Information Retrieval with Dense and Sparse Representations

by

Yung-Sung Chuang

B.S., National Taiwan University (2020)

Submitted to the Department of Electrical Engineering and Computer Science

in partial fulfillment of the requirements for the degree of

Masters of Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2024

Authored by: Yung-Sung Chuang
Department of Electrical Engineering and Computer Science
January 3, 2024

Information Retrieval with Dense and Sparse Representations

by

Yung-Sung Chuang

Submitted to the Department of Electrical Engineering and Computer Science
on January 3, 2024, in partial fulfillment of the
requirements for the degree of
Masters of Science

Abstract

Information retrieval, at the core of numerous applications such as search engines and open-domain question-answering systems, relies on effective textual representation and semantic matching. However, current approaches can lose nuanced lexical detail information due to an information bottleneck in dense retrieval, or rely on exact lexical matching and thus overlook the broader contextual relevance when using sparse retrieval. This thesis delves into improving both dense and sparse retrieval systems with advanced language models and training strategies. We first introduce DiffCSE, a difference-based contrastive learning framework for unsupervised sentence embedding and dense retrieval that can effectively capture minor differences in sentences, showcasing improved performance in semantic tasks and retrieval for open-domain question answering. We then address sparse retrieval's limitations by developing a query expansion and reranking procedure. Using pre-trained language models, we propose an expansion and reranking pipeline for better query expansion, achieving superior retrieval results both in-domain and out-of-domain, yet retaining sparse retrieval's computational efficiency. In summary, this thesis provides a comprehensive exploration of advancing information retrieval in the generation of large language models.

Contents

List of Figures

List of Tables

Acknowledgments

This page is written to the countless people who believed in me, cared about me, and supported me throughout this journey.

First and foremost, a huge thank you to my advisor, Jim Glass, who has always supported me in whatever choices I made and whatever research topics I wanted to explore. Thank you for always standing by my side and supporting me, even outside of research.

Thank you Yoon Kim for keeping inspiring me. You are always willing to take the time to brainstorm with me, come up with novel ideas, and even help revise papers, which is really invaluable.

Thanks to Daniel Li and Scott Yih from Meta AI. Thank you, Daniel, for being willing to trust my abilities. Collaboration with you laid the foundation for my research, before I came to MIT. Special thanks also to Scott, who has great insights and experience in this field and was always there to polish and refine my early naive ideas, and get me really thinking about what makes an impactful research question in NLP.

Thank you Yang Zhang, Shiyu Zhang, and Kaizhi Qian - for providing me with a wonderful internship experience at MIT-IBM last summer. Although the internship project did not yield great results, your patience and constant support always kept me motivated.

Many thanks to my lab mates: Hongyin, Wei, Alex, Jeff, Andrew, and Yuan. A special mention goes to Wei, who always discusses a lot with me, and contributed significantly to my EAR paper. Also, thanks Hongyin, Andrew, and Yuan, discussion with you has always encouraged me to continue doing research. I am very thankful to Alex and Jeff for helping me to survive my first few weeks in the U.S. and at MIT. The two of you are like older brothers to me and are role models for my ideal life in the US.

Special mention should be made of Professors Hung-yi Lee, Yun-Nung Chen, and Lin-shan Lee at NTU. Working with you during my undergraduate years was an inspiring and indispensable experience that led to my decision to pursue a PhD degree. I could not have gotten into MIT without your guidance and encouragement.

Thanks to my friends whom I won't list in the space available. You have made my life at MIT so happy and enjoyable.

Finally, to my family: Mom and Dad, thank you for raising me without expecting anything in return. My brother, thank you for your unconditional care and support.

This thesis was partially sponsored by the Centre for Perceptual and Interactive Intelligence (CPII) Ltd under the Innovation and Technology Fund (InnoHK) and the United States Air Force Artificial Intelligence Accelerator.

Bibliographic Note

The content in this thesis has previously been published in peer-reviewed publications. Chapter 2 was published in Chuang et al. (2022) at NAACL 2022. Chapter 3 was published in Chuang et al. (2023) at ACL 2023.

Chapter 1

Introduction

1.1 Dense Retrieval

Existing information retrieval methods fall into two main categories: dense retrieval and sparse retrieval. Dense retrieval methods (Karpukhin et al., 2020) use contrastive learning with dense representations from deep learning models to represent documents and questions, so it can find relevant documents simply by inner product search in the vector space. Dense retrieval can achieve state-of-the-art results on the current benchmarks for open-domain question answering. However, dense retrieval models still suffer from potential information loss when compressing long passages into fixed-dimensional vectors (Luan et al., 2021a), which makes it hard to match rare entities exactly (Sciavolino et al., 2021). Thus, we aim to improve the ability of dense retrieval models to distinguish between minor lexical changes in sentences and consequently make them better at compressing detailed information into dense vectors.

1.2 Sparse Retrieval

In contrast to dense retrieval, sparse retrieval methods only use term frequency, i.e. sparse vectors, to represent dense documents and questions (Robertson et al., 2009). Sparse retrievers can effectively find relevant documents by simply matching keywords. It is a very classic and traditional method for information retrieval that is known for its low-computational cost and efficiency. However, sparse retrieval is often outperformed by dense retrieval because it cannot match relevant documents that have low lexical overlap with the given question. In recent years, sparse retrieval has been surpassed by dense retrieval. However, recent advances in large language models provide new opportunities for sparse retrieval by generative query expansion. Recent studies (Mao et al., 2021b) found that after expanding user questions by adding related contexts, sparse retrieval can also achieve good performance that is close to dense retrieval. Our initial study indicates that this kind of generative query expansion has great potential because when sampling multiple query expansions from the same language model, some of them contain very high-quality query expansions that can outperform dense retrieval significantly. As a result, we propose to train a query reranker to select these high-quality query expansions, so as to unlock the full potential of sparse retrieval.

1.3 Contributions

In Chapter 2, we first focus on dense retrieval. We first recognize that the issue of the current contrastive learning framework in sentence embedding does not encourage the sentence encoder to distinguish between tiny changes in similar but different sentence pairs. Therefore, we propose a difference-based contrastive learning framework, DiffCSE, to capture the tiny differences between original sentences and edited sentences. DiffCSE includes a generator and discriminator framework while using an information bottleneck to compress the semantic details from sentences. We first examine DiffCSE for unsupervised sentence embedding learning and then later switch to dense passage retrieval. In our experiments, DiffCSE not only improves the quality of unsupervised sentence representations, resulting in better performance in semantic textual similarity and sentence classification tasks, but it also proved to be useful when applied to dense retrieval in open-domain question answering.

In Chapter 3, we focus on enhancing sparse retrieval with the power of pre-trained language models. While sparse retrieval can effectively match keywords that occur in both questions and passages with a low computational cost, it still has difficulty finding relevant content without lexical overlap. To address this issue, we propose an Expansion and Reranking (EAR) procedure, which first generates a diverse set of query expansions using pretrained language models, and then leverages the query reranking model to select high-quality queries that are expected to achieve better retrieval results. EAR can effectively generate better queries that contain more lexical overlap with the context, and improve the performance both in in-domain and out-of-domain datasets to outperform dense retrieval models, while inheriting the advantage of sparse retrieval at low computational cost.

This thesis presents a comprehensive exploration of both dense and sparse retrieval methodologies, offering accurate and efficient solutions for information retrieval in the era of large language models.

1.4 Outline

Chapter 2 first provides comprehensive background information including existing techniques for learning sentence embeddings, and the concept of equivariant contrastive learning (Dangovski et al., 2021). Following this concept, we introduce DiffCSE, a new contrastive learning method for unsupervised sentence embeddings. The empirical results contain experiments of semantic tasks, qualitative study, and analysis. The chapter concludes by extending the proposed method to the application of dense passage retrieval.

Chapter 3 starts with comprehensive background details on open-domain question answering and generation-augmented retrieval. The core of this chapter introduces the proposed retrieval-independent/dependent query rerankers. The experiments include in-domain datasets and cross-dataset generalization. A comparative analysis between query reranking and passage reranking is explored. The chapter concludes with discussions on efficiency, related works, and potential limitations. Appendices are provided at the end of this thesis for the experiment details for both Chapter 2 and 3.

Chapter 2

DiffCSE: Difference-based Contrastive Learning for Sentence Embeddings and Dense Retrieval

2.1 Introduction

Learning “universal” sentence representations that capture rich semantic information and are at the same time performant across a wide range of downstream NLP tasks without task-specific finetuning is an important open issue in the field (Conneau et al., 2017; Cer et al., 2018; Kiros et al., 2015; Logeswaran and Lee, 2018; Giorgi et al., 2020; Yan et al., 2021b; Gao et al., 2021). Recent work has shown that finetuning pretrained language models with *contrastive learning* makes it possible to learn good sentence embeddings without any labeled data (Giorgi et al., 2020; Yan et al., 2021b; Gao et al., 2021). Contrastive learning uses multiple augmentations on a single datum to construct positive pairs whose representations are trained to be more similar to one another than negative pairs. While different data augmentations (random cropping, color jitter, rotations, etc.) have been found to be crucial for pretraining vision models (Chen et al., 2020), such augmentations have generally been unsuccessful when applied to contrastive learning of sentence embeddings. Indeed, (Gao et al., 2021) find that constructing positive pairs via a simple dropout-based augmentation works much better than more complex augmentations such as word deletions or replacements based on synonyms or masked language models. This is perhaps unsurprising in hindsight; while the training objective in contrastive learning encourages representations to be *invariant* to augmentation transformations, direct augmentations on the input (e.g., deletion, replacement) often change the meaning of the sentence. That is, ideal sentence embeddings should *not* be invariant to such transformations.

