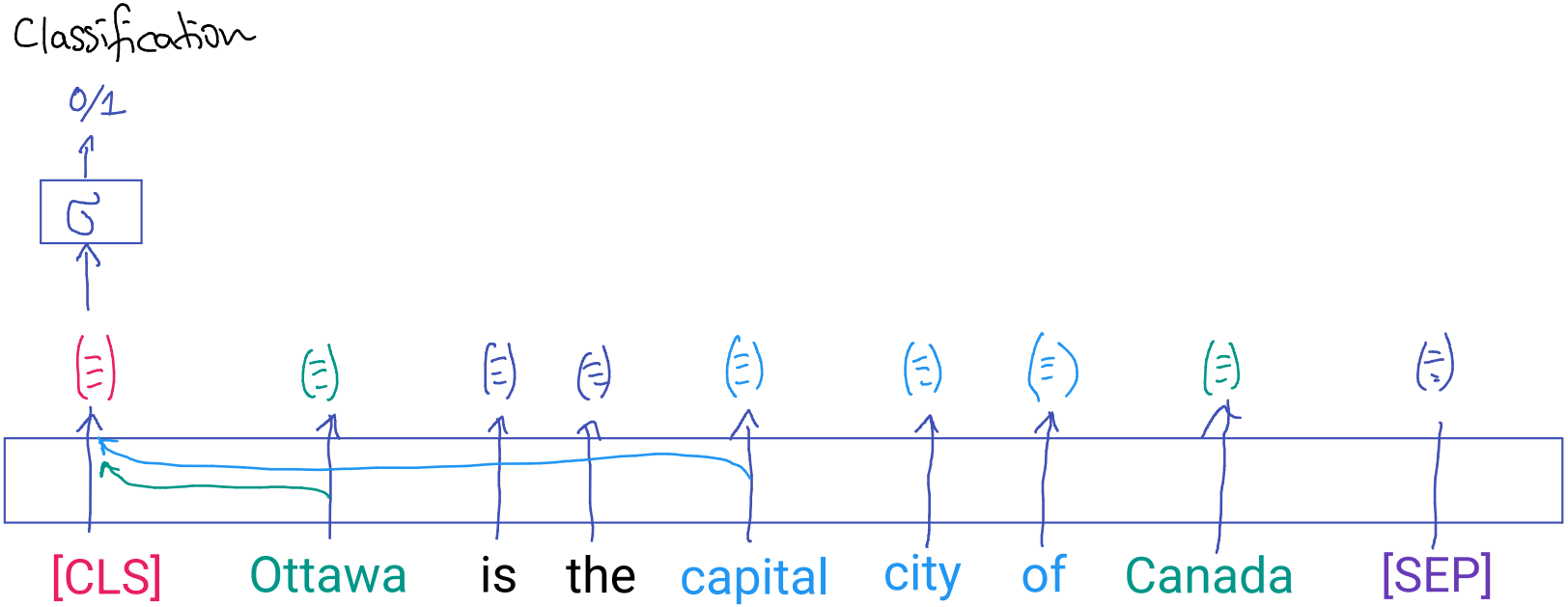
Each word vector is aware of neighboring words!

→Contextualized representation

How does this "awareness" work?

LLMs for Classification

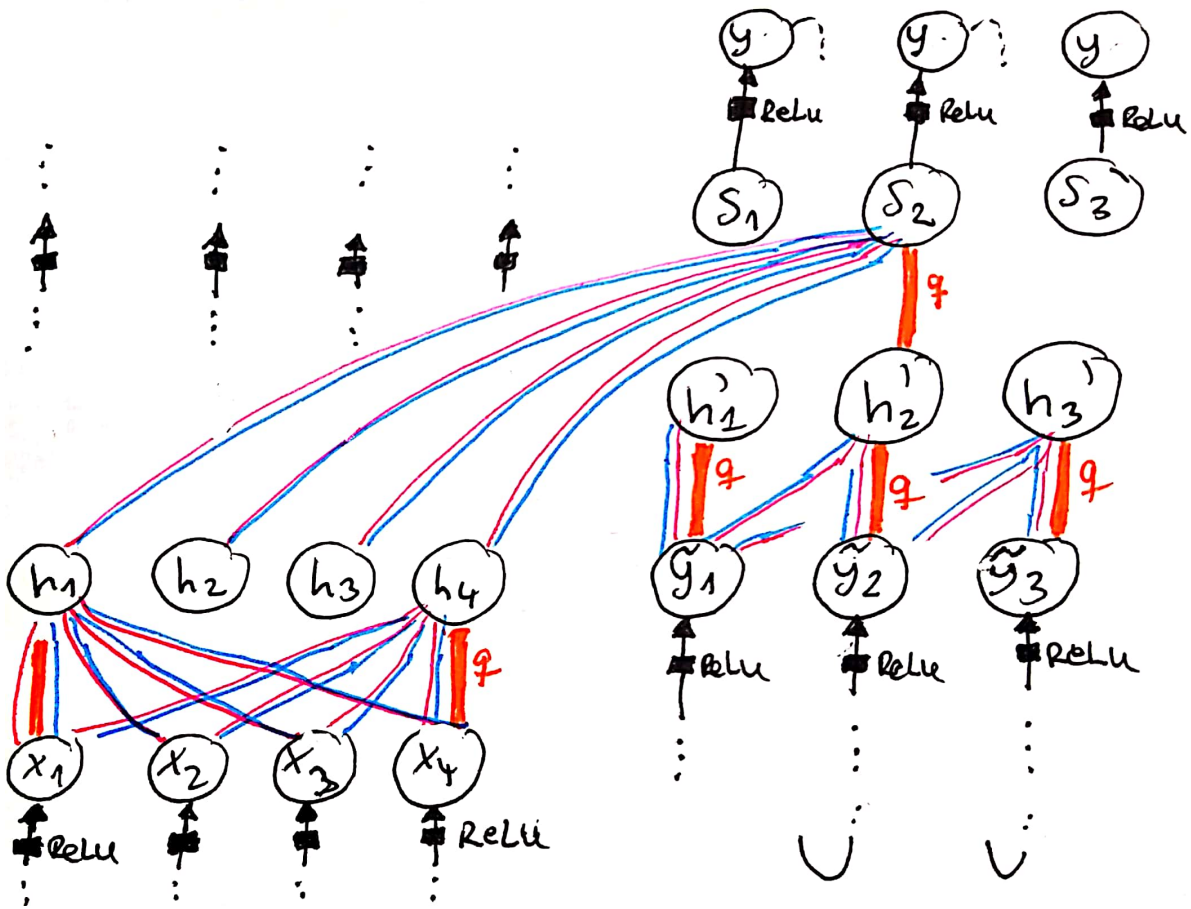
- Connect ground truth to vector of the [CLS] token
- Connect to a logistic/softmax layer to predict yes/no



The [CLS] token is also informed by neighboring words.

