Relation Extraction

Using Deep Learning

Relation Extraction task

- Relation Extraction is the task of predicting attributes and relations for entities in a sentence

Barack Obama was born in Honolulu, Hawaii.

e_1 e_2 e_3

<u>**predict each relation - *r***</u>

(Barack Obama, ***r1***, Honolulu)

(Barack Obama, ***r2***, Hawaii)

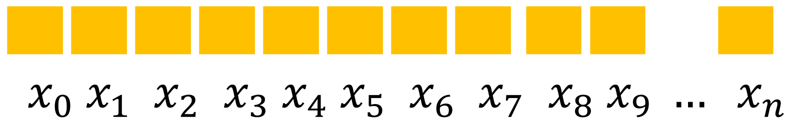
(Honolulu, ***r3***, Hawaii)

Ideally - what should be predicted

r1 = was born in
r2 = was born in
r3 = is in

Matching the Blanks: Distributional Similarity for Relation Learning https://arxiv.org/pdf/1906.03158.pdf

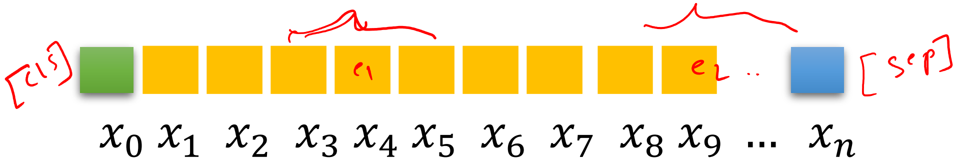
- The focus is to learn mappings from natural language relation statements to relation representations.
- Formally, let $x=[x_0 \ldots x_n]$ be a sequence of tokens, where x_0=[CLS] and x_n=[SEP] are special start and end markers.
- Let $s_1=(i, j)$ and $s_2=(k, l)$ be pairs of integers such that $0 < i < j-1$, $j < k$, $k \leq l-1$ and $l \leq n$.
- A relation statement is a triplet $r=(x, s_1, s_2)$, where the indices in s_1 and s_2 delimit entity mentions in x:
  - the sequence $[x_i \ldots x_{j-1}]$ mentions an entity, and so does the sequence $[x_k \ldots x_{l-1}]$.



Matching the Blanks: Distributional Similarity for Relation Learning

https://arxiv.org/pdf/1906.03158.pdf

- The focus is to learn mappings from natural language relation statements to relation representations.
- Formally, let $x=[x_0 \ldots x_n]$ be a sequence of tokens, where x_0=[CLS] and x_n=[SEP] are special start and end markers.
- Let $s_1=(i, j)$ and $s_2=(k, l)$ be pairs of integers such that $0 < i < j-1$, $j < k$, $k \le l-1$ and $l \le n$.
- A relation statement is a triplet $r=(x, s_1, s_2)$, where the indices in s_1 and s_2 delimit entity mentions in x:
  - the sequence $[x_i \ldots x_{j-1}]$ mentions an entity, and so does the sequence $[x_k \ldots x_{l-1}]$.



$s_1=(i,j)$ $s_2=(k,l)$

$r=\{x, s_1, s_2\}$

$R_1 \to$ President of
$R_2 \to$ Son of
$R_3 \to$ born in
⋮

$$\tilde{\mathbf{r}}^i = (\tilde{\mathbf{x}}^i, \mathbf{s}_1^i, \mathbf{s}_2^i)$$

$r^{(1)}$ Barack Obama, the 44th president of the United States

$r^{(2)}$ Barack Obama was born in Honolulu, Hawaii.

The goal is to learn a function $h_r=f_\theta(r)$ that maps the relation statement to a fixed-length vector $h_r \in \mathbb{R}^d$ that represents the relationship expressed in x between the entities marked by s_1 and s_2.

Key Intuition

e2 US
e1 BO born in US
is the of
s1 → BO
s2 → BO

Barack Obama born in Hono...
Honolulu

BO

Key insight: distributional similarity, applied to relations

"... if two entities participate in a relation, any sentence that contain those two entities ~~might~~ express that relation." [Mintz et al., 2009]

is more likely to

Jeff Buckley, inspired by Cale's earlier cover, recorded one of the most acclaimed versions of "Hallelujah"

e_1 e_2

Sing

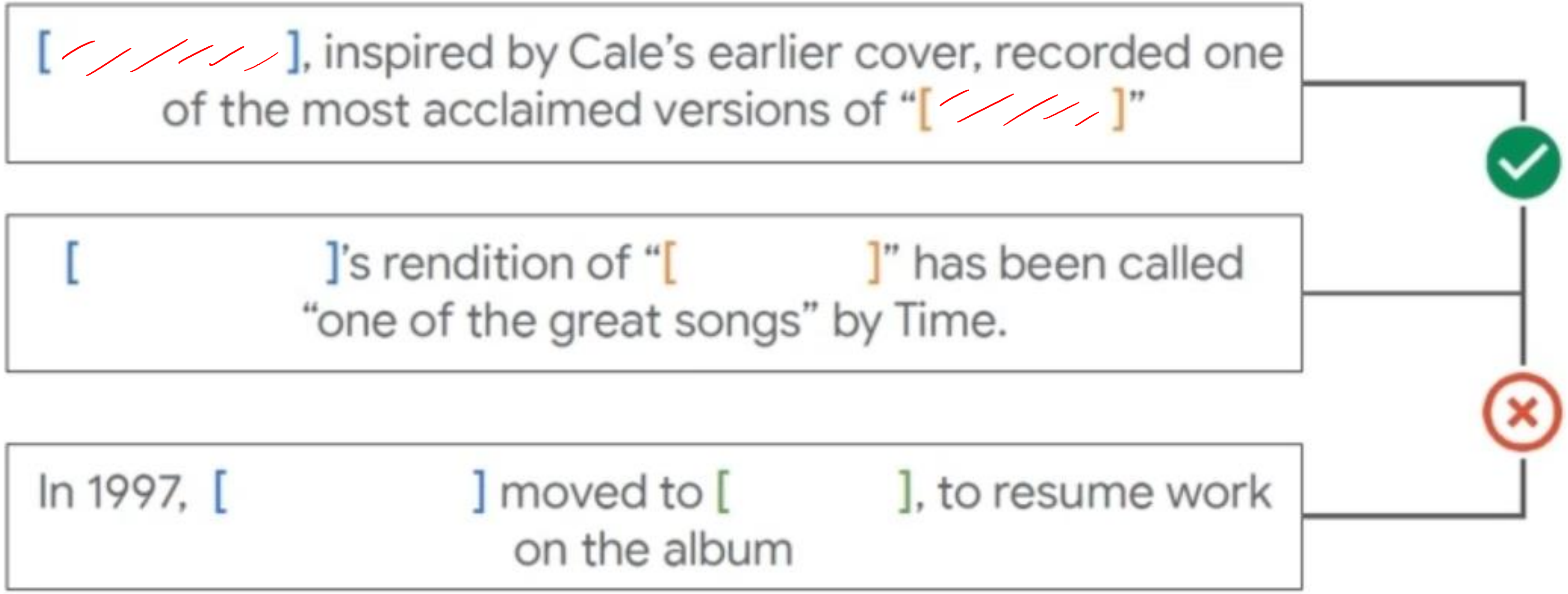
e_2

Jeff Buckley's rendition of "Hallelujah" has been called "one of the great songs" by Time.

