IR From Bag-of-words to BERT and Beyond through Practical Experiments

A Search Solutions 2022 Tutorial with PyTerrier

Sean MacAvaney*
Craig Macdonald*
Nicola Tonellotto*

*(*Alphabetical ordering)*

Part 2: Neural Re-Ranking:

Neural re-rankers such as
BERT, T5, EPIC, ColBERT

Adaptive Re-Ranking

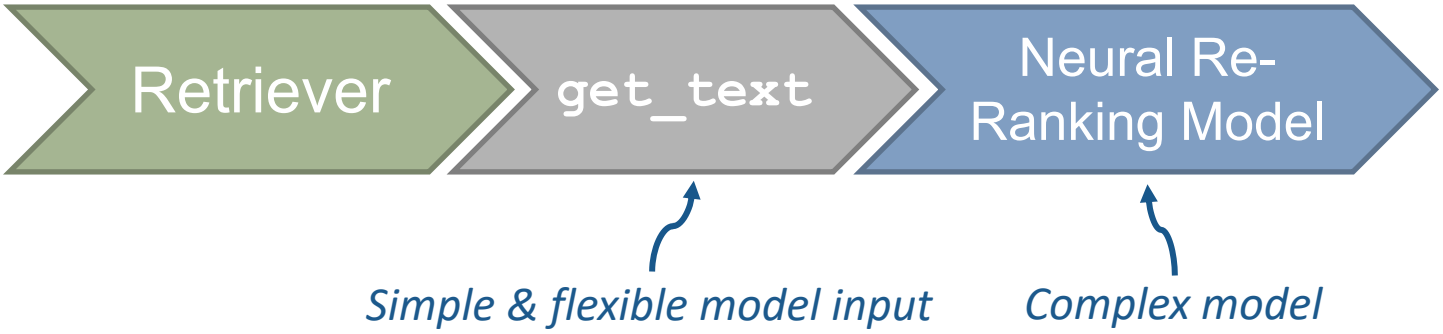
Moving From Features to Text

Università di Pisa | University of Glasgow

Learning to rank approaches require you to design effective features:

Performance heavily depends on these features

Neural methods adapted from NLP can learn patterns from the text itself, reducing the need for manually designed features:



Moving from Features to Text

Università di Pisa | University of Glasgow

In this session, we will cover:

- the basics of neural re-ranking methods.
- some of the current SOTA methods using BERT and T5 for ranking.
- pre-computation of document representations, reducing index time cost
- adaptive re-ranking, an approach for overcoming poor first stage recall

You will experience:

- re-ranking with models like BERT and T5 using PyTerrier and OpenNIR.
- performing re-ranking thresholding tuning.
- re-ranking with pre-computed representations to reduce query latency.
- performing index augmentation approaches like DeepCT and doc2query.

Intended Learning Outcomes

Università di Pisa | University of Glasgow

Part 2 – *Neural Re-Ranking*:
Neural re-rankers such as BERT, T5, EPIC, ColBERT.

ILO 2A. Understand contemporary retrieval architectures, such as using BERT, EPIC, ColBERT, and T5 as neural re-rankers.

ILO 2B. Understand how query time latency can be reduced by pre-computing representations.

ILO 2C. Understand how poor first stage recall can be addressed using adaptive re-ranking.

ILO 2D. Perform experiments with BERT, EPIC, and T5 in a Python notebook.

Outline of Part 2

Università di Pisa | University of Glasgow

Part 2A: **Background**

Part 2B: **Neural Re-Ranking Models (e.g., MonoT5)**

Part 2C: **Managing Efficiency (e.g., EPIC)**

Part 2D: **Adaptive Re-Ranking (e.g., GAR)**

Part 2E: **Wrap up and move to practical session**

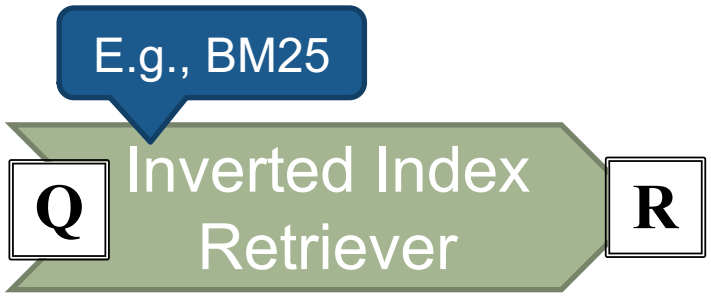
University of Glasgow | Università di Pisa

SEARCH SOLUTIONS London, UK

Part 2A

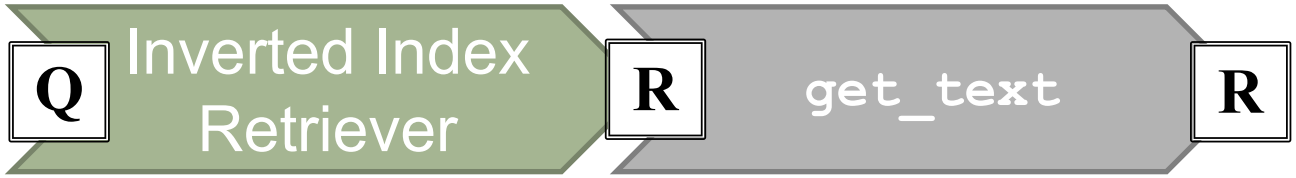
NEURAL RE-RANKING BACKGROUND

Moving from Features to Text



qid	query	docno	score
0	london…	43242	0.1252
0	london…	13746	0.1196
0	london…	86454	0.0934
…	…	…	…

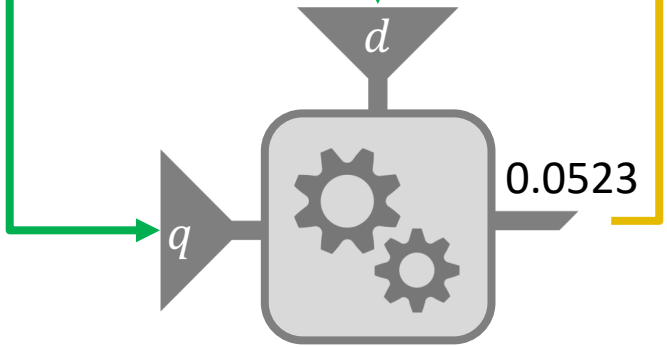
Moving from Features to Text



qid	query	docno	score	text
0	london…	43242	0.1252	London features a humid temperate…
0	london…	13746	0.1196	Temperatures in the London England…
0	london…	86454	0.0934	Weather forecast: England, London…
…	…	…	…	…

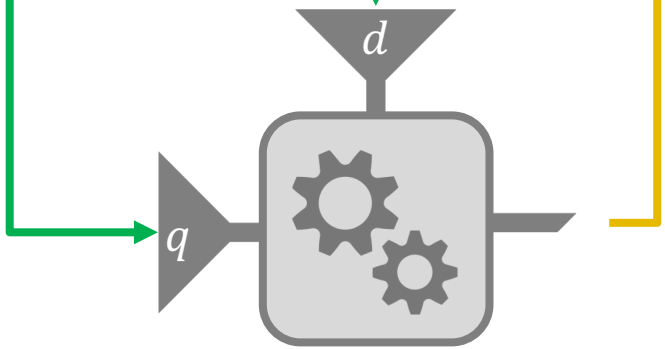
Moving from Features to Text

qid	query	docno	score	text
0	london…	43242	0.0523	London features a humid temperate…
0	london…	13746	0.1196	Temperatures in the London England…
0	london…	86454	0.0934	Weather forecast: England, London…
…	…	…	…	…



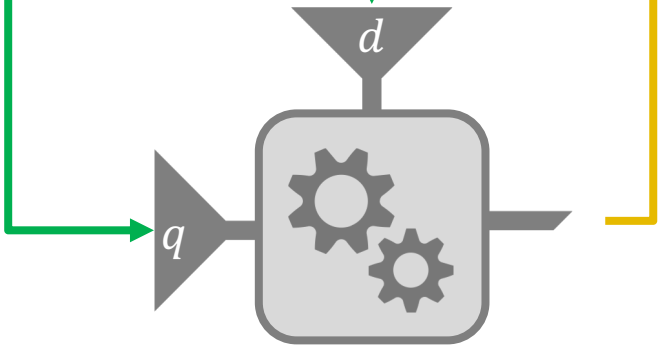
Moving from Features to Text

qid	query	docno	score	text
0	london…	43242	0.0523	London features a humid temperate…
0	london…	13746	0.8411	Temperatures in the London England…
0	london…	86454	0.0034	Weather forecast: England, London…
…	…	…	…	…



Moving from Features to Text

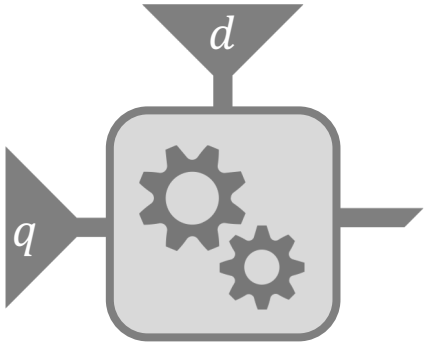
qid	query	docno	score	text
0	london…	43242	0.0523	London features a humid temperate…
0	london…	13746	0.8411	Temperatures in the London England…
0	london…	86454	0.1535	Weather forecast: England, London…
…	…	…	…	…



Moving from Features to Text

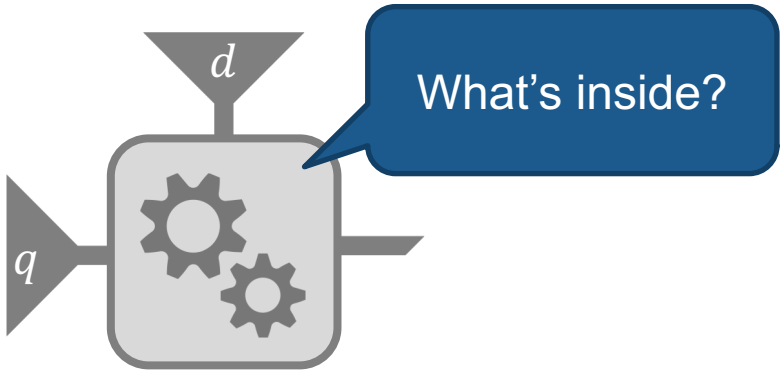
Università di Pisa | University of Glasgow

qid	query	docno	score	text
0	london…	43242	0.0523	London features a humid temperate…
0	london…	13746	0.8411	Temperatures in the London England…
0	london…	86454	0.1535	Weather forecast: England, London…
…	…	…	…	…



Moving from Features to Text

qid	query	docno	score	text
0	london…	13746	0.8411	Temperatures in the London England…
0	london…	86454	0.1535	Weather forecast: England, London…
0	london…	43242	0.0523	London features a humid temperate…
…	…	…	…	…



Representing Text

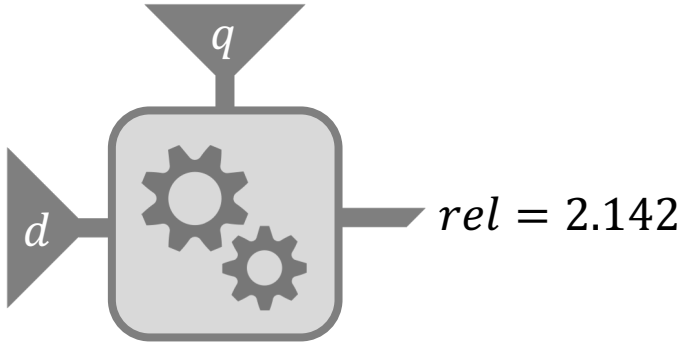
Università di Pisa | University of Glasgow

Coronavirus Early Symptoms

Document:

Title: How can we evaluate an interrelation of symptoms?

Abstract: A pandemic of 2019 novel coronavirus (COVID-19) is an international problem and factors associated with increased risk of mortality have been reported. However, there exists limited statistical method to estimate a comprehensive risk for a case in which a patient has several characteristics…



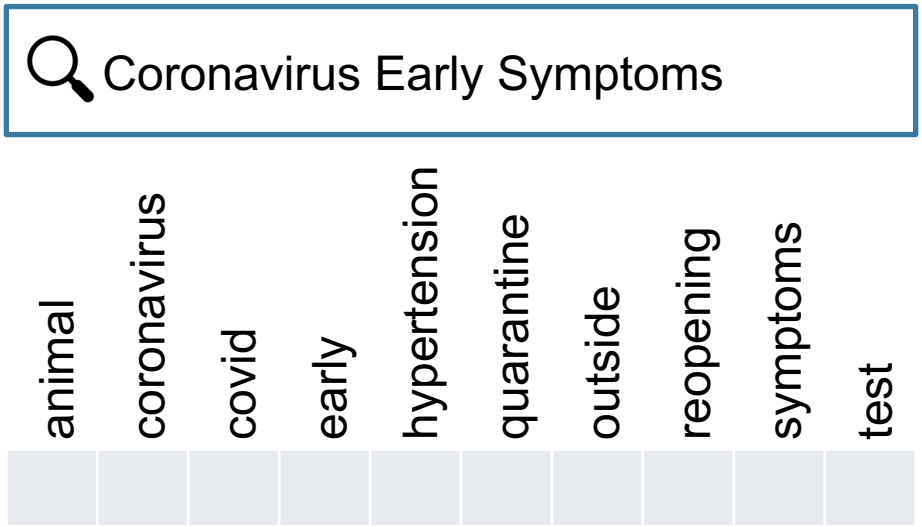
Representing Text

Option 1: One-hot encoding

animal	coronavirus	covid	early	hypertension	quarantine	outside	reopening	symptoms	test	...

Representing Text

Option 1: One-hot encoding



Representing Text

Option 1: One-hot encoding

	animal	**coronavirus**	covid	**early**	hypertension	quarantine	outside	reopening	**symptoms**	test	...
Bag of words		1		1					1		

Representing Text

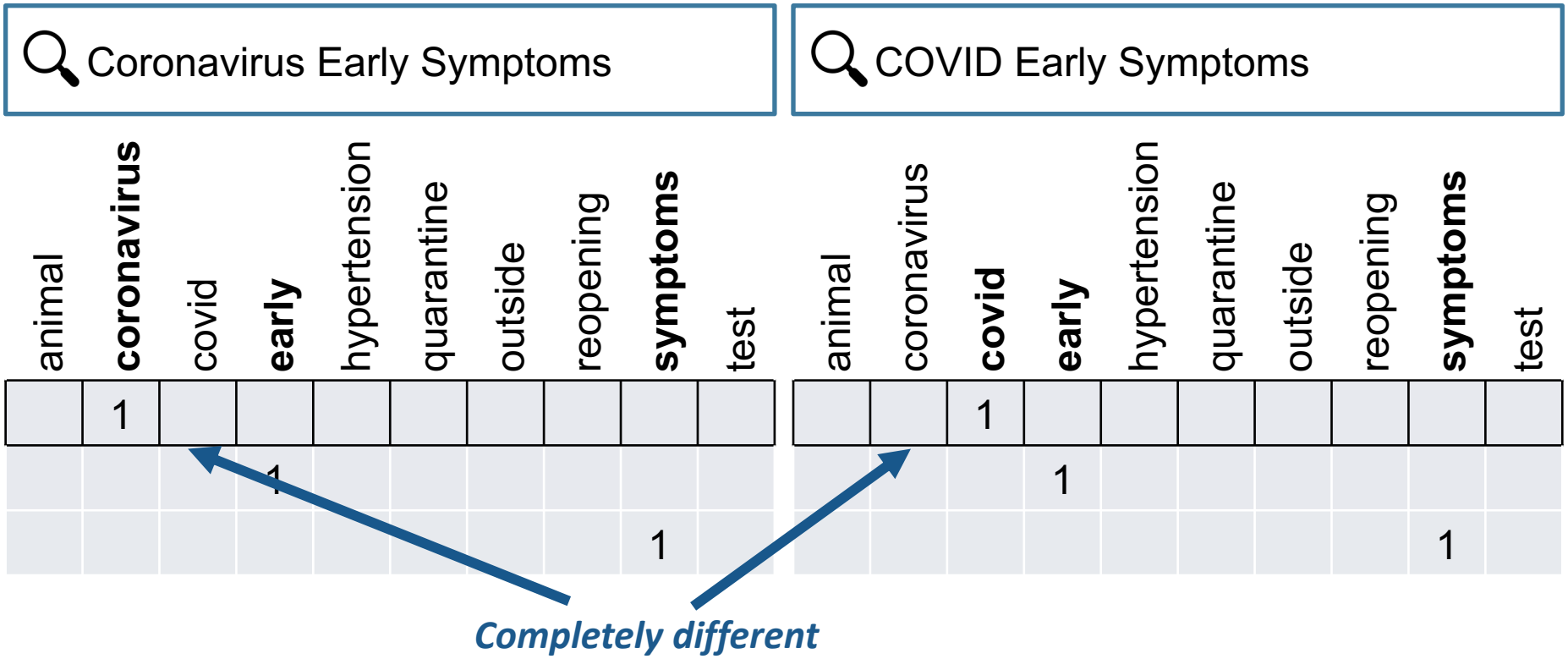
Option 1: One-hot encoding

Coronavirus Early Symptoms

	animal	**coronavirus**	covid	**early**	hypertension	quarantine	outside	reopening	**symptoms**	test	...
		1									
Sequence				1							
									1		

Representing Text

Problem: no relationship between words



✗ Doesn't handle words with similar meanings

*"**Coronavirus**" has a completely different vector than "**COVID**"*

Representing Text

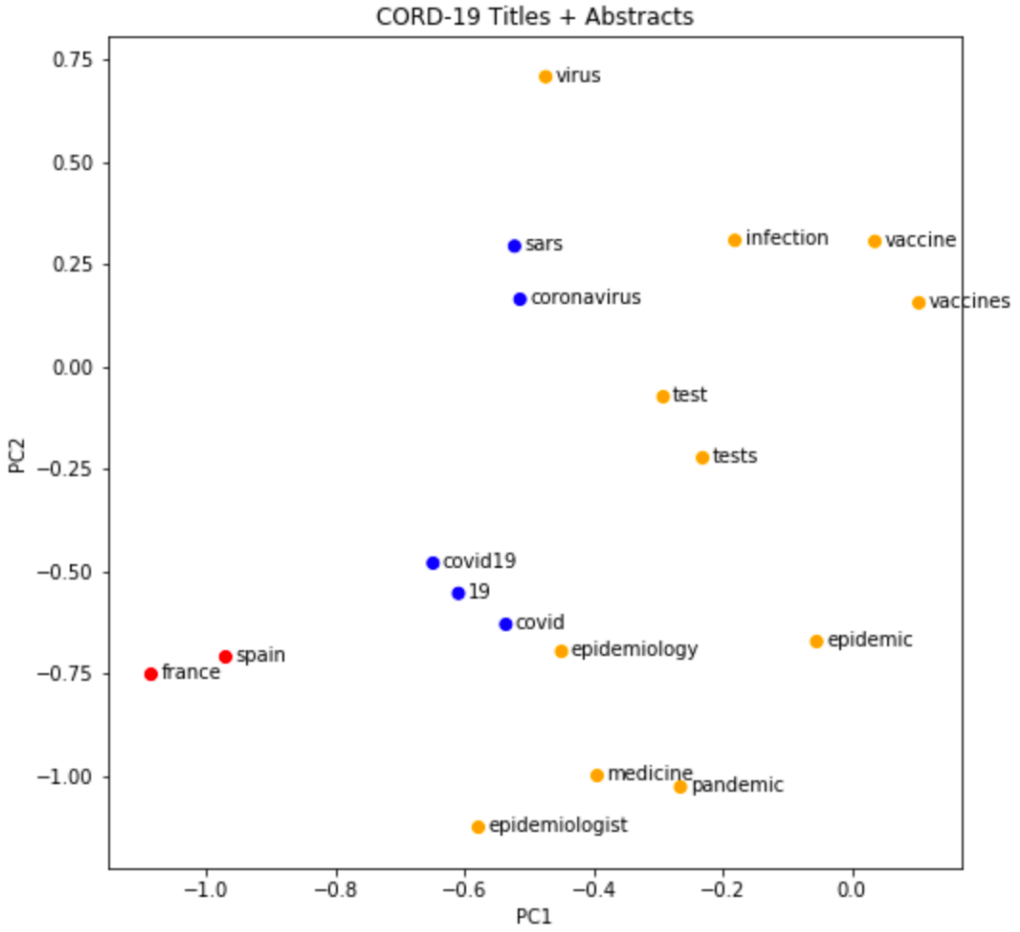
Word Vectors: Map each word to a dense vector

covid =

```
[0.60586,
 0.04596,
 0.12191,
-0.18414,
-0.04422,
 0.13495,
 0.31471,
 0.33992,
 0.01285,
-0.18592,
-0.43352,
-0.62741,
 0.24341,
 0.07149,
...]
```

coronavirus =

```
[0.33853,
-0.27798,
 0.08317,
-0.19729,
-0.49235,
 0.26514,
 0.03004,
 0.25704,
-0.38031,
-0.32722,
-0.47273,
-0.01596,
 0.32322,
-0.04947,
...]
```

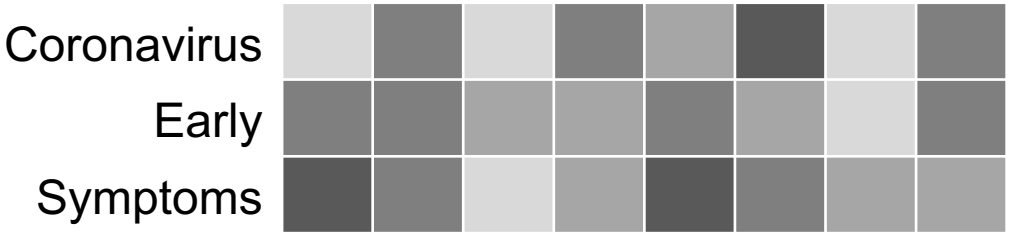


Representing Text

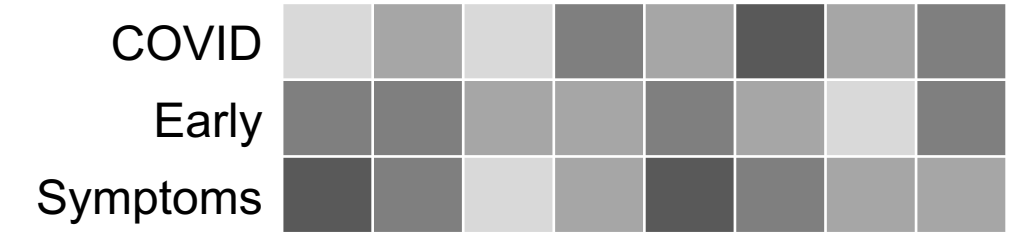
Università di Pisa | University of Glasgow

