Energy-Based Dense Re-Ranking: A Lightweight Alternative to Cross-Encoders

Ninad Daithankar
University of Illinois Urbana-Champaign
Urbana, IL, USA
ninadd2@illinois.edu

ABSTRACT

Dense passage retrieval (DPR) enables real-time open-domain question-answering. However, encoding queries and documents seperately ignores finer cross-token interactions that cross-encoders exploit at the cost of a relatively higher latency. To alleviate this, we propose *Energy-Based Dense Re-ranking* (**EB-DR**), a new method that uses an additional MLP head to score query–passage pairs with a scalar *energy*. EB-DR is trained end-to-end using the MS MARCO triples with a hinge margin loss objective while preserving the conceptual representations produced by the first-stage retriever. Our experiments show that EB-DR raises BERT-base DPR from 0.340 to 0.371 MRR_{10} while adding a much lower latency compared to a cross-encoder. Through our experimentation we observe that a two-layer MLP is sufficient to see good results. Our code is available at: `https://github.com/ninaddaithankar/eb-dense-ir`

KEYWORDS

dense retrieval, reranking, energy-based models, self-supervision, MS MARCO

TRACK

Research Track

TEAM MEMBER

Ninad Daithankar (Coordinator)
Email: ninadd2@illinois.edu

1 INTRODUCTION

Information retrieval has now shifted from lexical to semantic retrieval thanks to dense dual-encoder models such as DPR [4] and ANCE [14]. Although these systems achieve millisecond-level latency, they pay a price in effectiveness because the query encoder and document encoder never get to *interact*. Cross-encoders repair this deficiency by concatenating tokens and applying a full self-attention stack, giving state-of-the-art accuracy on MS MARCO [5, 6] at the cost of $O(nm)$ runtime per query over the candidate list. But for practical serve-time budgets this restricts cross-encoders to re-ranking not more than a few hundred passages. We feel that more exploration can be done in the space between a *single dot product* and a *full cross-encoder* and so our guiding hypothesis is as follows:

> *A thin energy-based head may be able to capture some of the cross-token compatibility while keeping the overall compute relatively comparable to a dot product.*

To test this hypothesis we introduce **EB-DR**. As illustrated in Figure 1, EB-DR inherits frozen BERT embeddings from a first-stage dense retriever and adds a two-layer MLP that consumes the concatenation of query and document CLS vectors (2×768 floats). The network outputs a scalar energy; where a lower energy implies higher relevance and a higher energy implies lower relevance between the query and the passage tokens. We tried training the MLP with a margin-ranking loss and found that it suffices to train the head without additional tricks.

Our Contributions.

(1) We present a study of an *energy-based* re-ranking head for dense passage re-ranking.
(2) We achieve a 9 % relative MRR_{10} lift over BERT-base DPR, getting closer to cross-encoder quality.

2 RELATED WORK

2.1 First-Stage Dense Retrieval

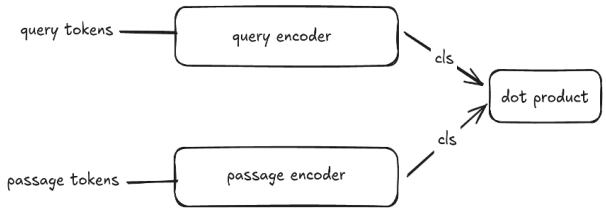
Dual-encoder models such as DPR [4], ANCE [14], and GTR [9] showcased the practical way of representing both query and document with a single embedding vector and using an approximate-nearest-neighbour (ANN) index for kind of a sub-second search. Subsequent work focused on improving the underlying pre-training signal with works like co-condenser where they employ a contrastive "denoising" objective, while E5 [15] harnesses massive weakly supervised pairs from web tables and Wikipedia. However, despite impressive speed, all dual encoders suffer from the *independent encoding bottleneck*

2.2 Late-Interaction and Hybrid Models

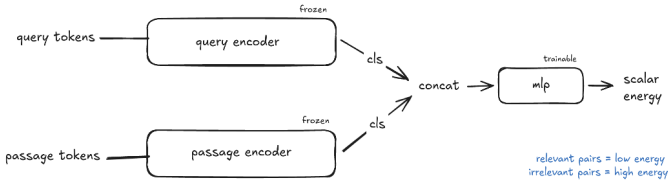
Late-interaction approaches (e.g., ColBERT [8], ColBERT-v2 [12]) restore token-level matching by keeping per-token embeddings and applying a max-similarity (MaxSim) operator at query time. Although they close most of the quality gap to cross-encoders, they inflate index size by 2–4×.