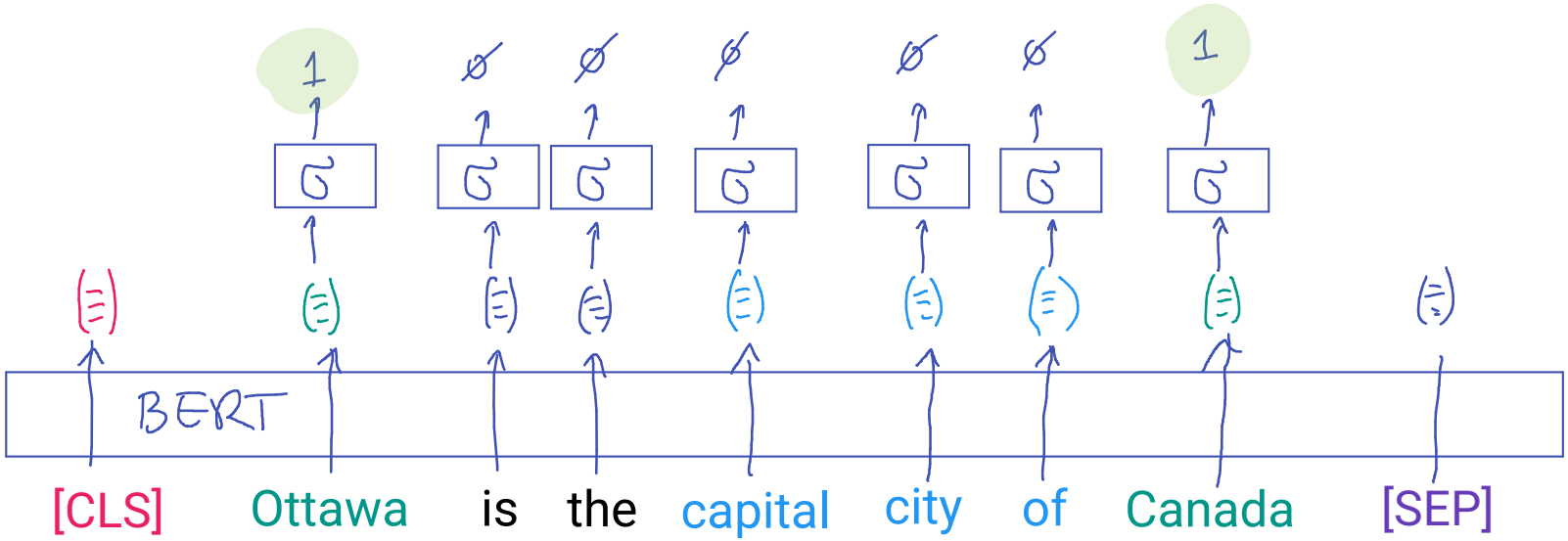
Transformers

- More than just [CLS] vectors
- Different answer document can depend on different query words



Tagging

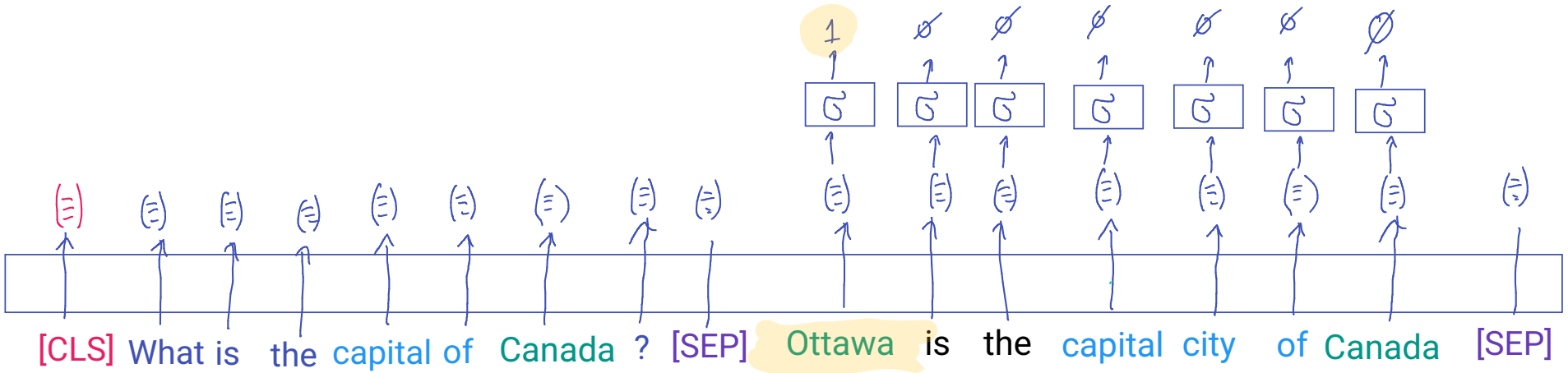
Example: Tag all LOCATION entities in the sentence (green)



- Connect ground truth to vector of words
- Connect to a logistic layer to predict yes/no