Word Vectors: Map each word to a dense vector

Coronavirus Early Symptoms

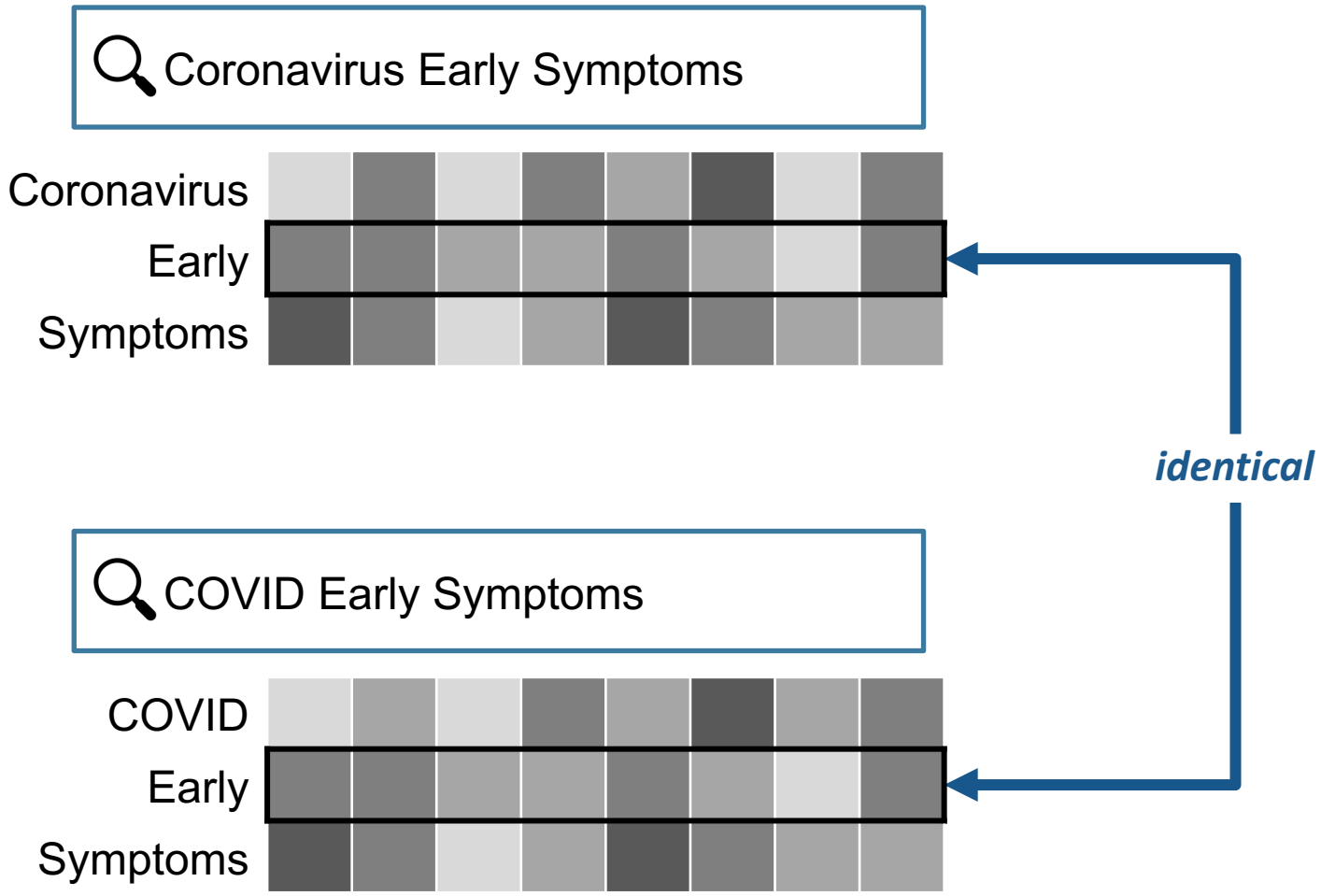


COVID Early Symptoms



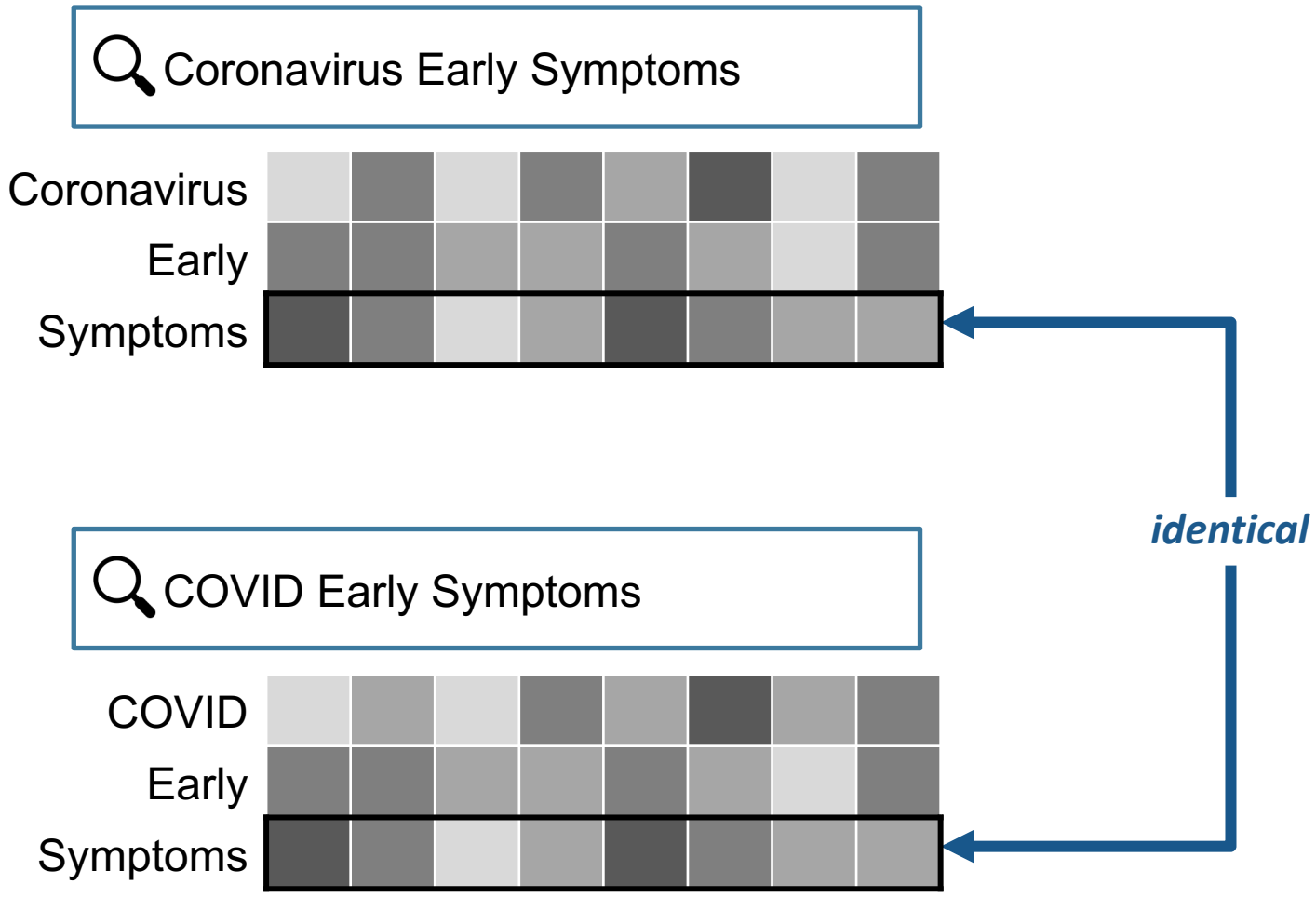
Representing Text

Word Vectors: Map each word to a dense vector



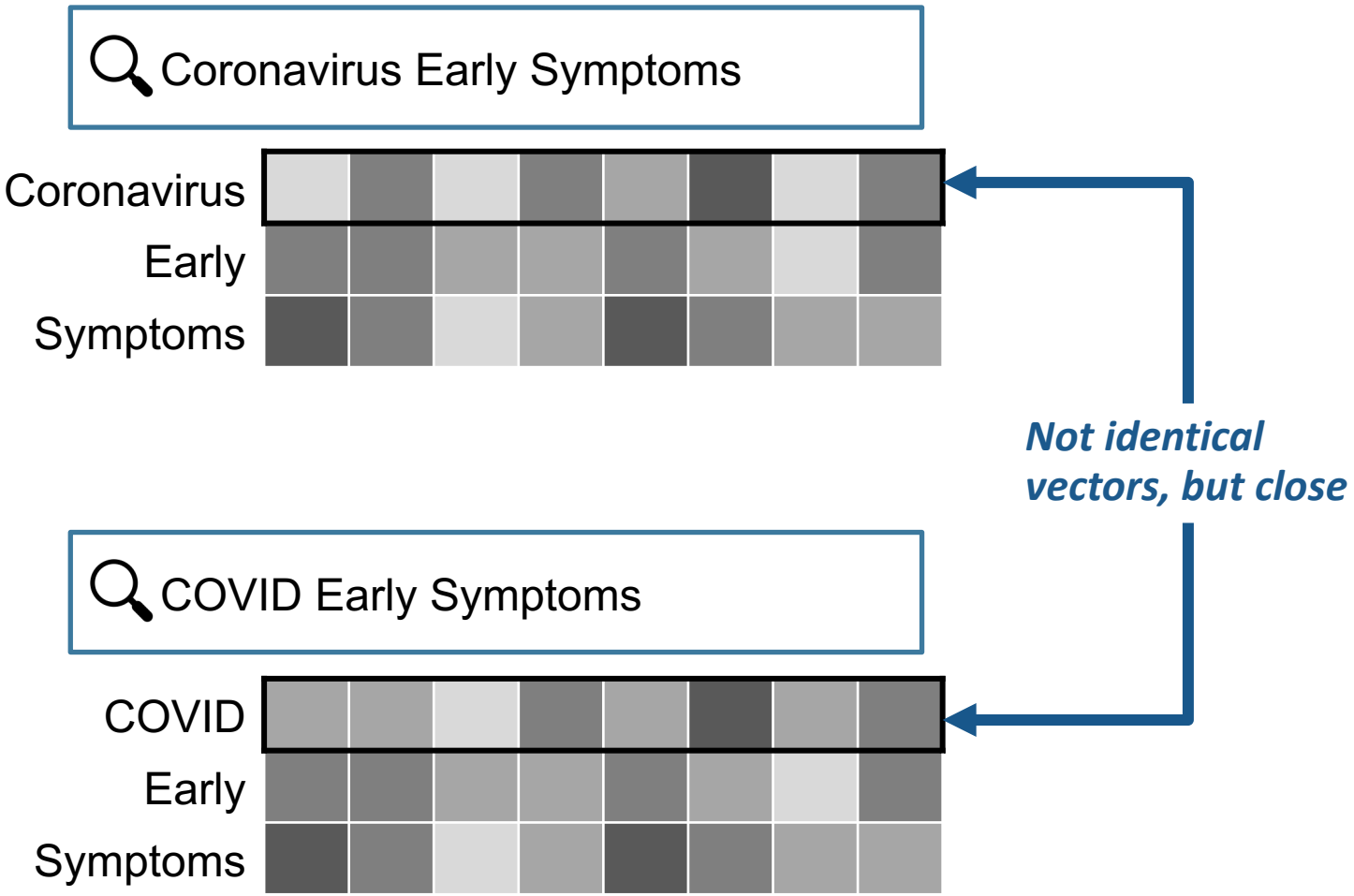
Representing Text

Word Vectors: Map each word to a dense vector



Representing Text

Word Vectors: Map each word to a dense vector



Representing Text

Building word vectors

- Based on word co-occurrences
- Trained using neural network
- e.g. word2vec, gloVe, etc.
- Each word assigned its own vector

Problem: Not Context-Aware

✅ Handles different words with similar meanings

"**Coronavirus**" *has a similar vector to* "**COVID**"

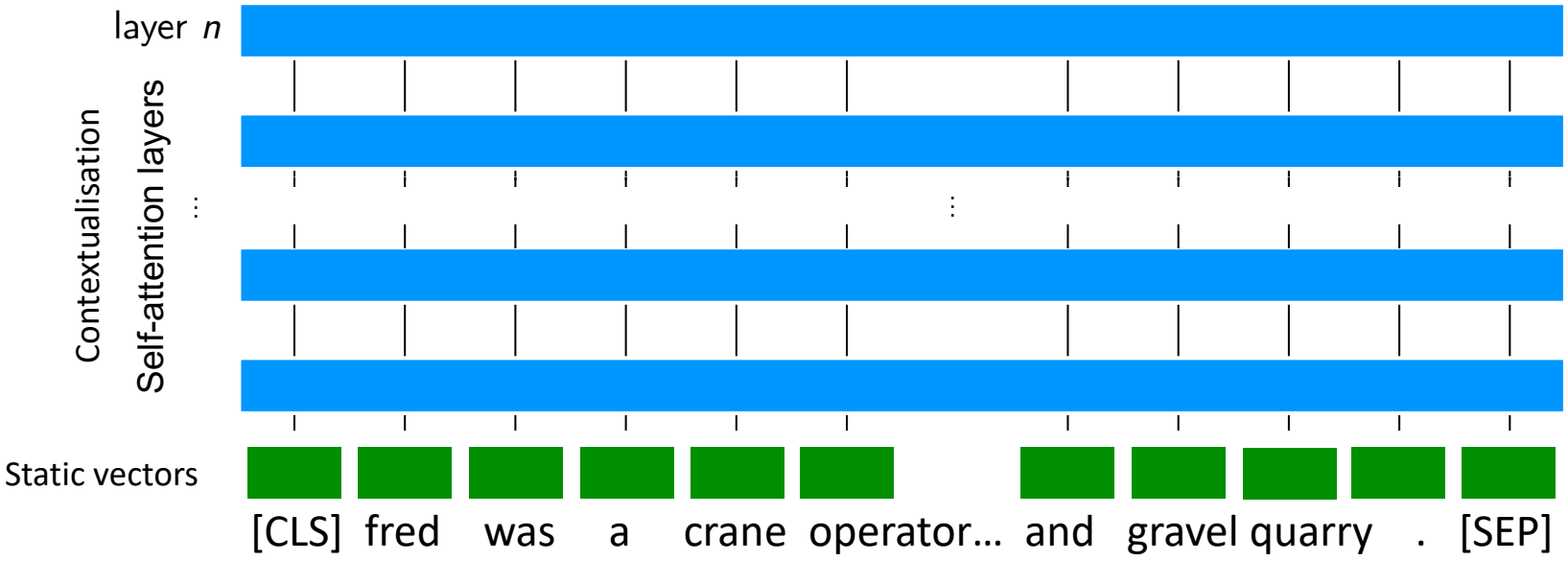

❌ Doesn't handle a single word with multiple possible meanings

"Fred was a **crane** operator at the Slade Rock and Gravel Quarry."

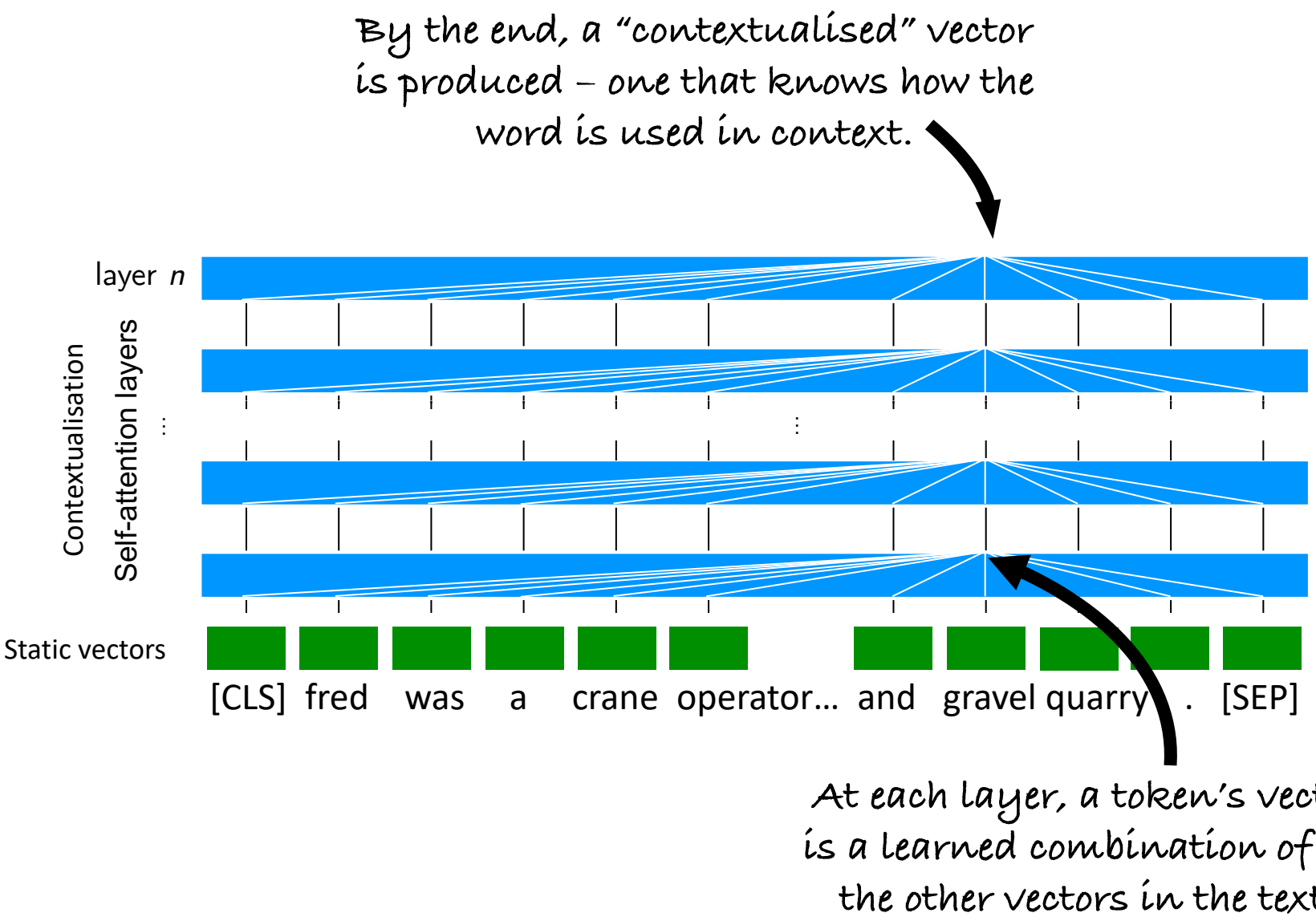
has the same vector as

"Learn how to make an origami **crane** with this origami video."

