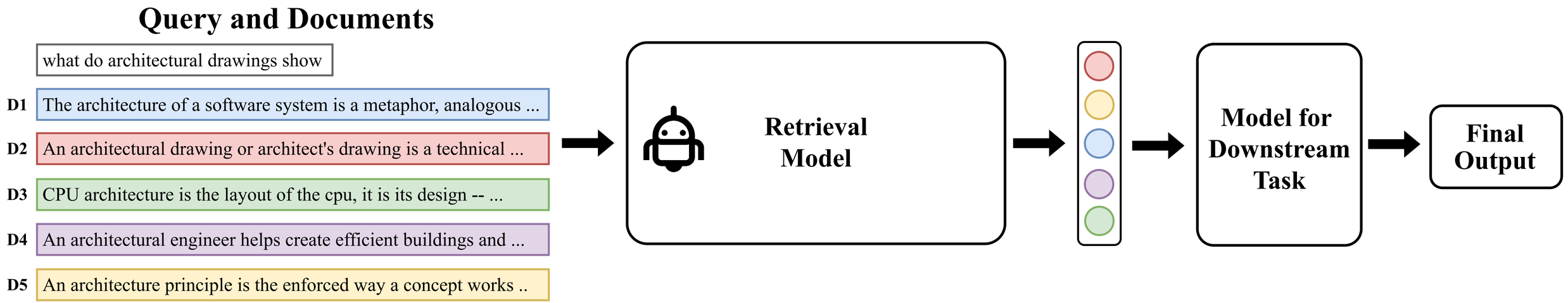
# Policy-Gradient Training of Language Models for Ranking

Ge Gao, Jonathan D. Chang, Claire Cardie, Kianté Brantley, Thorsten Joachims

# Retrieval Basics

- Task definition: rank documents based on their relevance to a query
- Application:
  - Ranked documents are input to some downstream models
  - Separate from training retrieval models



# Retrieval Basics

- Conventional training objectives: contrastive loss
  - Requires ground truth annotation for relevant documents and estimation for truly irrelevant documents (i.e., hard negatives)

**Query and Documents**

what do architectural drawings show

| | | |
|---|---|---|
| D1 | The architecture of a software system is a metaphor, analogous ... | Irrelevant |
| D2 | An architectural drawing or architect's drawing is a technical ... | Relevant |
| D3 | CPU architecture is the layout of the cpu, it is its design -- ... | Irrelevant |
| D4 | An architectural engineer helps create efficient buildings and ... | Irrelevant |
| D5 | An architecture principle is the enforced way a concept works .. | Irrelevant |

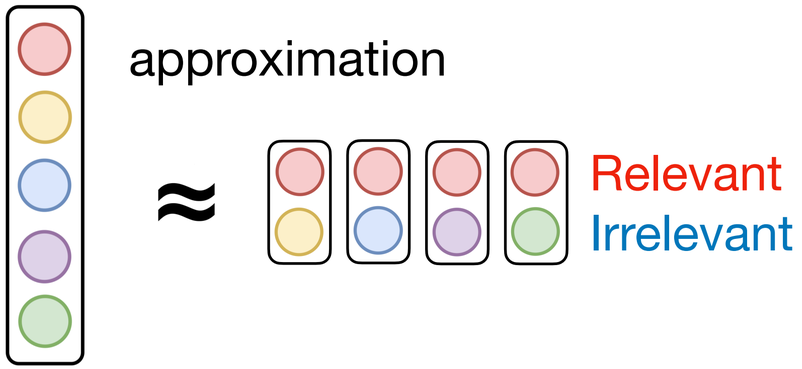
# Retrieval Basics

- Conventional training objectives: contrastive loss
  - Requires ground truth annotation for relevant documents and estimation for truly irrelevant documents (i.e., hard negatives)
  - Pairwise approximation of the listwise ranking objective