Question Answering: Answer Extraction

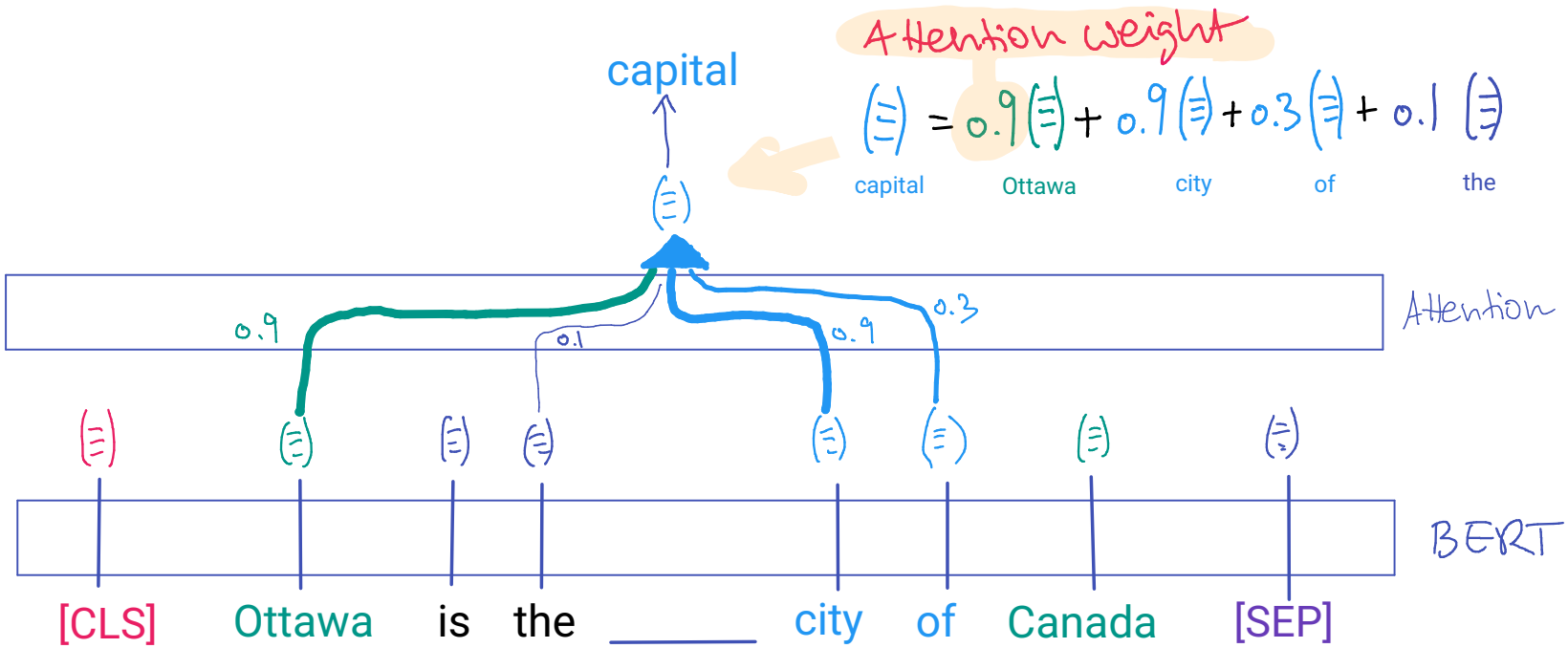
Example: What is the and Rankingcapital of Canada?
Find the answer in the given passage.



- Connect ground truth to vector of words
- Connect to a logistic layer to predict yes/no

Attention Mechanisms = Adding vectors

- Selectively focuses on relevant parts for each prediction
- Improves the performance especially for long input sequences

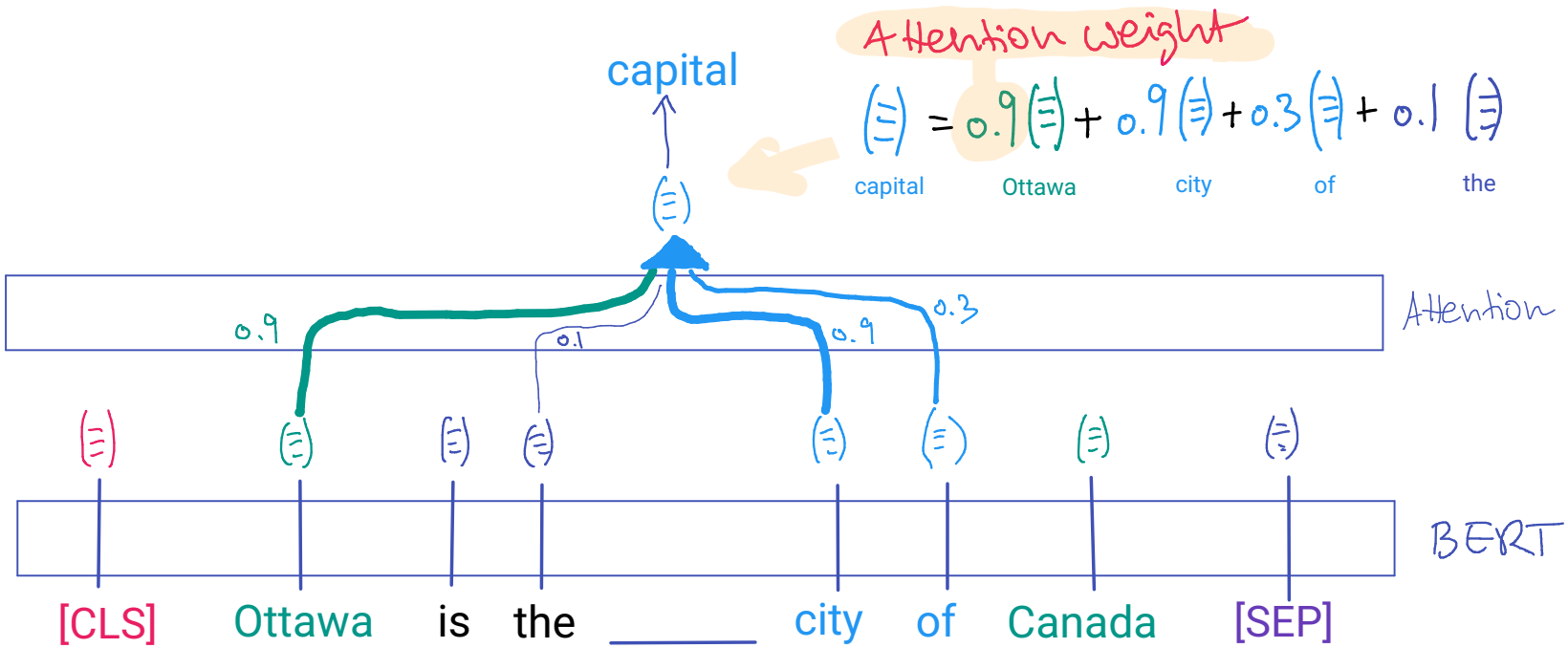


What does "selectively focus" mean?

