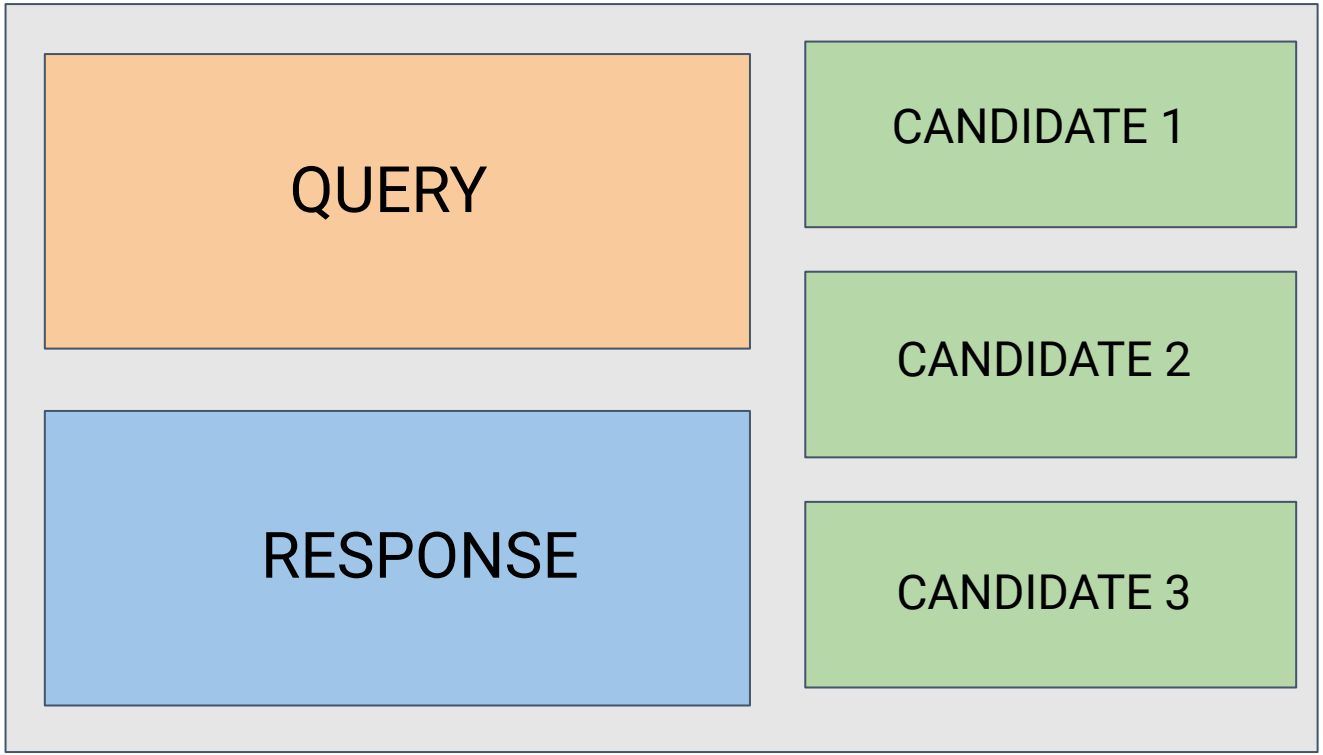
“ CONVERSATIONS

Just AI

Создание бота-суфлера на трансформерах: почти терминатор, но нет

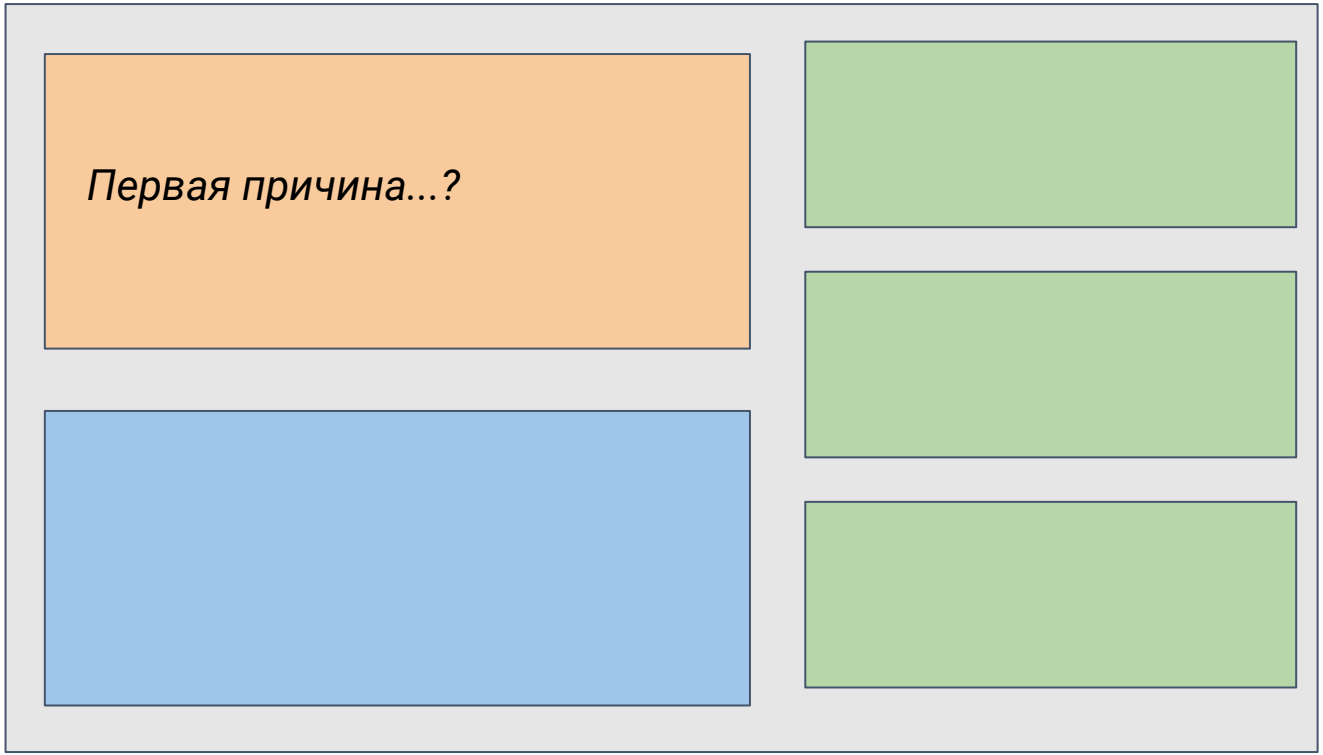
Darya Serdyuk, Just AI
RND team

What is prompter bot



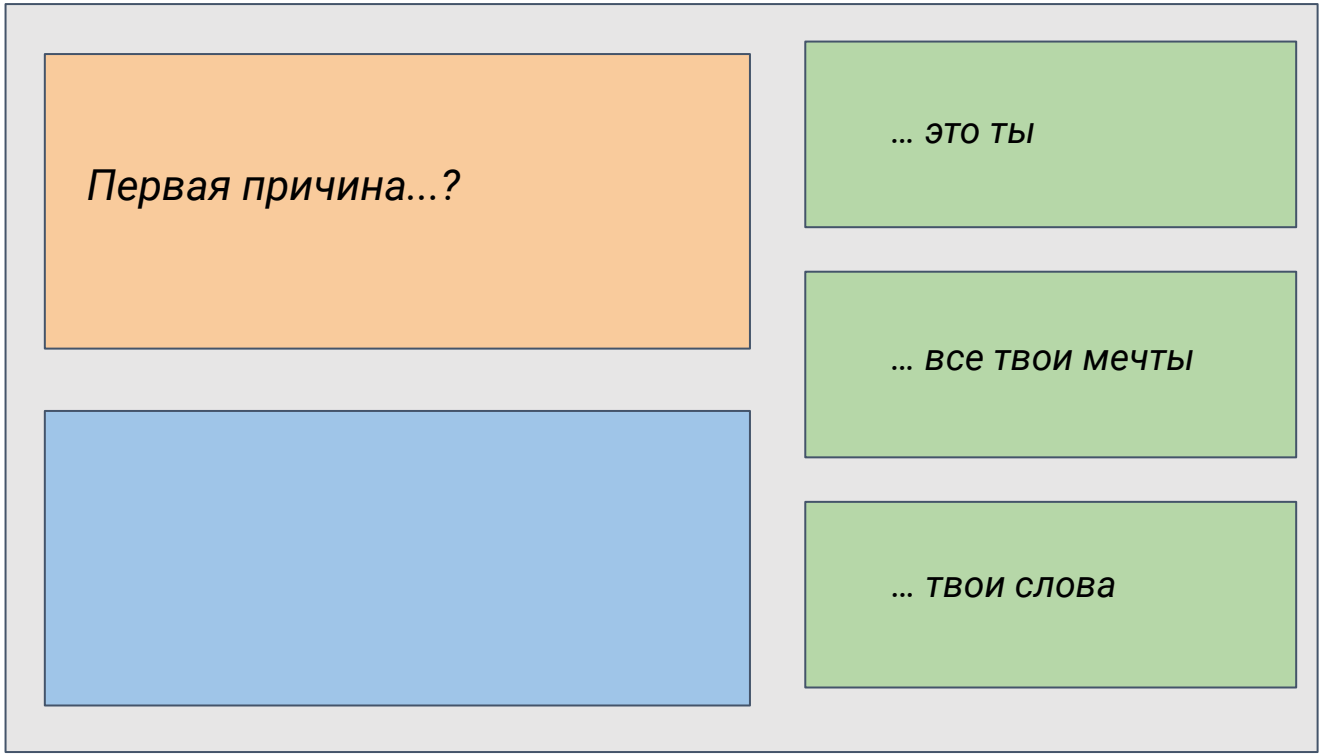
What is prompter bot

1



What is prompter bot

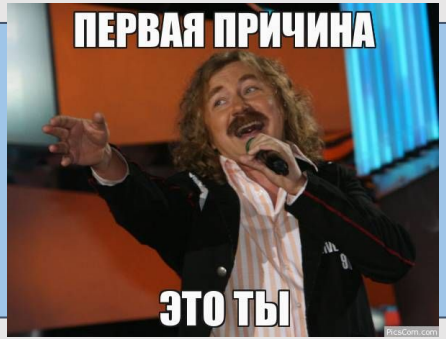
2



What is prompter bot

3

Первая причина...?