**Query and Documents**

what do architectural drawings show

| | | |
|---|---|---|
| D1 | The architecture of a software system is a metaphor, analogous ... | Irrelevant |
| D2 | An architectural drawing or architect's drawing is a technical ... | Relevant |
| D3 | CPU architecture is the layout of the cpu, it is its design -- ... | Irrelevant |
| D4 | An architectural engineer helps create efficient buildings and ... | Irrelevant |
| D5 | An architecture principle is the enforced way a concept works .. | Irrelevant |

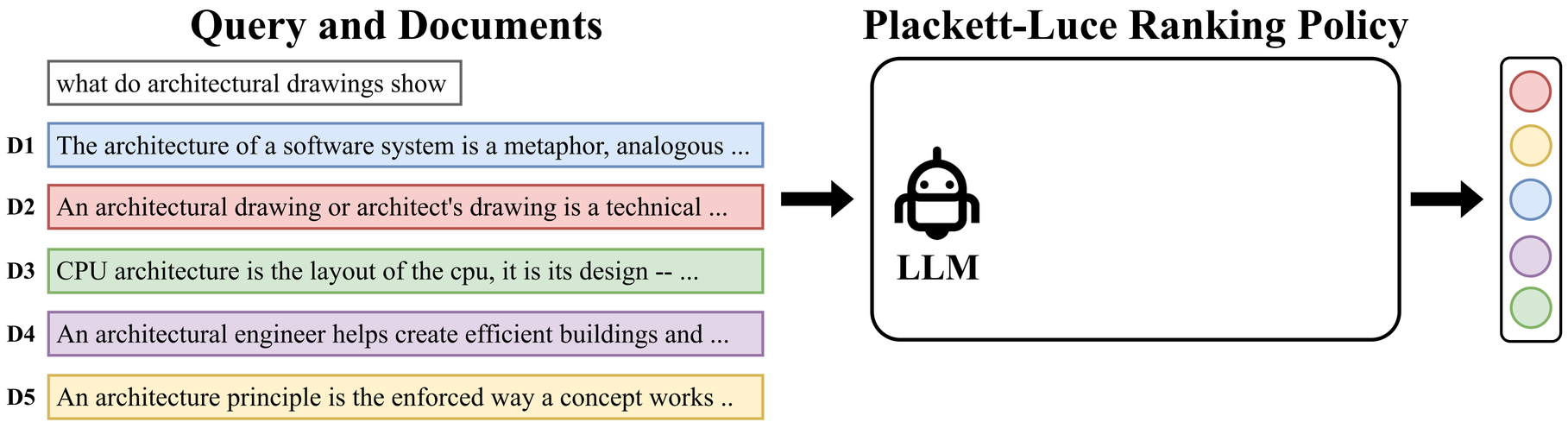


# Research Question

- How to improve LLM-based retrieval models in a pipeline setup?
    - Need to adapt to the downstream applications
    - No document-level annotation available
    - Output high-quality ranking over >2 docs for practical usage

