

Нейросетевой IR

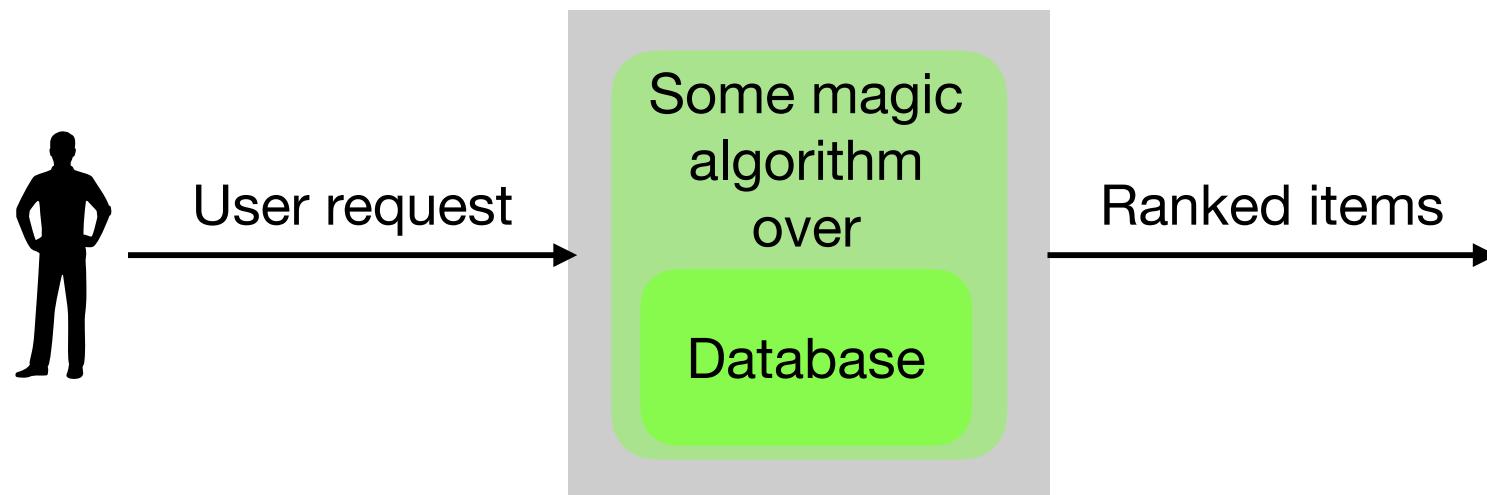
Андрсов Дмитрий, 07.04.2025, AI Masters

План лекции

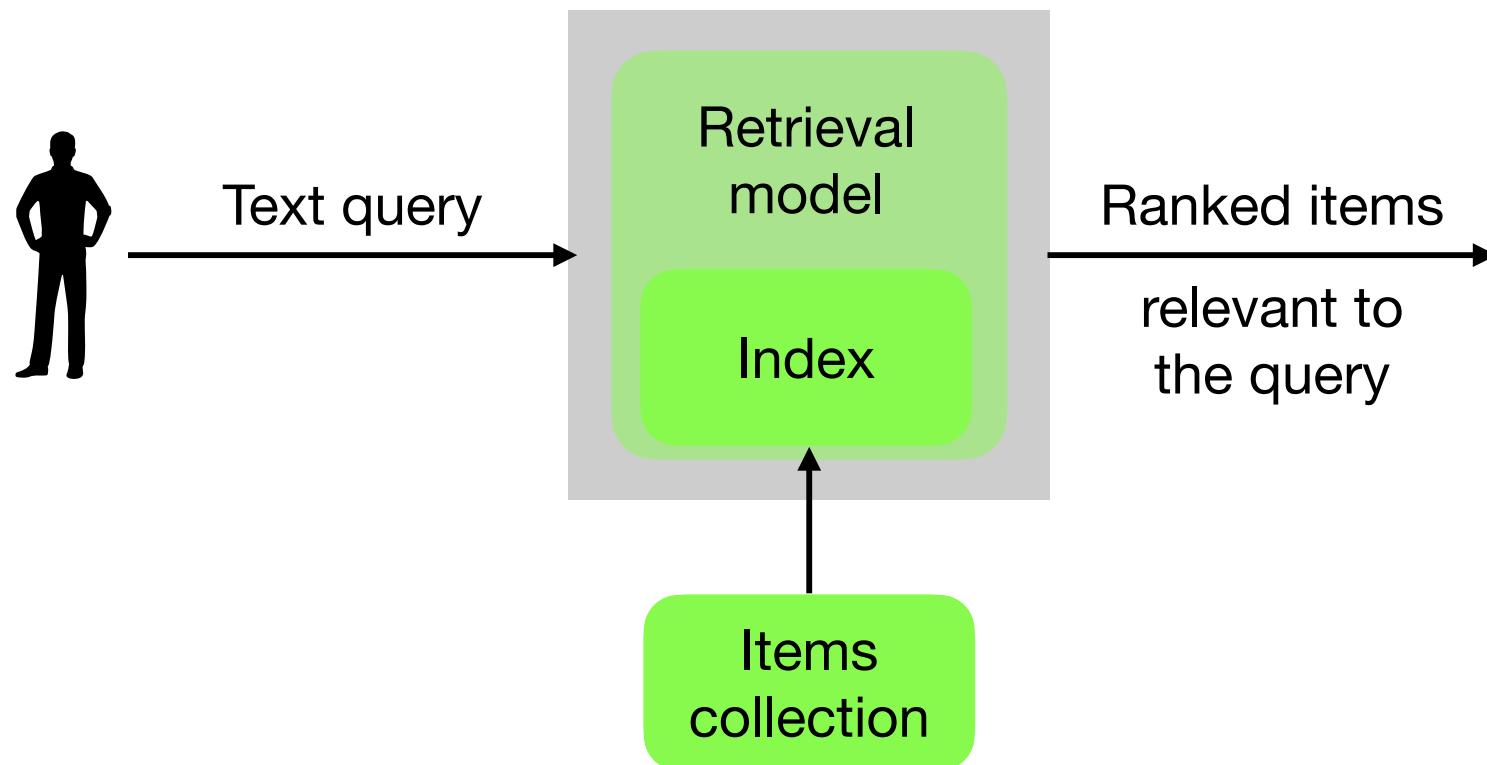
- Вспоминаем основы информационного поиска
- Освежаем NLP-модели
- Нейросетевые модели релевантности
- Обучение моделей релевантности

Вспоминаем IR

Ранжирующая система



Поисковая система



Задача информационного поиска

- Дано:
 - Набор документов $D = \{d_1, d_2, \dots, d_n\}$
 - Запрос $Q = \{q_1, q_2, \dots, q_m\}$
- Найти:
 - Набор документов $D^* \subset D$, релевантных запросу Q

BM25 Model

Близость (релевантность) запроса и документа

$$score(Q, d) = \sum_{t \in Q \cap d} \log \left(\frac{N - df(t) + 0.5}{df(t) + 0.5} + 1 \right) \cdot \frac{(k_1 + 1) \cdot tf(t, d)}{k_1 \cdot \left(1 - b + b \cdot \frac{dl}{avdl} \right) + tf(t, d)}$$

- $\sum_{t \in Q \cap d}$ — чем больше общих слов в запросе и документе, тем лучше
- $tf(t, d)$ — чем больше встречаемость токена, тем лучше
- $\log \left(\frac{N - df(t) + 0.5}{df(t) + 0.5} + 1 \right)$ — более частые токены — менее важные

Основные проблемы

- Запрос и документ рассматриваются как мешок слов (BoW)
 - Поиск только по четкому совпадению слов
 - Не учитывается контекст
 - Не учитываются схожести слов и синонимы
- Требуется предварительная лингвистическая обработка текста
 - Иногда очень сложно сделать её правильно
- Все формулы моделей релевантности в основном эвристические
 - Хотя и имеют какую-то теоретическую основу

Освежаем NLP-модели

<https://jalammar.github.io/illustrated-transformer/>

<https://www.deeplearning.ai/short-courses/how-transformer-langs-work/>

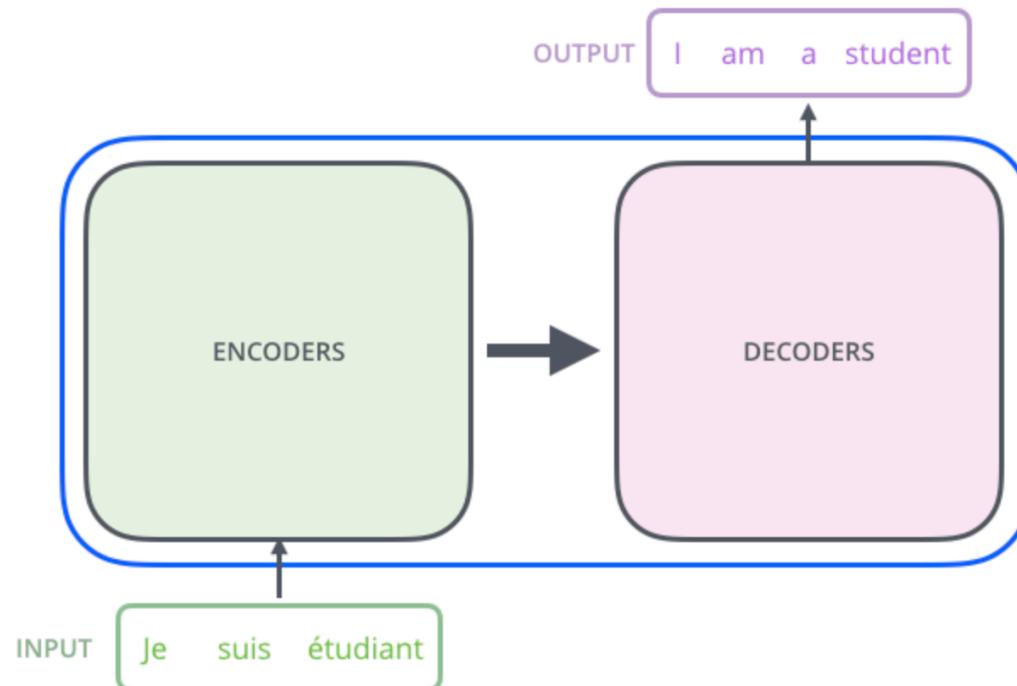
Transformer

Модель как чёрный ящик



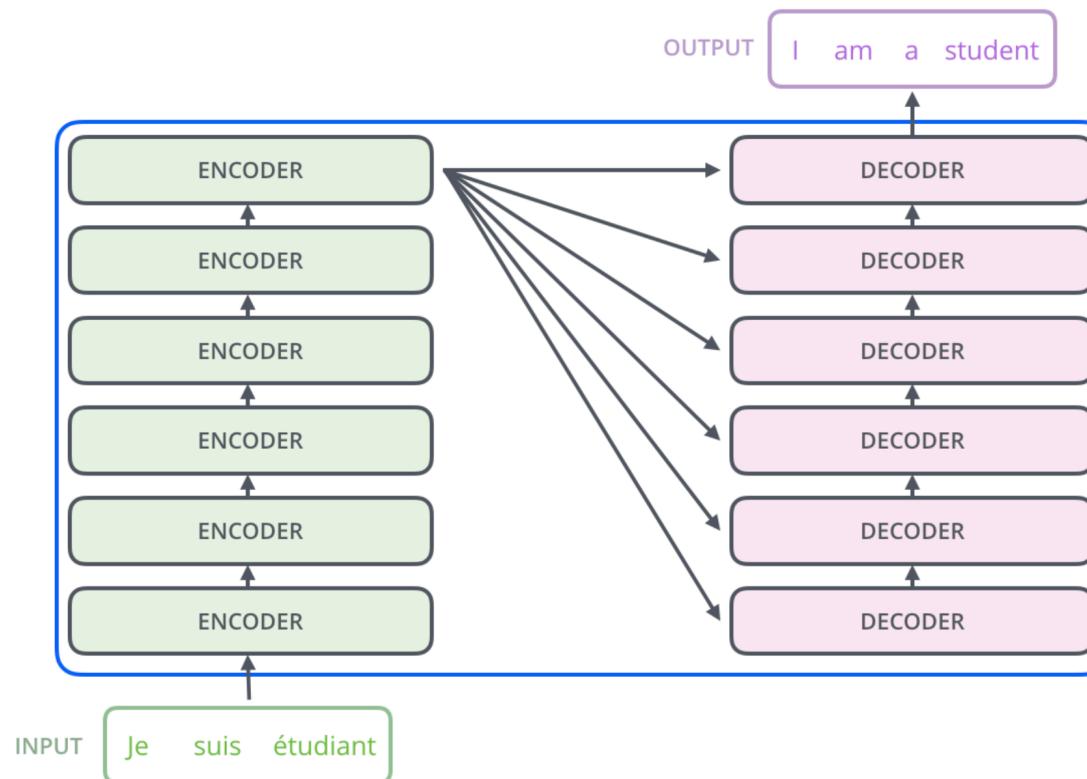
Transformer

Заглядываем внутрь



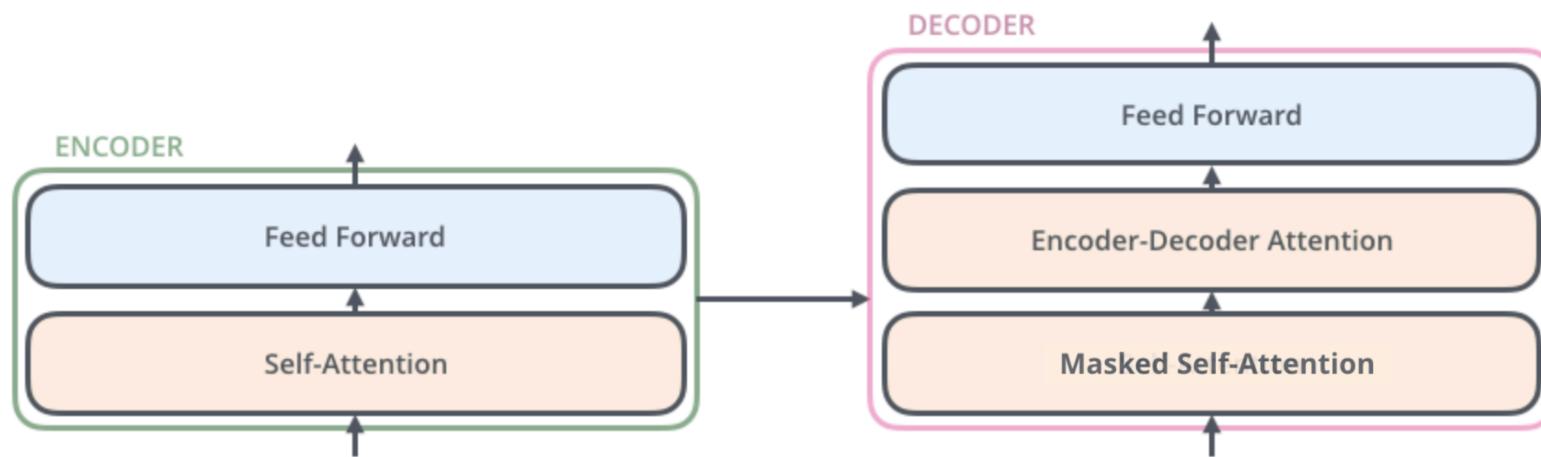
Transformer

Заглядываем внутрь

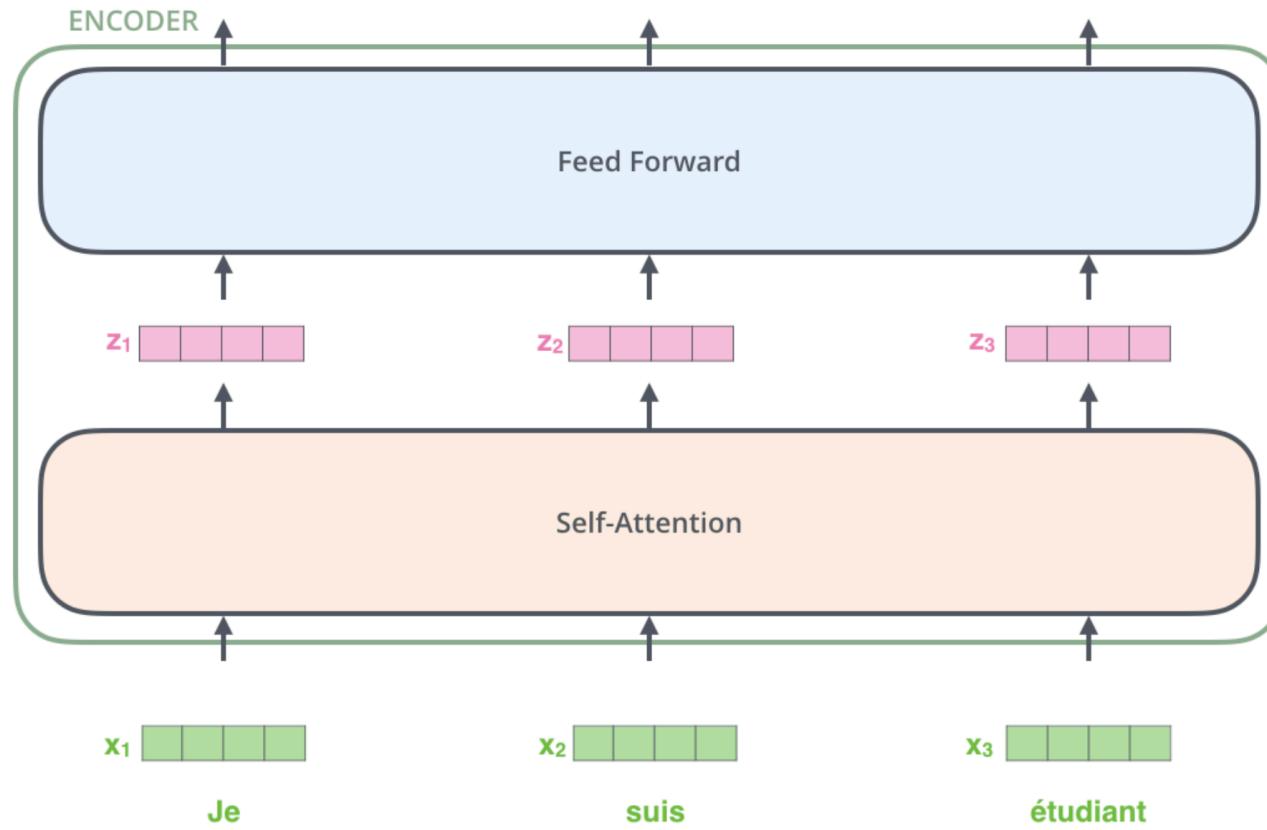


Transformer

Encoder и Decoder

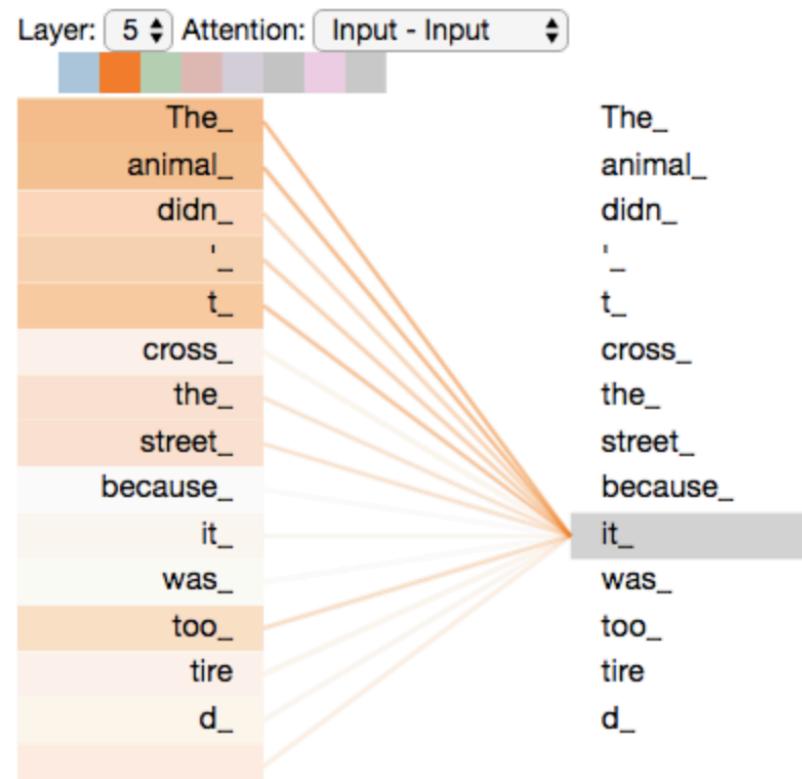


Transformer Encoder



Transformer

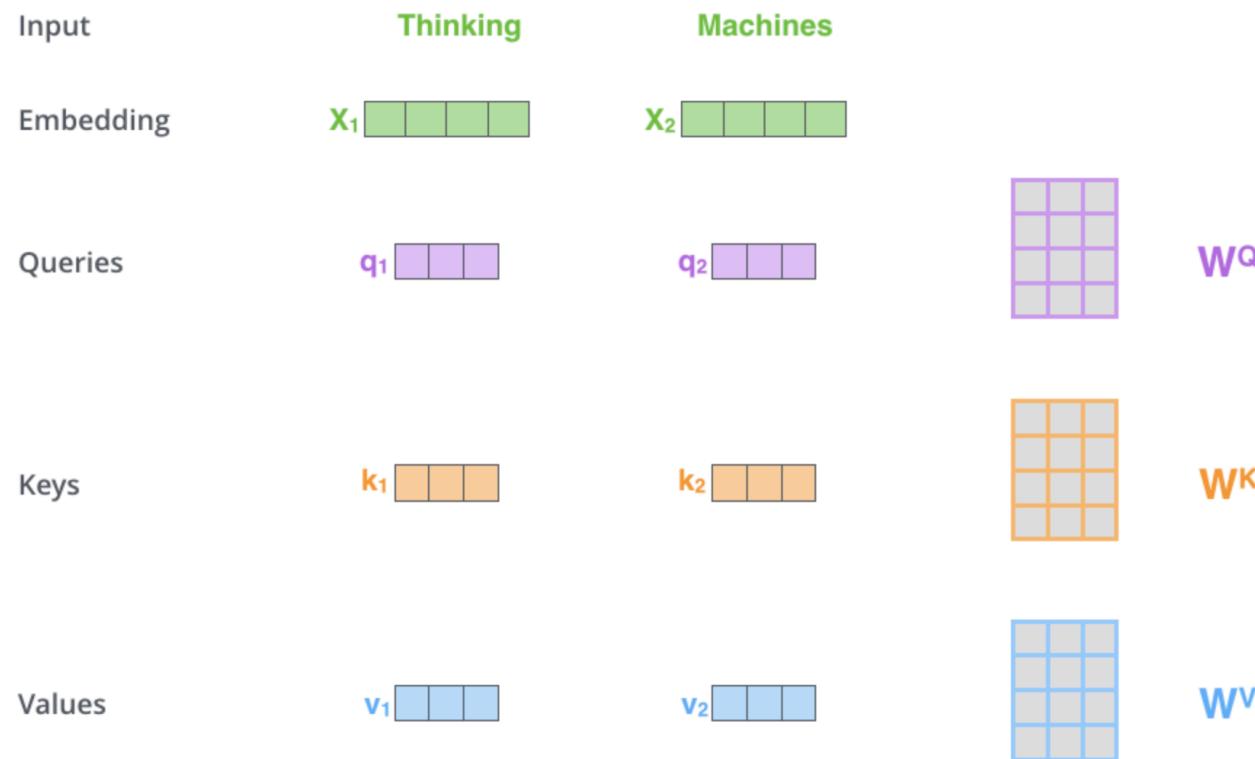
Self-attention – верхнеуровнево



Механизм внимания позволяет ассоциировать токен «it_» с токеном «animal_»

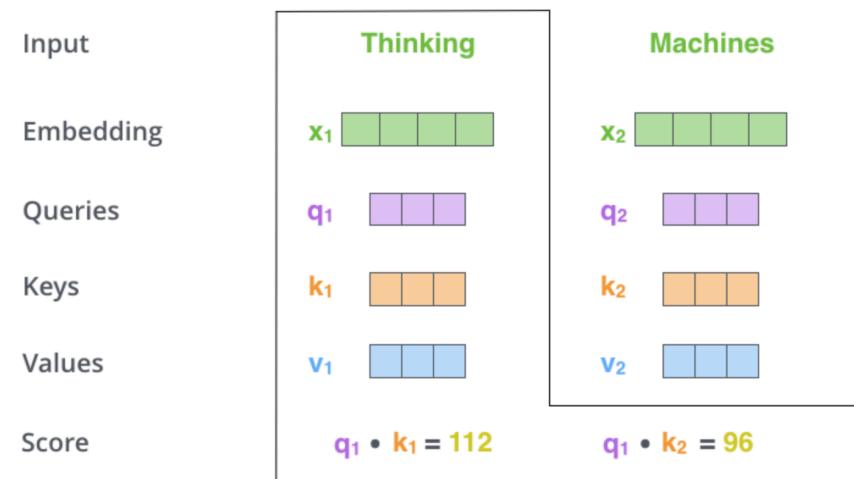
Transformer

Self-attention – детально



Transformer

Self-attention – детально



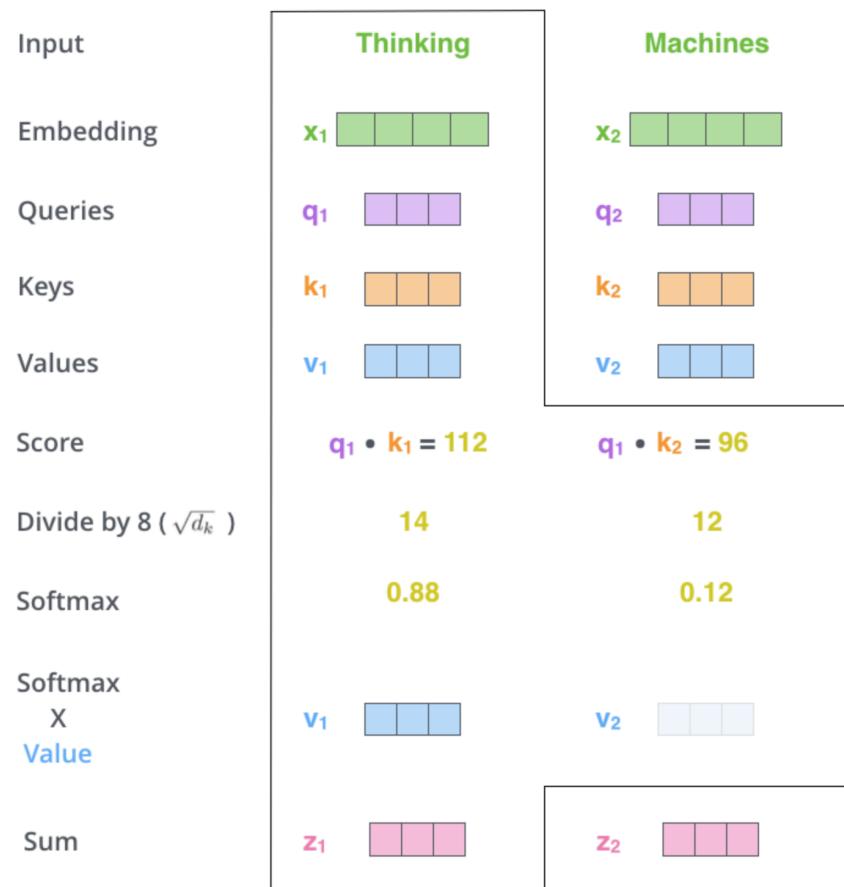
Transformer

Self-attention – детально

Input	Thinking	Machines
Embedding	x_1	x_2
Queries	q_1	q_2
Keys	k_1	k_2
Values	v_1	v_2
Score	$q_1 \cdot k_1 = 112$	$q_1 \cdot k_2 = 96$
Divide by 8 ($\sqrt{d_k}$)	14	12
Softmax	0.88	0.12

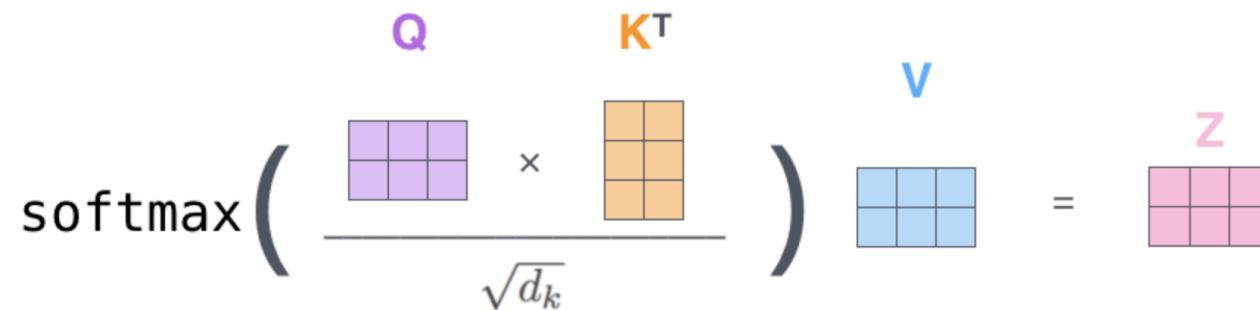
Transformer

Self-attention – детально



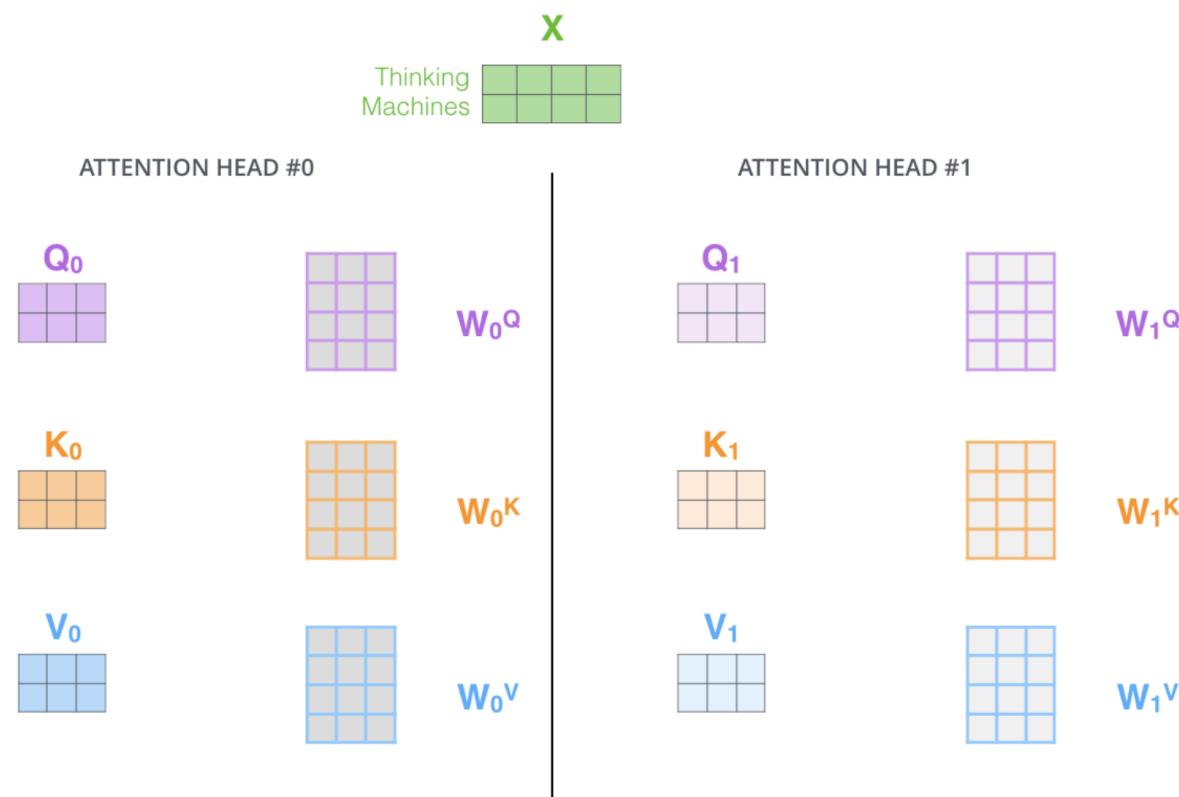
Transformer

Self-attention – матричная запись

$$\text{softmax} \left(\frac{\begin{array}{c} \text{Q} \\ \times \\ \text{K}^T \end{array}}{\sqrt{d_k}} \right) \text{V} = \text{z}$$


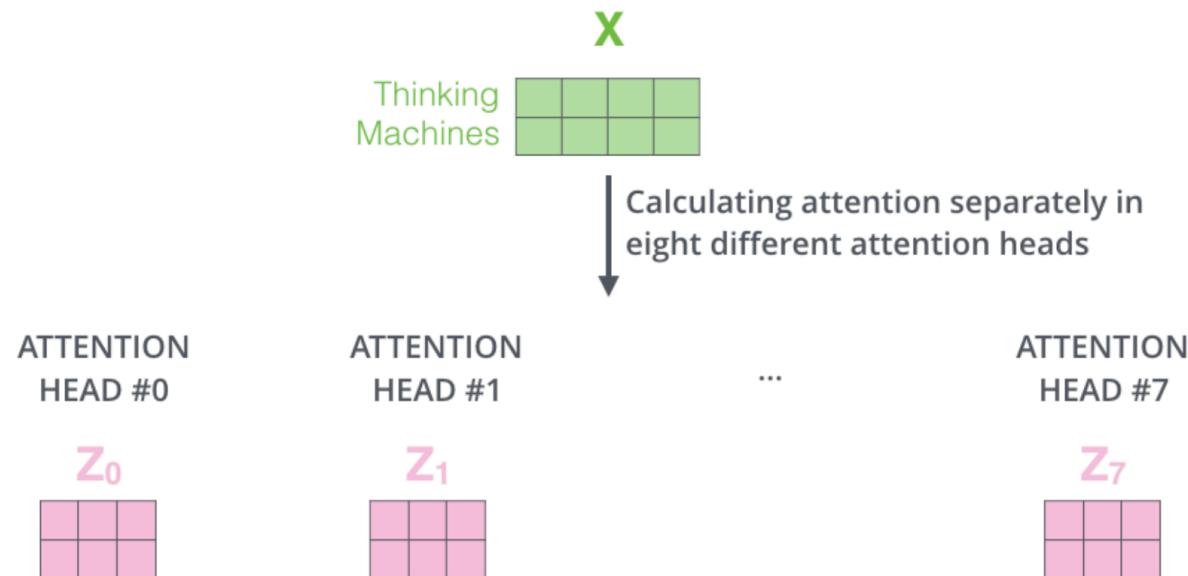
Transformer

Multi-head self-attention



Transformer

Multi-head self-attention



Transformer

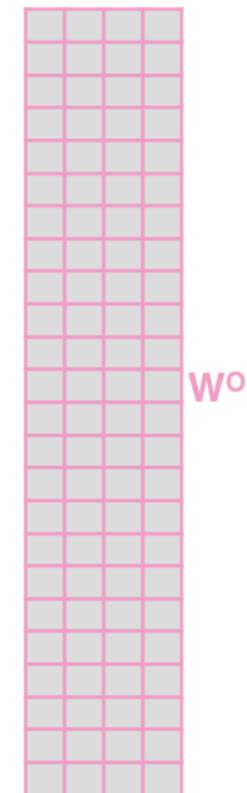
Multi-head self-attention

1) Concatenate all the attention heads



2) Multiply with a weight matrix W^o that was trained jointly with the model

\times



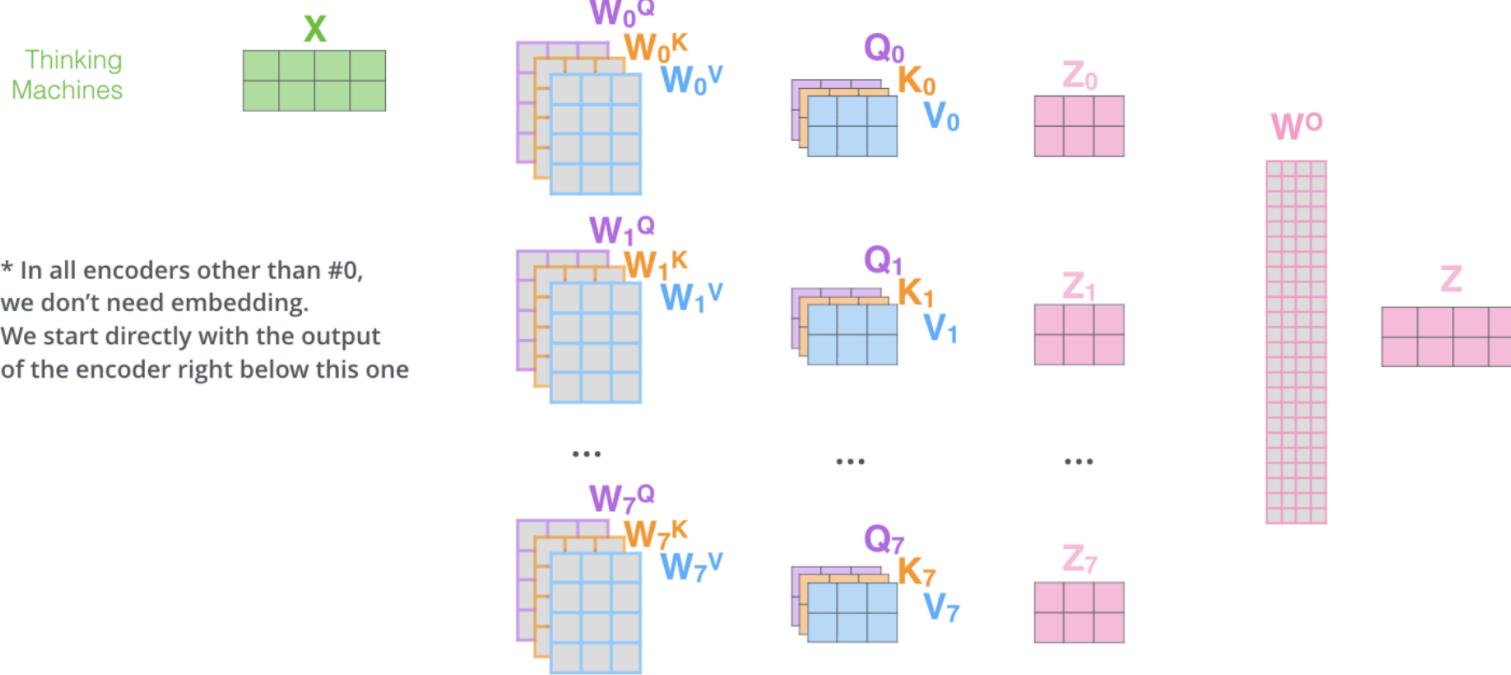
3) The result would be the Z matrix that captures information from all the attention heads. We can send this forward to the FFNN

$$= \begin{matrix} Z \\ \hline \end{matrix}$$

Transformer

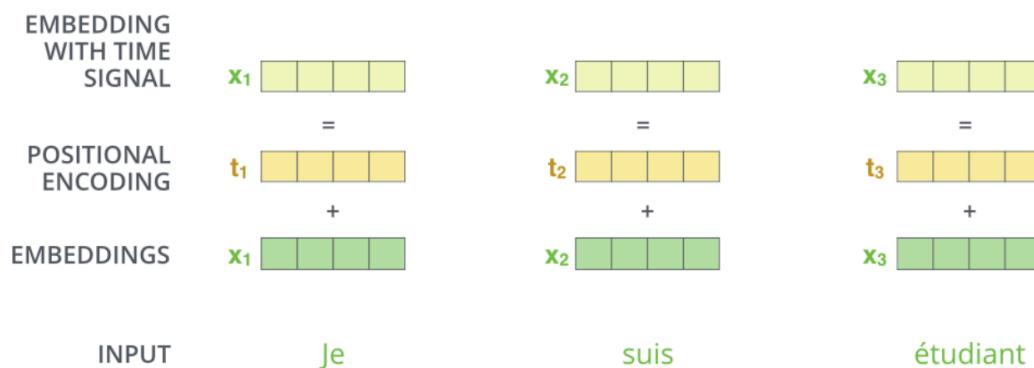
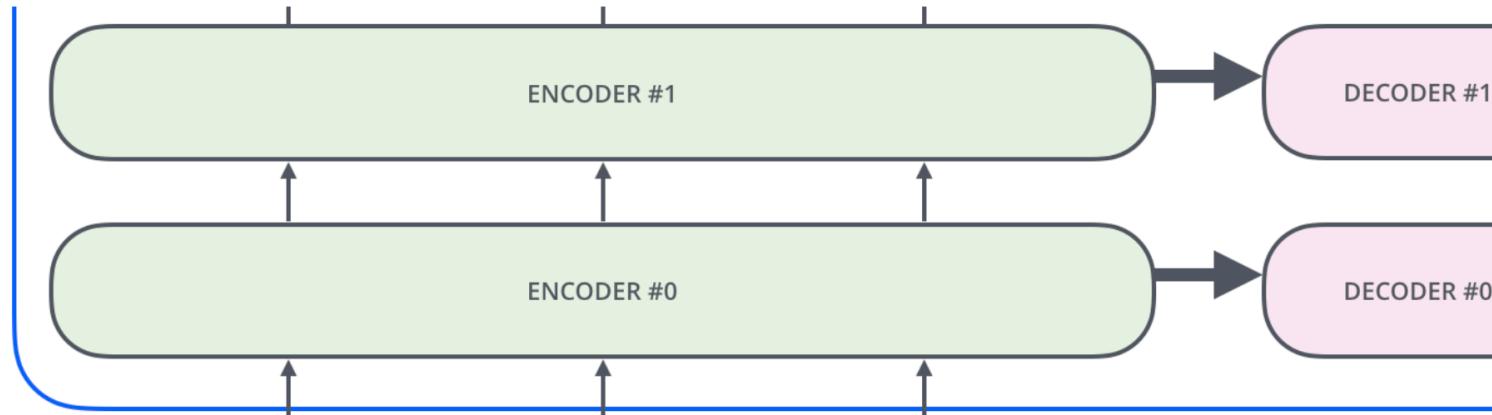
Multi-head self-attention