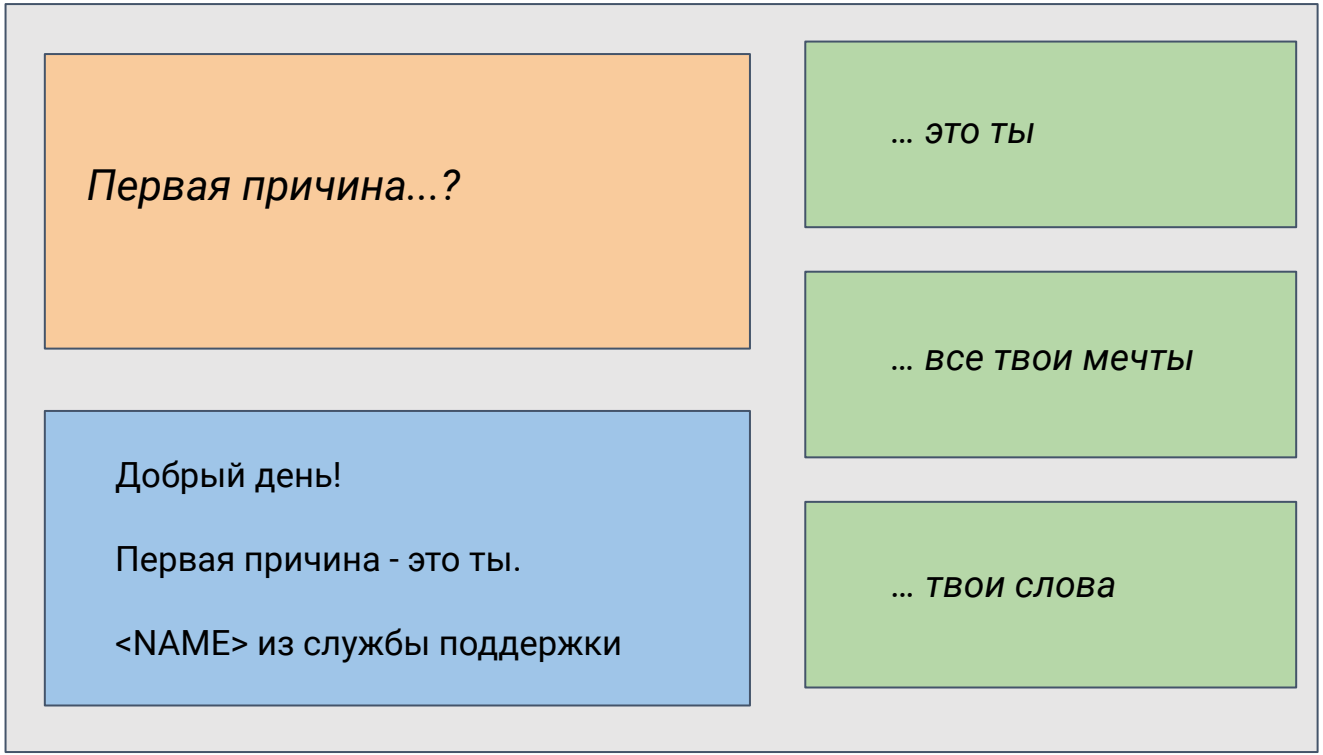
... это ты

... все твои мечты

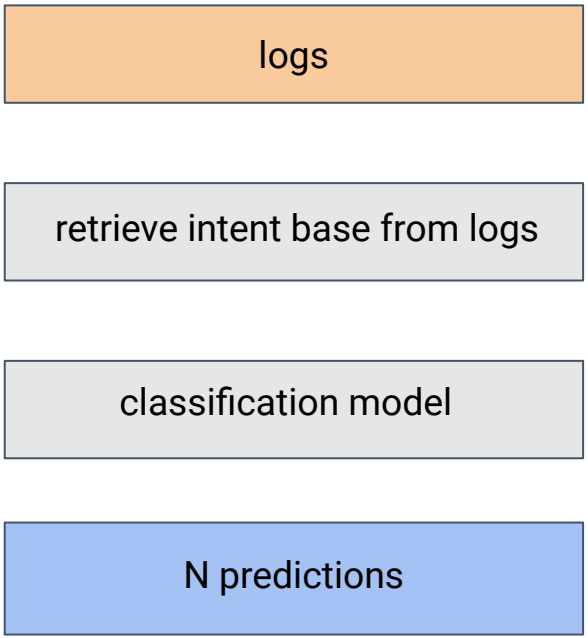
... твои слова

What is prompter bot

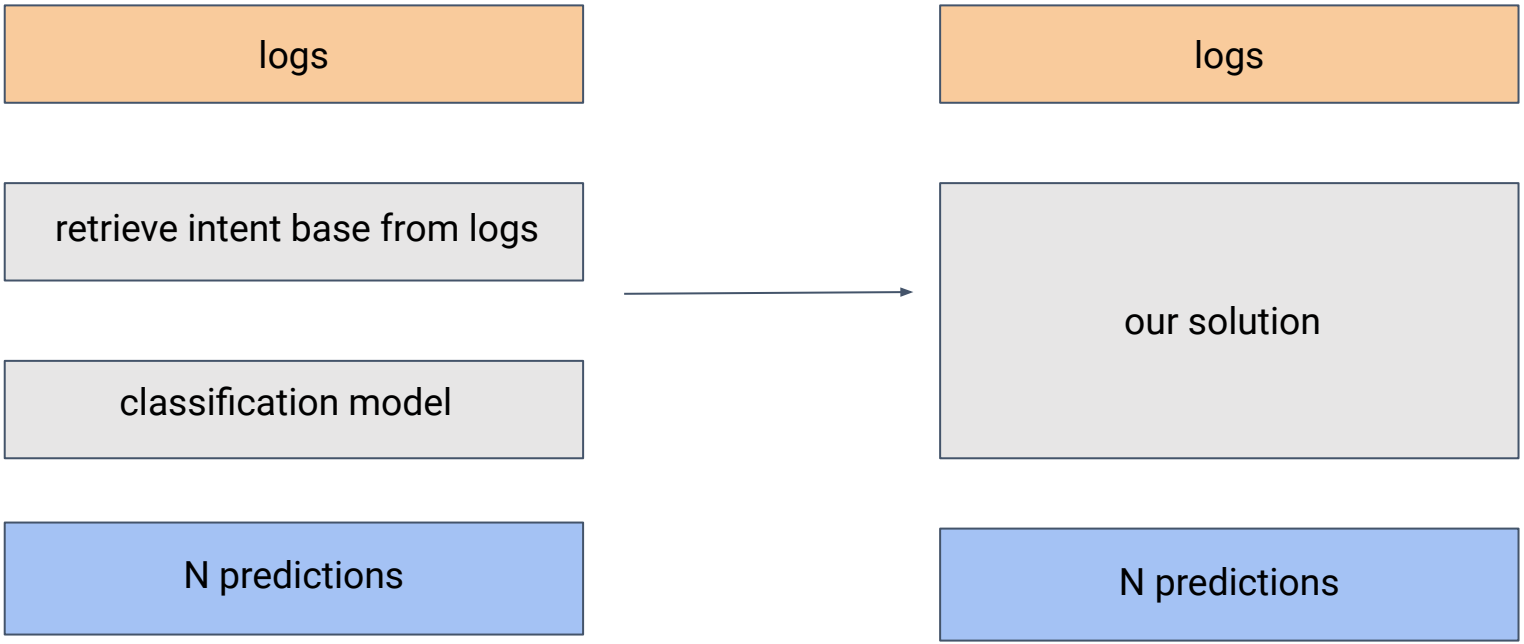
3



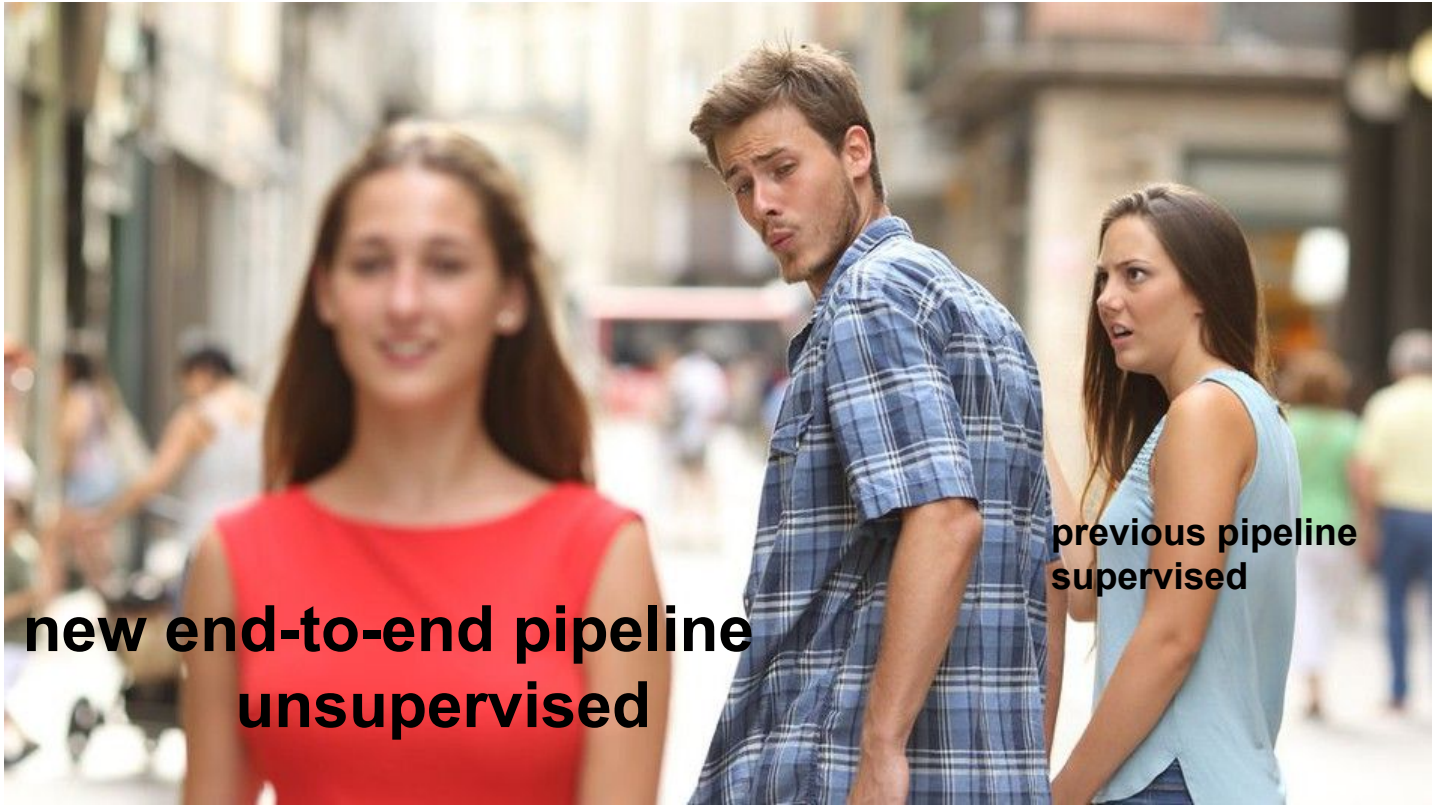
Motivation


logs
retrieve intent base from logs
classification model
N predictions

Motivation


logs
retrieve intent base from logs
classification model
N predictions
logs
our solution
N predictions