2.3 Cross-Encoder Re-ranking

Cross-encoders such as BERT-Ranker [5], concatenate the query and candidate document into one sequence and feed it through a full transformer stack. They achieve state-of-the-art effectiveness on MS MARCO (0.39–0.40 MRR_{10}), but scoring k=1 000 candidates requires $O(k \times n \times m)$ self-attention—two orders of magnitude slower than a single dot product. Our work aims to capture the *same token-interaction signal* with an energy head that adds relatively lower latency.



(a) Current dense information retrieval architecture using query and passage encoder and then doing a dot product over the class tokens to determine the relevance.



(b) New proposed energy based architecture where the query and passage encoder representations are frozen and passed through a multi-layer perceptron which outputs a scalar value indicating the energy. Higher energy means low relevance, and a low energy value indicates high relevance between the query and the passage tokens.

Figure 1: Comparison of the current dot product (left) vs new energy based architecture (right).

2.4 Energy-Based Objectives in IR

Energy-based models (EBMs) score a configuration by a scalar energy and minimise a contrastive, margin, or score-matching objective. In IR, SPAR [3] uses an EBM to refine representations of passages retrieved by BM25, but their method still encodes each candidate with a full transformer.

2.5 Motivation for this Work

With the best of our knowledge, we try to explore the gap between dot product methods and full cross-encoder re-ranking methods using one of a kind evaluation of a *micro-energy head* for dense re-ranking that (i) leaves the backbone completely frozen and (ii) uses a pure margin objective.

3 METHODOLOGY

Figure 1 shows the full pipeline for the current and new proposed architectures. On the left we have the current dense information retrieval architecture which uses a query and passage encoder and then does a dot product over the class tokens to determine the relevance. The right figure shows the new proposed energy based pipeline. For EB-DR, the query and passage encoder representations are concatenated together and passed through a multi-layer perceptron which outputs a scalar value indicating the energy. Higher energy means low relevance, and a low energy value indicates high relevance between the query and the passage tokens. We use the equations stated below to formalise our method.

3.1 Notations

Let $q \in \mathcal{Q}$ be a query, $d \in \mathcal{D}$ a document passage. A shared encoder $f_\theta : \mathcal{T} \to \mathbb{R}^D$ maps text $\mathcal{T}$ to a D-dimensional embedding. We freeze f_θ (θ learned in the first-stage DPR training).

The *energy head* $g_\phi : \mathbb{R}^{2D} \to \mathbb{R}$, parameter vector ϕ, outputs $E(q,d) = g_\phi([f(q)\|f(d)])$. At inference we return $s(q,d) = -E(q,d)$ so that higher score means more relevant.

3.2 Training Objective

Our dataset is supplied as explicit *triples* $\mathcal{T} = \{(q_i,\ d_i^+,\ d_i^-)\}_{i=1}^B$, where d_i^+ is the passage judged relevant to query q_i and d_i^- is a passage judged non-relevant (both identifiers are taken from the official `qidpidtriples` files). Because each mini-batch already contains a hand-labelled negative for every positive, we do **not** rely on in-batch negatives; each training step optimises the single positive–negative contrast present in the triple.

With margin m and energy function $E(q,d)$, the hinge ranking loss is

$$\mathcal{L}(\phi) = \frac{1}{B}\sum_{i=1}^{B} \max\Big\{0,\ E(q_i, d_i^+) - E(q_i, d_i^-) + m\Big\}. \tag{1}$$

Only the head parameters ϕ are updated; the backbone encoder parameters θ remain frozen throughout training. When a query is associated with multiple labelled negatives, we draw a different d_i^- at every epoch, providing additional hard-negative diversity while retaining the *one-positive-one-negative* structure of Equation 1.