- 1) This is our input sentence* each word*
- 2) We embed
- 3) Split into 8 heads. We multiply X or R with weight matrices
- 4) Calculate attention using the resulting $Q/K/V$ matrices
- 5) Concatenate the resulting Z matrices, then multiply with weight matrix W^O to produce the output of the layer

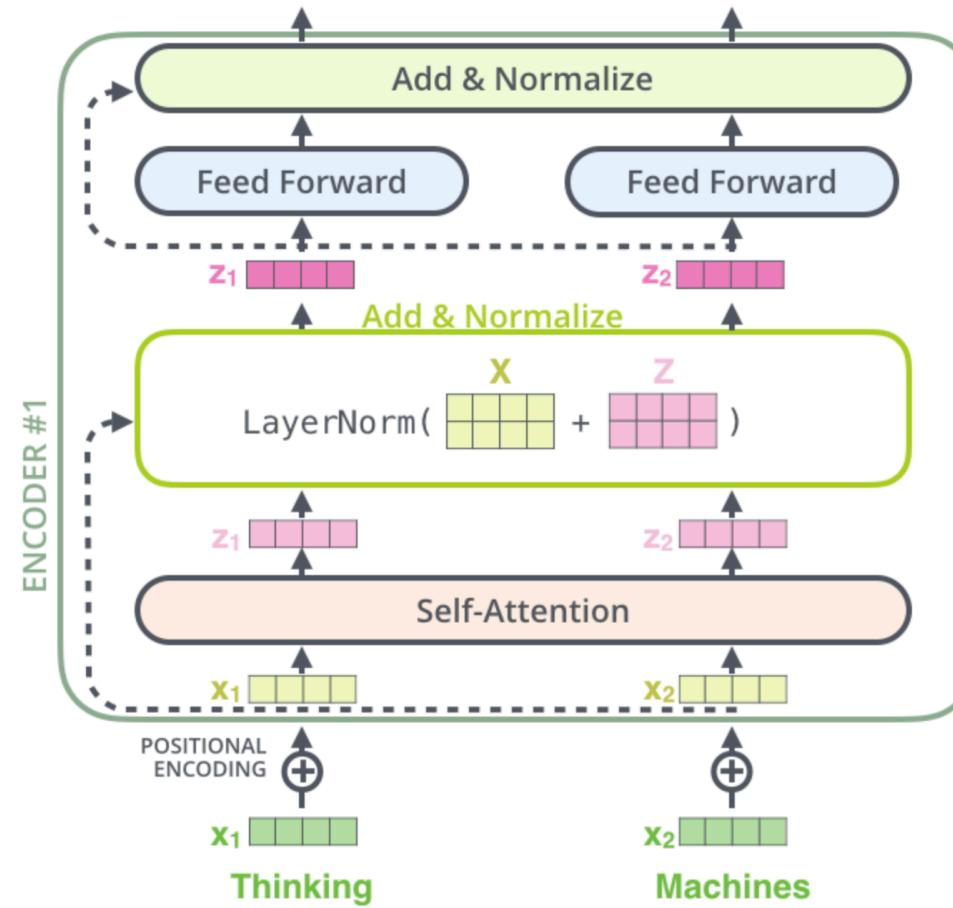


Transformer

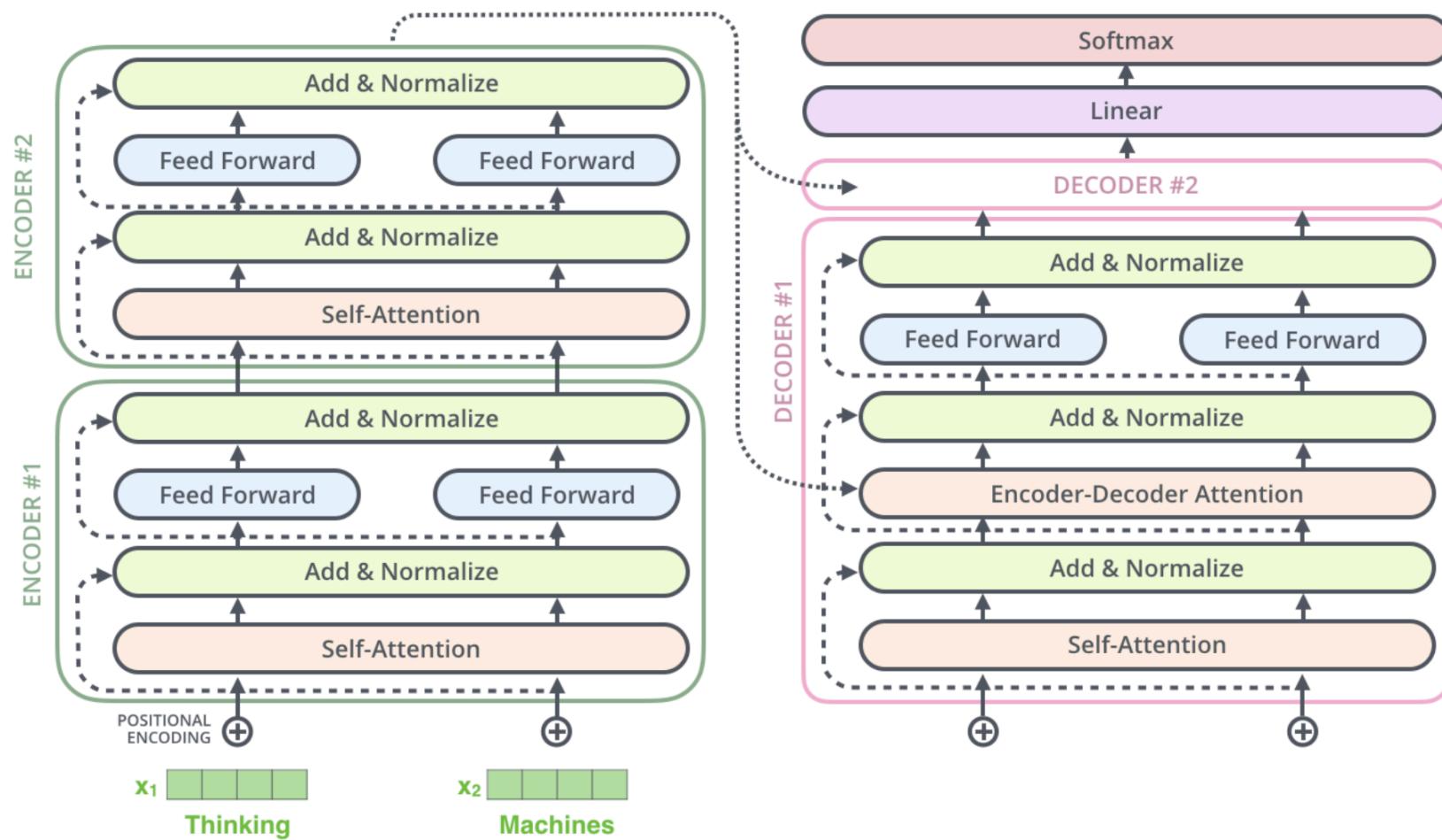
Информация о последовательности (времени)



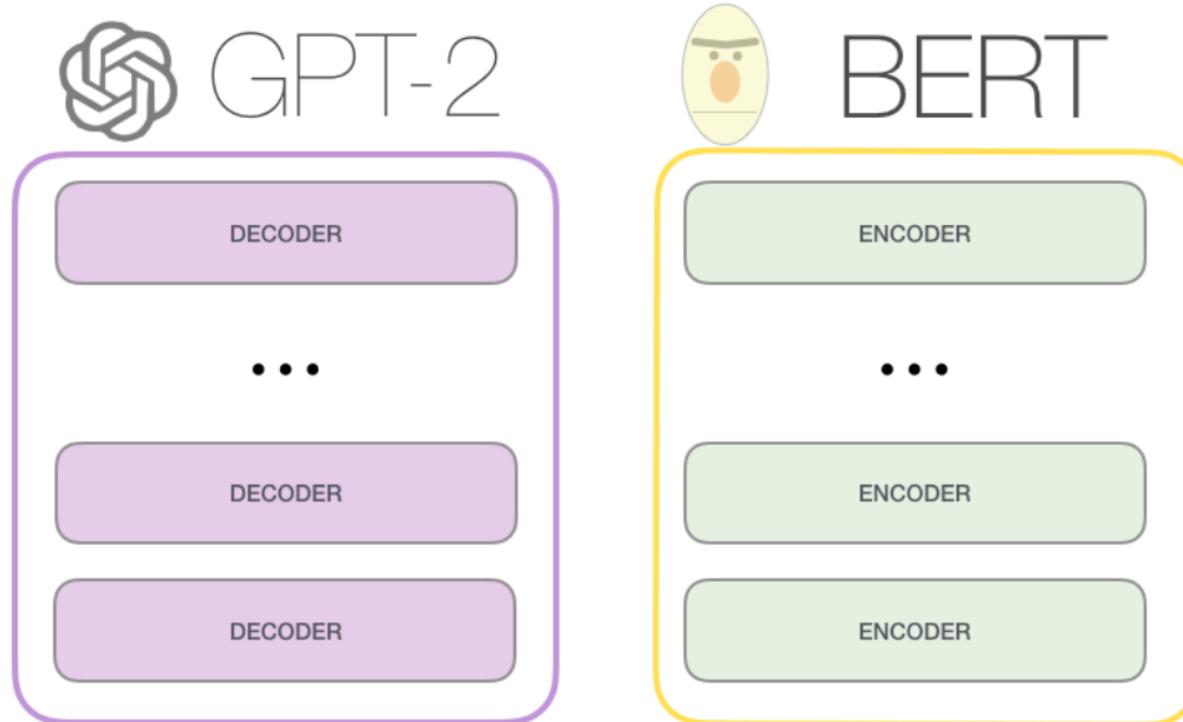
Transformer Encoder



Transformer

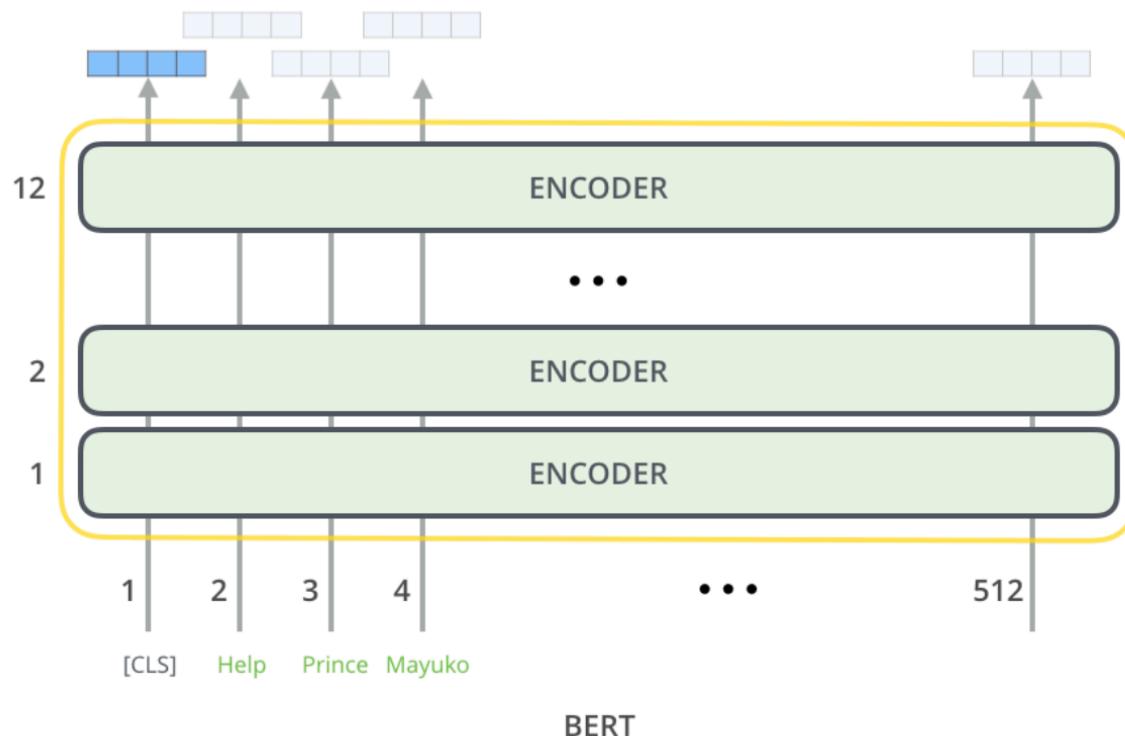


BERT и GPT



BERT

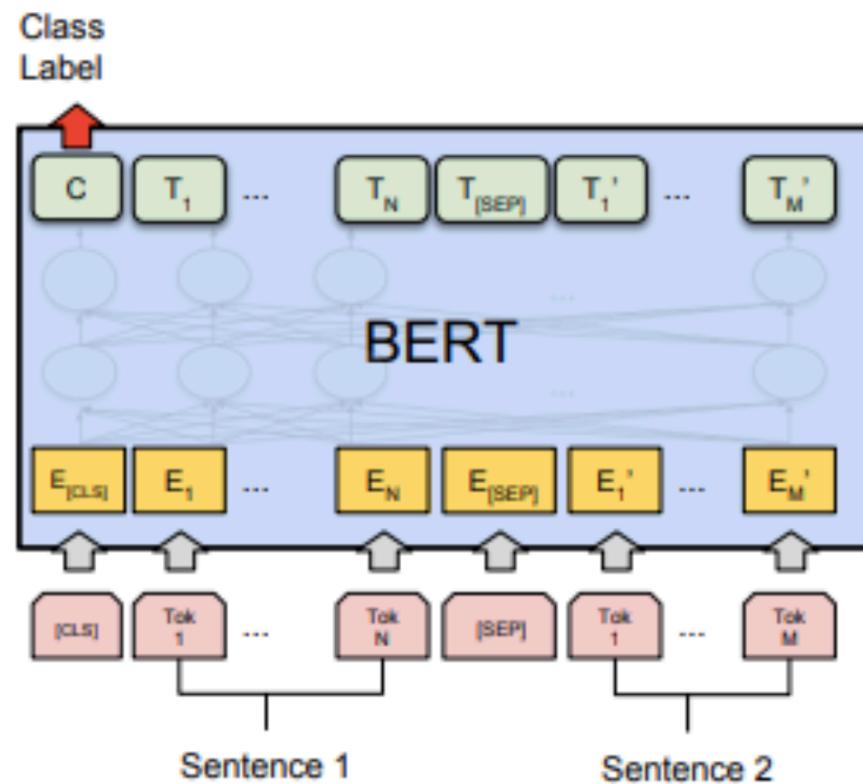
Bidirectional Encoder Representations from Transformers



<https://jalammar.github.io/illustrated-bert/>

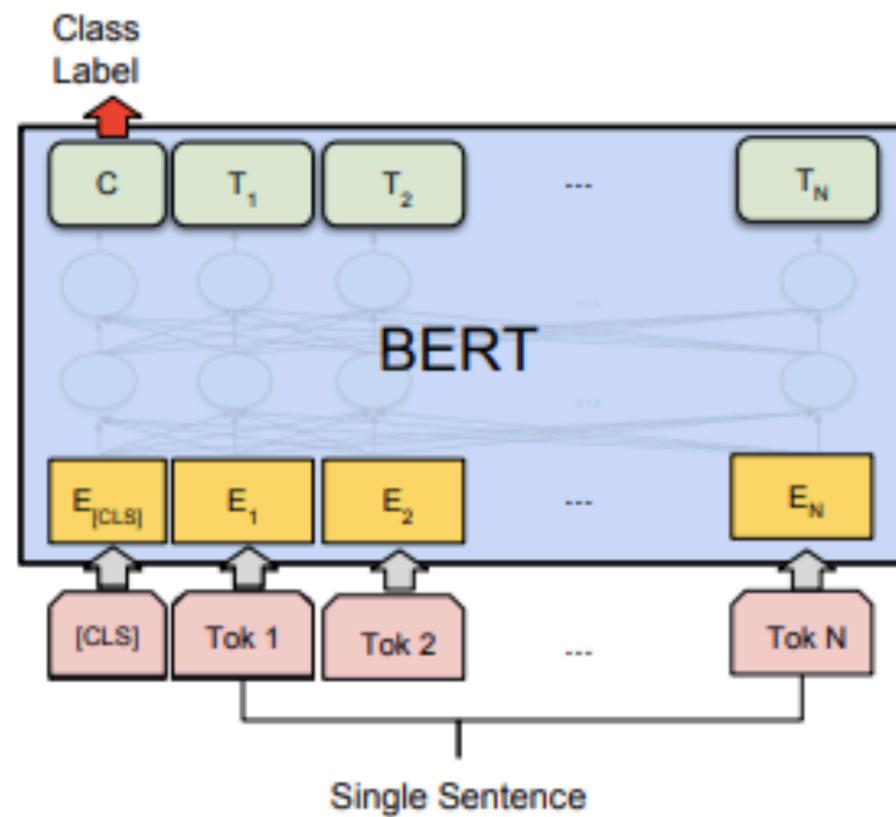
BERT - Применение для различных NLP-задач

Sentence Pair Classification Task



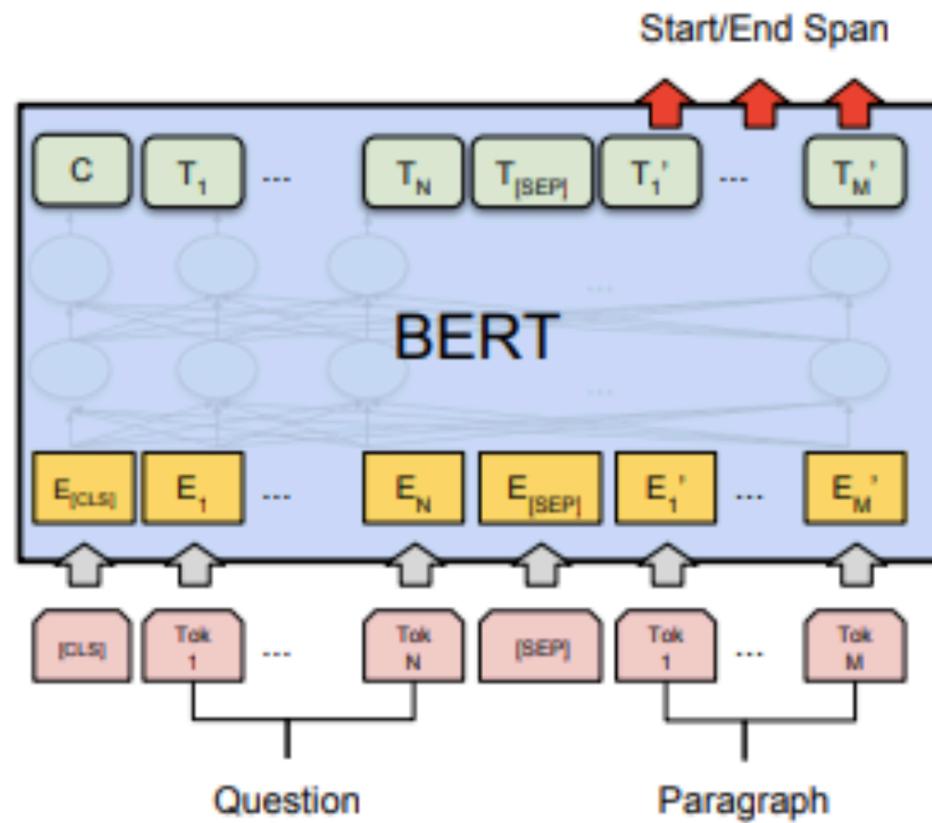
BERT - Применение для различных NLP-задач

Single Sentence Classification Task



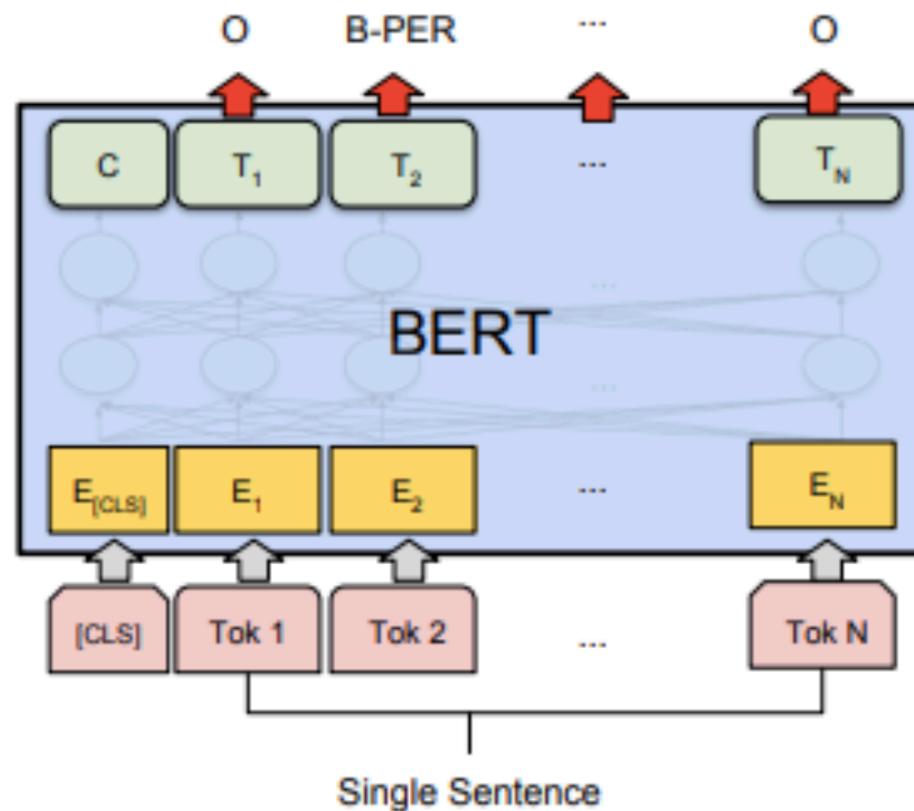
BERT - Применение для различных NLP-задач

Question Answering Task



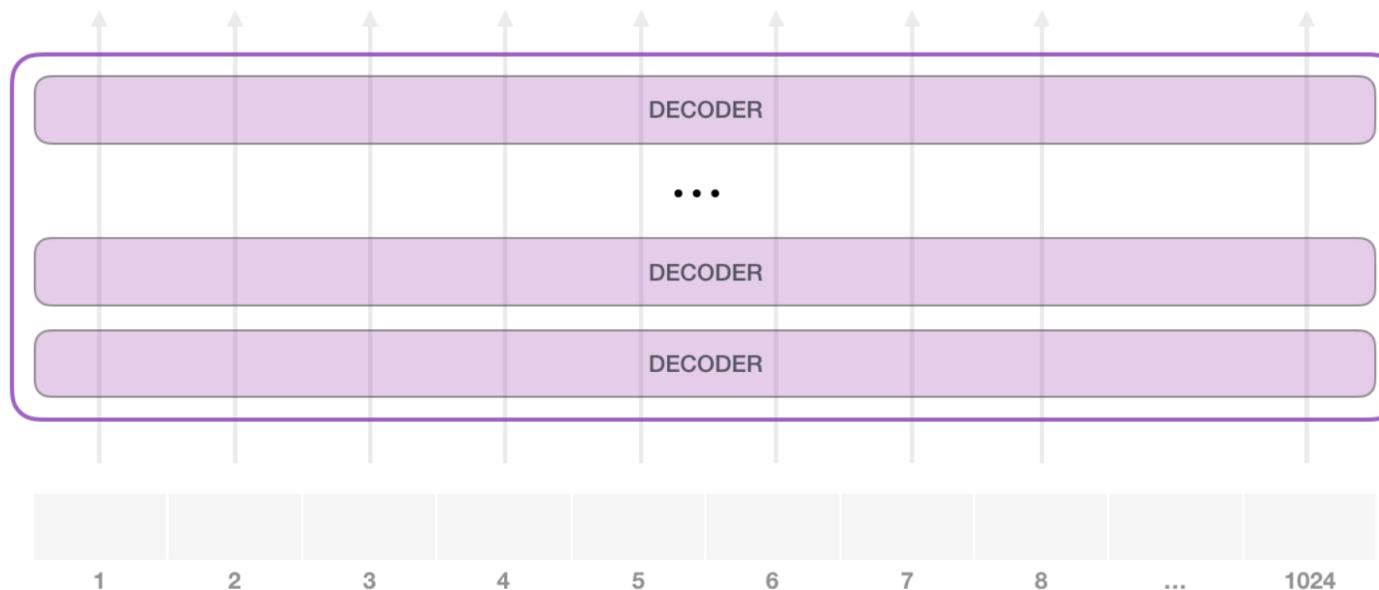
BERT - Применение для различных NLP-задач

Single Sentence Tagging Task



GPT

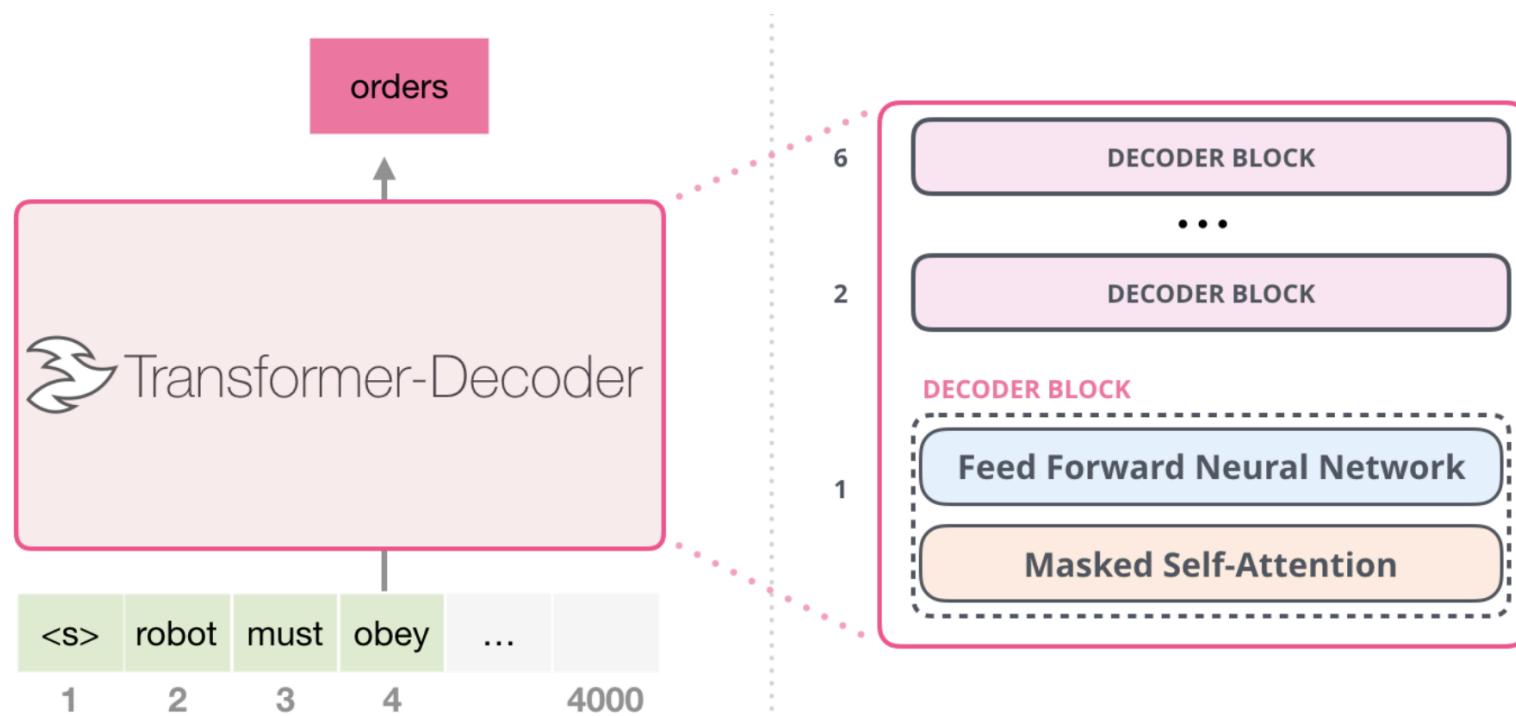
Generative Pre-trained Transformer



<https://jalammar.github.io/illustrated-gpt2/>

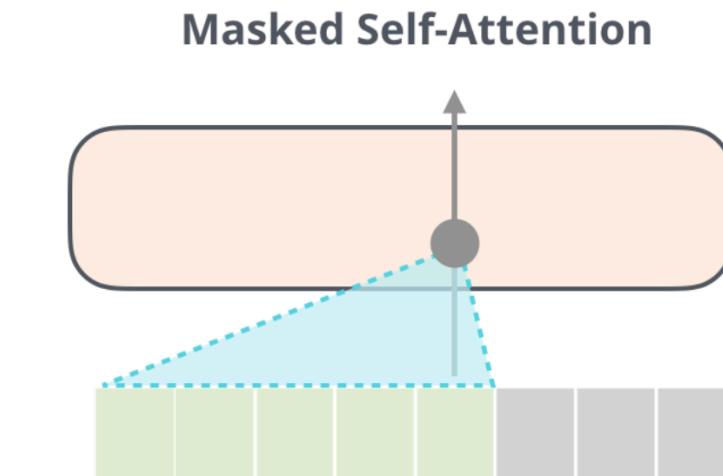
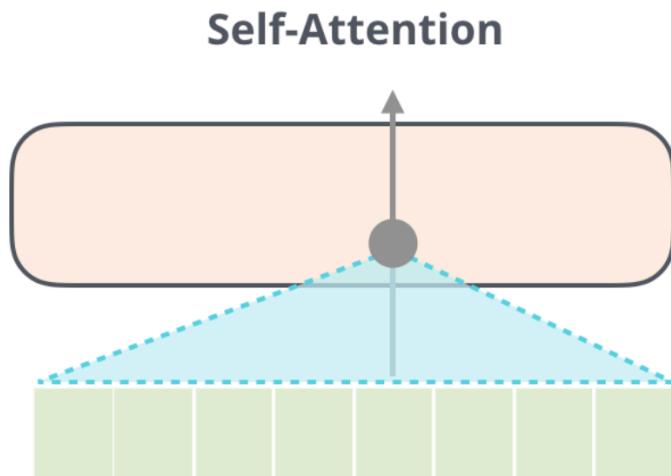
GPT

Generative Pre-trained Transformer



GPT

Masked self-attention



Токенизация текста

BPE-токенизация

Byte-Pair Encoding

- Есть корпус текста, текст построен на символах, которые представляют собой последовательность байтов
- Представили весь корпус в виде последовательности байтов
- Объявляем словарь токенизатора, исходно состоящий из всех уникальных байтов, встречающихся в корпусе
- Ищем в нашем корпусе наиболее популярную пару байтов словаря и добавляем эту пару в словарь как новый токен
- Продолжаем эту процедуру, пока не достигнем заданного размера словаря

BPE-токенизация

Byte-Pair Encoding

- Таким образом, наш словарь состоит из слов и частей слов
- Используя этот словарь, мы можем закодировать любой текст в виде последовательности токенов
- Словарь не должен быть очень большим, поскольку представления будут иметь очень большую размерность
- Словарь не должен быть очень маленьким, поскольку тогда токенизированное представление текста будет длинным, что усложнит последующую работу с ним
- Обычно размер словаря = $10^4 - 10^5$

BPE-токенизация

Пример

- Корпус текстов: [«мама», «рама», «мало», «сало»]

ВРЕ-токенизация

Пример

- Корпус текстов: [«мама», «рама», «мало», «сало»]
- Словарь: [«М», «р», «а», «Л», «О», «С»]
- Токенизированное представление корпуса по словарю:
 - [([«М», «а», «М», «а»), («р», «а», «М», «а»), («М», «а», «Л», «О»), («С», «а», «Л», «О»)]

BPE-токенизация

Пример

- Корпус текстов: [«мама», «рама», «мало», «сало»]
- Словарь: [«М», «р», «а», «Л», «О», «С»]
- Токенизированное представление корпуса по словарю:
 - [(«М», «а», «М», «а»), («р», «а», «М», «а»), («М», «а», «Л», «О»), («С», «а», «Л», «О»)]
- Находим наиболее популярную пару токенов словаря:
 - [«ма»: 4, «ам»: 2, «ал»: 2, «ло»: 2, «ра»: 1, «са»: 1]

ВРЕ-токенизация

Пример

- Корпус текстов: [«мама», «рама», «мало», «сало»]
- Словарь: [«М», «р», «а», «Л», «О», «С»]
- Токенизированное представление корпуса по словарю:
 - [(«М», «а», «М», «а»), («р», «а», «М», «а»), («М», «а», «Л», «О»), («С», «а», «Л», «О»)]
- Находим наиболее популярную пару токенов словаря:
 - [«ма»: 4, «ам»: 2, «ал»: 2, «ло»: 2, «ра»: 1, «са»: 1]
- Добавляем наиболее популярную пару в словарь
 - [«М», «р», «а», «Л», «О», «С», «ма»]

ВРЕ-токенизация

Пример

- Корпус текстов: [«мама», «рама», «мало», «сало»]
- Словарь: [«М», «р», «а», «Л», «О», «С», «ма»]
- Токенизированное представление корпуса по словарю:
 - [(«ма», «ма»), («р», «а», «ма»), («ма», «Л», «О»), («С», «а», «Л», «О»)]

BPE-токенизация

Пример

- Корпус текстов: [«мама», «рама», «мало», «сало»]
- Словарь: [«М», «р», «а», «Л», «О», «С», «ма»]
- Токенизированное представление корпуса по словарю:
 - [(«ма», «ма»), («р», «а», «ма»), («ма», «Л», «О»), («С», «а», «Л», «О»)]
- Находим новую наиболее популярную пару токенов словаря:
 - [«ло»: 2, «мама»: 1, «ра»: 1, «ама»: 1, «мал»: 1, «са»: 1, «ал»: 1]

BPE-токенизация

Пример

- Корпус текстов: [«мама», «рама», «мало», «сало»]
- Словарь: [«М», «р», «а», «Л», «О», «С», «ма»]
- Токенизированное представление корпуса по словарю:
 - [(«ма», «ма»), («р», «а», «ма»), («ма», «Л», «О»), («С», «а», «Л», «О»)]
- Находим новую наиболее популярную пару токенов словаря:
 - [«ло»: 2, «мама»: 1, «ра»: 1, «ама»: 1, «мал»: 1, «са»: 1, «ал»: 1]
- Добавляем наиболее популярную пару в словарь
 - [«М», «р», «а», «Л», «О», «С», «ма», «ло»]