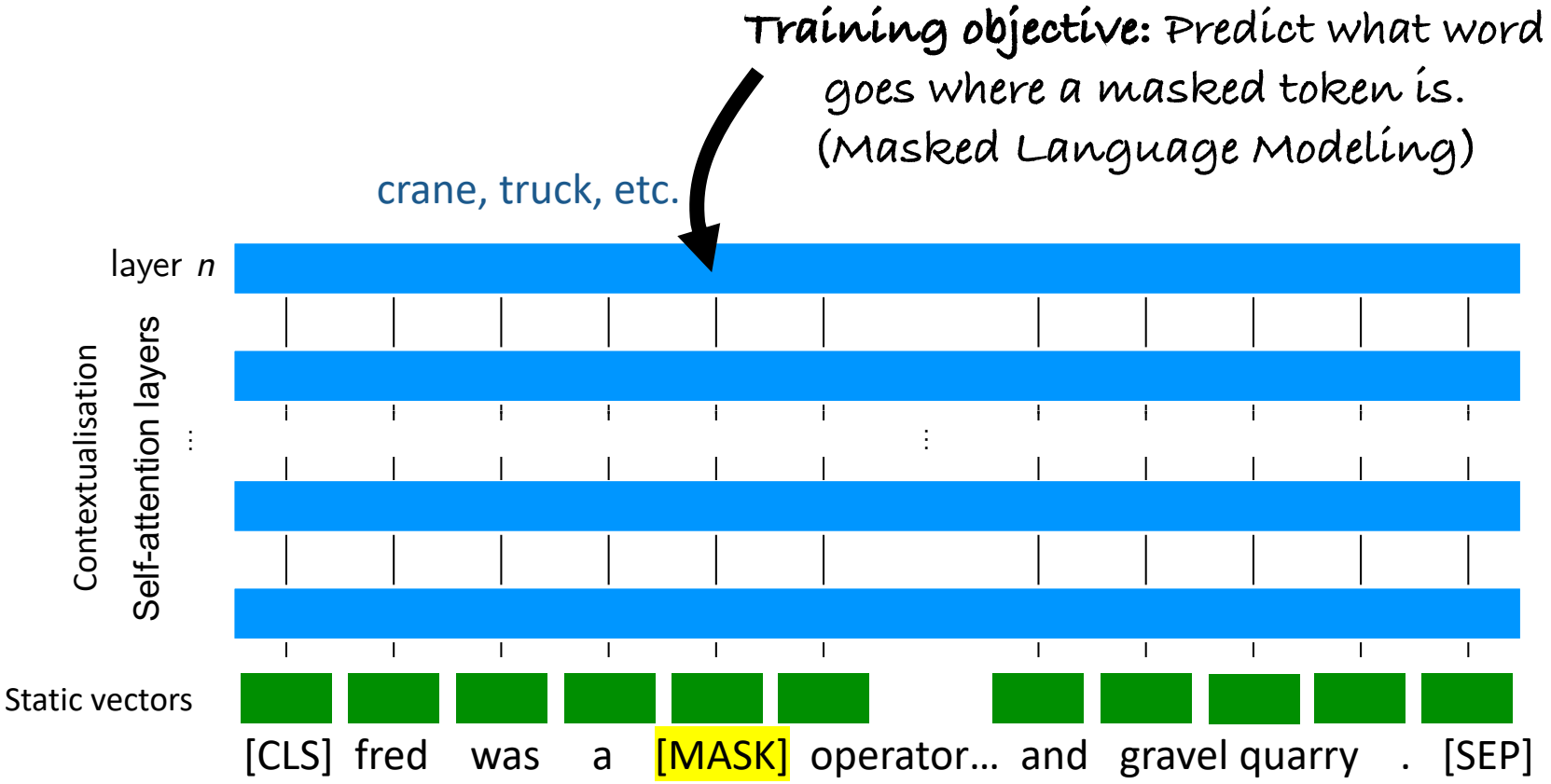
BERT & Transformer Networks



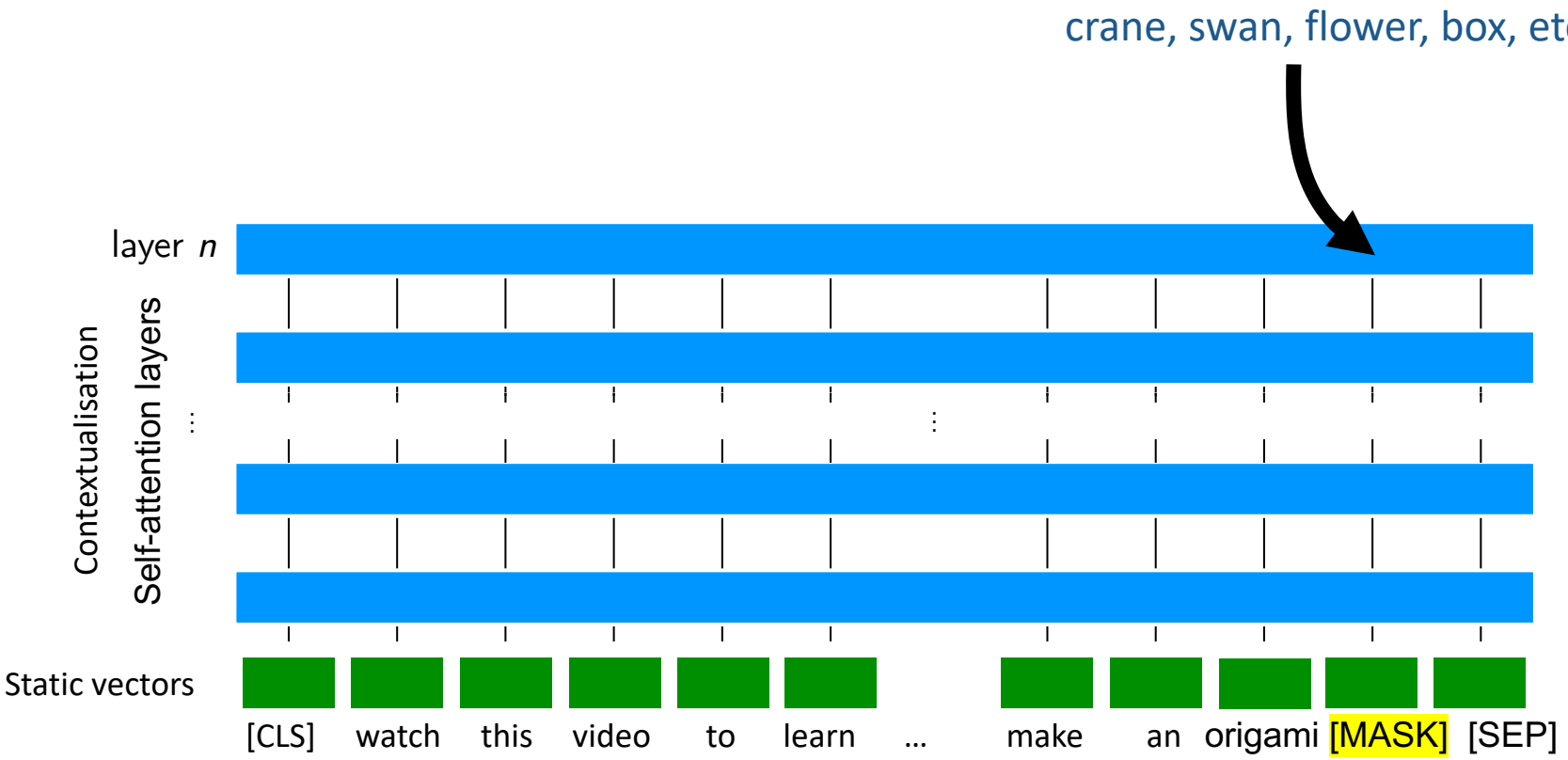
BERT & Transformer Networks



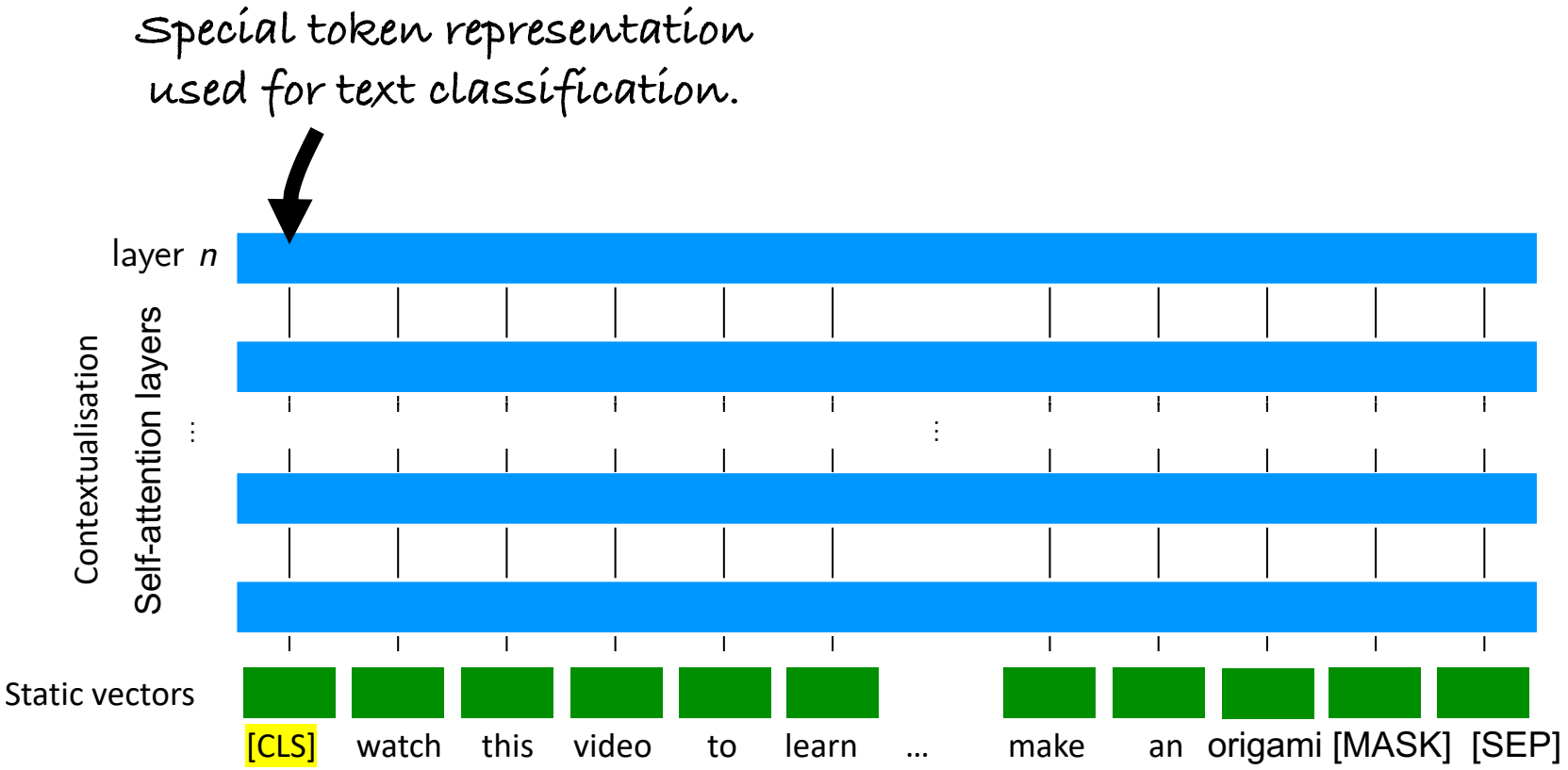
BERT & Transformer Networks



BERT & Transformer Networks



BERT & Transformer Networks



Context-Aware Embeddings

✅ Handles different words with similar meanings

"**Coronavirus**" *has a similar vector to* "**COVID**"


✅ Handle a single word with multiple possible meanings

"Fred was a **crane** operator at the Slade Rock and Gravel Quarry."

has a different vector vector than

"Learn how to make an origami **crane** with this origami video."

Part 2B

NEURAL RE-RANKING MODELS

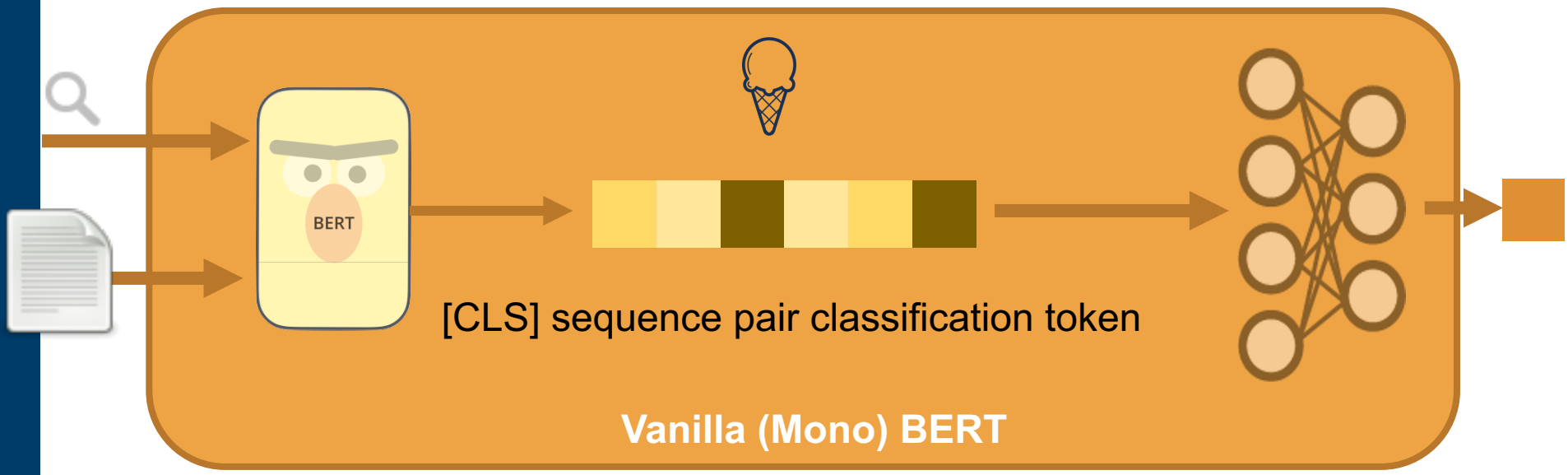
Neural Re-Ranking Models

Università di Pisa | University of Glasgow

General Techniques:

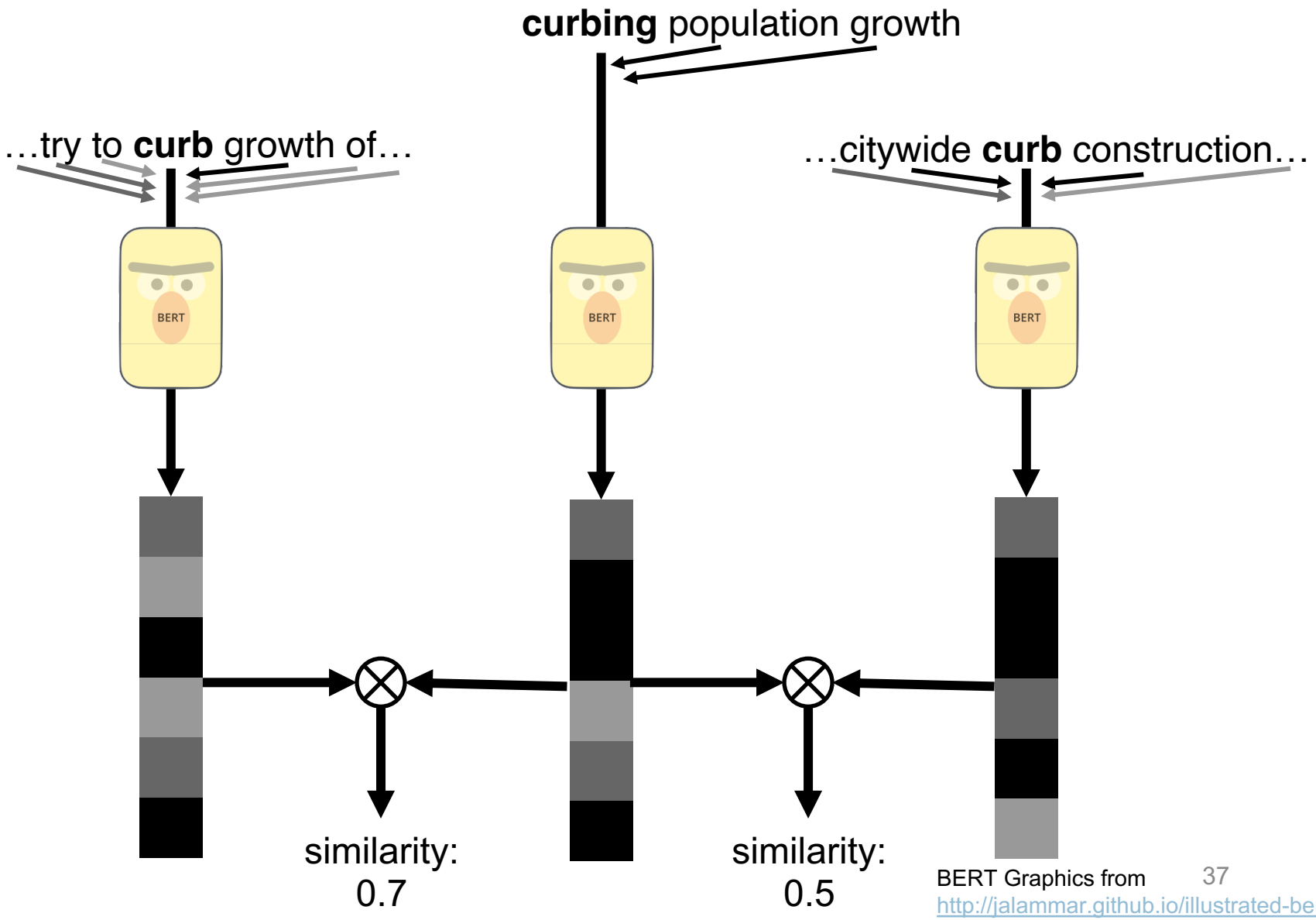
- **Using classification [CLS] mechanism**
- **Using decoder model**
- **Using contextualised word embedding similarity**
- **Building representations**

Vanilla BERT



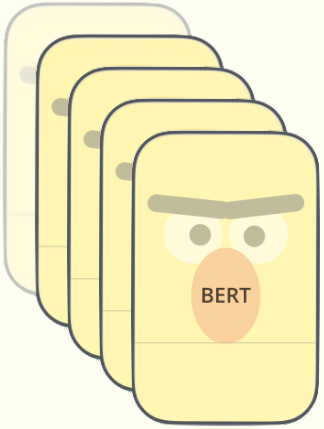
BERT Graphics from Jay Alammar, http://jalammar.github.io/illustrated-bert/

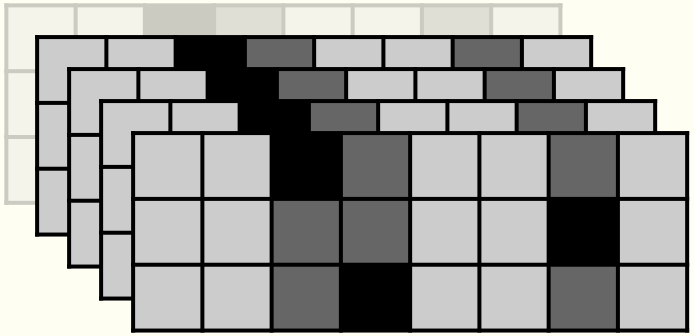
Contextualized Word Vectors



BERT Graphics from http://jalammar.github.io/illustrated-bert/

BERT consists of multiple layers. We build a similarity matrix for each layer.



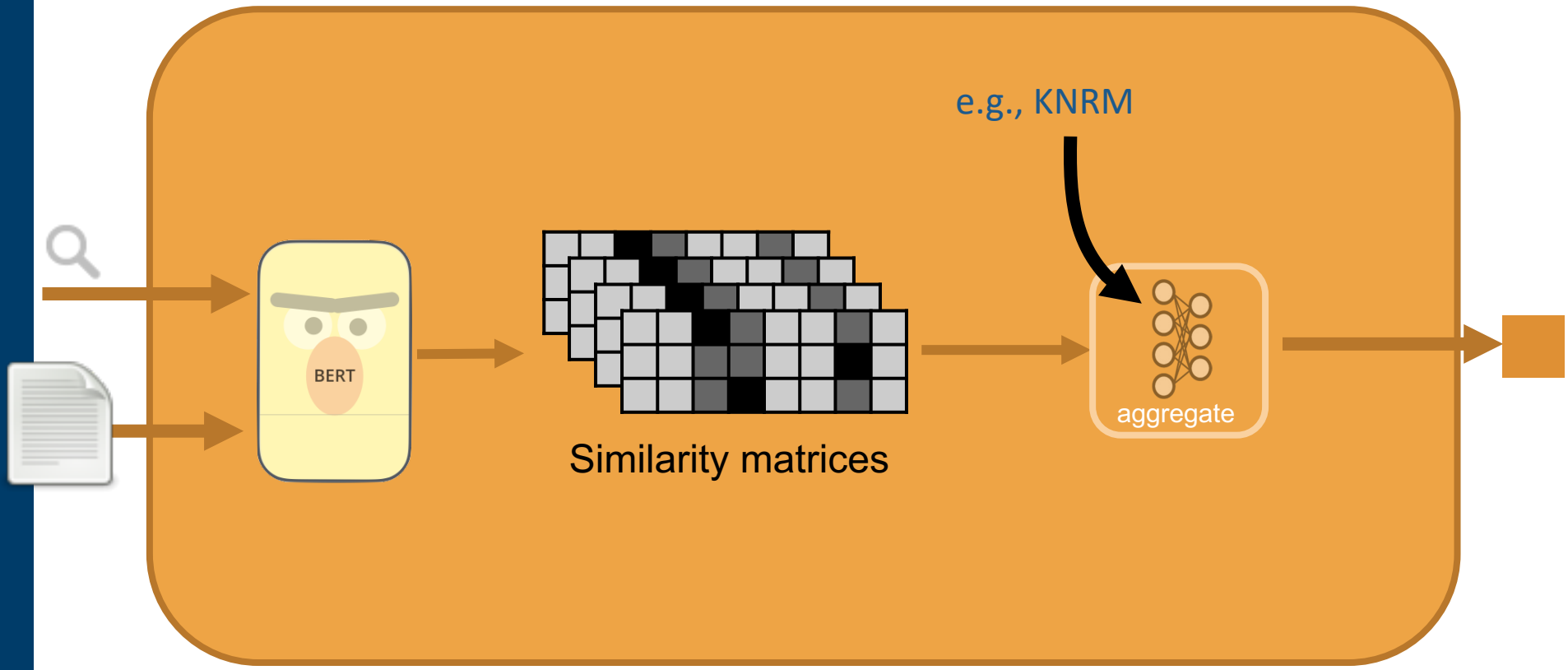


$BERT_{BASE}$: 13 Layers

* Including 1 input layer (non-transformed)

BERT Graphics from Jay Alammar, http://jalammar.github.io/illustrated-bert/

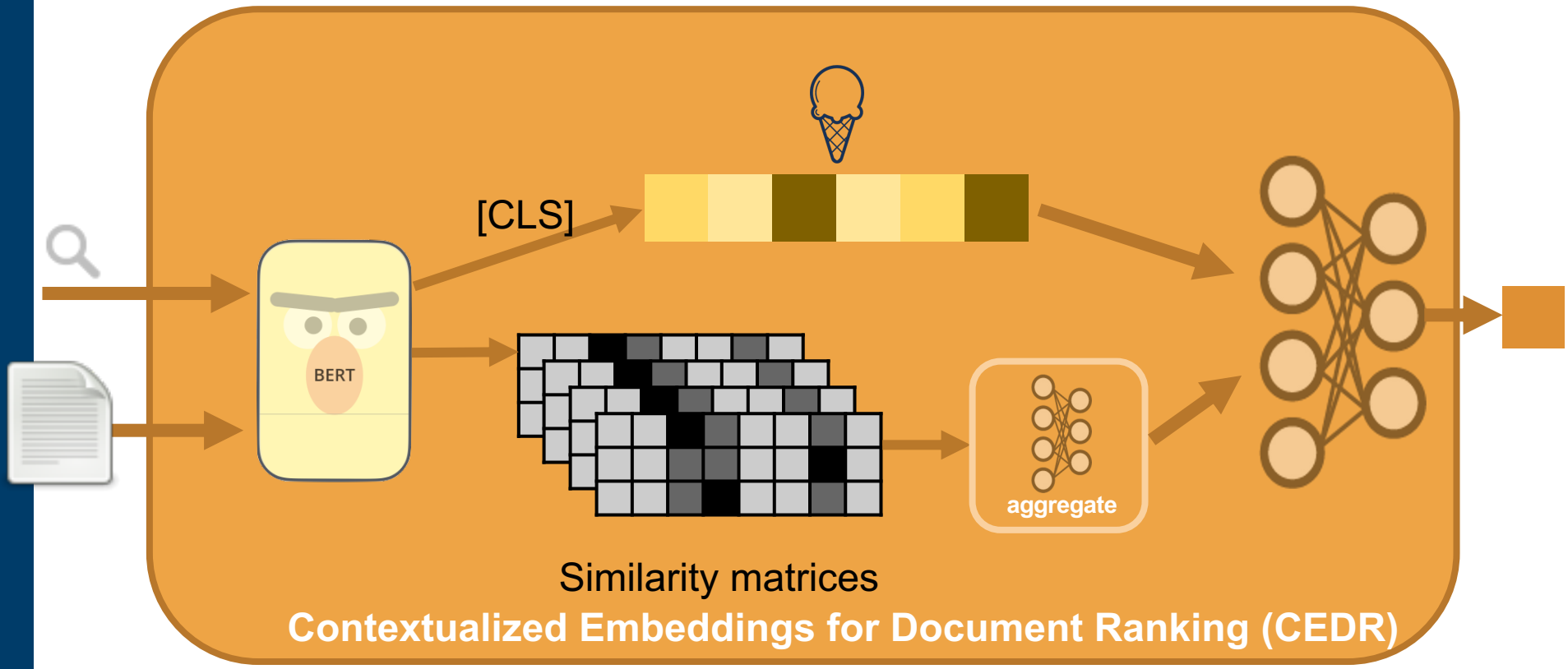
Contextualized Word Vectors



MacAvaney et al. CEDR. SIGIR 2019.