e_1 Sing

In 1997, Jeff Buckley moved to Memphis, to resume work on the album

Key Intuition



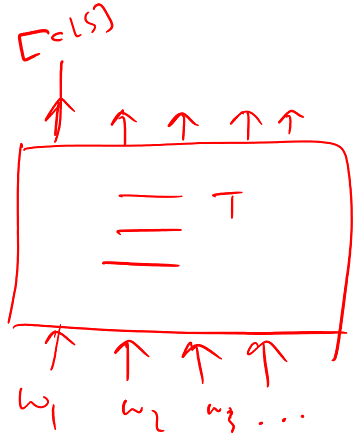
Blank out entity mentions to force model to focus on **relation context** (but not all the time)

Architectures for relational learning using BERT

- Two key questions have to be answered:
  1. How can the entities of interest be presented to the model?
  2. How can one extract a fixed-length relation representation from BERT's output?

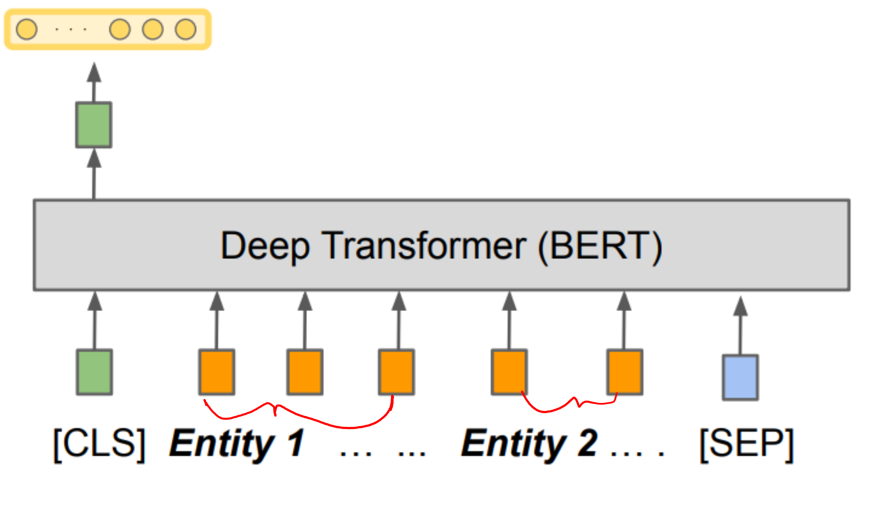
Barack Obama was born in Honolulu, Hawaii.

e_1 e_2 e_3



How can the entities of interest be presented to the model?

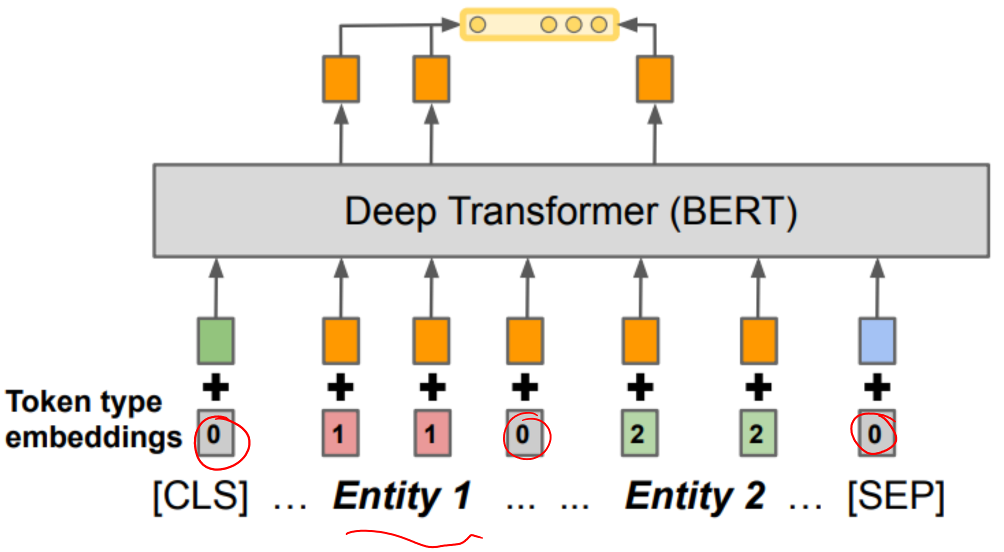
1. The easiest way is to apply the ostrich strategy and completely ignore it. This however raises problems in longer sentences with multiple entities as the model does not know about the entity combination for which it should extract the relation.



(a) STANDARD – [CLS]

How can the entities of interest be presented to the model?

2. The second approach is passing positional embeddings for each word in the sentence that serve as entity markers.



POSITIONAL EMB. – MENTION POOL.

Position Embedding

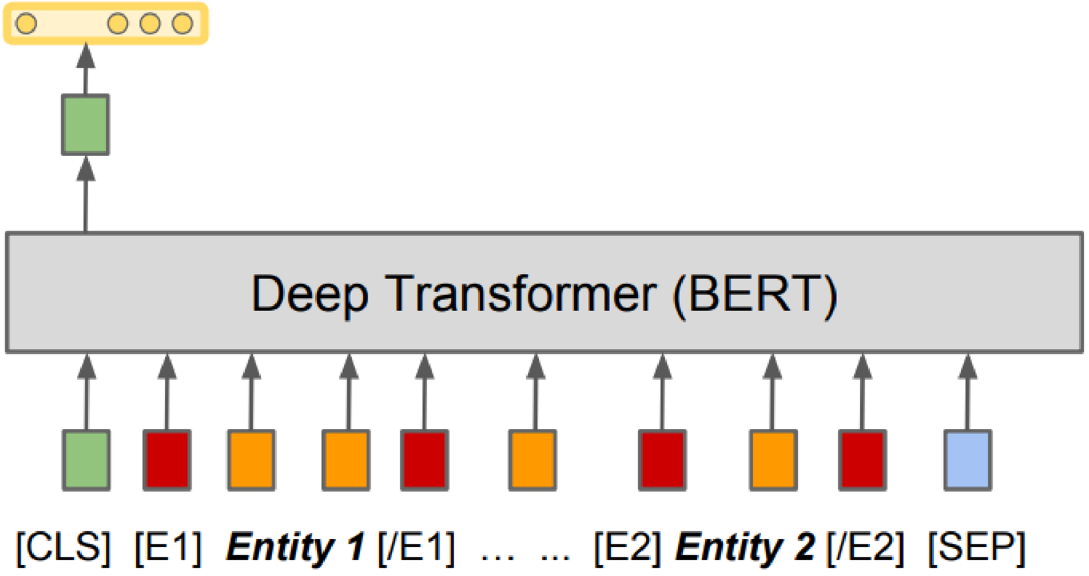
- Why Position Embedding matters?