How Attention is Trained

Thought experiment:

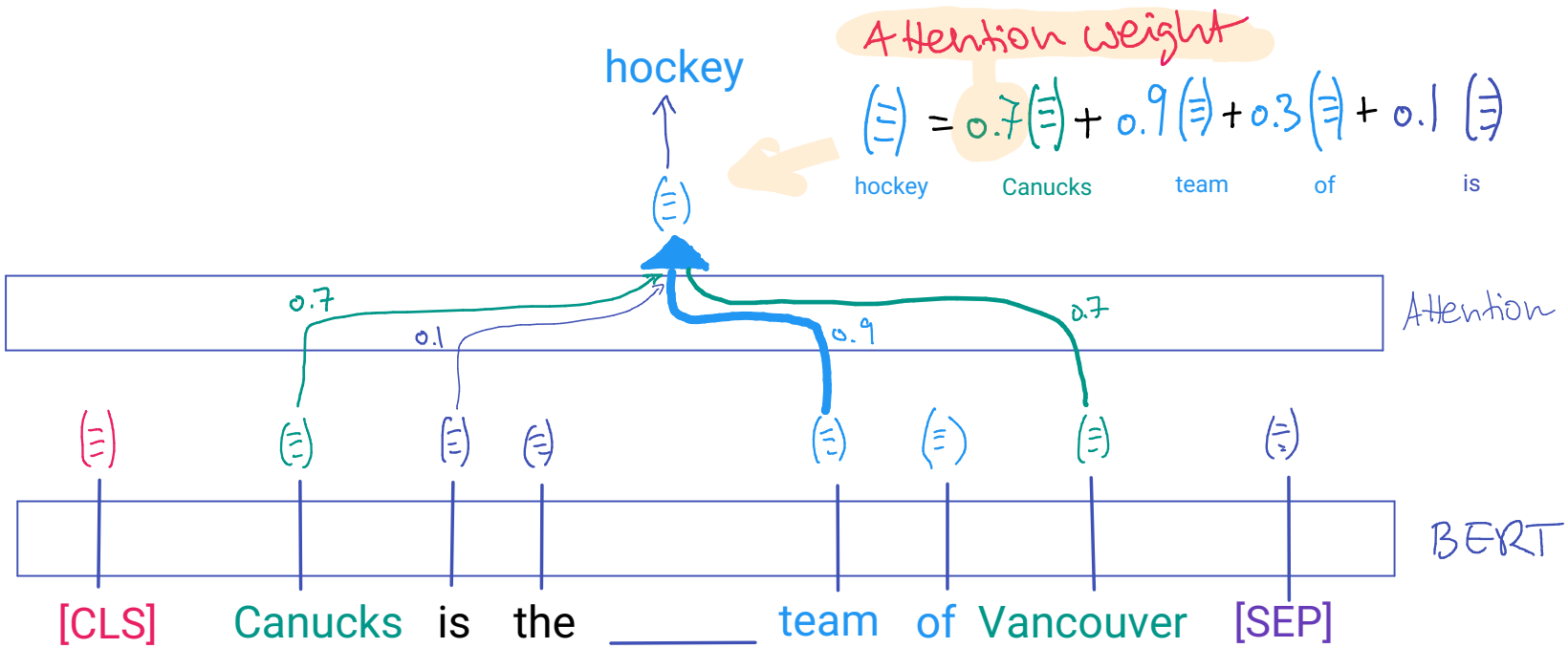
- Which tokens should we sum, so we get the best representation?
- Train to assign those tokens a high attention weight!



DuoBERT implementation Nogueira et al. "Multi-stage document ranking with BERT"

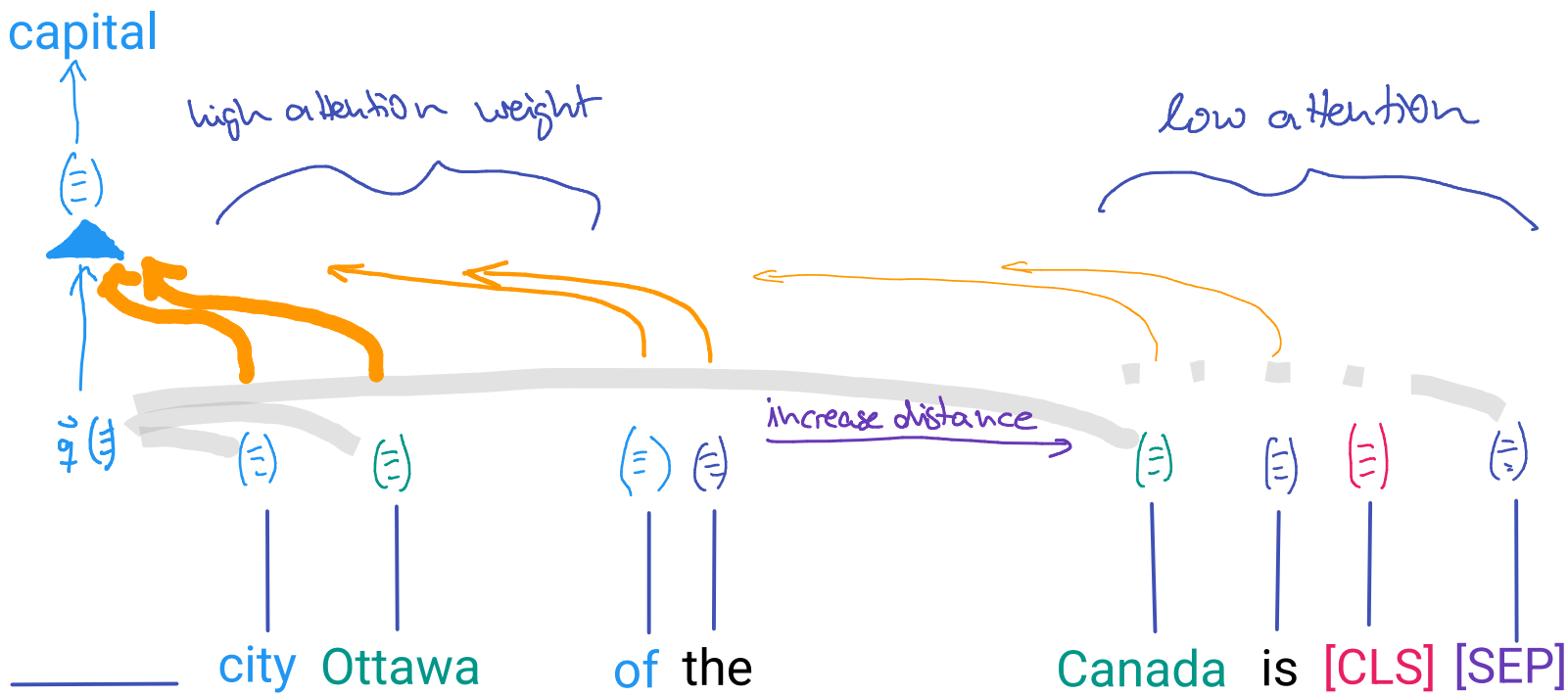
Generalizing Attention to Unseen Text

- Need a model to assign these "ideal" attention weights
- .... needs to generalize to unseen text



