Query-focused Referentiability Learning for Zero-shot Retrieval

Jaeyoung Kim[1], **Dohyeon Lee**[2], **Seung-won Hwang**[12*]
[1]Interdisciplinary Program in Artificial Intelligence, Seoul National University
[2]Computer Science and Engineering, Seoul National University
{jae.young, waylight3, seungwonh}@snu.ac.kr

Abstract

Dense passage retrieval enhances Information Retrieval (IR) by encoding queries and passages into representation space. However, passage representations often fail to be referenced by their gold queries under domain shifts, revealing a weakness in representation space. One desirable concept for representations is "argmaxable". Being argmaxable ensures that no representations are theoretically excluded from selection due to geometric constraints. To be argmaxable, a notable approach is to increase isotropy, where representations are evenly spread out in all directions. These findings, while desirable also for IR, focus on passage representation and not on query, making it challenging to directly apply their findings to IR. In contrast, we introduce a novel query-focused concept of "referentiable" tailored for IR tasks, which ensures that passage representations are referenced by their gold queries. Building upon this, we propose Learning Referentiable Representation (LRR), and two strategic metrics, Self-P and Self-Q, quantifying how the representations are referentiable. Our experiments compare three dense model versions: Naive, Isotropic, and Referentiable, demonstrating that LRR leads to enhanced zero-shot performance, surpassing existing naive and isotropic versions.

1 Introduction

There have been notable advancements in the field of Information Retrieval (IR) through the use of dense representations. However, well-designed dense models such as DPR (Karpukhin et al., 2020), Condenser (Gao and Callan, 2021), and coCondenser (Gao and Callan, 2022) have shown limitations in zero-shot scenarios on BEIR benchmark (Thakur et al., 2021). This limitation suggests that passage representations often fail to align with their corresponding gold queries, highlighting a weakness in representation space. This paper explores strategies to overcome these limitations.

To ensure the representation of language models meets the minimum criteria, one important condition is "argmaxable". It indicates that the representations should lie in a space where they can be selected through an argmax operation (Demeter et al., 2020; Grivas et al., 2022). For example, in a two-dimensional representation set $\mathcal{T} = \{(1,1),(1,-1),(-1,0),(0.5,0)\}$, the fourth representation $(0.5,0)$ is an interior point relative to the first three. As a result, any point satisfying $\{(x,y)|x < 1, 2y < x+1, 2y > -x-1\}$, including the fourth representation, always yields lower values when multiplied by any vector, as it is overshadowed by the first three representations. While the concept of argmaxable focuses on the theoretical selection of representation by any vector, our interest lies in whether passage representation can practically be selected by query representation in IR. With this argument, we introduce an extended query-focused concept of "referentiable," which determines whether passage representation can practically be referenced as top-1 by its gold query representation.

To ensure representations are argmaxable, previous works (Li et al., 2020; Zhou et al., 2021; Su et al., 2021; Biś et al., 2021; Liu et al., 2023) increased isotropy. This approach mitigates unargmaxable representations by evenly spreading them in all directions, preventing them from being within the interior points of the convex hull. This indiscriminate spreading of representations, without considering queries, is effective for word or sentence level tasks such as Semantic Textual Similarity. However, it is challenging to apply this approach in IR, since thoughtless dispersion may inadvertently render other passages non-referentiable. If all gold queries for each passage were known in advance, it would be straightforward to make all passages referentiable by appropriately spreading

[*] Corresponding author

Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 5345–5358
April 29 - May 4, 2025

passage representations towards their corresponding gold queries. However, this is impractical in real-world scenarios.

To address this limitation, we propose Learning Referentiable Representation (LRR) leveraging two strategic metrics, Self-P and Self-Q. These metrics address two types of gold queries: (1) narrow intent queries, finding specific information (Hosey et al., 2019), and (2) broad intent queries focused on diverse information (Li et al., 2019). **First, Self-P**: Narrow intent queries contain substantial information overlapping with their relevant passages. Thus, Self-P uses the passage itself as a proxy for narrow intent query, and quantifies whether each passage is referenced by its own representation, namely passage locality. Optimizing Self-P encourages passage locality, ensuring the passage can be referenced as top-1 when using itself as gold query. **Second, Self-Q**: To address broad intent queries, we predict diverse intents from the given passage using query generator. Self-Q uses their distribution as broad intent gold queries, and quantifies whether the passage can be referenced as top-1 by predicted intent query. Optimizing Self-Q expands passage representations specifically toward the predicted intent query distribution, in contrast to isotropy, which uniformly expands representations in all directions. Therefore, LRR optimizes both Self-P and Self-Q together, aiming to 1) ensure referentiable passages through passage locality and 2) achieve referentiable passages through space-effective expansion.

To demonstrate the effectiveness of our approach for dense passage retrieval in zero-shot setting, we compare three versions of dense models: Naive, Isotropic, and Referentiable. We then show that our approach generalizes from well-established models like DPR, Condenser, and coCondenser to more recent models such as E5 (Wang et al., 2022) and BGE (Xiao et al., 2023). In Table 2, our approach notably outperforms the naive and isotropic versions in full-ranking retrieval, underscoring the importance of LRR for better zero-shot performance. Our contributions can be summarized as the following.

- We identify challenges in adapting the concepts of argmaxable and isotropy to IR.
- We define a novel query-focused concept called referentiable.
- We propose two metrics, Self-P and Self-Q, for Learning Referentiable Representation (LRR).
- We demonstrate the benefits of LRR for zero-shot performance on BEIR benchmark.

2 Preliminary

In this section, we review existing approaches in IR, and the concepts of argmaxable and isotropy.

2.1 Dense models and Zero-shot retrieval

BERT (Devlin et al., 2019) has shown promise for dense retrieval tasks by encoding both queries and passages into embeddings. Karpukhin et al. (2020) introduced DPR, which fine-tunes BERT with a dual-encoder architecture. Gao and Callan (2021) proposed Condenser, enhancing attention mechanism to aggregate information onto CLS token. Gao and Callan (2022) introduced coCondenser, incorporating an unsupervised corpus-level contrastive loss to refine representation space. Recently, Wang et al. (2022) proposed E5 model, which learns general-purpose embeddings with weak supervision, and Xiao et al. (2023) trained BGE model using RetroMAE (Xiao et al., 2022). During the fine-tuning step, these models are trained with contrastive loss as follows:

$$\mathcal{L}_{\text{CL}} = -\log \frac{\exp(\text{sim}(q,p^+))}{\exp(\text{sim}(q,p^+)) + \sum\limits_{p^- \in B^-} \exp(\text{sim}(q,p^-))}, \tag{1}$$

where q, p^+, and p^- represent query, positive, and negative passage, respectively. B^- is a set of negative passages, and $\text{sim}(\cdot,\cdot)$ represents similarity function.