  - Even though she did not win the match, she was satisfied
  - Even though she did win the match, she was not satisfied

How can the entities of interest be presented to the model?

3. The final and best working approach is to wrap the entities of interest in entity marker icons so that the final sequence would look like this.

$$x = [x_0 ... [E1_{start}] x_i ... x_{j-1} [E1_{end}] ... [E2_{start}] x_k ... x_{l-1} [E2_{end}] ... x_n]$$

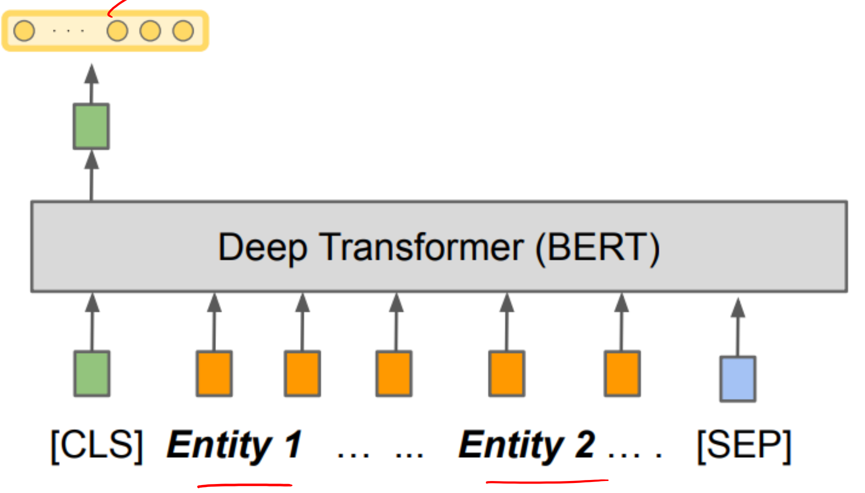


$s_1 = (i, j)$, $s_2 = (k, l)$

$$\boxed{\tilde{s}_1 = (i+1, j+1), \tilde{s}_2 = (k+3, l+3)}$$

How to extract a fixed-length relation representation:

- The first approach is fairly simple using the [CLS] token of BERT and adopt its embedding output h_0 as relation representation.



(a) STANDARD – [CLS]

$R_1 \to \langle v_1 \rangle$
R_2 $\langle v_2 \rangle$
R_3
R_k
R_{50}

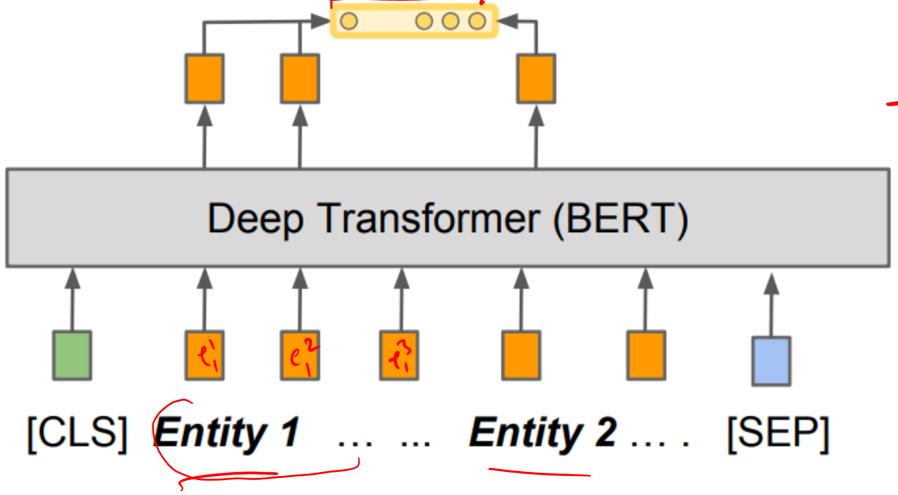
How to extract a fixed-length relation representation:

- The second approach is to obtain the entity relation representation h_r by max pooling the final hidden layers corresponding to the tokens in each entity mention to get two embeddings h_{e1}=MAXPOOL($[h_i \dots h_{j-1}]$) and h_{e2}=MAXPOOL($[h_k \dots h_{l-1}]$) which will then be concatenated to form the final relation representation h_r.