BERT Graphics from Jay Alammar, http://jalammar.github.io/illustrated-bert/

CEDR



MacAvaney et al. CEDR. SIGIR 2019.

BERT Graphics from Jay Alammar, http://jalammar.github.io/illustrated-bert/

BERT Practical

Università di Pisa | University of Glasgow

Vanilla model that uses BERT's [CLS] for ranking

```
vbert = onir_pt.reranker('vanilla_transformer', 'bert', text_field='title_abstract')
cedr_knrm = onir_pt.reranker('cedr_knrm', 'bert', text_field='title_abstract')
```

CEDR KNRM model (using both KNRM and [CLS])

	name	map	ndcg	ndcg_cut_10	mrt
0	BM25	0.077880	0.177728	0.644374	36.795420
1	BM25 >> BERT	0.065846	0.160599	0.416803	1847.491717
2	BM25 >> BERT KNRM	0.059786	0.151265	0.277798	1906.480065
3	BM25 >> CEDR KNRM	0.067430	0.165405	0.507656	1917.520533

You still need to train these models!

BERT Practical

Università di Pisa | University of Glasgow

Load trained model from URL

```
bert = onir_pt.reranker.from_checkpoint('https://macavaney.us/scibert-medmarco.tar.gz',
    text_field='title_abstract',
    expected_md5='854966d0b61543ffffa44cea627ab63b')
```

```
def cat_title_abstract(df):
    df['title_abstract'] = df['title'].str.cat(df['abstract'], sep=' ')
    return df

bert_pipeline = (br >>
                 pt.text.get_text(dataset, ['title', 'abstract']) >>
                 pt.apply.generic(cat_title_abstract) >>
                 bert)
```

Include both the title and abstract

```
pt.Experiment(
    [br, bert_pipeline],
    dataset.get_topics('title'),
    dataset.get_qrels(),
    names=['BM25', 'BM25 >> BERT'],
    eval_metrics=["map", "ndcg", 'ndcg_cut.10', 'mrt']
)
```

TREC COVID results

Title queries

	name	map	ndcg	ndcg_cut_10	mrt
0	BM25	0.073623	0.162657	0.583665	29.487896
1	BM25 >> BERT	0.077678	0.168394	0.650975	1637.205799

Description queries

	name	map	ndcg	ndcg_cut_10	mrt
0	BM25	0.077880	0.177728	0.644374	38.557969
1	BM25 >> BERT	0.085371	0.185821	0.740331	1748.576758

Causal Language Models

BERT is trained with a Masked Language Modeling (MLM) objective.

- Words are masked out in the input and it's trained to fill in the missing words

T5, GPT, etc. are trained with a Causal Language Modeling (CLM) objective.

- Models learn to predict the next word in a sequence.
- This allows them to be used in text generation settings.

Scoring with a CLM

Università di Pisa | University of Glasgow

Main idea: Train a model to generate the text "true" or "false", prompted with the query and document:

> **Query:** coronavirus early symptoms **Document:** Nidovirus subgenomic mRNAs contain a leader sequence derived from the 5′ end of the genome fused to different sequences ('bodies') derived from the 3′ end. Their generation involves a unique mechanism of discontinuous subgenomic RNA synthesis that resembles copy-choice RNA recombination. During this process, the nascent RNA strand is transferred from one site in the template to another, during either plus or minus… **Relevant:**

Ask the model: Should the next word be "true" or "false"?

$$P(rel \mid q, d) \approx P(\text{“}true\text{”} \mid \text{“}Query\text{:”}\ \$q\ \text{“}Document\text{:”}\ \$d\ \text{“}Relevant\text{:”})$$

Train on relevant and non-relevant pairs.

Nogueira et al. Document Ranking with a Pretrained Sequence-to-Sequence Model. arXiv 2020.

MonoT5 Practical

Università di Pisa | University of Glasgow

```
from pyterrier_t5 import MonoT5ReRanker
monoT5 = MonoT5ReRanker()
```

MonoT5 is already implemented by the pyterrier_t5 package

```
br = pt.BatchRetrieve(indexref, wmodel='BM25') % 100
pt.Experiment(
    [br, br >> pt.text.get_text(dataset, 'text') >> monoT5],
    dataset.get_topics(),
    dataset.get_qrels(),
    names=['BM25', 'BM25 >> monoT5'],
    eval_metrics=[MAP, RR, P@10, nDCG@10]
)
```

monoT5 batches: 100% 2325/2325 [01:36<00:00, 24.00it/s]

	name	map	recip_rank	P_10	ndcg_cut_10
0	BM25	0.272523	0.725587	0.352688	0.446609
1	BM25 >> monoT5	0.295071	0.750839	0.413978	0.490703

University of Glasgow | Università di Pisa

SEARCH SOLUTIONS London, UK

Part 2C

EFFICIENCY

Problem: Efficiency

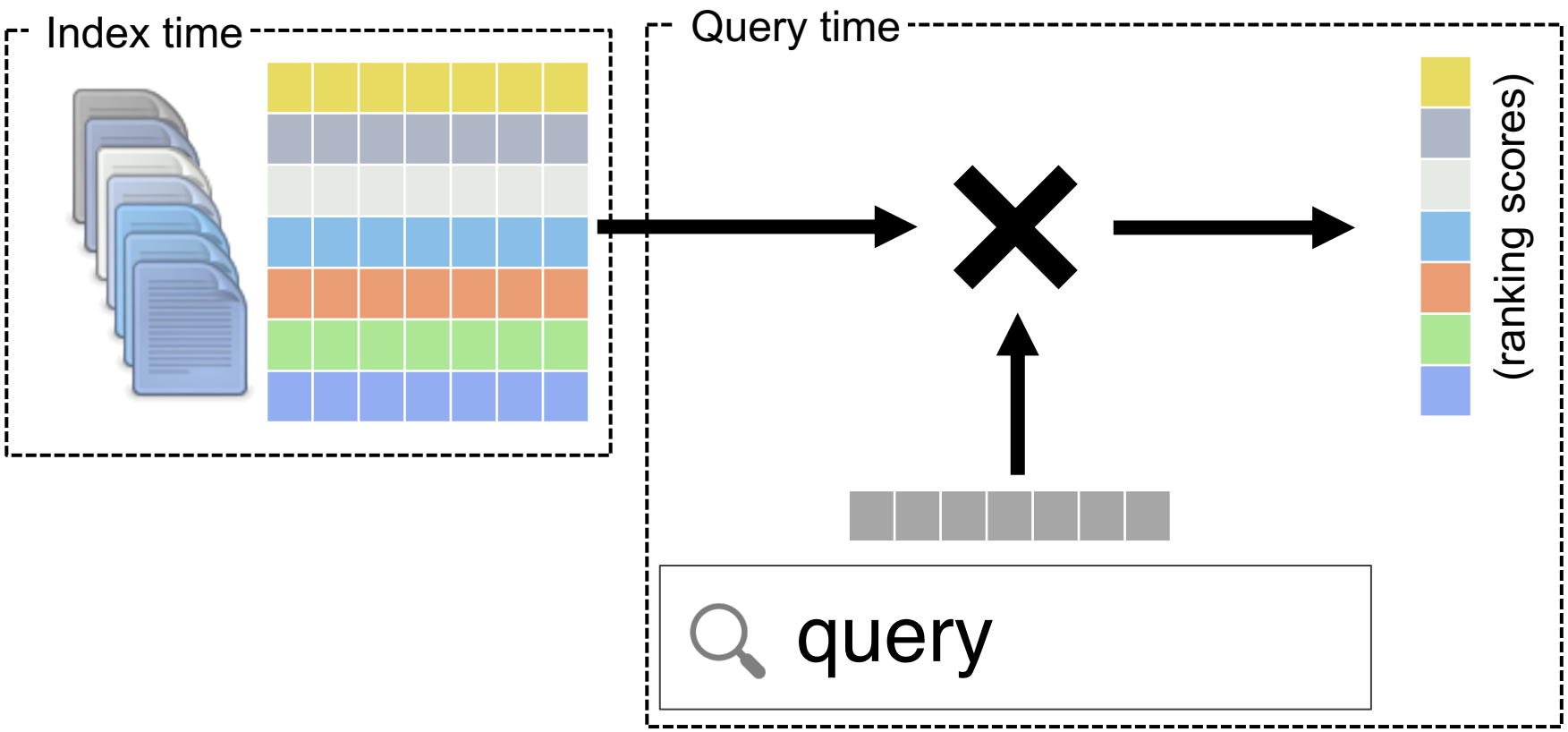
Università di Pisa | University of Glasgow

Using methods like BERT and T5 for ranking is effective, but also very slow.

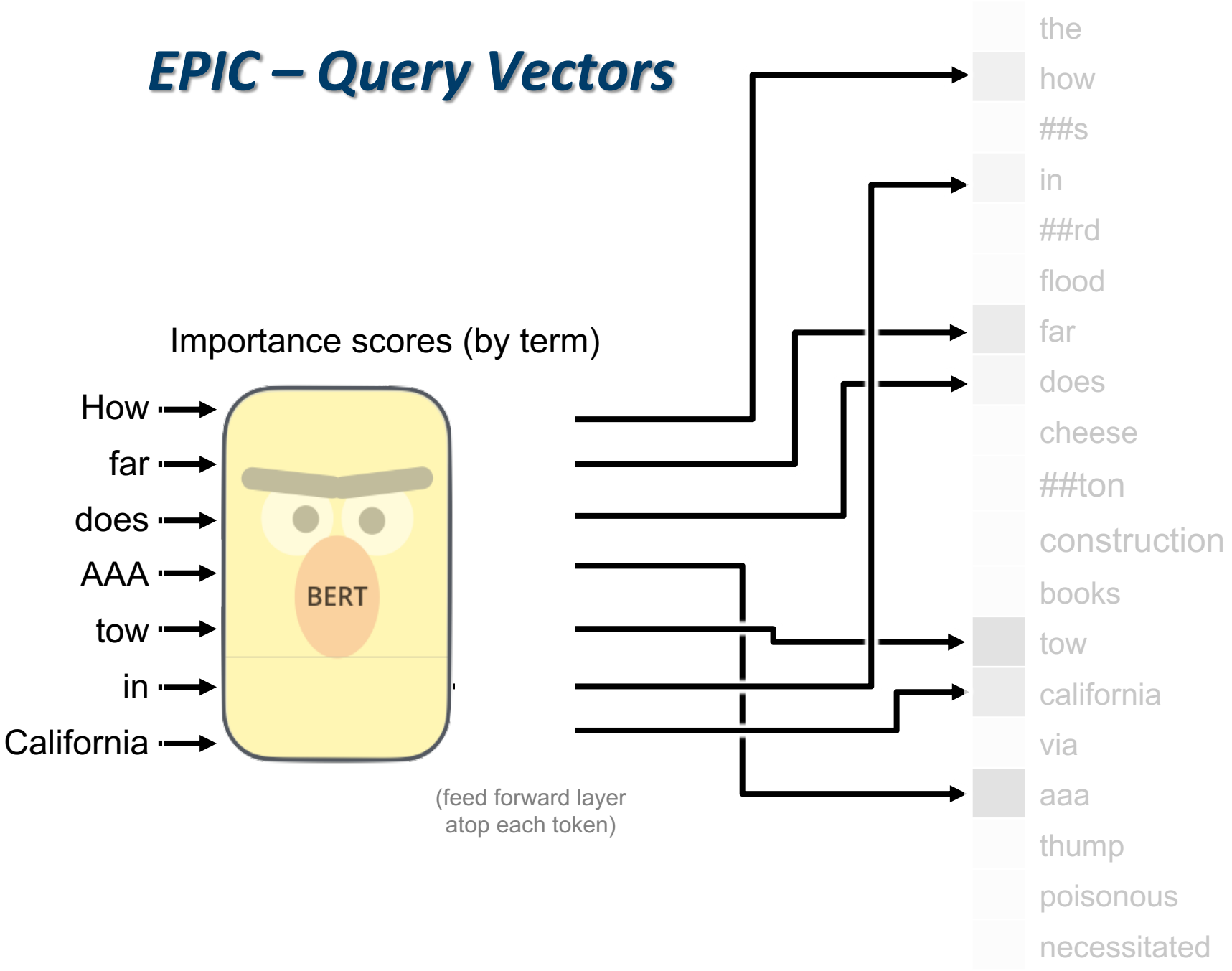
E.g., BERT takes 45x longer than BM25: (1.7 seconds)

	name	map	ndcg	ndcg_cut_10	mrt
0	BM25	0.077880	0.177728	0.644374	38.557969
1	BM25 >> BERT	0.085371	0.185821	0.740331	1748.576758

Pre-Computing Document Representations

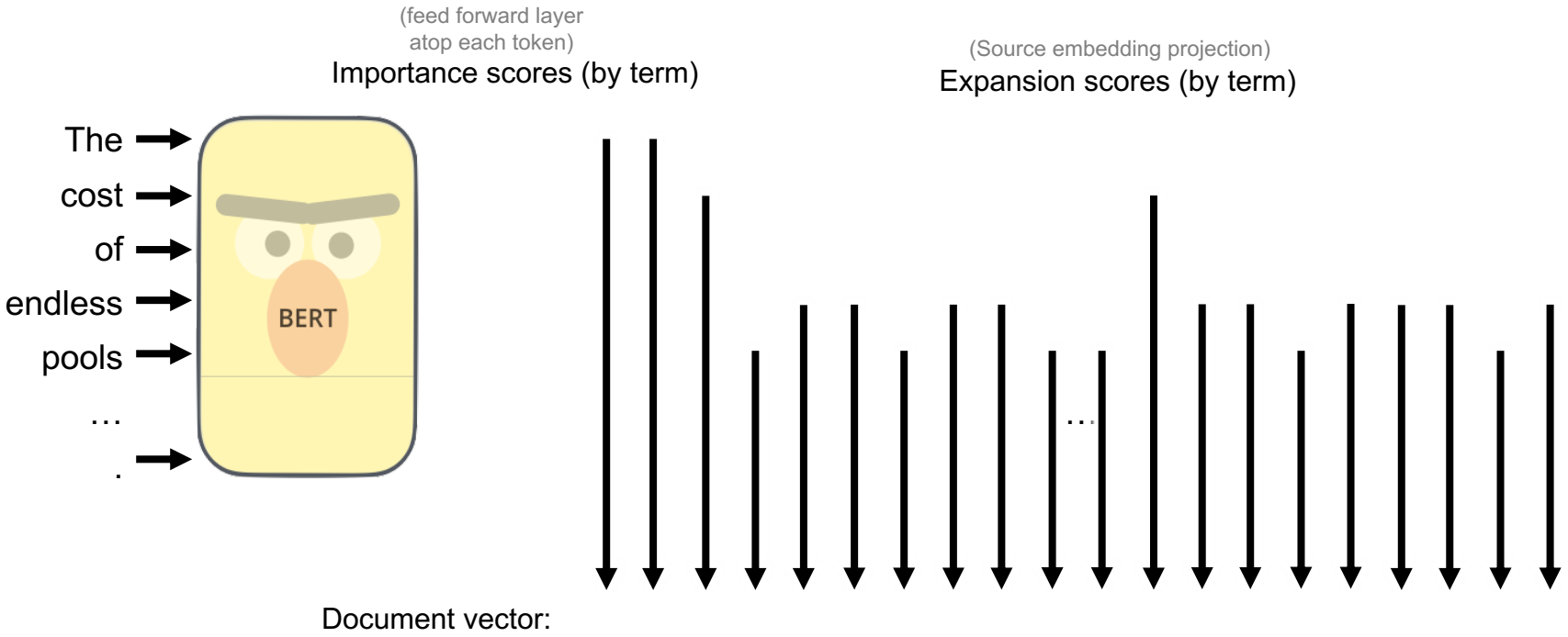


EPIC – Query Vectors



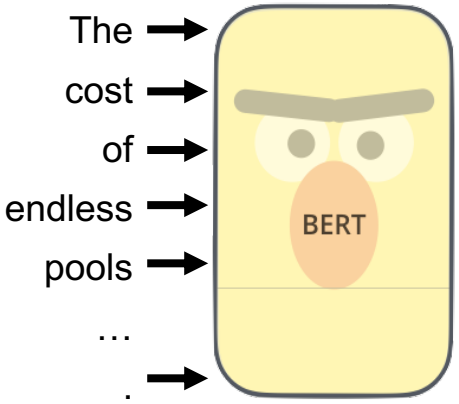
BERT Graphics from Jay Alammar, http://jalammar.github.io/illustrated-bert/

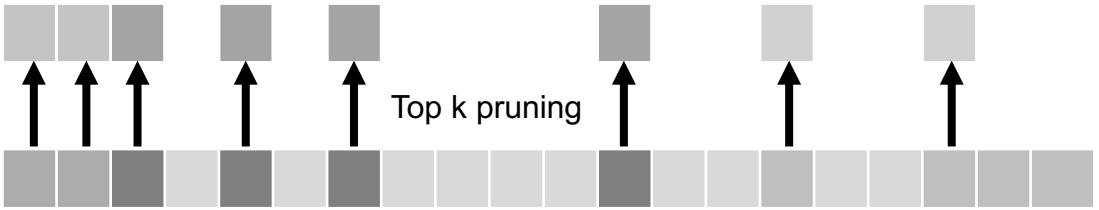
EPIC – Document Vectors



BERT Graphics from Jay Alammar, http://jalammar.github.io/illustrated-bert/

EPIC – Document Vectors





BERT Graphics from Jay Alammar, http://jalammar.github.io/illustrated-bert/

EPIC Practical

Lazy EPIC (not indexed)

```
# Load a version of EPIC trained on the MS-MARCO dataset
lazy_epic = onir_pt.reranker.from_checkpoint(
    'https://macavaney.us/epic.msmarco.tar.gz',
    expected_md5="2f6a16be1a6a63aab1e8fed55521a4db")
```

```
br = pt.BatchRetrieve(index) % 30
pipeline = (br >> pt.text.get_text(dataset, 'abstract')
               >> pt.apply.generic(lambda x: x.rename(columns={'abstract': 'text'}))
               >> lazy_epic)
pt.Experiment(
    [br, pipeline],
    dataset.get_topics('title'),
    dataset.get_qrels(),
    names=['DPH', 'DPH >> EPIC (lazy)'],
    eval_metrics=["recip_rank", "P.5", "mrt"]
)
```

	name	recip_rank	P_5	mrt
0	DPH	0.766833	0.684	36.734074
1	DPH >> EPIC (lazy)	0.817889	0.724	801.524438

EPIC Practical

EPIC (indexed)

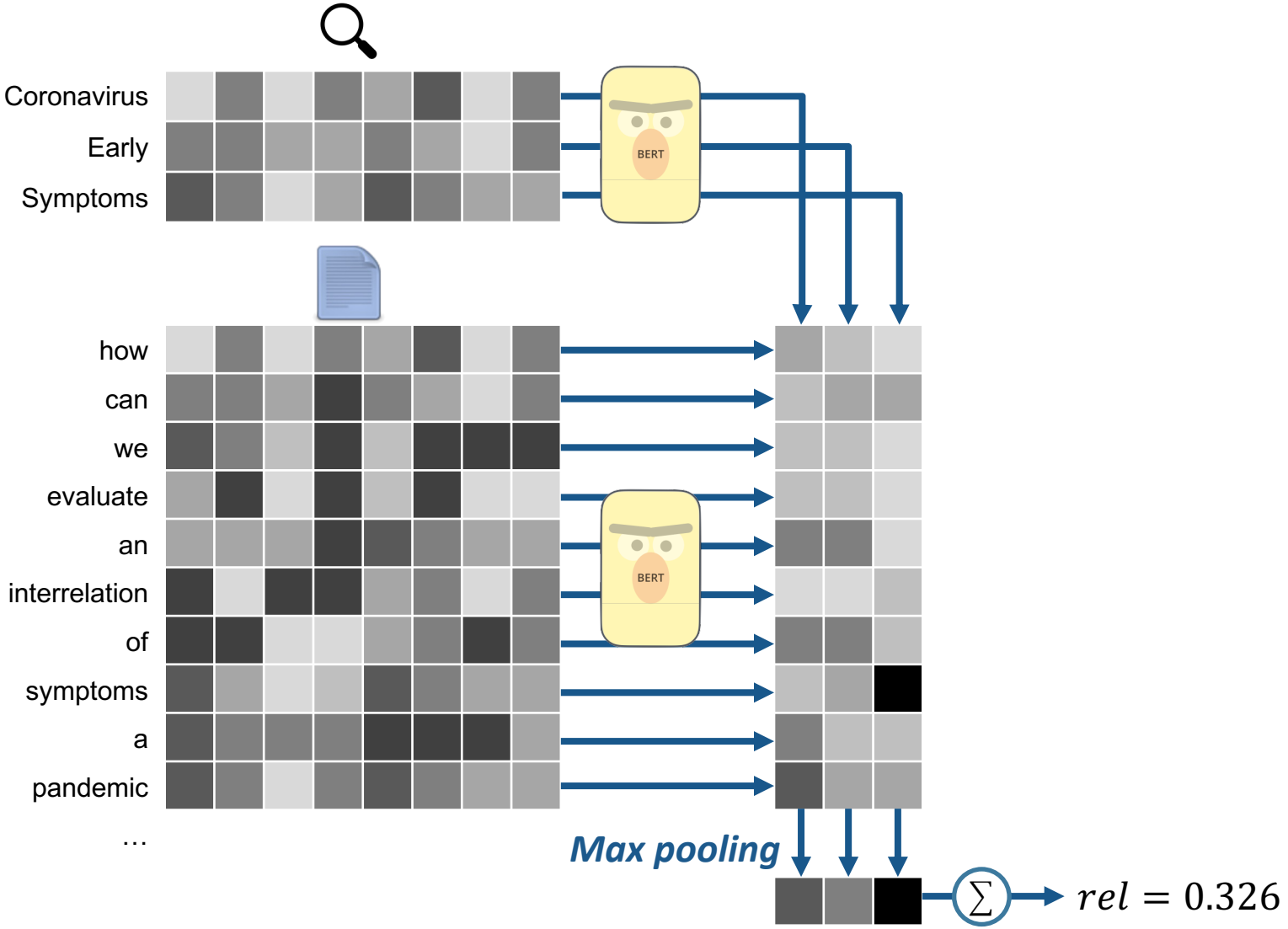
```
indexed_epic = onir_pt.indexed_epic.from_checkpoint(
    'https://macavaney.us/epic.msmarco.tar.gz',
    index_path='./epic_cord19')
```

```
indexed_epic.index(dataset.get_corpus_iter(), fields=('abstract',))
```

```
pipeline = br >> indexed_epic.reranker()
pt.Experiment(
    [br, pipeline],
    dataset.get_topics('title'),
    dataset.get_qrels(),
    names=["DPH", "DPH >> EPIC (indexed)"],
    eval_metrics=["recip_rank", "P.5", "mrt"]
)
```

	name	recip_rank	P_5	mrt
0	DPH	0.766833	0.684	30.500175
1	DPH >> EPIC (indexed)	0.821500	0.700	53.264584

Contextualized Late Interaction over BERT (ColBERT)



Khattab & Zaharia. ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. SIGIR 2020.