Motivation


new end-to-end pipeline
unsupervised
previous pipeline
supervised

Goal

- the most appropriate variants for response

Goal

- the most appropriate variants for response
- no manual log parsing

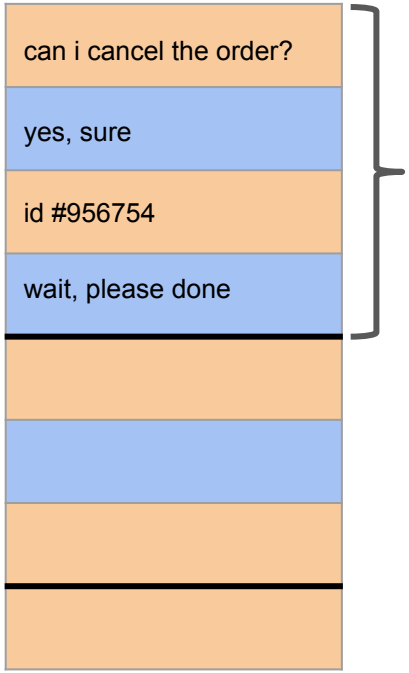
Goal

- the most appropriate variants for response
- no manual log parsing
- end-to-end solution

Goal

- the most appropriate variants for response
- no manual log parsing
- end-to-end solution
- zero-shot learning (add new responses without model retraining)

Input Data

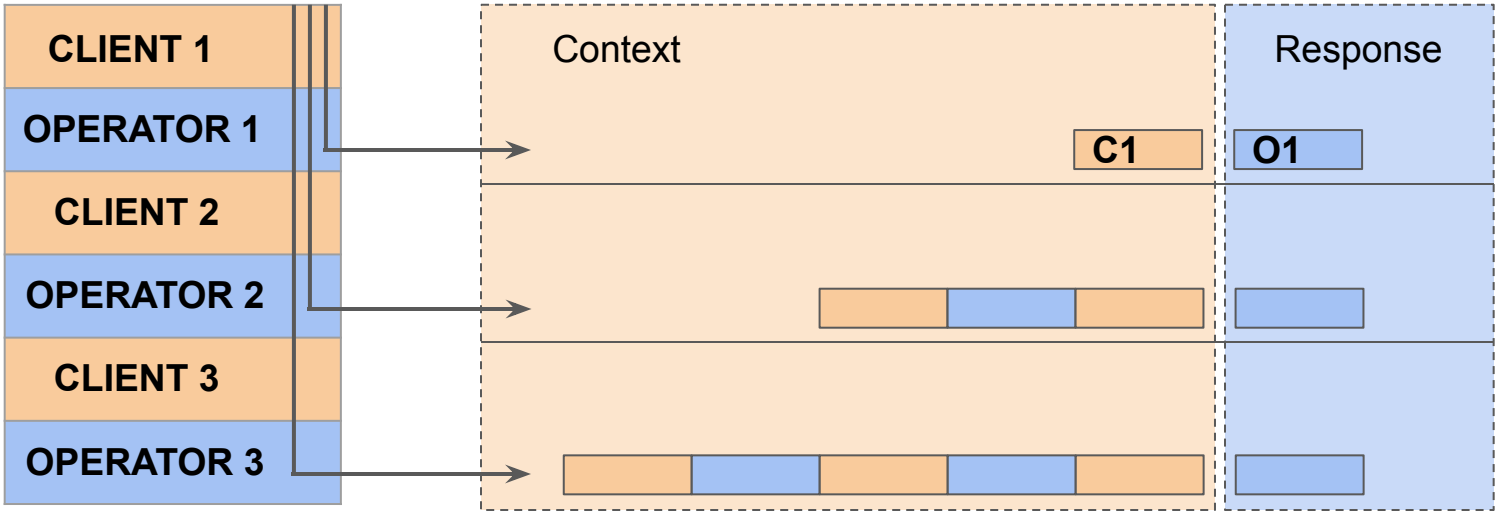


CLIENT-OPERATOR
Dialog session

All Dialog Sessions

	session_id	text	is_user
0	001	can i cancel the order?	1
1	001	yes, sure	0
2	001	id #956754	1
3	001	wait, please	0
4	001	done	0
5	002	hello	1
6	002	good evening, can i help you?	0

