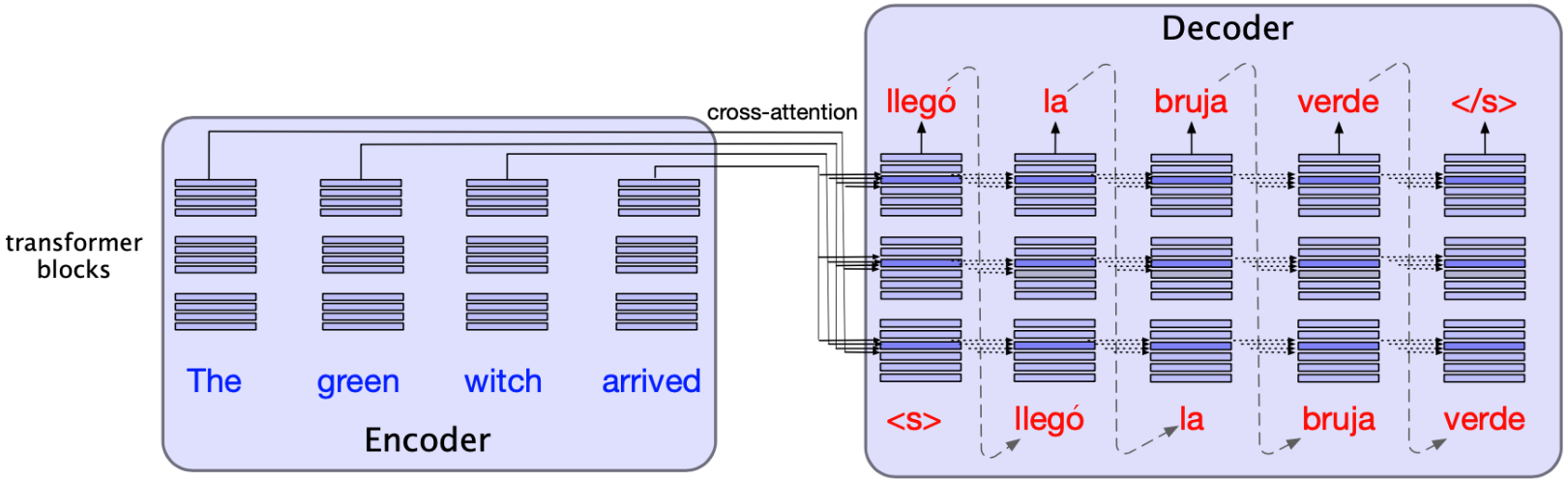
Machine Translation, RAG, and Evaluation