Zero-shot retrieval aims to handle queries and passages not seen during training. Dense retrieval models often underperform compared to others such as BM25 (Robertson et al., 2009) and DeepCT (Dai and Callan, 2020) on BEIR (Thakur et al., 2021), indicating weakness in their representations. This paper demonstrates that our approach not only overcomes these limitations by enhancing the representation space but also generalizes from well-established models such as DPR, Condenser, and coCondenser to more robust and recent models like E5 and BGE.

2.2 Argmaxable

Prior work (Demeter et al., 2020) discussed the stolen probability effect in language models, showing that certain word embeddings cannot be selected under argmax function if they lie within the convex hull of other word embeddings. Demeter et al. (2020); Grivas et al. (2022) demonstrated that infrequent tokens or classes might become interior points of convex hull due to their smaller embedding norms. Brody et al. (2023) showed that unargmaxable key vectors in attention mechanism of Transformer architecture (Vaswani et al., 2017) can impair performance. Xu et al. (2023) argued that kNN-LM solves the issue of stolen probabilities by assigning the highest probability to any word. These findings suggest that ensuring representations are argmaxable is beneficial for various tasks. In this paper, we extend the query-agnostic concept of argmaxable to the query-focused notion of referentiable for IR.

2.3 Isotropy

Isotropy refers to how evenly distributed the representations are. Li et al. (2020); Su et al. (2021) revealed that frequent word embeddings in language models tend to form a skewed distribution in embedding space. Puccetti et al. (2022); Rudman and Eickhoff (2023); Rudman et al. (2023) found that outlier dimensions, caused by infrequent tokens, promote anisotropic representations. These skewed distributions and anisotropic space may lead to unargmaxable embeddings by positioning them within the convex hull. To address these issues, Li et al. (2020); Su et al. (2021); Yu et al. (2022); Jung et al. (2023); Ji et al. (2023); Kim et al. (2024) increased isotropy and improved performance.

To measure isotropy, Ethayarajh (2019) proposed Avg-Cos, which computes the average cosine similarity among representations as follows:

$$\text{Avg-Cos} = \frac{1}{|M|^2} \sum_{p_i, p_j \in M} \cos(p_i, p_j),$$

where M i s a collection of sampled representations. To control isotropy, Gao et al. (2019) introduced a regularization term, CosReg, which uses Avg-Cos in the training step. When applied to IR, it can be formulated as follows:

$$\mathcal{L}_{\text{CosReg}} = \mathcal{L}_{\text{CL}} + \frac{\lambda}{|M|^2} \sum_{p_i, p_j \in M} \cos(p_i, p_j), \quad (2)$$

where λ is a hyperparameter set to decrease Avg-Cos, and M denotes a batch of passage representations. However, this regularization might result in non-referentiable representations due to thoughtless dispersion in all directions, highlighting the necessity for LRR.

3 Proposed Methods

In Section 3.1, we define referentiability (R), and reinterpret the concept of argmaxable and increasing isotropy. We then propose two strategic metrics, Self-P and Self-Q, to approximate referentiability in Section 3.2. With these metrics, we introduce Learning Referentiable Representation (LRR) in Section 3.3.

3.1 Referentiability (R)

If we know all gold queries for each passage, we can measure referentiability, denoted as R. This metric quantifies the extent to which a passage is referentiable for a given gold query by measuring its relative similarity compared to the most overrated passage. For a gold query q regarding a passage p, R is defined as follows:

$$\text{R} = R(p, q, V) = \max_{\substack{v \in V \\ v \neq p}} \frac{\text{sim}(q, v)}{\text{sim}(q, p)}, \quad (3)$$

$$\text{R}_{\text{ref}} = \begin{cases} p \text{ is referentiable} & \text{R} < 1 \\ p \text{ is non-referentiable} & \text{R} \geq 1 \end{cases}$$

where V denotes the collection of representations for negative passages for given query q. R_{ref} denotes the condition to determine whether a passage is referentiable. A referentiability greater than or equal to 1 shows that p is not the most relevant for q (i.e., non-referentiable), as it suggests there is another vector v scoring higher than p.

To reinterpret the concepts of argmaxable and isotropy using R, we consider a set of vectors where each vector has R below 1 for given p as follows:

$$\mathcal{S} = \{u \mid R(p, u, V) < 1\}$$

Conceptually, we can visualize this set as Voronoi cell in two dimensional space, where any vector within each cell selects the corresponding passage. Figure 1 illustrates four different scenarios of blue passage. In Figure 1(a), there is no blue passage's cell, suggesting that it can never be referenced by any vector (i.e., $|\mathcal{S}| = 0$). In Figure 1(b), although the blue passage now being argmaxable has corresponding cell (i.e., $|\mathcal{S}| > 0$), it fails to contain the gold query denoted by the blue triangle. Figure 1(c) illustrates the case of increasing isotropy, showing that passages are spread out evenly in all directions (i.e., $\mathcal{S} \leftarrow \mathcal{S} \cup \{v\}$, where v denotes vector around the blue passage, regardless of gold query), but it

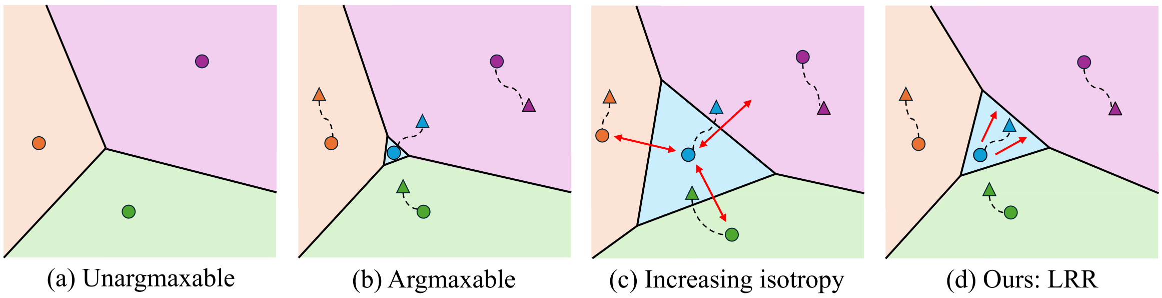
(a) Unargmaxable (b) Argmaxable (c) Increasing isotropy (d) Ours: LRR

Figure 1: Four different scenarios of blue passage using Voronoi cells. Each dot and triangle denote passage and query representation, respectively, with the dotted line indicating relevant query-passage pair. The colored areas represent Voronoi cells where any vector within each area selects the passage representation of corresponding color. (a) The blue passage cannot be referenced because it lacks a Voronoi cell. (b) The blue passage can be referenced, but not by its gold query. (c) Increasing isotropy uniformly expands the Voronoi cell to make it argmaxable, but may cause overlap with other passage's gold query. (d) In contrast, LRR focuses the expansion more precisely towards the direction of the gold query, reducing such overlaps.