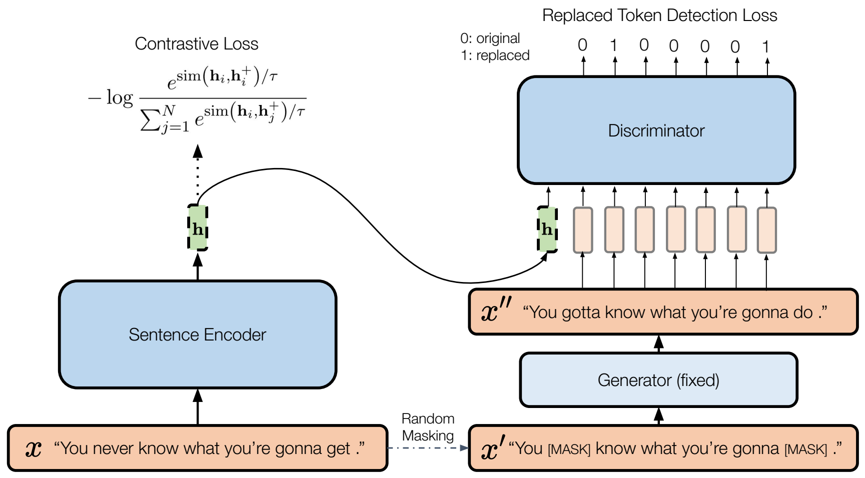

Contrastive Loss
$-\log \frac{e^{\mathrm{sim}(\mathbf{h}_i, \mathbf{h}_i^+)/\tau}}{\sum_{j=1}^{N} e^{\mathrm{sim}(\mathbf{h}_i, \mathbf{h}_j^+)/\tau}}$
Replaced Token Detection Loss
0: original
1: replaced
0 1 0 0 0 0 1
Discriminator
h
h
Sentence Encoder
x'' "You gotta know what you're gonna do ."
Generator (fixed)
Random Masking
x "You never know what you're gonna get ."
x' "You [MASK] know what you're gonna [MASK] ."

Figure 2-1: Illustration of DiffCSE. On the left-hand side is a standard SimCSE model trained with regular contrastive loss on dropout transformations. On the right hand side is a conditional difference prediction model which takes the sentence vector **h** as input and predict the difference between x and x''. During testing we discard the discriminator and only use **h** as the sentence embedding.

We propose to learn sentence representations that are *aware* of, but not necessarily invariant to, such direct surface-level augmentations. This is an instance of *equivariant* contrastive learning (Dangovski et al., 2021), which improves vision representation learning by using a contrastive loss on *insensitive* image transformations (e.g., grayscale) and a prediction loss on *sensitive* image transformations (e.g., rotations). We operationalize equivariant contrastive learning on sentences by using dropout-based augmentation as the insensitive transformation (as in SimCSE (Gao et al., 2021)) and MLM-based word replacement as the sensitive transformation. This results in an additional cross-entropy loss based on the *difference* between the original and the transformed sentence.

We conduct experiments on seven semantic textual similarity tasks (STS) and seven transfer tasks from SentEval (Conneau and Kiela, 2018) and find that this difference-based learning greatly improves over standard contrastive learning. Our DiffCSE approach can achieve around 2.3% absolute improvement on STS datasets over SimCSE, the previous state-of-the-art model. We also conduct a set of ablation studies to justify our designed architecture. Qualitative study and analysis are also included to look into the embedding space of DiffCSE.

2.2 Background and Related Work

2.2.1 Recent Advancements in Pretrained Language Models

The landscape of natural language processing (NLP) has been revolutionized by the development of pretrained language models (LMs) since 2018. GPT-1 (Radford et al., 2018) (left-to-right LM) and BERT (Devlin et al., 2019) (masked LM) first introduce the pipeline of pretraining and fine-tuning. Pretraining means training a self-supervised model on unlabeled large corpora, which can be efficiently collected from the internet. Fine-tuning means further optimizing the pretrained model with a small labeled dataset with supervised learning. These pretrained LMs demonstrate superior performance after fine-tuning on various NLP tasks, such as General Language Understanding Evaluation (GLUE) (Wang et al., 2018) and question answering (Rajpurkar et al., 2016). Since BERT, a series of innovative architectures, such as RoBERTa (Liu et al., 2019a), XLNet (Yang et al., 2019), ELECTRA (Clark et al., 2020), and DeBERTa (He et al., 2021) have emerged to enhance the performance of the pretrained LMs. These pretrained LMs have also been applied to learn sentence-level representations by continual training with contrastive objectives, such as Sentence-BERT (Reimers and Gurevych, 2019) and SimCSE (Gao et al., 2021), as well as document-level representations for passage retrieval, such as DPR (Karpukhin et al., 2020).

2.2.2 Learning and Evaluating Sentence Embeddings

Learning universal sentence embeddings has been studied extensively in prior work, including unsupervised approaches such as Skip-Thought (Kiros et al., 2015), Quick-Thought (Logeswaran and Lee, 2018) and FastSent (Hill et al., 2016), or supervised methods such as InferSent (Conneau et al., 2017), Universal Sentence Encoder (Cer et al., 2018) and Sentence-BERT (Reimers and Gurevych, 2019). Recently, researchers have focused on (unsupervised) contrastive learning approaches such as SimCLR (Chen et al., 2020) to learn sentence embeddings. SimCLR (Chen et al., 2020) learns image representations by creating semantically close augmentations for the same images and then pulling these representations to be closer than representations of random negative examples. The same framework can be adapted to learning sentence embeddings by designing good augmentation methods for natural language. ConSERT (Yan et al., 2021b) uses a combination of four data augmentation strategies: adversarial attack, token shuffling, cut-off, and dropout. DeCLUTR (Giorgi et al., 2020) uses overlapped spans as positive examples and distant spans as negative examples for learning contrastive span representations. Finally, SimCSE (Gao et al., 2021) proposes an extremely simple augmentation strategy by just switching dropout masks. While simple, sentence embeddings learned in this manner have been shown to be better than other more complicated augmentation methods.

To effectively evaluate the quality of sentence embedding models, various benchmarks have been proposed. SentEval (Conneau and Kiela, 2018) is a standardized evaluation toolkit for universal sentence representations. The first part of the task is the semantic textual similarity, which evaluates how the cosine distance between two sentences correlates with a human-labeled similarity score through Pearson and Spearman correlations. The second part contains several binary and multi-class classification tasks from a wide range of aspects including sentiment analysis, question type, product reviews, subjectivity/objectivity, opinion polarity, entailment, and paraphrase detection. We use SentEval in the experiments to estimate the quality of learned sentence embeddings.

2.2.3 Equivariant Contrastive Learning

DiffCSE is inspired by a recent generalization of contrastive learning in computer vision (CV) called equivariant contrastive learning (Dangovski et al., 2021). We now explain how this CV technique can be adapted to natural language.

Understanding the role of input transformations is crucial for successful contrastive learning. Past empirical studies have revealed useful transformations for contrastive learning, such as random resized cropping and color jitter for computer vision (Chen et al., 2020) and dropout for NLP (Gao et al., 2021). Contrastive learning encourages representations to be insensitive to these transformations, i.e. the encoder is trained to be invariant to a set of manually chosen transformations. The above studies in CV and NLP have also revealed transformations that are *harmful* for contrastive learning. For example, (Chen et al., 2020) showed that making the representations insensitive to rotations decreases the ImageNet linear probe accuracy, and (Gao et al., 2021) showed that using an MLM to replace 15% of the words drastically reduces performance on STS-B. While previous works simply omit these transformations from contrastive pre-training, here we argue that we should still make use of these transformations by learning representations that are *sensitive* (but not necessarily invariant) to such transformations.

The notion of (in)sensitivity can be captured by the more general property of equivariance in mathematics. Let T be a transformation from a group G and let $T(x)$ denote the transformation of a sentence x. Equivariance is the property that there is an induced group transformation T' on the output features (Dangovski et al., 2021):

$$f(T(x)) = T'(f(x)).$$

In the special case of contrastive learning, T''s target is the identity transformation, and we say that f is trained to be "invariant to T." However, invariance is just a trivial case of equivariance, and we can design training objectives where T' is not the identity for some transformations (such as MLM), while it is the identity for others (such as dropout). (Dangovski et al., 2021) show that generalizing contrastive learning to equivariance in this way improves the semantic quality of features in CV, and here we show that the complementary nature of invariance and equivariance extends to the NLP domain. The key observation is that the encoder should be equivariant to MLM-based augmentation instead of being invariant. We can operationalize this by using a conditional discriminator that combines the sentence representation with an edited sentence, and then predicts the *difference* between the original and edited sentences. This is essentially a conditional version of the ELECTRA model (Clark et al., 2020), which makes the encoder equivariant to MLM by using a binary discriminator which detects whether a token is from the original sentence or from a generator. We hypothesize that conditioning the ELECTRA model with the representation from our sentence encoder is a useful objective for encouraging f to be "equivariant to MLM."