Machine Translation

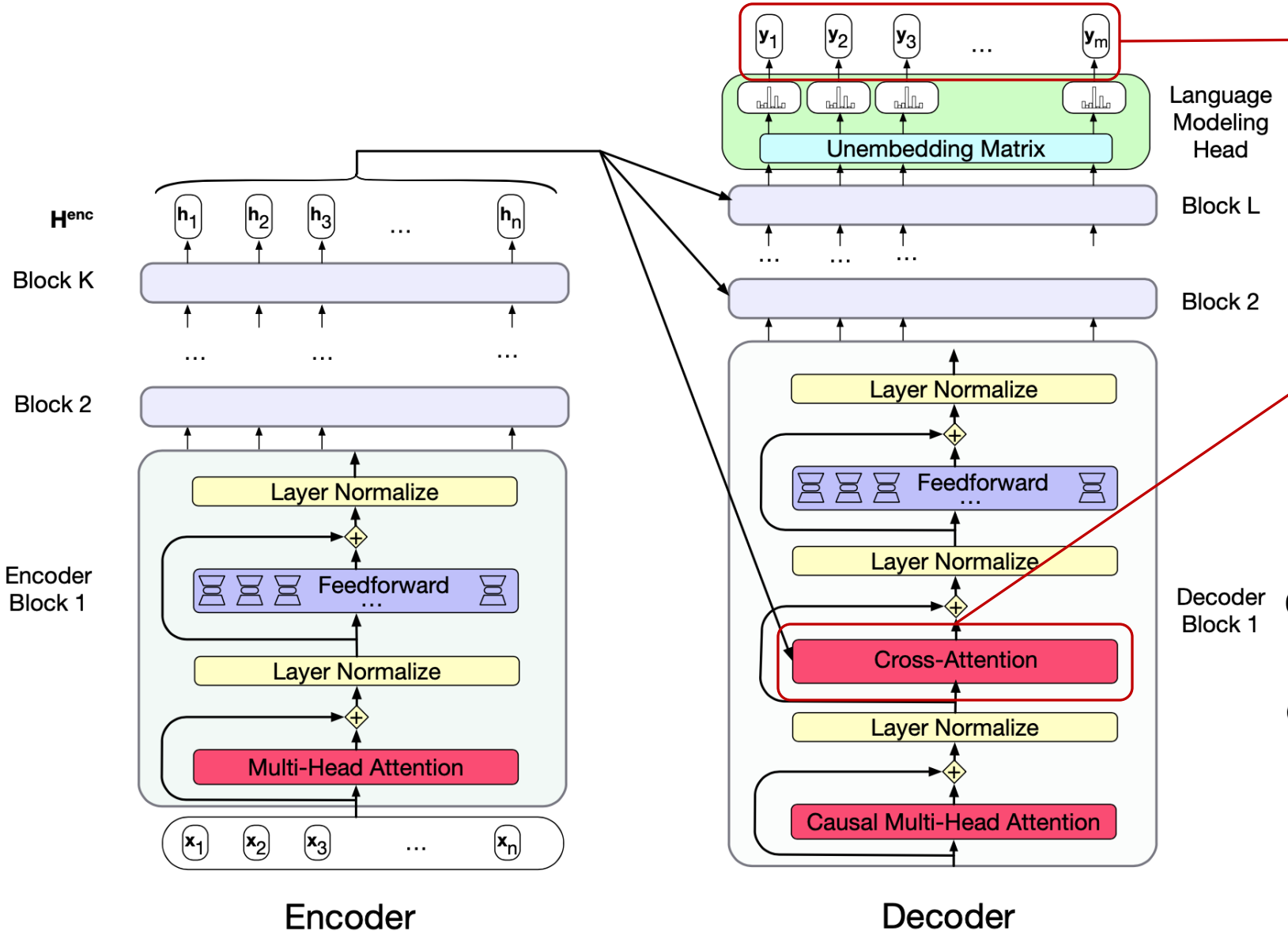
Linguistic issues in Machine Translation