Model	**MSMARCO**	**BEIR**
Naive	19.16%	21.71%
Isotropic	18.39%	20.09%

Table 1: Rate of **referentiable** passages for each dataset when encoding passage representations with naive and isotropic versions of coCondenser.

still fails to contain the gold query, and the green passage loses its gold query to the blue passage. Finally, in Figure 1(d), the blue passage being referentiable has corresponding cell, containing its gold query without stealing other passage's gold queries (i.e., $\mathcal{S} \leftarrow \mathcal{S} \cup \{v\}$, where v denotes vector around the blue passage in its gold query direction).

To verify if these scenarios actually occur, we examined whether gold query representations are included within their positive passage cells using the condition R_{ref}. Table 1 shows that only about 20% of passages are referentiable. Additionally, when encoding passages with isotropic model, there are fewer referentiable passages, indicating the stealing of other passages' gold query representations as depicted in Figure 1(c).

3.2 Self-P and Self-Q

In practice, optimizing R during training is infeasible due to the unknown set of gold queries and the large size of V. To address these challenges, we approximate R with Self-P and Self-Q, using passage itself and pseudo query as proxy, respectively.

Self-P: Using Passage Given that narrow intent query contains specific information about its relevant passage, the passage itself serves as a reliable approximation of narrow intent gold query. Thus, we propose Self-P, which approximates R by using the passage itself instead of gold query. Self-P can be formulated by substituting the unknown gold query with the passage itself in Eq. (3) as follows:

$$\text{Self-P} = R(p, p, V) = \max_{\substack{v \in V \\ v \neq p}} \frac{\text{sim}(p, v)}{\text{sim}(p, p)}, \qquad (4)$$

$$\text{Self-P}_{\text{ref}} = \begin{cases} p \text{ is referentiable} & \text{Self-P} < 1 \\ p \text{ is non-referentiable} & \text{Self-P} \geq 1 \end{cases}$$

where $\text{Self-P}_{\text{ref}}$ denotes the condition to determine whether p is referentiable for p itself.

However, computing Self-P in training step still remains unfeasible due to the large size of V. From Eq. (4), we notice that the vectors v causing p to be non-referentiable are clustered around p. Thus, we can replace V with local neighbor passages without compromising accuracy. Formally, Self-P can be approximated with local neighbor passages as follows:

$$\text{Self-P} = R(p, p, V) \approx R(p, p, f(p)), \qquad (5)$$

where f is local neighbor function. The set $f(p)$ consists of passages, which are local neighbors closely related to p. By default, we adopt BM25 as local neighbor function, and obtain $f(p)$ by treating p as query.

Optimizing Self-P suggests passage locality, ensuring that each passage is referenced by its own representation. Conversely, a passage being non-referentiable with $\text{Self-P}_{\text{ref}}$ indicates that it cannot be selected even if the query is identical to the passage. For example, let p_1=(2,1), p_2=(0.5,0.5). The dot product between p_1 and p_2 is 1.5, whereas the dot-product between p_2 and itself is 0.5, which is lower than the dot product between p_1 and p_2. This indicates that p_2 is non-referentiable with Self-P$_{\text{ref}}$. We demonstrate the prevalence of such passages in Section 4.2.1.

Self-Q: Using Pseudo Query To address broad intent queries, we predict diverse intents from given passage using query generator. We then propose Self-Q, which approximates R by using the predicted intents as gold queries. Similar to Self-P, Self-Q can be formulated by replacing the unknown gold query with the predicted intent in Eq. (3) as follows:

$$\text{Self-Q} = R(p, q', V) = \max_{\substack{v \in V \\ v \neq p}} \frac{\text{sim}(q', v)}{\text{sim}(q', p)}, \quad (6)$$

$$\text{Self-Q}_{\text{ref}} = \begin{cases} p \text{ is referentiable} & \text{Self-Q} < 1 \\ p \text{ is non-referentiable} & \text{Self-Q} \geq 1 \end{cases}$$

where q' denotes pseudo query containing predicted intent, and Self-Q$_{\text{ref}}$ represents the condition to determine whether a passages is referentiable for pseudo query. We use docT5query (Nogueira et al.) to generate pseudo queries for given passage. Since Eq. (6) still suffer from the large size of V, we compute Self-Q with local neighbor passages of pseudo query as follows:

$$\text{Self-Q} = R(p, q', V) \approx R(p, q', f(q')), \quad (7)$$

where f is the same local neighbor function used in Eq. (5). The set $f(q')$ consists of local neighbor passages of q', obtained by using q' as the query.

Optimizing Self-Q encourages the expansion of passage representations toward their respective pseudo query directions, whereas increasing isotropy disperses representations indiscriminately in all directions. If pseudo queries accurately mimic the distribution of gold queries, we expect that optimizing Self-Q should reduce the number of non-referentiable passages for gold queries.

3.3 Learning Referentiable Representation (LRR)

In this section, we discuss how to optimize Self-P and Self-Q, and propose LRR as loss function.

As a similarity function, we adopt the dot-product operation since cosine-similarity reportedly leads to significantly lower performance (Karpukhin et al., 2020), failing to reflect the magnitude of vectors. More details are provided in Appendix A.1. With the dot-product operation as similarity function, Self-P and Self-Q can be rewritten as follows:

$$\text{Self-P} \approx \max_{\substack{v \in f(p) \\ v \neq p}} \frac{p \cdot v}{p \cdot p} = \max_{\substack{v \in f(p) \\ v \neq p}} \frac{|v|}{|p|} \frac{\cos(p, v)}{\cos(p, p)} \quad (8)$$

$$\text{Self-Q} \approx \max_{\substack{v \in f(q') \\ v \neq p}} \frac{q' \cdot v}{q' \cdot p} = \max_{\substack{v \in f(q') \\ v \neq p}} \frac{|v|}{|p|} \frac{\cos(q', v)}{\cos(q', p)} \quad (9)$$

where $(\cdot)$ represents dot-product operation and cos denotes cosine similarity.