Attention as Metric Learning

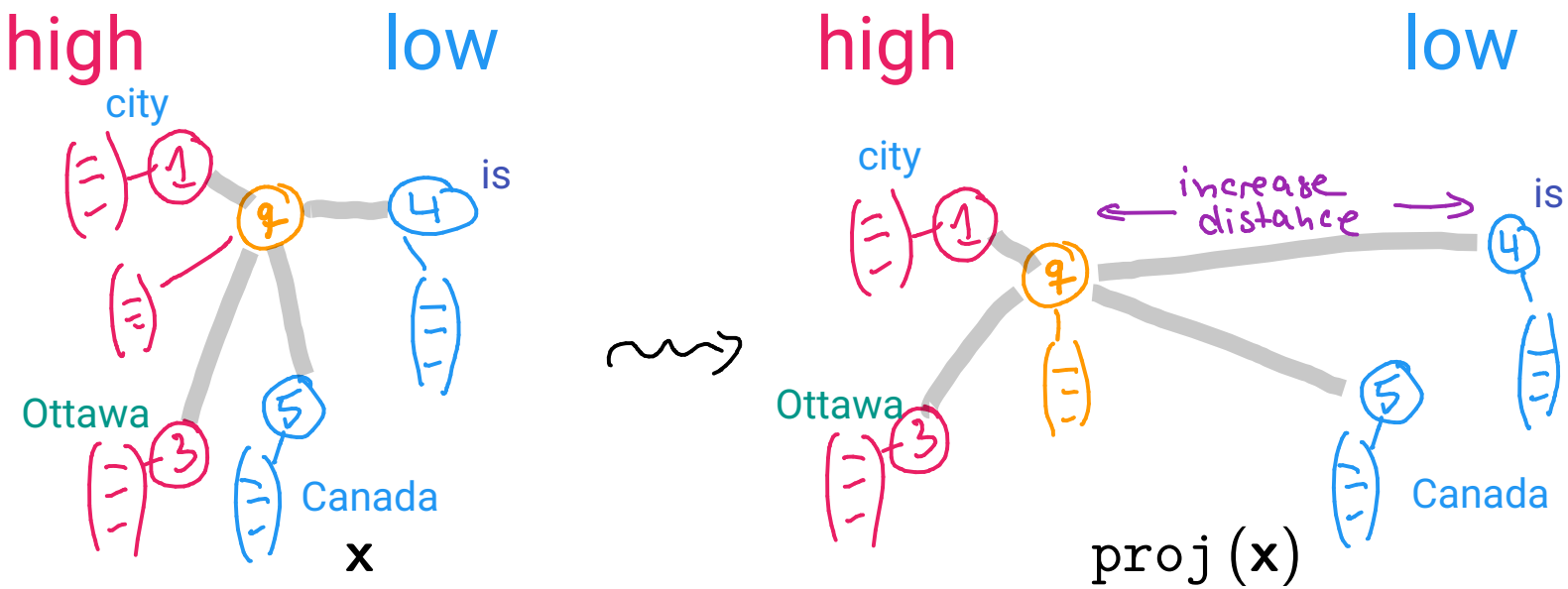
- Train latent space, where high attention words are close
- Use projection and a vector similarity (e.g. dot product)



The blank word plays the role of "the query"
How to "rank" other words?

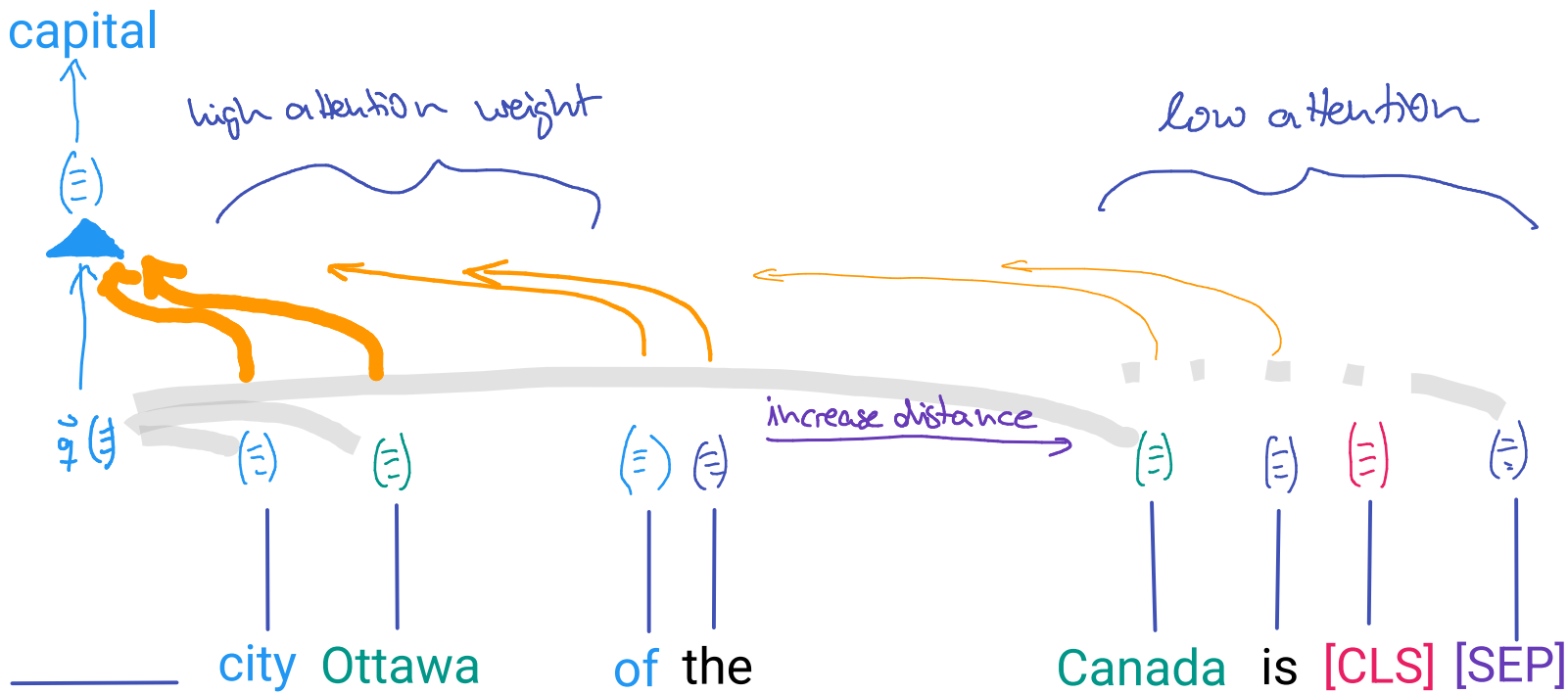
Model for Projections and Similarity

- Project each point **x** with a function `proj`.
  - $\mathbf{y} = \texttt{proj}(\mathbf{x}) = \mathbf{W}\mathbf{x} + \mathbf{b}$
  - where **W** and **b** are trainable parameters.