- Languages have some *universal* traits (there are words for referring to people, eating, being polite, and so on)
- Still they have many differences
  - Word Order Typology (SVO, SOV, VSO and use of prepostitions or postpositions)
  - Lexical Divergences (one word in L_S → different words in L_T, where more context is needed)
  - Morphological Typology (from *isolated* to *fused* morphemes)
  - Referential Density (things the language tends to omit)
    - *I talked to Juliet yesterday. She was very sad.* → *Ieri ho parlato con Juliet. Era molto triste.*

MT with the encoder-decoder model



MT with the encoder-decoder model



Teacher forcing with loss:

$$L_{CE}(\hat{\mathbf{y}}_t, \mathbf{y}_t) = -\log \hat{\mathbf{y}}_t[w_{t+1}]$$

Extra layer implementing attention on the encoder context $\mathbf{H}^{enc}$ of size [n x d]

At the layer ℓ:

Decoder Block 1

$$\mathbf{Q} = \mathbf{H}^{dec[\ell-1]}\mathbf{W}^{\mathbf{Q}};\ \mathbf{K} = \mathbf{H}^{enc}\mathbf{W}^{\mathbf{K}};\ \mathbf{V} = \mathbf{H}^{enc}\mathbf{W}^{\mathbf{V}}$$

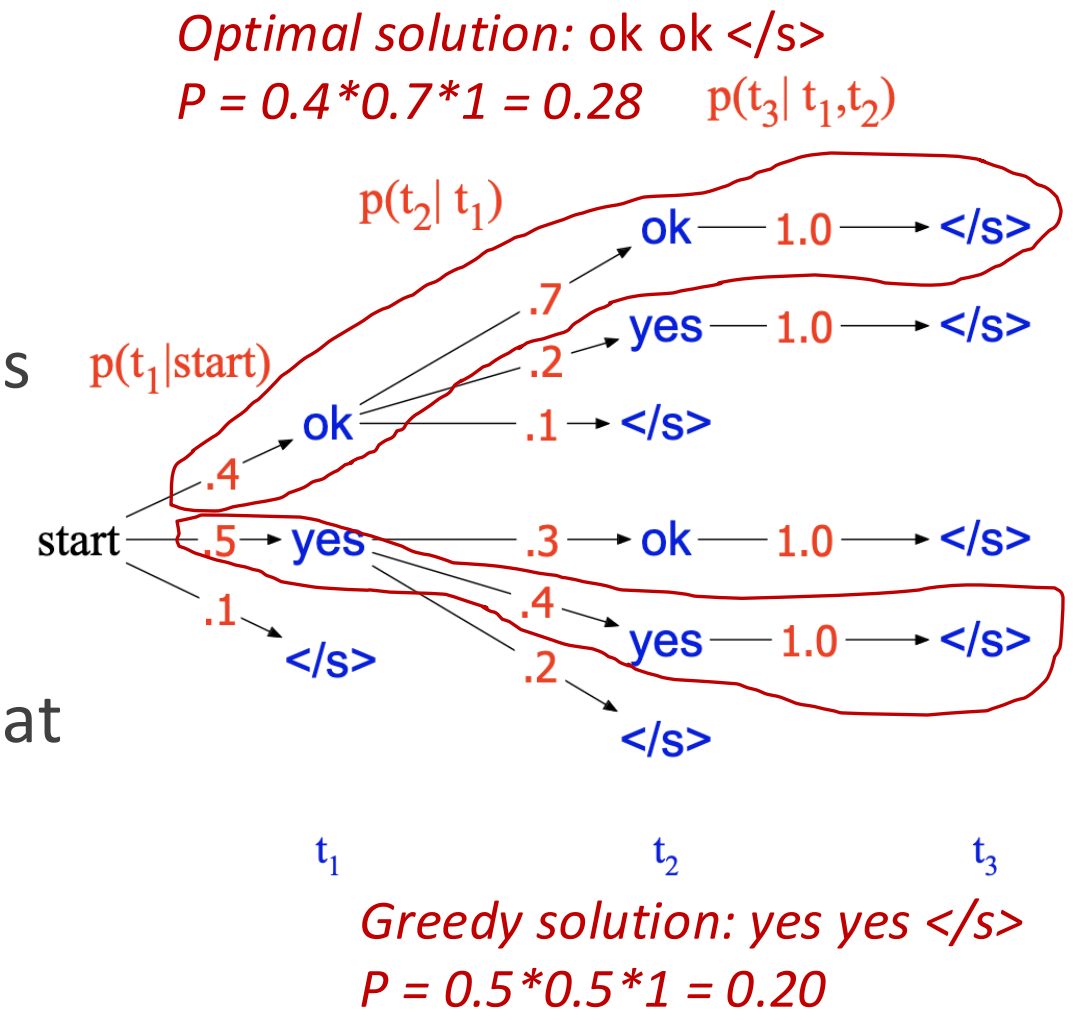
$$\text{CrossAttention}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^{\mathsf{T}}}{\sqrt{d_k}}\right)\mathbf{V}$$

MT with the encoder-decoder model

Decoding with greedy search is *locally optimal* in the space of solutions

We need to implement search strategies that are optimal in a global way

The only algorithm that guarantees optimal solution is exahustive search that is infeasible (V^T possibilities)