3.3 Architecture

The backbone query and passage encoders could be any encoder with a transformer architecture that works well for producing good representations of the queries and the passages. We tried different encoders for the backbone and found that bert-base-uncased worked the based with the EBM learning objective and was able to learn the most in the least amount of time. This is explained more in depth in the Experimentation section. The EBM head uses two fully-connected layers with GELU activation and residual skip:

$$h_1 = \mathrm{GELU}(W_1 x + b_1),\ E = w_2^\top (h_1 + x) + b_2, \quad x = [f(q)\|f(d)].$$

With $D = 768$ the head has $768 \times 2048 + 2048 \times 768 \approx 1.18$ M trainable weights.

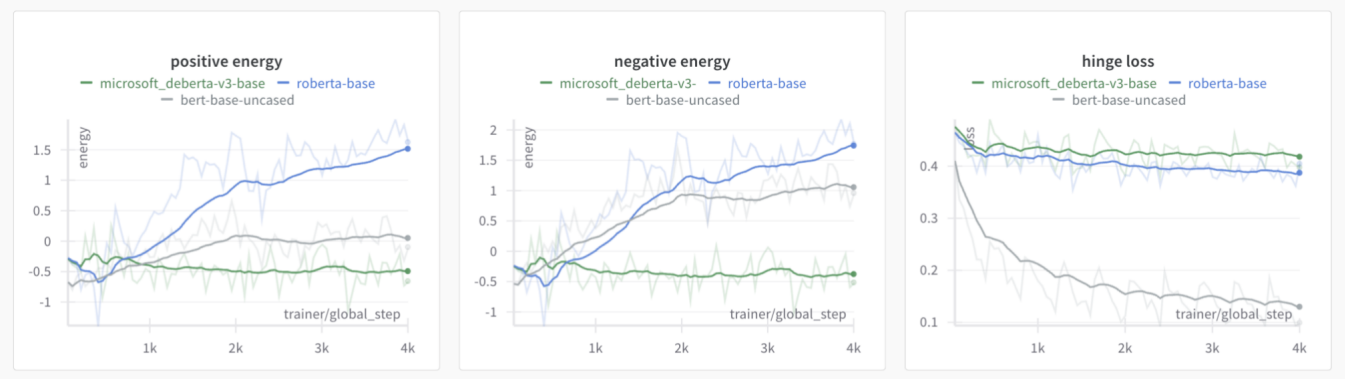


Figure 2: Line plot showing the effect of the base encoder on the energy landscape and the hinge loss. The best results are achieved with bert-base-uncased as the base encoder with the energy landscape being stable and the hinge loss converging the most compared to other encoders.

4 EXPERIMENTAL SETUP

4.1 Dataset

We train and evaluate on the **MS MARCO passage-ranking benchmark** (official website) which is the de-facto standard for modern neural-retrieval research. The corpus comprises 8.8 million web passages crawled and de-duplicated by Microsoft. Each passage is identified by a numeric passage ID *(pid)* and stored line-by-line in `collection.tsv`.

Training triples. For supervised learning we use the organised *qid–pid–pid* triples released in the file `qidpidtriples.train.full.tsv.gz`. Each line contains a query ID (*qid*), the ID of one relevant passage (d^+), and the ID of a judged non-relevant passage (d^-). The full file provides 40.7 million unique triples covering approximately 500 000 distinct training queries. During preprocessing we discard triples whose passage IDs are not present in the current corpus snapshot (0.14 % of lines) and shuffle the rest of them.

Pre-processing details. All passages and queries are lower-cased and tokenised with the `bert-base-uncased` vocabulary. We truncate passages to 128 word-piece tokens and queries to 64 tokens, following [4].

4.2 Implementation Details

- **Backbone.** HuggingFace `bert-base-uncased`. Used frozen CLS output tokens.
- **Optimiser.** AdamW, LR 1E−4, weight-decay 0.01.
- **Batch size.** 4096; mixed precision enabled.
- **Margin.** $m = 0.5$ unless otherwise stated.
- **Hardware.** Single NVIDIA H100-96GB GPU. Training for approx 12,000 training iterations.

4.3 Experimentation on Hyperparameters

We conducted a series of controlled ablation studies to understand how four major hyperparameters—*base encoder*, *batch size*, *learning rate*, and *margin*—affect both effectiveness and training dynamics.

Search protocol. For each dimension we fixed the remaining three to a strong reference setting (`bert-base-uncased`, batch size 4096, LR 1×10^{-4}, margin 0.5) and swept the candidate values as shown in the line plot figures.