ВРЕ-токенизация

Пример

- Корпус текстов: [«мама», «рама», «мало», «сало»]
- Словарь: [«М», «р», «а», «Л», «О», «С», «ма», «ло»]
- Токенизированное представление корпуса по словарю:
 - [(«ма», «ма»), («р», «а», «ма»), («ма», «ло»), («С», «а», «ло»)]

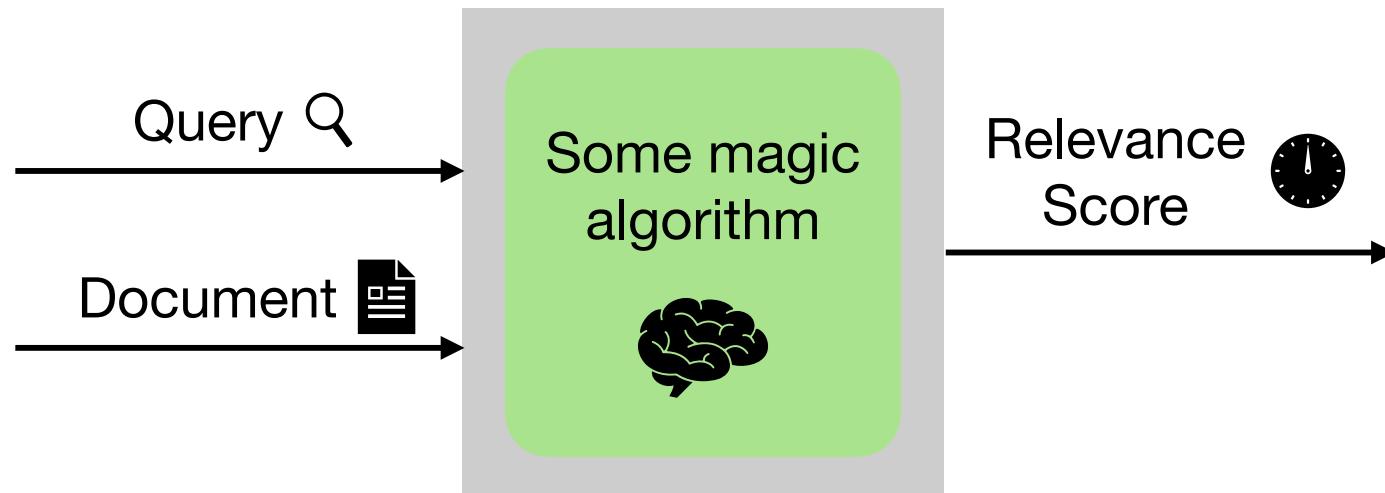
ВРЕ-токенизация

Пример

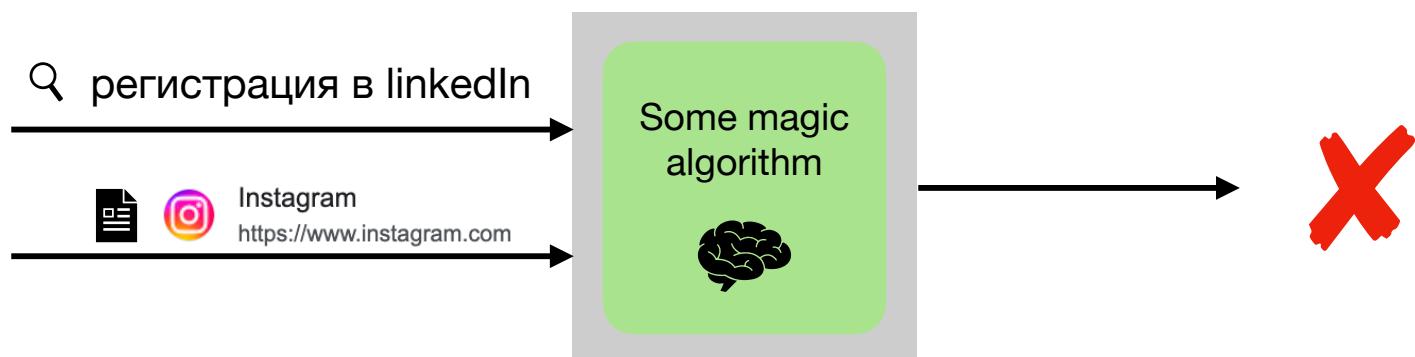
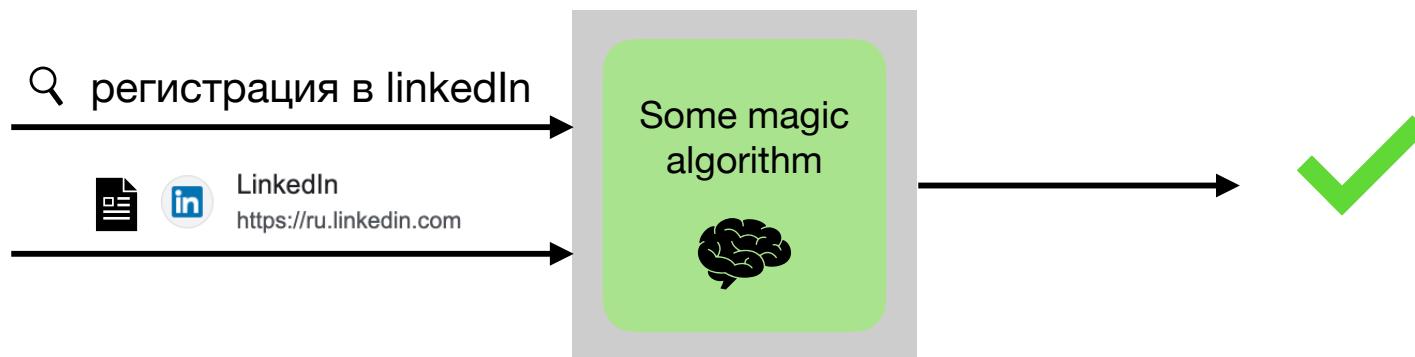
- Корпус текстов: [«мама», «рама», «мало», «сало»]
- Словарь: [«М», «р», «а», «Л», «О», «С», «ма», «ло»]
- Токенизированное представление корпуса по словарю:
 - [(«ма», «ма»), («р», «а», «ма»), («ма», «ло»), («С», «а», «ло»)]
- ...
- Продолжаем алгоритм, пока не достигнем максимального размера словаря

Нейросетевой IR

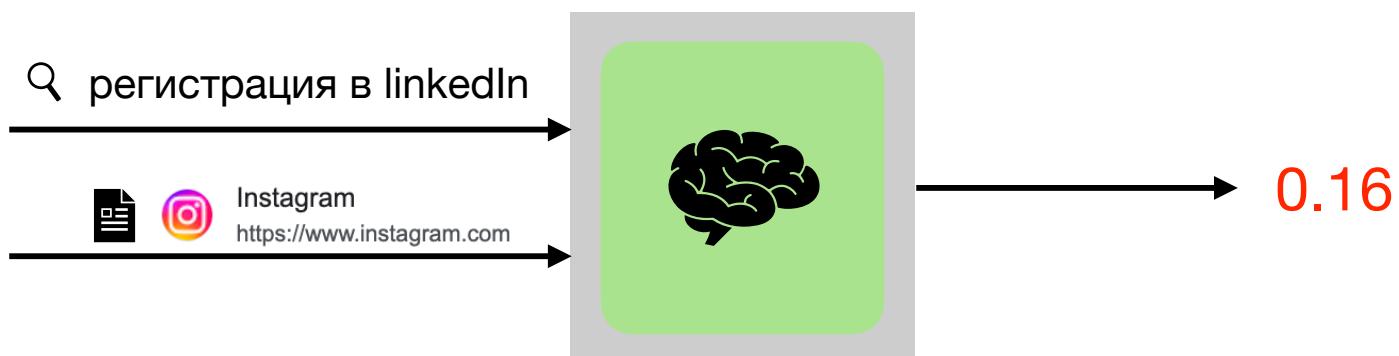
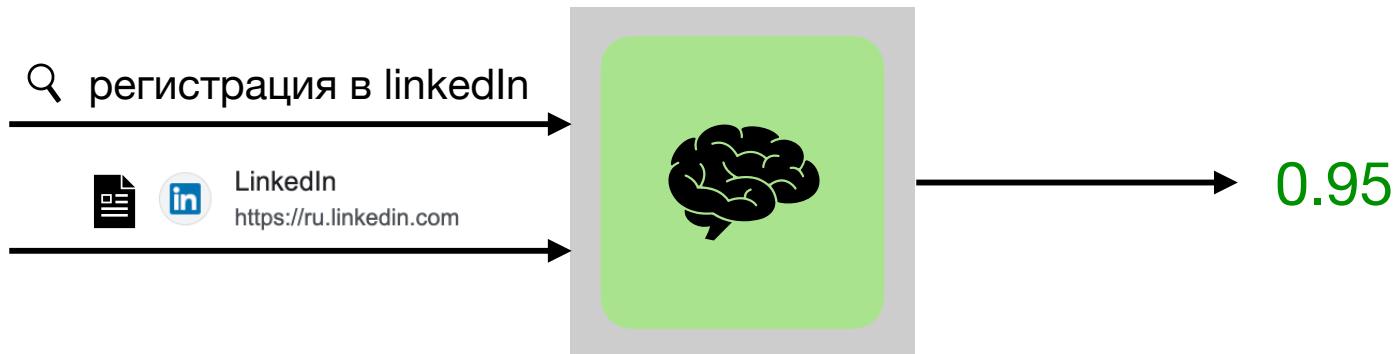
Модели релевантности



Модели релевантности



Модели релевантности



Модели релевантности

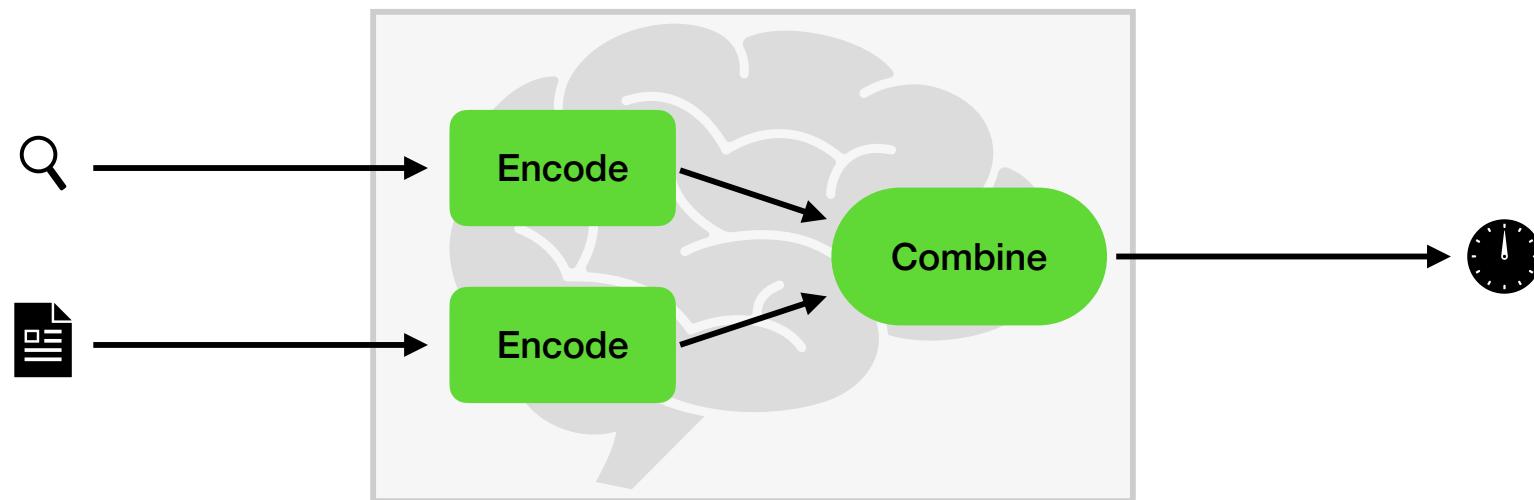


Таблица моделей релевантности

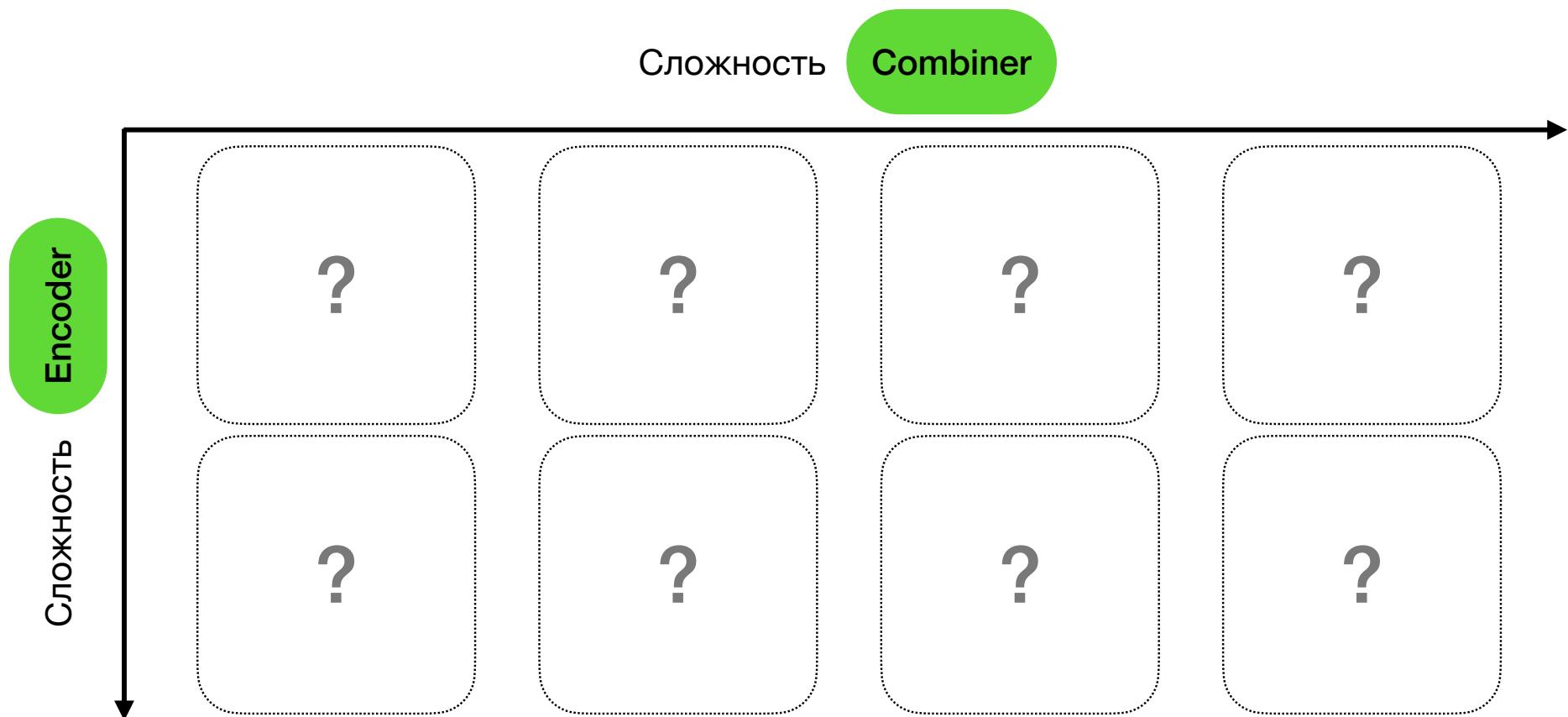


Таблица моделей релевантности

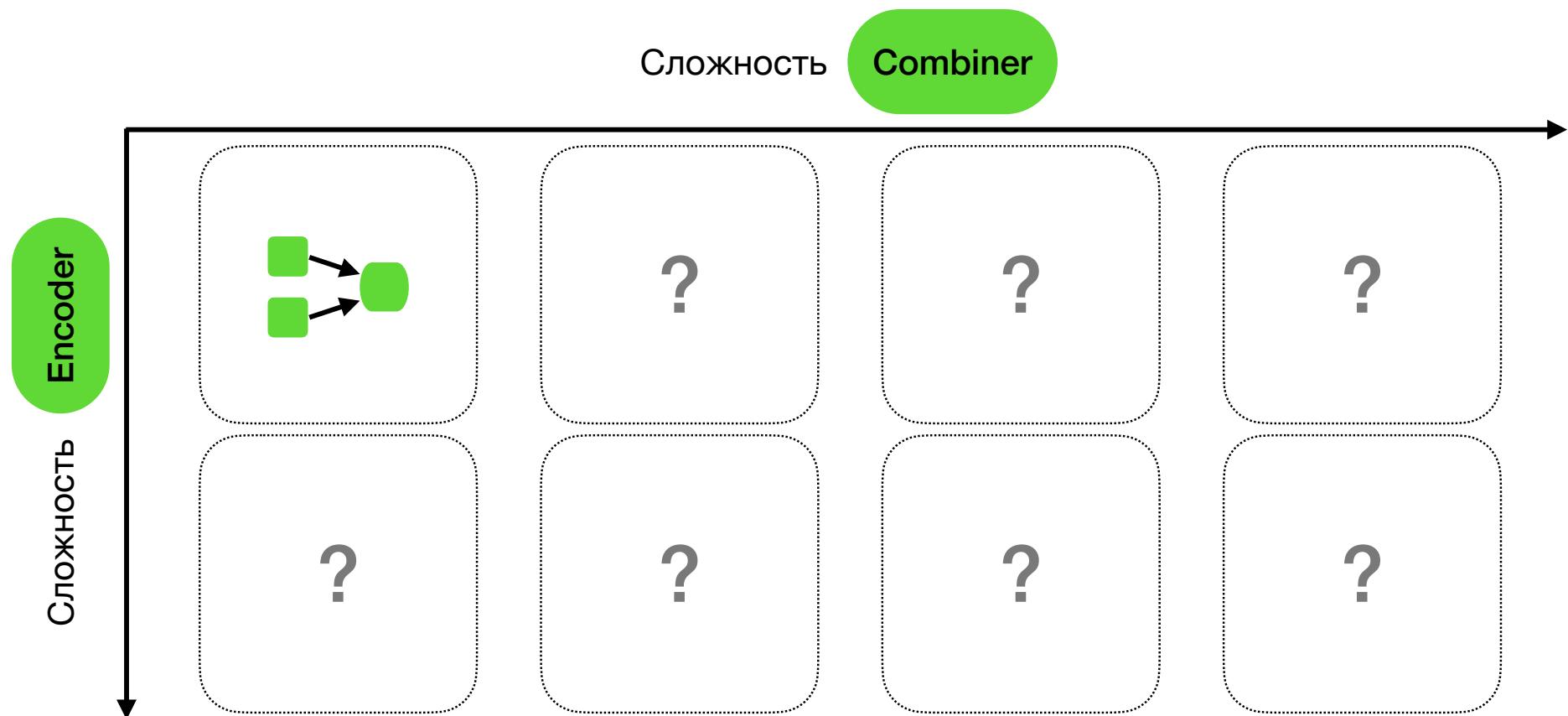


Таблица моделей релевантности

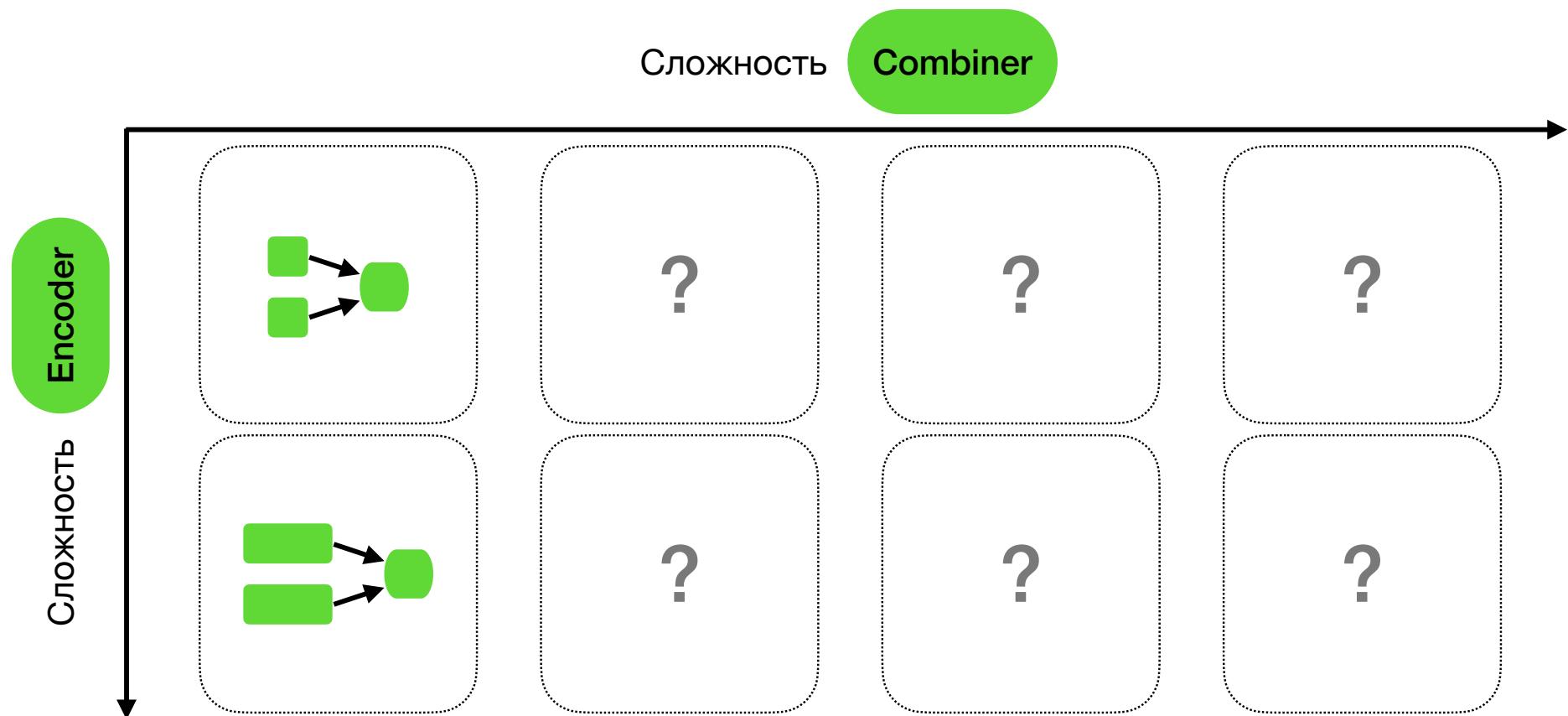


Таблица моделей релевантности

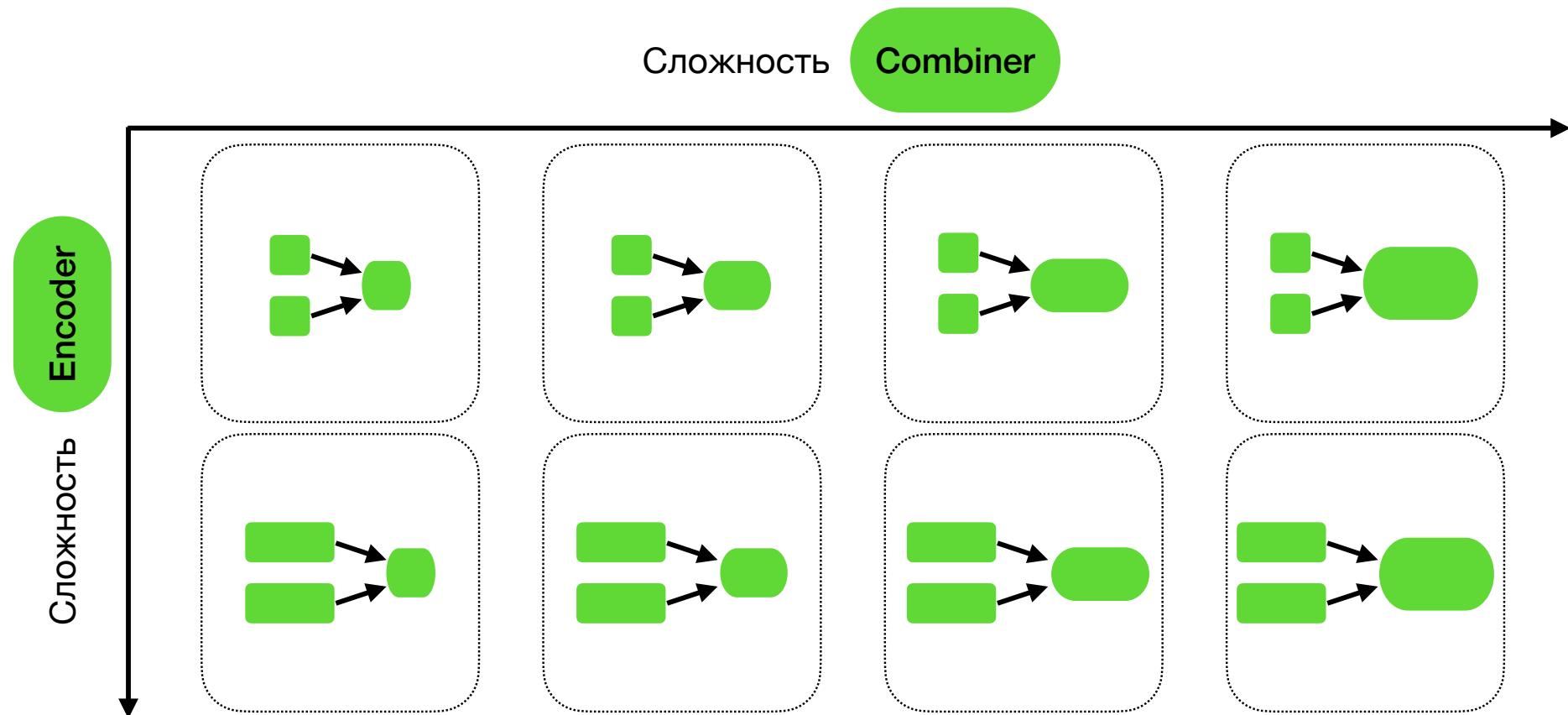
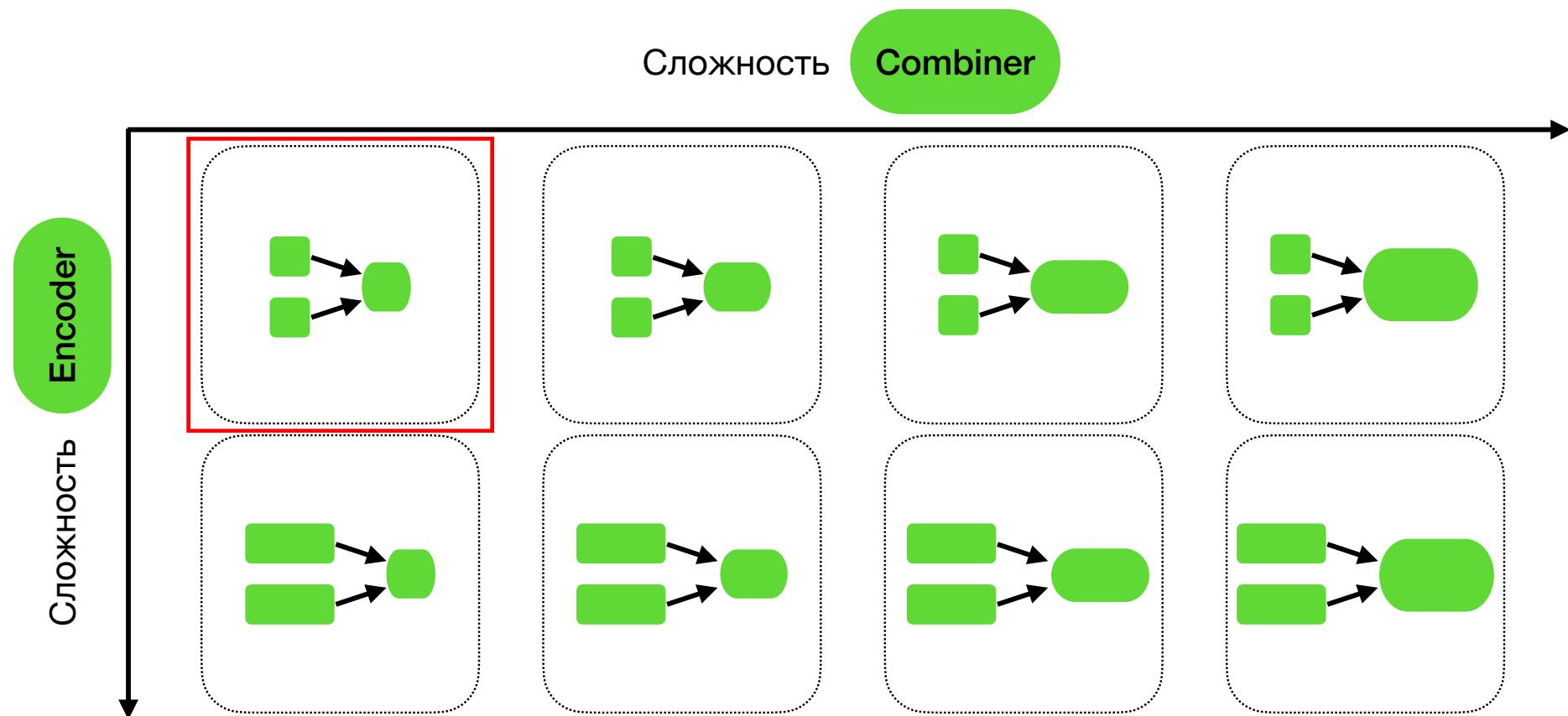


Таблица моделей релевантности



Term Frequency

Encode



регистрация в linkedIn

Encode

{«регистрация»: 1,
«в»: 1, «linkedin»: 1}



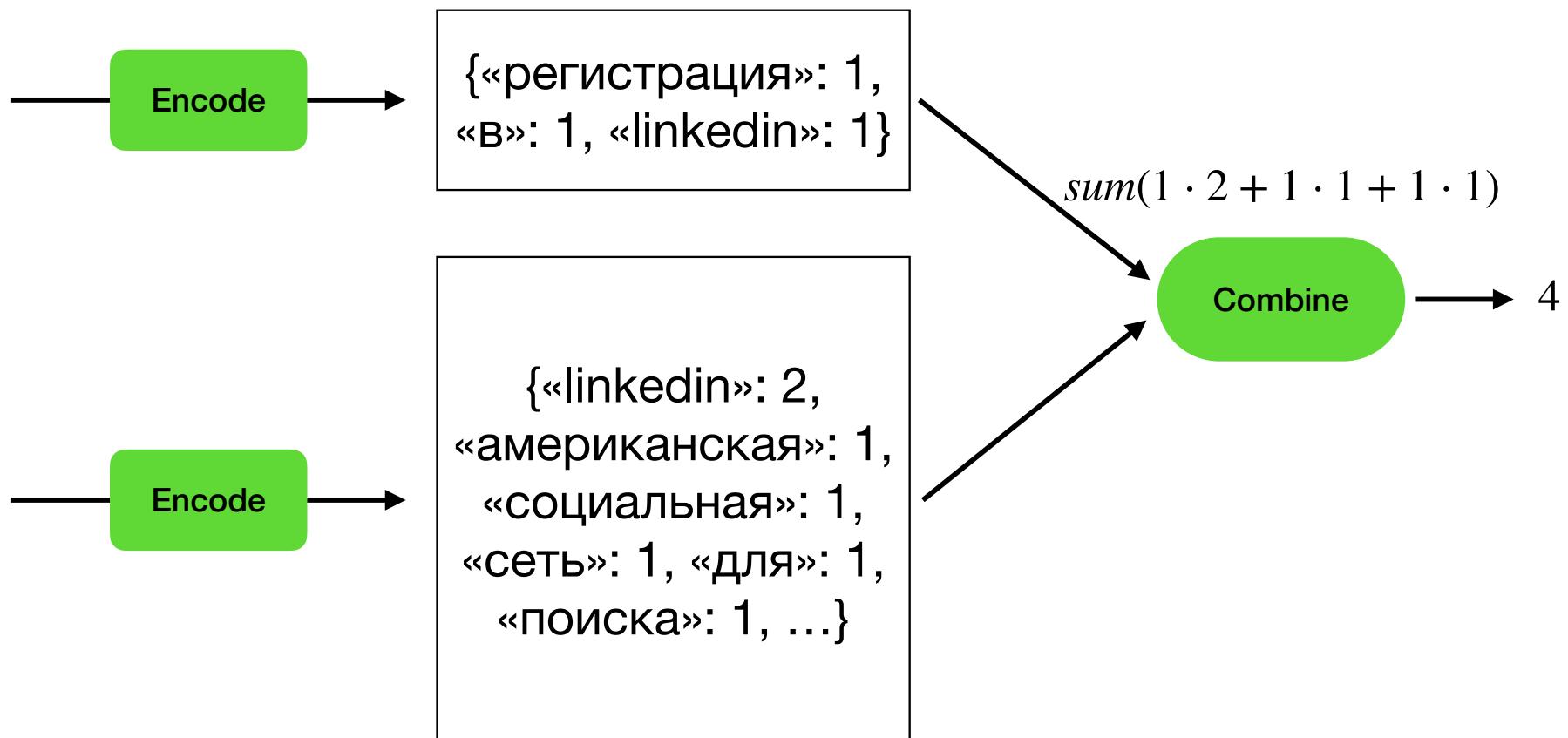
LinkedIn –
американская
социальная сеть для
поиска и установления
деловых контактов. В
LinkedIn
зарегистрирован
1 млрд пользователей

Encode

{«linkedin»: 2,
«американская»: 1,
«социальная»: 1,
«сеть»: 1, «для»: 1,
«поиска»: 1, ...}

Term Frequency

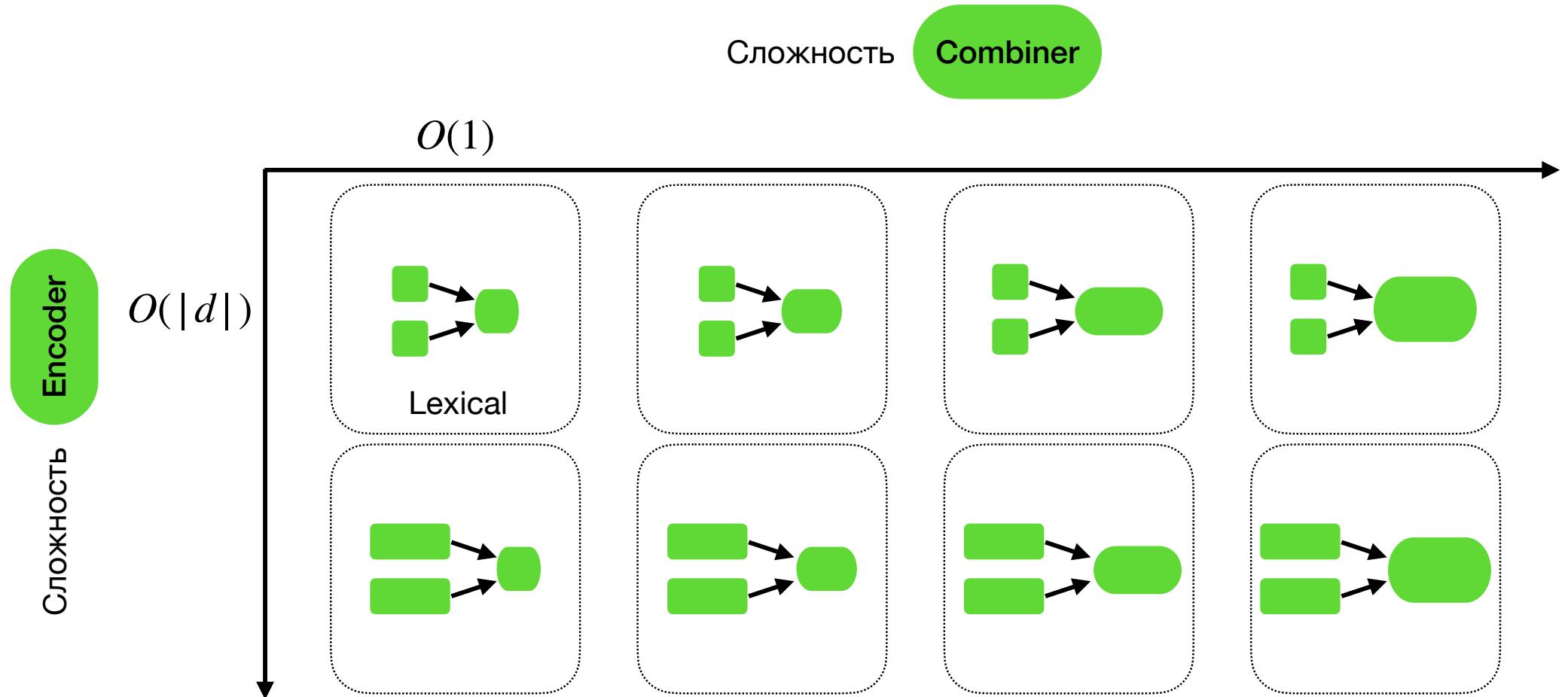
Combine – dot product



Лексические модели релевантности

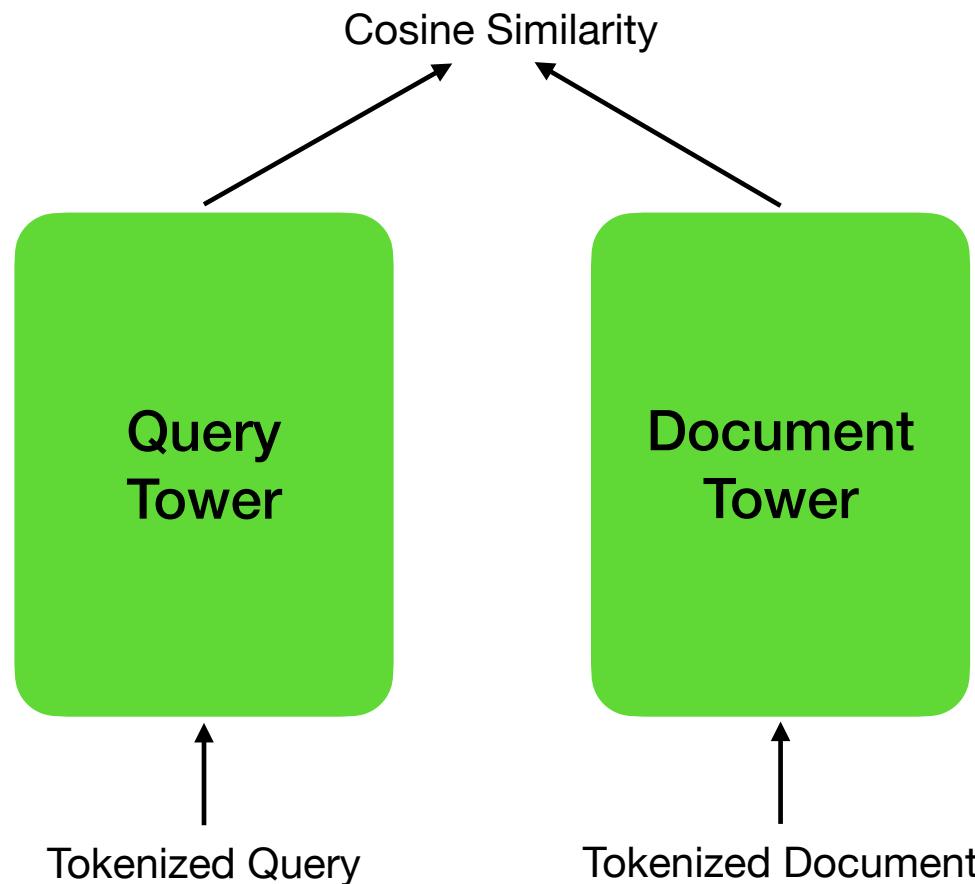
- Представление основывается на лексической форме (BoW)
- Комбинация основывается на пересечении наборов токенов
- Вариации:
 - Стоп-слова
 - Стемминг / Лемматизация
 - Term Frequency
 - Inverse Document Frequency
 - Нормализация на длину документа
 - etc.
- Методы:
 - TF-IDF
 - BM25