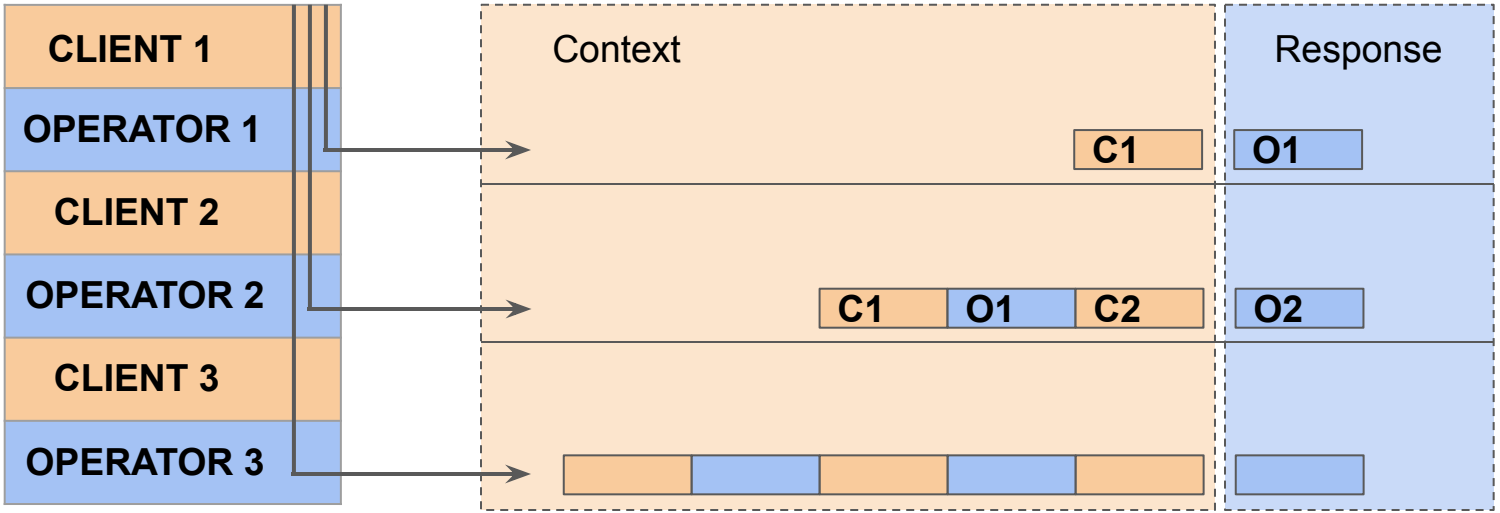
Data Preprocessing



One Dialog Session

Split each Dialog Session on **CONTEXT**-**RESPONSE** samples

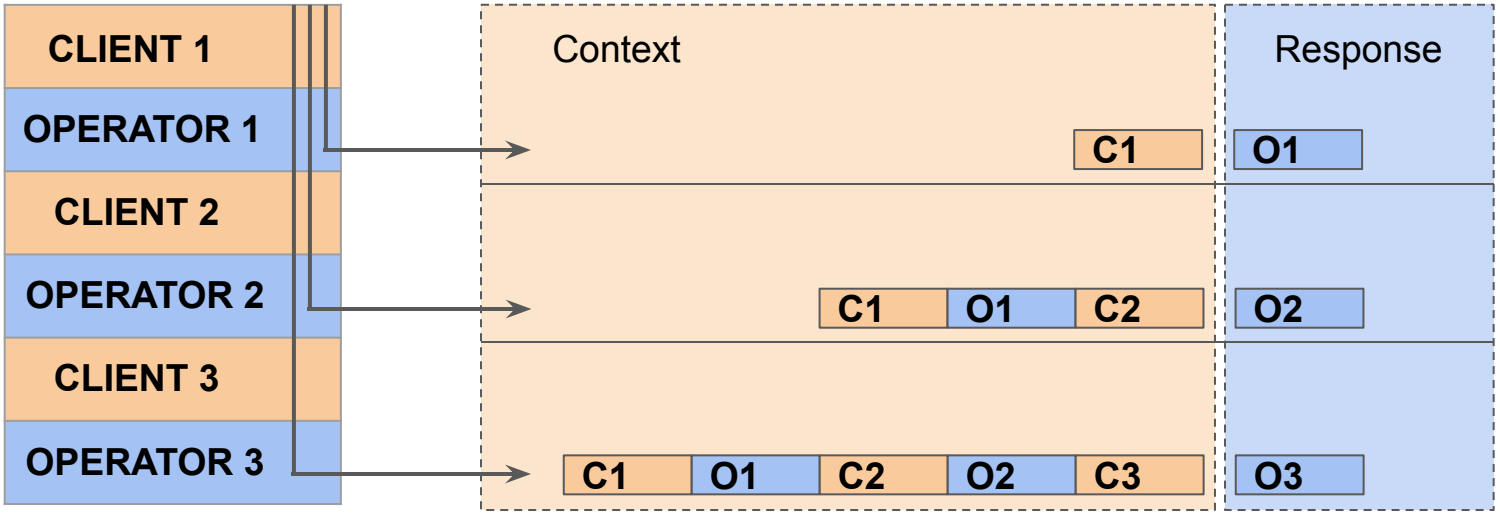
Data Preprocessing



One Dialog Session

Split each Dialog Session on **CONTEXT**-**RESPONSE** samples

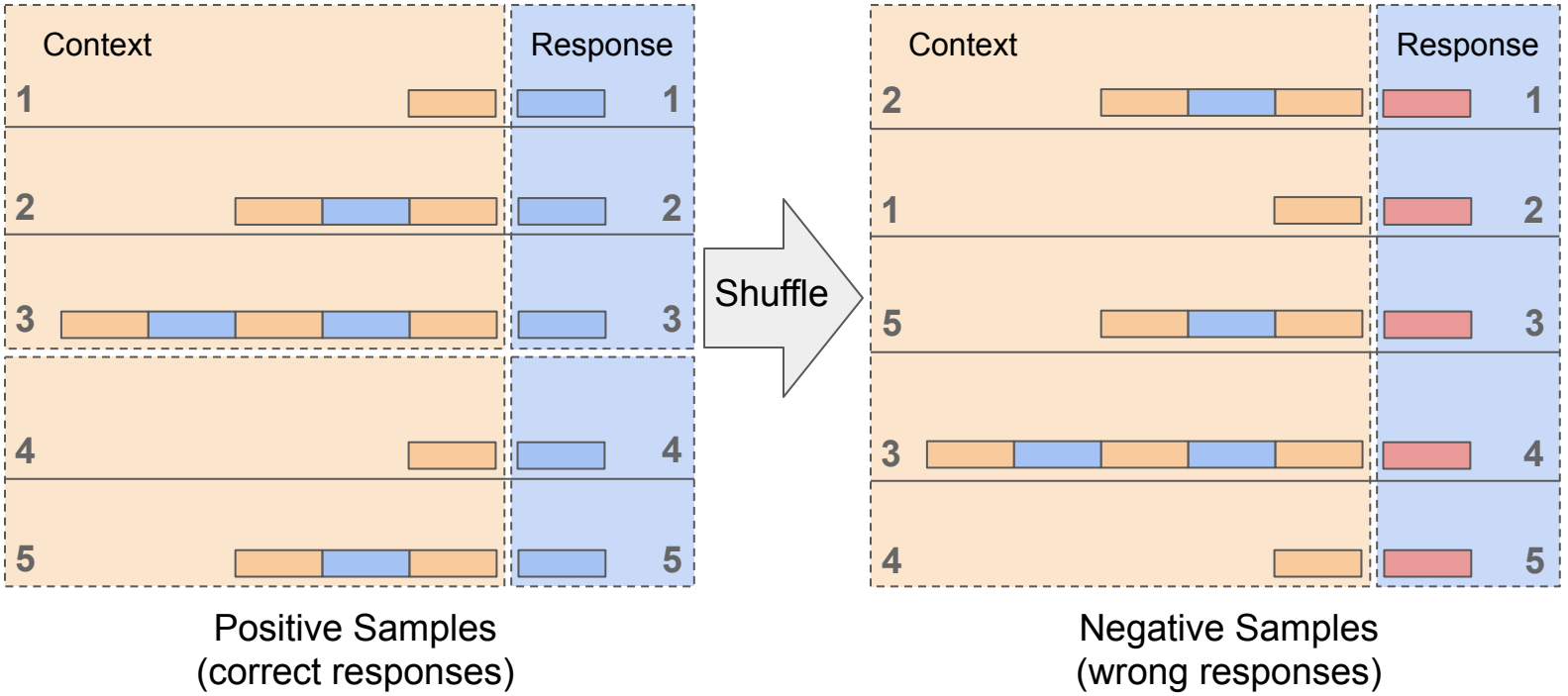
Data Preprocessing



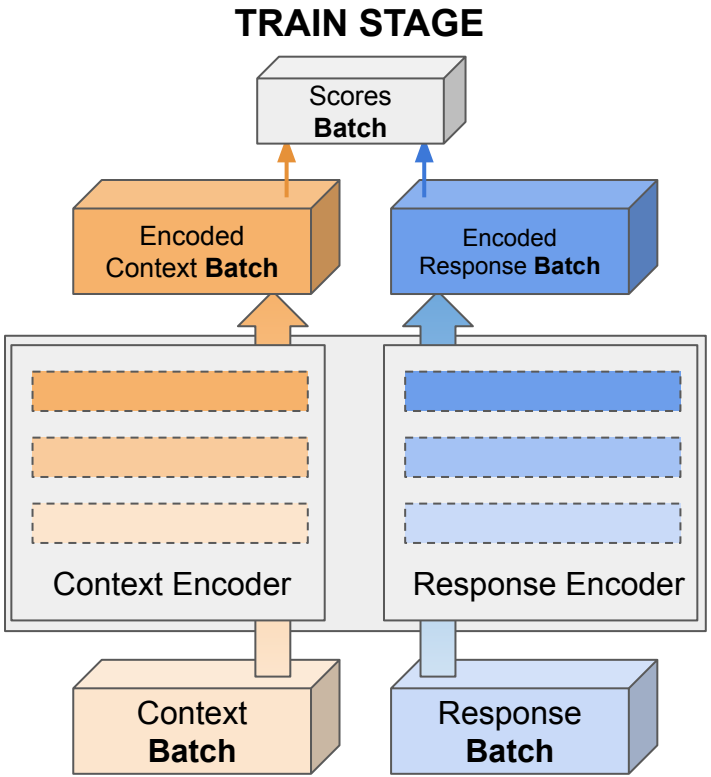
One Dialog Session

Split each Dialog Session on **CONTEXT**-**RESPONSE** samples