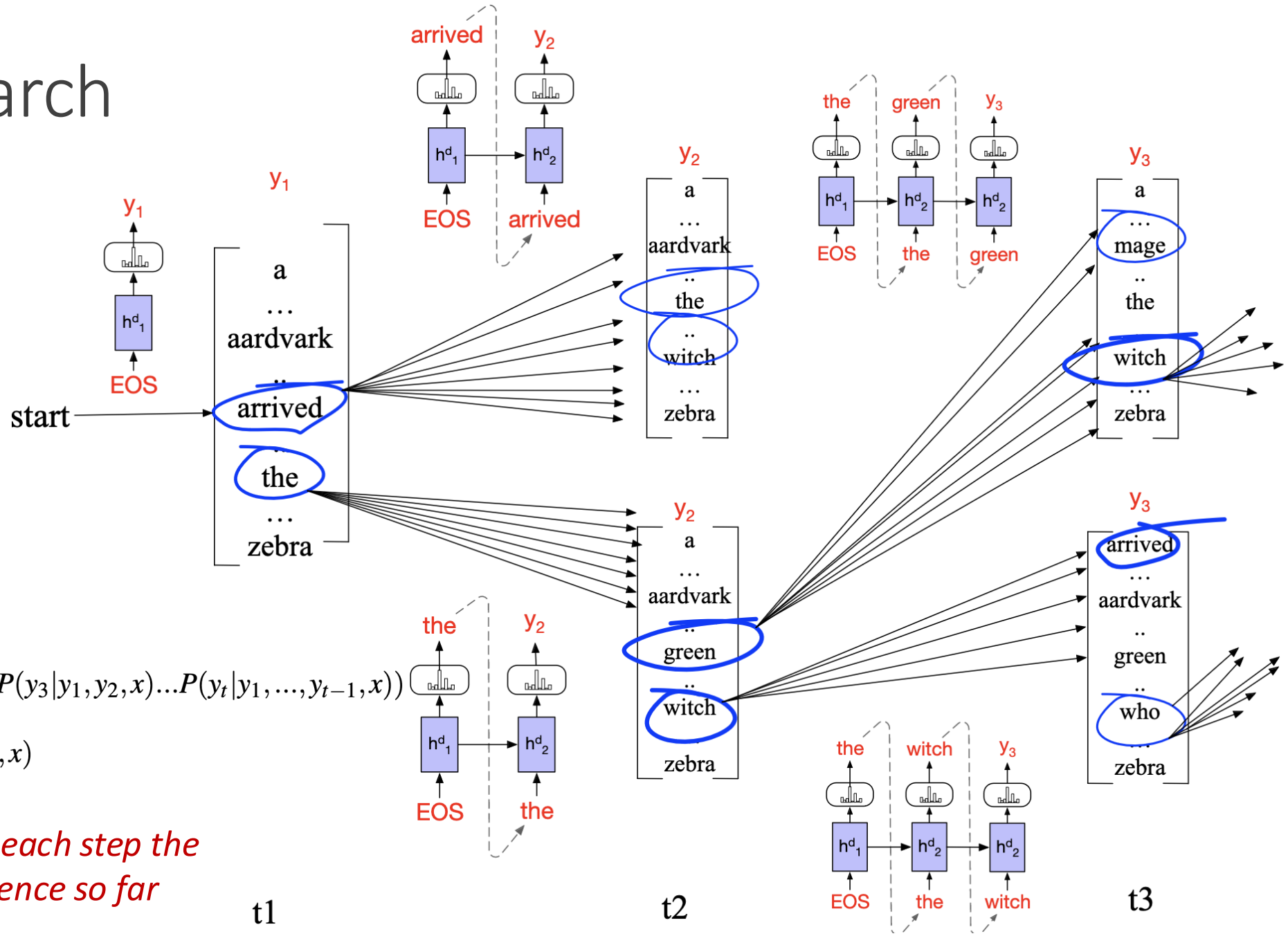
Beam Search

Decoding is achieved keeping the best *k* tokens at each step

1. At each step t_i in the path till the END token:
   a) For each of the *k* best hypotheses so far (i.e. the *k* best generated sequences)
      1. Score the entire vocabulary starting from the current hypothesis by $P(y_i|x, y_{<i})$
   b) Prune the *k*V* scores maintaining the top *k* new hypotheses
2. Remove completed paths and set *k* = *k* - 1
3. Restart from 1 till k == 0

```
function BEAMDECODE(c, beam_width) returns best paths

  y_0, h_0 ← 0
  path ← ()
  complete_paths ← ()
  state ← (c, y_0, h_0, path)          ;initial state
  frontier ← ⟨state⟩                   ;initial frontier

  while frontier contains incomplete paths and beamwidth > 0
      extended_frontier ← ⟨⟩
      for each state ∈ frontier do
          y ← DECODE(state)
          for each word i ∈ Vocabulary do
              successor ← NEWSTATE(state, i, y_i)
              extended_frontier ← ADDTOBEAM(successor, extended_frontier,
                                            beam_width)

      for each state in extended_frontier do
          if state is complete do
              complete_paths ← APPEND(complete_paths, state)
              extended_frontier ← REMOVE(extended_frontier, state)
              beam_width ← beam_width - 1
      frontier ← extended_frontier

    return completed_paths

function NEWSTATE(state, word, word_prob) returns new state

function ADDTOBEAM(state, frontier, width) returns updated frontier

  if LENGTH(frontier) < width then
      frontier ← INSERT(state, frontier)
  else if SCORE(state) > SCORE(WORSTOF(frontier))
      frontier ← REMOVE(WORSTOF(frontier))
      frontier ← INSERT(state, frontier)
  return frontier
```