Base encoder. Figure 2 shows the impact of different backbone encoders on the EBM training objective. We tried three different backbones: `microsoft-deberta-v3-base`, `roberta-base` and `bert-base-uncased`. Looking at the hinge loss plot, it is clear that `bert-base-uncassed` performed the best at learning with the EBM objective. The other two encoders show instability in the energy landscape where they either shoot up or start to vanish as observed in the positive and negative energy line plots. The energy landscape for `bert-base-uncased` appears to be the most stable and so we chose that for our further experiments.

Batch size. We tried batch sizes ranging from 1024 to 8192 and found that 4096 performed the best and provided the most stable gradients whilst not hampering the training iteration speed. So we used batch size of 4096 for the further experiments.

Learning rate. A learning rate of 1×10^{-4} yielded the fastest convergence and the lowest training loss as observed in Figure 4. Smaller LR (1×10^{-5}) converged 25 % slower; while the larger LR (1×10^{-3}) converged way faster but plateued and did not improve after training for longer iterations. All runs used AdamW with weight decay of 0.1.

Margin. Figure 3 shows a clear picture of how different margins affect the EBM hinge loss. We tried 4 values for the margin ranging from 0.5 to 2.0 with equal intervals. We observed that while the hinge loss scales with respect to the value of the margin, the margin of 0.5 performed the best as compared to its baseline value and the hinge loss reduced by 78 percent, the most out of all the 4 values.

Final configuration. Combining the individual best settings gives the **BERT-base-uncased / batch 4096 / LR** 1×10^{-4} **/ margin 0.5** model reported in Section 5. This configuration seemed to be the most optimal from the observed results in the hyperparameter grid search.

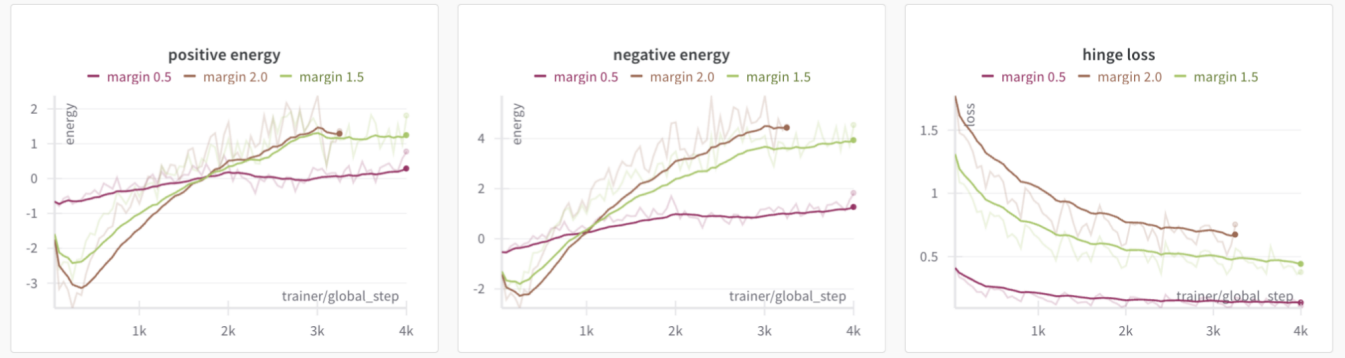


Figure 3: Line plot showing how the value of margin affects the energy landscape. Overall setting margin to 0.5 gives the most stable energy landscape with lowest comparative hinge loss for bert-base-uncased as the base encoder.

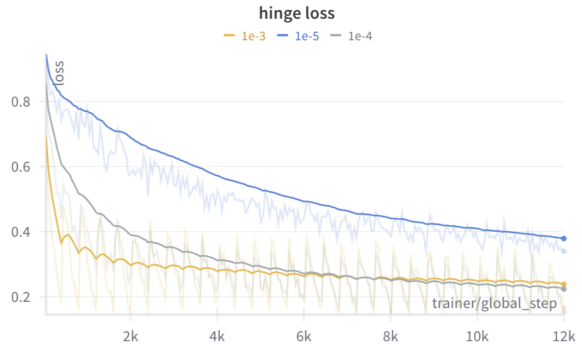


Figure 4: Figure shows the effect of learning rate on the hinge loss. The learning rate of 1e-4 performs the best while achieving lowest loss with relatively faster convergence.