h_{e_3}

h_{e_4}

$h_{e_1} \oplus h_{e_2}$



STANDARD – MENTION POOLING

R_1

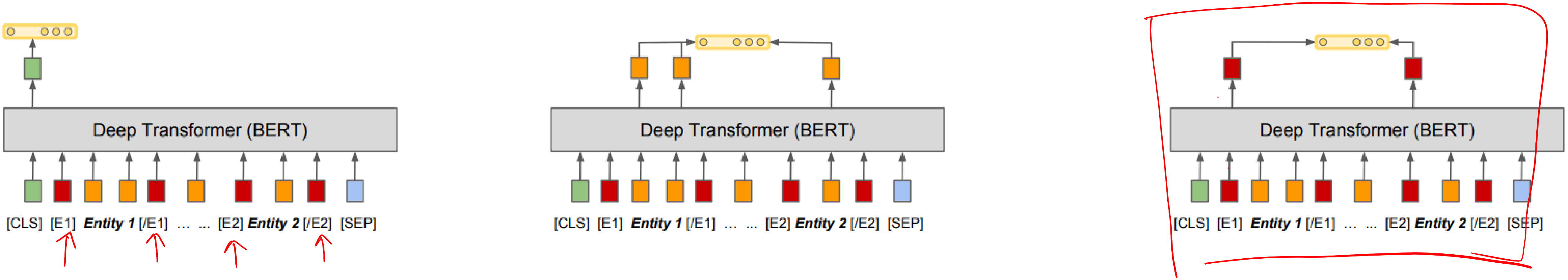
R_2

$e_1 \to h_1^{(1)}\ h_2^{(2)}\ h_3^{(1)}$ — M

$e_2 \to h_1^{(1)}\ h_2^{(1)}\ h_3^{(1)}$

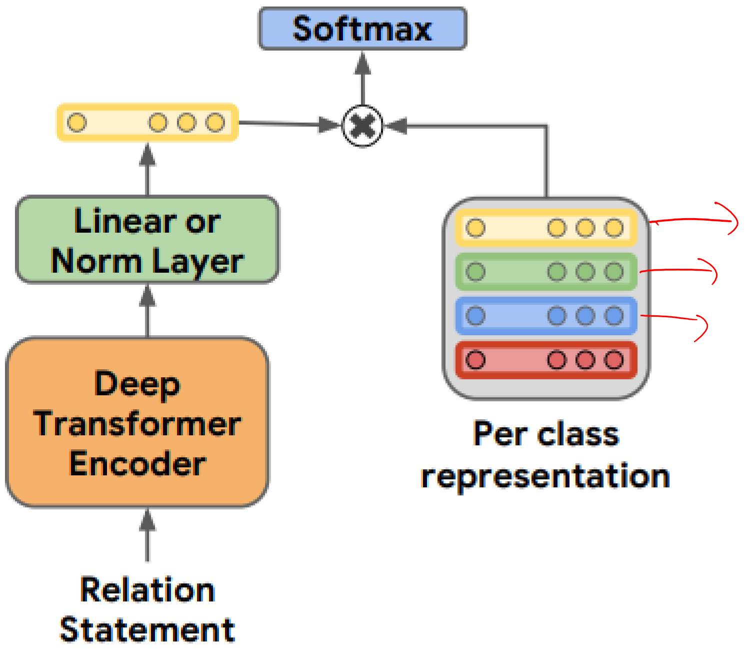
How to extract a fixed-length relation representation:

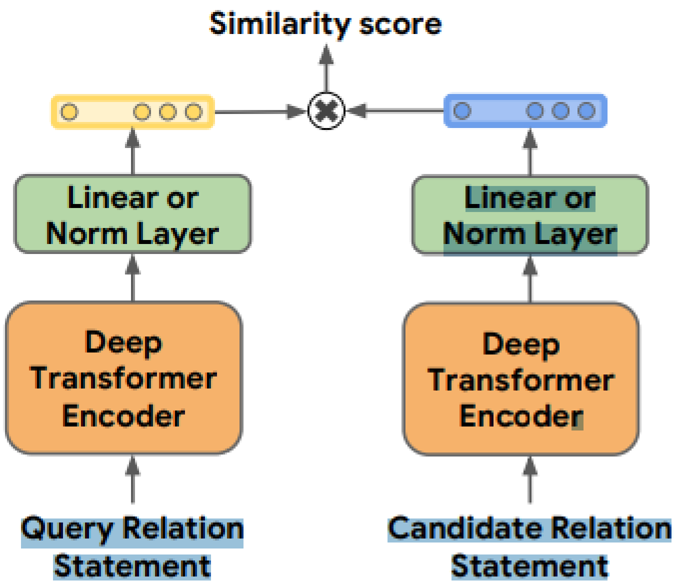
- Finally, the authors propose an approach they refer to as *Entity Start State.*
- The approach is pretty straightforward, it just concatenates the final hidden states of the Entity start tokens $E1_s$ and $E2_s$.



Note: this approach can only be applied if one uses the entity markers to solve the first problem. The image below depicts the different approaches of defining an architecture that is able to learn relation representations of two entities given a sequential input.

Illustration of losses used in our models





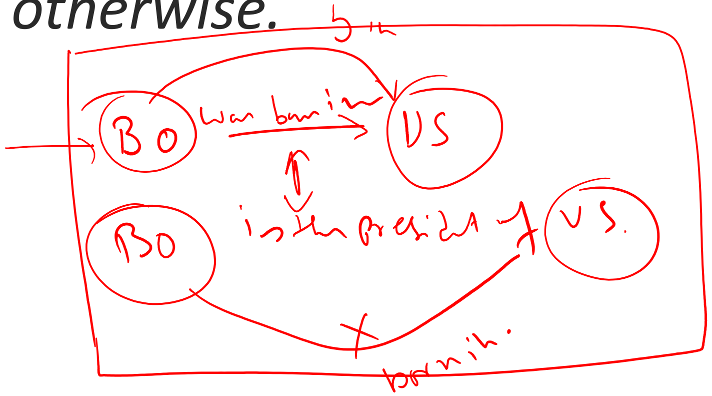
The left figure depicts a model suitable for supervised training, where the model is expected to classify over a predefined dictionary of relation types. The figure on the right depicts a pairwise similarity loss used for few-shot classification task.

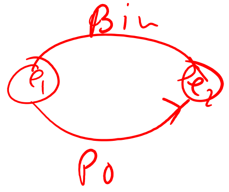
Introducing Blanks

- We're able to train a relation extractor on pre-labeled data.
- However, finding a labeled dataset that has the necessary size is hard, so we need to find a way to train the model on an unlabeled dataset.
- To get to that point one has to make a few assumptions:

$r \rightarrow h^{(r)} \sim h^{(t)}$

For any pair of relation statement r and t, the inner product $f\vartheta(r)\, f\vartheta(t)$ should be high if r and t express semantically similar relation statements and low otherwise.





Introducing Blanks

Given those assumptions, the authors aim to learn a statement encoder fθ that can be used to determine whether or not two relation statements encode the same relation. To do that they define the following binary classifier

$$p\left(l=1 \mid \mathbf{r}, \hat{\mathbf{r}}\right)=\frac{1}{1+\exp f_{\theta}(\mathbf{r})^{\top} f_{\theta}(\hat{\mathbf{r}})}$$

to assign a probability to the case that r and r̂ encode the same relation (l=1) or not.

Introducing Blanks

[~~BLANK~~], inspired by Cale's earlier cover, recorded one of the most acclaimed versions of "[BLANK]"