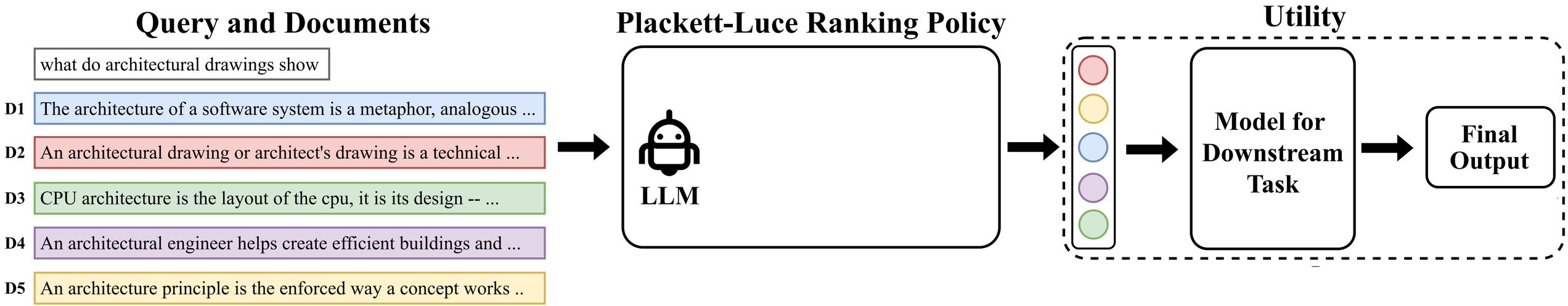
# Our Contribution

- **Neural PG-RANK:** train LLM-based retrievers in a pipeline setup based on downstream feedback
  - Directly optimize the listwise objective — use Plackett-Luce ranking policy



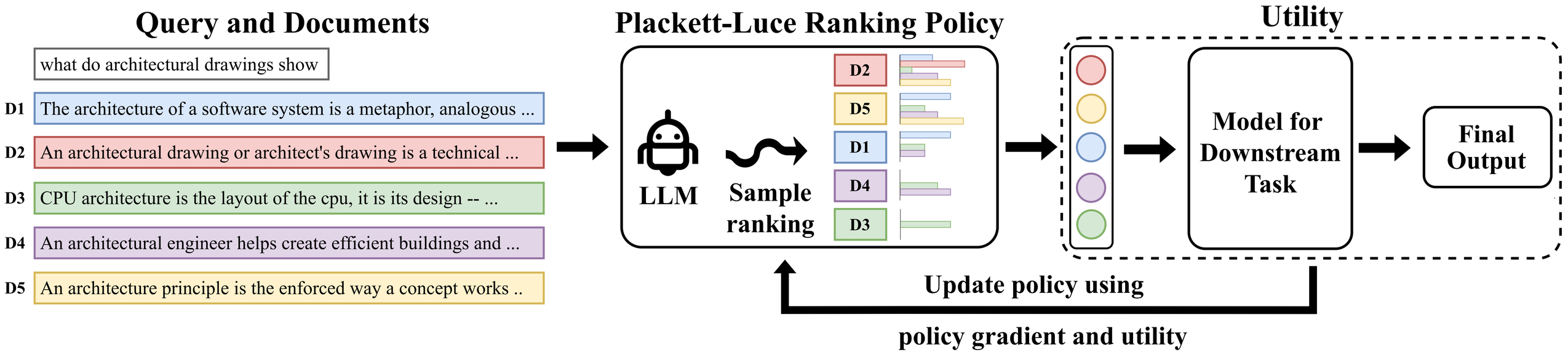
# Our Contribution

- **Neural PG-RANK:** train LLM-based retrievers in a pipeline setup based on downstream feedback
  - Directly optimize the listwise objective — use Plackett-Luce ranking policy
  - Do not require document-level relevance annotation — use downstream utility



# Our Contribution

- **Neural PG-RANK:** train LLM-based retrievers in a pipeline setup based on downstream feedback
  - Directly optimize the listwise objective — use Plackett-Luce ranking policy
  - Do not require document-level relevance annotation — use downstream utility
  - Adapt to downstream tasks by unifying the training objective with the downstream application



# Setting

- Define the utility of a ranking policy for a given query

$$U(\pi|q) = \mathbb{E}_{r\sim\pi(\cdot|q)}\left[\Delta(r|q)\right]$$

Utility function

Query

Ranking

- Learning objective is to learn a ranking policy that optimizes the expected utility over the query distribution

$$\pi^\star = \underset{\pi\in\Pi}{\operatorname{argmax}}\, \mathbb{E}_{q\sim\mathcal{Q}}\left[U(\pi|q)\right]$$