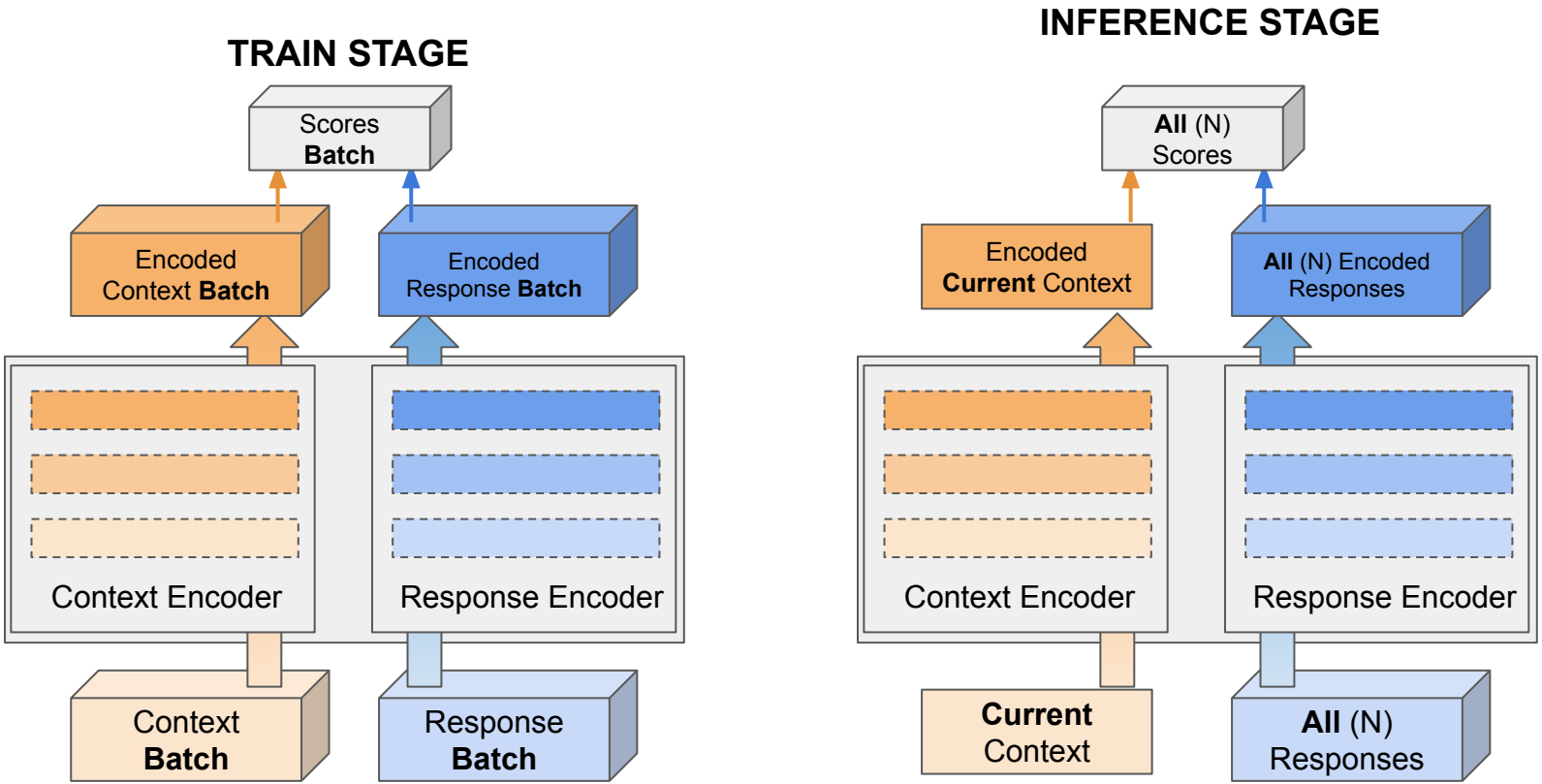
Negative Sampling



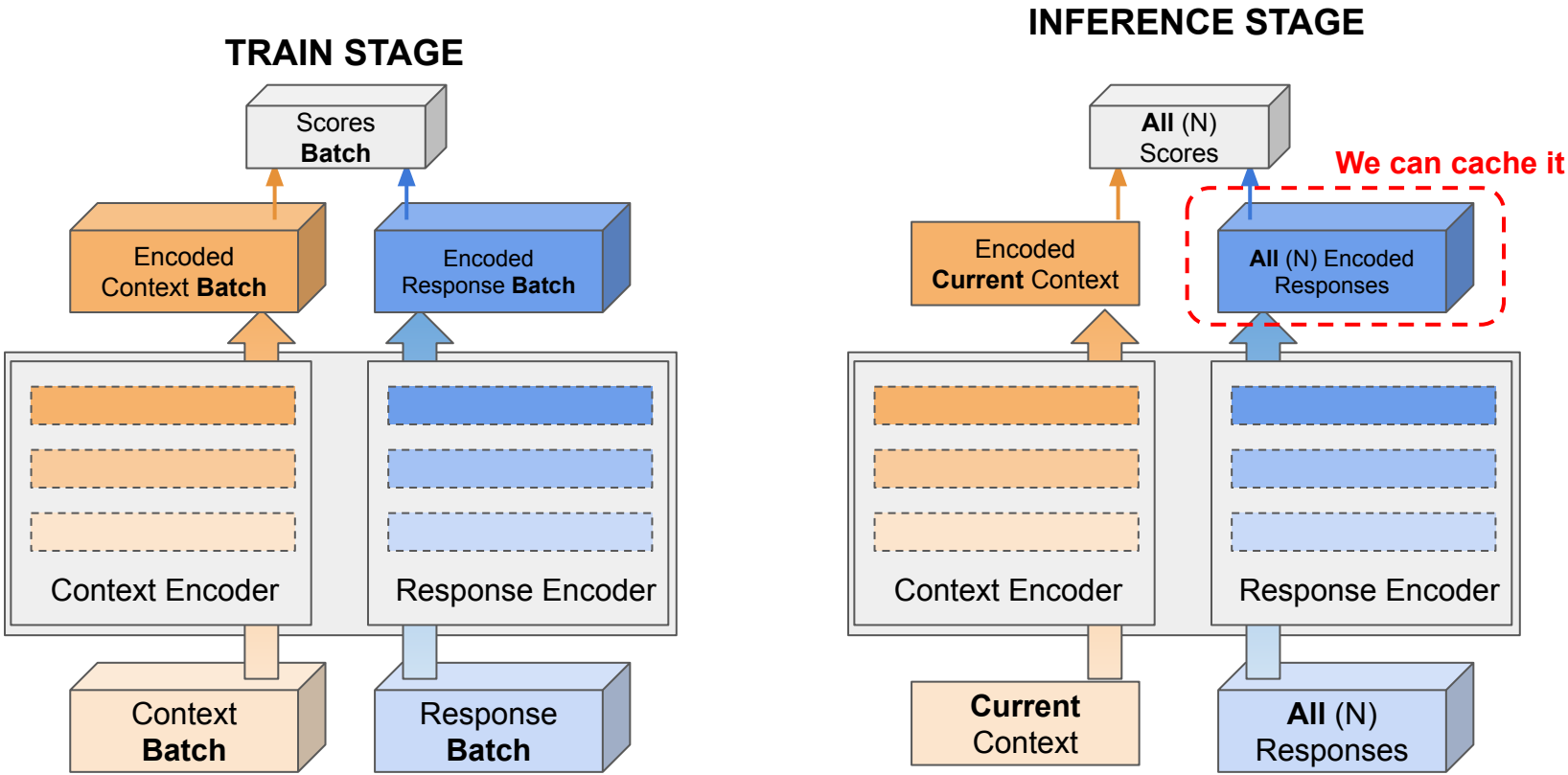
Model Architecture: Bi-Encoder



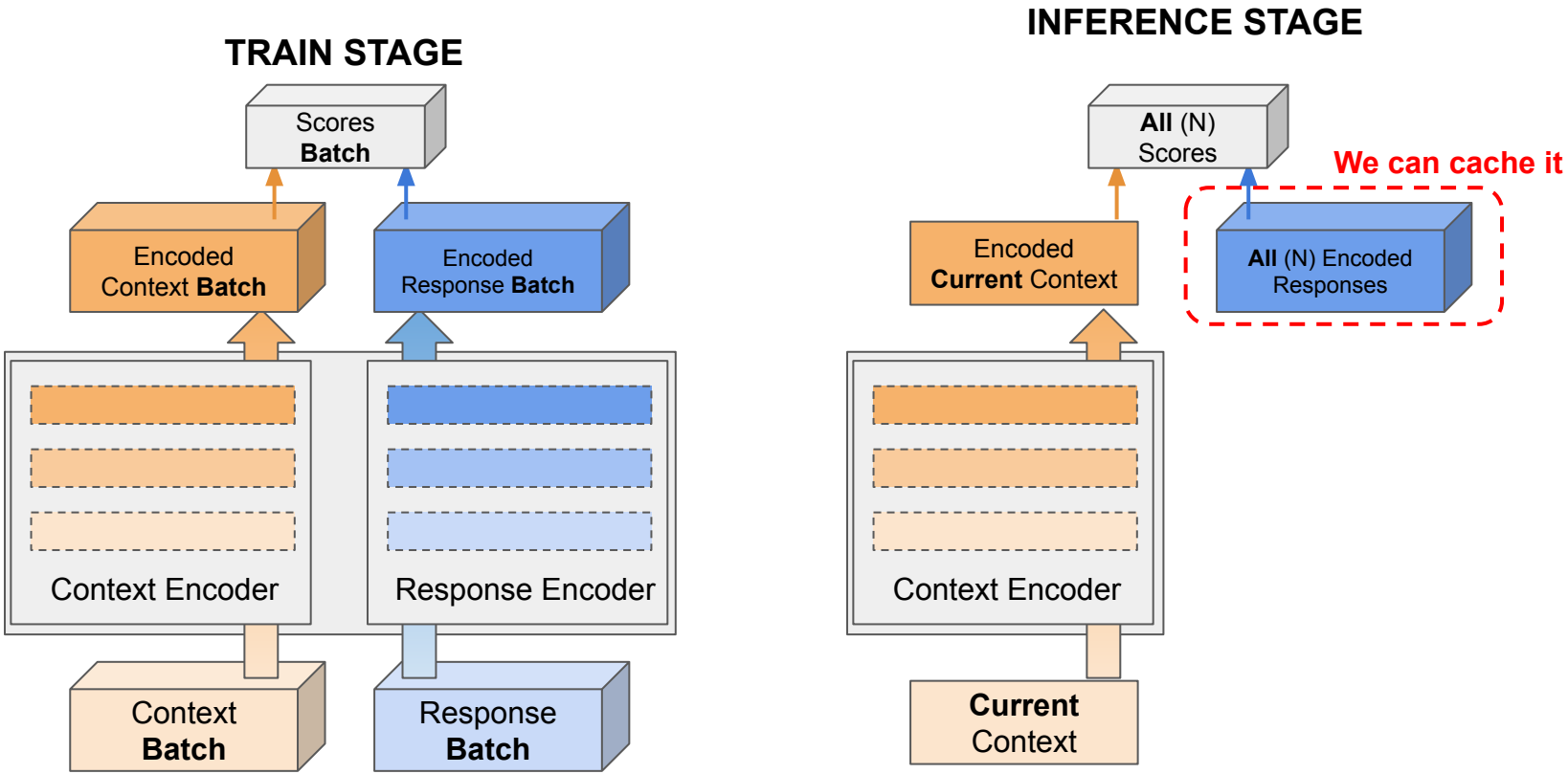
Model Architecture: Bi-Encoder



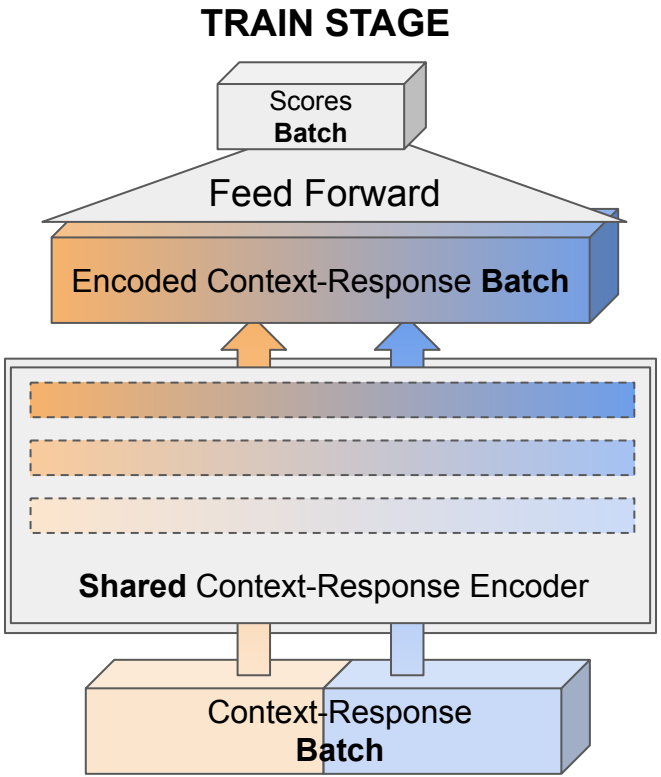
Model Architecture: Bi-Encoder



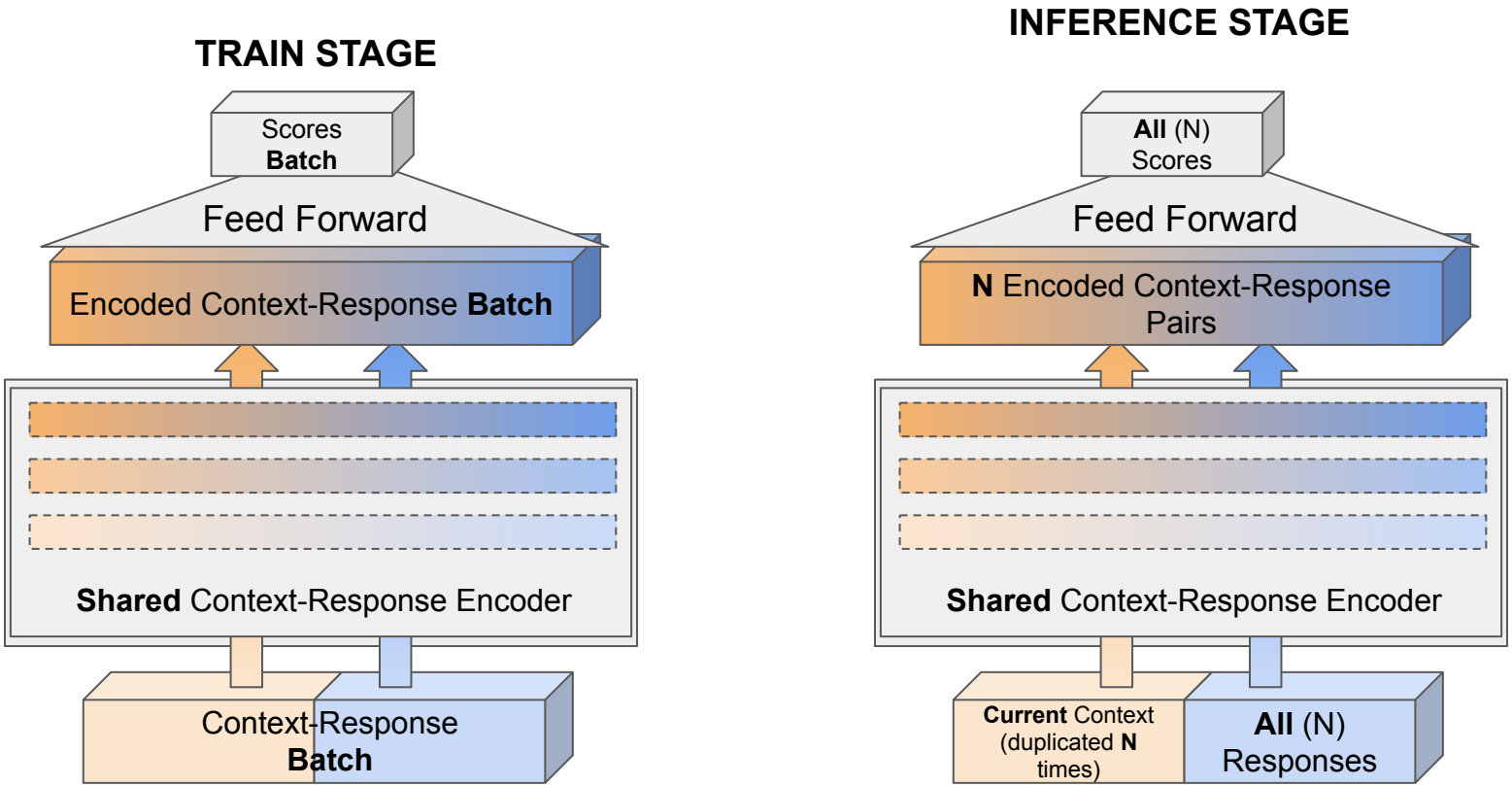
Model Architecture: Bi-Encoder



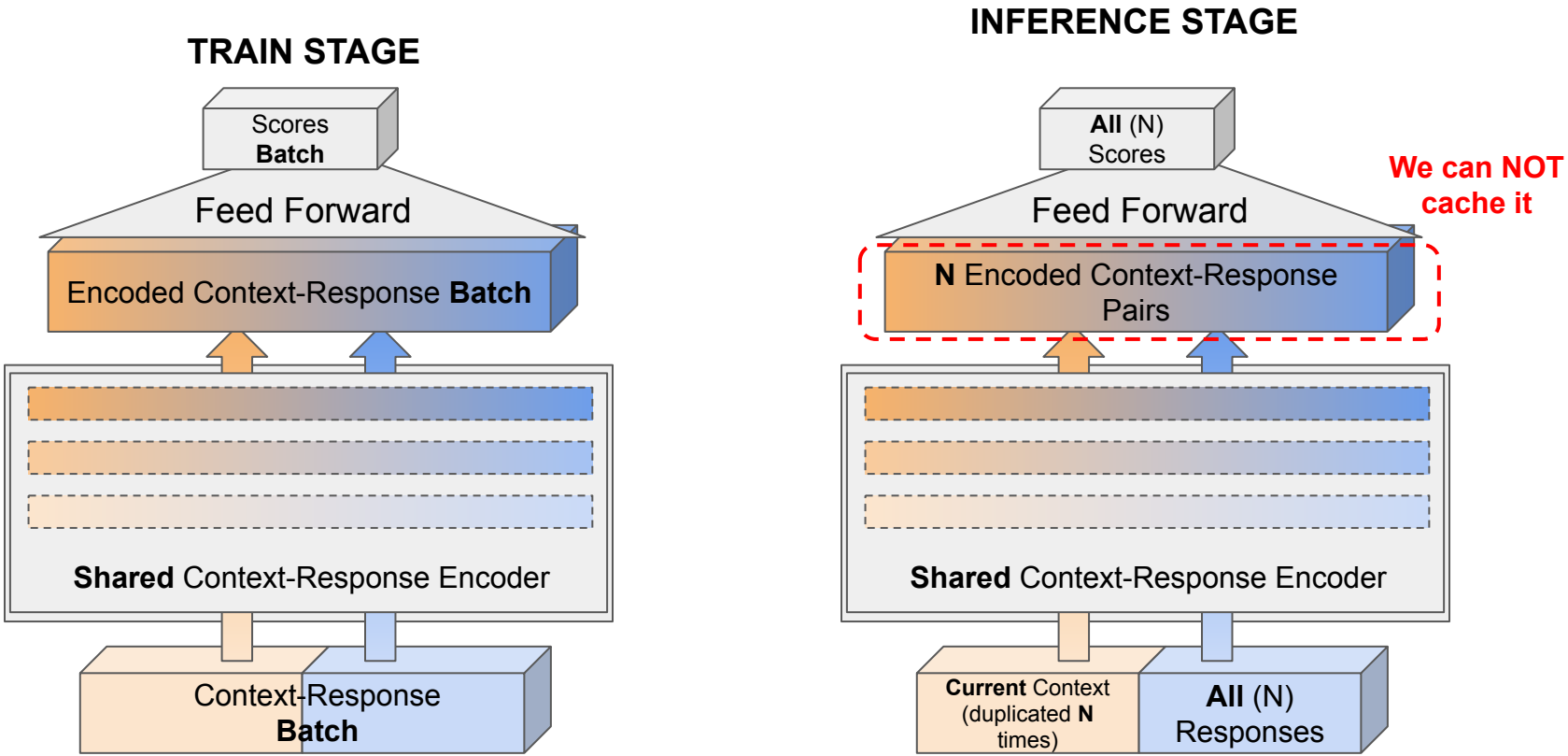
Model Architecture: Cross-Encoder