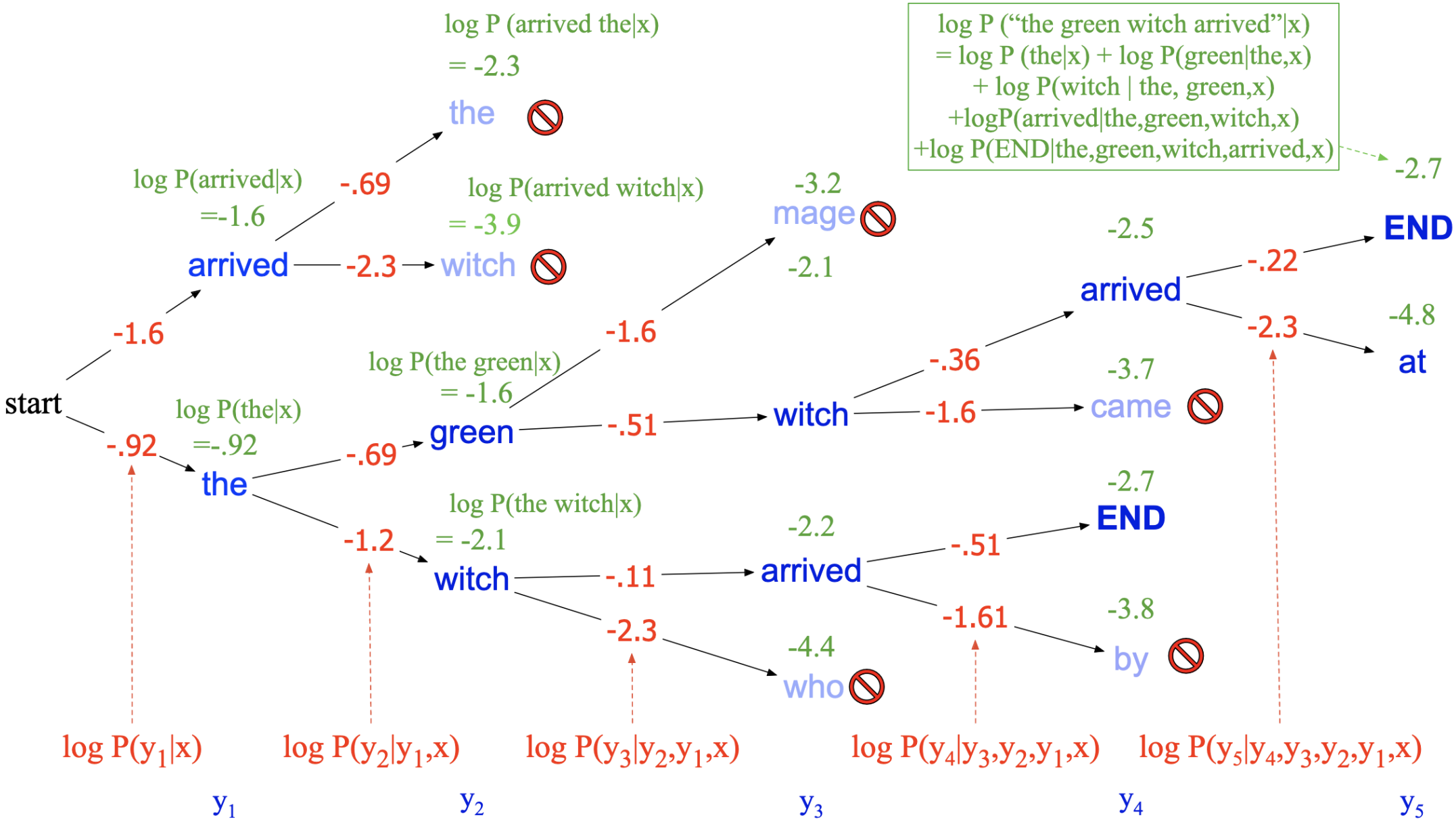
Beam Search



$$
\begin{aligned}
score(y) &= \log P(y|x) \\
&= \log\left(P(y_1|x)P(y_2|y_1,x)P(y_3|y_1,y_2,x)...P(y_t|y_1,...,y_{t-1},x)\right) \\
&= \sum_{i=1}^{t} \log P(y_i|y_1,...,y_{i-1},x)
\end{aligned}
$$

We simply need to add at each step the logprob of the prefix sequence so far

Beam Search


log P (arrived the|x)
= -2.3
the
log P(arrived|x)
=-1.6
-.69
log P(arrived witch|x)
= -3.9
arrived
-2.3
witch
-3.2
mage
-2.1
-1.6
-1.6
log P(the green|x)
= -1.6
start
log P(the|x)
=-.92
-.92
the
-.69
green
-.51
witch
-.36
-1.6
arrived
-2.5
-.22
END
-2.3
-4.8
at
came
-3.7
-1.2
log P(the witch|x)
= -2.1
witch
-.11
-2.3
arrived
-2.2
-.51
-2.7
END
-1.61
-3.8
by
-4.4
who
log P (“the green witch arrived”|x)
= log P (the|x) + log P(green|the,x)
+ log P(witch | the, green,x)
+logP(arrived|the,green,witch,x)
+log P(END|the,green,witch,arrived,x)
-2.7
log P(y1|x)
log P(y2|y1,x)
log P(y3|y2,y1,x)
log P(y4|y3,y2,y1,x)
log P(y5|y4,y3,y2,y1,x)
y1
y2
y3
y4
y5