To make passages referentiable with Self-P$_{\text{ref}}$, we aim to optimize Self-P to ensure it is below 1 for all passage representations. It is important to note that each v can later serve as p in Eq. (8). In other words, making p referentiable might inadvertently render v non-referentiable, since simply decreasing Self-P could diminish the norm of v in Eq. (8), thus making it non-referentiable. Therefore, we decompose Eq. (8) into two parts: (1) the scale term, $|v|/|p|$ and (2) the cosine term, $\cos(p, v)/\cos(p, p)$. Instead of decreasing the scale term, we reduce variance of the norm of passage representations to encourage similar norms for p and v. Meanwhile, the cosine term can be optimized directly. The loss functions for the scale term and the cosine term are written as follows:

$$\mathcal{L}_{\text{scale}}(H) = \text{variance}(H),$$

$$\mathcal{L}_{\text{cos}}(p, q, H) = \frac{1}{|H|} \sum_{\substack{v \in H \\ v \neq p}} \frac{\cos(q, v)}{\cos(q, p)},$$

where H represents the set of local neighbor passages, and variance(H) computes the variance of the norms of elements in H. With these defined loss functions, we can formulate the loss function for Self-P as follows:

$$\mathcal{L}_{\text{Self-P}} = \mathcal{L}_{\text{scale}}(f(p)) + \gamma \mathcal{L}_{\text{cos}}(p, p, f(p)), \quad (10)$$

where γ is a hyperparameter to balance between the scale term and the cosine term.

Similarly, we optimize Self-Q to ensure that it remains below 1 for pseudo query. Although Eq. (9) includes the same scale term as Eq. (8), we focus solely on the cosine term, since optimizing the scale term in Eq. (9) could interfere with the optimization of the scale term in Eq. (8). We then formulate the loss function for Self-Q as follows:

$$\mathcal{L}_{\text{Self-Q}} = \mathcal{L}_{\text{cos}}(p, q', f(q'))$$

Model(→)	Naive			Isotropic			Referentiable		
Dataset(↓)	DPR	Condenser	coCondenser	DPR	Condenser	coCondenser	DPR	Condenser	coCondenser
MS MARCO (dev)	32.63	33.80	35.65	32.61	33.70	35.20	**33.18**	**34.36**	**36.15**
TREC-COVID	**58.69**	**61.45**	**68.8**	58.23	55.29	62.25	56.21	57.92	65.22
BioASQ	20.35	21.90	25.42	20.52	21.33	24.05	**21.37**	**22.66**	**25.69**
NFCorpus	25.23	**26.37**	30.65	24.88	25.40	29.81	**25.46**	26.3	**30.69**
NQ	**40.14**	**41.47**	42.98	39.02	39.88	38.82	39.52	40.92	**43.54**
HotpotQA	39.91	43.43	45.60	38.96	41.96	45.15	**40.84**	**44.41**	**47.50**
FiQA-2018	18.00	**20.01**	**25.04**	**18.40**	19.65	23.43	18.01	19.81	24.84
Signal-1M (RT)	17.99	21.47	24.04	11.64	16.01	16.36	**22.12**	**23.15**	**25.56**
TREC-NEWS	32.18	33.47	34.99	31.57	**34.23**	34.35	**33.51**	34.12	**36.18**
Robust04	**27.28**	**30.05**	34.59	26.47	28.50	33.79	27.01	28.31	**34.66**
ArguAna	23.68	23.99	25.93	23.47	24.32	25.50	**24.45**	**27.30**	**27.94**
Touché-2020	15.91	12.33	13.54	**17.18**	**14.20**	12.94	13.57	12.35	**13.90**
CQADupStack	23.84	24.67	28.57	23.22	24.22	25.75	**25.02**	**25.13**	**29.11**
Quora	80.35	82.30	84.15	9.72	63.02	71.42	**81.18**	**82.61**	**84.72**
DBPedia	26.43	28.69	30.19	26.08	27.68	26.78	**27.27**	**29.26**	**30.99**
SCIDOCS	10.49	11.27	12.30	10.34	11.37	12.74	**10.94**	**11.99**	**13.27**
FEVER	**64.23**	**65.37**	62.30	60.62	63.32	**63.16**	60.55	63.94	62.91
Climate-FEVER	14.81	14.96	12.7	15.48	17.66	**17.80**	**17.13**	**18.52**	17.43
SciFact	46.08	51.62	54.31	46.91	49.57	54.11	**48.10**	**52.18**	**56.11**
Avg.	32.54†	34.14†	36.41†	28.17†	32.17†	34.39†	**32.92**†	**34.49**†	**37.18**†
Avg. w/o MS MARCO	32.53†	34.16†	36.45†	27.93†	32.09†	34.35†	**32.90**†	**34.49**†	**37.24**†

Table 2: nDCG@10 performance in full-ranking on DPR, Condenser, and coCondenser. The best performing results are highlighted in bold for each backbone. The symbol † denotes the results with a p-value < 0.05.

Then, the final loss function for LRR is given by:

$$\mathcal{L}_{\text{LRR}} = \mathcal{L}_{\text{CL}} + \alpha\mathcal{L}_{\text{Self-P}} + \beta\mathcal{L}_{\text{Self-Q}}, \qquad (11)$$

where α and β are hyperparameters, and $\mathcal{L}_{\text{CL}}$ refers to Eq. (1).

4 Experiment

4.1 Experimental Setting

Dataset and Evaluation Metric We use MS MARCO (Nguyen et al., 2016), which consists of 8.8M passages for training, and BEIR (Thakur et al., 2021) for evaluating zero-shot performance. While BEIR collection includes MS MARCO, all evaluations and analyses were conducted excluding MS MARCO. Our primary focus lies in improving the Normalized Discounted Cumulative Gain (nDCG) metric for full-ranking retrieval on BEIR.

Implementation Details To explore our approach, we train three versions of dense models: Naive, Isotropic, and Referentiable, using the loss functions $\mathcal{L}_{\text{CL}}$, $\mathcal{L}_{\text{CosReg}}$, and $\mathcal{L}_{\text{LRR}}$, respectively. For brevity, we focus on coCondenser backbone in Section 4.2, given its robustness as noted in Gao and Callan (2022), and extend the analysis to recent backbones and framework in Section 4.2.4. We further validate our findings with additional backbones, reporting consistent results in Appendix A.3.

Metric	**MSMARCO** (↓)	**BEIR** (↓)
Self-P$_{\text{ref}}$	10.09	2.67
Self-Q$_{\text{ref}}$	79.15	64.68
R$_{\text{ref}}$	80.84	78.29

Table 3: Rate of **non-referentiable** passages on each dataset. The symbol for percent(%) is omitted from each column for simplicity.