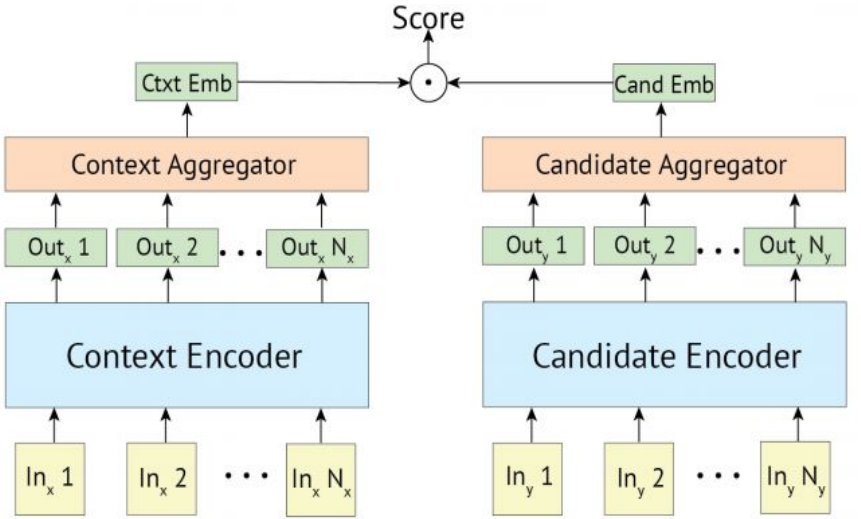
Model Architecture: Cross-Encoder



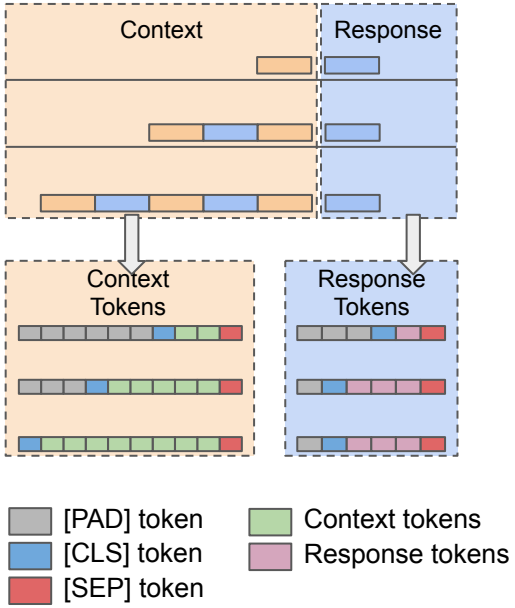
Model Architecture: Cross-Encoder



BERT-Based Bi-Encoder



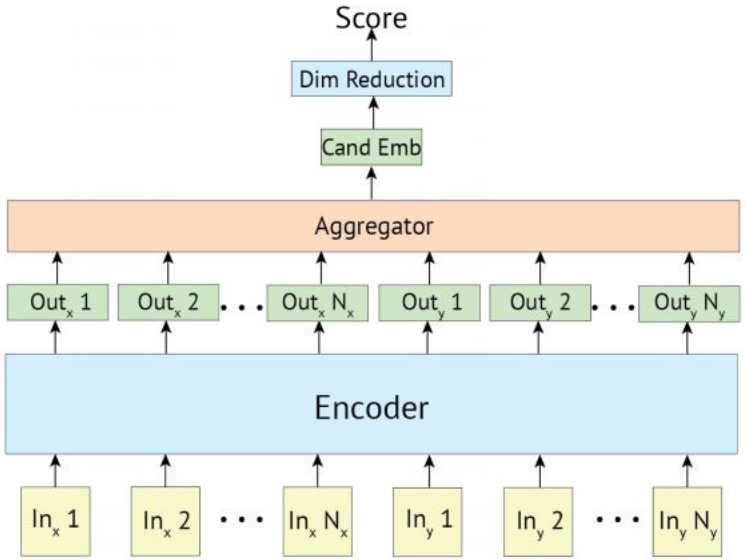
BERT Bi-Encoder Model



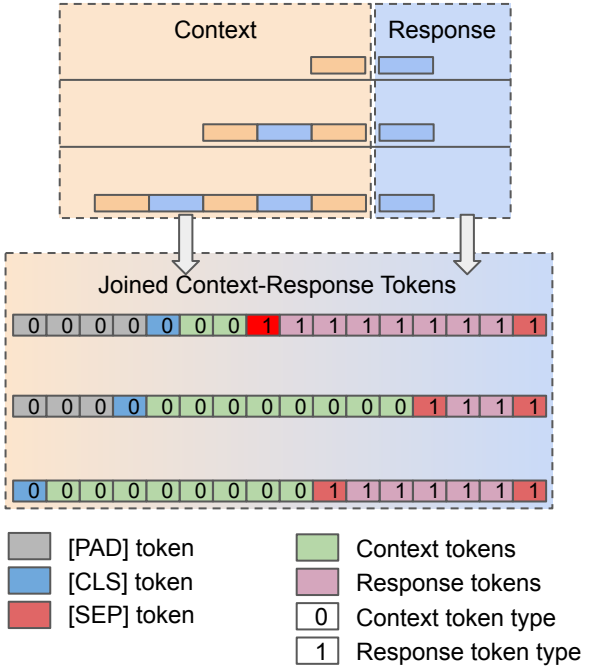
BERT Bi-Encoder Tokenization

Poly-encoders: Transformer Architectures and Pre-training Strategies for Fast and Accurate Multi-sentence Scoring https://arxiv.org/pdf/1905.01969.pdf