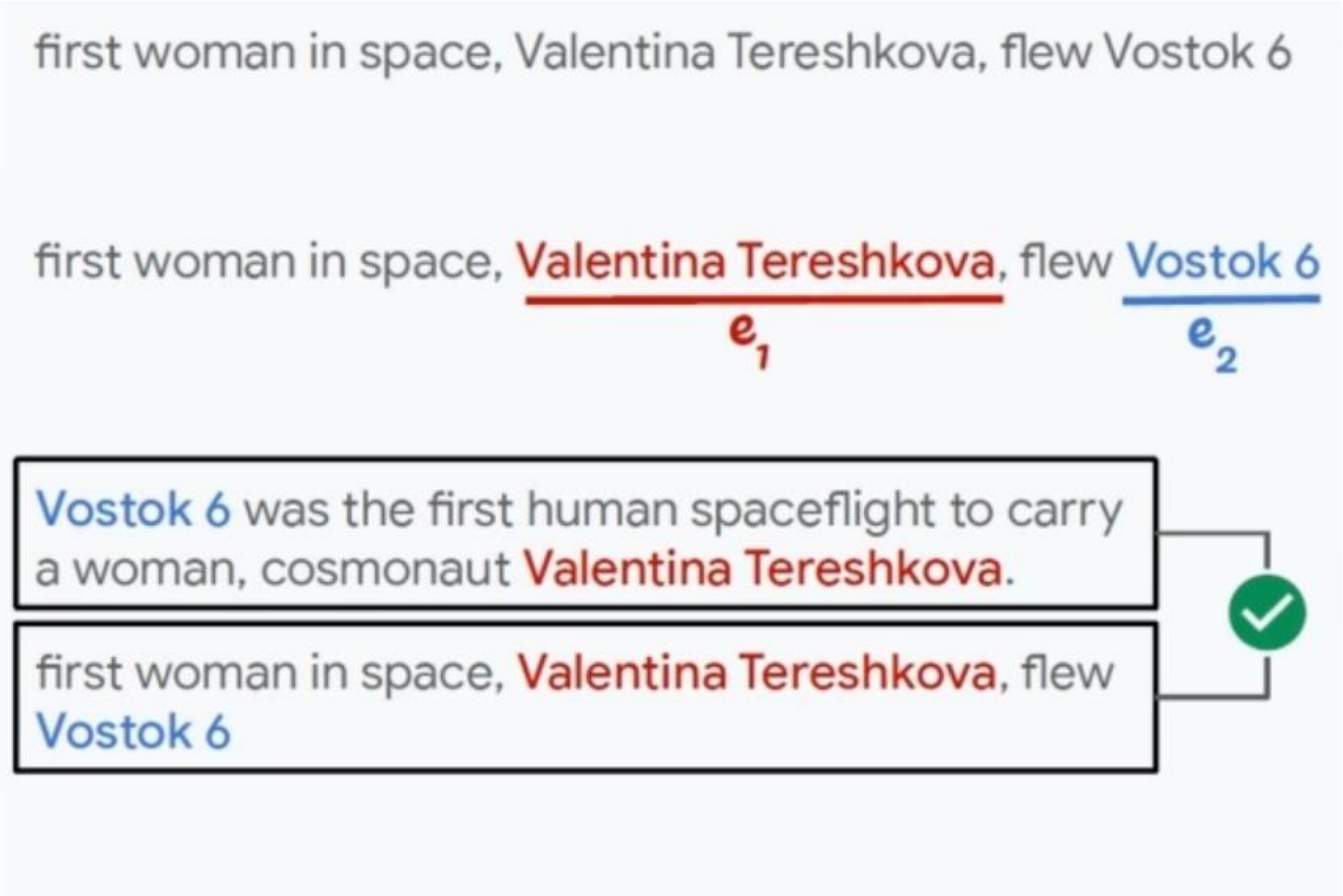
[BLANK]'s rendition of "[BLANK]" has been called "one of the great songs" by Time, and is included on Rolling Stone's list of "The 500 Greatest Songs of All Time".

Example of "matching the blanks" automatically generated training data

$\mathbf{r}_A$	In 1976, e_1 (then of Bell Labs) published e_2, the first of his books on programming inspired by the Unix operating system.
$\mathbf{r}_B$	The "e_2" series spread the essence of "C/Unix thinking" with makeovers for Fortran and Pascal. e_1's Ratfor was eventually put in the public domain.
$\mathbf{r}_C$	e_1 worked at Bell Labs alongside e_3 creators Ken Thompson and Dennis Ritchie.
Mentions	e_1 = Brian Kernighan, e_2 = Software Tools, e_3 = Unix

Training

1. English Wikipedia
2. Annotate with entity ids
3. At most 40 tokens
4. At least 2 grounded entities
5. Pair statements with same 2 entities

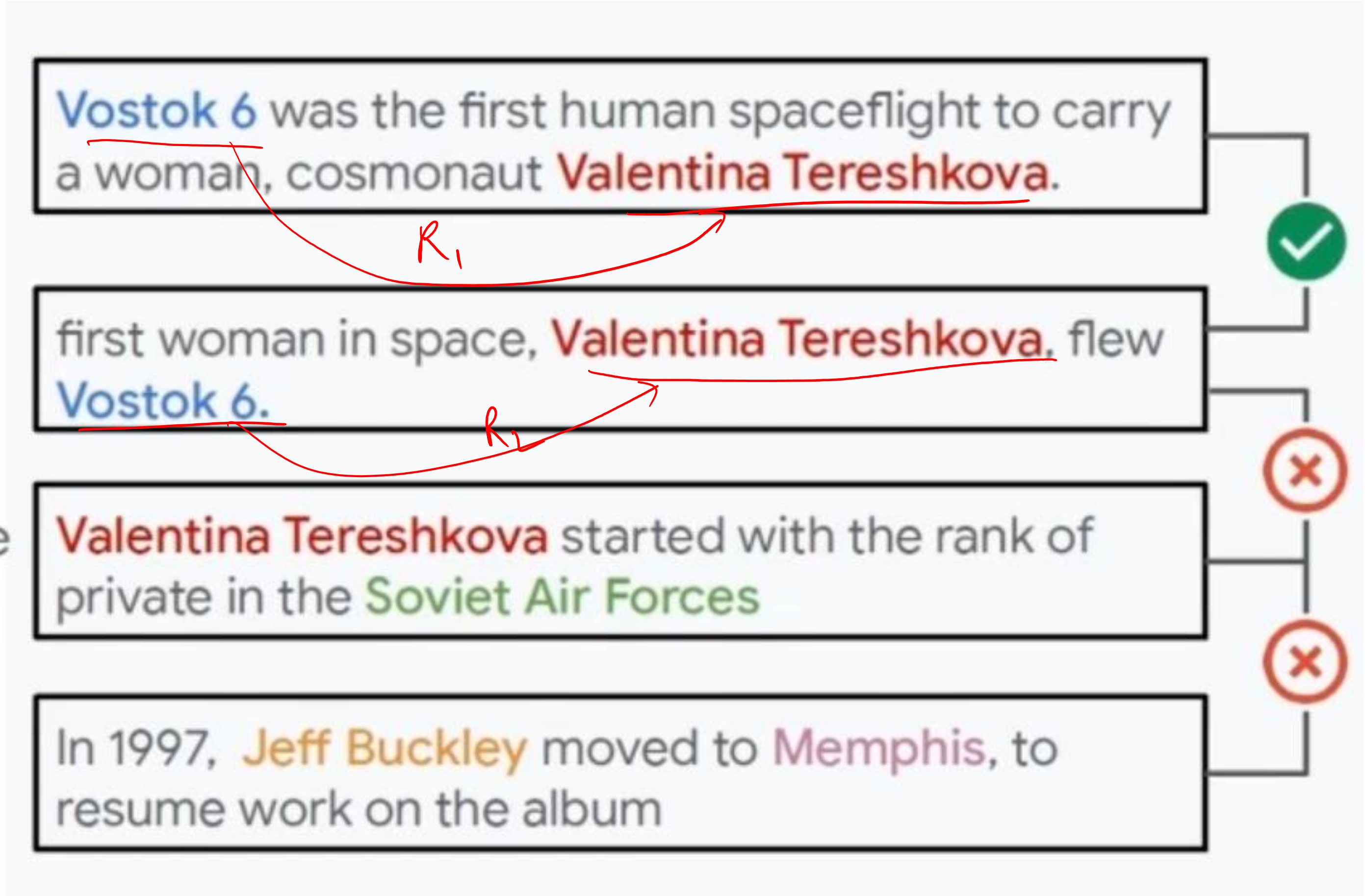


Training

Two types of negatives:

6. For *hard* negatives, pair contexts with a single shared entity

7. We also use batch examples as *easy* negative examples



Training

Two types of negatives:

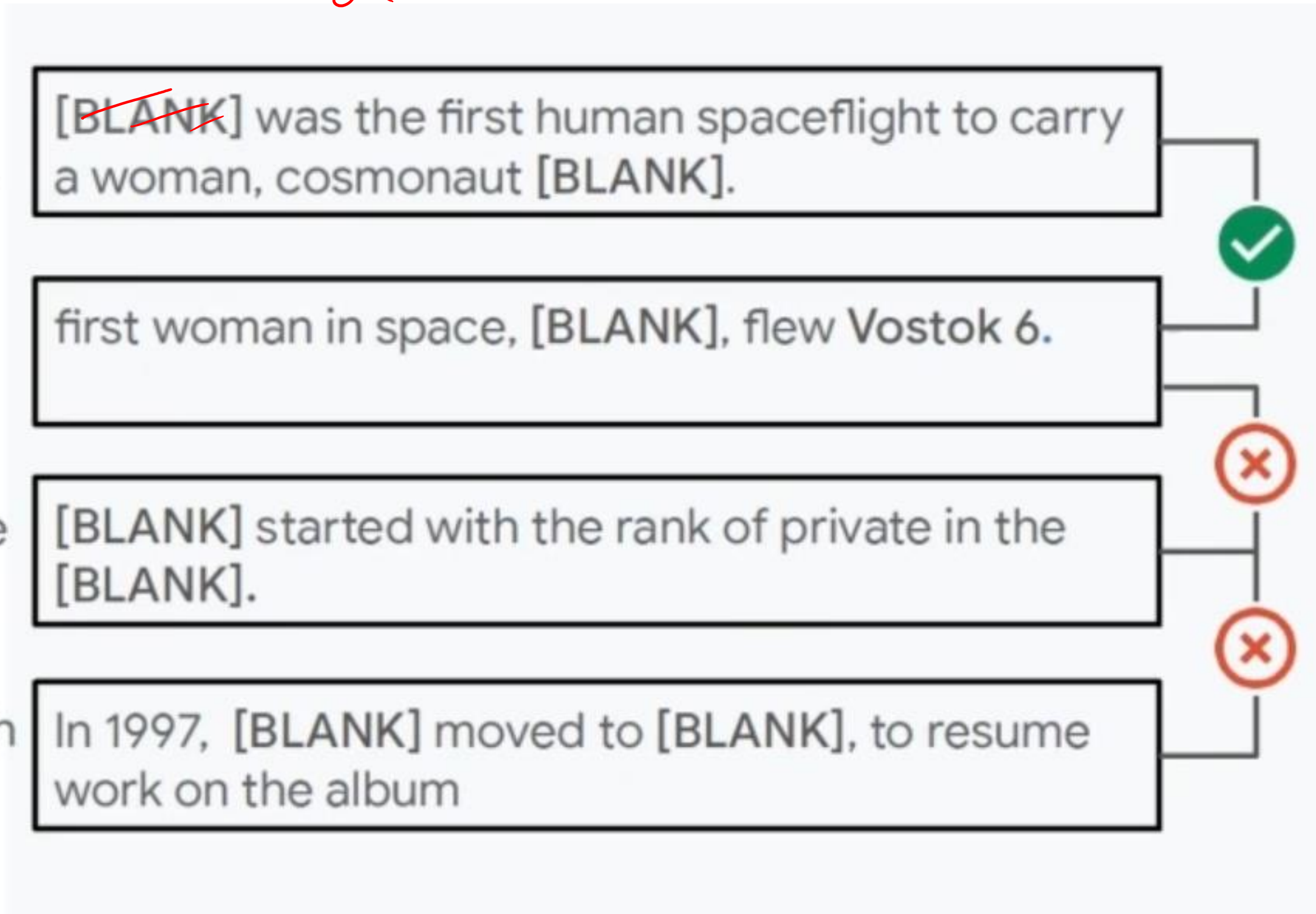
6. For *hard* negatives, pair contexts with a single shared entity

$\alpha = 70\%$.

7. We also use batch examples as *easy* negative examples

8. Blank out mentions (with 70% probability)

$\alpha =$



RESIDE: Improving Distantly-Supervised Neural Relation Extraction using Side Information

https://aclanthology.org/D18-1157/

Matt coffin is an Exe. of lower my bills

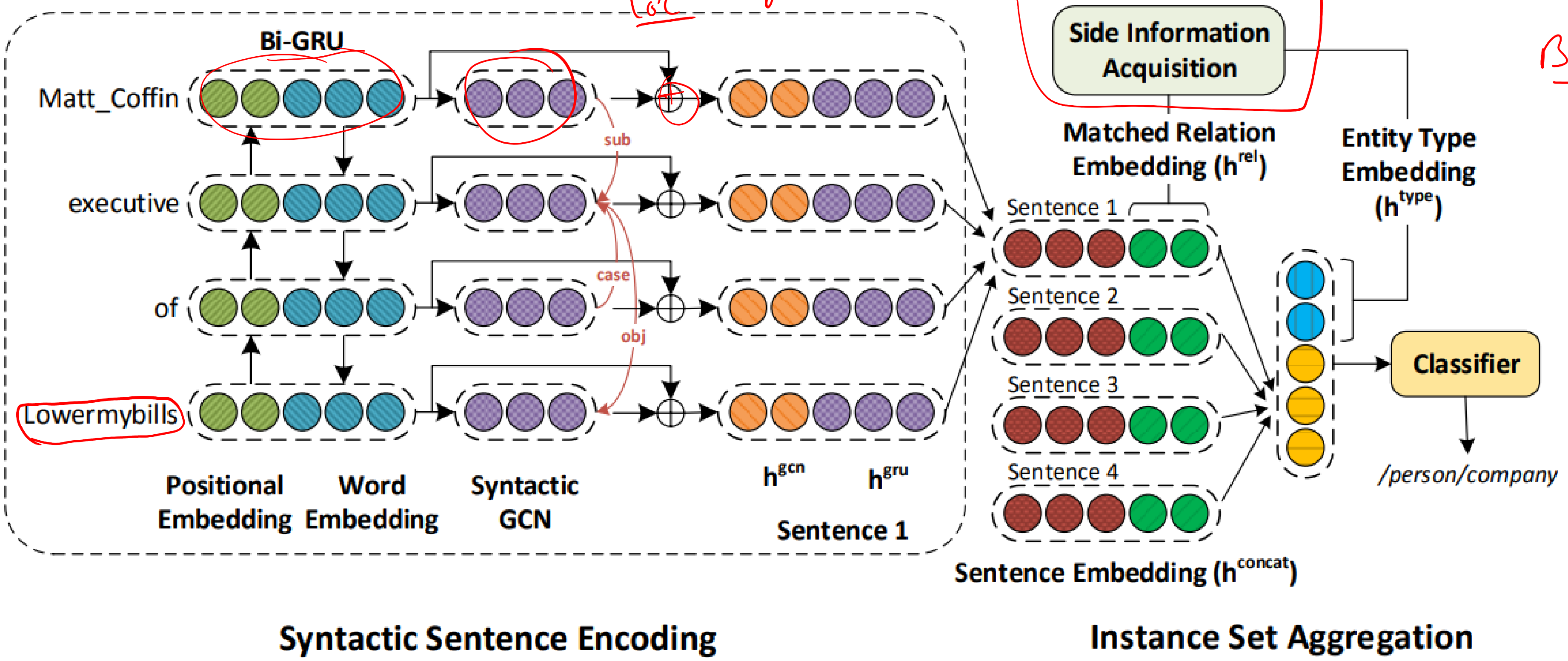
→ Loc.