To the best of our knowledge, we are the first to observe and highlight the above parallel between CV and NLP. In particular, we show that equivariant contrastive learning extends beyond CV, and that it works for transformations even without algebraic structures, such as diff operations on sentences. Further, insofar as the canonical set of useful transformations is less established in NLP than is in CV, DiffCSE can serve as a diagnostic tool for NLP researchers to discover useful transformations.

2.3 Difference-based Contrastive Learning

Our approach is straightforward and can be seen as combining the standard contrastive learning objective from SimCSE (Figure 2-1, left) with a *difference prediction* objective which conditions on the sentence embedding (Figure 2-1, right).

Given an unlabeled input sentence x, SimCSE creates a positive example x^+ for it by applying different dropout masks. By using the BERT$_{\mathsf{base}}$ encoder f, we can obtain the sentence embedding $\mathbf{h} = f(x)$ for x (see section 2.4 for how $\mathbf{h}$ is obtained). The training objective for SimCSE is:

$$\mathcal{L}_{\text{contrast}} = -\log \frac{e^{\operatorname{sim}\left(\mathbf{h}_i, \mathbf{h}_i^+\right)/\tau}}{\sum_{j=1}^{N} e^{\operatorname{sim}\left(\mathbf{h}_i, \mathbf{h}_j^+\right)/\tau}},$$

where N is the batch size for the input batch $\{x_i\}_{i=1}^N$ as we are using in-batch negative examples, $\operatorname{sim}(\cdot, \cdot)$ is the cosine similarity function, and τ is a temperature hyperparameter.

On the right-hand side of Figure 2-1 is a conditional version of the difference prediction objective used in ELECTRA (Clark et al., 2020), which contains a generator and a discriminator. Given a sentence of length T, $x = [x_{(1)}, x_{(2)}, ..., x_{(T)}]$, we first apply a random mask $m = [m_{(1)}, m_{(2)}, ..., m_{(T)}], m_{(t)} \in [0, 1]$ on x to obtain $x' = m \cdot x$. We use another pretrained MLM as the generator G to perform masked language modeling to recover randomly masked tokens in x' to obtain the edited sentence $x'' = G(x')$. Then, we use a discriminator D to perform the Replaced Token Detection (RTD) task. For each token in the sentence, the model needs to predict whether it has been replaced or not. The cross-entropy loss for a single sentence x is:

$$\begin{aligned}\mathcal{L}_{\mathrm{RTD}}^{x} = \sum_{t=1}^{T}\Big(&-1\left(x''_{(t)} = x_{(t)}\right)\log D\left(x'', \mathbf{h}, t\right)\\ &-1\left(x''_{(t)} \neq x_{(t)}\right)\log\left(1 - D\left(x'', \mathbf{h}, t\right)\right)\Big)\end{aligned}$$

And the training objective for a batch is $\mathcal{L}_{\mathrm{RTD}} = \sum_{i=1}^{N} \mathcal{L}_{\mathrm{RTD}}^{x_i}$. Finally we optimize these two losses together with a weighting coefficient λ:

$$\mathcal{L} = \mathcal{L}_{\text{contrast}} + \lambda \cdot \mathcal{L}_{\mathrm{RTD}}$$

The difference between our model and ELECTRA is that our discriminator D is *conditional*, so it can use the information of x compressed in a fixed-dimension vector $\mathbf{h} = f(x)$. The gradient of D can be backward-propagated into f through $\mathbf{h}$. By doing so, f will be encouraged to make $\mathbf{h}$ informative enough to cover the full meaning of x, so that D can distinguish the tiny difference between x and x''. This approach essentially makes the conditional discriminator perform a "diff operation", hence the name DiffCSE.

When we train our DiffCSE model, we fix the generator G, and only the sentence encoder f and the discriminator D are optimized. After training, we discard D and only use f (which remains fixed) to extract sentence embeddings to evaluate on the downstream tasks.

2.4 Experiments

2.4.1 Setup

In our experiment, we follow the setting of unsupervised SimCSE (Gao et al., 2021) and build our model based on their PyTorch implementation.[1] We also use the checkpoints of BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019b) as the initialization of our sentence encoder f. We add an MLP layer with Batch Normalization (Ioffe and Szegedy, 2015) (BatchNorm) on top of the `[CLS]` representation as the sentence embedding. We will compare the model with/without BatchNorm in section 2.5. For the discriminator D, we use the same model as the sentence encoder f (BERT/RoBERTa). For the generator G, we use the smaller DistilBERT and DistilRoBERTa (Sanh et al., 2019) for efficiency. Note that the generator is fixed during training unlike the ELECTRA paper (Clark et al., 2020). We will compare the results of using different size model for the generator in section 2.5. More training details are shown in Appendix A.1.

2.4.2 Data

For unsupervised pretraining, we use the same 10^6 randomly sampled sentences from English Wikipedia that are provided by the source code of SimCSE.[1] We evaluate our model on 7 semantic textual similarity (STS) and 7 transfer tasks in SentEval.[2] STS tasks includes STS 2012–2016 (Agirre et al., 2016), STS Benchmark (Cer et al., 2017) and SICK-Relatedness (Marelli et al., 2014). All the STS experiments are fully unsupervised, which means no STS training datasets are used and all embeddings are fixed once they are trained. The transfer tasks are various sentence classification tasks,

[1]`https://github.com/princeton-nlp/SimCSE`
[2]`https://github.com/facebookresearch/SentEval`

including MR (Pang and Lee, 2005), CR (Hu and Liu, 2004), SUBJ (Pang and Lee, 2004), MPQA (Wiebe et al., 2005), SST-2 (Socher et al., 2013), TREC (Voorhees and Tice, 2000) and MRPC (Dolan and Brockett, 2005). In these transfer tasks, we will use a logistic regression classifier trained on top of the frozen sentence embeddings, following the standard setup (Conneau and Kiela, 2018).

2.4.3 Results

Baselines We compare our model with many strong unsupervised baselines including SimCSE (Gao et al., 2021), IS-BERT (Zhang et al., 2020), CMLM (Yang et al., 2020), DeCLUTR (Giorgi et al., 2020), CT-BERT (Carlsson et al., 2021), SG-OPT (Kim et al., 2021) and some post-processing methods like BERT-flow (Li et al., 2020) and BERT-whitening (Su et al., 2021) along with some naive baselines like averaged GloVe embeddings (Pennington et al., 2014) and averaged first and last layer BERT embeddings.

Model	**STS12**	**STS13**	**STS14**	**STS15**	**STS16**	**STS-B**	**SICK-R**	**Avg.**
GloVe embeddings (avg.)♣	55.14	70.66	59.73	68.25	63.66	58.02	53.76	61.32
BERT$_{\text{base}}$ (first-last avg.)♢	39.70	59.38	49.67	66.03	66.19	53.87	62.06	56.70
BERT$_{\text{base}}$-flow♢	58.40	67.10	60.85	75.16	71.22	68.66	64.47	66.55
BERT$_{\text{base}}$-whitening♢	57.83	66.90	60.90	75.08	71.31	68.24	63.73	66.28
IS-BERT$_{\text{base}}$ ♡	56.77	69.24	61.21	75.23	70.16	69.21	64.25	66.58
CMLM-BERT$_{\text{base}}$ ♠ (1TB data)	58.20	61.07	61.67	73.32	74.88	76.60	64.80	67.22
CT-BERT$_{\text{base}}$ ♢	61.63	76.80	68.47	77.50	76.48	74.31	69.19	72.05
SG-OPT-BERT$_{\text{base}}$ †	66.84	80.13	71.23	81.56	77.17	77.23	68.16	74.62
SimCSE-BERT$_{\text{base}}$ ♢	68.40	82.41	74.38	80.91	78.56	76.85	**72.23**	76.25
∗ SimCSE-BERT$_{\text{base}}$(reproduce)	70.82	82.24	73.25	81.38	77.06	77.24	71.16	76.16
∗ DiffCSE-BERT$_{\text{base}}$	**72.28**	**84.43**	**76.47**	**83.90**	**80.54**	**80.59**	71.23	**78.49**
RoBERTa$_{\text{base}}$ (first-last avg.)♢	40.88	58.74	49.07	65.63	61.48	58.55	61.63	56.57
RoBERTa$_{\text{base}}$-whitening♢	46.99	63.24	57.23	71.36	68.99	61.36	62.91	61.73
DeCLUTR-RoBERTa$_{\text{base}}$ ♢	52.41	75.19	65.52	77.12	78.63	72.41	68.62	69.99
SimCSE-RoBERTa$_{\text{base}}$ ♢	**70.16**	81.77	73.24	81.36	80.65	80.22	68.56	76.57
∗ SimCSE-RoBERTa$_{\text{base}}$(reproduce)	68.60	81.36	73.16	81.61	80.76	80.58	68.83	76.41
∗ DiffCSE-RoBERTa$_{\text{base}}$	70.05	**83.43**	**75.49**	**82.81**	**82.12**	**82.38**	**71.19**	**78.21**

Table 2.1: The performance on STS tasks (Spearman's correlation) for different sentence embedding models. ♣: results from Reimers and Gurevych (2019); ♡: results from Zhang et al. (2020); ♢: results from Gao et al. (2021); ♠: results from Yang et al. (2020); †: results from Kim et al. (2021); ∗: results from our experiments.