BERT-Based Cross-Encoder



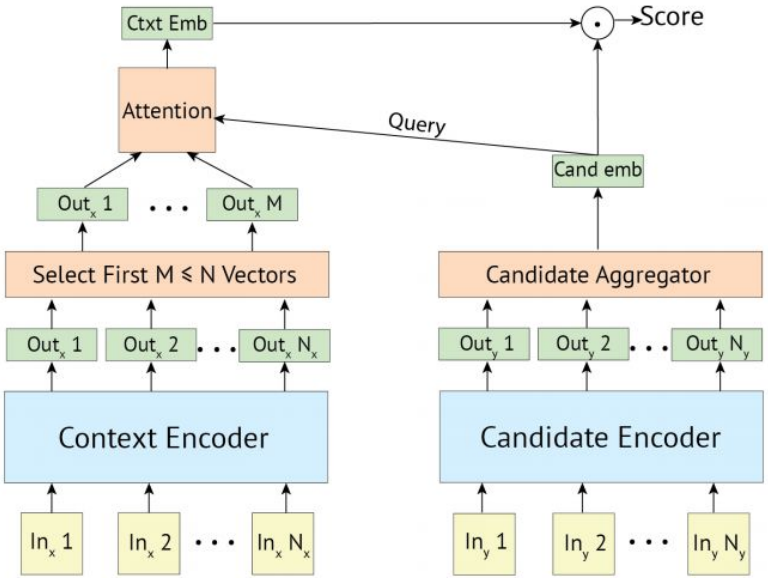
BERT Cross-Encoder Model



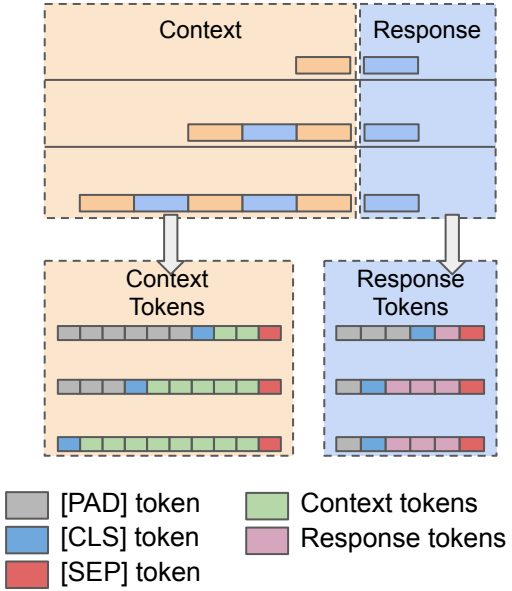
BERT Cross-Encoder Tokenization

Poly-encoders: Transformer Architectures and Pre-training Strategies for Fast and Accurate Multi-sentence Scoring https://arxiv.org/pdf/1905.01969.pdf

BERT-Based Poly-Encoder



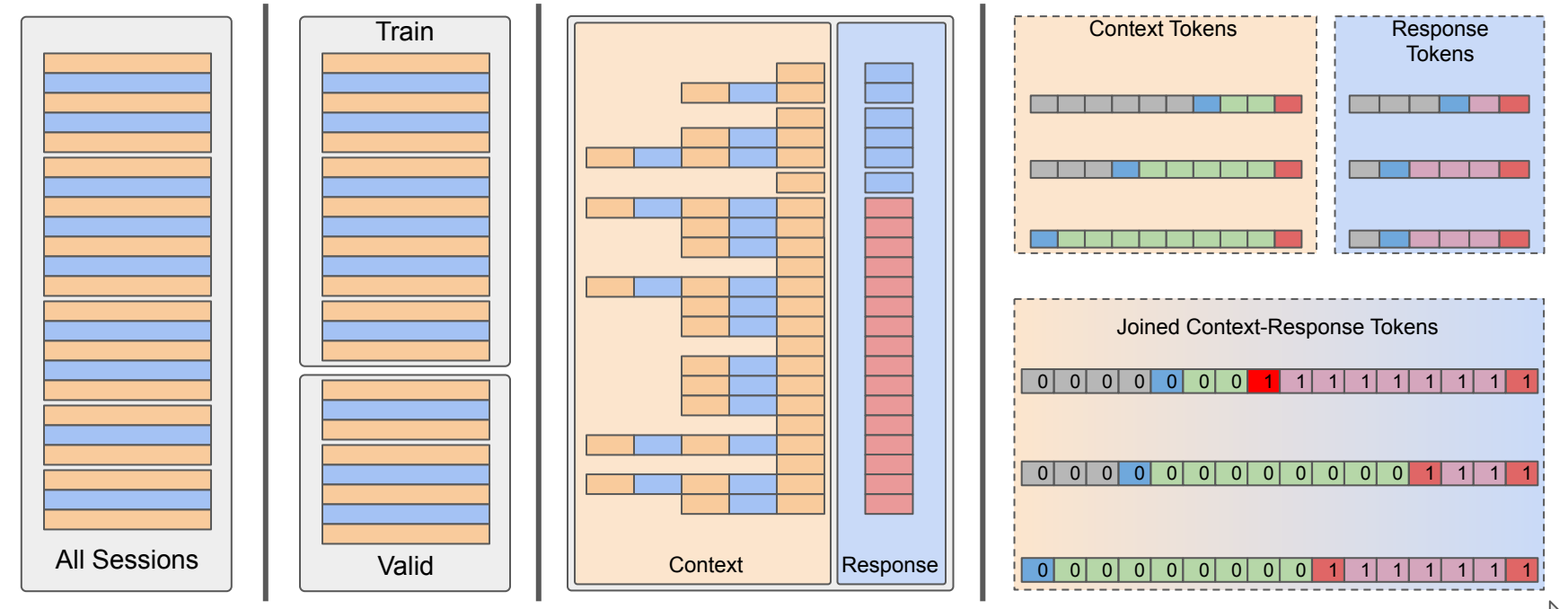
BERT Poly-Encoder Model



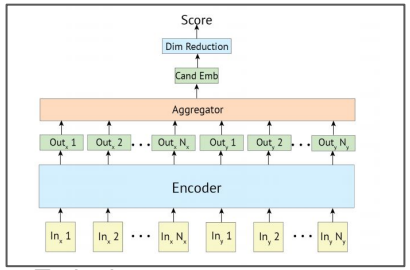
BERT Poly-Encoder Tokenization

Poly-encoders: Transformer Architectures and Pre-training Strategies for Fast and Accurate Multi-sentence Scoring https://arxiv.org/pdf/1905.01969.pdf

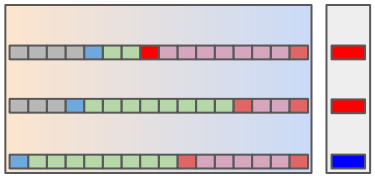
Data Preparation Pipeline



Final Modelling Pipeline

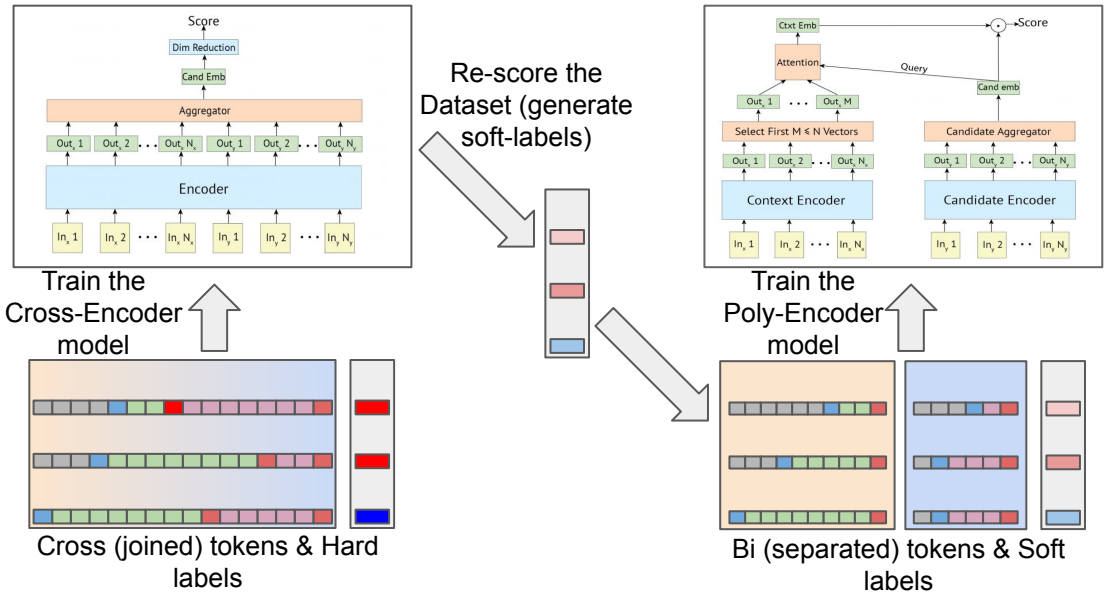


Train the Cross-Encoder model



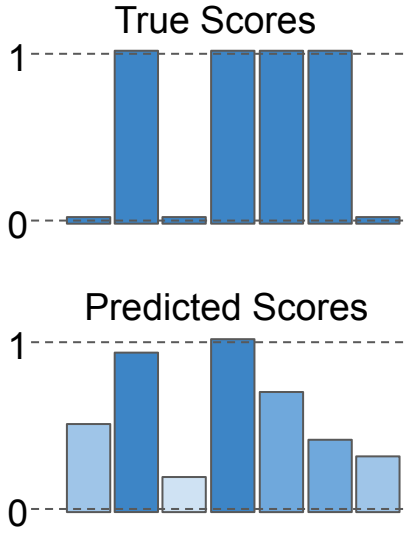
Cross (joined) tokens & Hard labels

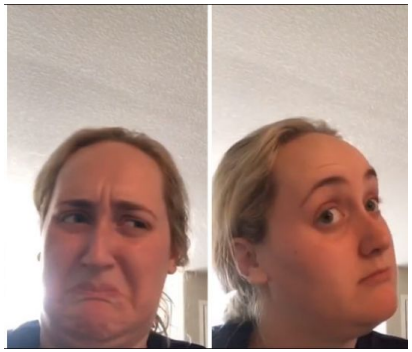
Final Modelling Pipeline



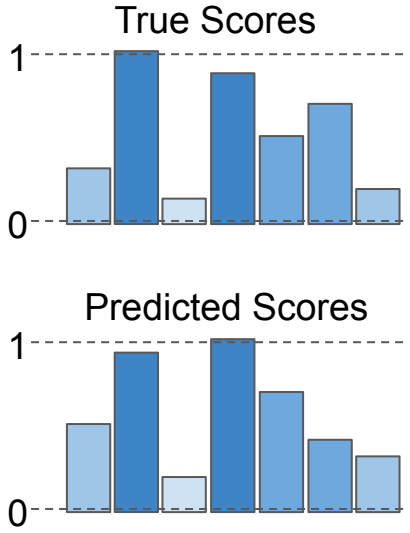
Hard Labels VS Soft Labels

Hard Labels





Soft Labels



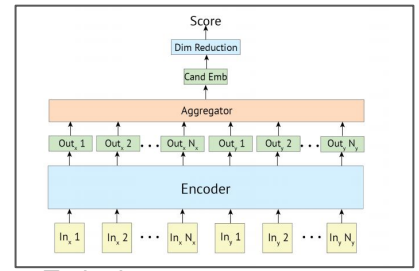
BCE-Loss

$$J(\mathbf{w}) = -\frac{1}{N}\sum_{n=1}^{N}\left[y_n \log \hat{y}_n + (1-y_n)\log(1-\hat{y}_n)\right]$$