ColBERT Re-Ranking practical

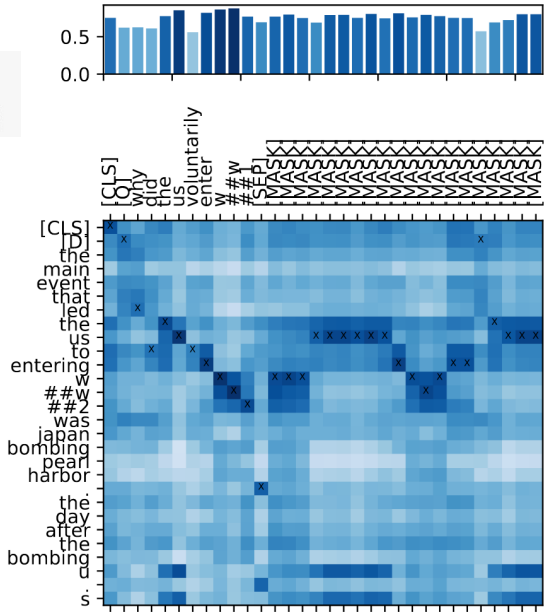
```
colbert_factory = pyterrier_colbert.ranking.ColBERTFactory(
    "http://www.dcs.gla.ac.uk/~craigm/colbert.dnn.zip", None, None)
```

```
colbert = colbert_factory.text_scorer(doc_attr='abstract')
```

```
br = pt.BatchRetrieve(index) % 100
pipeline = br >> pt.text.get_text(dataset, 'abstract') >> colbert
```

	name	map	P_10	ndcg	ndcg_cut_10	mrt
0	DPH	0.068006	0.658	0.165607	0.609058	52.517634
1	DPH >> ColBERT	0.074670	0.752	0.172034	0.690447	615.640942

```
colbert_factory.explain_text(query, text)
```



Macdonald, Tonellotto & Ounis. On Single and Multiple Representations in Dense Passage Retrieval. IIR 2021.

Other Re-Ranking Models

There's a bunch of other great re-ranking models:

- **TK/TKL – Contextualized kernel-based ranking.**
  - Hofstätter et al. Interpretable & time-budget-constrained contextualization for re-ranking. ECAI 2020.
- **PARADE – Handling long documents through passage representation aggregation**
  - Li et al. PARADE: Passage Representation Aggregation for Document Reranking. arXiv 2020.
- **Duet – combine representation and interaction models.**
  - Mitra et al. Learning to match using local and distributed representations of text for web search. WWW 2017.
- **And variants using other models (e.g., ELECTRA, RoBERTa)…**

Limitation: First Stage Retrieval Performance

Goal: maximise **recall** from first stage.

- Limitation: Lexical retrievers like BM25

To improve improve first stage performance:

- Query expansion/rewriting (e.g., RM3)
- Adaptive Re-Ranking ← Now
- Index Augmentation & Dense Retrieval ← Part 3

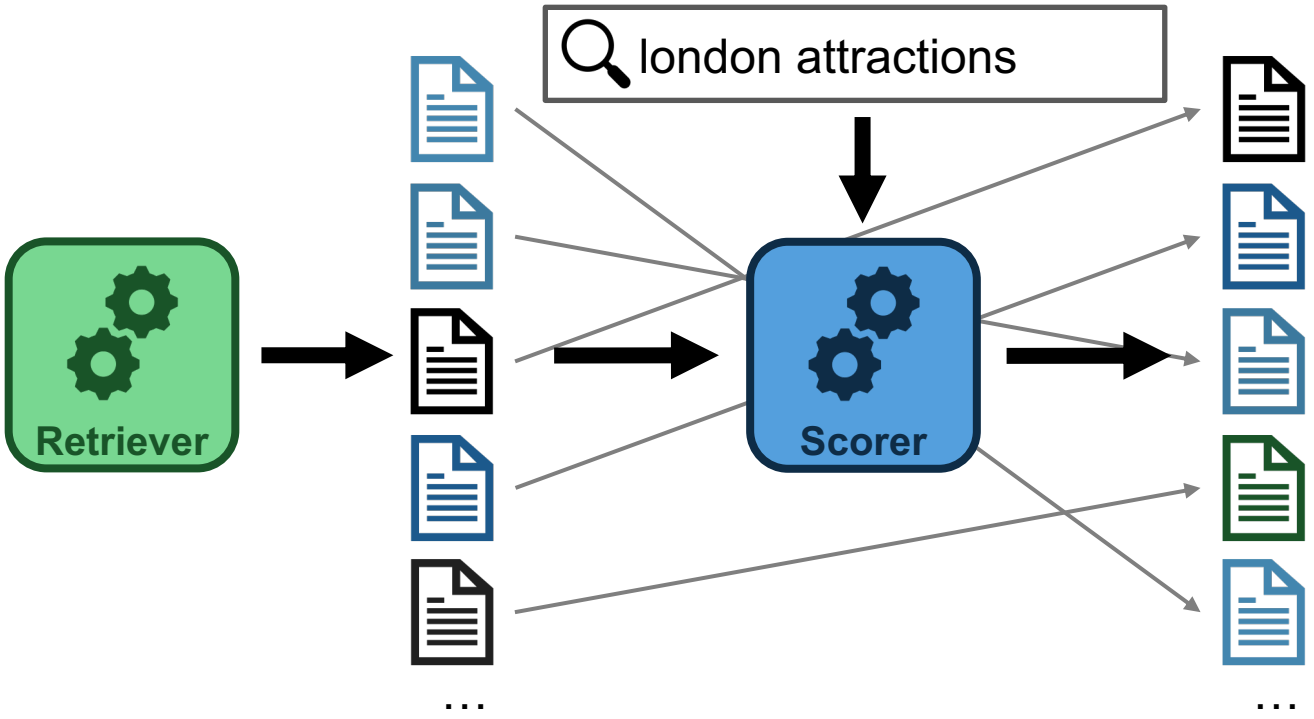
University of Glasgow | Università di Pisa

SEARCH SOLUTIONS London, UK

Part 2D

ADAPTIVE RE-RANKING

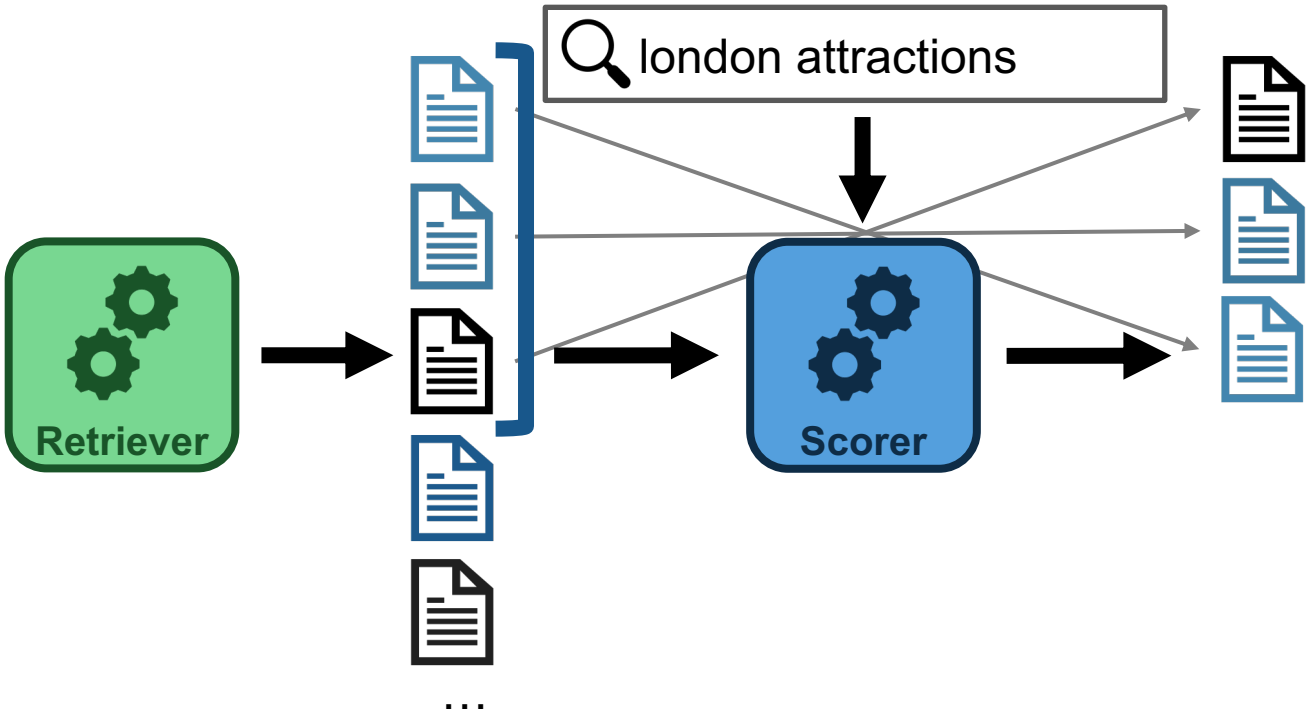
(Non-Adaptive) Re-Ranking



hundreds/thousands of documents

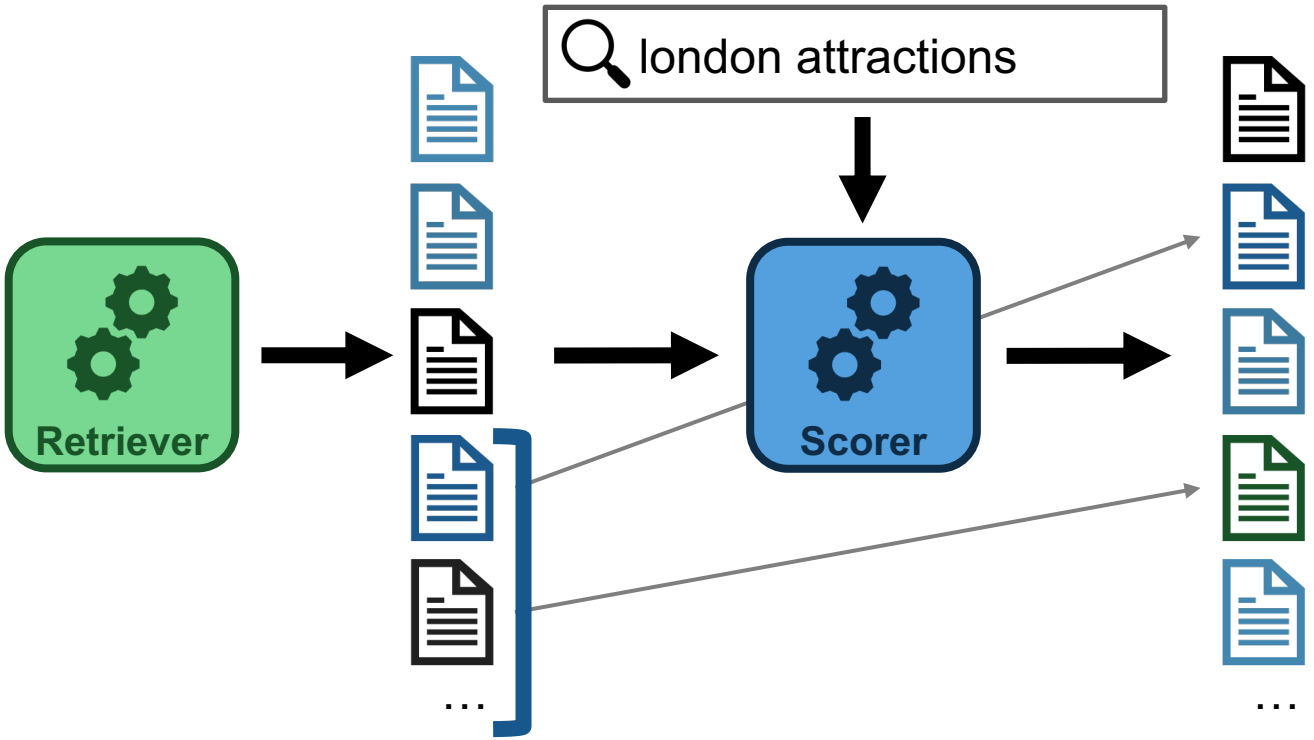
same documents, new order

In practice, scoring is done in batches



hundreds/thousands of documents

In practice, scoring is done in batches



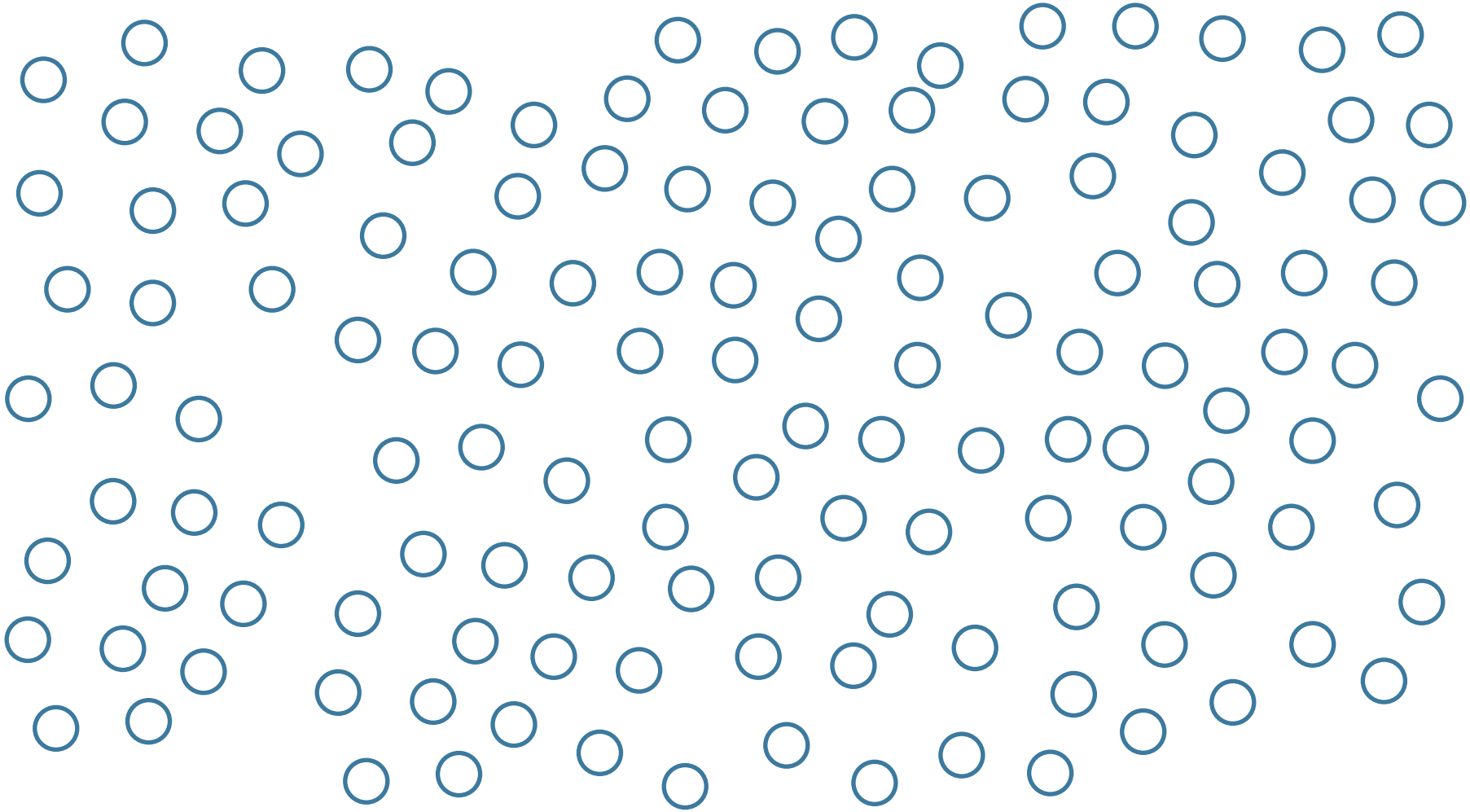
hundreds/thousands of documents

same documents, new order

Adaptive Re-Ranking

Università di Pisa | University of Glasgow

Consider the corpus as a graph – edges between similar docs

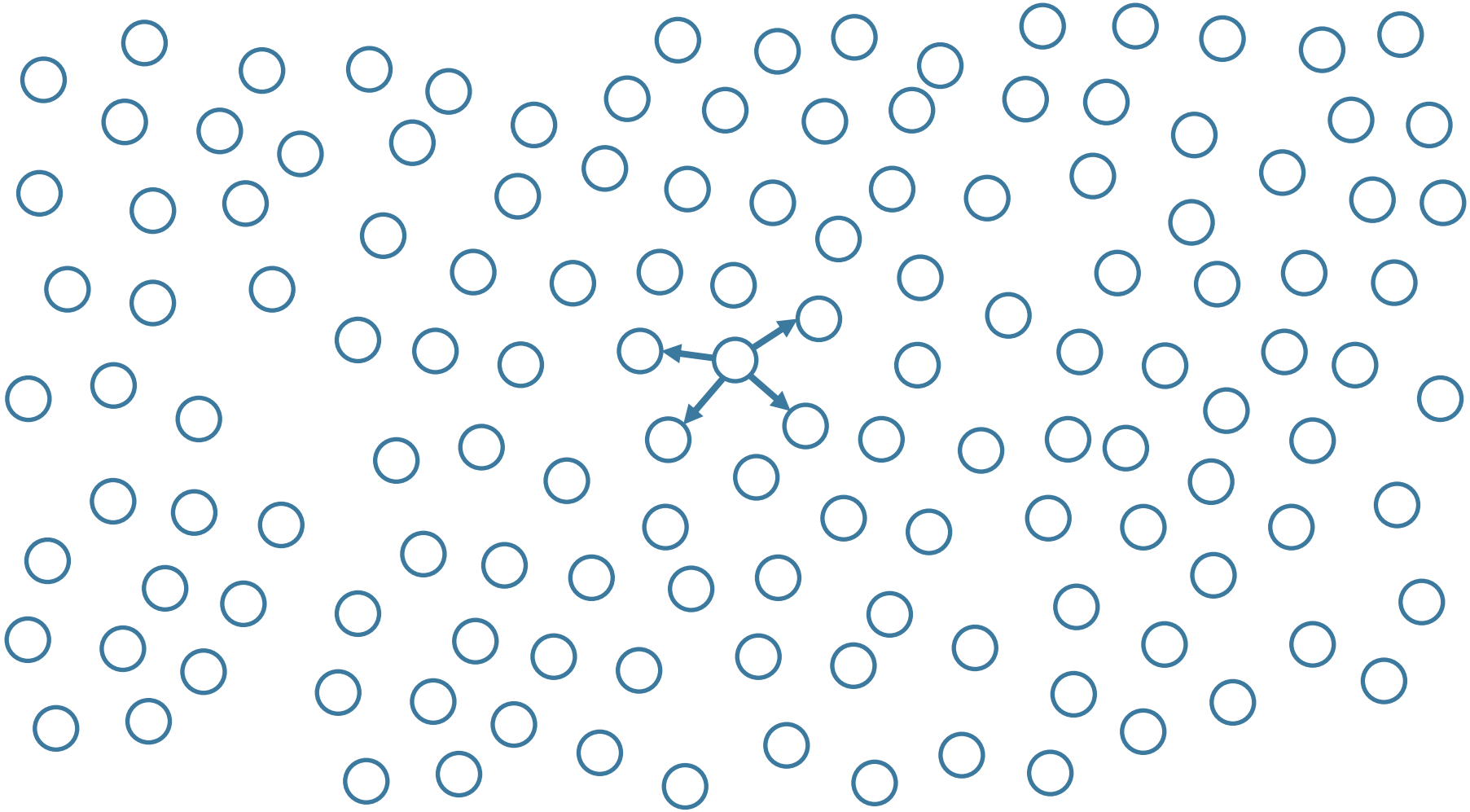


Adaptive Re-Ranking

Università di Pisa

University of Glasgow

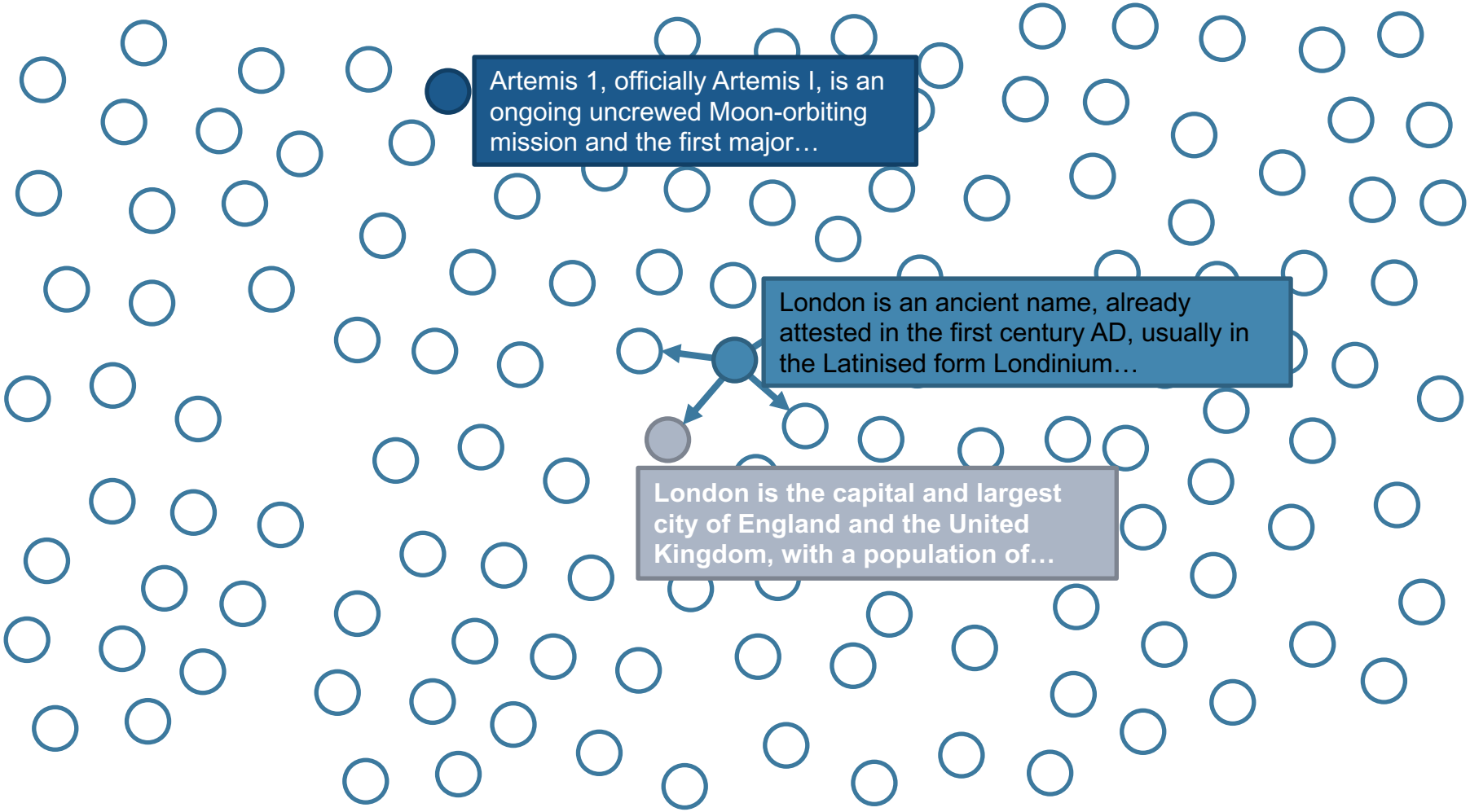
Consider the corpus as a graph – edges between similar docs