Таблица моделей релевантности

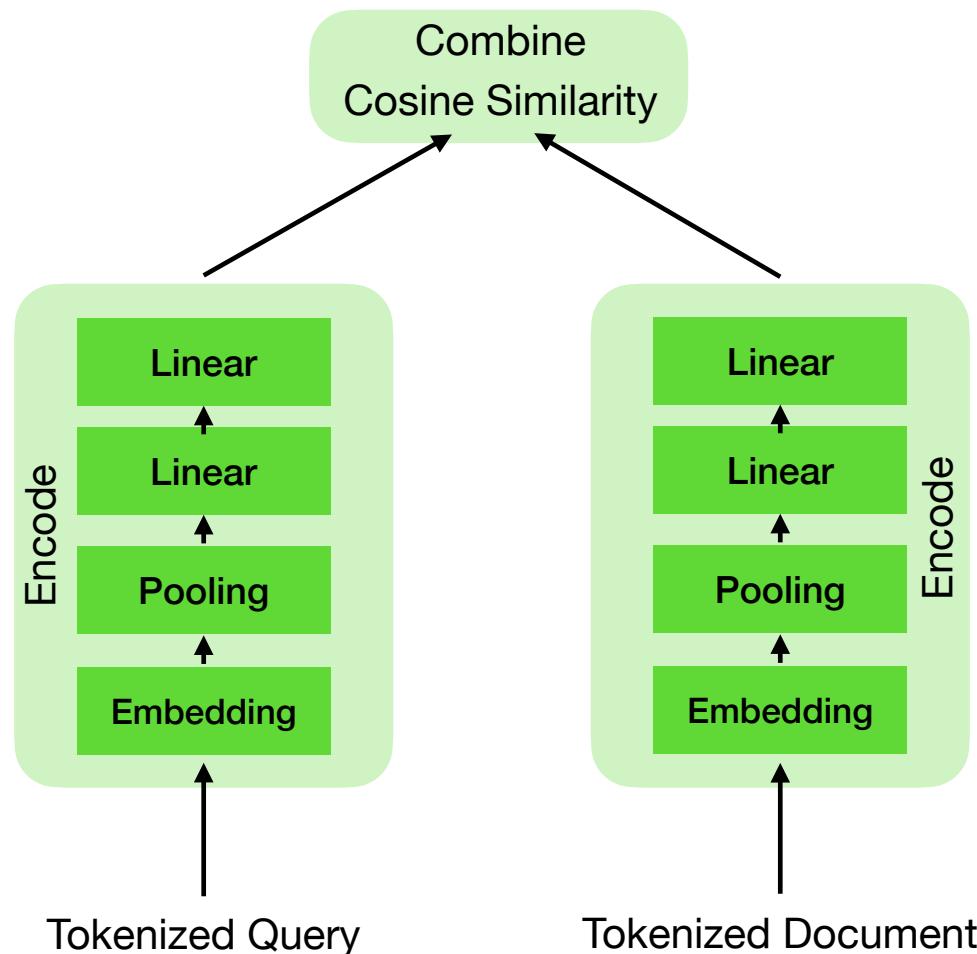


DSSM (2013, Microsoft)

Deep Structured Semantic Model



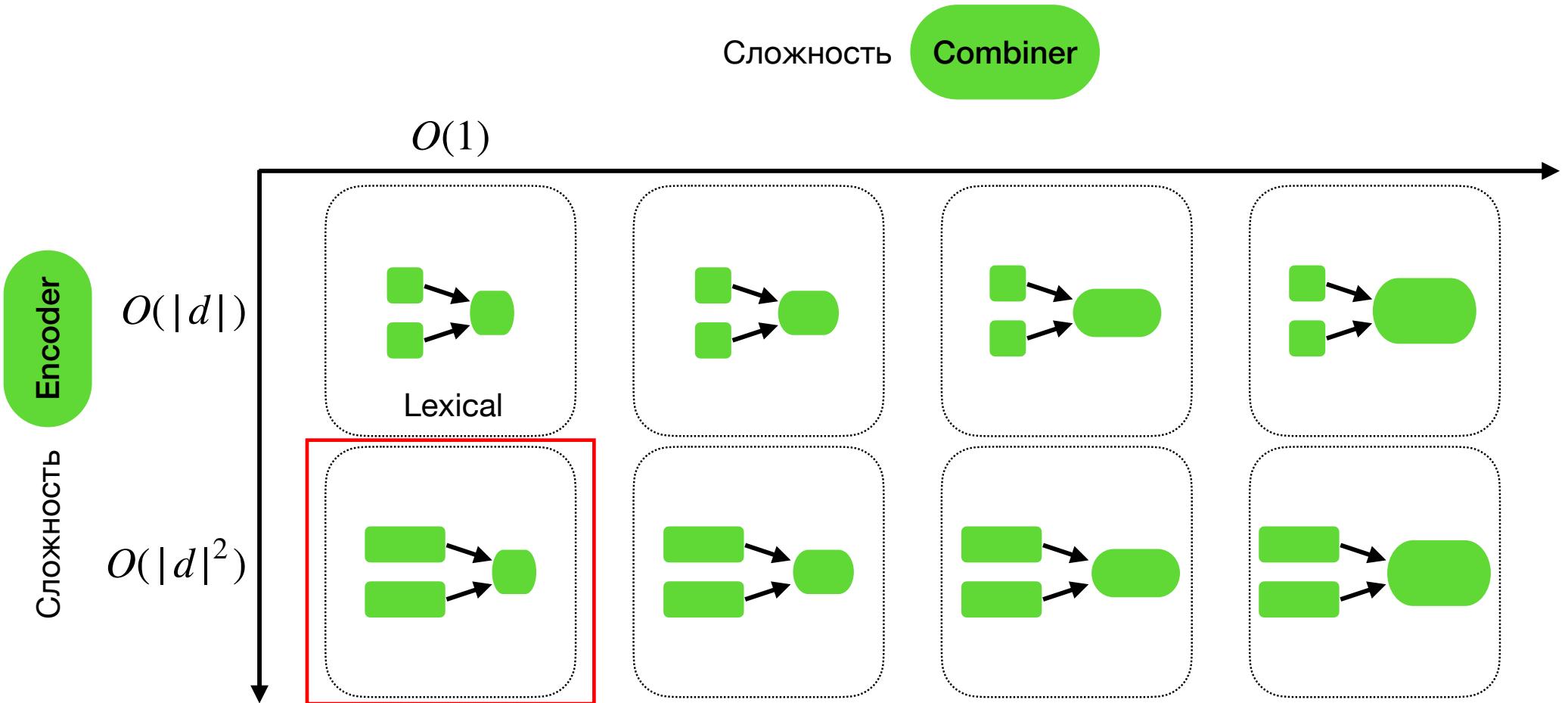
DSSM (2013, Microsoft)



DSSM (2013, Microsoft)

- Запрос и документ всё ещё представляются набором токенов (BoW)
- Эмбеддинги токенов агрегируются в единый вектор (усреднение / взвешенное усреднение / etc.), который затем прогоняется через MLP, формируя векторное представление всего текста запроса или документа
- Комбинация вычисляется как косинусная близость представлений запроса и документа
- Вариации:
 - Токенизация
 - Агрегация представлений токенов
 - Количество и размеренность слоёв сети
 - etc.

Таблица моделей релевантности

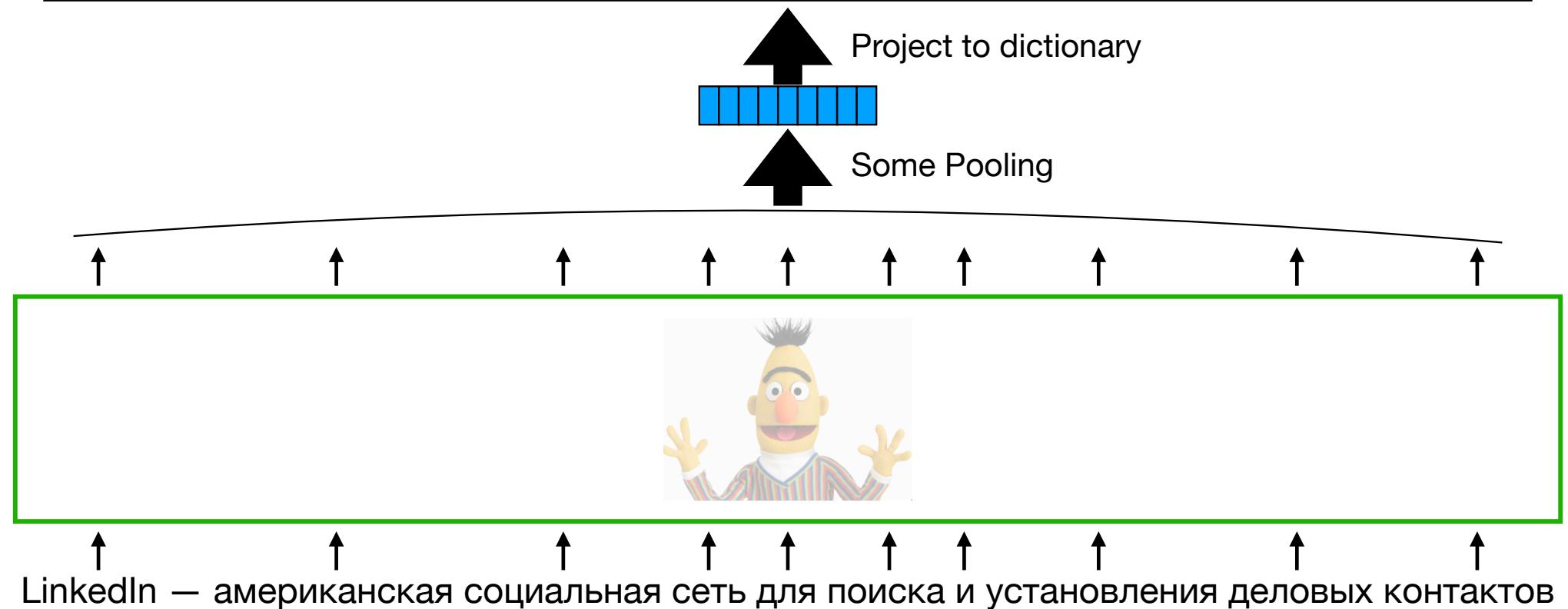


Языковые модели



Языковые модели

{«linkedin»: 2.3, «американская»: 0.8, «социальная»: 1.7, «сеть»: 1.4,
«поиска»: 1.3, «instagram»: 0.1, «регистрация»: 2.2, ...}

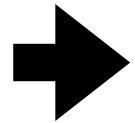
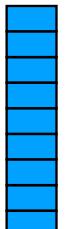


Проекция на словарь

- У нас есть словарь токенов определенного размера
- Получившийся вектор мы проецируем на словарь через обучаемую матрицу
- Таким образом, получаем распределение вероятностей (скоров) для всех токенов словаря

$$\begin{matrix} (1 \times 768) \\ \text{[blue bars]} \end{matrix} \times \begin{matrix} (768 \times 65536) \\ \text{[blue grid]} \end{matrix} = \begin{matrix} (1 \times 65536) \\ \text{[blue bars]} \end{matrix}$$

Языковые модели



{«linkedin»: 2.3, «американская»: 0.8, «социальная»: 1.7, «сеть»: 1.4,
«поиска»: 1.3, «instagram»: 0.1, «регистрация»: 2.2, ...}

Project to
dictionary



↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑
[CLS] LinkedIn – американская социальная сеть для поиска и установления деловых контактов

Learned Sparse Retrieval (LSR) Модели

- Представление обучаемое и проецируемое на словарь токенов
- Комбинация основывается на пересечении наборов токенов
- Вариации:
 - Взвешивание токенов
 - Токенизация
 - Разреживание (top-K, регуляризация)
 - etc.
- Модели:
 - DeepCT
 - EPIC
 - SPLADE
 - TILDE

DeepCT

Deep Contextualized Term Weighting

- Языковая модель используется только для оценки важности (весов) токенов
- Важность токенов хранится в обратном индексе вместо term frequency

DeepCT

Inverted Index scores



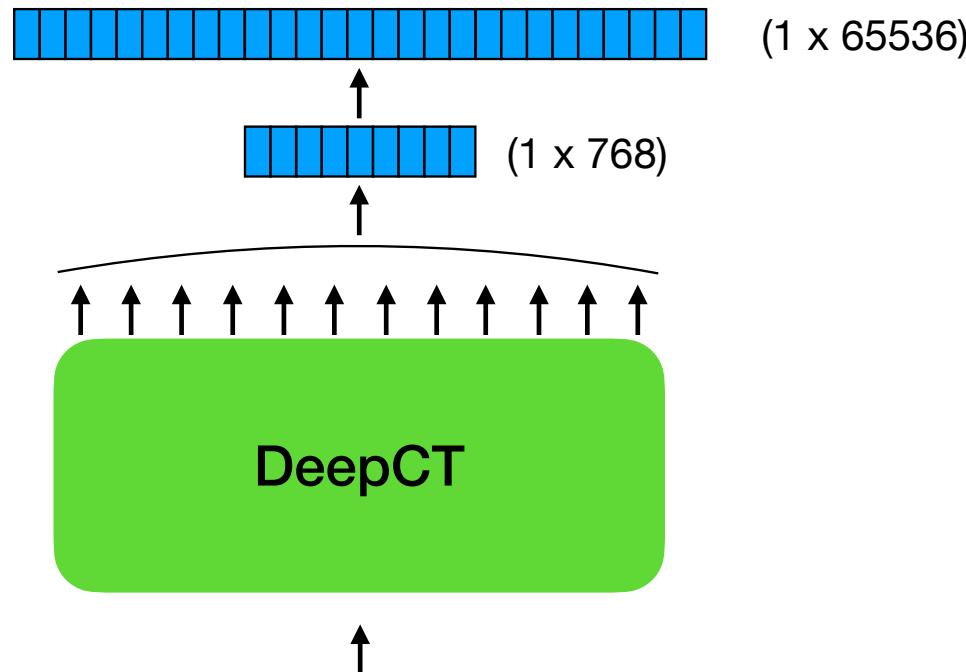
```
{«linkedin»: 2.3, «американская»: 0.8, «социальная»: 1.7, «сеть»: 1.4,  
«поиска»: 1.3, «установления»: 0.1, «контактов»: 2.2, ...}
```



LinkedIn – американская социальная сеть для поиска и установления деловых контактов

DeepCT

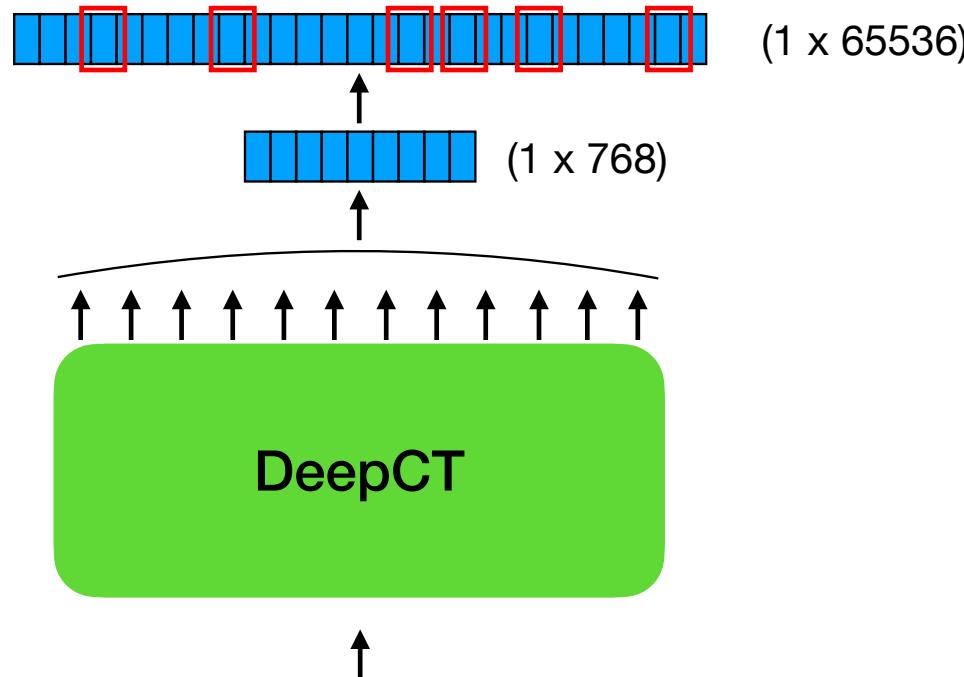
Преобразование обратного индекса



LinkedIn — американская социальная сеть для поиска и установления деловых контактов

DeepCT

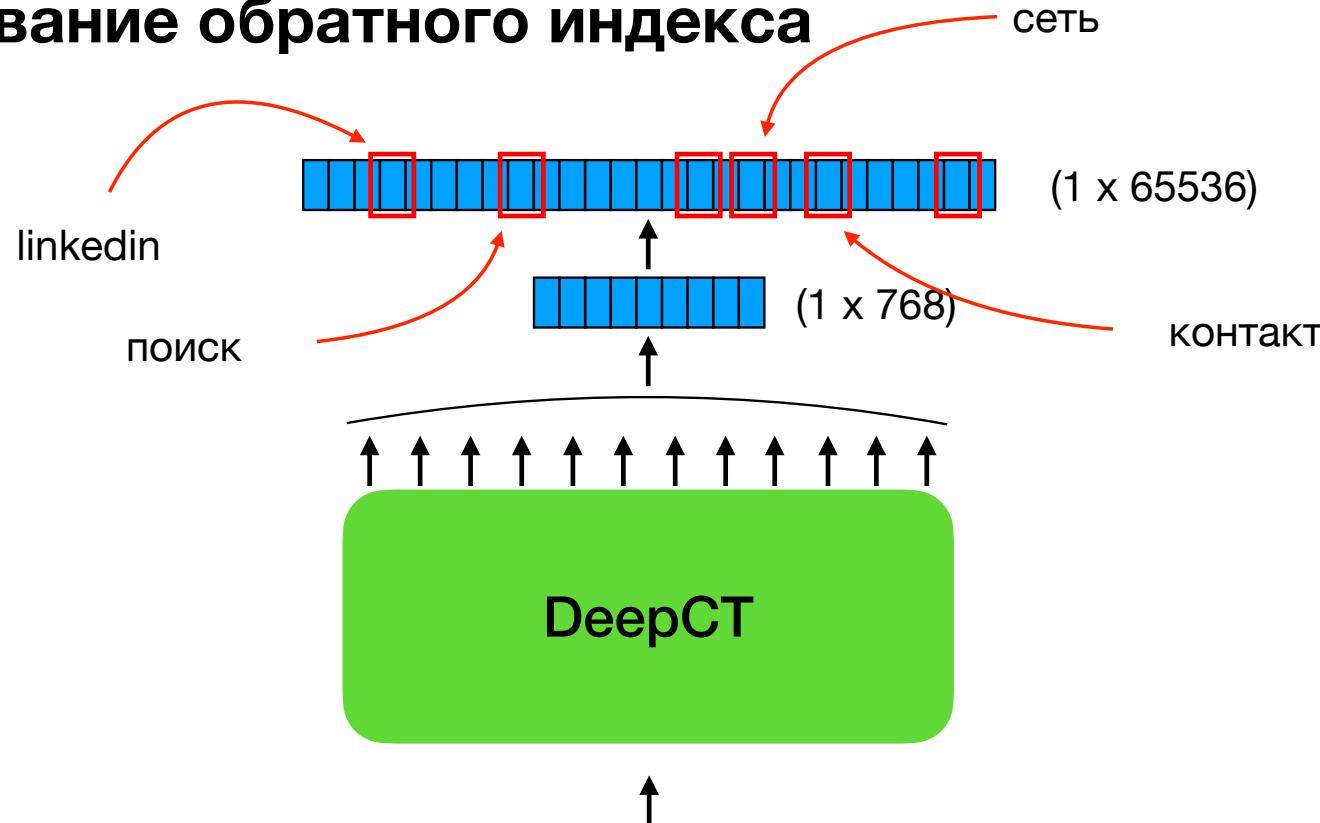
Преобразование обратного индекса



LinkedIn — американская социальная сеть для поиска и установления деловых контактов

DeepCT

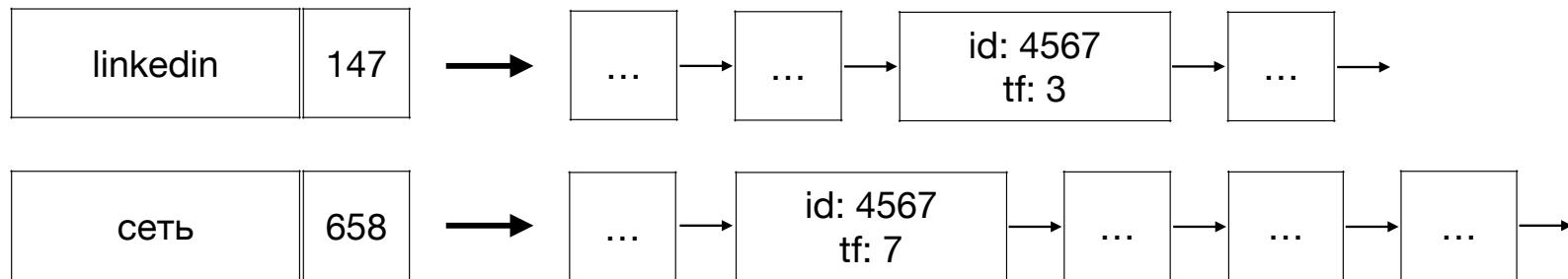
Преобразование обратного индекса



LinkedIn — американская социальная сеть для поиска и установления деловых контактов

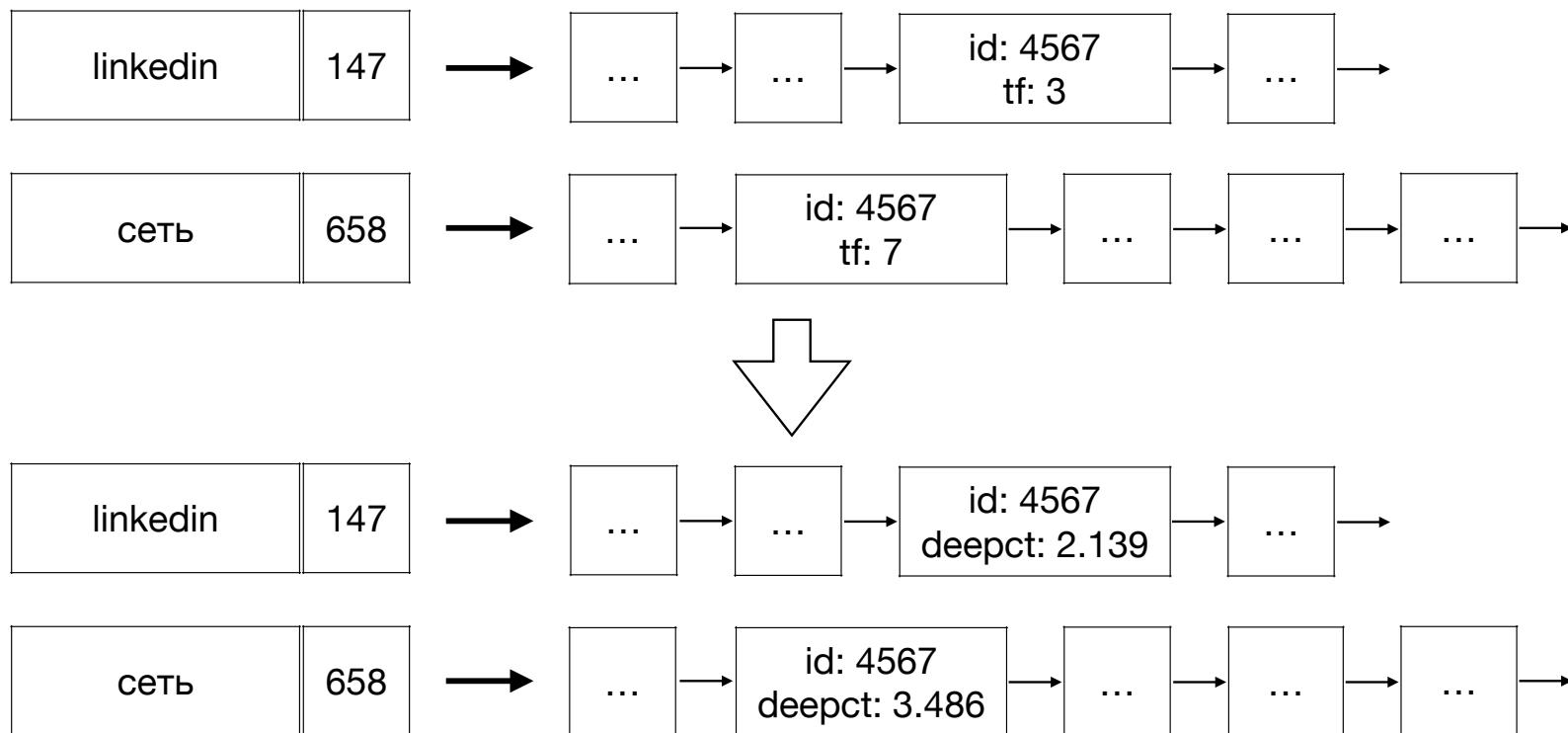
DeepCT

Преобразование обратного индекса



DeepCT

Преобразование обратного индекса



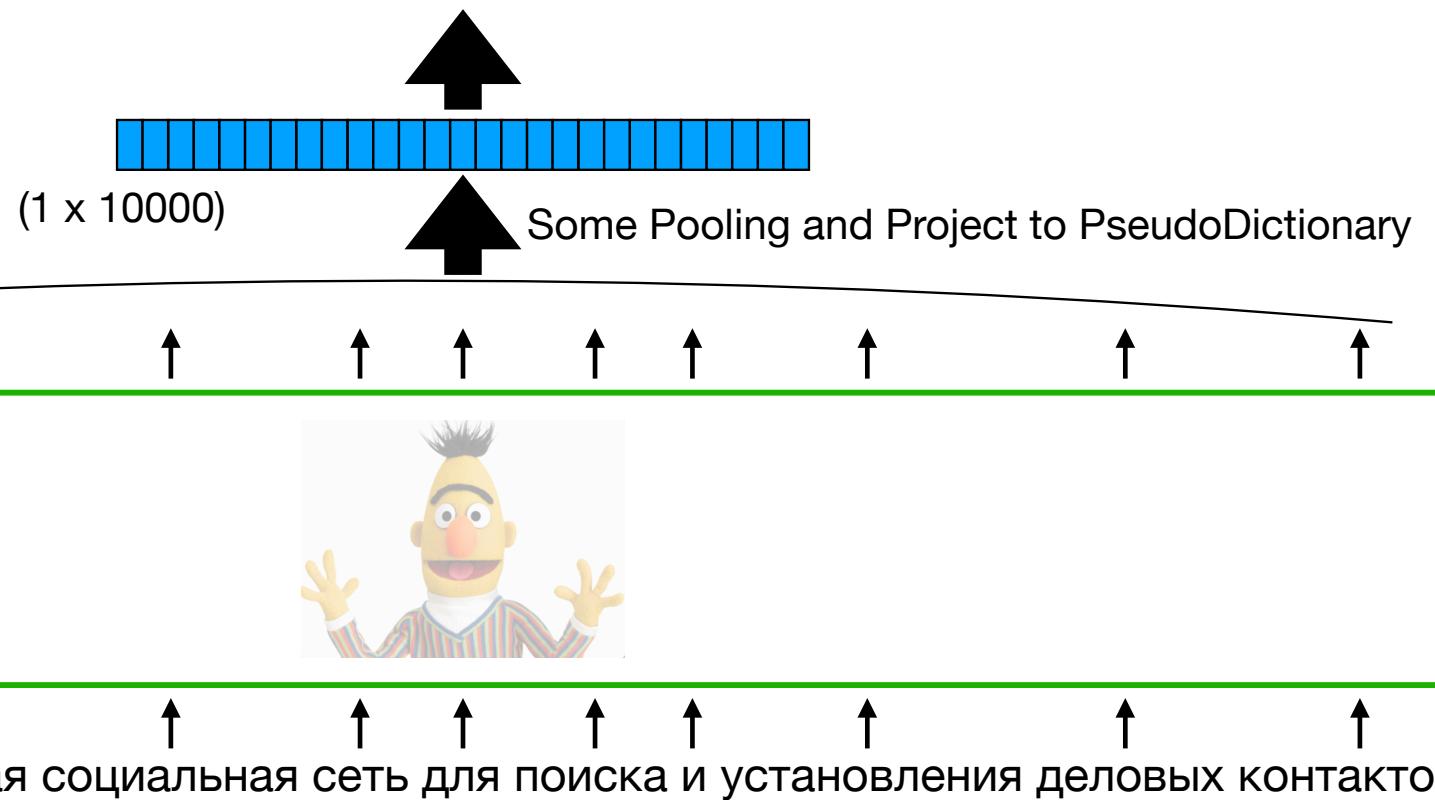
SNRM

Standalone Neural Ranking Model

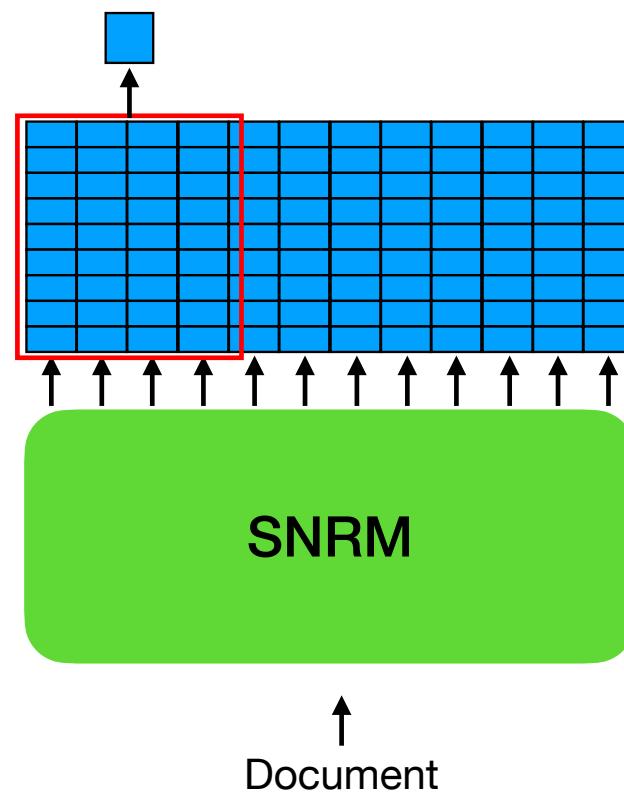
- Модель, кодирующая отдельно запрос и документ
- Представления — в пространстве «псевдотокенов»
- По представлению текста в пространстве псевдотокенов строится обратный индекс
- Весами токенов словаря являются выходы языковой модели
- Обработка запроса — представить запрос в пространстве псевдотокенов и получить по ним документы из обратного индекса

SNRM

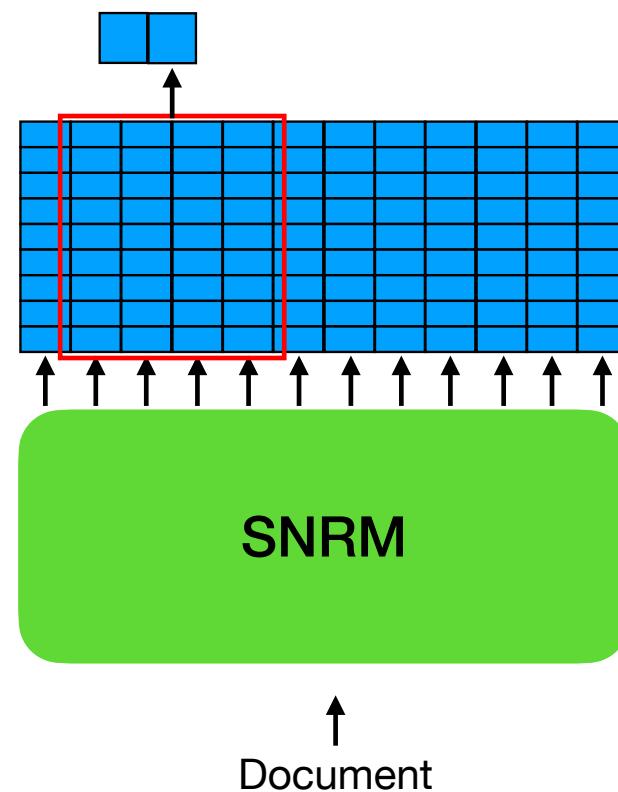
Construct Inverted Index on PseudoDictionary tokens



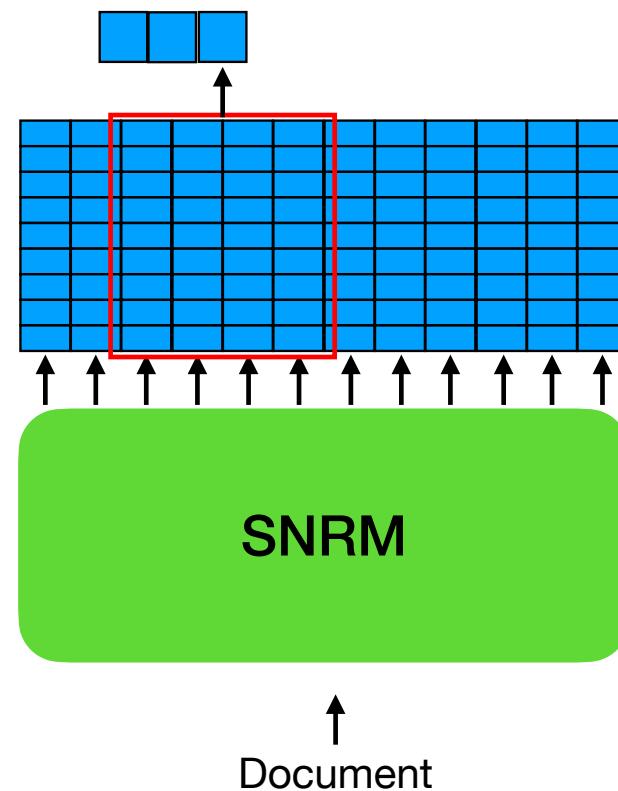
Convolution Pooling



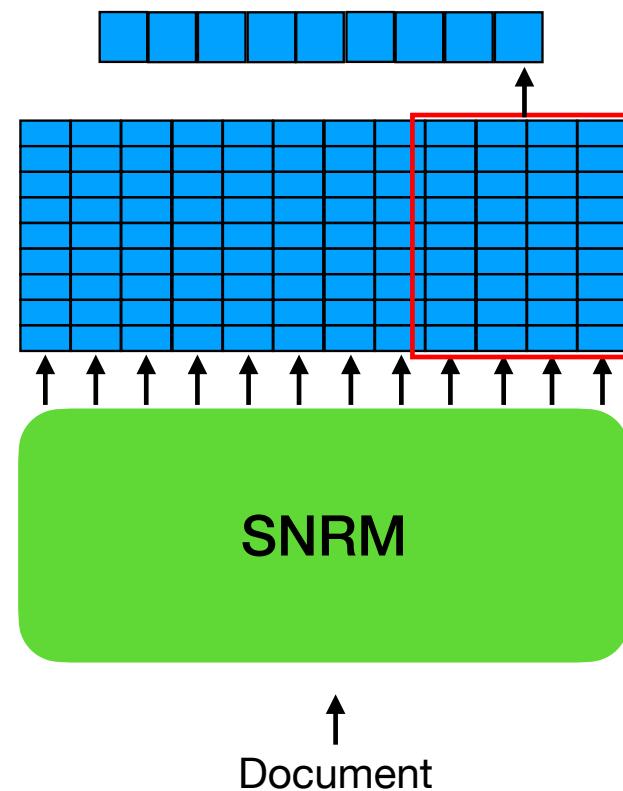
Convolution Pooling



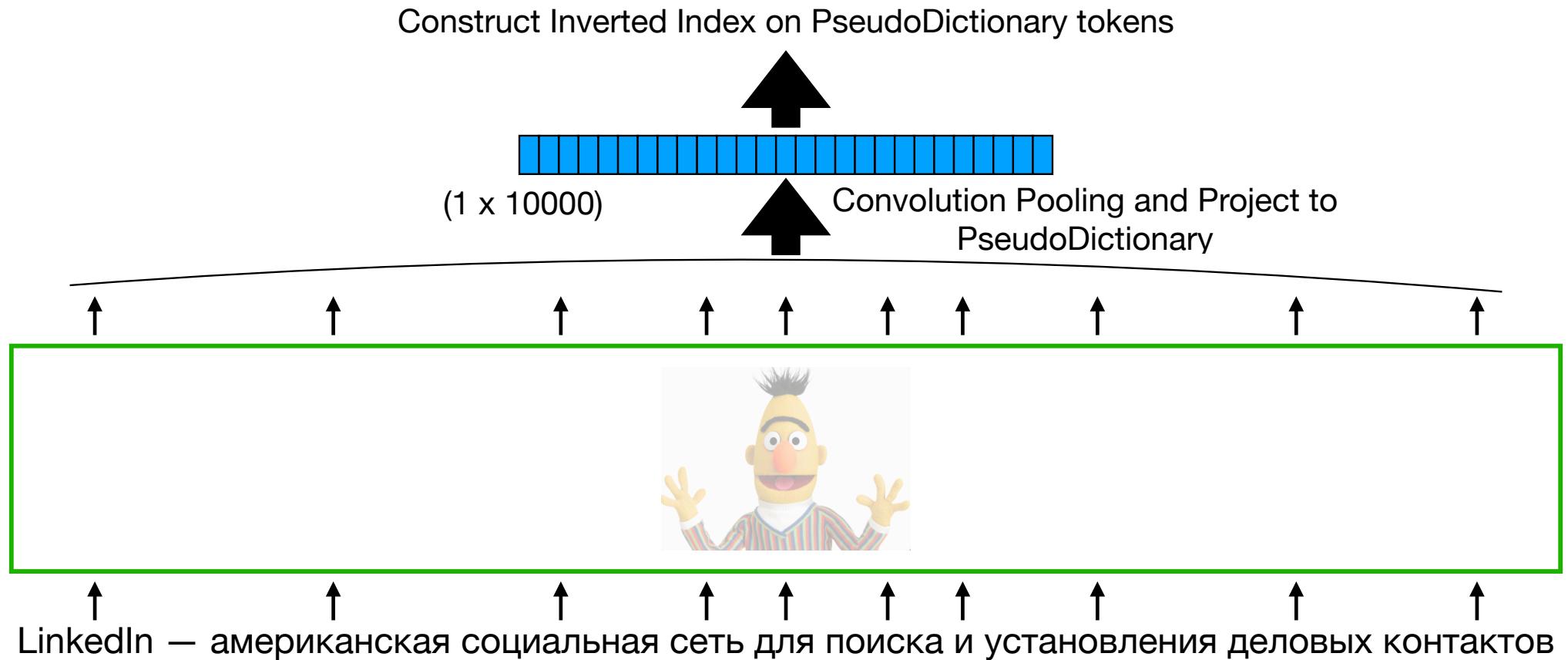
Convolution Pooling



Convolution Pooling



SNRM



SNRM

Получается, каждый документ будет
содержать все псевдотокены словаря?