→ FO

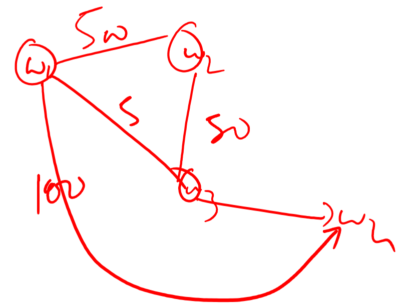
Loc org name.

Freebase

BO → WS



Syntactic Sentence Encoding:



- GLOVE + Positional Embedding
  - The combined token embeddings are stacked together to get the sentence representation.
- Followed by Bi-GRU
- While Bi-GRUs are capable of capturing local context, it fails to capture long-range dependencies which can be captured through dependency edges.
  - The captain who the sailor greeted is tall
  - The captain who the sailor predicted that the weather would frighten turned back to port.
  - The rat the cat the dog chased killed ate the malt
- Employ Syntactic Graph Convolution Networks for encoding this information.

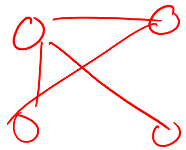
w_0 w_1 w_2 w_3

GCN on Labeled Directed Graph

- For a directed graph, G = (V, E), where V and E represent the set of vertices and edges respectively, an edge from node u to node v with label luv is represented as (u, v, l_{uv}).
- On employing GCN, we get an updated d-dimensional hidden representation h_v.

$$h_v = f\left(\sum_{u \in \mathcal{N}(v)} (W_{l_{uv}} x_u + b_{l_{uv}})\right)$$

$$h_v^{k+1} = f\left(\sum_{u \in \mathcal{N}(v)} \left(W_{l_{uv}}^k h_u^k + b_{l_{uv}}^k\right)\right)$$



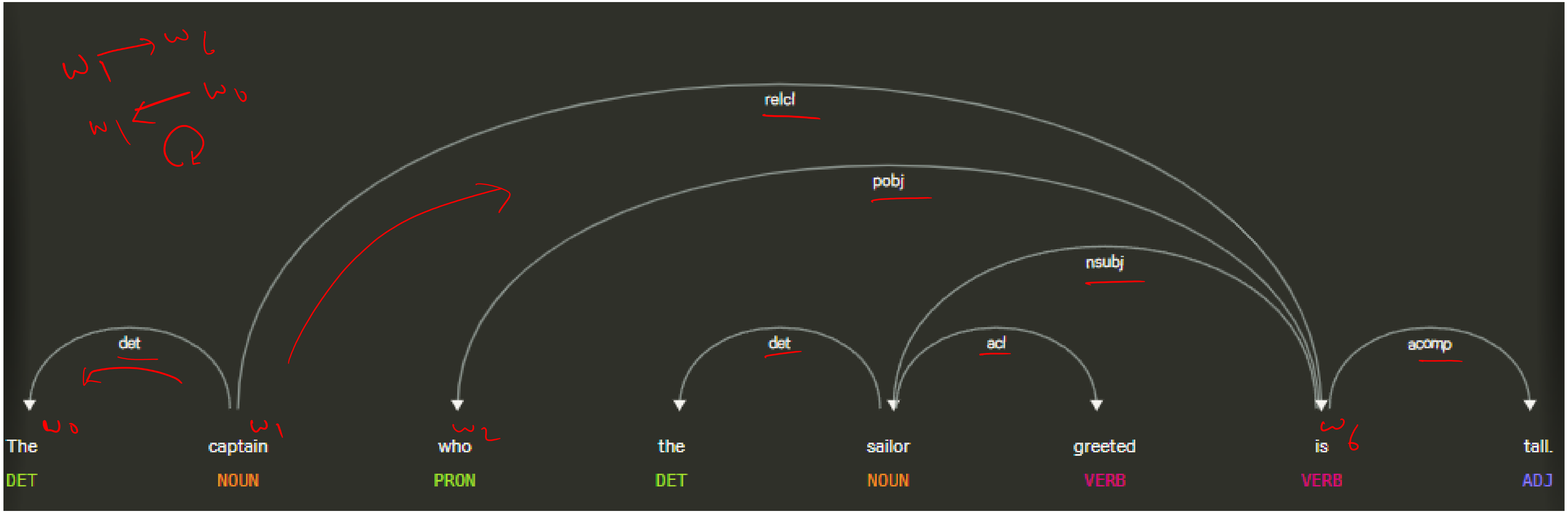
- Integrating Edge Importance: $g_{uv}^k = \sigma\left(h_u^k \cdot \hat{w}_{l_{uv}}^k + \hat{b}_{l_{uv}}^k\right)$

- The final GCN embedding for a node v after kth layer is given as:

$$h_v^{k+1} = f\left(\sum_{u \in \mathcal{N}(v)} g_{uv}^k \times \left(W_{l_{uv}}^k h_u^k + b_{l_{uv}}^k\right)\right)$$

GCN on Labeled Directed Graph

- For a given sentence, we generate its dependency tree using Stanford CoreNLP.