Key-Query-Value Attention (Scaled Dot-Product)

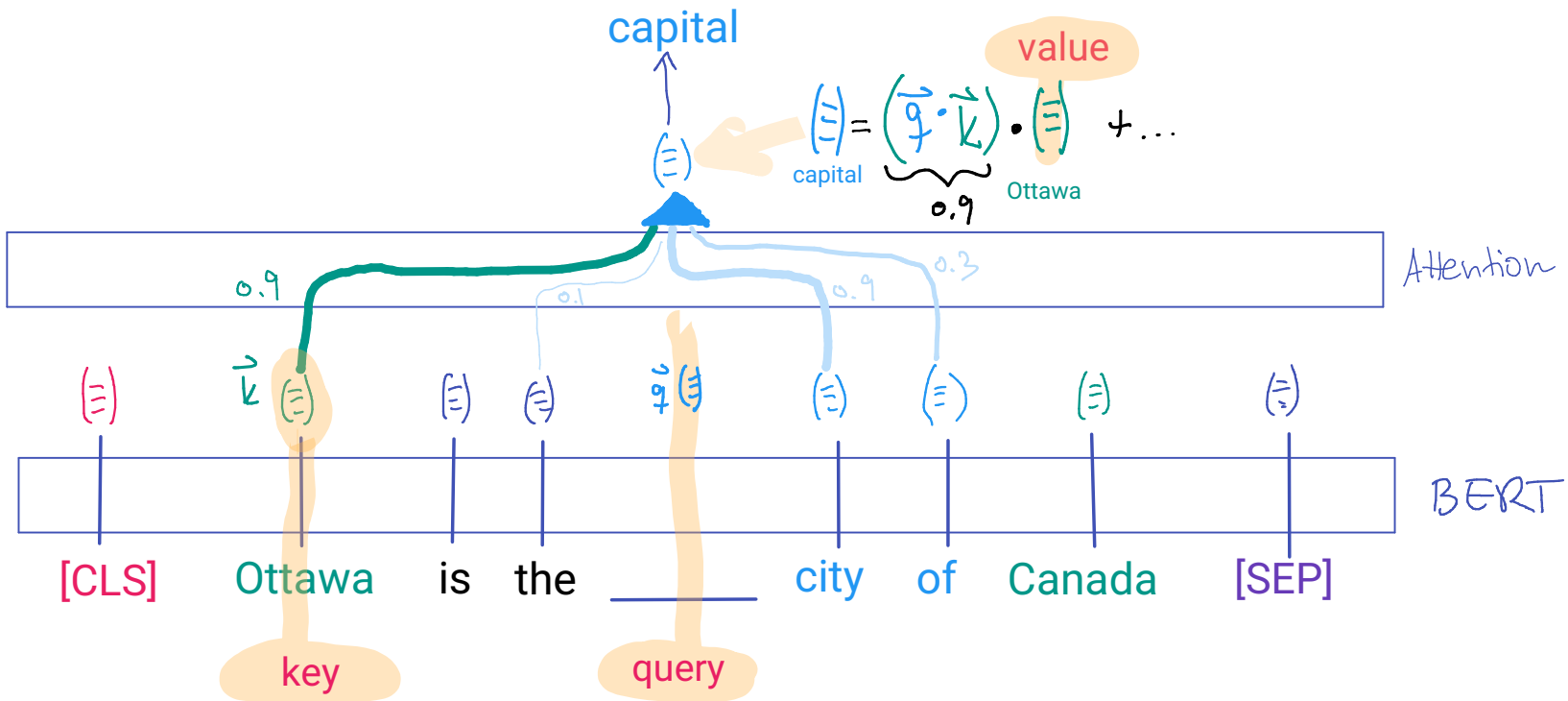
- Project query word: $\texttt{proj}(\mathbf{q})$
- Project other word: $\texttt{proj}(\mathbf{k})$
- Set trainable projection parameters to that:
- the ideal attention weights are $a_{qk} = \texttt{proj}(\mathbf{q}) \cdot \texttt{proj}(\mathbf{k})$



$\texttt{proj}(\text{capital}) \cdot \texttt{proj}(\text{Ottawa}) \approx 1$ $\texttt{proj}(\text{capital}) \cdot \texttt{proj}(\text{is}) \approx 0$

Training Projections for Query and Key

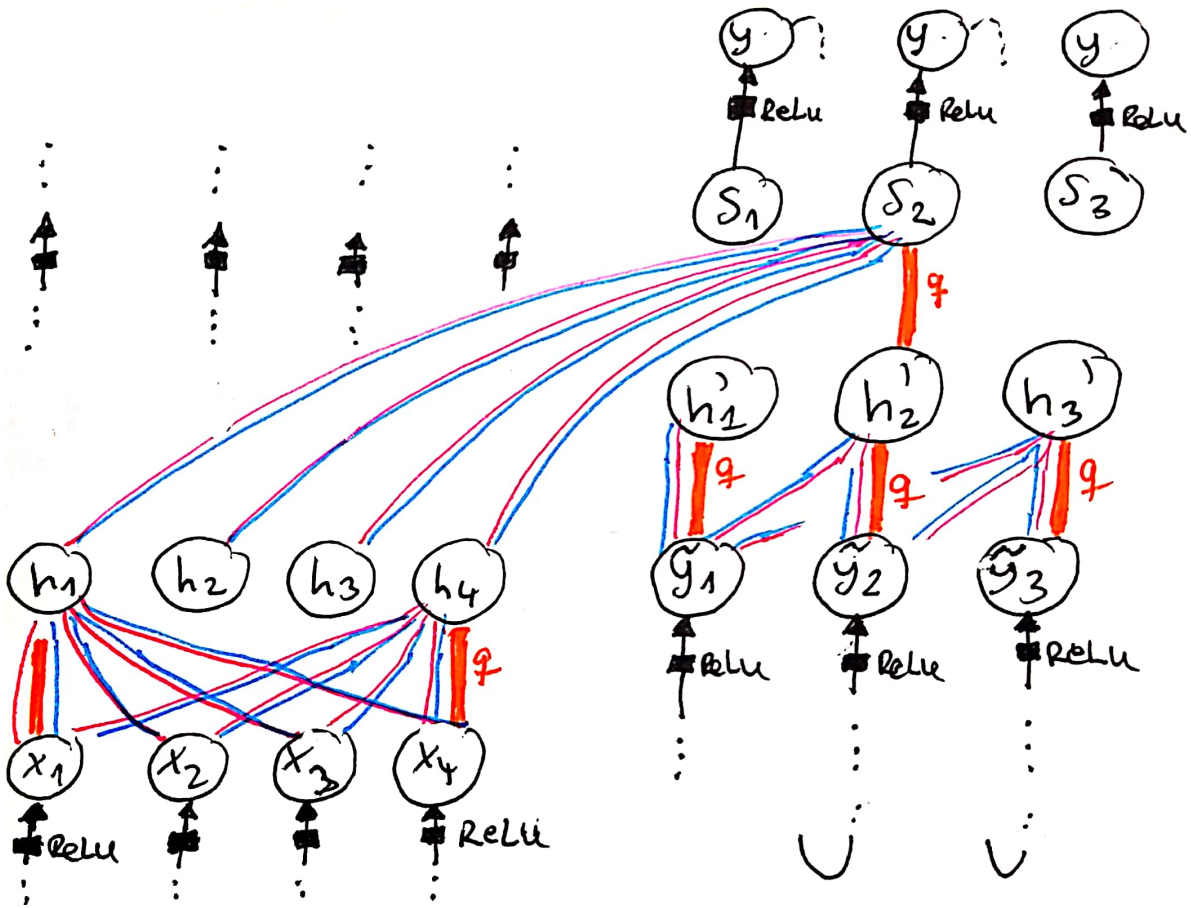
- Different projections for query and key so that
- the right words k will have high attention weight