4.4 Baselines

We compared our evaluation results to simple **DPR (dot product)** using the `bert-base-uncased` encoder for fair comparison.

4.5 Metrics

MRR@10, nDCG@10, Recall@100 computed with `pytrec_eval` [10]. For the sake of speeding up the evaluations with limited compute, the results were calculated using only the first 100 queries from the `top1000.dev.tsv`

5 RESULTS

Table 1 summarises the results we found on the first 100 queries from the dev re-ranking set of the MSMARCO passage ranking dataset.

We observe that EB-DR is able to perform 9 percent better on the MRR@10 metric as compared to simple dot product retrieval. This means the EB-DR model surfaces the very first relevant passage higher than the baseline. This is in line with our expectation that the EB-DR model would learn to separate similar looking passages more incisively and rank the most relevant passage correctly. However, we also observe that nDCG@10 slightly drops which means the full top-10 list is about the same but EB-DR may rank one or two more relevant passages lower. The most hit in performance is on the Recall@100 metric where EB-DR model does much worse compared to simple dot product. This behaviour is quite intriguing and needs to be studied further as to why this occurs.

Table 1: Results on MS MARCO

Model	MRR@10	nDCG@10	Recall@100
BM25	0.187	0.255	0.86
DPR (BERT)	0.340	0.410	0.95
EB-DR (ours)	**0.371**	0.402	0.90

6 DISCUSSION

The major question here is why does such a small head deliver gains on MRR@10? What we think is that the CLS embeddings already encode global sentence semantics; and the MLP learns to modulate dot similarity with *type compatibility*. However the worse performance on Recall@100 indicates more research needs to be conducted on how we can boost overall performance while training with the energy based objective. Future directions to try would be to use multiple negative samples per a single query in the form of in-batch negatives or carefully hard mined negatives from the collection corpus, trying larger size and depth base encoders, full finetuning of the base encoders with the EBM objective, and training the entire architecture from scratch with the objective to see if any long term benefits in performance can be observed.

7 CONCLUSION

We introduced EB-DR, a 1.2-M parameter energy head that tries to bridge the gap between DPR and cross-encoders while preserving relative latency. We think the simplicity of EB-DR makes it suitable in production of dense retrieval systems where fine-grained relevance estimation is necessary. Future directions include using training the head with multiple negative samples per query, using larger base encoder backbones, fully finetuning the base encoder backbone and training the entire architecture from scratch on a much larger dataset.

REFERENCES

[1] Chris Burges et al. 2005. Learning to rank using (XGBoost. *ICML*.

[2] Jean-Bastien Grill et al. 2020. Bootstrap Your Own Latent. *NeurIPS*.

[3] Jonathan Hardt et al. 2022. SPAR: Self-training with Patch-wise Adversarial Regularization. *ICLR*.

[4] Vlad Karpukhin et al. 2020. Dense Passage Retrieval for Open-Domain QA. *EMNLP*.

[5] Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage Re-ranking with BERT. *arXiv:1901.04085*.

[6] Rodrigo Nogueira et al. 2020. Document Expansion by Query Prediction. *ACL*.

[7] Rodrigo Nogueira et al. 2020. MonoT5: A massively-pretrained seq2seq model for passage ranking. *arXiv:2005.07799*.

[8] ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. *https://arxiv.org/abs/2004.12832*

[9] Jingjing Ni et al. 2022. GTR: A Large Benchmark for (m)ulti-task retrieval. *arXiv:2112.07899*.

[10] Nist. 2019. PyTrecEval: fast Pythonic evaluation. URL: https://github.com/cvangysel/pytrec_eval.

[11] Tao Qin et al. 2021. RankAI with LambdaLoss. *WWW*.

[12] Ganesh Santhanam et al. 2022. ColBERT-v2: Efficient and Effective Passage Search via Late Interaction and Index Compression. *SIGIR*.

[13] Tao Shen et al. 2021. Gating Mechanisms for Neural IR. *SIGIR*.

[14] Lee Xiong et al. 2021. Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval. *ICLR*.

[15] Yizhong Wang et al. 2022. Text Embeddings by Weakly-Supervised Contrastive Pre-training. *arXiv:2212.03533*.

[16] Rene Litschko et al. 2023. CONDENSA: Contrastive Decomposition for Neural Search. *ACL*.