Semantic Textual Similarity (STS) We show the results of STS tasks in Table 2.1 including BERT$_{\text{base}}$ (upper part) and RoBERTa$_{\text{base}}$ (lower part). We also reproduce the previous state-of-the-art SimCSE (Gao et al., 2021). DiffCSE-BERT$_{\text{base}}$ can significantly outperform SimCSE-BERT$_{\text{base}}$ and raise the averaged Spearman's correlation from 76.25% to 78.49%. For the RoBERTa model, DiffCSE-RoBERTa$_{\text{base}}$ can also improve upon SimCSE-RoBERTa$_{\text{base}}$ from 76.57% to 77.80%.

Model	**MR**	**CR**	**SUBJ**	**MPQA**	**SST**	**TREC**	**MRPC**	**Avg.**
GloVe embeddings (avg.)$^\clubsuit$	77.25	78.30	91.17	87.85	80.18	83.00	72.87	81.52
Skip-thought$^\heartsuit$	76.50	80.10	93.60	87.10	82.00	92.20	73.00	83.50
Avg. BERT embeddings$^\clubsuit$	78.66	86.25	94.37	88.66	84.40	**92.80**	69.54	84.94
BERT-[CLS]embedding$^\clubsuit$	78.68	84.85	94.21	88.23	84.13	91.40	71.13	84.66
IS-BERT$_{\text{base}}$ $^\heartsuit$	81.09	87.18	94.96	88.75	85.96	88.64	74.24	85.83
SimCSE-BERT$_{\text{base}}$ $^\diamondsuit$	81.18	86.46	94.45	88.88	85.50	89.80	74.43	85.81
w/ MLM	82.92	87.23	95.71	88.73	86.81	87.01	78.07	86.64
$*$ DiffCSE-BERT$_{\text{base}}$	**82.69**	**87.23**	**95.23**	**89.28**	**86.60**	90.40	**76.58**	**86.86**
CMLM-BERT$_{\text{base}}$(1TB data)	83.60	89.90	96.20	89.30	88.50	91.00	69.70	86.89
SimCSE-RoBERTa$_{\text{base}}$ $^\diamondsuit$	81.04	87.74	93.28	86.94	86.60	84.60	73.68	84.84
w/ MLM	**83.37**	87.76	**95.05**	87.16	**89.02**	**90.80**	75.13	86.90
$*$ DiffCSE-RoBERTa$_{\text{base}}$	82.82	**88.61**	94.32	**87.71**	88.63	90.40	**76.81**	**87.04**

Table 2.2: Transfer task results of different sentence embedding models (measured as accuracy). $\clubsuit$: results from Reimers and Gurevych (2019); $\heartsuit$: results from Zhang et al. (2020); $\diamondsuit$: results from Gao et al. (2021).

Transfer Tasks We show the results of transfer tasks in Table 2.2. Compared with SimCSE-BERT$_{\text{base}}$, DiffCSE-BERT$_{\text{base}}$ can improve the averaged scores from 85.56% to 86.86%. When applying it to the RoBERTa model, DiffCSE-RoBERTa$_{\text{base}}$ also improves upon SimCSE-RoBERTa$_{\text{base}}$ from 84.84% to 87.04%. Note that the CMLM-BERT$_{\text{base}}$ (Yang et al., 2020) can achieve even better performance than DiffCSE. However, they use 1TB of the training data from Common Crawl dumps while our model only use 115MB of the Wikipedia data for pretraining. We put their scores in Table 2.2 for reference. In SimCSE, the authors propose to use MLM as an auxiliary task for the sentence encoder to further boost the performance of transfer tasks. Compared with the results of SimCSE with MLM, DiffCSE still can have a little improvement around 0.2%.

2.5 Ablation Studies

In the following sections, we perform an extensive series of ablation studies that support our model design. We use $\text{BERT}_{\text{base}}$ model to evaluate on the development set of STS-B and transfer tasks.

2.5.1 Removing Contrastive Loss

In our model, both the contrastive loss and the RTD loss are crucial because they maintain what should be sensitive and what should be insensitive respectively. If we remove the RTD loss, the model becomes a SimCSE model; if we remove the contrastive loss, the performance of STS-B drops significantly by 30%, while the average score of transfer tasks also drops by 2% (see Table 2.3). This result shows that it is important to have insensitive and sensitive attributes that exist together in the representation space.

2.5.2 Next Sentence vs. Same Sentence

Some methods for unsupervised sentence embeddings like Quick-Thoughts (Logeswaran and Lee, 2018) and CMLM (Yang et al., 2020) predict the next sentence as the training objective. We also experiment with a variant of DiffCSE by conditioning the ELECTRA loss based on the next sentence. Note that this kind of model is not doing a "diff operation" between two similar sentences, and is not an instance of equivariant contrastive learning. As shown in Table 2.3 (use next sent. for x'), the score of STS-B decreases significantly compared to DiffCSE while transfer performance remains similar. We also tried using the same sentence and the next sentence at the same time for conditioning the ELECTRA objective (use same+next sent. for x'), and did not observe improvements.

	STS-B	**Avg. transfer**
SimCSE	81.47	83.91
DiffCSE	**84.56**	**85.95**
w/o contrastive loss	54.48	83.46
use next sent. for x'	82.91	85.83
use same+next sent. for x'	83.41	85.82
Conditional MLM		
for same sent.	83.08	84.43
for next sent.	75.82	85.68
for same+next sent.	82.88	84.82
Conditional Corrective LM	79.79	85.30

Table 2.3: Development set results of STS-B and transfer tasks for DiffCSE model variants, where we vary the objective and the use of same or next sentence.

Augmentation	**STS-B**	**Avg. transfer**
MLM 15%	**84.48**	85.95
randomly insert 15%	82.20	85.96
randomly delete 15%	82.59	**85.97**
combining all	82.80	85.92

Table 2.4: Development set results of STS-B and transfer tasks with different augmentation methods for learning equivariance.

2.5.3 Other Conditional Pretraining Tasks

Instead of a conditional binary difference prediction loss, we can also consider other conditional pretraining tasks such as a conditional MLM objective proposed by (Yang et al., 2020), or corrective language modeling,[3] proposed by COCO-LM (Meng et al., 2021). We experiment with these objectives instead of the difference prediction objective in Table 2.3. We observe that conditional MLM on the same sentence does not improve the performance either on STS-B or transfer tasks compared with DiffCSE. Conditional MLM on the next sentence performs even worse for STS-B, but slightly better than using the same sentence on transfer tasks. Using both the same and the next sentence also does not improve the performance compared with DiffCSE. For the

[3]This task is similar to ELECTRA. However, instead of a binary classifier for replaced token detection, corrective LM uses a vocabulary-size classifier with the copy mechanism to recover the replaced tokens.

corrective LM objective, the performance of STS-B decreases significantly compared with DiffCSE.

	STS-B	**Avg. transfer**
DiffCSE		
w/ BatchNorm	**84.56**	**85.95**
w/o BatchNorm	83.23	85.24
SimCSE		
w/ BatchNorm	82.22	85.66
w/o BatchNorm	81.47	83.91

Table 2.5: Development set results of STS-B and transfer tasks for DiffCSE and SimCSE with and without BatchNorm.

2.5.4 Augmentation Methods: Insert/Delete/Replace

In DiffCSE, we use MLM token replacement as the equivariant augmentation. It is possible to use other methods like random insertion or deletion instead of replacement.[4] For insertion, we choose to randomly insert mask tokens to the sentence, and then use a generator to convert mask tokens into real tokens. The number of inserted masked tokens is 15% of the sentence length. The task is to predict whether a token is an inserted token or the original token. For deletion, we randomly delete 15% tokens in the sentence, and the task is to predict for each token whether a token preceding it has been deleted or not. The results are shown in Table 2.4. We can see that using either insertion or deletion achieves a slightly worse STS-B performance than using MLM replacement. For transfer tasks, their results are similar. Finally, we find that combining all three augmentations in the training process does not improve the MLM replacement strategy.

2.5.5 Pooler Choice

In SimCSE, the authors use the pooler in BERT's original implementation (one linear layer with tanh activation function) as the final layer to extract features for computing

[4]Edit distance operators include *insert, delete* and *replace.*

contrastive loss. In our implementation (see details in Appendix A.1), we find that it is better to use a two-layer pooler with Batch Normalization (BatchNorm) (Ioffe and Szegedy, 2015), which is commonly used in contrastive learning framework in computer vision (Chen et al., 2020; Grill et al., 2020; Chen and He, 2021; Hua et al., 2021). We show the ablation results in Table 2.5. We can observe that adding BatchNorm is beneficial for either DiffCSE or SimCSE to get better performance on STS-B and transfer tasks.