$\texttt{proj}\,(\text{capital}) \cdot \texttt{proj}\,(\text{Ottawa}) \approx 1$ $\texttt{proj}\,(\text{capital}) \cdot \texttt{proj}\,(\text{is}) \approx 0$

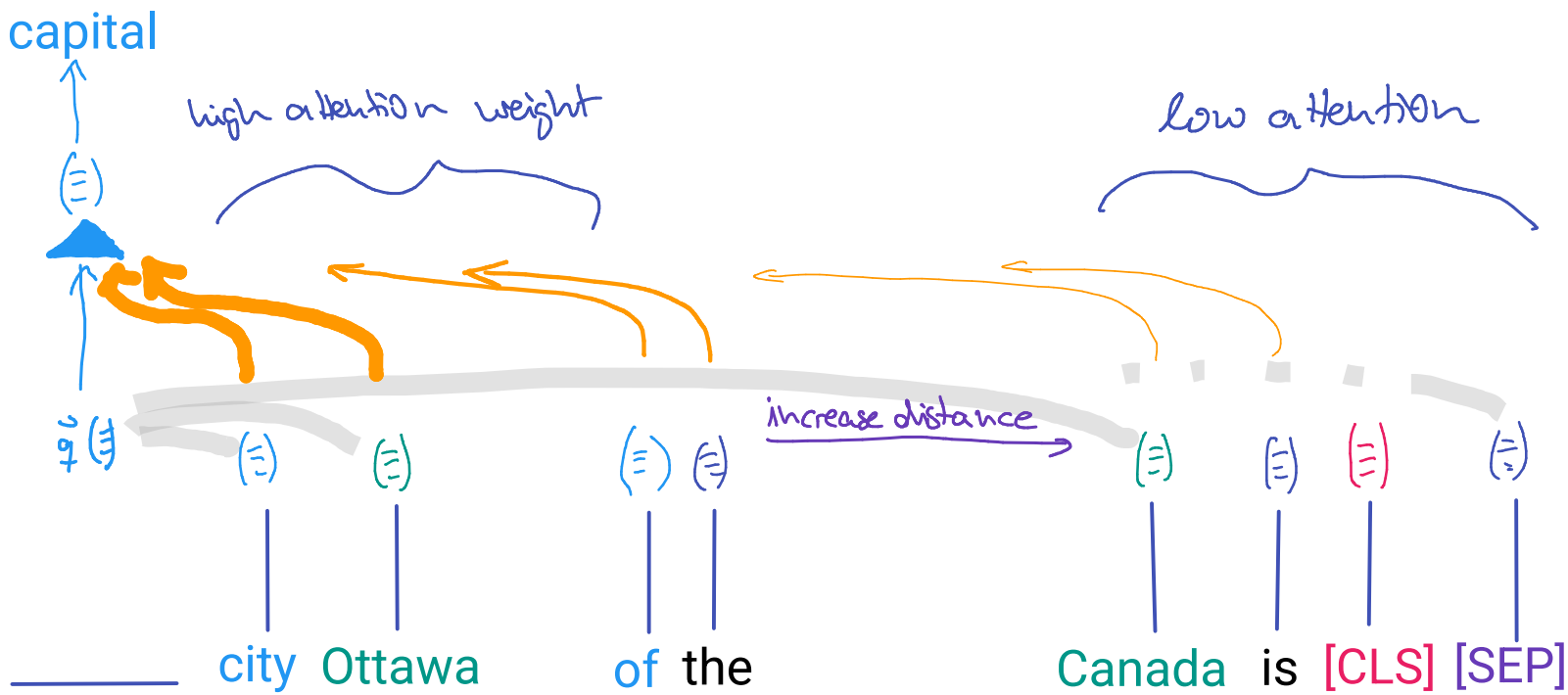
Transformers

- Several layers with attention mechanism = Capture context
- Fine-tuned for specific tasks, such as generation or classification
- Encoder → State → Decoder



Attention and "long-range dependencies"

- "Attention can model long range dependencies" – How?
- Because: Attention does not consider the sequence
- Add **position encoding** to teach about "neighboring words"



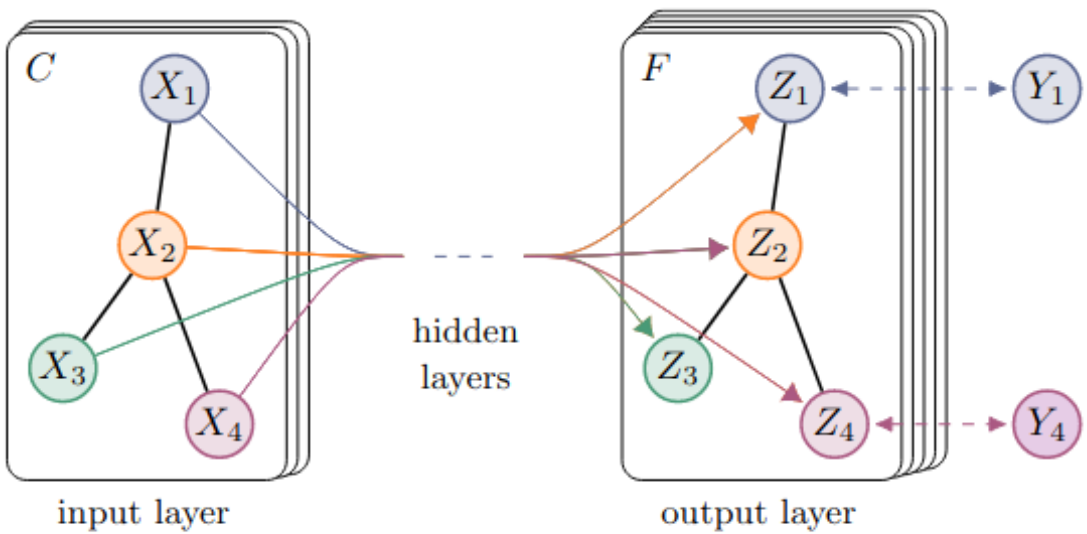
Graph Attention Networks

Given a graph, where

- nodes: entities
- edges: relations

Task:
Which nodes are researchers?