Details on hyperparameters are provided in Appendix A.6.

4.2 Experimental Results

Research Questions To evaluate the effectiveness of our approach, we address the following research questions:

- RQ1: How many non-referentiable passages exist in reality?
- RQ2: Does optimizing Self-P and Self-Q enhance performance and why?
- RQ3: Does LRR improve zero-shot performance?
- RQ4: Is LRR approach generalizable?

Model	Self-$\mathbf{P_{ref}}$ (↑)	Self-$\mathbf{Q_{ref}}$ (↑)	$\mathbf{R_{ref}}$ (↑)
Naive	97.33	35.32	21.71
Isotropic	93.33 (-4.00)	31.55 (-3.77)	20.09 (-1.62)
Referentiable	**97.85** (+0.52)	**36.25** (+0.93)	**22.63** (+0.92)

Table 4: Rate of **referentiable** passages on BEIR. The symbol for percent(%) is omitted from each column for simplicity.

4.2.1 RQ1: Non-referentiable passages in reality

In this section, we explore how many non-referentiable passages within MS MARCO and BEIR when encoding with naive coCondenser.

Table 3 shows the rates of non-referentiable passages for each condition. We observed the prevalence of non-referentiable passages for Self-P_{ref}, with rates of 10.09% for MS MARCO and 2.67% for BEIR. These results indicate that even if Self-P_{ref} uses the passage itself as gold query to determine referentiable or not, certain passages are still non-referentiable. The detailed examples are described in Appendix A.2

For MS MARCO, the rate of non-referentiable passages is 79.15% for Self-Q_{ref}, closely aligning with the rate of 80.84% for R_{ref}. This suggests that Self-Q effectively approximates R. The slightly increased discrepancy between Self-Q_{ref} and R_{ref} rates in BEIR may be attributed to docT5query, a query generator trained on MS MARCO.

Overall, these findings underscore a significant proportion of non-referentiable passages, indicating that these passages are not referenced by their respective queries. This highlights the necessity for addressing non-referentiable passages. In the following sections, we demonstrate that our referentiable models effectively tackle this challenge.

4.2.2 RQ2: Effectiveness of Self-P and Self-Q

In this section, we verify the effectiveness of Self-P and Self-Q, and how LRR improves performance.

Learning referentiable passages Table 4 shows the rates of referentiable passages across three version models. We observed that the rate of referentiable passages in isotropic model dropped by 4% for Self-P_{ref} and 3.77% for Self-Q_{ref}. It suggests that isotropic model fails to learn referentiable passages, leading to 1.62% decrease in R_{ref}. In contrast, the referentiable model increases the rate of referentiable passages by +0.52%, +0.93%, and +0.92% for each metric, indicating its effectiveness in learning referentiable passages.

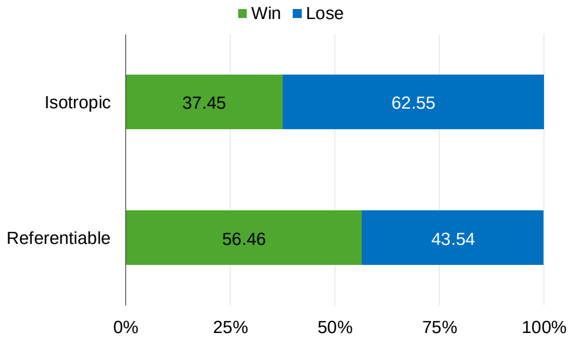

Figure 2: Relative performance on the queries similar to passages in BEIR compared to naive model.

Passage Locality To delve into effect of Self-P, we explore the impact of passage locality as learned by Self-P. As passage locality ensures each passage is referenced by its own representation, we expect the passage to be better referenced by narrow intent query, containing specific information about the passage. With this expectation, we collected the top 10% most frequent terms from queries and passages, respectively. We then subtracted these sets to identify query-specific terms that appear frequently only in queries. Finally, we gathered the queries which contain terms similar to those in passages, excluding any query-specific terms. These narrow intent queries account for 62.36% of total queries in BEIR.

Figure 2 illustrates the relative performance compared to naive model on the collected queries. The results show that referentiable model wins 6.46% more queries compared to naive model, which is 19.01% higher than isotropic model. These findings underscore the effectiveness of passage locality in referentiable model.

Expansions toward pseudo query To explore the effectiveness of Self-Q, we examine the impact of expansion toward pseudo query as learned by Self-Q. Desirably, if generated pseudo queries accurately follow the distribution of gold queries, we expect referentiable model to outperform naive model. With this expectation, on Figure 3, we compare nDCG@10 for each interval of Jaccard score between pseudo queries and gold queries. Interestingly, referentiable model outperforms naive model when increasing jaccard score. Furthermore, we observed similar trends with docT5query when

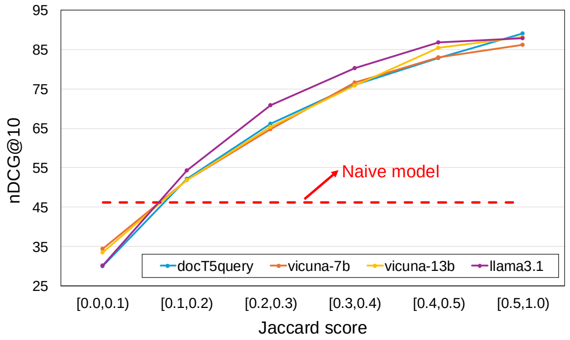

Figure 3: nDCG@10 in full-ranking on BEIR and Jaccard score with different query generators.

Loss	**nDCG@10**	**Self-P$_{\text{ref}}$**	**Self-Q$_{\text{ref}}$**	**R$_{\text{ref}}$**
$\mathcal{L}_{\text{CL}}$	36.45	97.33	35.32	21.71
$+\mathcal{L}_{\text{Self-P}}$ (P)	37.12 (+0.67)	**97.88** (+0.55)	36.01 (+0.69)	22.59 (+0.88)
$+\mathcal{L}_{\text{Self-Q}}$ (Q)	36.59 (+0.14)	97.18 (-0.15)	35.77 (+0.45)	22.11 (+0.40)
+ P and Q	**37.24** (+0.79)	97.85 (+0.52)	**36.25** (+0.93)	**22.63** (+0.92)

Table 5: Ablation study for regularization terms. The numbers indicate nDCG@10 and rate of referentiable passages on BEIR.

using large language models (LLMs) to generate pseudo queries. This trend was most pronounced with the latest LLM, Llama 3.1, indicating that pseudo queries generated by advanced LLMs may enhance referentiable representations.