2.5.6 Size of the Generator

In our DiffCSE model, the generator can be in different model size from BERT$_{\texttt{large}}$, BERT$_{\texttt{base}}$ (Devlin et al., 2019), DistilBERT$_{\texttt{base}}$ (Sanh et al., 2019), BERT$_{\texttt{medium}}$, BERT$_{\texttt{small}}$, BERT$_{\texttt{mini}}$, BERT$_{\texttt{tiny}}$ (Turc et al., 2019). Their exact sizes are shown in Table 2.6 (L: number of layers, H: hidden dimension). Notice that although DistilBERT$_{\texttt{base}}$ has only half the number of layers of BERT, it can retain 97% of BERT's performance due to knowledge distillation.

We show our results in Table 2.6, we can see the performance of transfer tasks does not change much with different generators. However, the score of STS-B decreases as we switch from BERT-medium to BERT-tiny. This finding is not the same as ELECTRA, which works best with generators 1/4-1/2 the size of the discriminator. Because our discriminator is conditional on sentence vectors, it will be easier for the discriminator to perform the RTD task. As a result, using stronger generators (BERT$_{\texttt{base}}$, DistilBERT$_{\texttt{base}}$) to increase the difficulty of RTD would help the discriminator learn better. However, when using a large model like BERT$_{\texttt{large}}$, it may be a too-challenging task for the discriminator. In our experiment, using DistilBERT$_{\texttt{base}}$, which has the ability close to but slightly worse than BERT$_{\texttt{base}}$, gives us the best performance.

	STS-B	**Avg. transfer**
SimCSE	81.47	83.91
DiffCSE w/ generator:		
BERT$_{\texttt{large}}$ (L=24, H=1024)	82.93	85.88
BERT$_{\texttt{base}}$ (L=12, H=768)	83.63	85.85
DistilBERT$_{\texttt{base}}$ (L=6, H=768)	**84.56**	**85.95**
BERT$_{\texttt{medium}}$ (L=8, H=512)	82.25	85.80
BERT$_{\texttt{small}}$ (L=4, H=512)	82.64	85.66
BERT$_{\texttt{mini}}$ (L=4, H=256)	82.12	85.90
BERT$_{\texttt{tiny}}$ (L=2, H=128)	81.40	85.23

Table 2.6: Development set results of STS-B and transfer tasks with different generators.

Ratio	*15%*	*20%*	*25%*	*30%*	*40%*	*50%*
STS-B	84.48	84.04	84.49	**84.56**	84.48	83.91

Table 2.7: Development set results of STS-B under different masking ratio for augmentations.

2.5.7 Masking Ratio

In our conditional ELECTRA task, we can mask the original sentence in different ratios for the generator to produce MLM-based augmentations. A higher masking ratio will make more perturbations to the sentence. Our empirical result in Table 2.7 shows that the difference between difference masking ratios is small (in 15%-40%), and a masking ratio of around 30% can give us the best performance.

2.5.8 Coefficient λ

In Section 2.3, we use the λ coefficient to weight the ELECTRA loss and then add it with contrastive loss. Because the contrastive learning objective is a relatively easier task, the scale of contrastive loss will be 100 to 1000 smaller than ELECTRA loss. As a result, we need a smaller λ to balance these two loss terms. In the Table 2.8 we show the STS-B result under different λ values. Note that when λ goes to zero, the model becomes a SimCSE model. We find that using $\lambda = 0.005$ can give us the best

λ	*0*	*0.0001*	*0.0005*	*0.001*
STS-B	82.22	83.90	84.40	84.24
λ	*0.005*	*0.01*	*0.05*	*0.1*
STS-B	**84.56**	83.44	84.11	83.66

Table 2.8: Development set results of STS-B under different λ.

performance.

2.6 Analysis

2.6.1 Qualitative Study

A very common application for sentence embeddings is the retrieval task. Here we show some retrieval examples to qualitatively explain why DiffCSE can perform better than SimCSE. In this study, we use the 2,758 sentences from STS-B testing set as the corpus, and then use sentence query to retrieve the nearest neighbors in the sentence embedding space by computing cosine similarities. We show the retrieved top-3 examples in Table 2.9. The first query sentence is "you can do it, too." The SimCSE model retrieves a very similar sentence but has a slightly different meaning ("you can use it, too.") as the rank-1 answer. In contrast, DiffCSE can distinguish the tiny difference, so it retrieves the ground truth answer as the rank-1 answer. The second query sentence is "this is not a problem". SimCSE retrieves a sentence with opposite meaning but very similar wording, while DiffCSE can retrieve the correct answer with less similar wording. We also provide a third example where both SimCSE and DiffCSE fail to retrieve the correct answer for a query sentence using double negation.

2.6.2 Retrieval Task

Besides the qualitative study, we also show the quantitative result of the retrieval task. Here we also use all the 2,758 sentences in the testing set of STS-B as the

SimCSE-BERT$_{\text{base}}$	**DiffCSE-BERT$_{\text{base}}$**
Query: you can do it, too.	
1) you can use it, too. 2) can you do it? 3) yes, you can do it.	1) yes, you can do it. 2) you can use it, too. 3) can you do it?
Query: this is not a problem.	
1) this is a big problem. 2) you have a problem. 3) i don 't see why that should be a problem.	1) i don 't see why this could be a problem. 2) i don 't see why that should be a problem. 3) this is a big problem.
Query: i think that is not a bad idea.	
1) i do not think it's a good idea. 2) it's not a good idea . 3) it is not a good idea .	1) i do not think it's a good idea . 2) it is not a good idea. 3) but it is not a good idea.

Table 2.9: Retrieved top-3 examples by SimCSE and DiffCSE from STS-B test set.

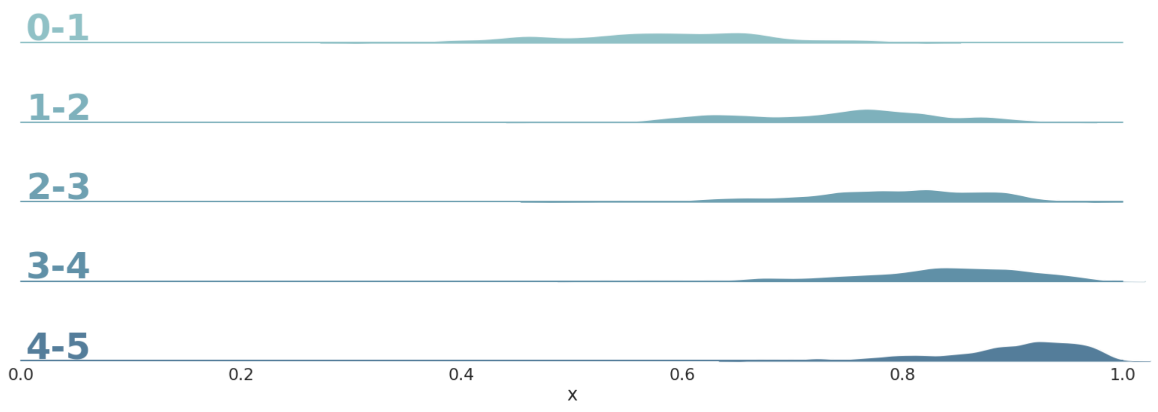
Model/Recall	**@1**	**@5**	**@10**
SimCSE-BERT$_{\text{base}}$	77.84	92.78	95.88
DiffCSE-BERT$_{\text{base}}$	**78.87**	**95.36**	**97.42**

Table 2.10: The retrieval results for SimCSE and DiffCSE.

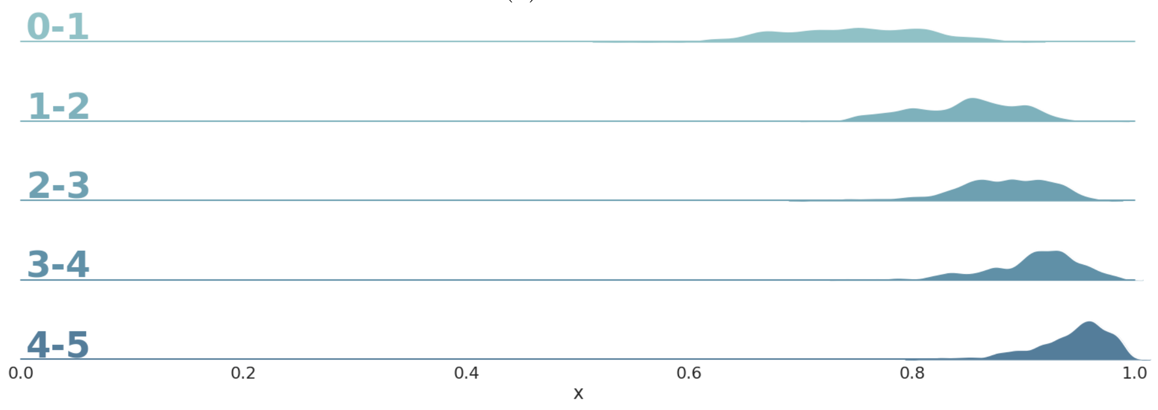
corpus. There are 97 positive pairs in this corpus (with 5 out of 5 semantic similarity scores from human annotation). For each positive pair, we use one sentence to retrieve the other one, and see whether the other sentence is in the top-1/5/10 ranking. The recall@1/5/10 of the retrieval task are shown in Table 2.10. We can observe that DiffCSE can outperform SimCSE for recall@1/5/10, showing the effectiveness of using DiffCSE for the retrieval task.