# Method

- Define a Plackett-Luce ranking policy

**Definition 1** (Plackett-Luce Model (Plackett, 1975; Luce, 1959)). *Given the utility scores of the $N$ items, $\boldsymbol{w} = [w_1, w_2, \cdots, w_N]^T$, the probability of observing a certain ordered list of these items, $(i_1, i_2, \cdots, i_N)$, is defined as*

$$p((i_1, i_2, \cdots, i_N); \boldsymbol{w}) = \prod_{j=1}^{N} \frac{\exp(w_{i_j})}{\sum_{l=j}^{N} \exp(w_{i_l})}. \qquad (1)$$

# Method

- Define a Plackett-Luce ranking policy
  - expressed as a product of softmax distributions
  - based on query-document relevance scores

Scoring function
Query
Doc

$$\pi_\theta(r|q) = \prod_{i=1}^{n} \frac{\exp s_\theta(q, d_{r(i)})}{\sum_{j \in \{r(i),\ldots,r(n)\}} \exp s_\theta(q, d_j)}$$

# Method

- We use REINFORCE

$$\begin{aligned}\nabla_\theta U(\pi_\theta|q) &= \nabla_\theta \mathbb{E}_{r\sim\pi_\theta(\cdot|q)}\left[\Delta(r|q)\right] \\ &= \mathbb{E}_{r\sim\pi_\theta(\cdot|q)}\left[\nabla_\theta \log \pi_\theta(r|q)\Delta(r|q)\right]\end{aligned}$$

# Method

- We use REINFORCE

  + Monte Carlo sampling with N samples

  + Variance reduction with leave-one-out baseline

$$\widehat{\nabla}_\theta U(\pi_\theta|q) = \frac{1}{N}\sum_i \Big[\nabla_\theta \log \pi_\theta(r_i|q)\Big(\Delta(r_i|q) - \frac{1}{N-1}\sum_{j\neq i}\Delta(r_j|q)\Big)\Big]$$

# Utility

- Utility function scores a ranking based on how useful it is in the downstream
- In our pilot study, we use nDCG@10 as an approximation of the downstream utility
  - nDCG@10 is a measure of ranking quality
  - We assume higher nDCG@10 relates to better downstream performance

**Policy-Gradient Training of Language Models for Ranking**

**Ge Gao** **Jonathan D. Chang** **Claire Cardie** **Kianté Brantley** **Thorsten Joachims**
Department of Computer Science, Cornell University

# Utility

- We use REINFORCE

  + Monte Carlo sampling with N samples

  + Variance reduction with leave-one-out baseline

  + nDCG@10 as utility function

$$\widehat{\nabla}_\theta U(\pi_\theta|q) = \frac{1}{N}\sum_i \Big[\nabla_\theta \log \pi_\theta(r_i|q)\Big(\Delta(r_i|q) - \frac{1}{N-1}\sum_{j\neq i}\Delta(r_j|q)\Big)\Big]$$

$$= \frac{1}{N}\sum_i \Big[\sum_k \nabla_\theta \log \pi_\theta(r_{i,k}|q, r_{i,1:k-1})$$

$$\Big(\mathbf{nDCG}(r_{i,k:}|q, r_{i,1:k-1}) - \frac{1}{N-1}\sum_{j\neq i}\mathbf{nDCG}(r_{j,k:}|q, r_{i,1:k-1})\Big)\Big]$$