4.2.3 RQ3: LRR for zero-shot performance

In this section, we evaluate three versions of dense models on BEIR to verify the effect of LRR on zero-shot performance.

Zero-shot performance Table 2 shows zero-shot performance in full-ranking. Remarkably, our referentiable models demonstrate superior performance for all backbones, while isotropic versions perform worse than naive versions. As shown in Table 4, we observed that the performance drop in isotropic versions is due to their failure to learn referentiable passages. Conversely, referentiable models achieve the best rate in all metrics by learning referentiable representation. These results suggest that promoting referentiable passages leads to improvements in zero-shot performance.

Model	**nDCG@10** (↑)	**R$_{\text{ref}}$** (↑)
E5		
Naive	39.43	23.59
Isotropic	39.88 (+0.45)	23.82 (+0.23)
Referentiable	**40.16** (+0.73)	**24.37** (+0.78)
BGE		
Naive	39.62	23.72
Isotropic	38.13 (-1.49)	22.25 (-1.47)
Referentiable	**40.02** (+0.40)	**24.20** (+0.48)

Table 6: Generalization of LRR to recent backbones. The results ensure a p-value below < 0.05.

Model	**nDCG@10** (↑)	**R$_{\text{ref}}$** (↑)
Naive	37.31	22.58
Isotropic	35.68 (-1.63)	21.35 (-1.23)
Referentiable	**38.17** (+0.86)	**23.21** (+0.63)

Table 7: Comparison when all versions are trained using generated pseudo queries.

Ablation study for loss function We conduct an ablation study to investigate the impact of regularization terms. In Table 5, we observed that $\mathcal{L}_{\text{Self-P}}$ and $\mathcal{L}_{\text{Self-Q}}$ encourage referentiable passages with Self-P$_{\text{ref}}$ and Self-Q$_{\text{ref}}$, respectively. Additionally, our findings reveal that both $\mathcal{L}_{\text{Self-P}}$ and $\mathcal{L}_{\text{Self-Q}}$ contribute to improvements in R$_{\text{ref}}$. Notably, the individual contributions of both $\mathcal{L}_{\text{Self-P}}$ and $\mathcal{L}_{\text{Self-Q}}$ are evident in the observed improvements, highlighting the effectiveness of each term in zero-shot setting.

4.2.4 RQ4: Generalizability of LRR

This section shows how our proposed LLR, utilizing R$_{\text{ref}}$ metric, generalizes to more recent backbones and other training scheme.

Recent backbones To confirm generalizability of our approach, we use more recent robust backbones such as E5 and BGE [1]. Table 6 shows that the referentiable versions of both E5 and BGE models increase the rate of referentiable passages by +0.78% and +0.48%, respectively. These enhanced representations also lead to performance improvements of +0.73 and +0.4 in nDCG@10.

[1] We used intfloat/E5-base-v2 model released in May 2023 and BAAI/bge-base-en-v1.5 model released in September 2023.

Our findings indicate that the proposed concept of referentiability remains important even in the latest dense retrieval models, and enhancing it leads to improved zero-shot performance.

Contrastive loss with pseudo queries In IR, training with pseudo queries is a common methodology to enhance performance (Liang et al., 2020; Ma et al., 2023). To validate our approach within this framework, we trained coCondenser by adding contrastive loss calculated with pseudo queries to final loss. In Table 7, we confirm a consistent trend where nDCG@10 improves with an increasing rate of referentiable passages, suggesting the validity of our approach in other frameworks.

5 Conclusion

We tackle the challenge of improving dense representations in zero-shot IR setting. While prior studies have focused solely on increasing isotropy to ensure being argmaxable, we introduce a novel query-focused concept, referentiable. Building on this, we proposed LRR to learn referentiable representations. Our extensive analysis demonstrates effectiveness of LRR in improving the representation space for IR tasks.

6 Limitations

We demonstrated the effectiveness of referentiable representations in single-vector dense retrieval models. Extending LRR to multi-vector retrieval is straightforward by ensuring that each passage's multiple representations can be referenced as top-1 by at least one of the query's multiple representations. However, multi-vector retrieval introduces additional challenges beyond referentiability, such as determining the importance of each vector, which is an interesting direction for future work.

We used docT5query to compute Self-Q for aligning with gold query distributions and observed similar trends with Llama 3.1 and Vicuna models. Leveraging more powerful large language model, along with careful preprocessing to handle irrelevant pseudo queries, could further improve referentiable passage representations.

Acknowledgements

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (RS-2024-00414981). This work has been financially supported by SNU-NAVER Hyperscale AI Center. This work was also supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) [NO.RS-2021-II211343, Artificial Intelligence Graduate School Program (Seoul National University)].

References

Daniel Biś, Maksim Podkorytov, and Xiuwen Liu. 2021. Too much in common: Shifting of embeddings in transformer language models and its implications. In *Proceedings of the 2021 conference of the North American chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5117–5130.

Shaked Brody, Uri Alon, and Eran Yahav. 2023. On the expressivity role of layernorm in transformers' attention. *arXiv preprint arXiv:2305.02582*.

Zhuyun Dai and Jamie Callan. 2020. Context-aware term weighting for first stage passage retrieval. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 1533–1536.

David Demeter, Gregory Kimmel, and Doug Downey. 2020. Stolen probability: A structural weakness of neural language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2191–2197.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? comparing the geometry of bert, elmo, and gpt-2 embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 55–65.

Jun Gao, Di He, Xu Tan, Tao Qin, Liwei Wang, and Tie-Yan Liu. 2019. Representation degeneration problem in training natural language generation models. *arXiv preprint arXiv:1907.12009*.

Luyu Gao and Jamie Callan. 2021. Condenser: a pre-training architecture for dense retrieval. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 981–993.

Luyu Gao and Jamie Callan. 2022. Unsupervised corpus aware language model pre-training for dense passage retrieval. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2843–2853.

Andreas Grivas, Nikolay Bogoychev, and Adam Lopez. 2022. Low-rank softmax can have unargmaxable classes in theory but rarely in practice. In *60th Annual Meeting of the Association for Computational Linguistics*, pages 6738–6758. Association for Computational Linguistics.

Christine Hosey, Lara Vujović, Brian St. Thomas, Jean Garcia-Gathright, and Jennifer Thom. 2019. Just give me what i want: How people use and evaluate music search. In *Proceedings of the 2019 chi conference on human factors in computing systems*, pages 1–12.

Yixin Ji, Jikai Wang, Juntao Li, Hai Ye, and Min Zhang. 2023. Isotropic representation can improve zero-shot cross-lingual transfer on multilingual language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 8104–8118.