2.6.3 Distribution of Sentence Embeddings

To look into the representation space of DiffCSE, we plot the cosine similarity distribution of sentence pairs from STS-B test set for both SimCSE and DiffCSE in Figure 2-2. We observe that both SimCSE and DiffCSE can assign cosine similarities consistent with human ratings. However, we also observe that under the same human rating, DiffCSE assigns slightly higher cosine similarities compared with SimCSE. This phenomenon may be caused by the fact that ELECTRA and other



(a) SimCSE



(b) DiffCSE

Figure 2-2: The distribution of cosine similarities from SimCSE/DiffCSE for STS-B test set. Along the y-axis are 5 groups of data splits based on human ratings. The x-axis is the cosine similarity.

Transformer-based pretrained LMs have the problem of squeezing the representation space, as mentioned by (Meng et al., 2021). As we use the sentence embeddings as the input of ELECTRA to perform conditional ELECTRA training, the sentence embedding will be inevitably squeezed to fit the input distribution of ELECTRA.

2.6.4 Uniformity and Alignment

Wang and Isola (2020) propose to use two properties, *alignment* and *uniformity*, to measure the quality of representations. Given a distribution of positive pairs p_{pos} and the distribution of the whole dataset p_{data}, *alignment* computes the expected distance

Model	Alignment	Uniformity	STS
Avg. BERT$_{\text{base}}$	0.172	-1.468	56.70
SimCSE-BERT$_{\text{base}}$	0.177	**-2.313**	76.16
DiffCSE-BERT$_{\text{base}}$	**0.097**	-1.438	**78.49**

Table 2.11: *Alignment* and *Uniformity* (Wang and Isola, 2020) measured on STS-B test set for SimCSE and DiffCSE. The smaller the number is better. We also show the averaged STS score in the right-most column.

between normalized embeddings of the paired sentences:

$$\ell_{\text{align}} \triangleq \underset{(x,x^+)\sim p_{\text{pos}}}{\mathbb{E}} \left\|f(x) - f\left(x^+\right)\right\|^2 .$$

Uniformity measures how well the embeddings are uniformly distributed in the representation space:

$$\ell_{\text{uniform}} \triangleq \log \underset{x,y \overset{i.i.d.}{\sim} p_{\text{data}}}{\mathbb{E}} e^{-2\|f(x)-f(y)\|^2}.$$

The smaller the values of both alignment and uniformity, the better the quality of the representation space is indicated.

We follow the above definition of uniformity and alignment to measure the quality of representation space for DiffCSE and SimCSE in Table 2.11. Compared to averaged BERT embeddings, SimCSE has similar alignment (0.177 v.s. 0.172) but better uniformity (-2.313). In contrast, DiffCSE has similar uniformity as Avg. BERT (-1.438 v.s. -1.468) but much better alignment (0.097). It indicates that SimCSE and DiffCSE are optimizing the representation space in two different directions. And the improvement of DiffCSE may come from its better alignment.

2.7 Extension to Dense Passage Retrieval

In addition to evaluating DiffCSE on semantic textual similarity and transfer tasks, we also explore applying it to dense passage retrieval (DPR) (Karpukhin et al., 2020) for open-domain question answering, which is the ultimate goal of our research. DPR aims to retrieve the most relevant passage from a large corpus for a given question.

Table 2.12: Comparison of DPR and DPR with DiffCSE on different top-k accuracies.

Model	Top-1	Top-5	Top-10	Top-20	Top-50	Top-100
DPR	40.0	64.2	71.5	76.4	81.9	84.6
DPR+DiffCSE	42.2	66.2	73.0	78.0	82.7	85.5

We evaluate our method on the Natural Questions (NQ) (Kwiatkowski et al., 2019), which contains real user queries from Google search with annotated answers from Wikipedia passages.

To apply DiffCSE to DPR, we fine-tune the query and passage encoders with the DiffCSE objective on NQ. We follow the implementation of dpr-scale [5] and leverage the shared encoder architecture, which uses the same weights for the query/passage encoder. The shared encoder is initialized from RoBERTa-base (Liu et al., 2019b) and trained with both the contrastive and the RTD losses, which requires a generator (DistilRoBERTa) to produce MLM-augmentation on the queries and passages, and a discriminator (RoBERTa-base) for the RTD task.

We compare fine-tuning the standard DPR model versus a DPR model fine-tuned with the DiffCSE objective. As shown in Table 2.12, adding the DiffCSE objective improves the retrieval accuracy across different top-k accuracies. The Top-20 accuracy increases significantly from 76.4% to 78.0% when enhancing DPR with DiffCSE. The consistent gains demonstrate the benefits of making the query/passage encoder representation aware of, and sensitive to, word replacements through our proposed DiffCSE pipeline.

2.8 Using Augmentations as Positive/Negative Examples

In Section 2.5, we try to use different augmentations (e.g. insertion, deletion, replacement) for learning equivariance. In Table 2.13 we provide the results of using these augmentations as additional positive or negative examples along with the SimCSE

[5] `https://github.com/facebookresearch/dpr-scale`

training paradigm. We can observe that using these augmentations as additional positives only decreases the performance. The only method that can improve the performance a little bit is to use MLM with 15% replaced examples as additional negative examples. Overall, none of these results can perform better than our proposed method, e.g. using these augmentations to learn equivariance.

Method	**STS-B**	**Avg. transfer**
SimCSE	81.47	83.91
+ Additional positives		
MLM 15%	73.59	83.33
random insert 15%	80.39	83.92
random delete 15%	78.58	81.80
+ Additional negatives		
MLM 15%	83.02	84.49
random insert 15%	55.65	79.86
random delete 15%	55.13	82.56
+ Equivariance (Ours)		
MLM 15%	**84.48**	85.95
randomly insert 15%	82.20	85.96
randomly delete 15%	82.59	**85.97**

Table 2.13: Development set results of STS-B and transfer tasks for using three types of augmentations (replace, insert, delete) in different ways.

2.9 Chapter Summary

In this chapter, we present DiffCSE, a new unsupervised sentence embedding framework that is aware of, but not invariant to, MLM-based word replacement. Empirical results on semantic textual similarity tasks and transfer tasks both show the effectiveness of DiffCSE compared to current state-of-the-art sentence embedding methods. We also conduct extensive ablation studies to demonstrate the different modeling choices in DiffCSE. Qualitative study and the retrieval results also show that DiffCSE can produce a better embedding space for sentence retrieval. We further apply DiffCSE to dense passage retrieval for open-domain question answering. The significantly improved results show that DiffCSE is useful not only in unsupervised sentence embedding but also in supervised passage retrieval tasks. We believe that our work can provide researchers in the NLP community a new way to utilize augmentations for natural language and thus produce better sentence embeddings.

Chapter 3

Expand, Rerank, and Retrieve: Query Reranking for Open-Domain Question Answering

3.1 Introduction

Open-domain question answering (QA) (Chen and Yih, 2020), the task of answering a wide range of factoid questions from diverse domains, is often used to benchmark machine intelligence (Kwiatkowski et al., 2019), and has a direct application to fulfilling a user's information needs (Voorhees et al., 1999). To provide faithful answers with provenance, and to easily update knowledge from new documents, passage *retrieval*, which finds relevant text chunks to given questions, is critical to the success of a QA system. Retrieval in early open-domain QA systems (Chen et al., 2017) was typically based on term-matching methods, such as BM25 (Robertson et al., 2009) or TF-IDF (Salton et al., 1975). Such methods are sometimes called *sparse* retrievers, as they represent documents and queries with high-dimensional sparse vectors, and can efficiently match keywords with an inverted index and find relevant passages. Despite their simplicity, sparse retrievers are limited by their inability to perform semantic matching for relevant passages that have low lexical overlap with the query. Lately, dense retrievers (Karpukhin et al., 2020), which represent documents and queries with dense, continuous semantic vectors, have been adopted by modern QA systems. Dense retrievers usually outperform their sparse counterparts, especially when there exists enough in-domain training data.

However, dense retrievers have certain weaknesses compared to sparse ones, including: 1) being computationally expensive in training and inference, 2) potential information loss when compressing long passages into fixed-dimensional vectors (Luan et al., 2021b), which makes it hard to match rare entities exactly (Sciavolino et al., 2021), and 3) difficulty in generalizing to new domains (Reddy et al., 2021). As a result, dense retrievers and sparse ones are usually complementary to each other and can be combined to boost performance. Recent studies on query expansion, such as GAR (Mao et al., 2021b), have attempted to improve sparse retrievers by adding relevant contexts to the query using pre-trained language models (PLMs), which has been shown effective in closing the gap between sparse and dense retrievers.