Construct Inverted Index on PseudoDictionary tokens



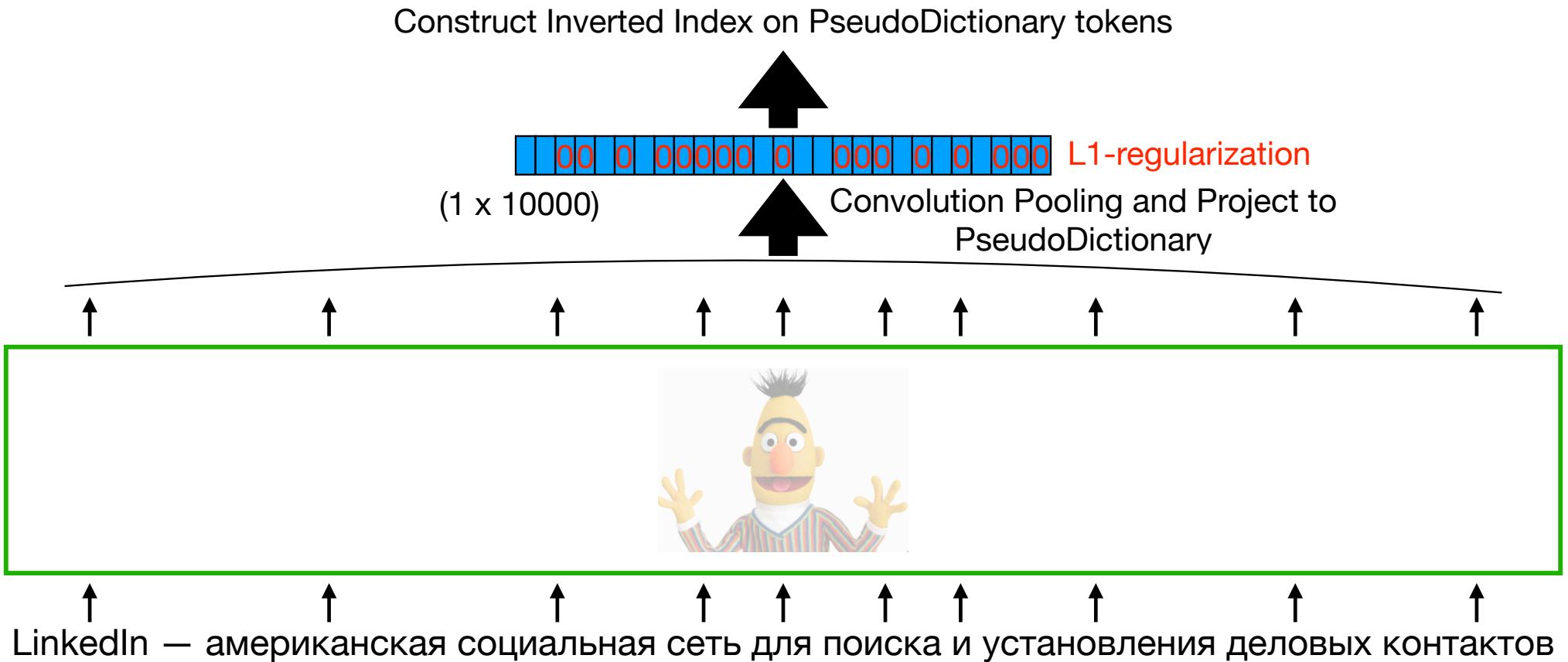
(1 x 10000)

Convolution Pooling and Project to
PseudoDictionary



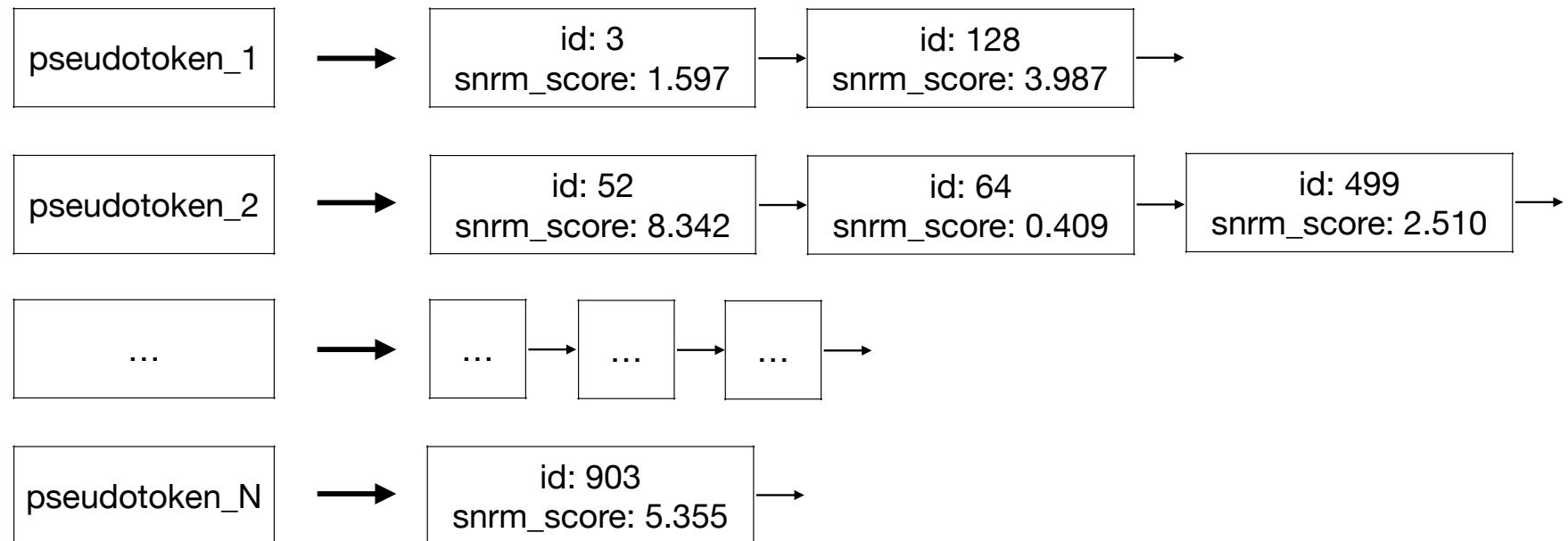
LinkedIn – американская социальная сеть для поиска и установления деловых контактов

SNRM

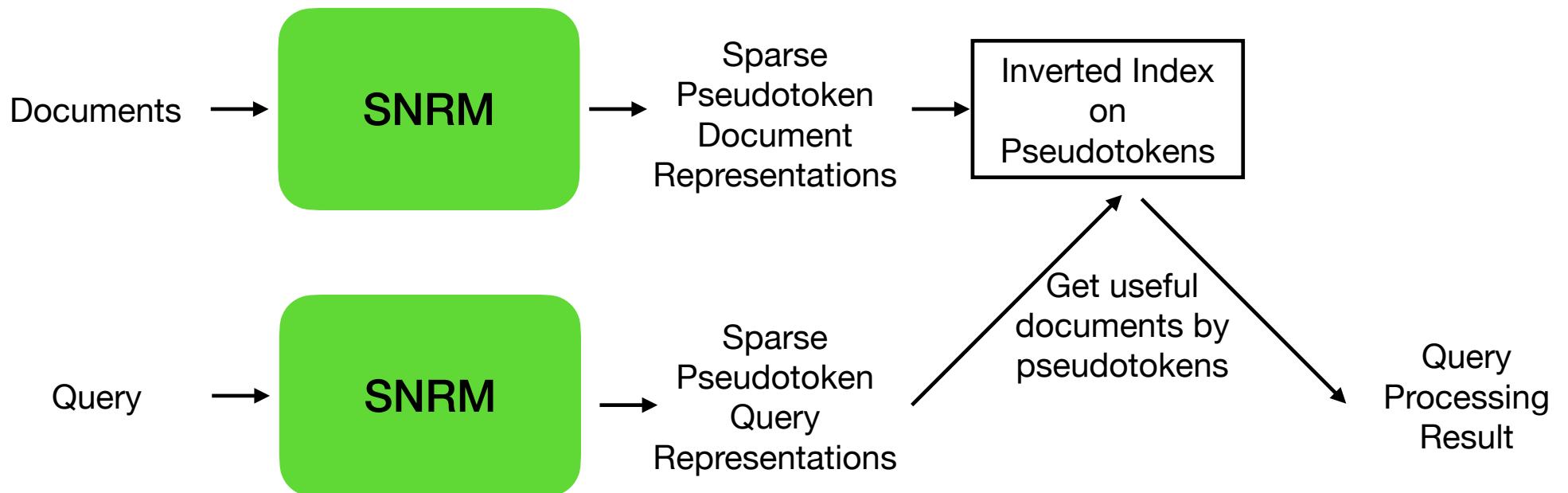


SNRM

Inverted Index on PseudoDictionary tokens



SNRM

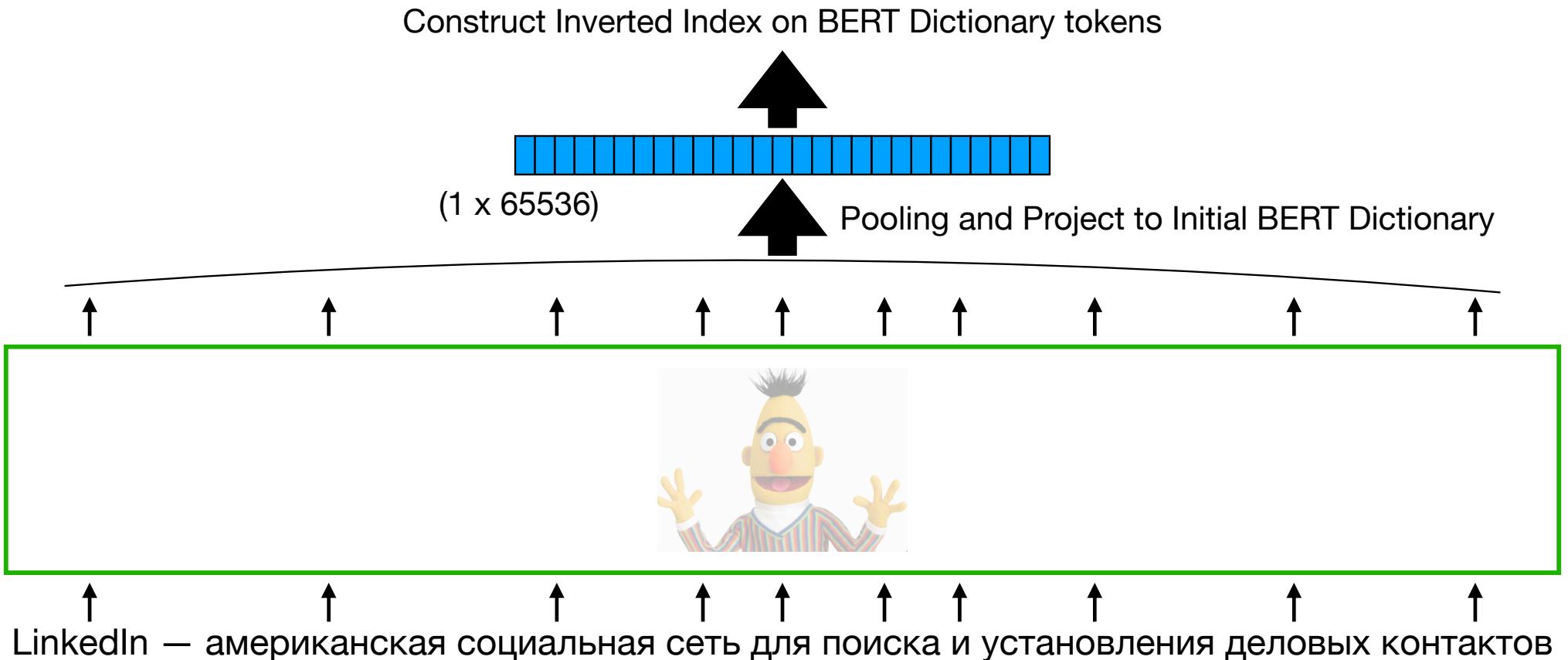


SparTerm

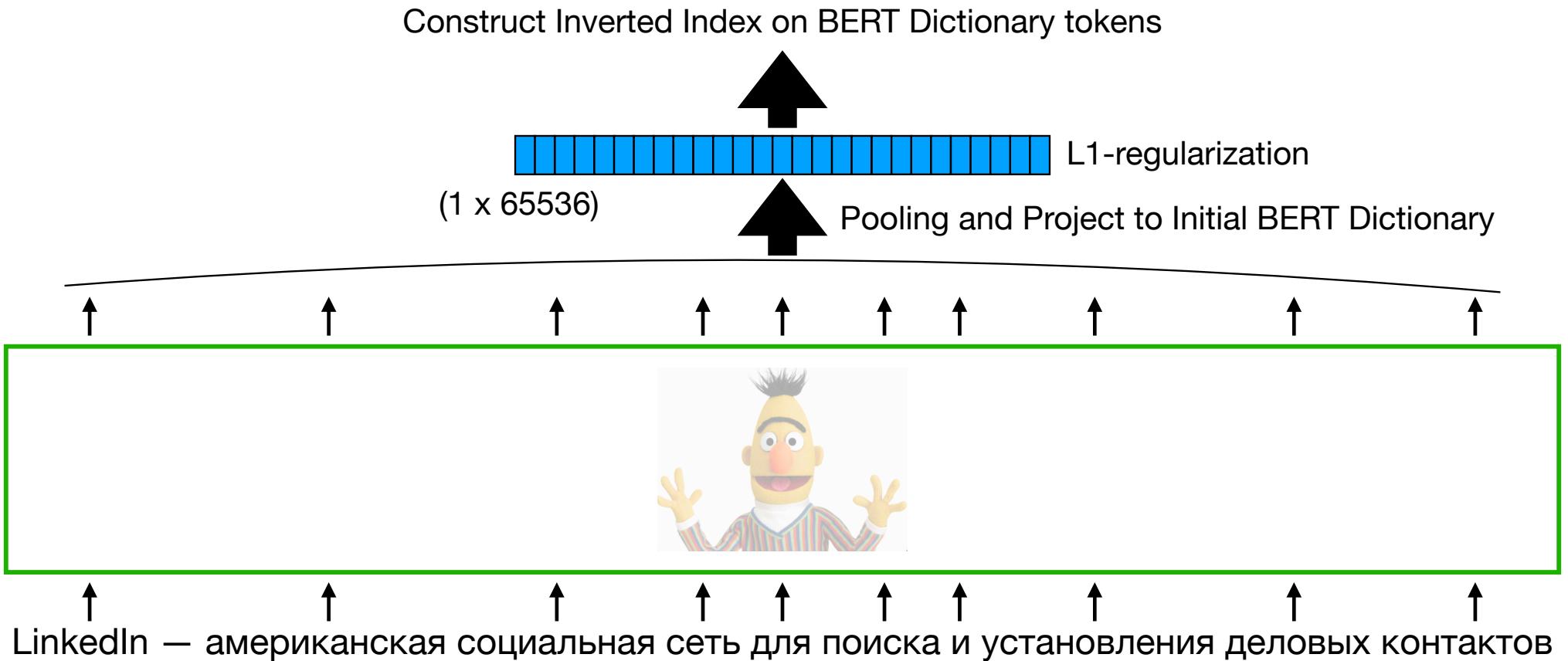
Term-based Sparse representations

- В отличие от SNRM, в этой модели представления текста — не в пространстве псевдотокенов, а в пространстве токенов BERT-а

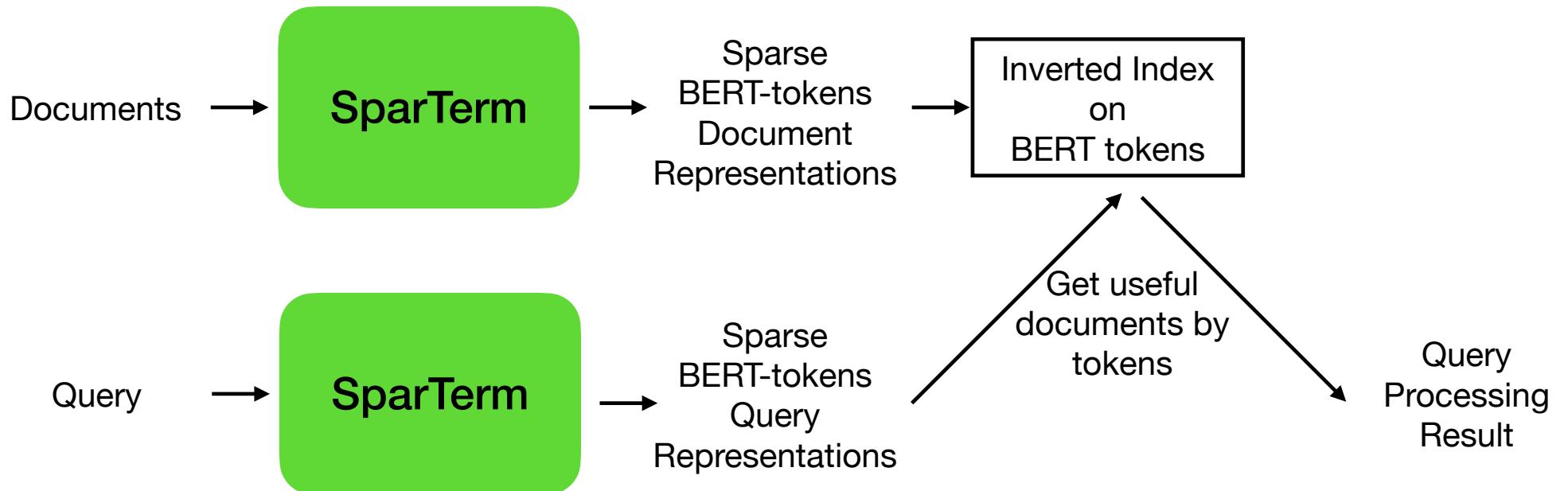
SparTerm



SparTerm



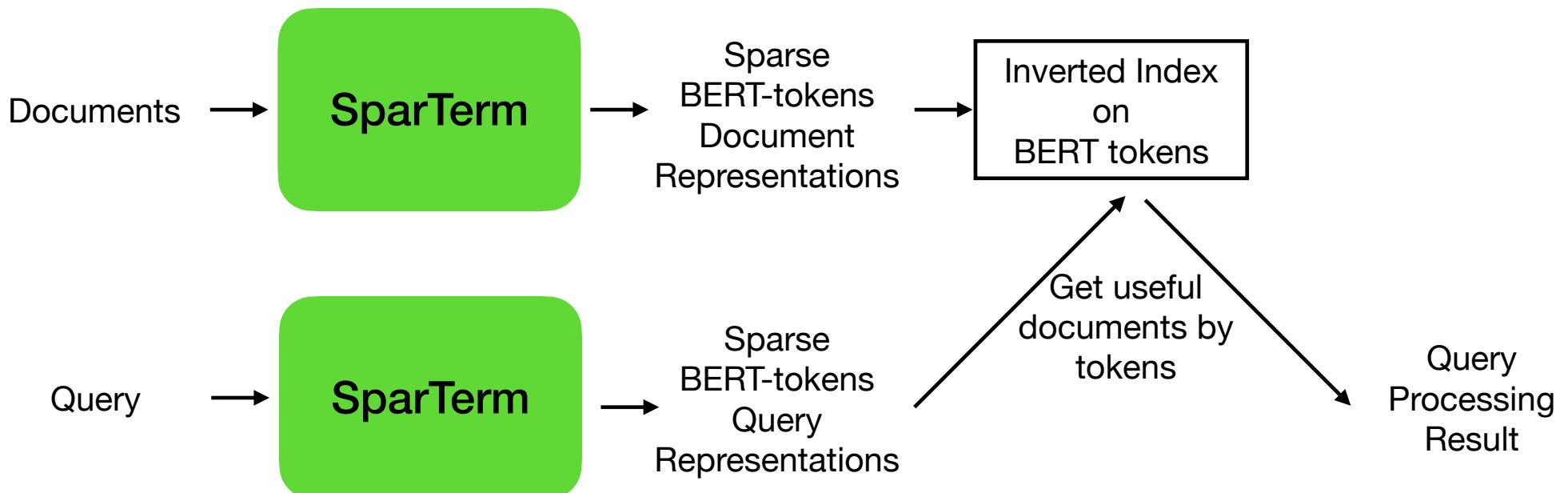
SparTerm



SPADE

System for Prompt Analysis and Delta-Based Evaluation

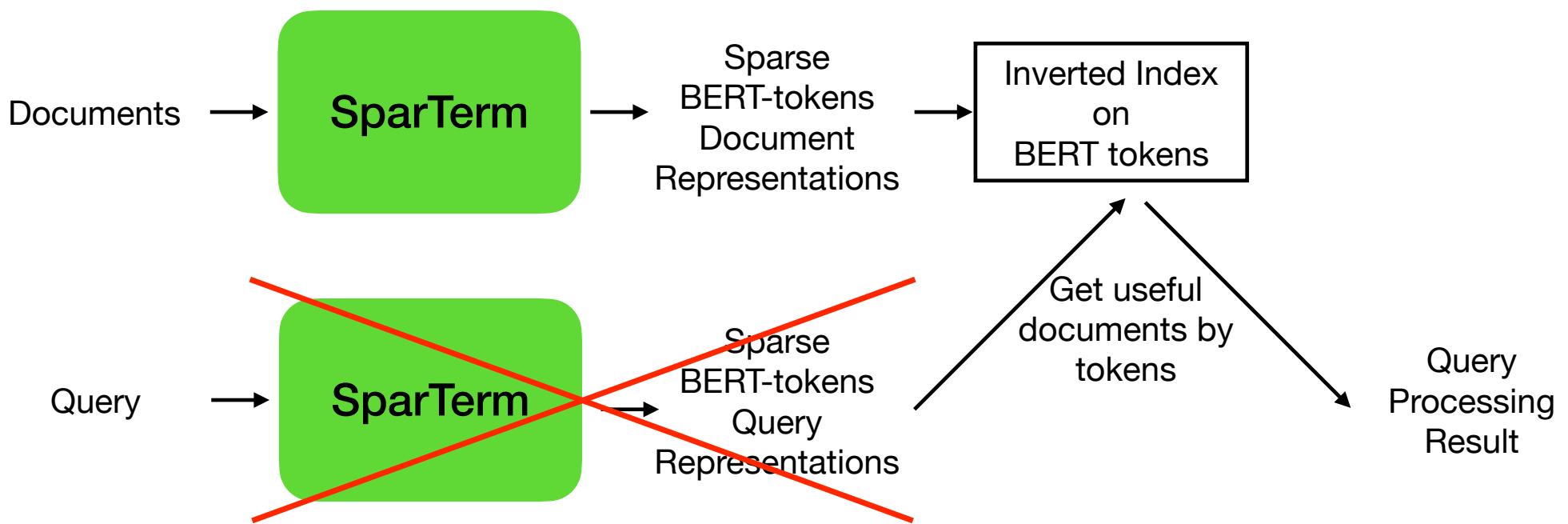
- Основная проблема предыдущих моделей — нужно инференсить большую языковую модель в рантайме (чтобы закодировать запрос)



SPADE

System for Prompt Analysis and Delta-Based Evaluation

- Основная проблема предыдущих моделей — нужно инференсить большую языковую модель в рантайме (чтобы закодировать запрос)



SPADE

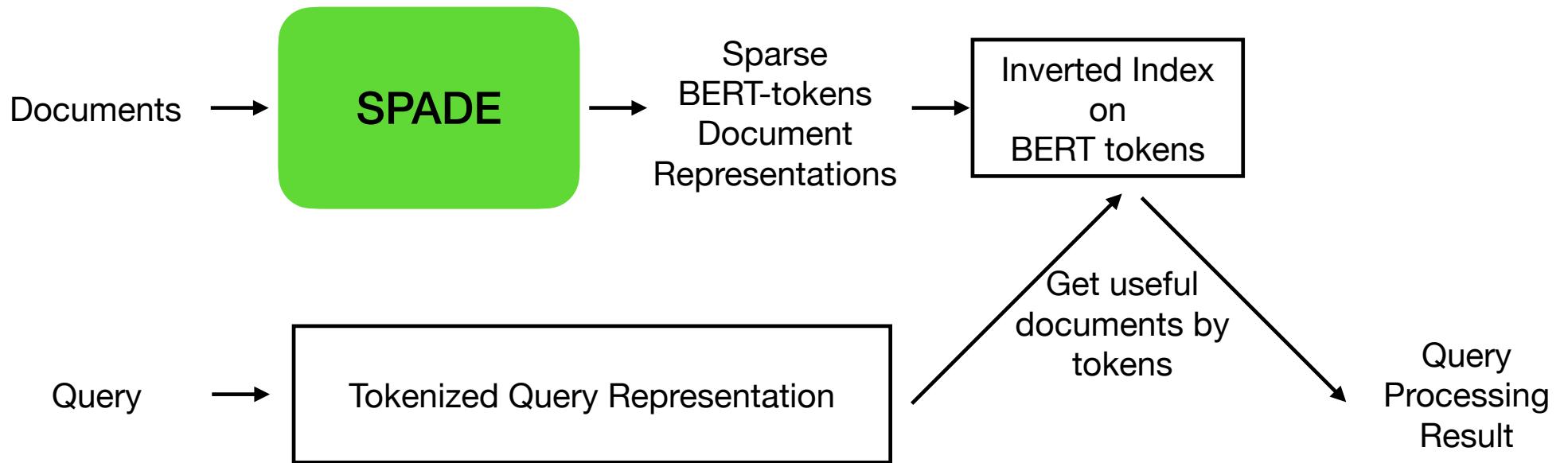
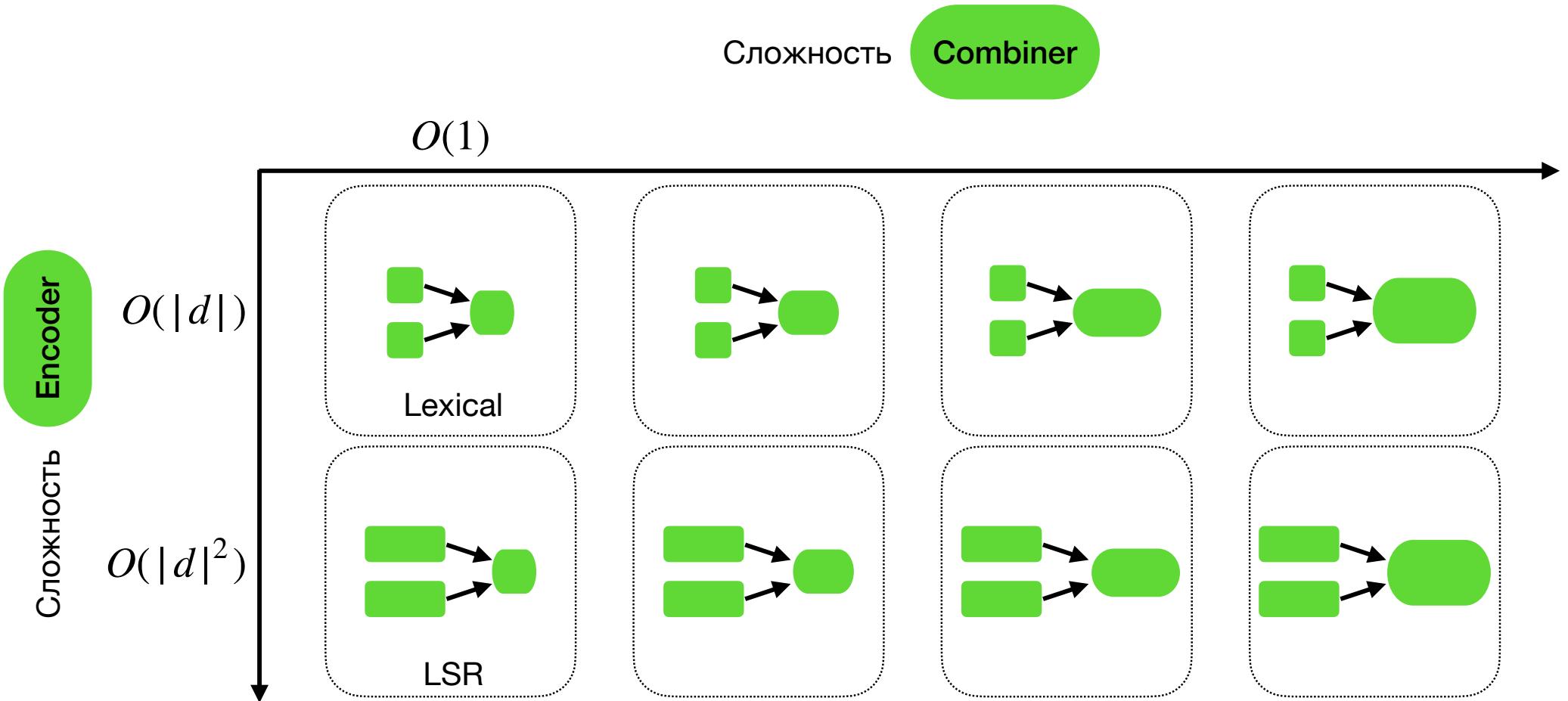
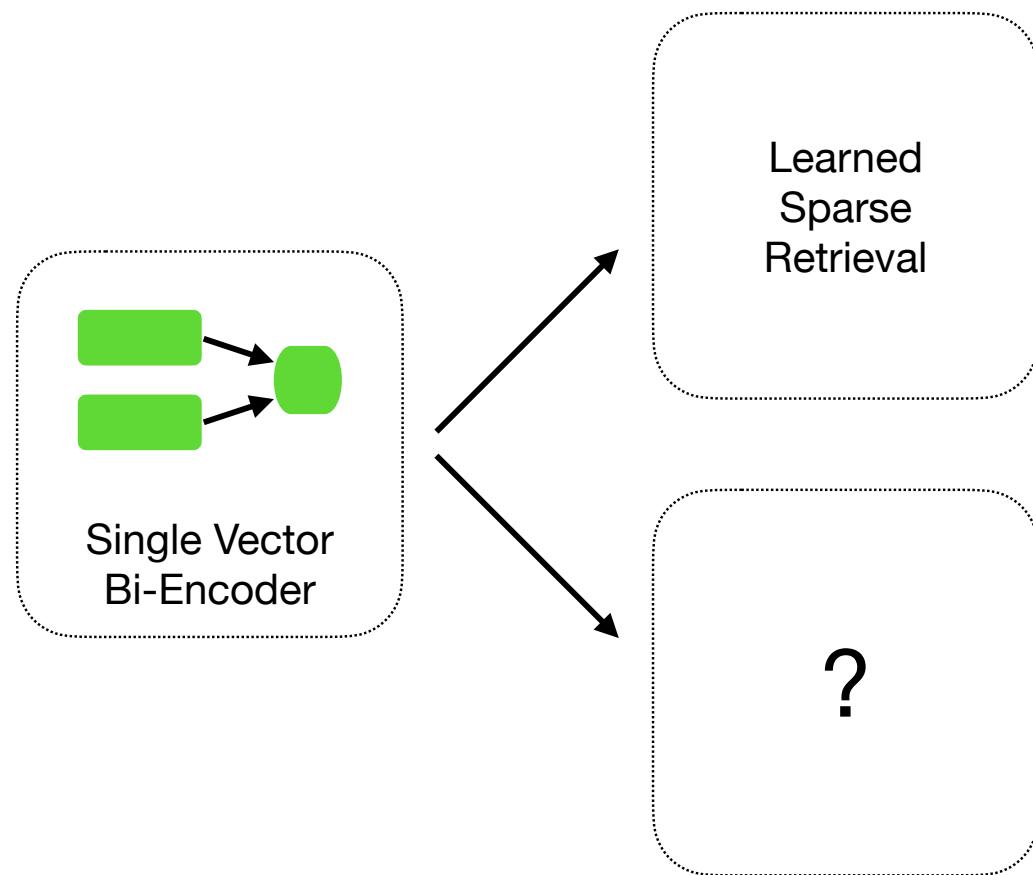


Таблица моделей релевантности



Single Vector Bi-Encoders

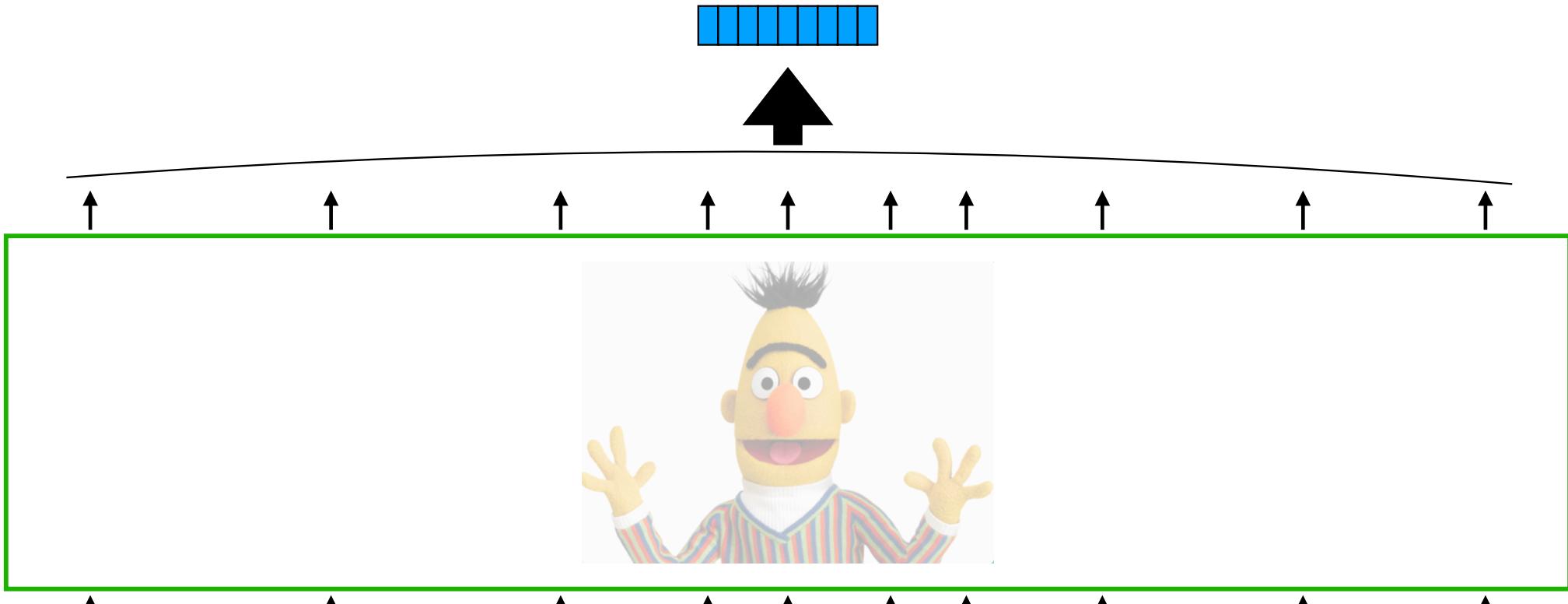


Языковые модели



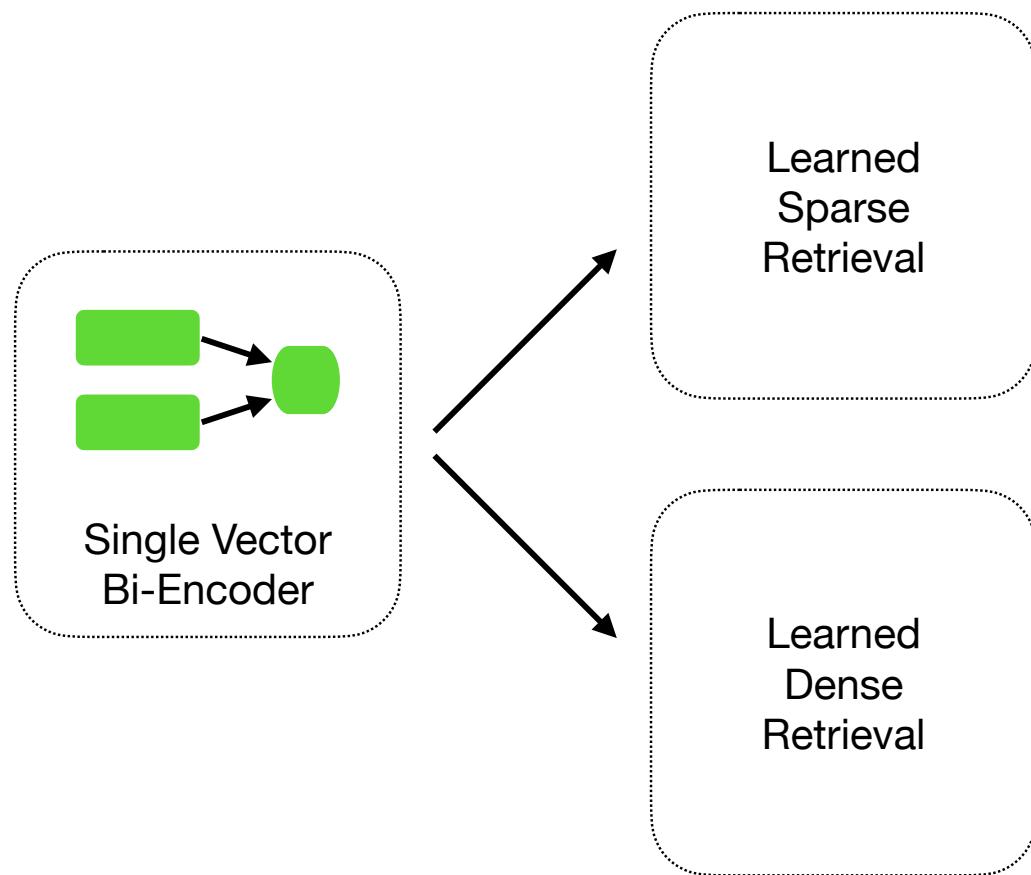
[CLS] LinkedIn – американская социальная сеть для поиска и установления деловых контактов