Euna Jung, Jungwon Park, Jaekeol Choi, Sungyoon Kim, and Wonjong Rhee. 2023. Isotropic representation can improve dense retrieval. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 125–137. Springer.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781.

JaeYoung Kim, Dohyeon Lee, and Seung won Hwang. 2024. Hil: Hybrid isotropy learning for zero-shot performance in dense retrieval. In *2024 Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

Ang Li, Jennifer Thom, Praveen Chandar, Christine Hosey, Brian St Thomas, and Jean Garcia-Gathright. 2019. Search mindsets: Understanding focused and non-focused information seeking in music search. In *The World Wide Web Conference*, pages 2971–2977.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9119–9130.

Davis Liang, Peng Xu, Siamak Shakeri, Cicero Nogueira dos Santos, Ramesh Nallapati, Zhiheng Huang, and Bing Xiang. 2020. Embedding-based zero-shot retrieval through query generation. *arXiv preprint arXiv:2009.10270*.

Siyang Liu, Naihao Deng, Sahand Sabour, Yilin Jia, Minlie Huang, and Rada Mihalcea. 2023. Task-adaptive tokenization: Enhancing long-form text generation efficacy in mental health and beyond. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 15264–15281.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. 2023. Query rewriting in retrieval-augmented large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5303–5315.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. Ms marco: A human generated machine reading comprehension dataset. In *CoCo@ NIPS*.

Rodrigo Nogueira, Jimmy Lin, and AI Epistemic. From doc2query to docttttquery.

Giovanni Puccetti, Anna Rogers, Aleksandr Drozd, and Felice Dell'Orletta. 2022. Outlier dimensions that disrupt transformers are driven by frequency. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 1286–1304.

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

William Rudman, Catherine Chen, and Carsten Eickhoff. 2023. Outlier dimensions encode task specific knowledge. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14596–14605.

William Rudman and Carsten Eickhoff. 2023. Stable anisotropic regularization. In *The Twelfth International Conference on Learning Representations*.

Jianlin Su, Jiarun Cao, Weijie Liu, and Yangyiwen Ou. 2021. Whitening sentence representations for better semantics and faster retrieval. *arXiv preprint arXiv:2103.15316*.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*.

Shitao Xiao, Zheng Liu, Yingxia Shao, and Zhao Cao. 2022. Retromae: Pre-training retrieval-oriented language models via masked auto-encoder. *arXiv preprint arXiv:2205.12035*.

Shitao Xiao, Zheng Liu, Peitian Zhang, and N Muennighof. 2023. C-pack: packaged resources to advance general chinese embedding. 2023. *arXiv preprint arXiv:2309.07597*.

Frank F Xu, Uri Alon, and Graham Neubig. 2023. Why do nearest neighbor language models work? In *International Conference on Machine Learning*, pages 38325–38341. PMLR.

Sangwon Yu, Jongyoon Song, Heeseung Kim, Seongmin Lee, Woo-Jong Ryu, and Sungroh Yoon. 2022. Rare tokens degenerate all tokens: Improving neural text generation via adaptive gradient gating for rare token embeddings. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 29–45.

Wenxuan Zhou, Bill Yuchen Lin, and Xiang Ren. 2021. Isobn: Fine-tuning bert with isotropic batch normalization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14621–14629.

A Appendices

A.1 Similarity function

As similarity function, cosine-similarity or dot-product are commonly used. However, cosine-similarity is reportedly associated with significantly lower performance, as noted in previous study (Karpukhin et al., 2020). This is attributed to its failure to reflect the magnitude of vectors. Thakur et al. (2021) demonstrated that cosine-similarity uses vectors of unit length, thus lacking a notion of the encoded text length. This deficiency is particularly important in our scenario, where the length of passages (Thakur et al., 2021) differ significantly.

A.2 Examples of non-referentiable passages

We analyze the non-referentiable cases by using the passage itself as a query. As shown in Table 8, when the first passage in each section (denoted as the query) serves as the query, the second passage (denoted as top-1) receives the highest score. This indicates that the dot-product score between the first passage and itself is lower than the score between the first and second passages. These non-referentiable passages contain similar but distinct content, highlighting the need for optimization using the LRR approach.

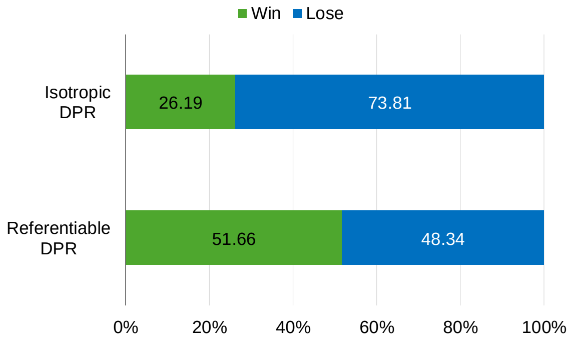

Figure 4: Relative performance on the queries similar to passages in BEIR compared to naive model, when using DPR as backbone.

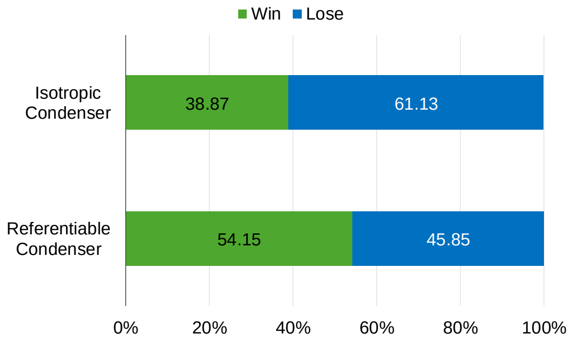

Figure 5: Relative performance on the queries similar to passages in BEIR compared to naive model, when using Condenser as backbone.

A.3 Analysis with other backbones

In this section, we conduct analysis using DPR and Condenser as backbone model.

Table 9 presents zero-shot performance and rates of referentiable passages for each condition. We observed that zero-shot performance improves with an increasing number of referentiable passages, showing the same tendency as when using coCondenser. Figure 4 and Figure 5 shows that relative performance on the queries similar to passages, described in Section 4.2.2, when using DPR and Condenser as backbone, respectively. We also found that referentiable DPR wins 1.66% more queries compare to naive DPR, and referentiable Condenser wins 4.15% more queries to naive Condenser. Figure 6 demonstrates that referentiable models outperform as the Jaccard score between pseudo queries and gold queries increases. It sug-

Examples of non-referentiable passage pairs
Passage 1 (query): Posttraumatic stress symptoms and attitude toward crisis mental health services ... (omitted) Passage 2 (top-1): Beliefs towards the COVID-19 pandemic among patients with emotional disorders in China ... (omitted)
Passage 1 (query): Diversity of Coronaviruses in Bats: Insights Into Origin of SARS Coronavirus ... (omitted) Passage 2 (top-1): Genomic characterization and infectivity of a novel SARS-like coronavirus in Chinese bats ... (omitted)

Table 8: Examples of non-referentiable passages.