In this work, we introduce a novel query **E**xpansion **A**nd **R**eranking approach, EAR, which enhances generative query expansion with **query reranking**. EAR first generates a diverse set of expanded queries with query expansion models, and then trains a query reranker to estimate the *quality* of these queries by directly predicting the rank order of a gold passage, when issuing these queries to a given retriever, such as BM25. At inference time, EAR selects the most promising query expansion as predicted by the query reranker and issues it to the same retriever to find relevant documents. EAR is motivated by a simple observation—while the greedy decoding output of a query expansion model, such as GAR, could be suboptimal, some randomly sampled query expansions achieve superior performance with BM25 (see Section 3.2.2). EAR better connects the query expansion model and the underlying retrieval method, and thus can select a more suitable query.

We empirically evaluated EAR in both *in-domain* and *cross-domain* settings. Our *in-domain* experimental results on Natural Questions and TriviaQA show that EAR significantly improves the top-5/20 accuracy by 3-8 points. For the *cross-domain* setting, while the query expansion model suffers from substantial performance degra-

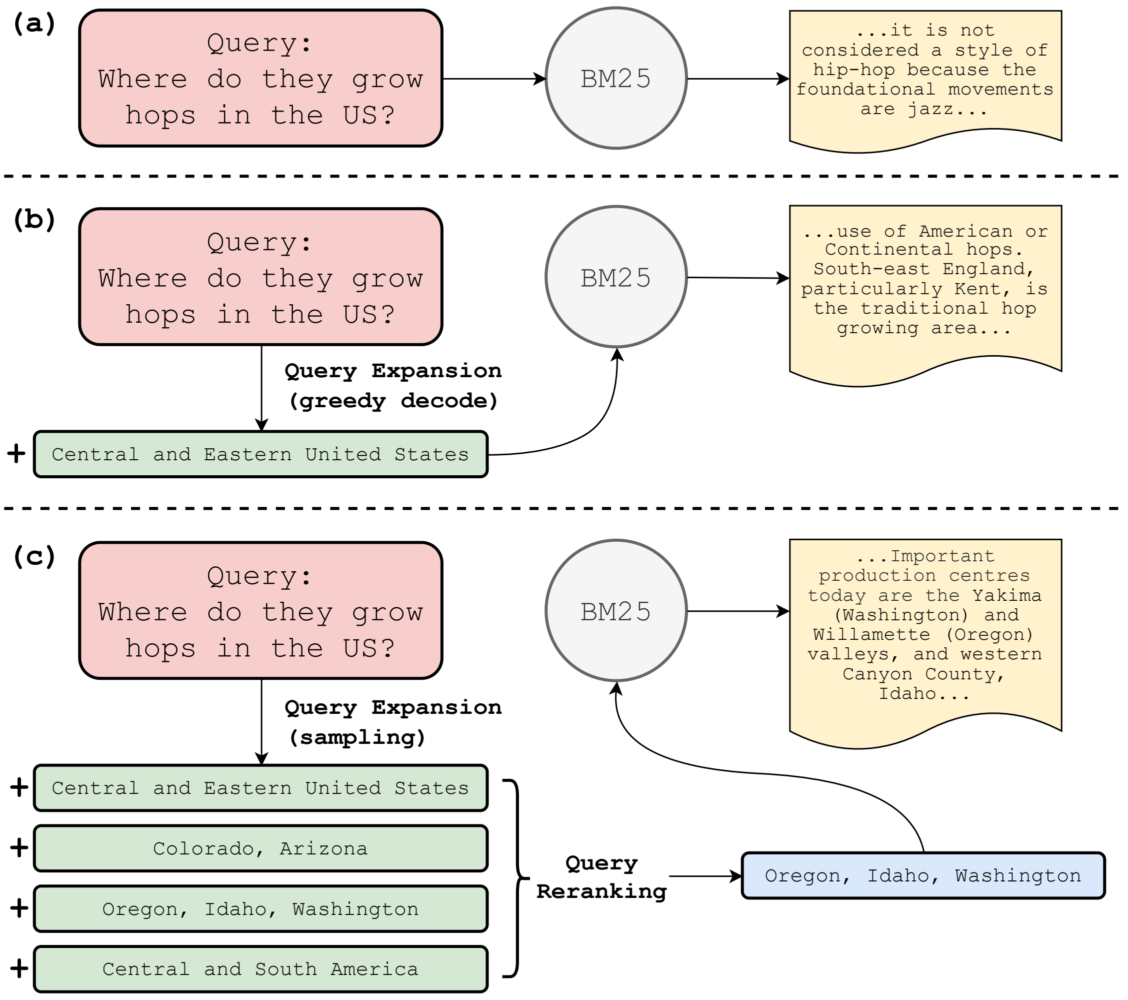

(a)
Query: Where do they grow hops in the US?
BM25
...it is not considered a style of hip-hop because the foundational movements are jazz...
(b)
Query: Where do they grow hops in the US?
Query Expansion (greedy decode)
+ Central and Eastern United States
BM25
...use of American or Continental hops. South-east England, particularly Kent, is the traditional hop growing area...
(c)
Query: Where do they grow hops in the US?
Query Expansion (sampling)
+ Central and Eastern United States
+ Colorado, Arizona
+ Oregon, Idaho, Washington
+ Central and South America
Query Reranking
Oregon, Idaho, Washington
BM25
...Important production centres today are the Yakima (Washington) and Willamette (Oregon) valleys, and western Canyon County, Idaho...

Figure 3-1: (a) Standard BM25 pipeline (b) Generation-Augmented Retrieval (Gar) with BM25 (c) Our proposed Expand and Rerank (Ear) pipeline.

dation when applied to new domains, Ear seems to be more domain-agnostic, and can still find useful queries from a diverse set of query expansions, which leads to a significant improvement over Gar and DPR by 5-10 points for top-5/20 accuracy.

Our contributions can be summarized as follows:

- We proposed Ear to select the best query from a diverse set of query expansions, by predicting which query can achieve the best BM25 result. This improves the connection of query expansion models and BM25, resulting in enhanced performance that surpasses DPR.
- Ear not only performs well on in-domain data, but also shows strong generalization abilities on out-of-domain data, outperforming Gar and DPR by a

Model	**Top-1**	**Top-5**	**Top-20**	**Top-100**
1) BM25	22.1	43.8	62.9	78.3
2) DPR	43.0	66.4	78.5	85.0
3) GAR (greedy)	37.0	60.8	73.9	84.7
4) GAR (beam=10)	38.6	61.6	75.2	84.8
5) GAR *best query*	68.8	81.9	88.1	92.0
6) GAR *concat*	39.5	60.3	72.7	83.6

Table 3.1: The potential top-k retrieval accuracy that can be achieved by query reranking on Natural Questions. GAR uses greedy-decoded/beam-searched queries; GAR *best query* randomly samples 50 queries and picks the oracle one with the best retrieval scores; GAR *concat* simply concatenates all 50 queries as a single long query.

large margin.

- End-to-end evaluation with a generative reader demonstrates the benefits of EAR in improving the exact match score.
- Lastly, we show that the improvements provided by EAR and passage reranking are complementary, allowing for effective aggregation of performance gains from both methods.

3.2 Background

3.2.1 Generation-Augmented Retrieval

Generation-Augmented Retrieval (GAR) (Mao et al., 2021b) aims to enhance sparse retrievers by query expansion with text generation from PLMs. Given the initial query, GAR generates relevant contexts including *the answer, answer sentence*, and *title of answer passages*, and then concatenates them to the initial query before performing retrieval with BM25. GAR achieves decent performance close to that of DPR while using the lightweight BM25 retriever. However, a limitation is that GAR is not aware of the existence of BM25, potentially generating suboptimal queries for retrieval. Additionally, GAR is only trained on in-domain data, limiting their ability to transfer to out-of-domain data.

3.2.2 Preliminary Experiments

Let us first take a look at some preliminary experimental results to better understand the motivation of our work. In Table 3.1, we present the top-k retrieval results on Natural Questions (Kwiatkowski et al., 2019) for BM25, DPR, and GAR (greedy/beam search) in rows 1-4. To investigate the potential of GAR, we randomly sampled 50 query expansions from GAR, ran BM25 separately for these queries, and chose the best one by looking at the BM25 results, which requires ground truth labels. The resulting scores are shown in row 5 (GAR *best query*).

From the results, we see that GAR *best query* can lead to a significant improvement of up to 20 points compared to DPR. Since we do not have access to labels for selecting the best query in reality, a naive solution is to concatenate all 50 query expansions together as a single, long query, which will definitely include high-quality expansions if they exist. However, as shown in row 6, the performance of GAR *concat* is even worse than that of GAR alone with greedy decoding outputs. This indicates that the single long query may include too much distracting information, negatively impacting the performance of the BM25 retriever.

From these preliminary results, we reach two conclusions: 1) GAR does have the ability to generate very useful query expansions; 2) however, the useful query expansions may not always be included in the GAR greedy decoding outputs. It is non-trivial to extract these useful query expansions from GAR. Motivated by these findings, we leverage a query reranker to estimate if a query will be beneficial to BM25 retrieval results, so as to unlock the potential of GAR and sparse retrievers.