Utility score for the partial ranking til k

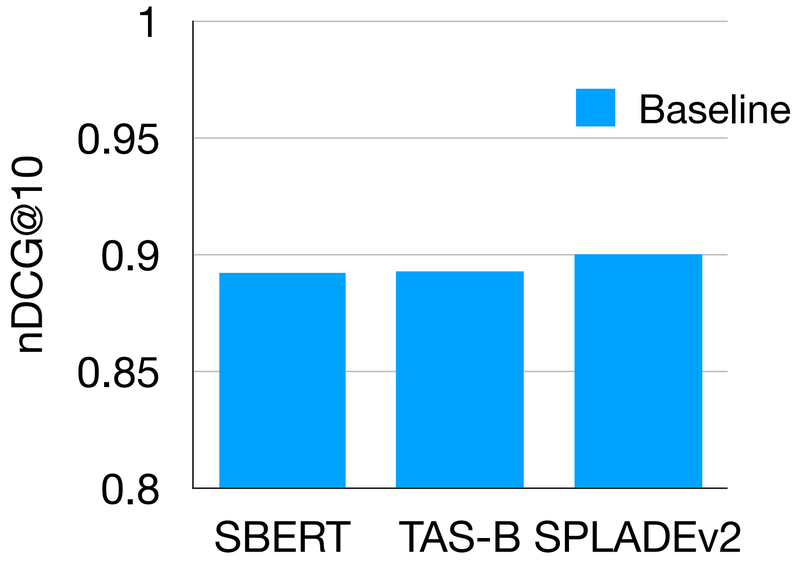
# Experimental Setup

- Data: MS MARCO for training; BEIR [Thakur et al., 2021] for evaluation
- Evaluation metric: nDCG@10 ↑
- Our ranking policy: either SBERT [Reimers & Gurevych, 2019] or TAS-B [Hofstätter et al., 2021] as warmstart, with Neural PG-RANK method as fine-tuning
- Comparison systems: supervised learning SOTA bi-encoder models

| Method | Source of Negative Docs | | | Additional Supervision | Loss |
|---|---|---|---|---|---|
| | In-Batch | BM25 | Dense Model | | |
| SBERT (Reimers & Gurevych, 2019) | | ✓ | ✓✓✓ | ✓ | MarginMSE + NLL |
| TAS-B (Hofstätter et al., 2021) | ✓ | ✓ | | ✓✓ | MarginMSE + Distillation |
| SPLADEv2 (Formal et al., 2021) | | ✓ | ✓✓ | ✓ | MarginMSE + Sparsity |
| Neural PG-RANK (Ours) | | | ✓ | | Utility Maximization |

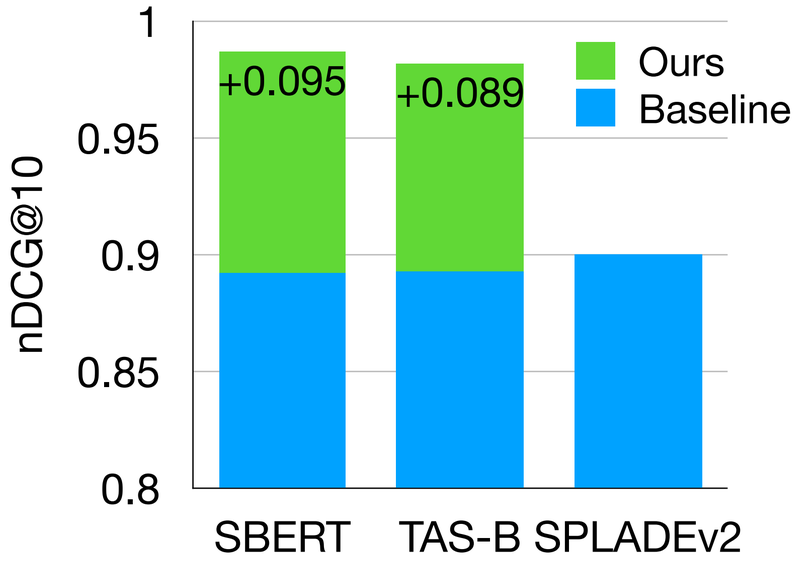
# Second-Stage Reranking

- Setup: search over a candidate set of 1k documents per query
- In-domain results: MS MARCO dataset (SBERT / TAS-B + Neural PG-RANK)