Model	**nDCG@10**	**Self-P$_{\text{ref}}$**	**Self-Q$_{\text{ref}}$**	**R$_{\text{ref}}$**
DPR				
Naive	32.53	95.84	33.02	19.41
Isotropic	27.93 (-4.6)	88.41 (-7.43)	29.30 (–3.72)	15.83 (-3.58)
Referentiable	**32.90** (+0.37)	**97.55** (+1.71)	**33.38** (+0.36)	**19.83** (+0.42)
Condenser				
Naive	34.16	96.76	33.75	20.56
Isotropic	32.09 (-2.07)	92.45 (-4.31)	32.81 (-0.94)	18.92 (-1.64)
Referentiable	**34.49** (+0.33)	**97.56** (+0.8)	**34.23** (+0.48)	**21.09** (+0.53)

Table 9: nDCG@10 and rate of referentiable passages for each condition on BEIR.

Model	f	**nDCG@10** (↑)	**R$_{\text{ref}}$** (↑)
Naive	-	36.45	21.71
Referentiable	Naive	36.94 (+0.49)	22.52 (+0.81)
Referentiable	BM25	**37.24** (+0.79)	**22.63** (+0.92)

Table 10: nDCG@10 in full-ranking and rate of referentiable passages on BEIR with different local neighbor function (f). We use coCondenser as backbone.

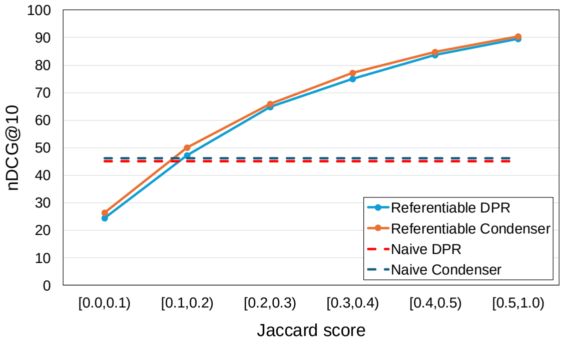

Figure 6: Relative performance on the queries similar to passages in BEIR compared to naive model, when using Condenser as backbone.

Prompt
Write a query that questions the given passage. Passage: {passage} Query:

Table 11: Prompt for generating query.

gests that our hypothesis still holds when using DPR and Condenser as backbone models, indicating that if the generated pseudo queries accurately follow the distribution of gold queries, the referentiable model would outperform the naive model.

A.4 Local neighbor function (f)

We explore local neighbor function, addressing the large size of collection of all passage representations V in Eq. (3). Table 10 presents the zero-shot performance of referentiable models with different local neighbor functions. We observed an improvement in performance and the rate of referentiable passages, when using a trained naive model as the local neighbor function. Notably, the referentiable passage rate of referentiable model using naive model as local neighbor function are lower than that with BM25, possibly because BM25 is sufficiently strong in zero-shot settings. It is important to note that the tendency of referentiable passages to enhance zero-shot performance still holds, even when using the naive model as the local neighbor function.

A.5 Query generation with LLMs

To generate pseudo queries, we explore large language models (LLMs) such as Llama 3.1, Vicuna-7B-v1.5 and Vicuna-13B-v1.5 [2], with the prompt described in Table 11. We observed that LLMs often generate irrelevant queries for a given pas-

[2] We use Llama 3.1 with ollama framework in `https://github.com/ollama/ollama`, and Vicuna-7B-v1.5 and Vicuna-13B-v1.5 described in `https://github.com/lm-sys/FastChat`.

Model	**nDCG@10** (↑)	$\mathbf{R_{ref}}$ (↑)
Isotropic (I-STAR)	29.20	17.90
Referentiable	**37.24** (+8.04)	**22.63** (+4.73)

Table 12: Comparison with I-STAR regularization.

sage. Table 13 shows examples of these irrelevant queries. In the first example, LLM generates a query that includes the answer, while in the second, it repeats text from the prompt. We discarded such queries when computing Self-Q.

A.6 Detailed hyperparameters

We train all models for 10 epochs using AdamW (Loshchilov and Hutter, 2018) optimizer. The learning rate is set to 2e-5 with 3,000 warm-up steps and linear decay. Hard negatives are sampled from top-100 BM25 retrieval results, and each query is paired with one positive and one hard negative. We use a batch size of 128 on MS MARCO-passage dataset. Additionally, we employ 3 local neighbors and 3 pseudo queries for computing Self-P and Self-Q during training. The hyperparameter λ in Eq. (2) is set to 0.1. The hyperparameters γ, α, and β in Eq. (10) and Eq. (11) are set to 0.1, 0.01, and 0.1, respectively. These values are chosen to ensure that the regularization term contributes approximately one-tenth of the total loss, suggesting potential for improvement through hyperparameter tuning. For BM25, we utilize the open-source implementation provided by Pyserini[3].

A.7 Other baseline

Since our goal is to compare against isotropic representations, we opted for the more widely used CosReg. However, for thorough experiments, we confirmed the results using I-STAR (Rudman and Eickhoff, 2023) as the regularization term with a positive λ of 0.1, to learn isotropic representations. As shown in Table 12, the results are consistent with our findings, indicating that I-STAR regularization also fails to learn referentiable representations.

[3] https://github.com/castorini/pyserini

Example 1: **Passage**
If you can look past its bargain interior and anonymous exterior, the Suzuki SX4 is one of the most reliable and affordable all-wheel-drive cars.
Example 1: **Generated query**
What does the word “bargain” mean in the passage? A. A luxurious or expensive item B. A high-quality item that is worth the price C. A low-priced item that is of low quality D. A moderately priced item that is of
Example 2: **Passage**
Check for operating system information in Windows 10. 1 Go to Start , enter About your PC, and then select About your PC. 2 Look under PC for Edition to find out which version and edition of Windows that your PC is running. 3 Look under PC for System type to see if you’re running a 32-bit or 64-bit version of Windows.
Example 2: **Generated query**
Write a query that questions the given passage. Passage: Check for operating system information in Windows 10. 1 Go to Start , enter About your PC, ... (omitted)

Table 13: Examples of irrelevant queries, generated by LLMs for given passage.