Языковые модели

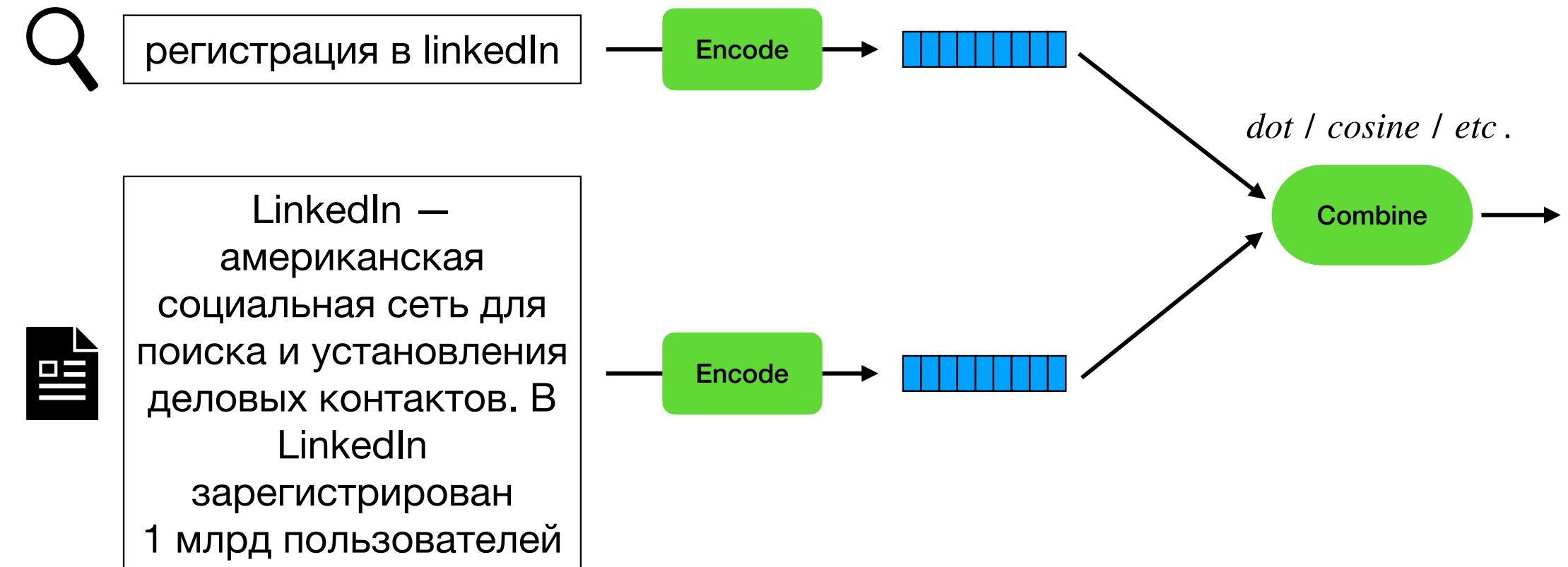


LinkedIn — американская социальная сеть для поиска и установления деловых контактов

Single Vector Bi-Encoders



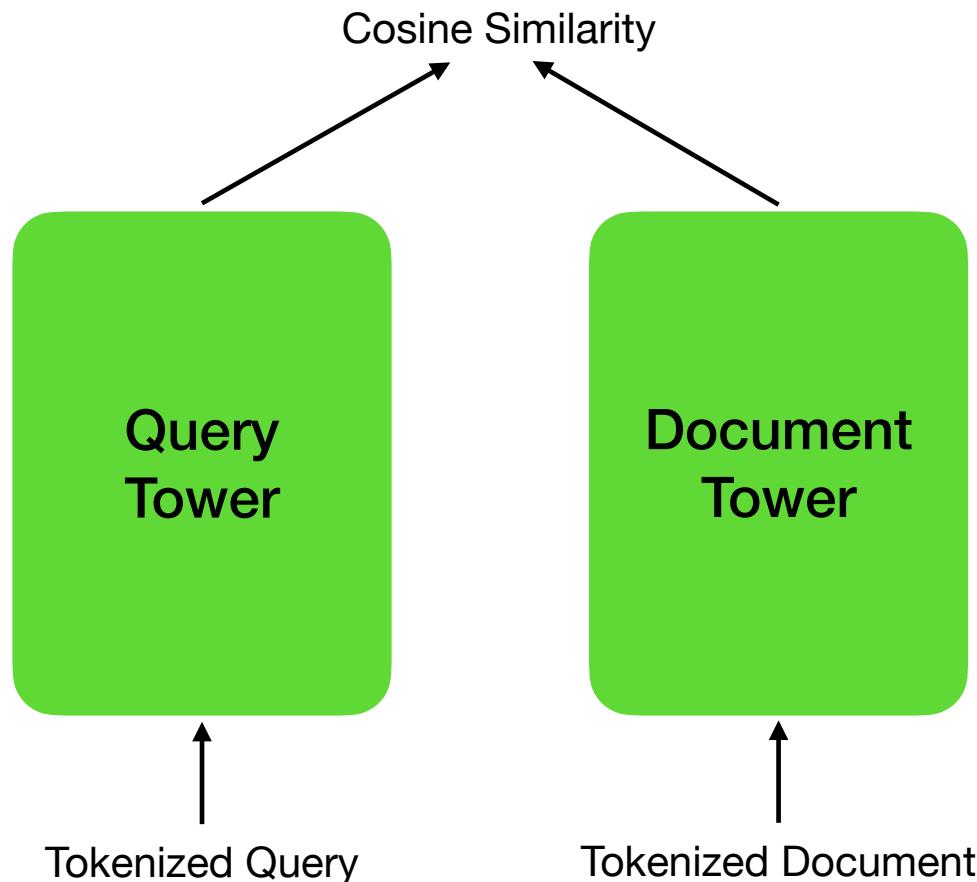
Single-Vector Dense Retrieval



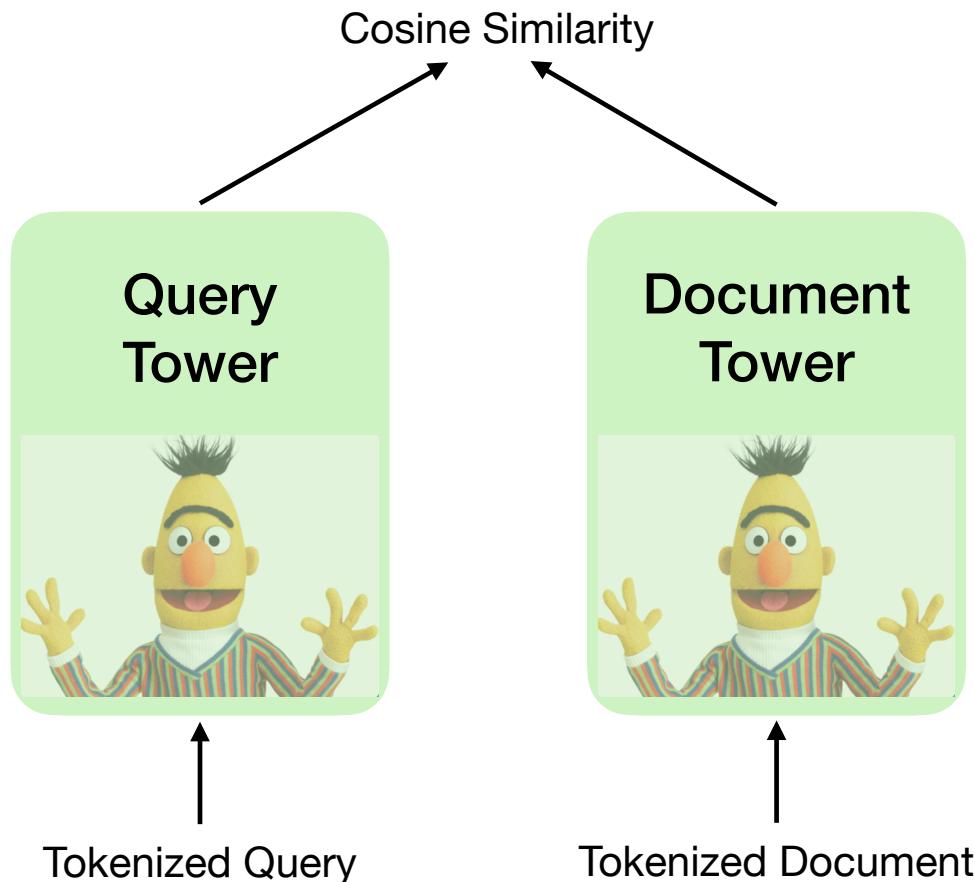
Single-Vector Dense Retrieval

- Представление обучаемым вектором фиксированного размера (обычно размеренности ~100-2000)
- Комбинация основывается на близости между векторами
- Вариации:
 - Pooling (CLS / mean pooling / etc.)
 - Комбинация (dot product / cosine / etc.)
 - Симметричные / асимметричные энкодеры запроса и документа
 - etc.
- Модели:
 - DSSM with BERT
 - DPR
 - TCT-CoBERT

DSSM with BERT



DSSM with BERT



Single-Vector Dense Retrieval

А как применять?

- В оффлайне кодируем каждый документ и складываем векторы в ANN-индекс
- В онлайне кодируем запрос пользователя и получаем вектор
- Производим векторный поиск и находим документы, близкие к запросу

Таблица моделей релевантности

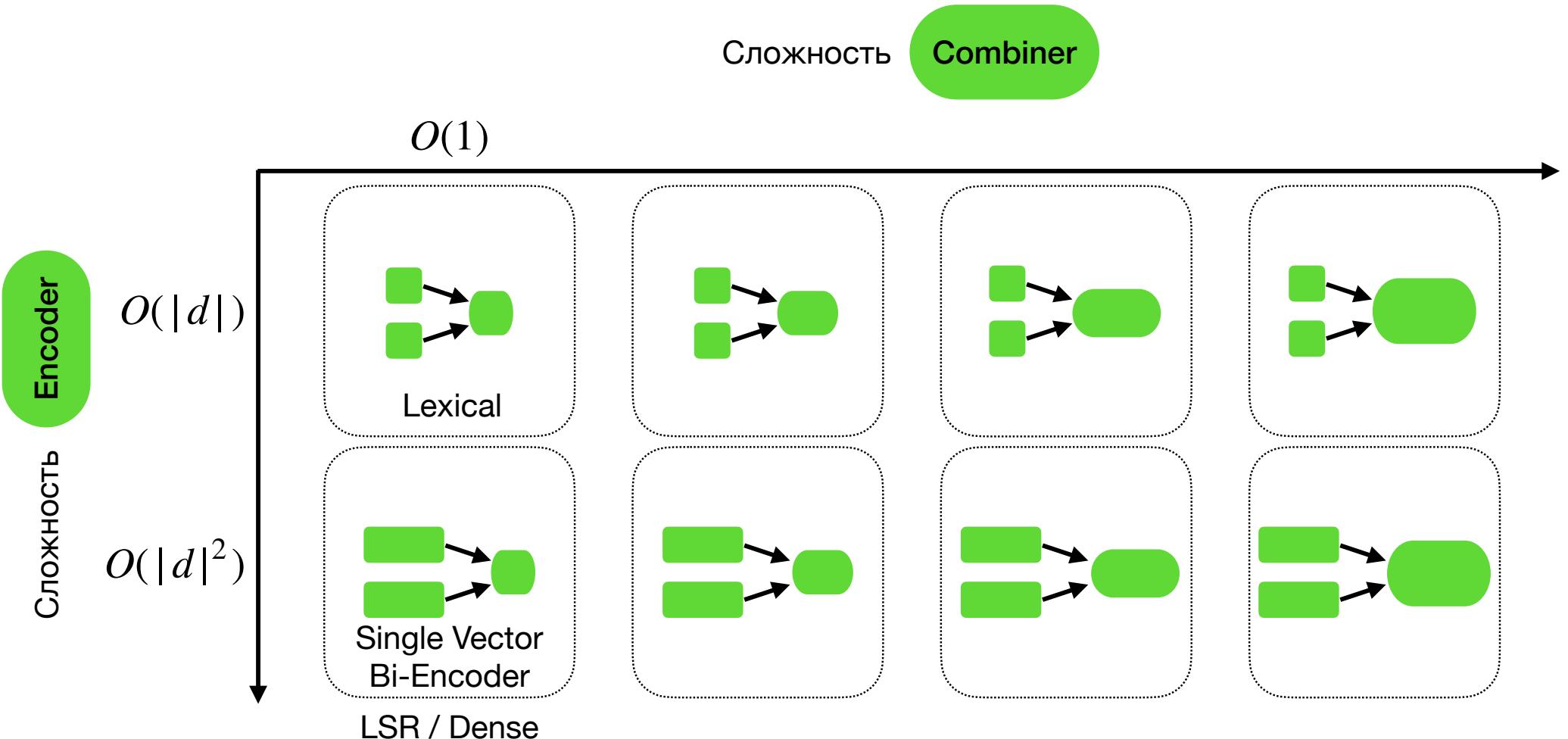
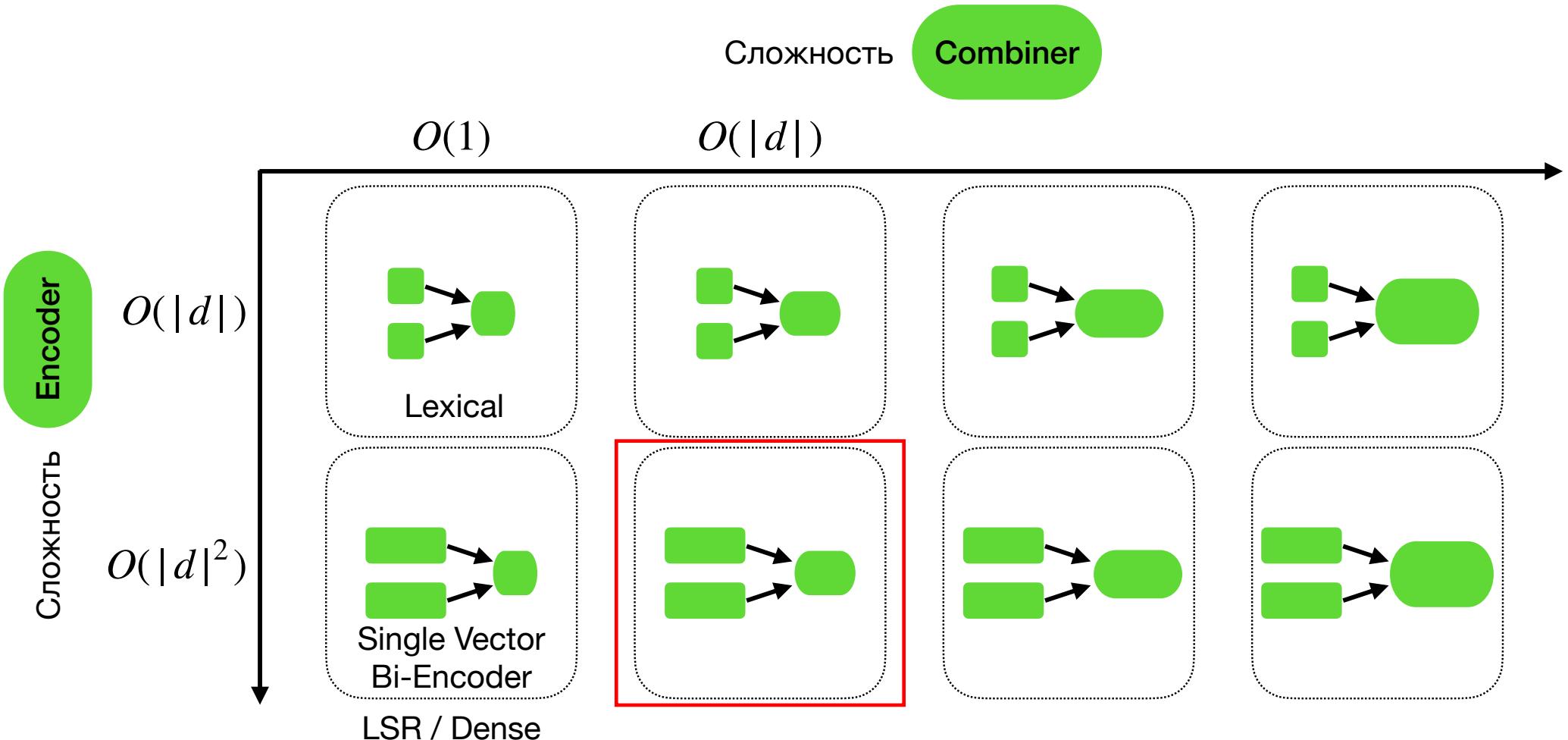
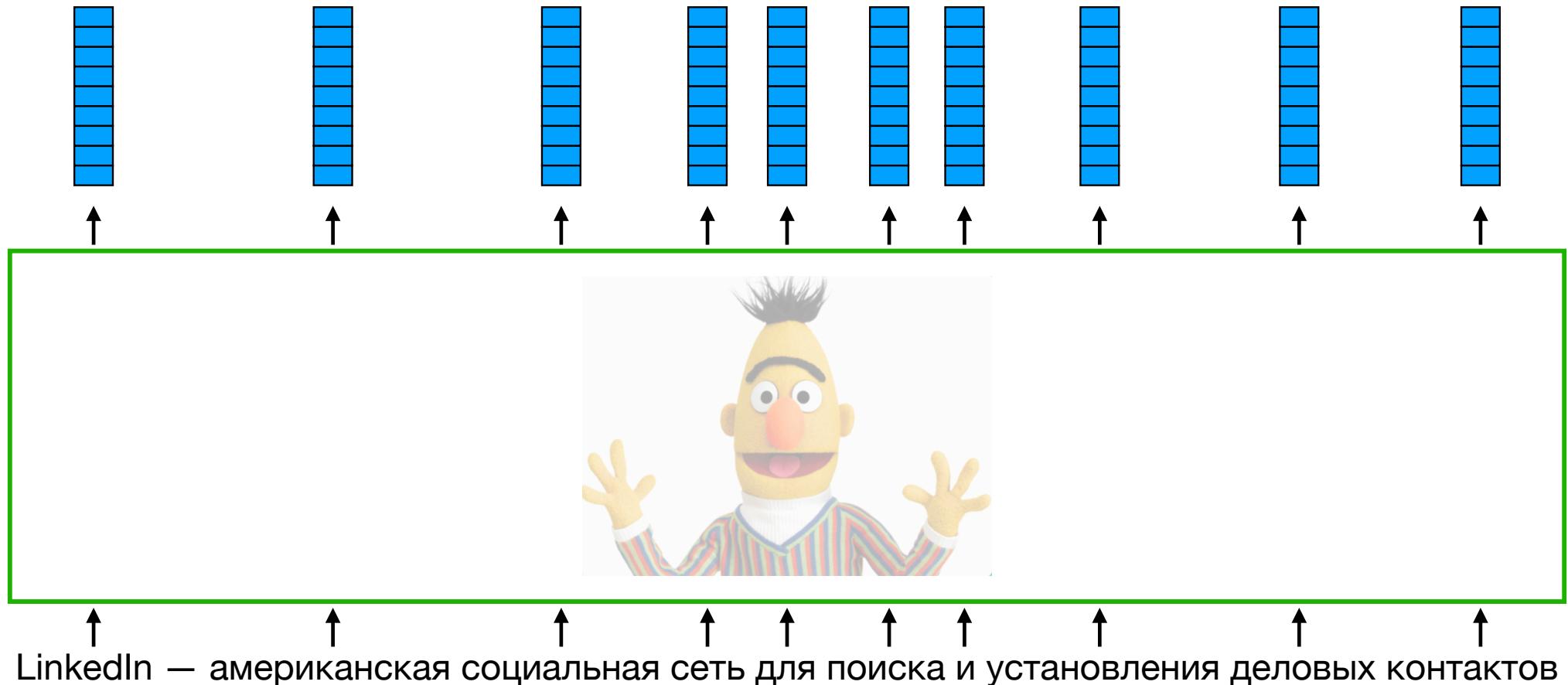


Таблица моделей релевантности



Multi-Vector Encoders

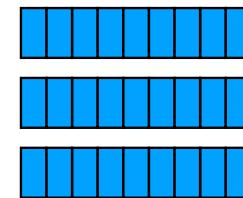


Multi-Vector Encoders



регистрация в linkedIn

Encode



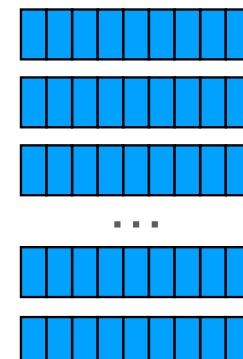
?

Combine

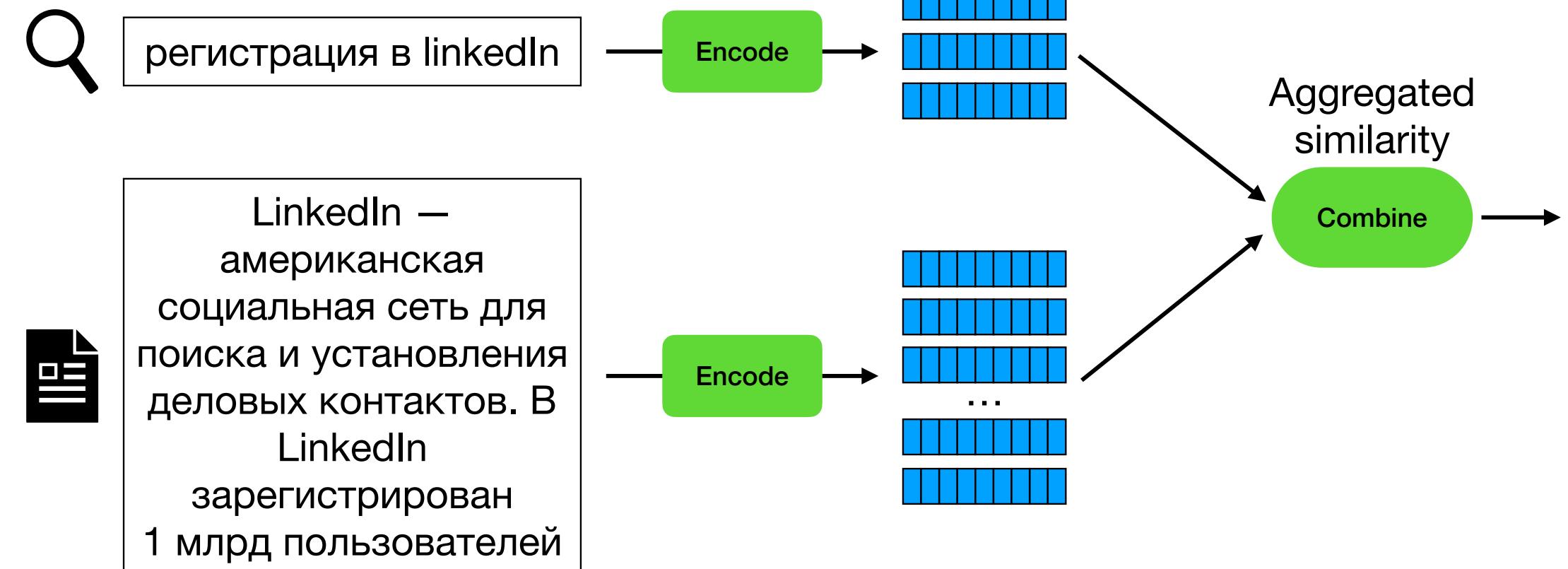


LinkedIn –
американская
социальная сеть для
поиска и установления
деловых контактов. В
LinkedIn
зарегистрирован
1 млрд пользователей

Encode



Multi-Vector Encoders

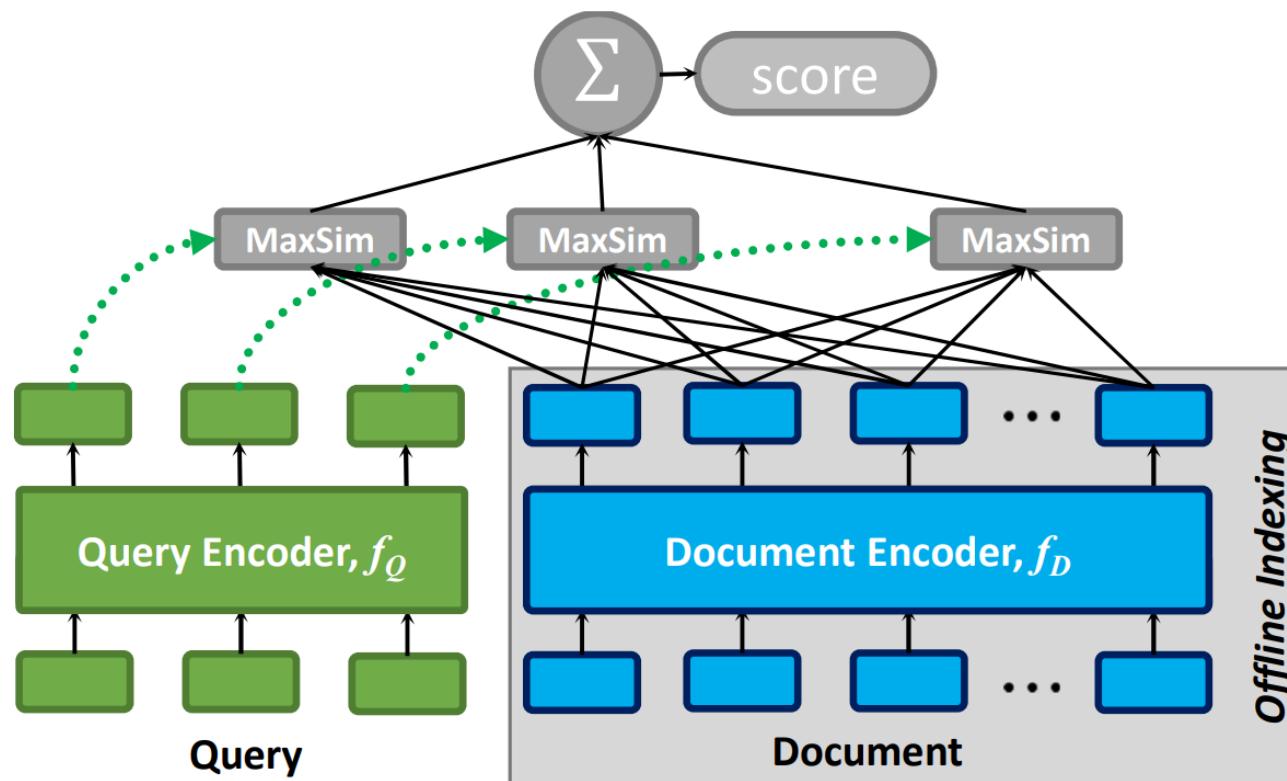


(Late) Interaction Models

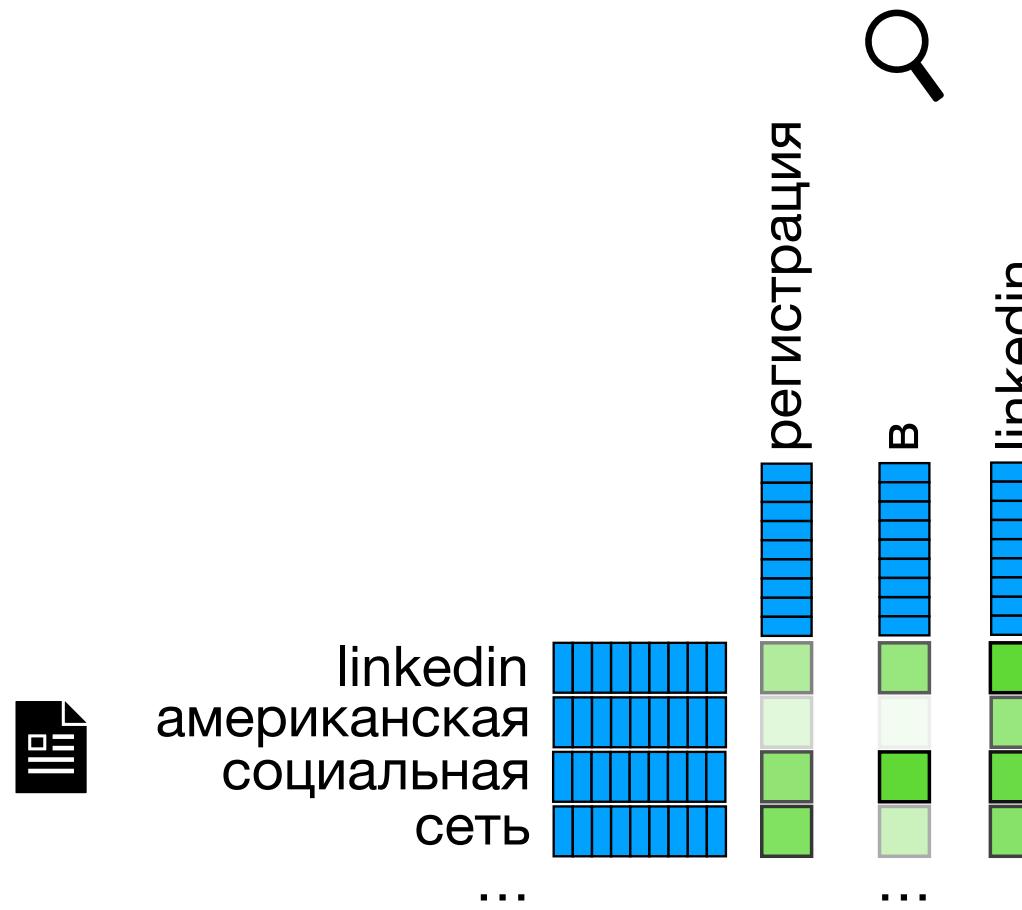
- Представление токенов обучаемыми векторами фиксированного размера
- Комбинация основывается на легковесной комбинации близостей токенов
- Вариации:
 - Комбинация
 - Стратегия кодирования
 - etc.
- Модели:
 - CoBERT
 - DRMM
 - KNRM
 - PACRR

CoBERT

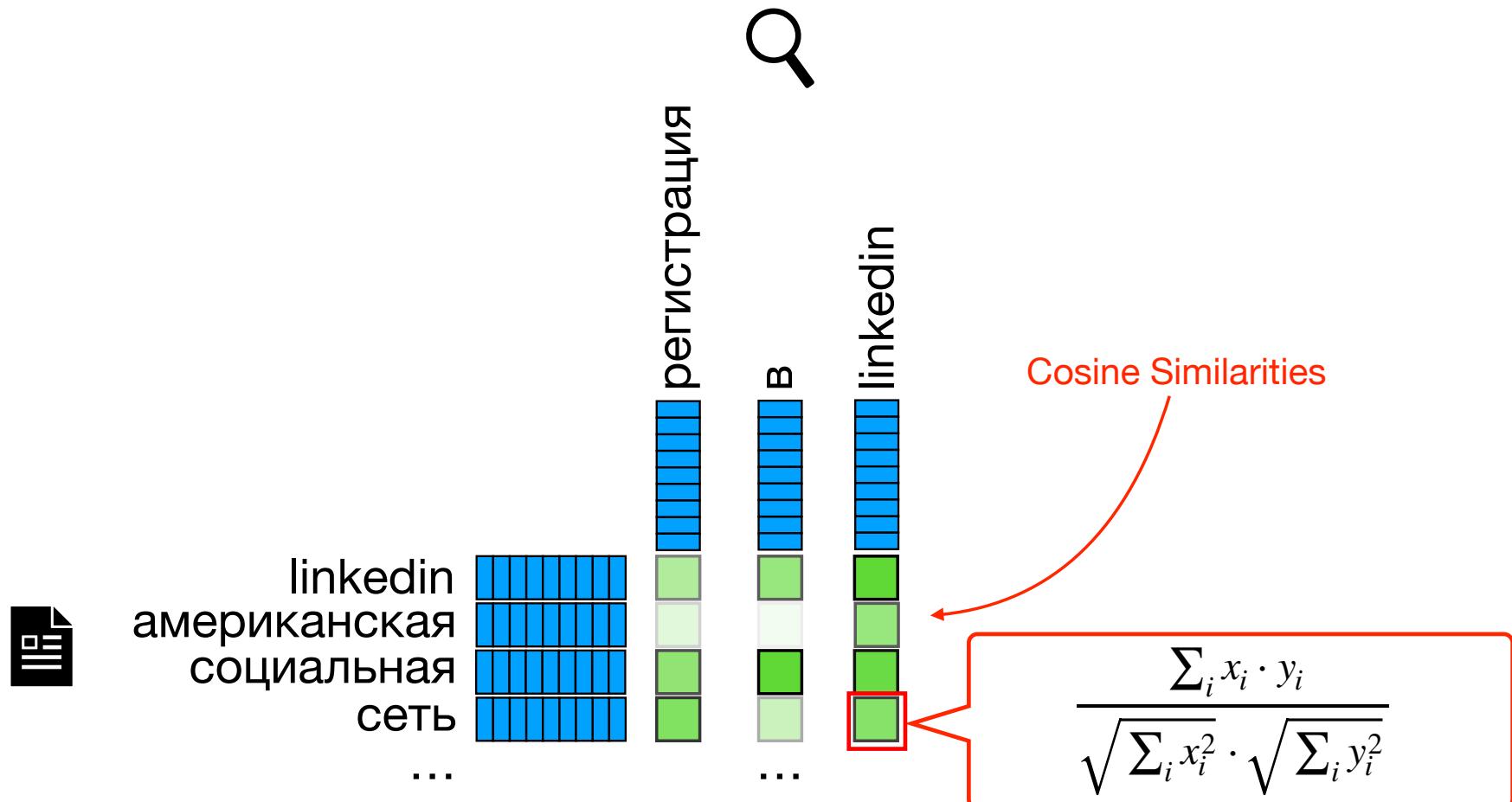
Contextualized Late Interaction over BERT