Beam Search

The score has to be normalized for sequences with different length

- The additive nature of the score penalizes long sequences

$$score(y) = -\log P(y|x) = \frac{1}{T}\sum_{i=1}^{t} -\log P(y_i|y_1, \ldots, y_{i-1}, x)$$

Minimum Bayes Risk (MBR) decoding

We use explicitly an evaluation metric util to measure the goodness of fit of the translation *c* with some set $\mathcal{Y}$ of candidate translation:

$$\hat{y} = \underset{y \in \mathcal{Y}}{\operatorname{argmax}} \sum_{c \in \mathcal{Y}} \mathrm{util}(y, c)$$

- Metrics should be used to evaluate against perfect human translation
- We do not know it, and approximate the space of all possible translation with $\mathcal{Y}$

Machine Translation, RAG, and Evaluation

Retrieval Augmented Generation

Question Answering

Question Answering (QA) systems are designed to fill human information needs. Since a lot of information is present in text form they perform similarly to search engines.

Question answering systems often focus on *factoid questions*, questions of fact or reasoning that can be answered with simple facts expressed in short or medium-length texts:

- Where is the Louvre Museum located?
- Where does the energy in a nuclear explosion come from?
- How to get a script l in latex?

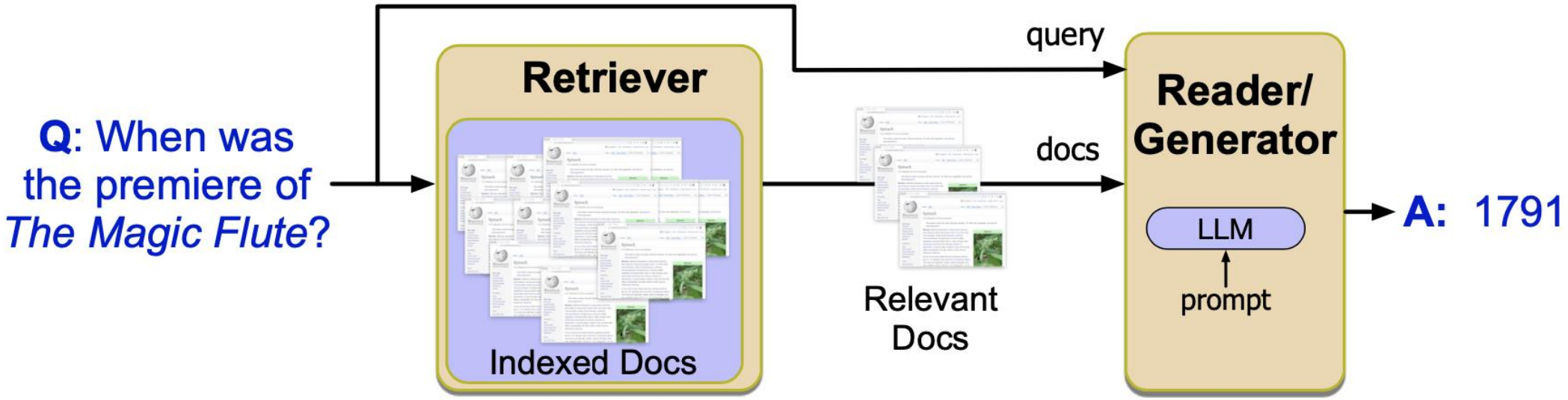
Question Answering

LLMs can be prompted for QA:

Q: Where is the Louvre Museum located? A:

But LLMs can hallucinate and they can be not aligned

Retrieval Augmented Generation (RAG)



How to make the Retriever?

Vector database, Knowledge base, hybrid …

Machine Translation, RAG, and Evaluation

Evaluation

Evaluation

Evaluation is needed in many generative tasks to assess the goodness of the sentence provided by the model.

A corpus (at least one) of reference sentences can be either at disposal or not.

Different approaches in the two cases.

Evaluation by human raters

In general, no reference corpus is available in this case.

Two properties to be assessed:

- *Adequacy* or *Faithfulness* (the meaning of the text)
- *Fluency* (the grammatical structure)

Human raters (in general crowdworkers) use either scales to score both properties or they can be provided with multiple translations that have to be ranked.