Adaptive Re-Ranking

Università di Pisa | University of Glasgow

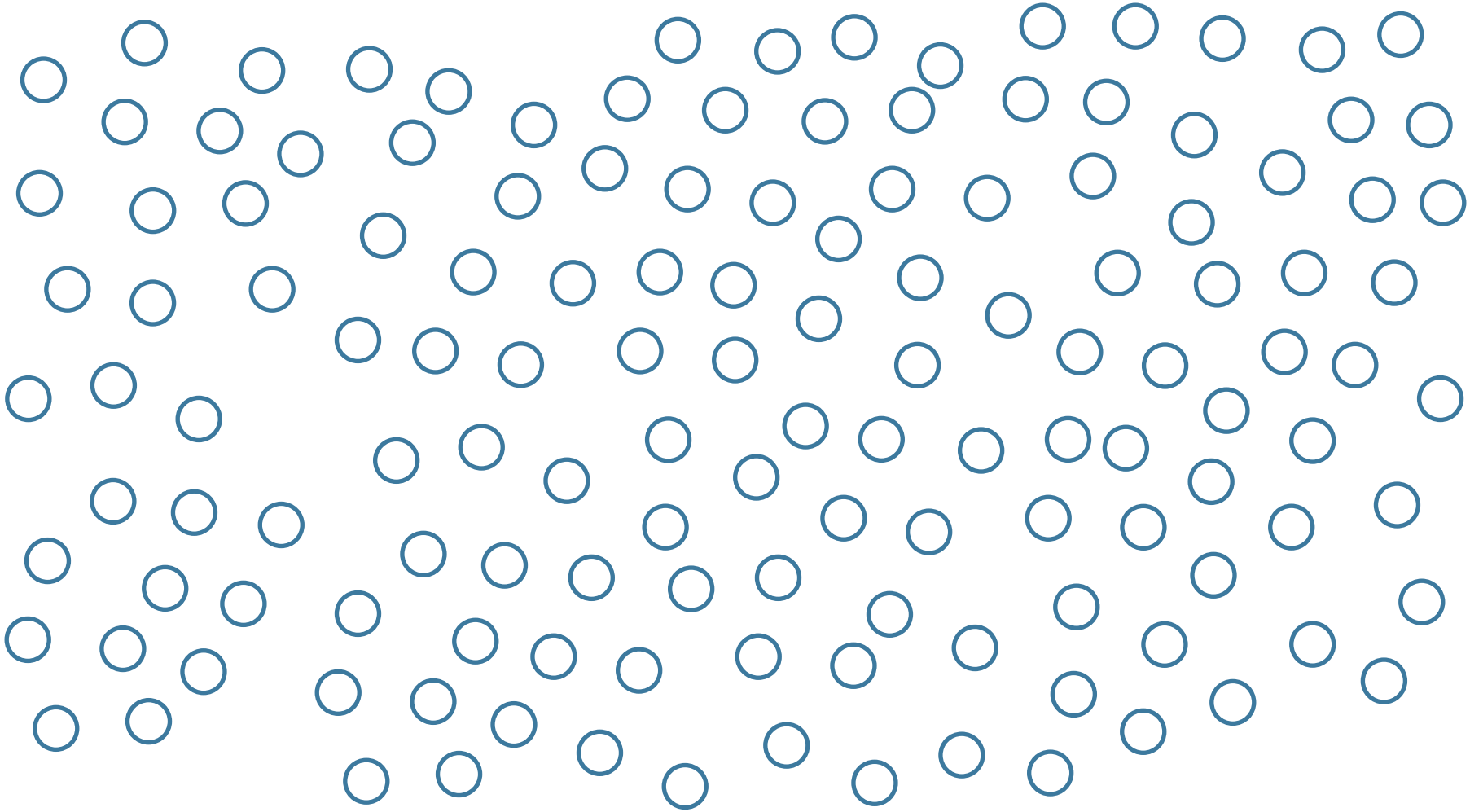
Consider the corpus as a graph – edges between similar docs



Adaptive Re-Ranking

Università di Pisa | University of Glasgow

Stage 2(a): Score batch of **initial** documents



Adaptive Re-Ranking

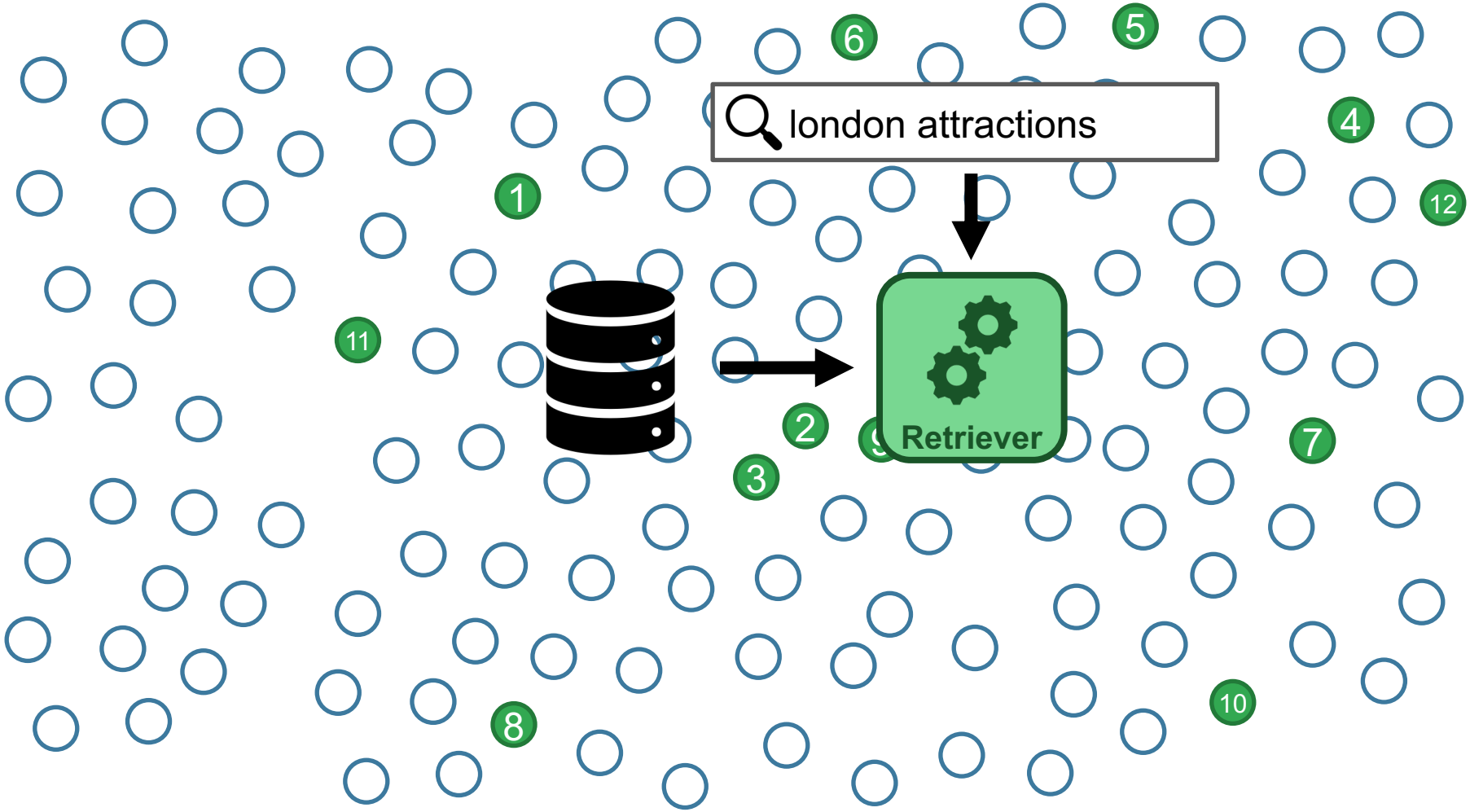
Università di Pisa | University of Glasgow

Stage 2(a): Score batch of initial documents

Adaptive Re-Ranking

Università di Pisa | University of Glasgow

Stage 2(a): Score batch of initial documents

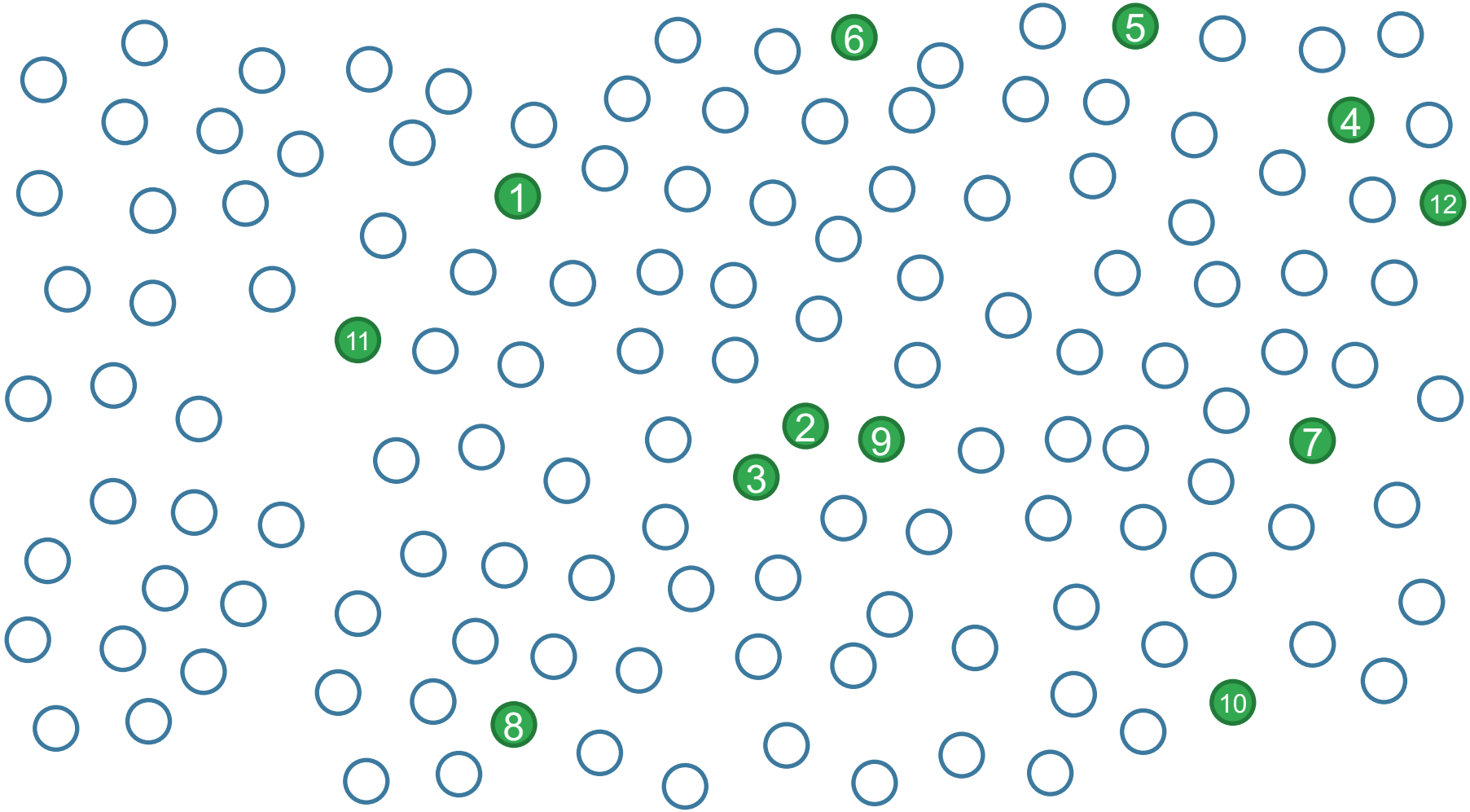


Adaptive Re-Ranking

UNIVERSITÀ DI PISA

University of Glasgow

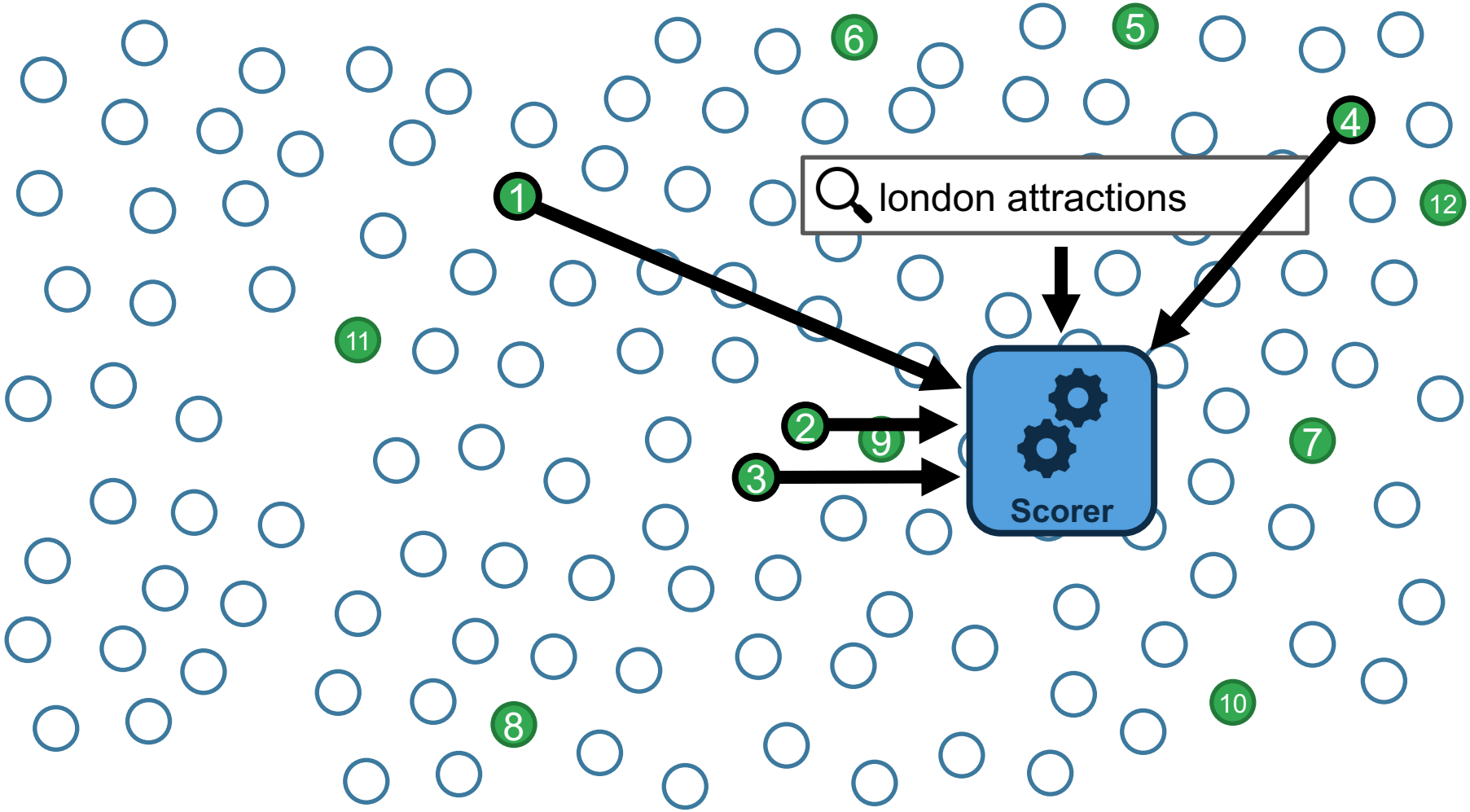
Stage 2(a): Score batch of initial documents



Adaptive Re-Ranking

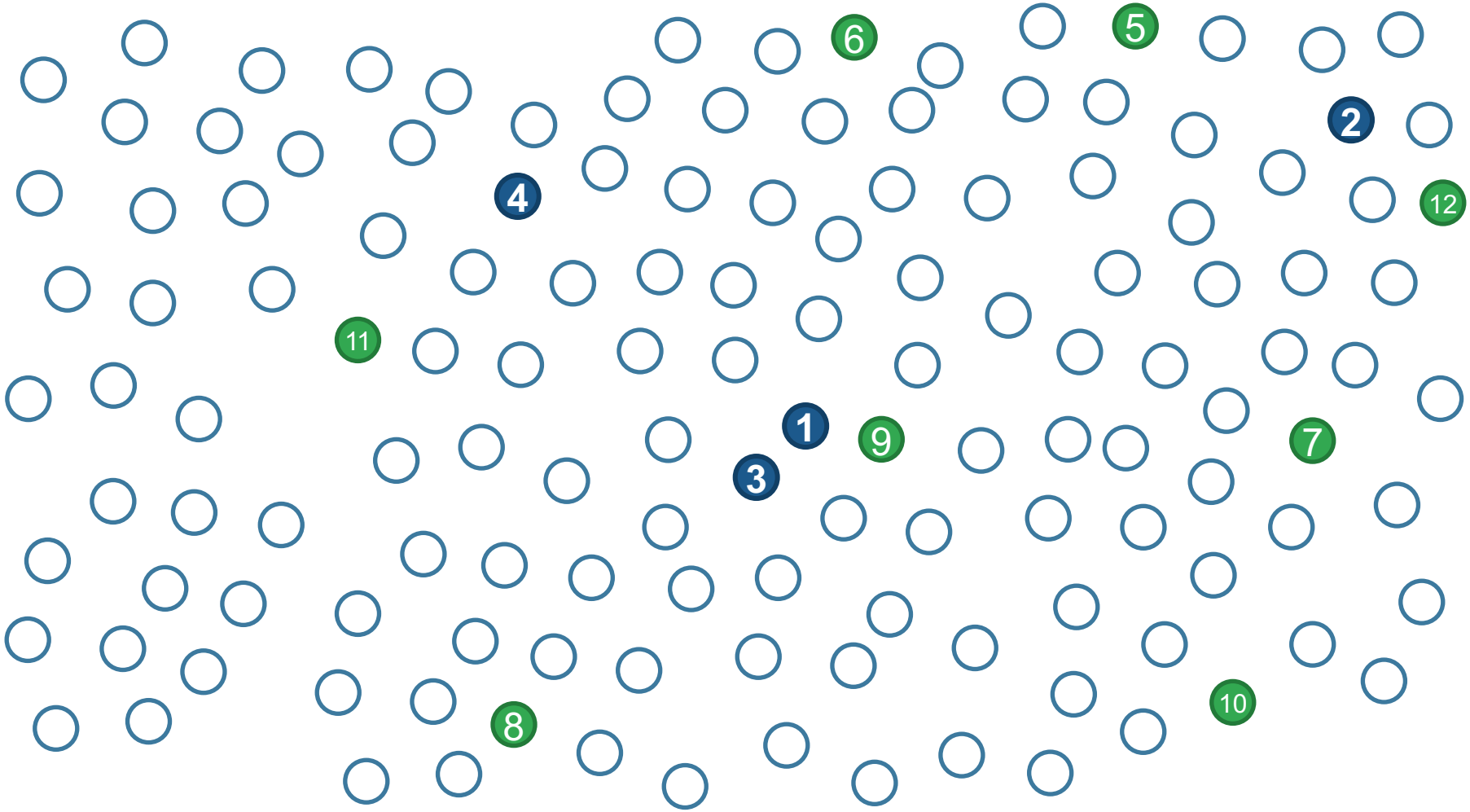
Università di Pisa | University of Glasgow