Aggregated Similarity – MaxSim



Aggregated Similarity – MaxSim



Aggregated Similarity – MaxSim

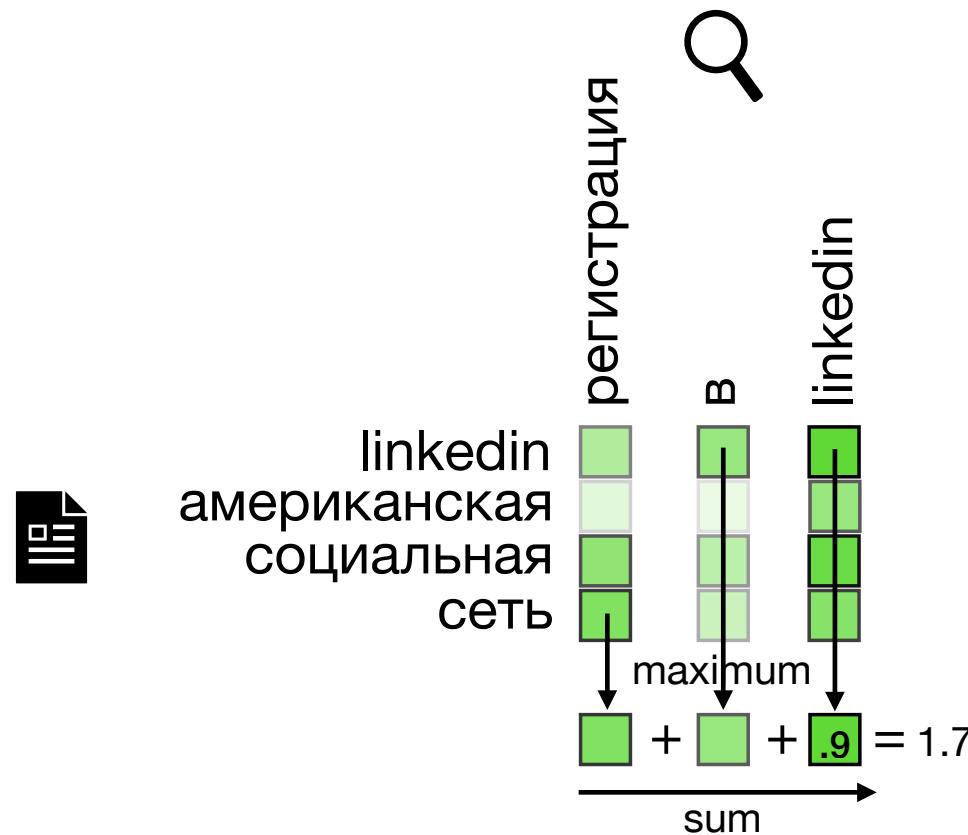
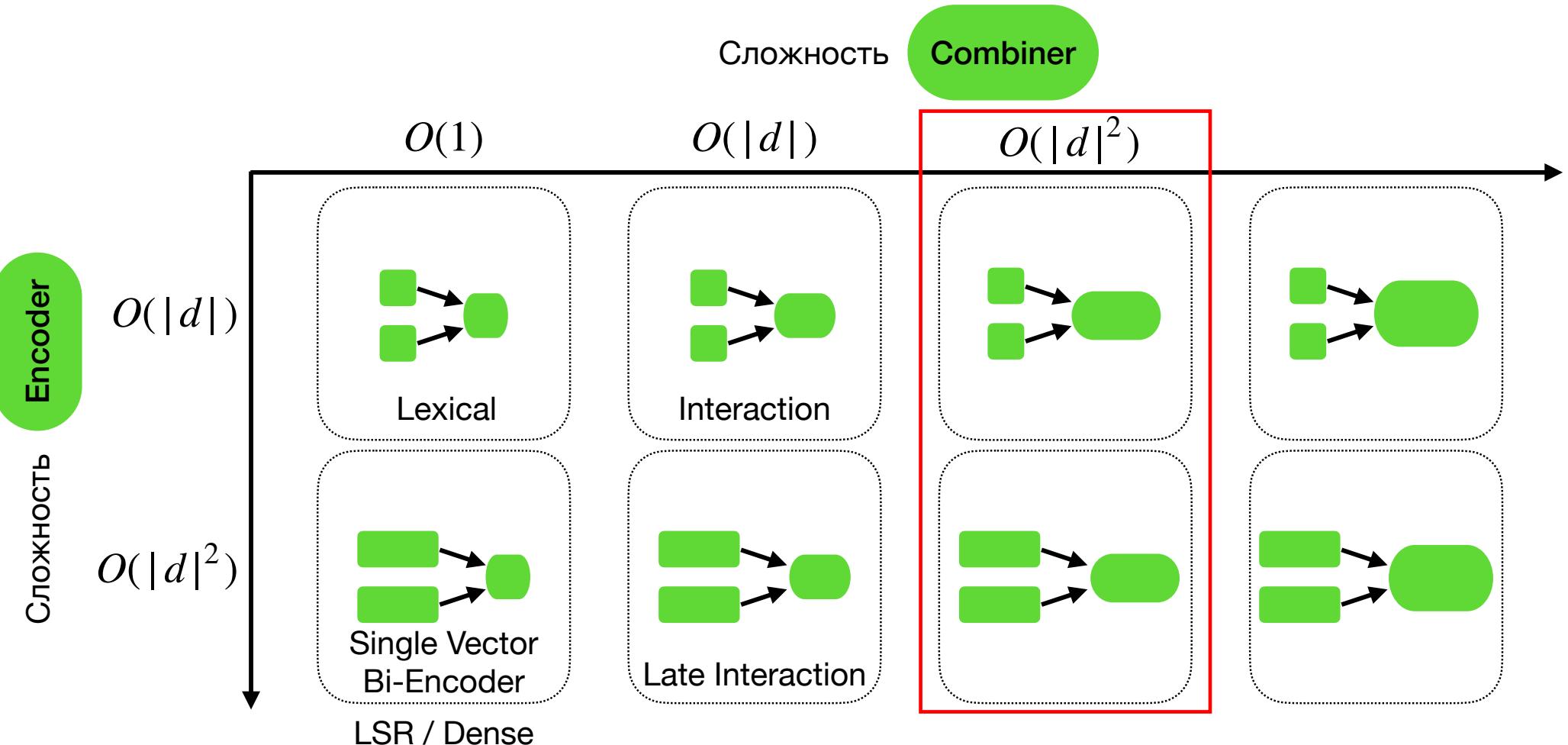
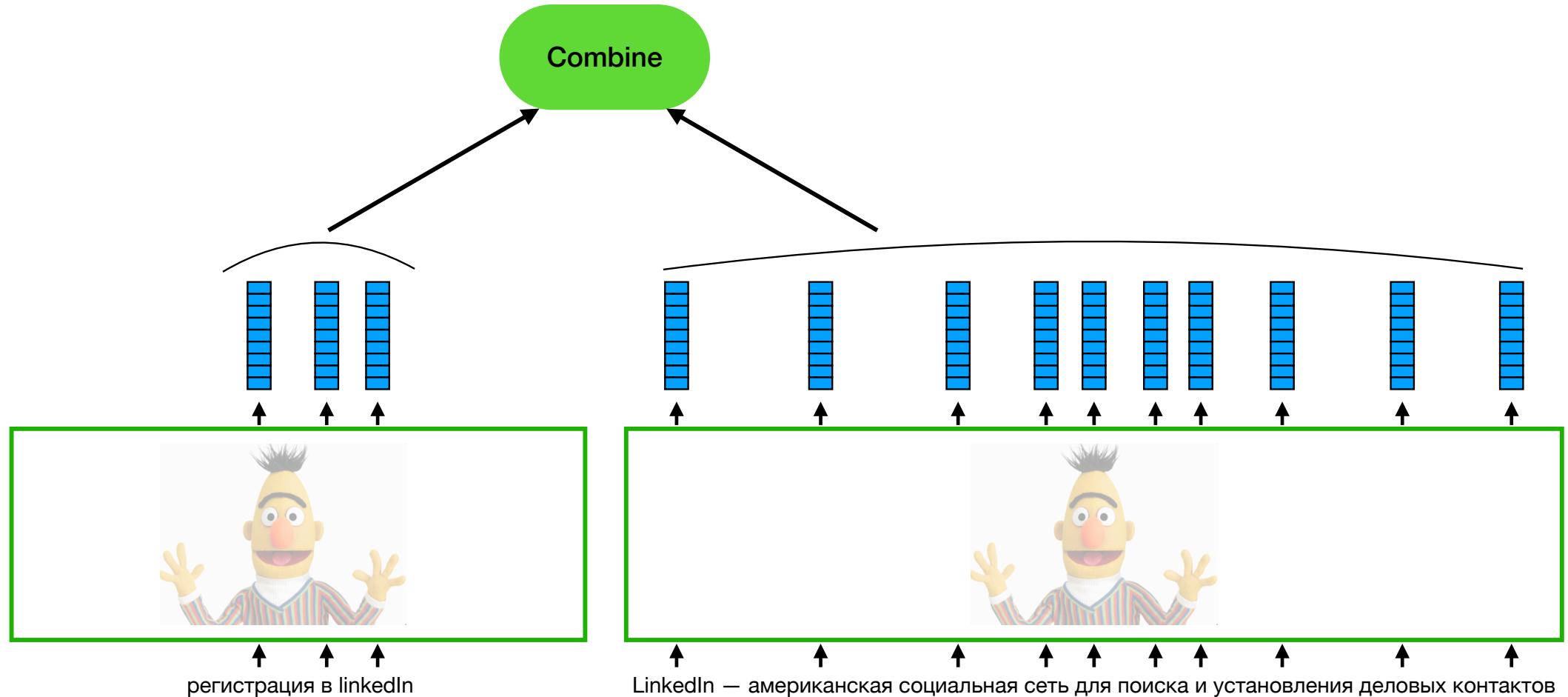


Таблица моделей релевантности



Late Interaction Models



Mono Encoders



Relevance
Score



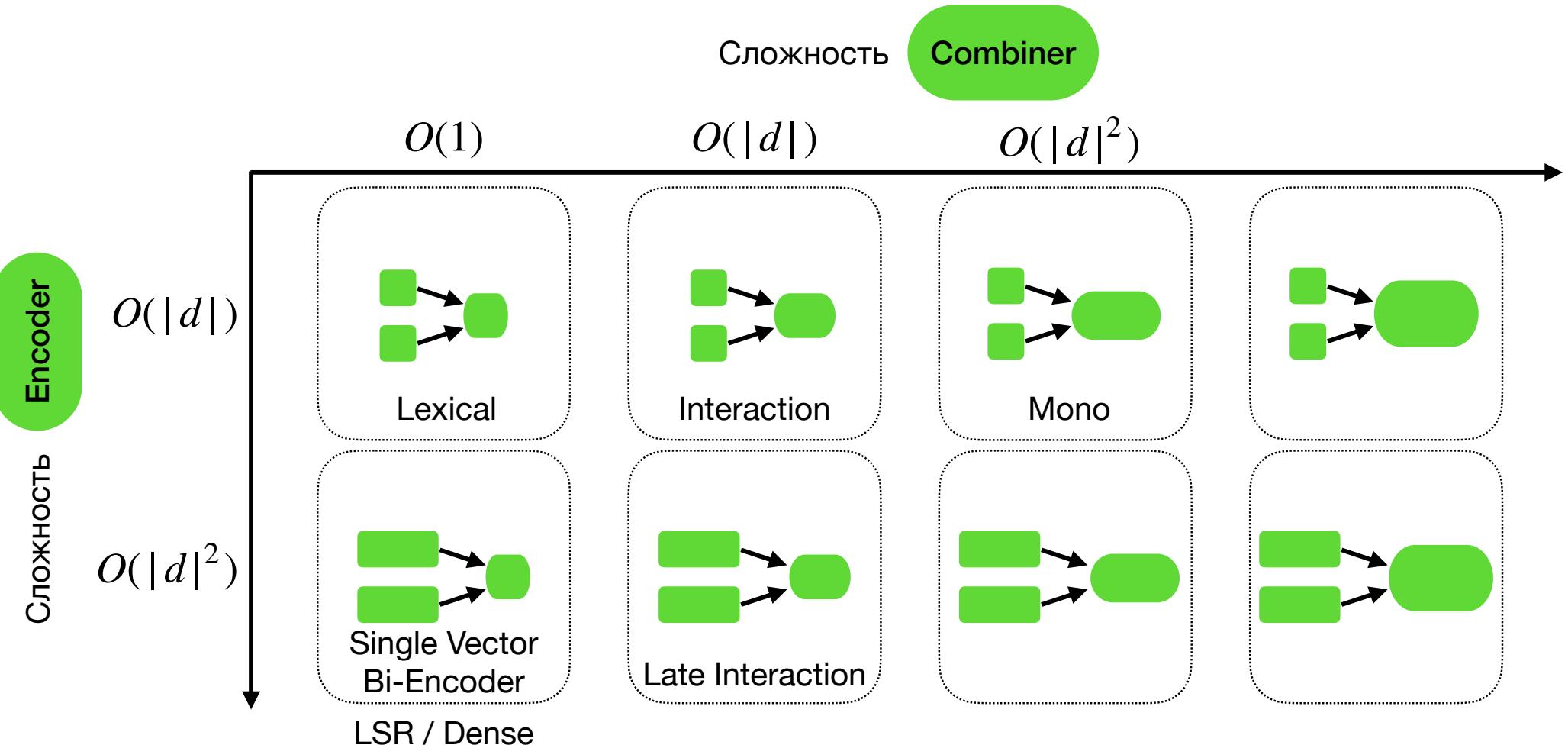
↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑

[CLS] регистрация в linkedIn [SEP] LinkedIn — американская социальная сеть для поиска и установления деловых контактов

Mono Relevance Models

- Минимум раздельного представления запроса и документа — только токенизация
- Комбинация основывается на кросс-внимании между запросом и документом
- Вариации:
 - Архитектура сети (encoder-only, encoder-decoder, decoder-only)
 - Attention Pruning (пропускаются некоторые отношения)
 - Размер модели
 - etc.
- Модели:
 - MonoBERT
 - MonoT5
 - RankT5

Таблица моделей релевантности



Late Mono Relevance Models



Relevance
Score



регистрация в LinkedIn



LinkedIn — американская социальная сеть для поиска и установления деловых контактов

Таблица моделей релевантности

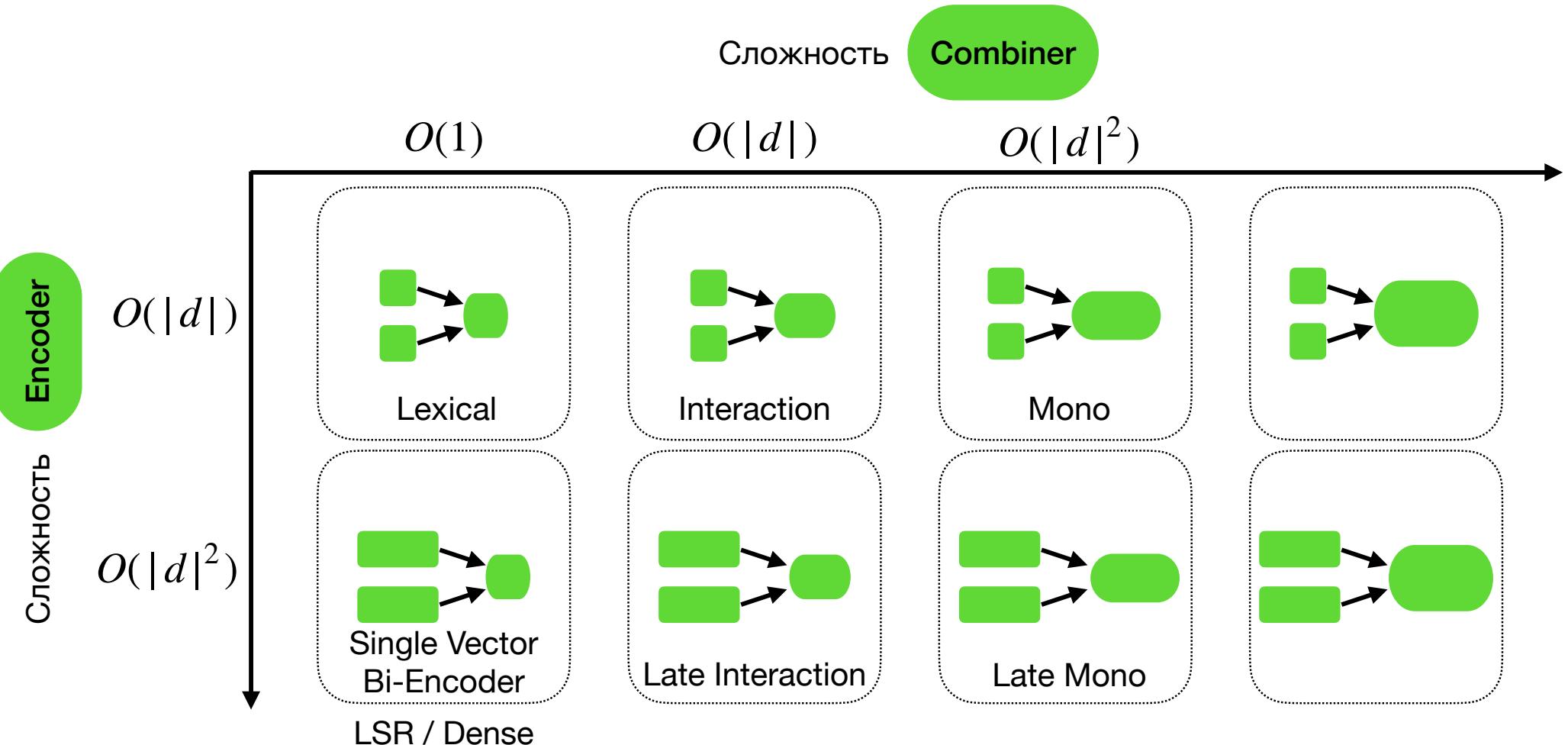
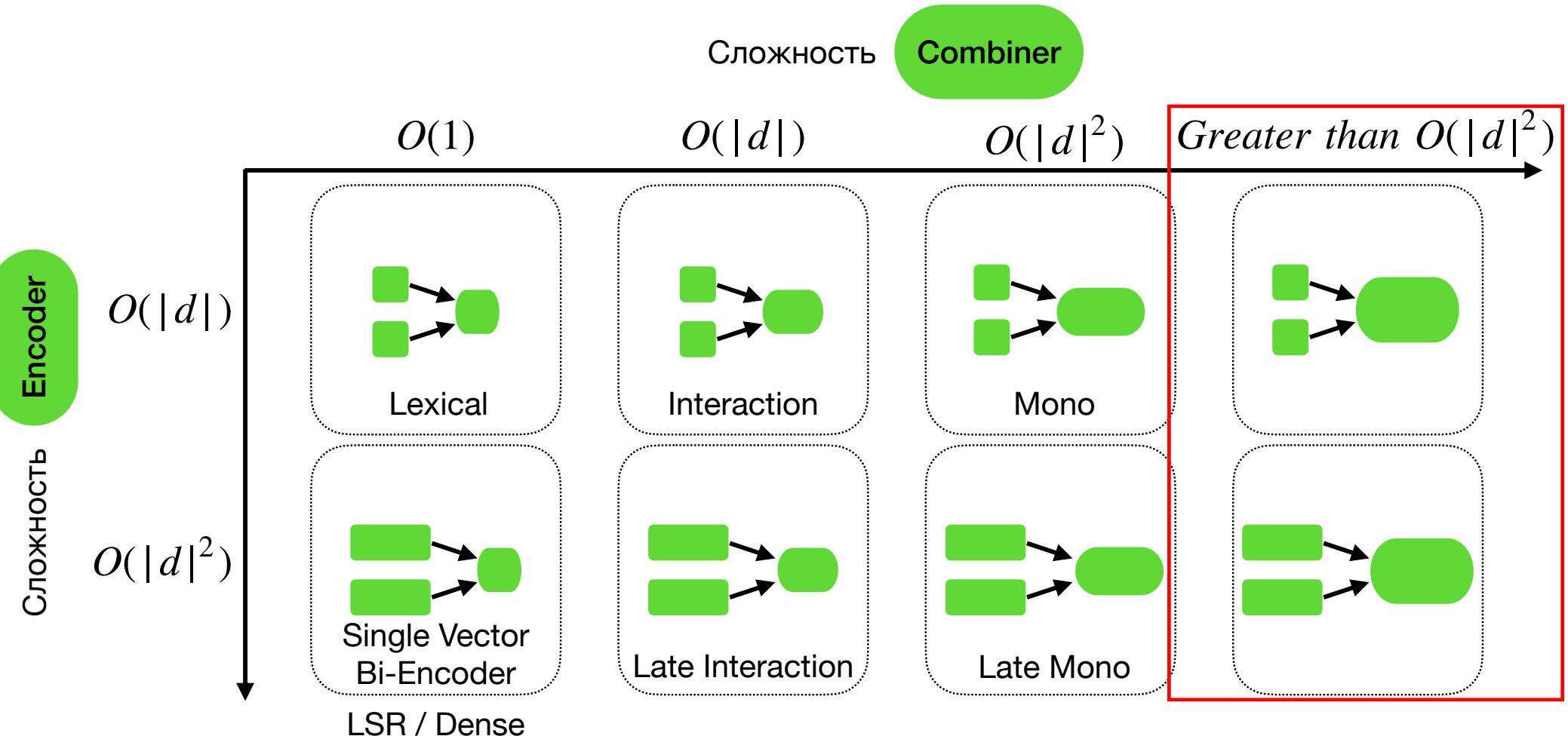
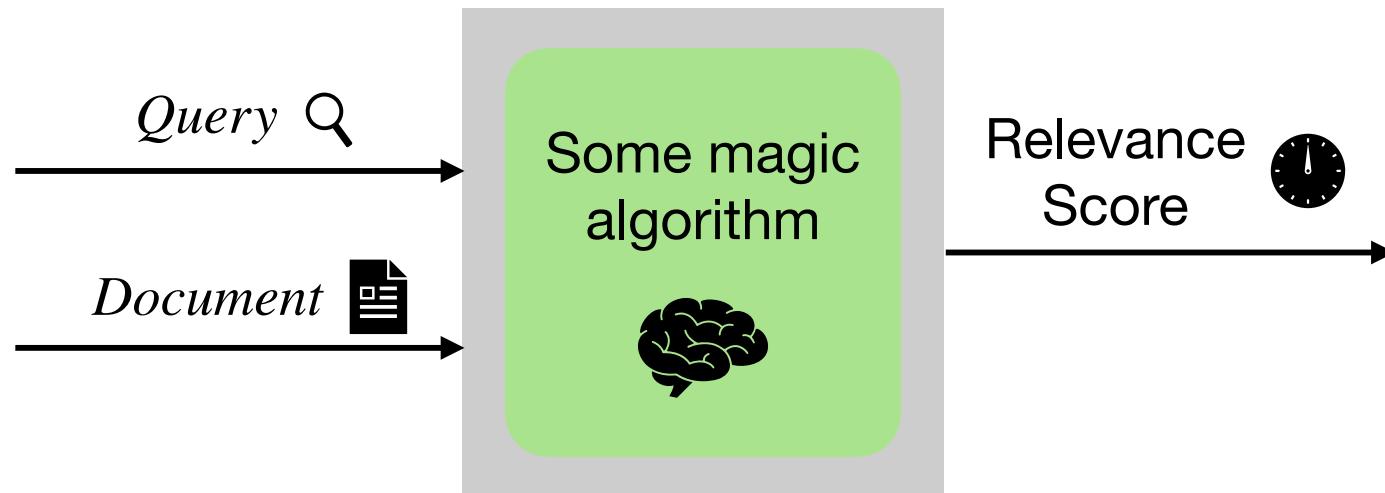


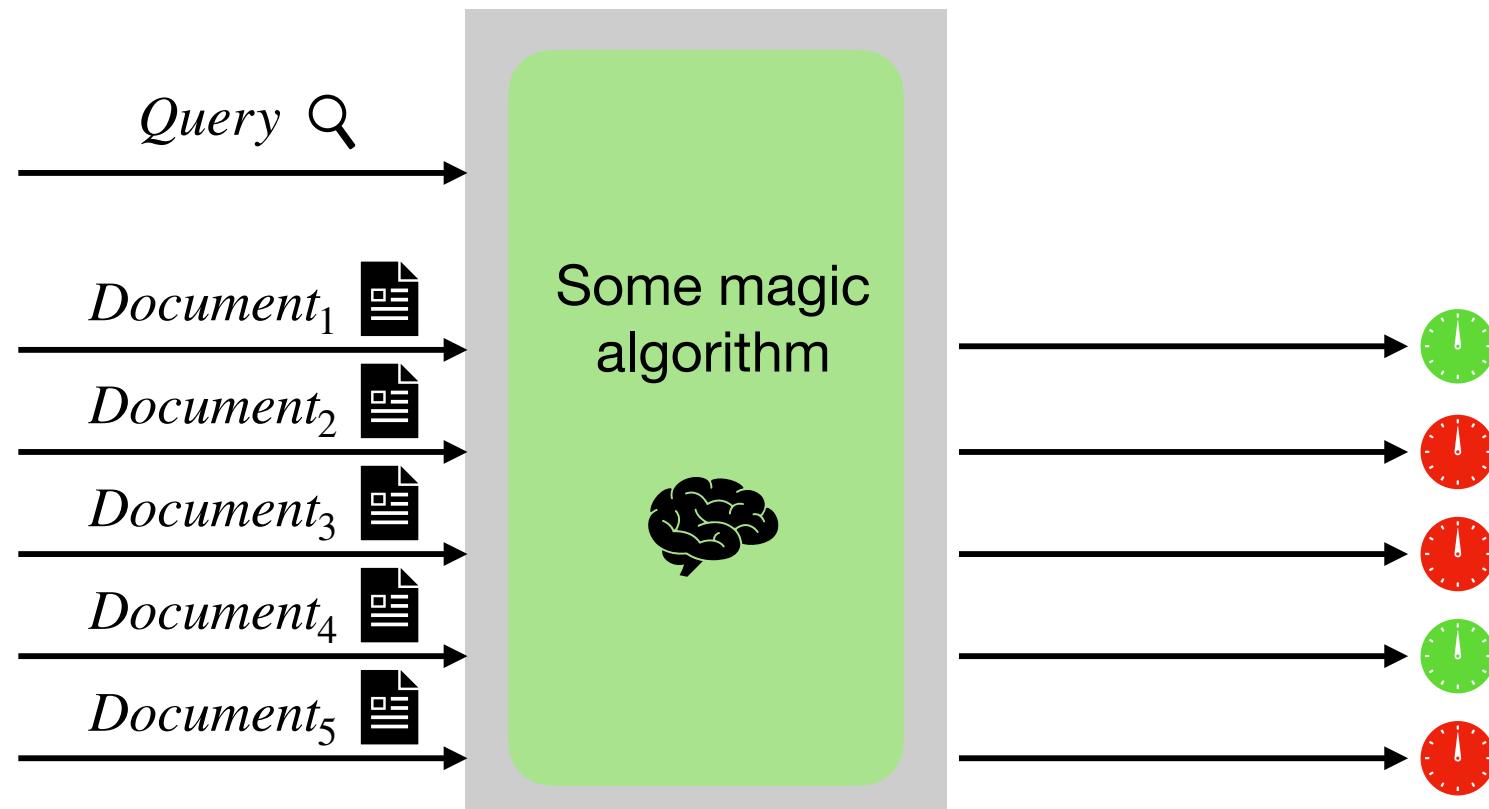
Таблица моделей релевантности



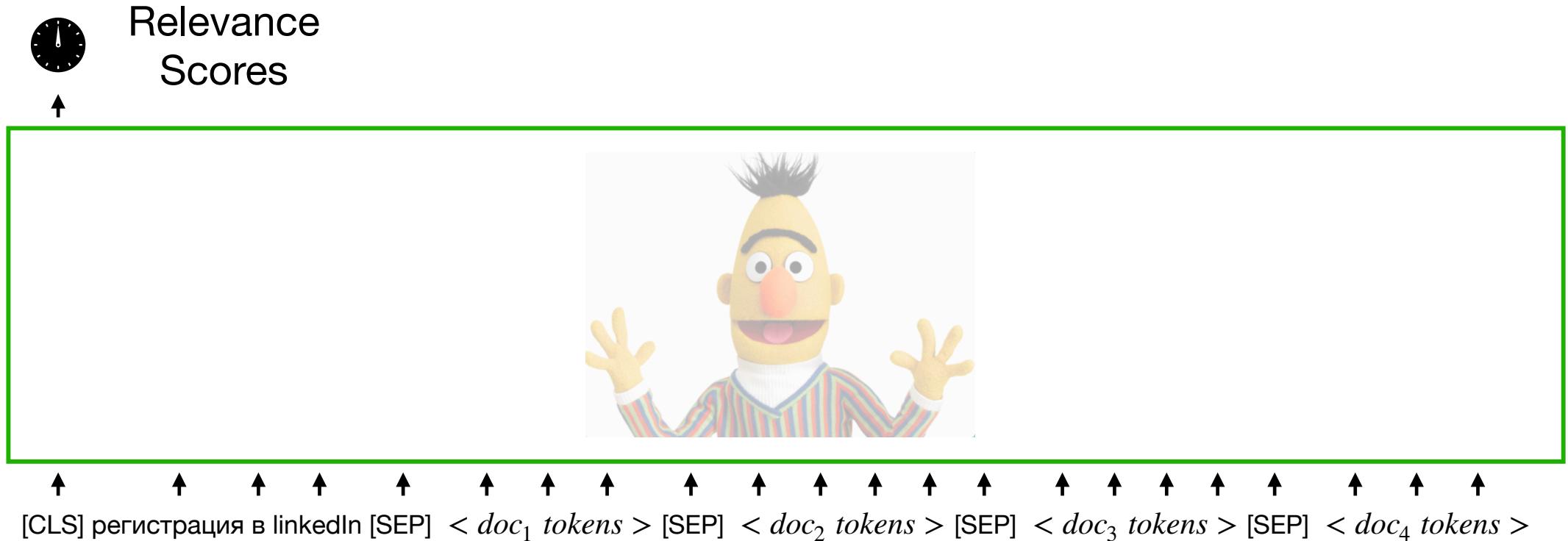
Модели релевантности



Модели релевантности



Duo, Listo, etc. Cross Encoders



Duo, Listo, etc. Cross Encoders

- Комбинация основывается на кросс-внимании между запросом и несколькими документами
- Модели:
 - DuoBERT
 - ListoBERT
 - RankGPT
 - PE-Rank
 - etc.