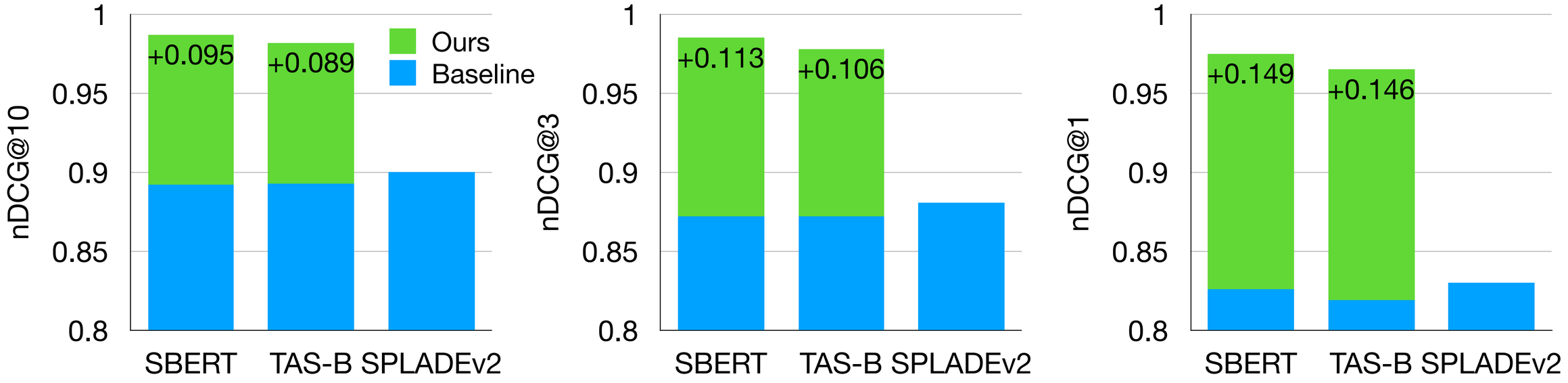
# Second-Stage Reranking

- Setup: search over a candidate set of 1k documents per query
- In-domain results: MS MARCO dataset (SBERT / TAS-B + Neural PG-RANK)
  - Performance gains with both warmstart models (nDCG@10) ↑



# Second-Stage Reranking

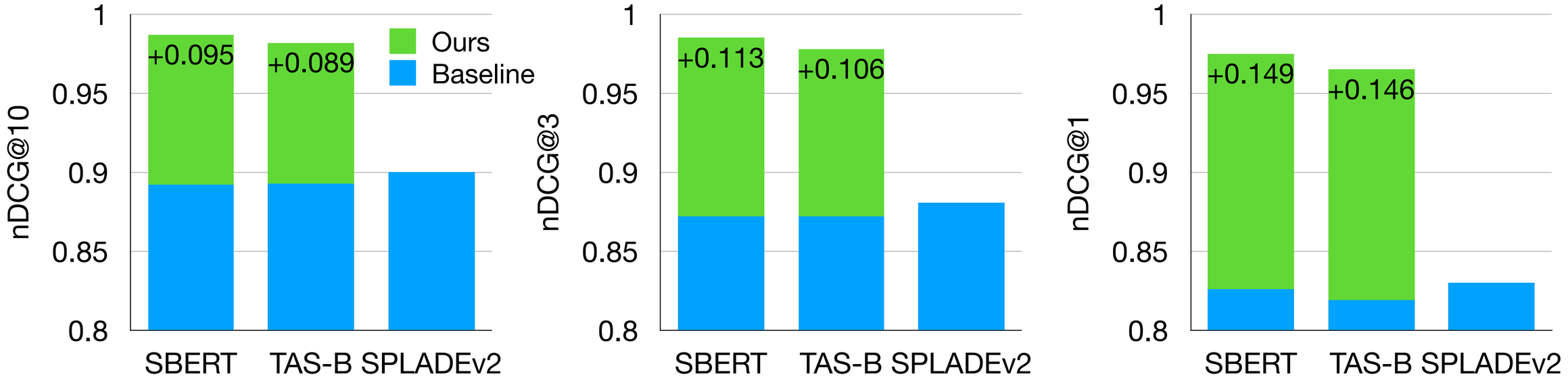
- Setup: search over a candidate set of 1k documents per query
- In-domain results: MS MARCO dataset (SBERT / TAS-B + Neural PG-RANK)
  - Performance gains with both warmstart models (nDCG@10) ↑
  - More gains in terms of nDCG@k with smaller k ↑↑↑