Represent nodes by adding vectors of neighbors

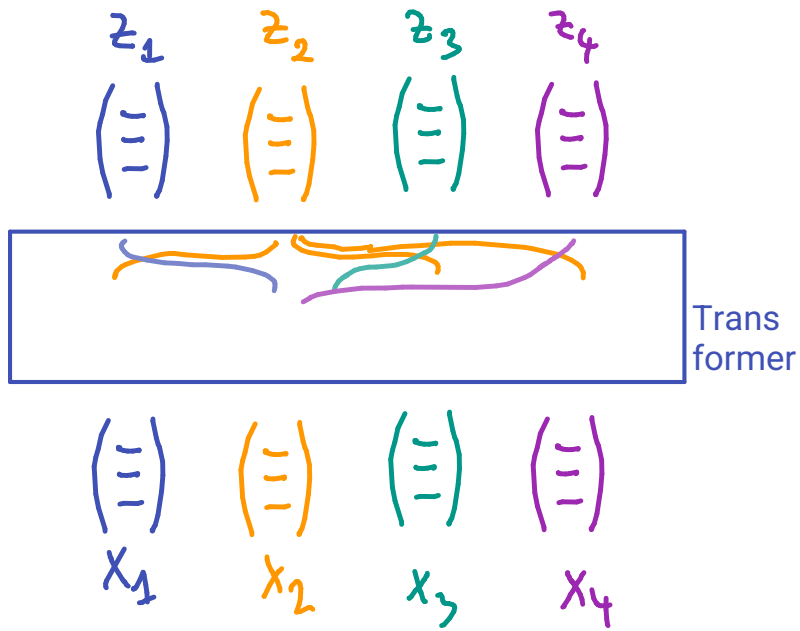


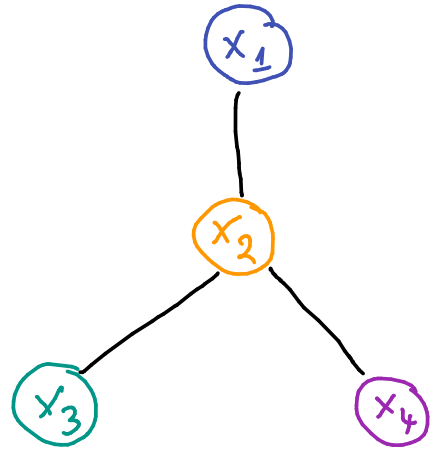
(a) Graph Convolutional Network

(b) Hidden layer activations

Kipf & Welling "SEMI-SUPERVISED CLASSIFICATION WITH GRAPH CONVOLUTIONAL

Graph Attention = Addition of Neighbors



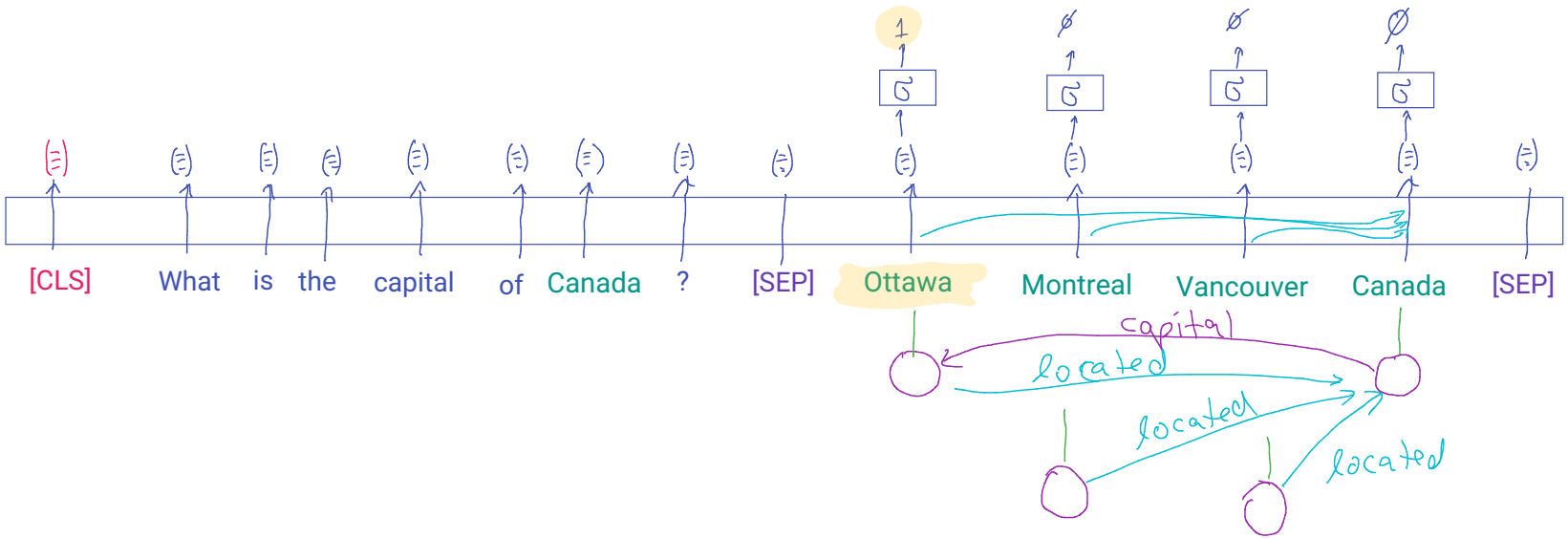


Attention a_{qk} is restricted to edges (k, q) — zero otherwise

$$\mathbf{z_2} = \sum_{k \text{ is neighbor of } 2} \underbrace{a_{2k}}_{\texttt{proj}(\mathbf{x_2}) \cdot \texttt{proj}(\mathbf{x_k})} \texttt{proj}(\mathbf{x_k})$$

Question Answering with Knowledge Graphs

Example: What is the capital of Canada?
Find the node in the knowledge graph that is the answer.



- Combine sequence and graph data via attention
- Graph Attention Networks to select a node = answer.