GCN on Labeled Directed Graph

nSub
pSub

- For a given sentence, we generate its dependency tree using Stanford CoreNLP.
- Then run GCN over the dependency graph

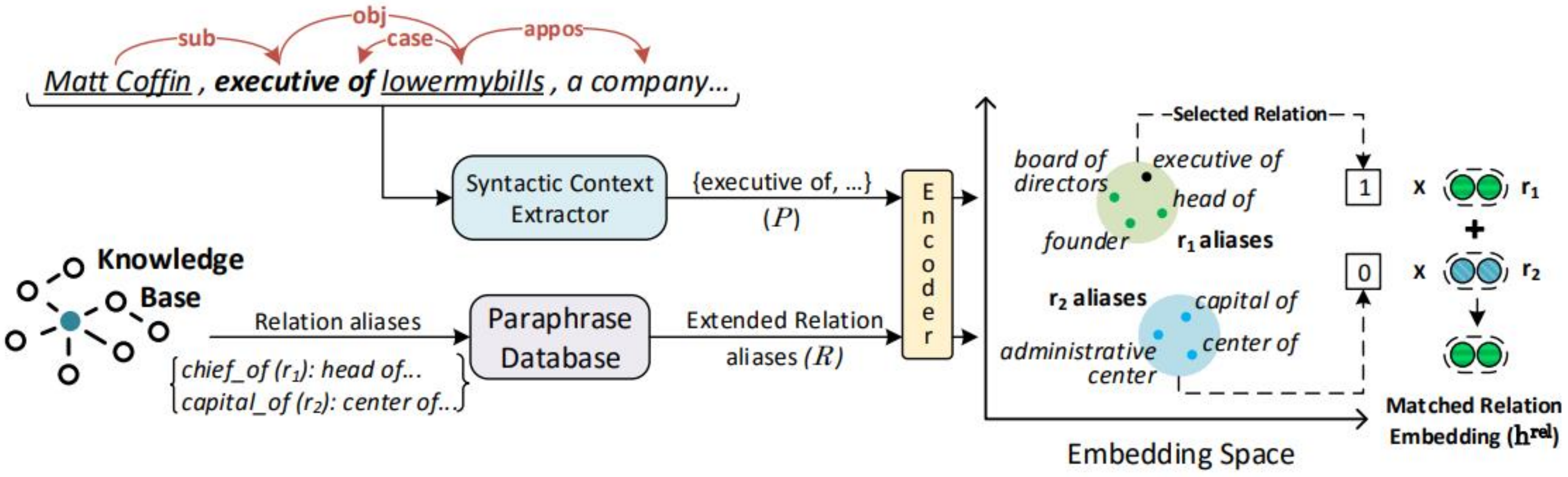
$$L_{uv} = \begin{cases} \rightarrow & \text{if edge exists in dependency parse} \\ \leftarrow & \text{if edge is an inverse edge} \\ \top & \text{if edge is a self-loop} \end{cases}$$

For each token w_i, GCN embedding $h^{gcn}_{i_{k+1}} \in \mathbb{R}^{d_{gcn}}$ after k^{th} layer is defined as:

$$h^{gcn}_{i_{k+1}} = f\left(\sum_{u \in \mathcal{N}(i)} g^k_{iu} \times \left(W^k_{L_{iu}} h^{gcn}_{u_k} + b^k_{L_{iu}} \right) \right).$$

Side Information Acquisition:

- In addition to relation instances, KBs often contain other relevant side information, such as aliases of relations (e.g., founded and co-founded are aliases for the relation founderOfCompany).



Resources

- SemEVAL-2010 Task-8: https://semeval2.fbk.eu/semeval2.php?location=tasks&area=Semantic%20relations
- KBP37: https://github.com/davidsbatista/Annotated-Semantic-Relationships-Datasets
- The TAC Relation Extraction Dataset: https://nlp.stanford.edu/projects/tacred/
- Riedel: https://github.com/davidsbatista/Annotated-Semantic-Relationships-Datasets/blob/master/README.md
- GIDS: https://research.googleblog.com/2013/04/50000- lessons-on-how-to-read-relation.html

Causal Relations

- Special type of relation extraction task
- Relationship between two events e_1 and e_2, where e_1 results in the occurrence of e_2.

Examples

- Causality: Relationship between two events e_1 and e_2, where e_1 results in the occurrence of e_2.

$e_1 \xrightarrow{\vec{c}} e_2$

e_1

Toyota recalls 150000 cars due to faulty airbags

OSHA cited J&J company for not removing employees from a hazardous work area and for failing to install cave-in protection

Serious adverse effect was observed in patients with heart disease due to high dosage of Spironolactone

Formal definition

- Causality can formally be defined as a binary function $f: E \times E \rightarrow \{0,1\}$ where

$$f(e_1, e_2) = \begin{cases} 1 & , if\ e_1\ causes\ e_2 \\ 0 & otherwise \end{cases}$$

- An event e_i is defined in terms of the six tuples $E = < P, A, O, I, L, t >$.
  - P → is the temporal action or state that the event's objects exhibit,
  - A → is the actor/entity performing P,
  - O → is the object/entity on which P is performed,
  - I → is the instrument with which the P was performed,
  - L → is the location and
  - t → is the actual time-stamp.

Formal definition

$e_1, e_2, e_3 \rightarrow e_k$

- **Case-I:**

$$f(e_1, e_2) = \begin{cases} P(e_1|e_i) & \text{if } e_1 \text{ causes } e_i, \\ 0 & \text{otherwise} \end{cases}$$

$e_1 \rightarrow$ Fb e_2

mass destruction e_3

e_4

- **Case-II:** a set of events $E' = \{e_1, e_2, \dots e_n\}$ together causes a particular effect e_p

$$f(E', e_p) = \begin{cases} P(e_p|E') & \text{if } E' \text{is the set of events causing } e_p, \\ 0 & \text{otherwise} \end{cases}$$

$e_i^{t_i} \rightarrow e_j^{t_j}$

- **Case-III:** any event e is associated with time-stamp t. Therefore, a sequence of temporal events $E_i^{t_k}$ can cause an event at a latter time-stamp t_{k+1}.

$$f\left(E_i^{t_k}, {e_p}^{t_{k+l}}\right) = \begin{cases} P\left({e_p}^{t_{k+l}} \middle| E_i^{t_k}\right) & * \\ 0 & \text{otherwise} \end{cases}$$

* →where ${E_i}^{t_k}$ is the minimal set of events at timestamp $(k, k+1, \dots k+j)$ causing ${e_p}^{t_{k+l}}$ at timestamp $k+l$ where $i > j$.

Types of causal sentences

- Cause-effect relations can be expressed in arbitrarily complex ways.
- **Marked and unmarked causality.**

Aircel files for bankruptcy over mounting financial troubles

Drive slowly, there are potholes.

- **Explicit and implicit causality.**

The burst has been caused by water hammer pressure

The car ran over his leg.

- **Event chains**

cause-2

effect-2

Maruti recalled 130000 small vehicles to fix loose bolts that could lead to engine stall

effect-1

cause-1

Why?

- Detection of causal relation from text has many analytical and predictive applications.
- Objective - Detect cause-effect relationships from text
- Predictive application
  - Sense events → Predict its possible effects – ***Early Warning Systems***
  - Need to curate large volume of cause-effect event pairs.
  - Similar events need to be grouped and generalized to super classes, over which the predictive frame work can be built

Approaches

- Existing works are based on linguistic rules and statistical machine learning techniques
  - Rule based methods
    - Require large set of rules.
    - Restricted to domains.
  - Supervised machine learning method.
    - Depend on careful feature engineering and linguistic knowledge.
- Deep learning approaches
  - In its nascent stage
  - Requires huge training dataset

CausalNet: Joint Modeling for Detecting Causal Relations from Text