# Second-Stage Reranking

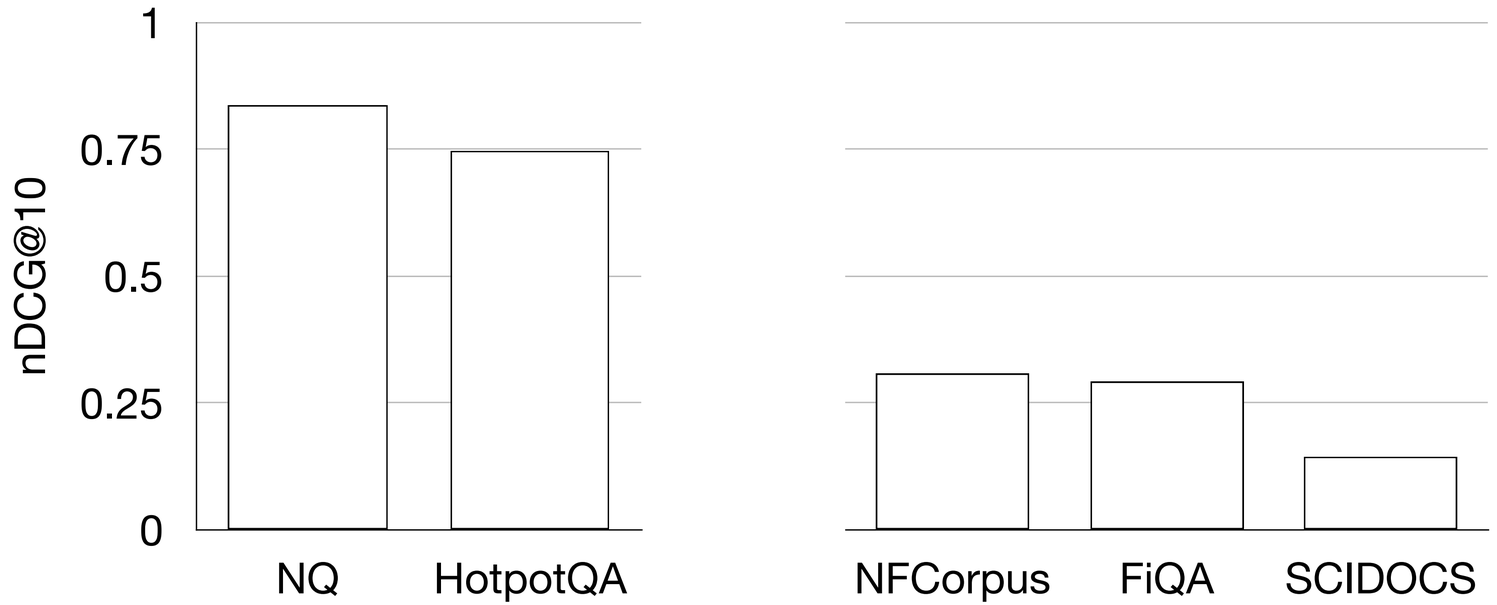
- Setup: search over a candidate set of 1k documents per query
- In-domain results: MS MARCO dataset (SBERT / TAS-B + Neural PG-RANK)
  - Performance gains with both warmstart models (nDCG@10) ↑
  - More gains in terms of nDCG@k with smaller k ↑↑↑