Stage 2(a): Score batch of initial documents



Adaptive Re-Ranking

Stage 2(a): Score batch of initial documents

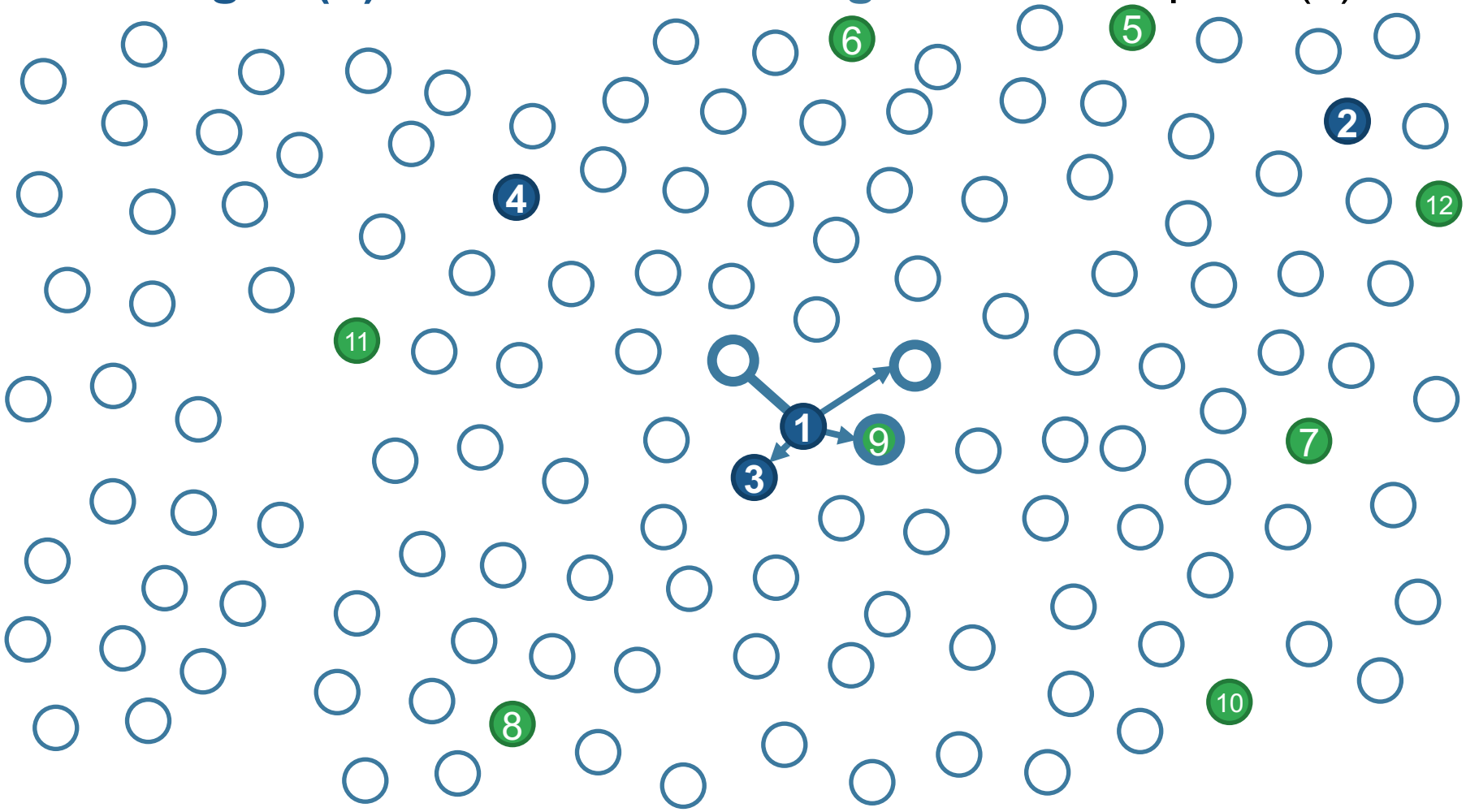


Adaptive Re-Ranking

UNIVERSITÀ DI PISA

University of Glasgow

Stage 2(b): Find and score neighbours of top doc(s)

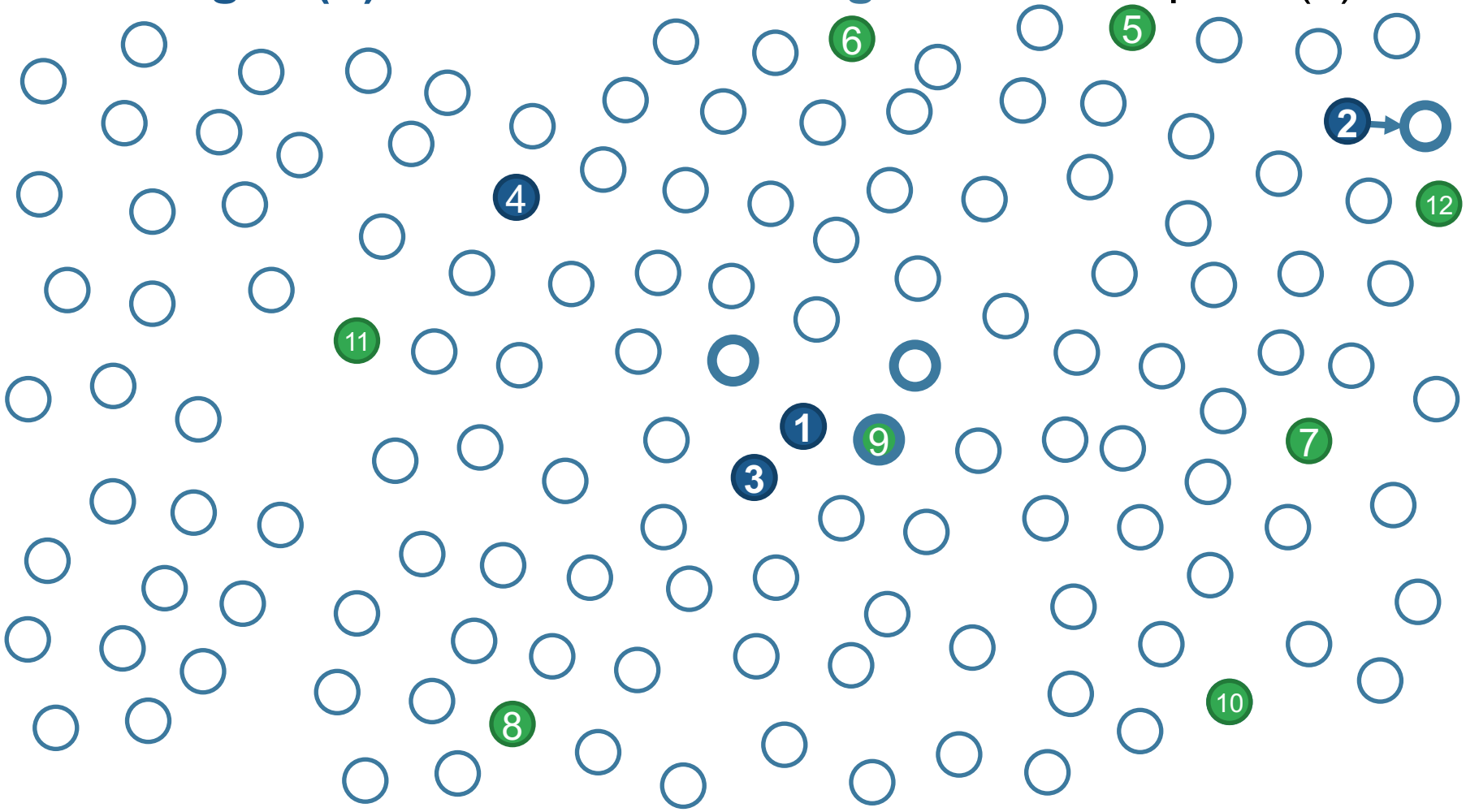


Adaptive Re-Ranking

UNIVERSITÀ DI PISA

University of Glasgow

Stage 2(b): Find and score neighbours of top doc(s)

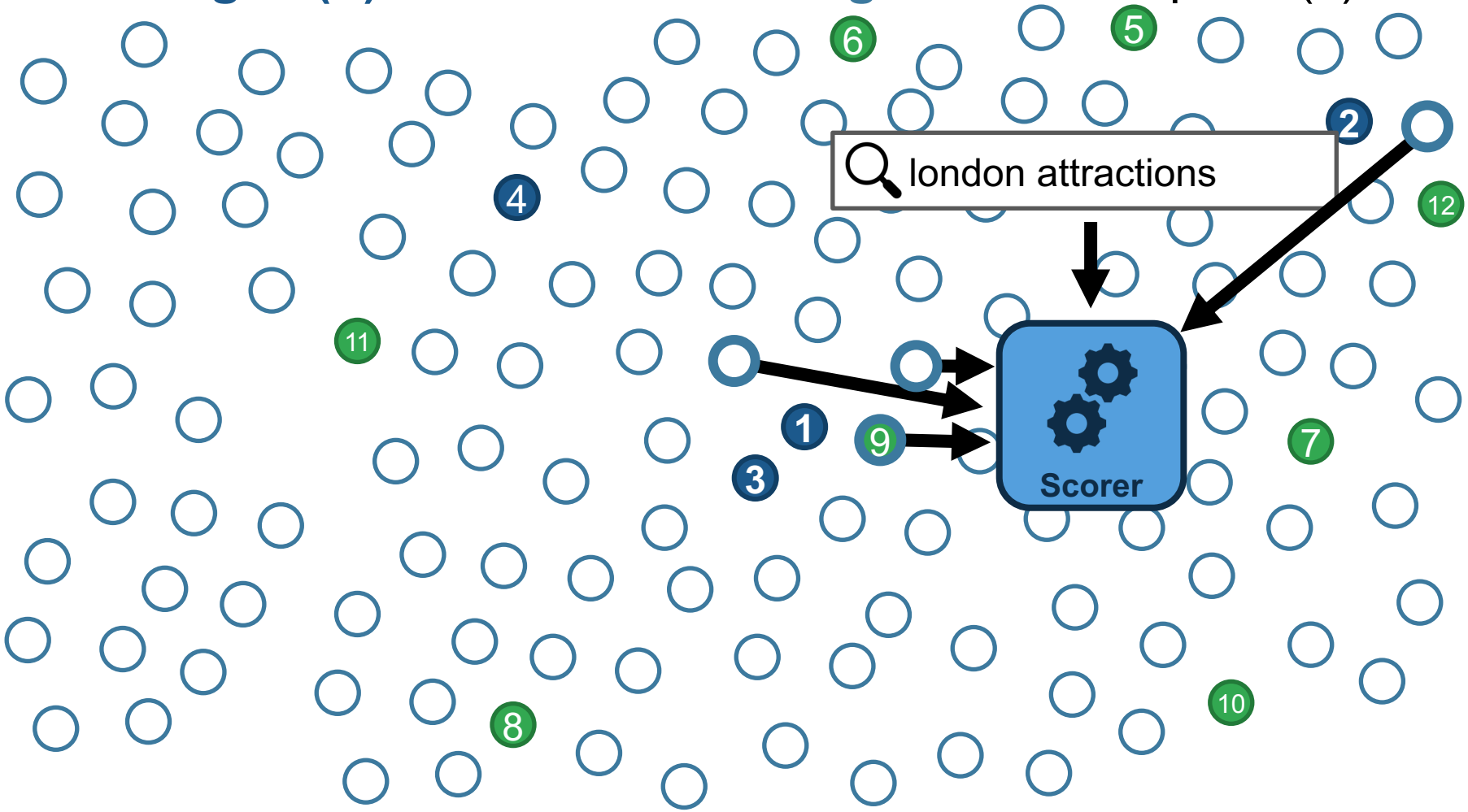


Adaptive Re-Ranking

UNIVERSITÀ DI PISA

University of Glasgow

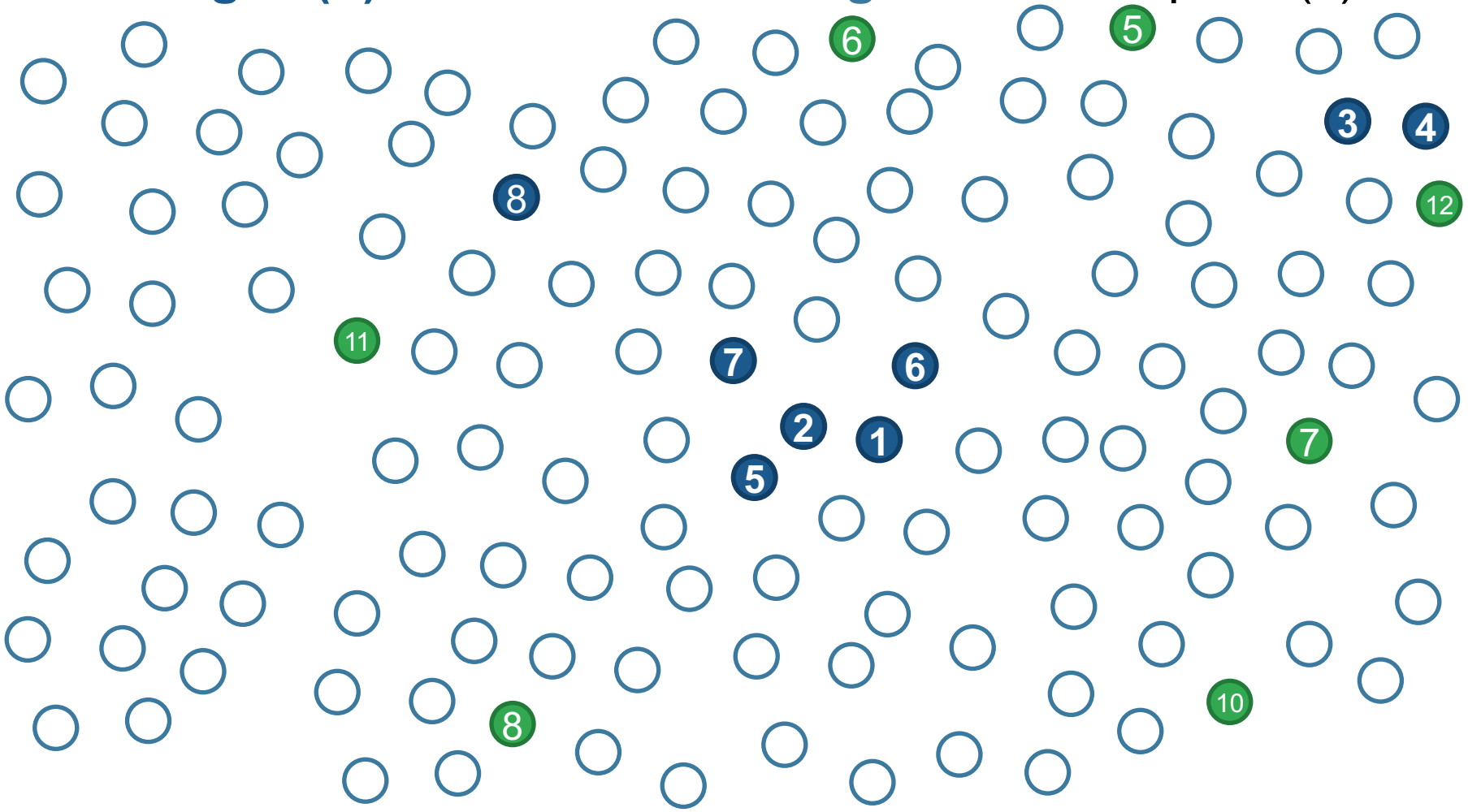
Stage 2(b): Find and score neighbours of top doc(s)



Adaptive Re-Ranking

Università di Pisa | University of Glasgow

Stage 2(b): Find and score neighbours of top doc(s)