Autmatic evaluation

Automatic evaluation compares a reference sentence *r* and a hypothesis *h* using either character overlap or embeddings similarity

These metrics are used in all the generative tasks where a reference sentence is available

character F-score (chrF)

Averages the percentage of common 1-grams, 2-grams, …, *k*-grams in *r* and in *h*.

chrP percentage of character 1-grams, 2-grams, ..., k-grams in the hypothesis that occur in the reference, averaged.

chrR percentage of character 1-grams, 2-grams,..., k-grams in the reference that occur in the hypothesis, averaged.

$$\text{chrF}\beta = (1+\beta^2)\frac{\text{chrP}\cdot\text{chrR}}{\beta^2\cdot\text{chrP}+\text{chrR}}$$

BiLingual Evaluation Understudy (BLEU)

Given a corpus of candidate and reference sentences $H = \{h_i\}$ and $R = \{r_i\}$ BLEU is a function of the n-gram word precision $p_n(H,R)$over all the reference sentences combined with a brevity penalty BP computed over the corpus as a whole.

$$p_n(H,R) = \frac{\sum_i \sum_{s \in D_n^{(h_i)}} \min(C(s,h_i), \max_{r_j} C(s,r_j))}{\sum_{s \in D_n^{(h_i)}} C(s,h_i)}$$

$$\mathrm{BLEU}(H,R) = \mathrm{BP} \cdot \exp\left(\frac{1}{N}\sum_{n=1}^{N} \log p_n(H,R)\right)$$

Recall-Oriented Understudy for Gisting Evaluation (ROUGE)

A set of metrics:

- ROUGE-N: N-gram overlap (typically ROUGE-1 and ROUGE-2) computed in terms of
- Precision: # of common N-grams/# of N-grams in *h*
- Recall: # of common N-grams/# of N-grams in *r*
- F1 = 2 P*R /(P+R)

Recall-Oriented Understudy for Gisting Evaluation (ROUGE)

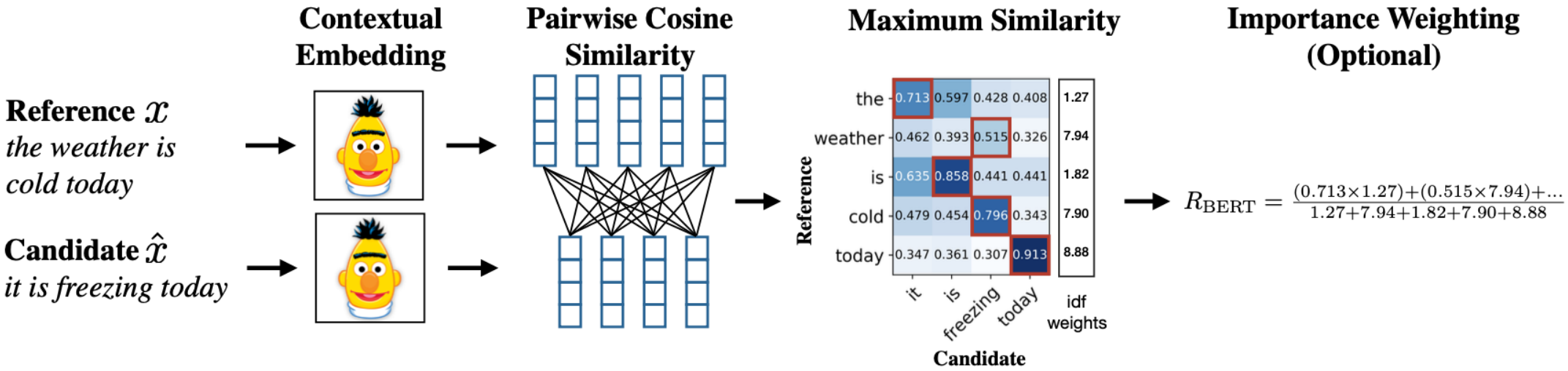
- ROUGE-L: Longest Common Subsequence (*LCS*) between *h* and *r*
- Precision: *LCS*(*h*, *r*)/ # of 1-grams in *h*
- Recall: *LCS*(*h*, *r*) / # of 1-grams in *r*
- F1 = 2 P*R /(P+R)

BERT score

N-grams overlap metrics do not perform well in case of synonims or paraphrases.