**Neural PG-RANK effectively ranks relevant docs high up**



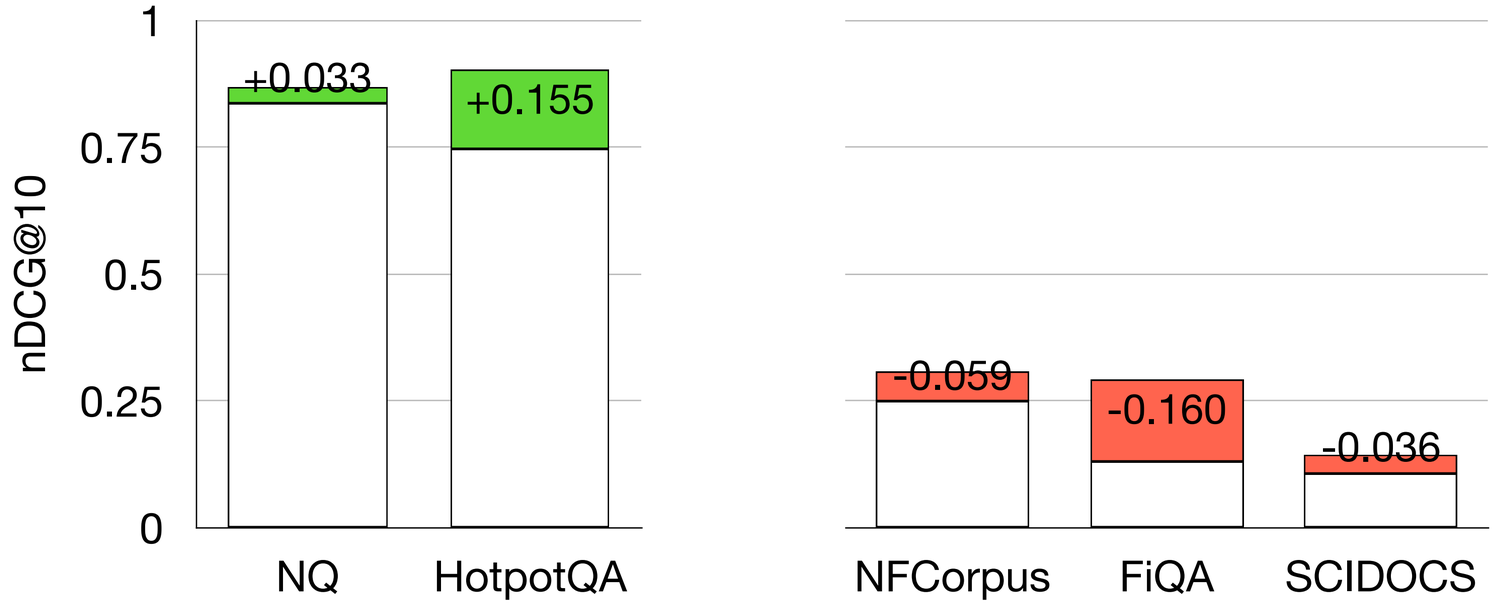
# Second-Stage Reranking

- Setup: search over a candidate set of 1k documents per query
- Out-of-domain results: from MS MARCO to other datasets (SBERT model + Neural PG-RANK)



# Second-Stage Reranking

- Setup: search over a candidate set of 1k documents per query
- Out-of-domain results: from MS MARCO to other datasets (SBERT model + Neural PG-RANK)
    - Notable improvements on widely-studied QA datasets ↑
    - Weaker in the domain of bio-medicine, finance, and science ↓



# Summary

- We introduce Neural PG-RANK to train LLM-based retriever and to improve retrieval-based pipeline based on downstream feedback
  - Directly optimize the listwise ranking objective using Plackett-Luce ranking policy
  - End-to-end training of the ranker as part of larger pipelines via policy gradient
  - Can adapt to downstream applications by optimizing any utility function for the downstream performance (no document-level annotation required)
- Pilot study: nDCG@10 as an approximation of the downstream utility