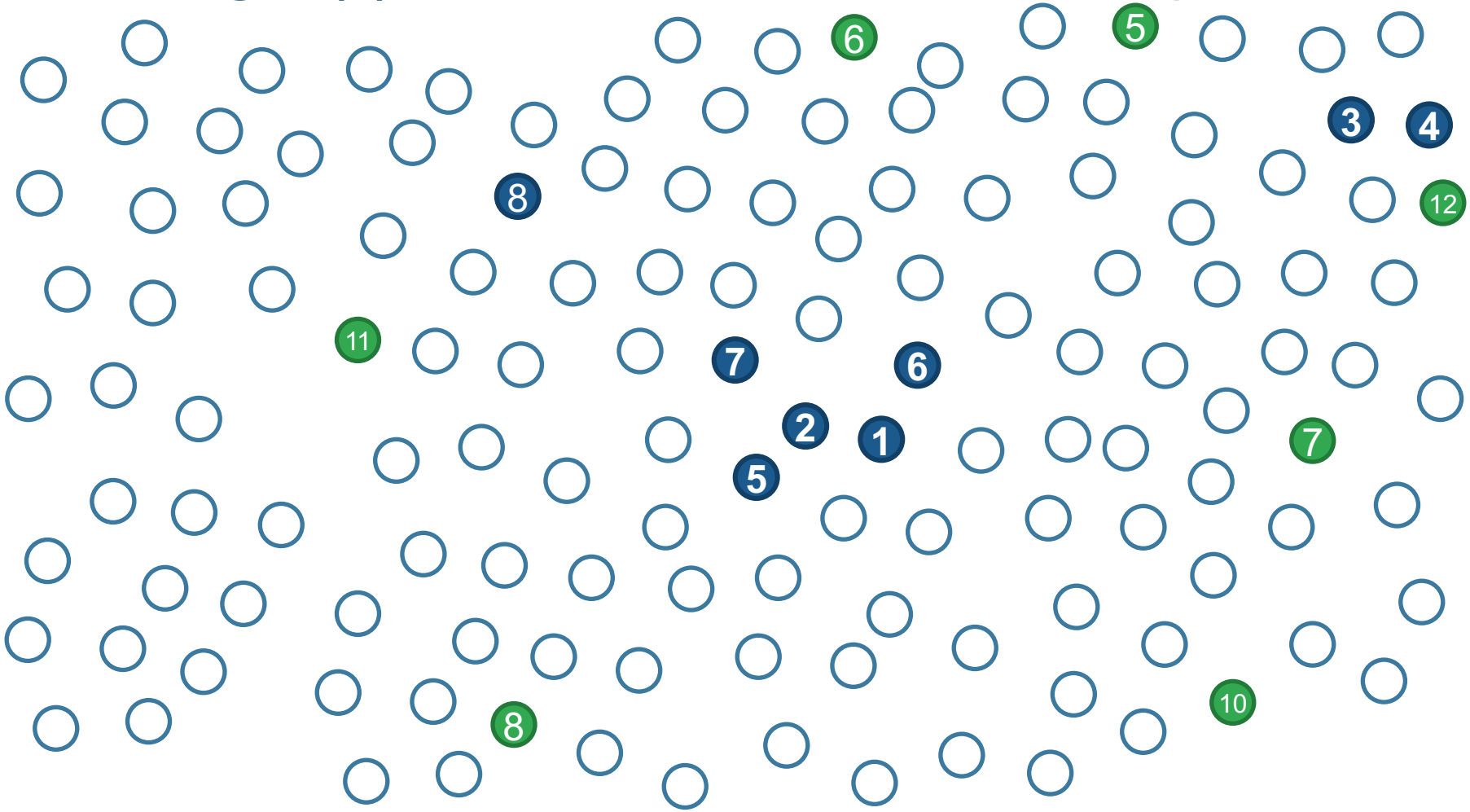


Adaptive Re-Ranking

UNIVERSITÀ DI PISA

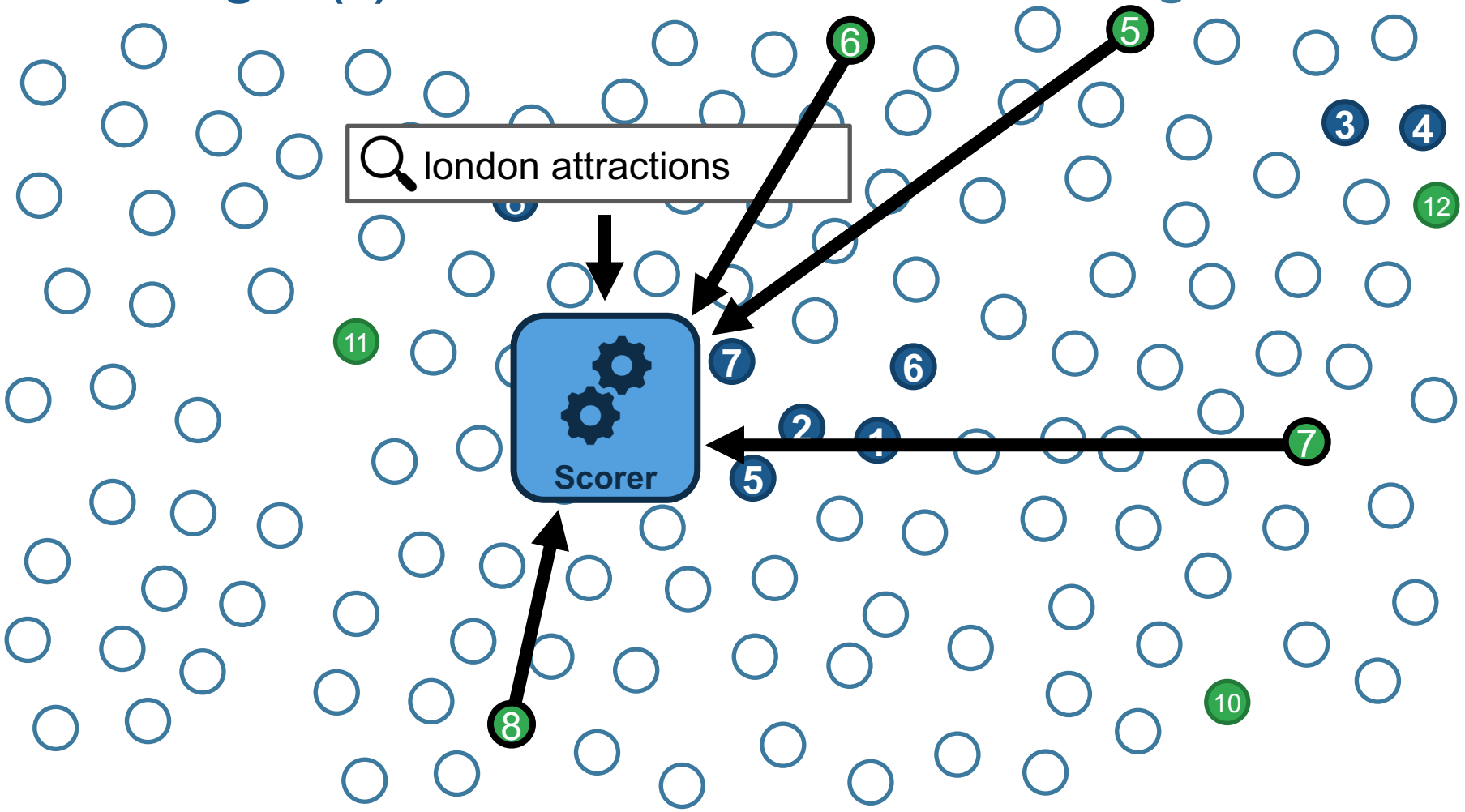
University of Glasgow

Stage 2(c): Alternate between initial and neighbours



Adaptive Re-Ranking

Stage 2(c): Alternate between **initial** and **neighbours**

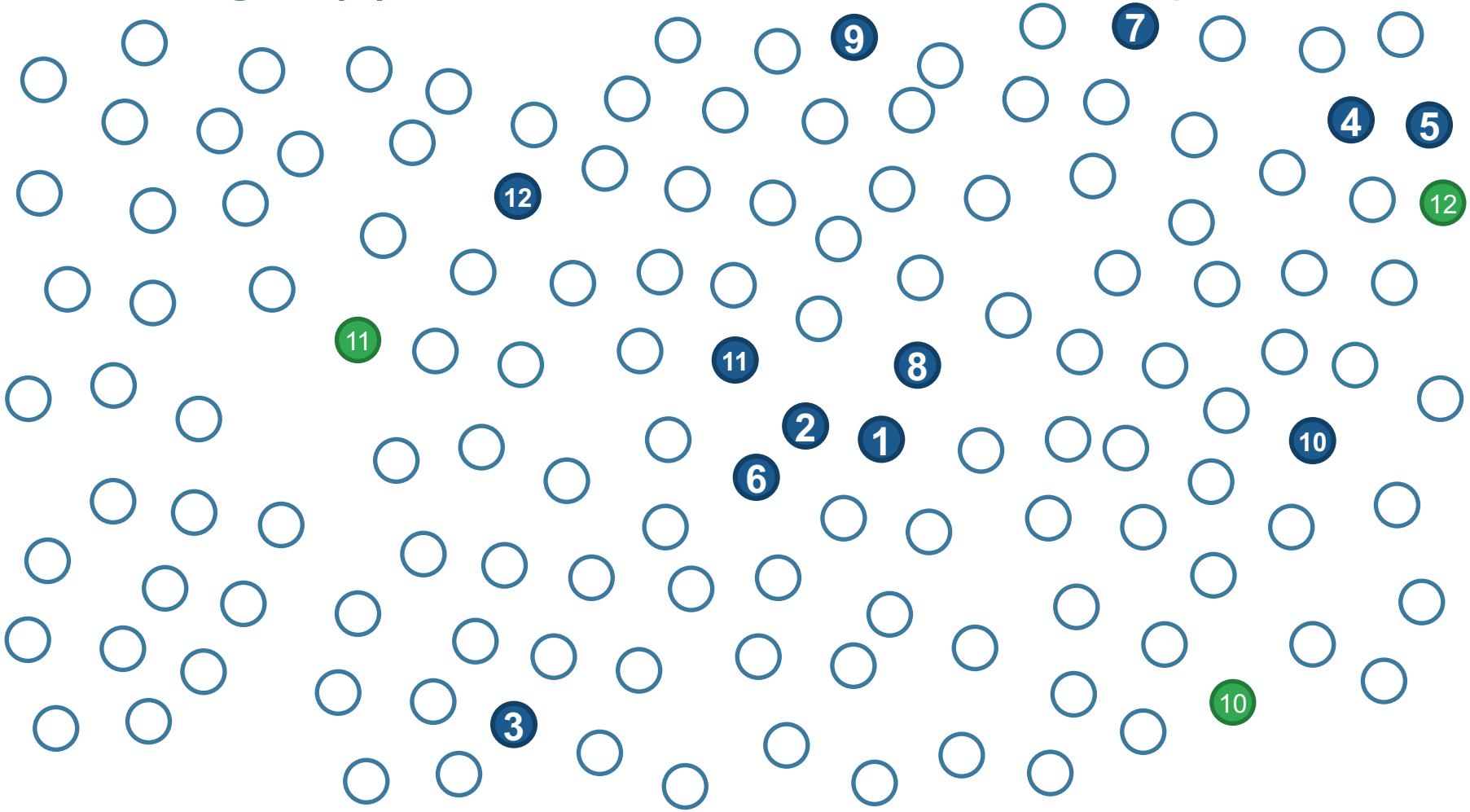


Adaptive Re-Ranking

UNIVERSITÀ DI PISA

University of Glasgow

Stage 2(c): Alternate between initial and neighbours

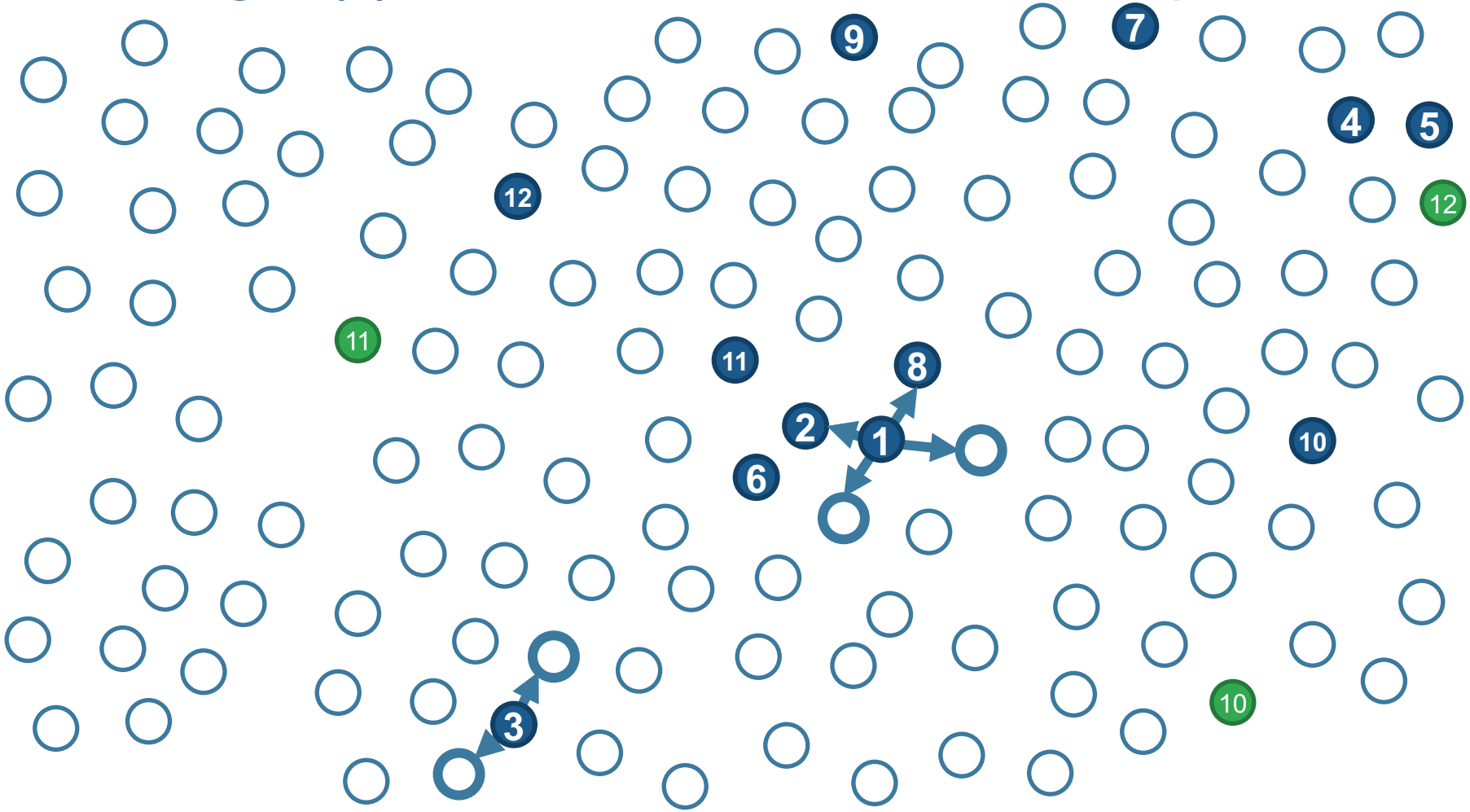


Adaptive Re-Ranking

UNIVERSITÀ DI PISA

University of Glasgow

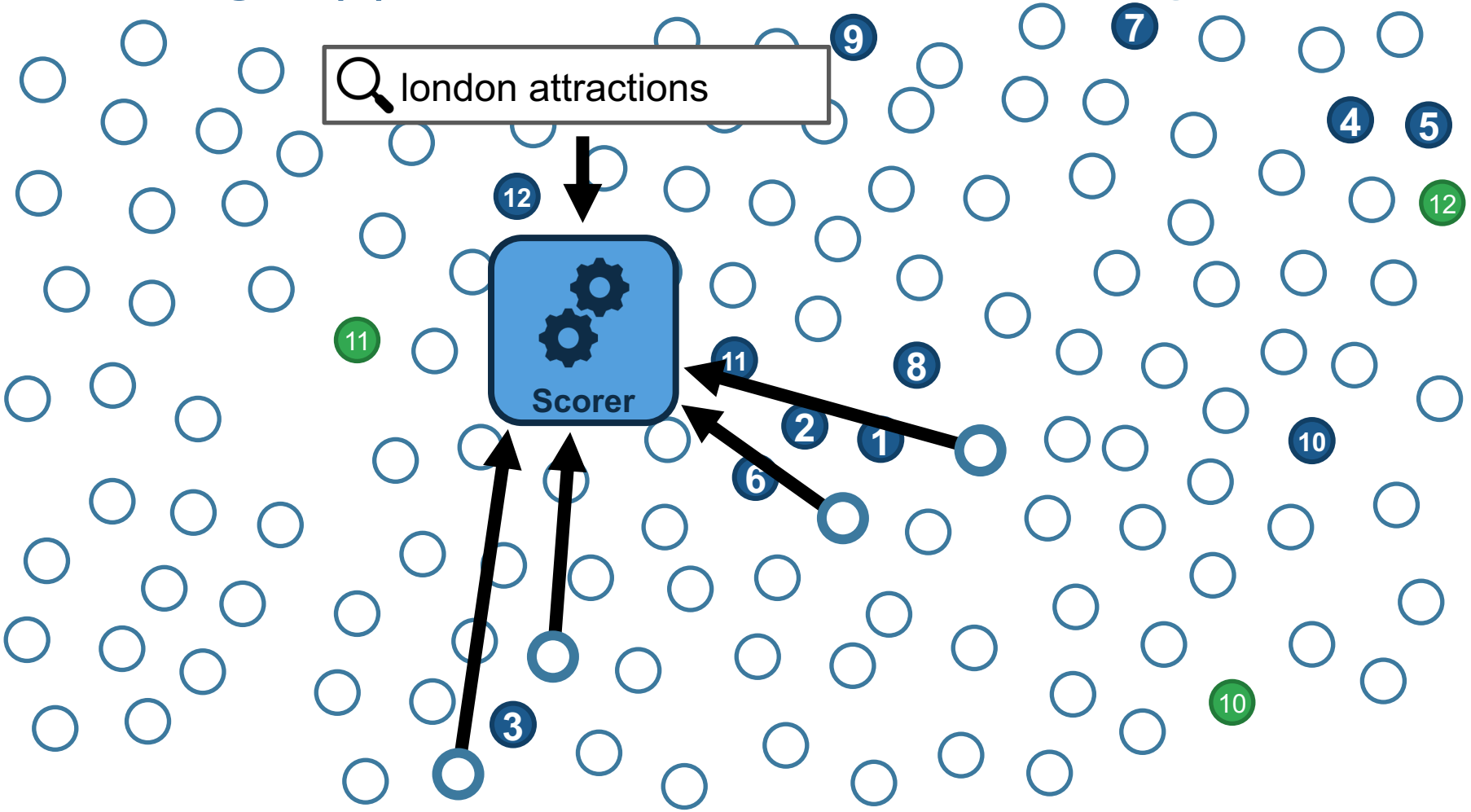
Stage 2(c): Alternate between initial and neighbours



Adaptive Re-Ranking

Università di Pisa | University of Glasgow

Stage 2(c): Alternate between initial and neighbours

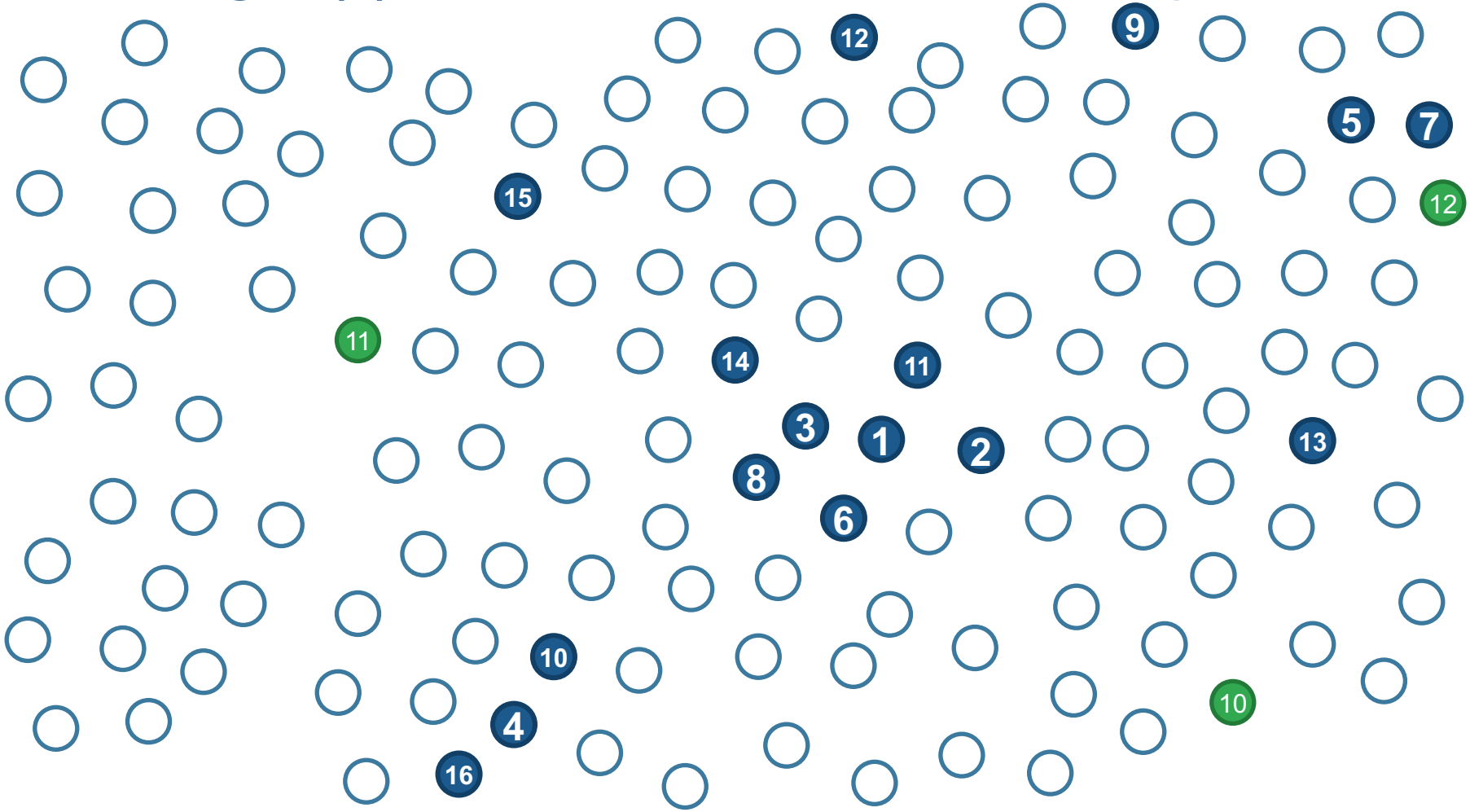


Adaptive Re-Ranking

UNIVERSITÀ DI PISA

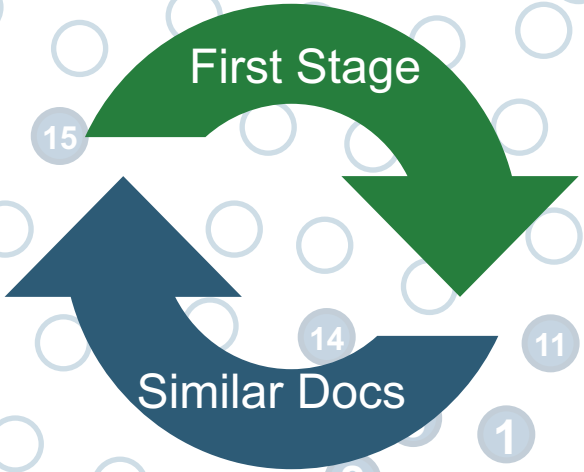
University of Glasgow

Stage 2(c): Alternate between initial and neighbours



Adaptive Re-Ranking

Stage 2(c): Alternate between **initial** and **neighbours**



👍 Identifies New Documents from Corpus

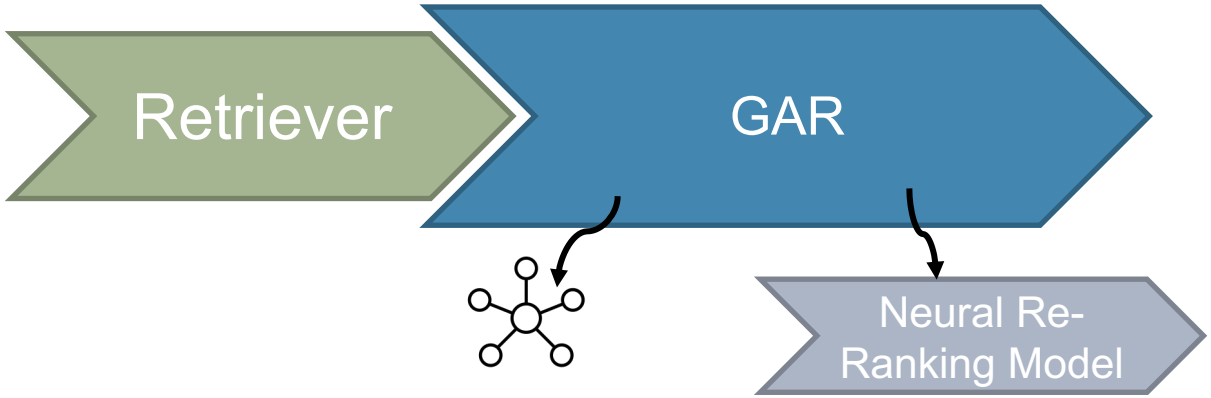
Those that were not in the candidate set.

👍 Prioritises Scoring Potentially Relevant Documents Earlier

Allows a lower re-ranking budget

Adaptive Re-Ranking with a Corpus Graph

Università di Pisa | University of Glasgow



Pre-computed Corpus Graph
(doc-doc similarity)

```
from pyterrier_adaptive import GAR, CorpusGraph

first_stage = index.bm25() # or your favourite first stage
scorer = MonoT5() # or your favourite scorer
graph = CorpusGraph.from_dataset('msmarco_passage', 'bm25')

pipeline = first_stage >> GAR(scorer, graph)
```

University of Glasgow | Università di Pisa

SEARCH SOLUTIONS London, UK

Part 2E

WRAPUP

Summary

Re-ranking approaches continue to be an appealing and active area of research:

- Handle vocabulary mismatch and text semantics
- Straightforward and highly effective
- Query latency can be managed
- Poor first-stage retrieval performance can be overcome with adaptive re-ranking techniques

PyTerrier Supported Re-Rankers

Model	Provider	Re-Ranking	Training	Indexing
DRMM	OpenNIR	✅	✅	
KNRM		✅	✅	
ConvKNRM		✅	✅	
PACRR		✅	✅	
MatchPyramid		✅	✅	
Vanilla BERT		✅	✅	
CEDR		✅	✅	
EPIC		✅	✅	✅
monoT5	pyterrier_t5	✅		
duoT5		✅		
ColBERT	pyterrier_colbert	✅		✅ (pt3)
GAR	pyterrier_adaptive	✅		

Easy to add a PyTerrier wrapper for your model for use in pipelines!

Transformer Classes

Università di Pisa | University of Glasgow

Transformer Class	Examples
Q ↦ R Retrieval	BM25, ColBERT, ANCE (dense retrieval)
R ↦ Q Query Expansion	RM3, Bo1, etc.
R ↦ R Re-ranking	Vanilla BERT, monoT5, etc.
Q ↦ Q Query Re-writing	SDM, IntenT5, etc.
D ↦ D Document Aug.	doc2query, DeepCT
D ↦ ∅ Indexers	Sparse Indexers, Dense Indexers

More on these in Part 3: Dense Retrieval

University of Glasgow | Università di Pisa

SEARCH SOLUTIONS London, UK

QUESTIONS?

What's the task in the notebooks?

Università di Pisa | University of Glasgow

In the notebooks, you will experience:

- Building untrained re-ranking models
- Using pre-trained BERT and T5 models
- Scoring with EPIC using pre-computed document vecs
- Tuning re-ranking thresholds

Practical Time

The tutorial Github repo has links to the notebook for Part 2

- https://github.com/terrier-org/searchsolutions2022-tutorial
- Press the 

link for each notebook to start a Colab session

Timings:

- Practical: 14:00 - 14:30
- Afternoon break: 14:30 - 14:45
- Part 3 resumes at 14:45

References

KNRM: *Xiong, et al. End-to-End Neural Ad-hoc Ranking with Kernel Pooling. SIGIR 2017.*

BERT: *Devlin, et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. NAACL 2019.*

CEDR: *MacAvaney, et al. CEDR: Contextualized Embeddings for Document Ranking. SIGIR 2019*.

MonoT5: *Nogueira, et al. Document Ranking with a Pretrained Sequence-to-Sequence Model. arXiv 2020.*

EPIC*: MacAvaney, et al. Expansion via Prediction of Importance with Contextualization. SIGIR 2020.*

ColBERT*: Khattab & Zaharia. ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. SIGIR 2020.*

GAR*: MacAvaney et al. Adaptive Re-Ranking using a Corpus Graph. CIKM 2022.*