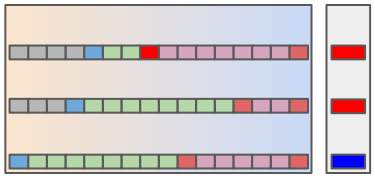
KLD-Loss (for Bernoulli distr.)

$$J(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N}\left[y_n \log(\frac{y_n}{\widehat{y_n}}) + (1-y_n)\log(\frac{1-y_n}{1-\widehat{y_n}})\right]$$

Final Modelling Pipeline

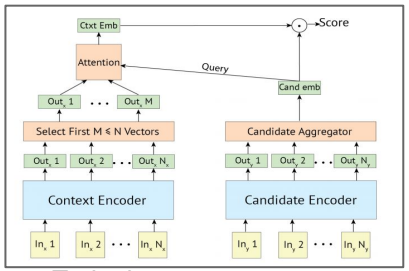


Train the Cross-Encoder model

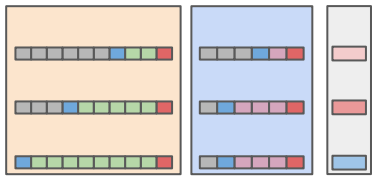


Cross (joined) tokens & Hard labels

Re-score the Dataset (generate soft-labels)

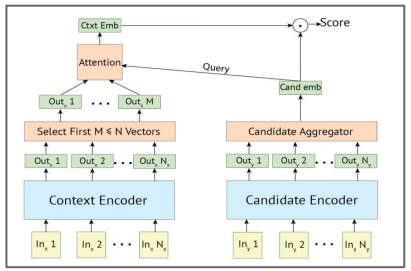


Train the Poly-Encoder model

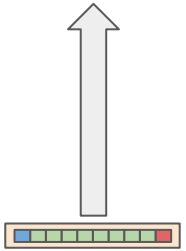


Bi (separated) tokens & Soft labels

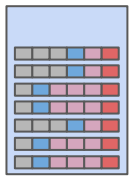
Use this model as a final one



Cache it in off-line

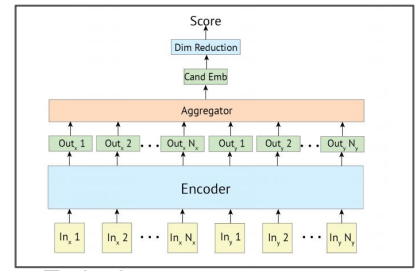


Current context

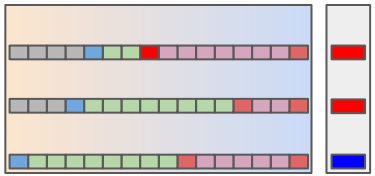


All available responses

Final Modelling Pipeline

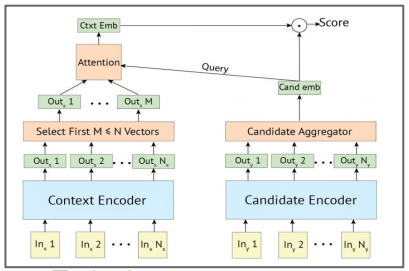


Train the Cross-Encoder model

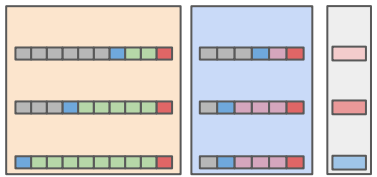


Cross (joined) tokens & Hard labels

Re-score the Dataset (generate soft-labels)

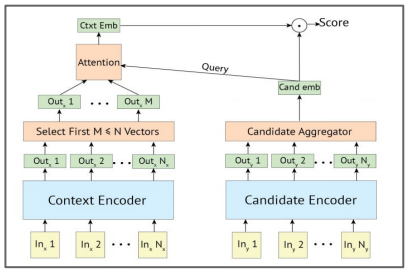


Train the Poly-Encoder model



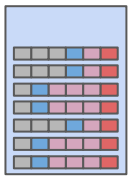
Bi (separated) tokens & Soft labels

Use this model as a final one

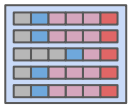


Current context

Cache it in off-line



All available responses



+ Unseen earlier responses

Results

Experiment	ROC-AUC	PR-AUC	MRR*
Bi-Encoder	0.923	0.561	0.032
Poly-Encoder	0.935	0.606	0.046
Cross-Encoder	0.943	0.650	-
Poly-Encoder + Soft Labels	0.942	0.631	0.079

* MRR was calculated on the holdout dataset with 5k samples

$$\mathrm{MRR} = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{\mathrm{rank}_i}.$$

Results

Experiment	ROC-AUC	PR-AUC	MRR*
Bi-Encoder	0.923	0.561	0.032
Poly-Encoder	0.935	0.606	0.046
Cross-Encoder	0.943	0.650	-
Poly-Encoder + Soft Labels	0.942	0.631	0.079

The best one, but very slow

* MRR was calculated on the holdout dataset with 5k samples

$$\mathrm{MRR} = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{\mathrm{rank}_i}.$$

Results

Experiment	ROC-AUC	PR-AUC	MRR*
Bi-Encoder	0.923	0.561	0.032
Poly-Encoder	0.935	0.606	0.046
Cross-Encoder	0.943	0.650	-
Poly-Encoder + Soft Labels	0.942	0.631	0.079

The best one, but very slow

The good one, and very fast

* MRR was calculated on the holdout dataset with 5k samples

$$\mathrm{MRR} = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{\mathrm{rank}_i}.$$

Conclusions

- use Poly-encoder architecture

Conclusions

- use Poly-encoder architecture (with advantages from Bi- and Cross-encoder)

Conclusions

- use Poly-encoder architecture (with advantages from Bi- and Cross-encoder)
- fast inference using caching

Conclusions

- use Poly-encoder architecture (with advantages from Bi- and Cross-encoder)
- fast inference using caching
- end-to-end

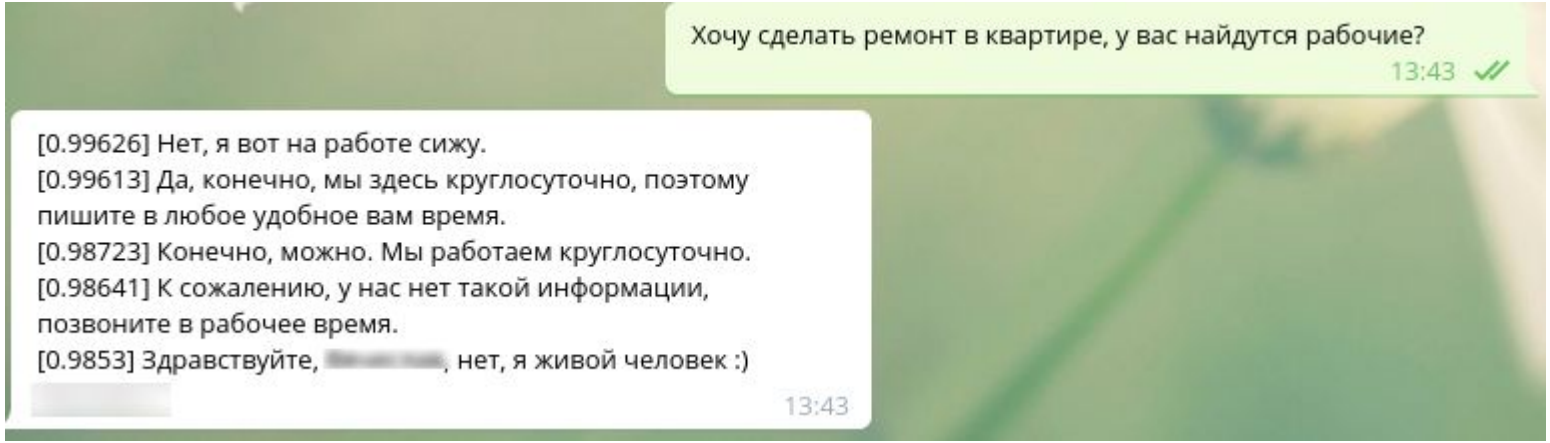
Conclusions

- use Poly-encoder architecture (with advantages from Bi- and Cross-encoder)
- fast inference using caching
- end-to-end
- no manual reading and expensive preprocessing

Conclusions

- use Poly-encoder architecture (with advantages from Bi- and Cross-encoder)
- fast inference using caching
- end-to-end
- no manual reading and expensive preprocessing
- zero-shot learning

Use case


Хочу сделать ремонт в квартире, у вас найдутся рабочие?
13:43
[0.99626] Нет, я вот на работе сижу.
[0.99613] Да, конечно, мы здесь круглосуточно, поэтому пишите в любое удобное вам время.
[0.98723] Конечно, можно. Мы работаем круглосуточно.
[0.98641] К сожалению, у нас нет такой информации, позвоните в рабочее время.
[0.9853] Здравствуйте, , нет, я живой человек :)
13:43

Just AI

“ CONVERSATIONS

Thank you!

Questions?

d.serdyuk@just-ai.com