Таблица моделей релевантности

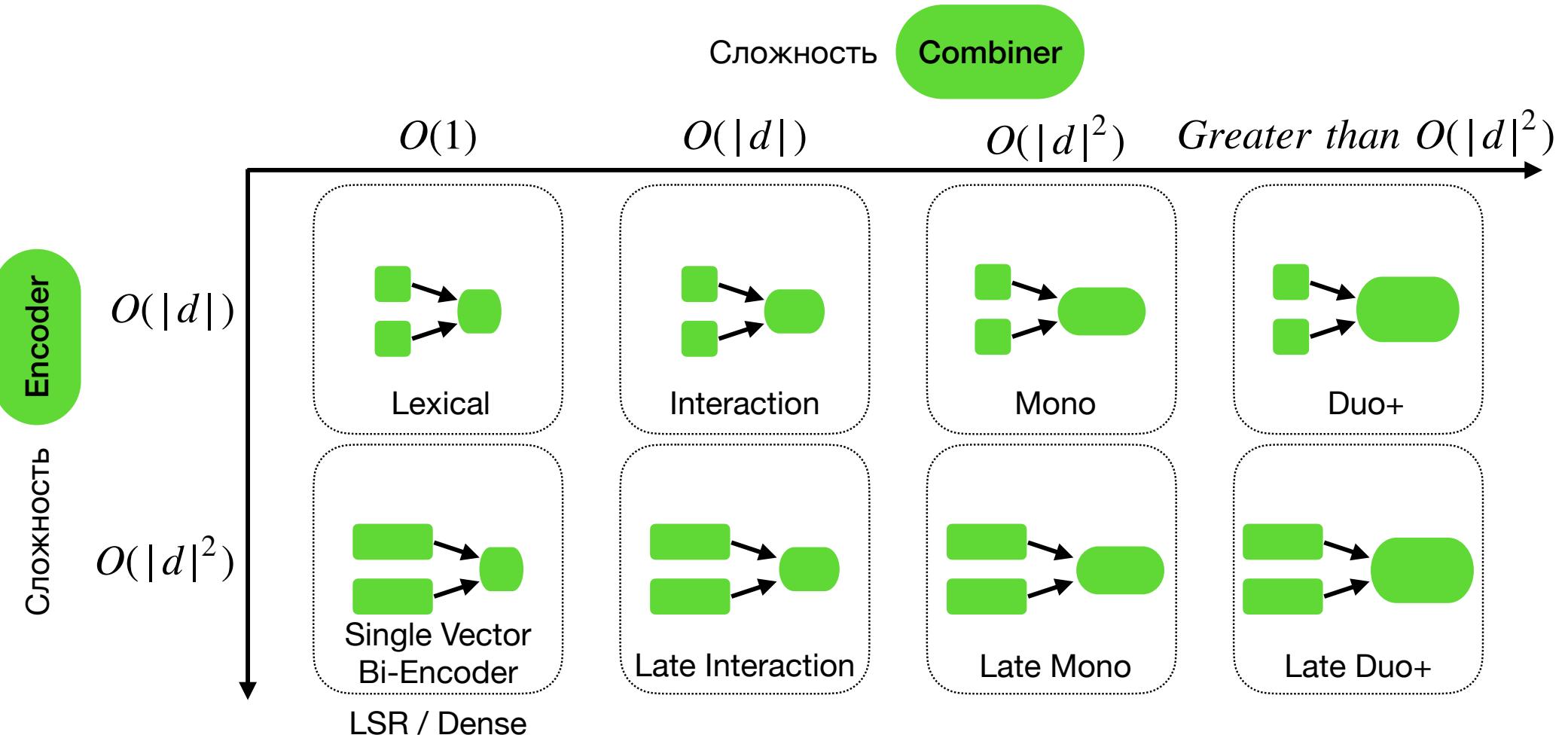


Таблица моделей релевантности

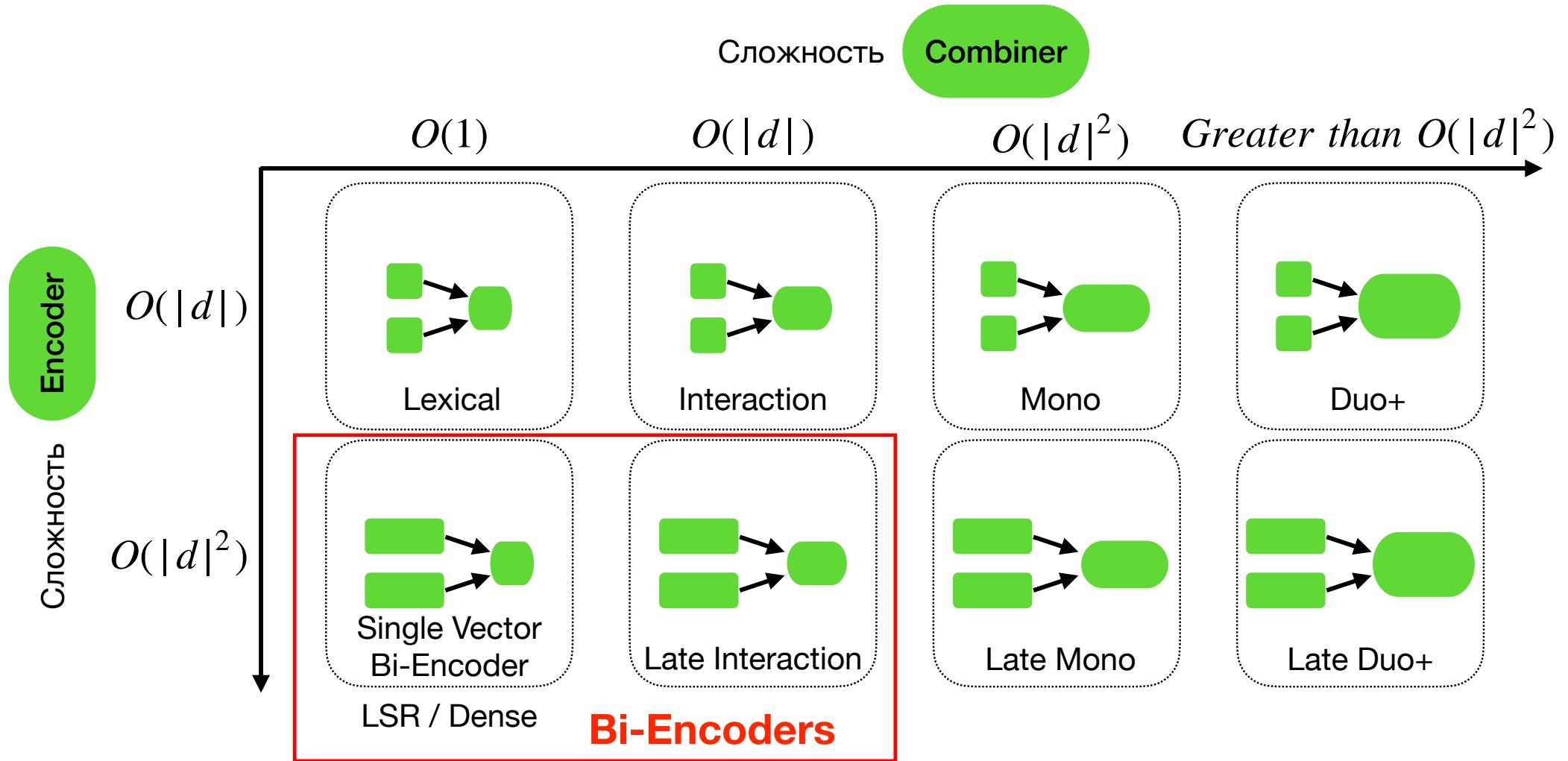


Таблица моделей релевантности

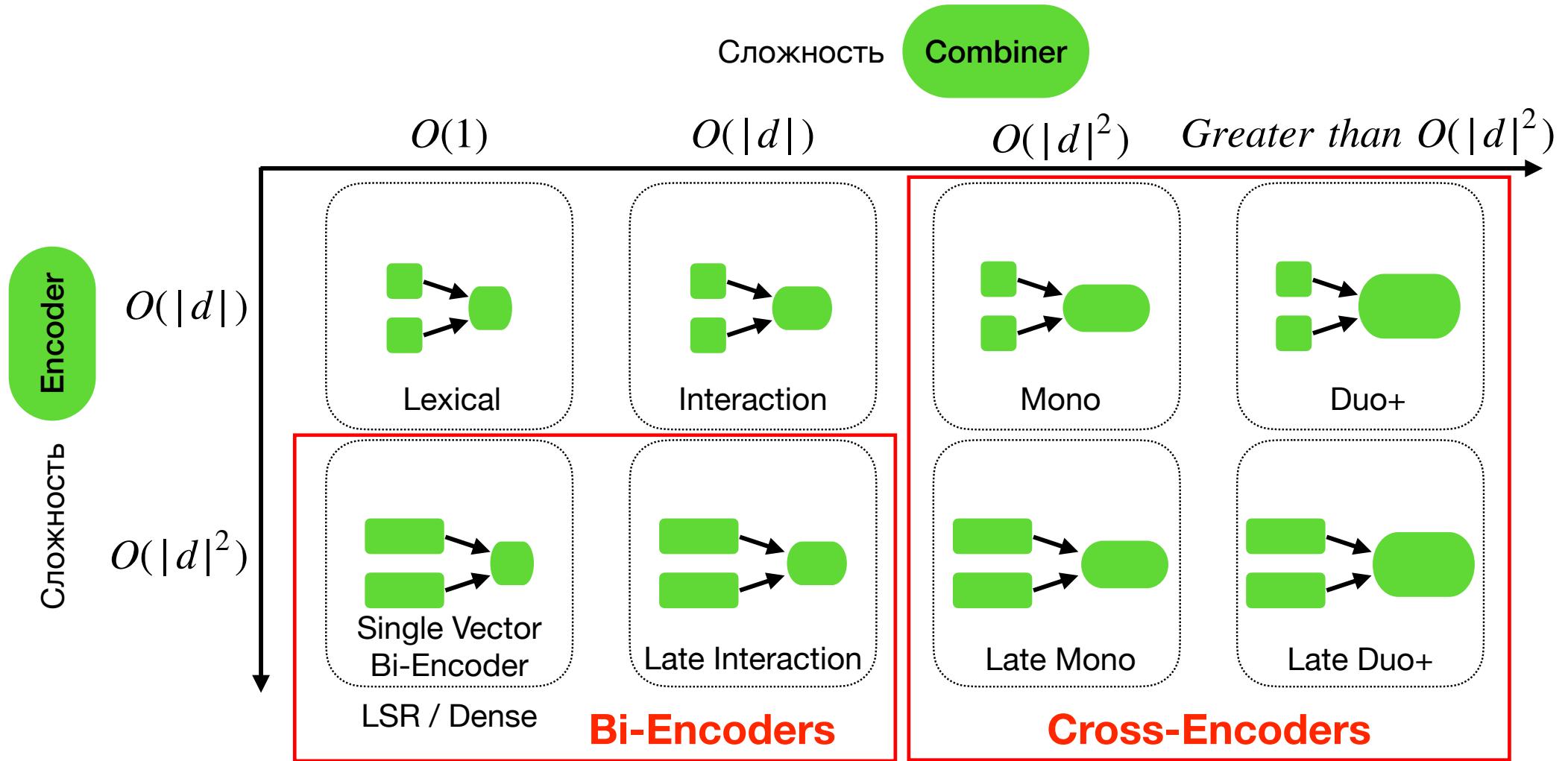


Таблица моделей релевантности

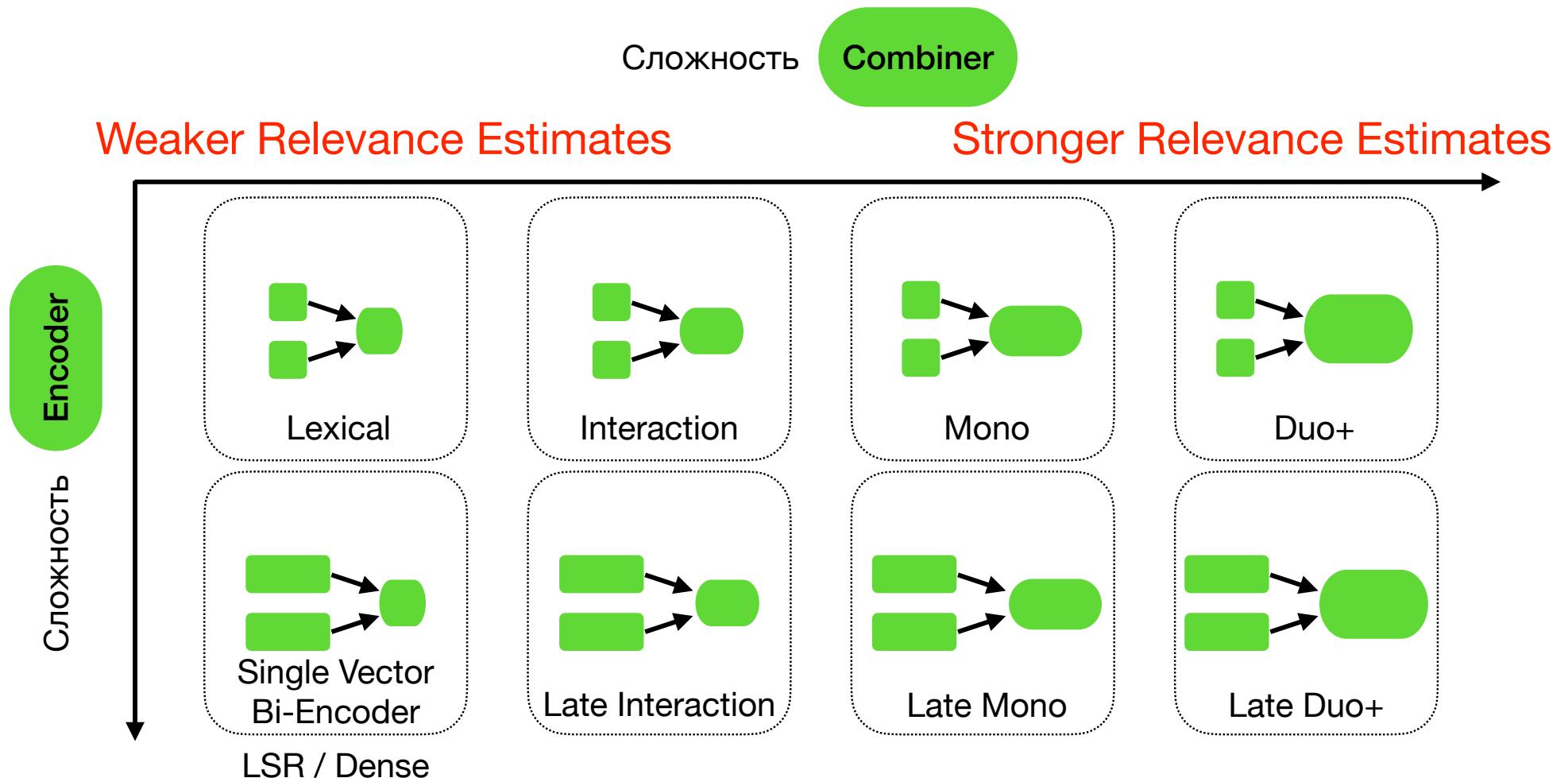


Таблица моделей релевантности

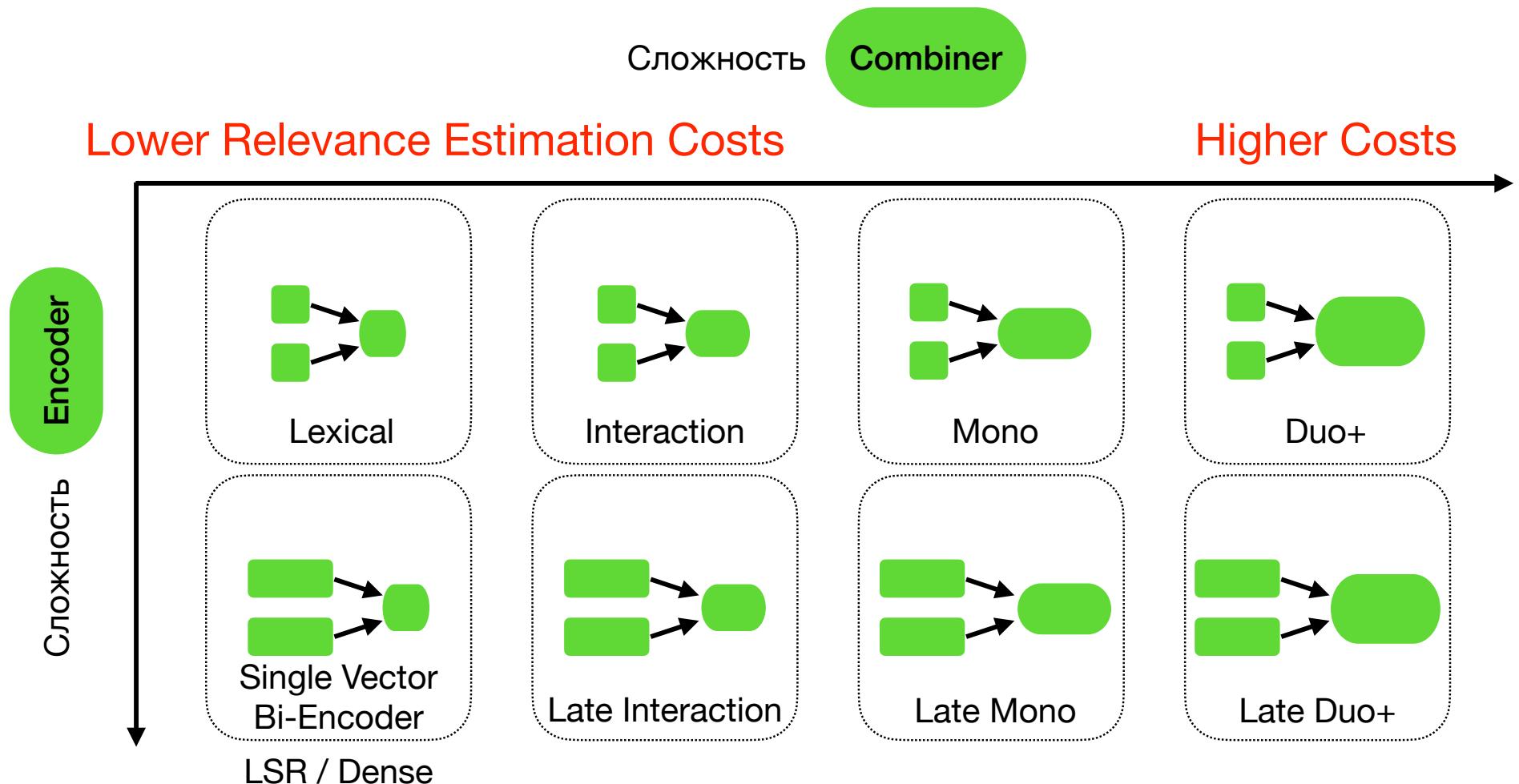


Таблица моделей релевантности

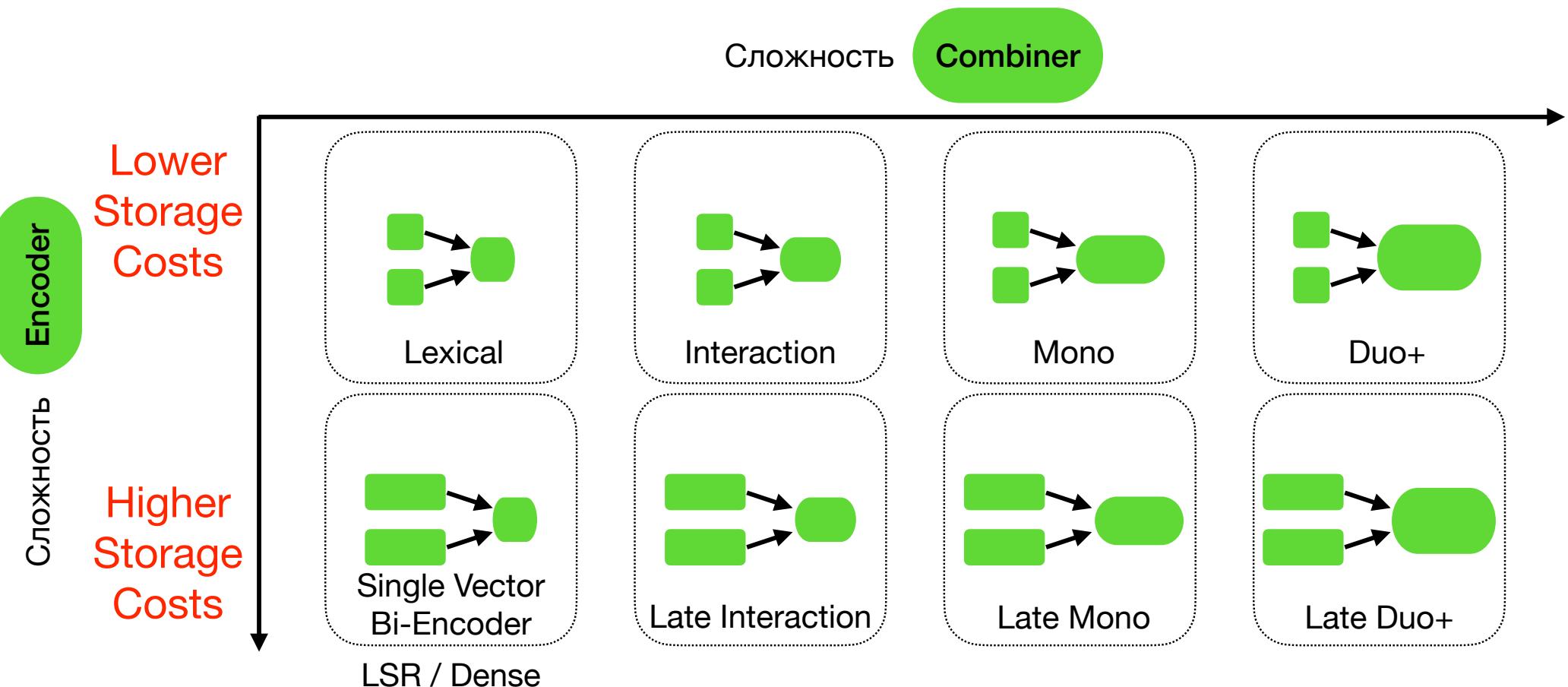
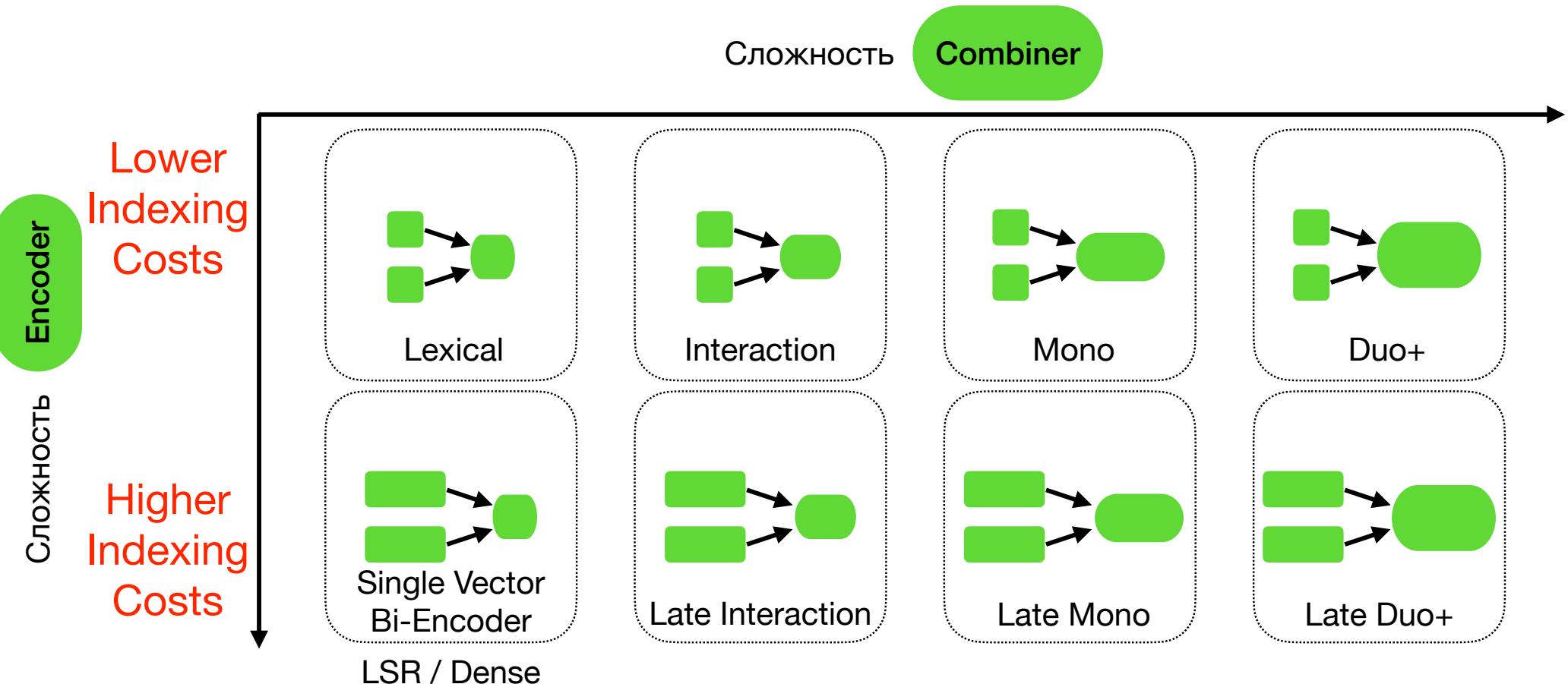


Таблица моделей релевантности



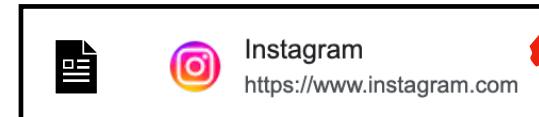
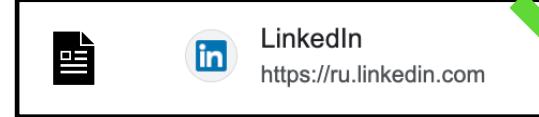
Обучение моделей релевантности

Human-labeled Training Data

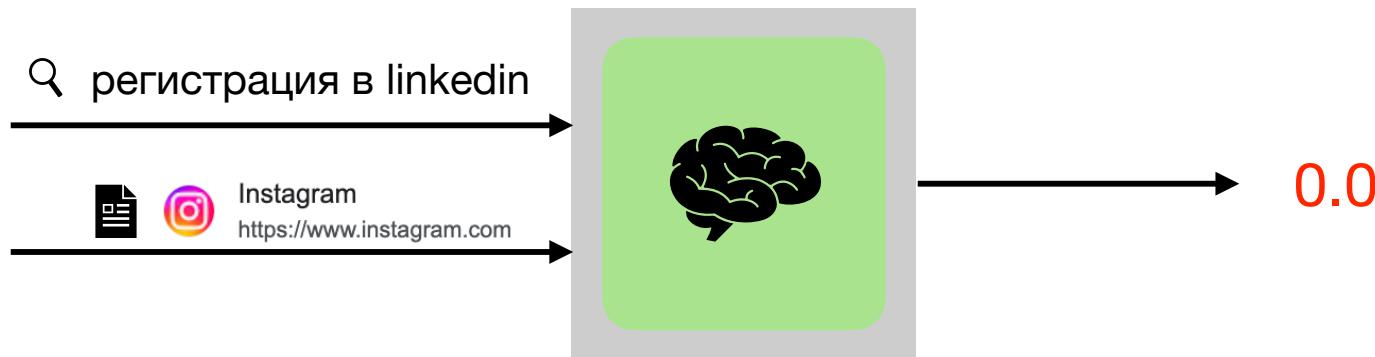
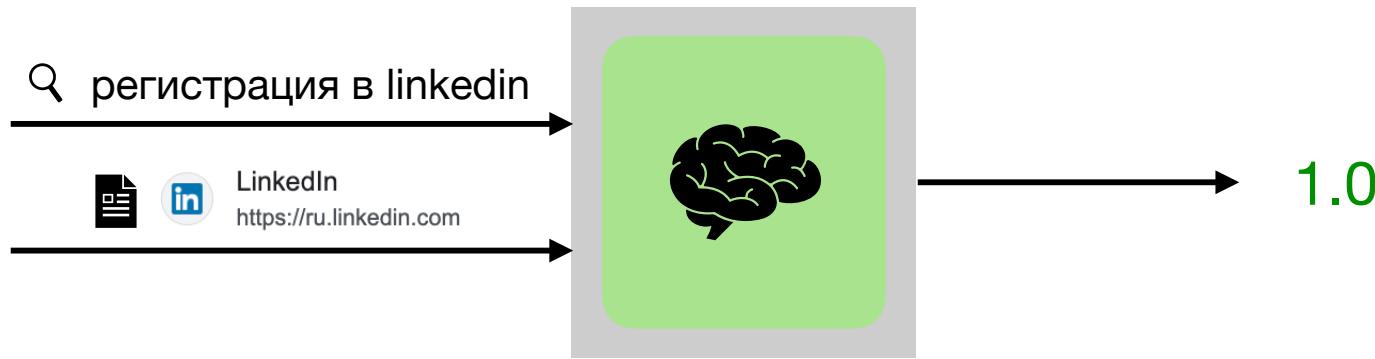
Размеченная людьми обучающая выборка:

- Sample 1:
 - Query: «регистрация в linkedin»
 - Document: «LinkedIn ...»
 - Label: 1 (relevant)
- Sample 2:
 - Query: «регистрация в linkedin»
 - Document: «Instagram ...»
 - Label: 0 (irrelevant)

🔍 регистрация в linkedin



Pointwise Training



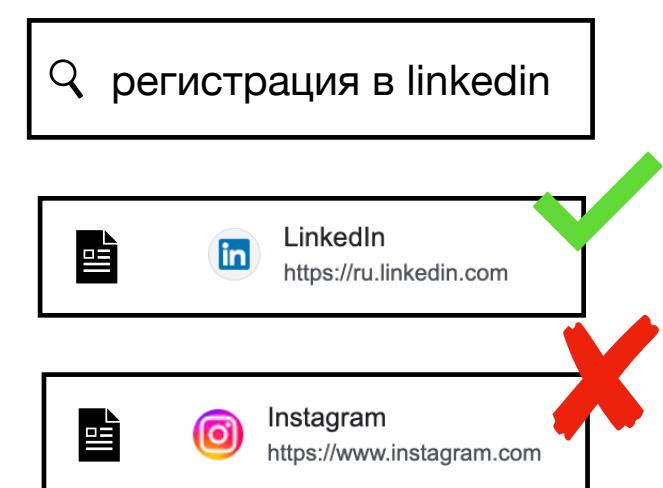
Пример: Mean Squared Error (MSE)

$$Loss(q, d, label) = (Rel(d | q) - label)^2$$

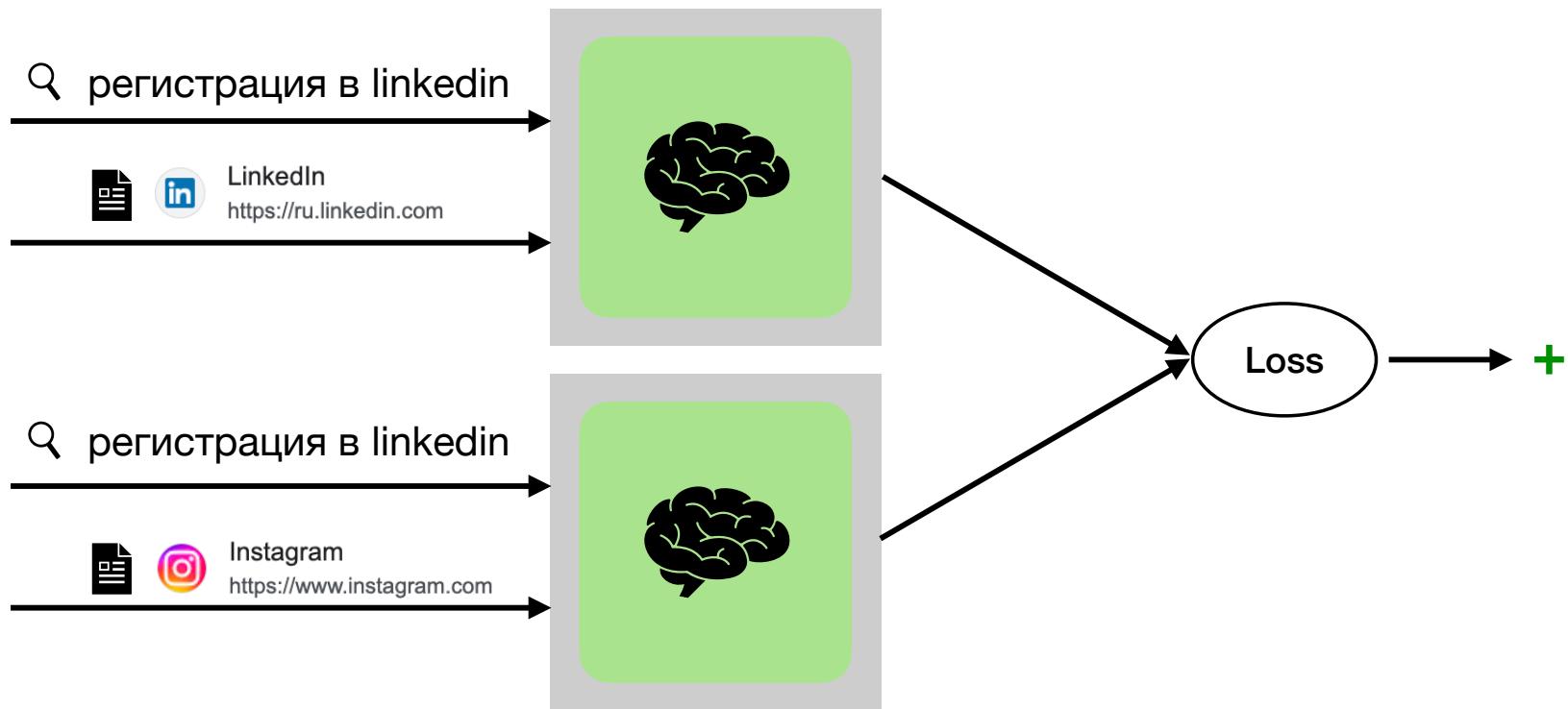
Contrastive Triples

Размеченная людьми обучающая выборка:

- Sample 1:
 - Query: «регистрация в linkedin»
 - Relevant document: «LinkedIn ...»
 - Non-relevant document: «Instagram ...»
- Sample 2:
 - ...



Contrastive Triples



Пример: Cross-Entropy Loss

$$Loss(q, d_+, d_-) = - \log \frac{e^{Rel(d_+|q)}}{e^{Rel(d_+|q)} + e^{Rel(d_-|q)}}$$

In-batch Negatives

Batch примеров обучающей выборки:

- Query1, Doc1+, Doc2–
- Query2, Doc3+, Doc4–
- Query3, Doc5+, Doc6–
- Query4, Doc7+, Doc8–

In-batch Negatives

Batch примеров обучающей выборки:

- **Query1, Doc1+, Doc2–**
- Query2, Doc3+, Doc4–
- Query3, Doc5+, Doc6–
- Query4, Doc7+, Doc8–

In-batch Negatives

Batch примеров обучающей выборки:

- **Query1, Doc1+, Doc2–**
- **Query2, Doc3+, Doc4–**
- **Query3, Doc5+, Doc6–**
- **Query4, Doc7+, Doc8–**

На практике в некотором приближении можно переиспользовать все остальные примеры из батча как негативы для первого запроса

In-batch Negatives

Batch примеров обучающей выборки:

- Query1, Doc1+, Doc2–
- Query2, Doc3+, Doc4–
- Query3, Doc5+, Doc6–
- Query4, Doc7+, Doc8–

На практике в некотором приближении можно переиспользовать все остальные примеры из батча как негативы для первого запроса

In-batch Negatives

Batch примеров обучающей выборки:

- Query1, **Doc1+**, **Doc2-**
- Query2, **Doc3+**, **Doc4-**
- **Query3, Doc5+, Doc6-**
- Query4, **Doc7+, Doc8-**

На практике в некотором приближении можно переиспользовать все остальные примеры из батча как негативы для первого запроса

In-batch Negatives

Batch примеров обучающей выборки:

- Query1, **Doc1+**, **Doc2-**
- Query2, **Doc3+**, **Doc4-**
- Query3, **Doc5+**, **Doc6-**
- **Query4, Doc7+, Doc8-**

На практике в некотором приближении можно переиспользовать все остальные примеры из батча как негативы для первого запроса

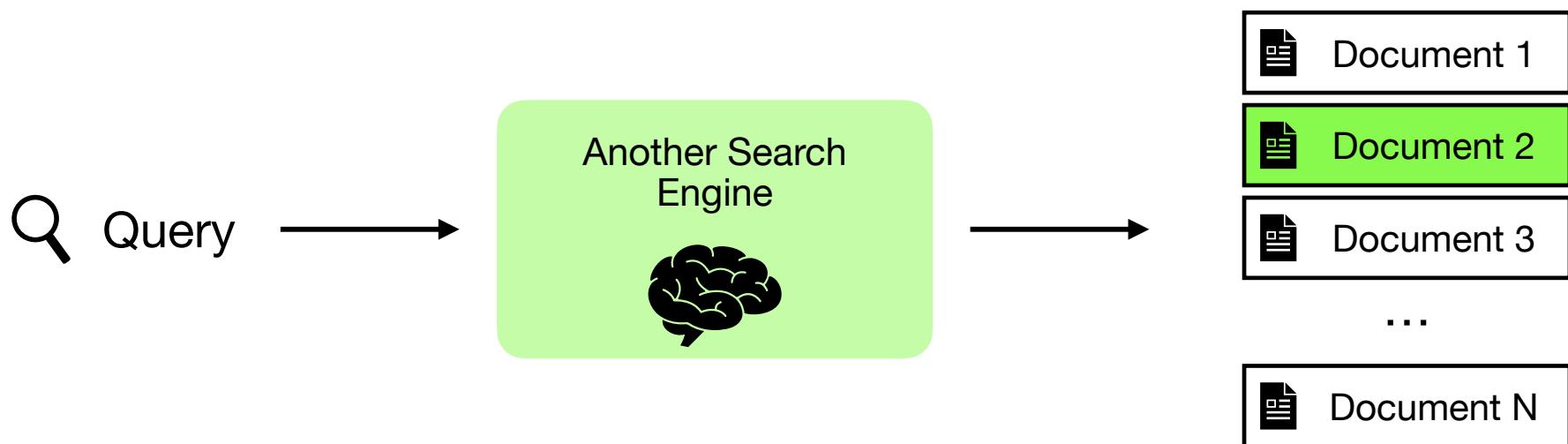
Hard Negatives Mining

На практике обычно у нас есть пары **запрос-позитив** (например, клики пользователей), но **негативы** могут быть неизвестны.

Hard Negatives Mining

На практике обычно у нас есть пары **запрос-позитив** (например, клики пользователей), но **негативы** могут быть неизвестны.

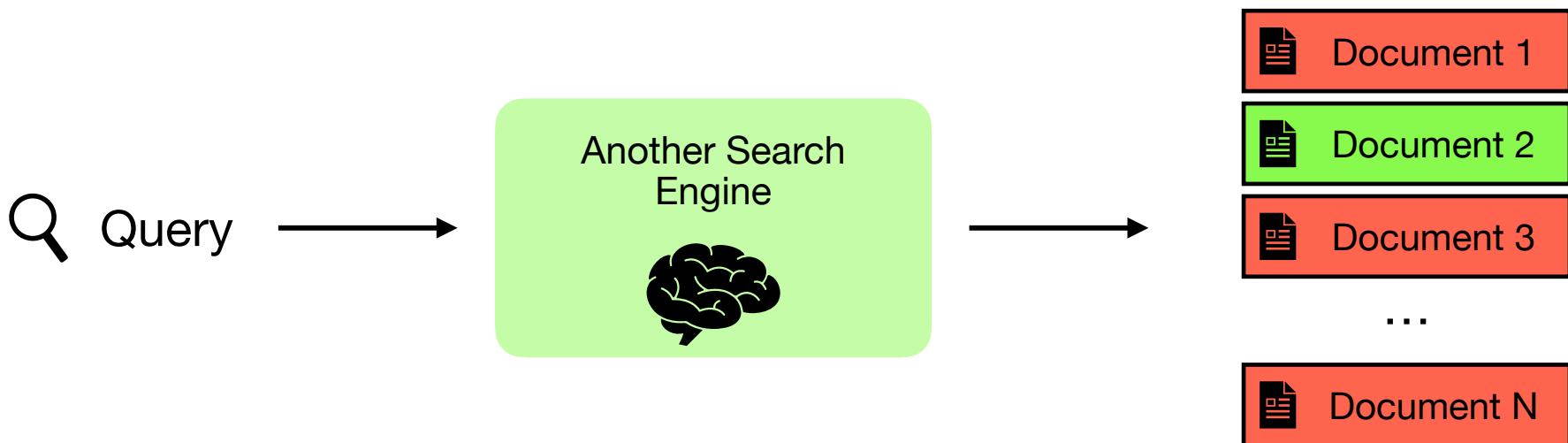
В этом случае можно использовать топ отранжированных другой моделью документов как негативы:



Hard Negatives Mining

На практике обычно у нас есть пары **запрос-позитив** (например, клики пользователей), но **негативы** могут быть неизвестны.

В этом случае можно использовать топ отранжированных другой моделью документов как негативы:



Hard Negatives Mining

Достоинства и недостатки

Плюсы:

- Негативы сильнее, чем случайно выбранные из корпуса документов

Минусы:

- Могут встречаться false-негативы (есть вероятность, что мы разметим релевантный документ как негатив)

Hard Negatives Mining

Достоинства и недостатки

Плюсы:

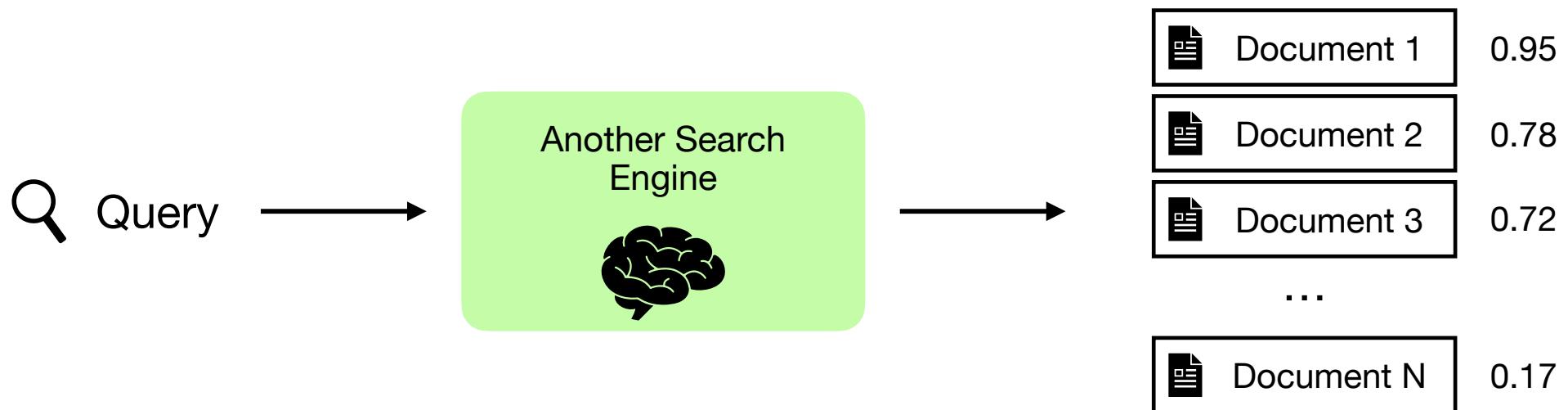
- Негативы сильнее, чем случайно выбранные из корпуса документов

Минусы:

- Могут встречаться false-негативы (есть вероятность, что мы разметим релевантный документ как негатив) — чтобы этого избежать, иногда берут в качестве hard-негатива документы не из топа выдачи, а чуть ниже, например, 100-й документ из отранжированного списка

Дистилляция знаний

Можно обучать модель предсказывать скоры документов подобно другой, более сильной модели:

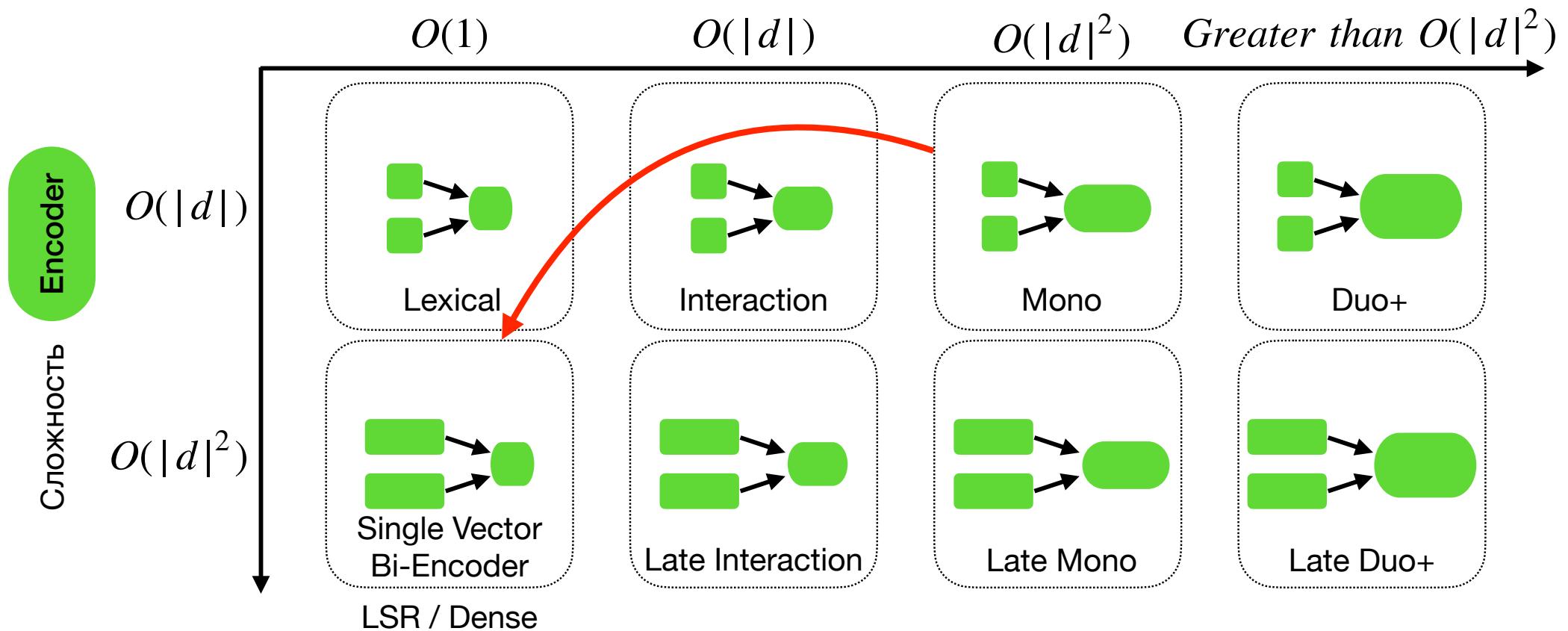


Пример: KL-дивергенция

$$Loss_{KL}(Strong, Weak) = \sum_{x \in X} Strong(x) \cdot \log \left(\frac{Strong(x)}{Weak(x)} \right)$$

Дистилляция знаний

Переносим знания более от сильной модели к более слабой



Нейросетевой IR

Андрсов Дмитрий, 07.04.2025, AI Masters