BERT score intuition: *compare token embeddings instead!!*

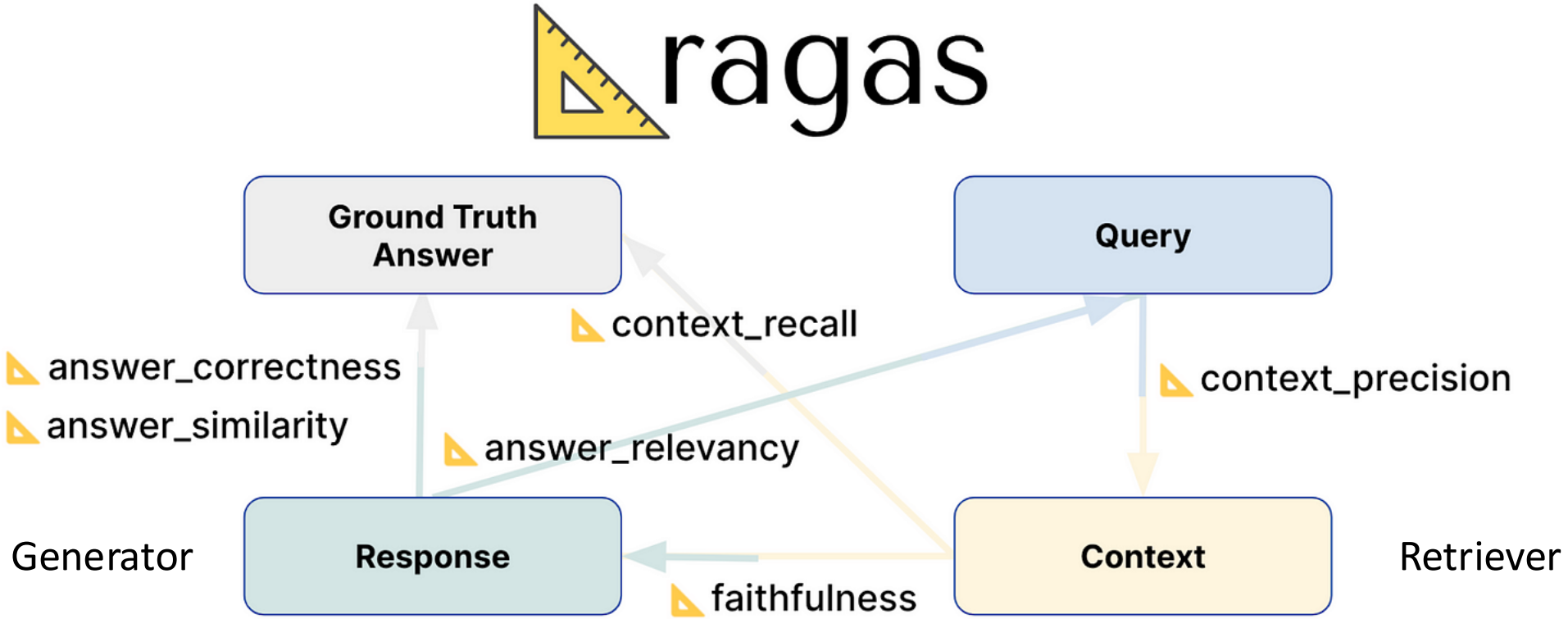
BERT score



$$R_{\text{BERT}} = \frac{1}{|x|}\sum_{x_i \in x} \max_{\tilde{x}_j \in \tilde{x}} x_i \cdot \tilde{x}_j \qquad P_{\text{BERT}} = \frac{1}{|\tilde{x}|}\sum_{\tilde{x}_j \in \tilde{x}} \max_{x_i \in x} x_i \cdot \tilde{x}_j$$

RAGAS

Framework designed to evaluate answers in RAG systems



https://bit.ly/49yRfoR

RAGAS

Faithfulness

assesses the factual accuracy of the generated answers by checking if the statements made in the answers are supported by the provided context. It is computed by analyzing the validity of each statement derived from the generated answer against the context.

Faithfulness = # of valid statements / # of statements

https://bit.ly/49yRfoR

RAGAS

Answer Relevancy

measures how relevant and directly related the generated answer is to the posed question. It evaluates the similarity between probable questions that could elicit the same answer and the actual question asked.

Answer relevancy = mean cosine similarity between the user input a number of answers reverse engineered from the response

https://bit.ly/49yRfoR

RAGAS

Context Recall

measures how many relevant documents or pieces of information (the Ground Truth – GT) were successfully retrieved. It focuses on not missing important results.

Context recall = |GT claims attributed to the context| / |GT claims|

https://bit.ly/49yRfoR

RAGAS

Context Precision

Context Precision is a metric that measures the proportion of relevant chunks in the retrieved contexts. It is calculated as the mean of the precision@k for each chunk in the context.

Precision@k is the ratio of the number of relevant chunks at rank *k* to the total number of chunks at rank *k* (that is in the top *k* results)

https://bit.ly/49yRfoR

RAGAS

Context Precision

$$\text{Context Precision@K} = \frac{\sum_{k=1}^{K} (\text{Precision@k} \times v_k)}{\text{Total number of relevant items in the top } K \text{ results}}$$

$$\text{Precision@k} = \frac{\text{true positives@k}}{(\text{true positives@k} + \text{false positives@k})}$$

v_k is the relevance indicator at rank k

https://bit.ly/49yRfoR