3.3 Proposed Method

We illustrate our proposed method, EAR, in Figure 3-1, along with a comparison with the BM25 and GAR pipelines. Given the original query q, EAR first generates a set of query expansions $E = \{e_1, e_2, ..., e_n\}$ using random sampling. We believe that among these n queries, some may achieve very good retrieval performance. Thus, we train a reranker model $\mathcal{M}$ to re-score all the queries. Here we propose two kinds of rerankers: 1) retrieval-independent (RI) reranker, and 2) retrieval-dependent (RD) reranker. Both rerankers can estimate the quality of a query expansion without using information from answer annotations.

3.3.1 Retrieval-Independent (RI) Reranker

The inputs to the RI reranker are quite simple: (q, e_i), which consists of the original query q and one of the query expansions e_i. When training this reranker, we first obtain the minimum answer passage ranking among all retrieved passages for each query, when issued to a BM25 retriever. We denote this minimum answer passage ranking as $R = \{r_1, r_2, ..., r_n\}$, which corresponds to each of the expanded queries $\{(q, e_1), (q, e_2), ..., (q, e_n)\}$.

To clarify the concept, let us consider an example with two query expansions, e_1 and e_2. Say the expanded query (q, e_1) retrieves the answer passage as the top result (first position), we assign $r_1 = 1$. Similarly, we assign $r_2 = 15$ if the expanded query (q, e_2) retrieves the answer passage in the 15th position. In this case, we conclude that e_1 is a better query expansion than e_2 since its corresponding ranking value, r_1, is lower than r_2.

r_i can be seen as the *score* that can be obtained by the query of (q, e_i), with smaller r_i corresponding to better quality of (q, e_i). We now train a scoring model to estimate the rank r_i for given inputs (q, e_i). However, considering that the scoring model will be used as a reranker, we only need to ensure the model's *relative* accuracy of estimating r_i, rather than its *absolute* value. Thus, we employ a "contrastive" loss rather than a "regression" loss, which is inspired by the contrastive method in summarization re-scoring (Liu and Liu, 2021).

For all pairs of query expansions (e_i, e_j) such that $r_i < r_j$ (which means e_i is a better expansion than e_j), the ranking loss is calculated as follows:

$$\mathcal{L}_{\text{Rank}} = \sum_{\substack{i,j \in [1,n] \\ r_i < r_j}} \max(0, \mathcal{M}(q, e_i) - \mathcal{M}(q, e_j) + (r_j - r_i) \cdot \alpha)$$

Here, $\mathcal{M}$ is a model that estimates the rank r_i for a given query expansion e_i. Instead of predicting the absolute rank of r_i, the model $\mathcal{M}$ is trained to predict the difference between r_i and r_j for each pair of expansion (e_i, e_j).

The ranking loss $\mathcal{L}_{\text{Rank}}$ forces the model to estimate a lower rank for e_i and a higher rank for e_j, such that the difference between $\mathcal{M}(q, e_i)$ and $\mathcal{M}(q, e_j)$ is greater than the threshold $(r_j - r_i) \cdot \alpha$, where α is a scalar. If some of the expansions do not retrieve the answer passages within the top-k results (e.g. within the top-100 results), we assign a constant value, `MAX_RANK`, to these expansions.

3.3.2 Retrieval-Dependent (RD) Reranker

The input to the RI Reranker only contains the original query q and the expansion e_i, which may not be sufficient to distinguish good expansions from bad expansions. For example, in Figure 3-1 (c), for the original query *Where do they grow hops in the US?*, it is easy to tell that *Central and South America* is a bad expansion because the US is not in Central and South America. However, for these two expansions: 1) Colorado, Arizona 2) Oregon, Idaho, Washington, it is very hard to tell which one is better without any external knowledge. To alleviate this problem, we propose the Retrieval-Dependent (RD) Reranker, which is able to see the top-1 passages $D = \{d_1, d_2, ..., d_n\}$ retrieved by each query expansion. [1] The inputs of RD reranker will contain the original query q, the query expansions e_i, and the top-1 passage d_i. We train RD reranker with the same ranking loss $\mathcal{L}_{\text{Rank}}$, but replace the model with $\mathcal{M}(q, e_i, d_i)$.

3.3.3 Training Examples Construction

To construct training examples, we generate diverse query expansions, run BM25 retrieval on them, and train the rerankers based on the results. However, using the GAR generators directly may not yield diverse sequences and limit the rerankers'

[1] For RD reranker, we need additional computational costs to retrieve the top-1 passage. However, this process can be efficiently parallelized for all queries using a lightweight BM25 retriever, so the required time is not excessive. We will discuss the latency of EAR further in Section 3.8.

Model	**Natural Questions**			**TriviaQA**		
	Top-5	**Top-20**	**Top-100**	**Top-5**	**Top-20**	**Top-100**
			Dense Retrieval			
DPR	68.3	80.1	86.1	72.7	80.2	84.8
			Lexical Retrieval			
BM25	43.8	62.9	78.3	67.7	77.3	83.9
GAR	60.8	73.9	84.7	71.8	79.5	85.3
SEAL	61.3	76.2	86.3	-	-	-
CCS	63.9	76.8	**86.7**	72.3	80.1	85.8
EAR-RI	63.2	76.4	85.9	73.4	80.8	85.9
EAR-RD	**69.3**	**78.6**	86.5	**77.6**	**82.1**	**86.4**
GAR *best query*	*81.9*	*88.1*	*92.0*	*85.0*	*88.1*	*90.1*
			Fusion (Dense + Lexical) Retrieval			
BM25 + DPR	69.7	81.2	88.2	71.5	79.7	85.0
GAR + DPR	72.3	**83.1**	88.9	75.7	82.2	86.3
CCS + DPR	72.7	83.0	89.1	76.1	82.5	86.4
EAR-RI + DPR	71.1	82.5	89.1	76.4	83.0	87.0
EAR-RD + DPR	**74.2**	**83.1**	**89.3**	**79.0**	**83.7**	**87.3**

Table 3.2: Top-k retrieval accuracy (%) on the NQ and TriviaQA test sets. Numbers for prior work are cited from Liu et al. (2022).

learning, since the GAR generators are trained with supervision and may have already overfit on the training set, which would lead to almost identical generation samples. To address this, we propose two alternatives: 1) Split the training set into K subsets, train K different GAR generators on (K-1) subsets and randomly sample from the remaining subset; and 2) Use a large language model (LLM) such as T0 (Sanh et al., 2021) to randomly sample query expansions directly without fine-tuning. Both options performed equally well in our experiments and will be further discussed in Section 3.6.1.

3.4 Experiments

3.4.1 Data

For in-domain experiments, we use two public datasets for training and evaluation: Natural Questions (NQ) (Kwiatkowski et al., 2019) and TriviaQA (Joshi et al., 2017). For out-of-domain (cross-dataset) experiments, we directly evaluate our in-domain models on three additional public datasets without using their training sets: WebQuestions (WebQ) (Berant et al., 2013), CuratedTREC (TREC) (Baudiš and Šedivỳ, 2015), and EntityQuestions (EntityQs) (Sciavolino et al., 2021). We show the number of train/dev/test examples in each dataset in Table 3.3. All experiments are performed with Wikipedia passages used in DPR (Karpukhin et al., 2020), consisting of 21M 100-word passages from the English Wikipedia dump of Dec. 20, 2018 (Lee et al., 2019).

Dataset	**Train**	**Dev**	**Test**
Natural Questions	58,880	8,757	3,610
TriviaQA	60,413	8,837	11,313
WebQuestions	-	-	2,032
TREC	-	-	694
EntityQs	-	-	22,075

Table 3.3: Number of train/dev/test examples in each dataset.

3.4.2 Setup

Model For sparse retrieval, we use Pyserini (Lin et al., 2021) for BM25 with its default parameters. For query rerankers, we use the DeBERTa V3 base (He et al., 2021) model from Huggingface Transformers (Wolf et al., 2020). For RI reranker, the input format is: `[CLS] <question> ? <expansion> [SEP]`; for RD reranker, the input format is `[CLS] <question> ? <expansion> [SEP] <top-1 retrieved passage> [SEP]`. Training details can be found in Appendix B.1.

Context Generator At training time, we use T0-3B (Sanh et al., 2021) to randomly sample 50 query expansions per question, as we mentioned in Section 3.3.3. We add a short prompt, *To answer this question, we need to know*, to the end of the original question, and let T0-3B complete the sentence. During inference, we still use the GAR generators to randomly sample 50 query expansions per question on the testing set, since the examples are not seen during GAR training and the generations are diverse enough. To speed up the inference process, we de-duplicate the query expansions that appear more than once. The average number of query expansions we use is 25 for Natural Questions and 34 for TriviaQA, respectively.

3.4.3 Baselines

We compare EAR with 1) DPR (Karpukhin et al., 2020): a standard BERT-based dense retriever; 2) BM25 (Robertson et al., 2009): a standard sparse retriever based on term matching; 3) GAR (Mao et al., 2021b): generation-augmented retrieval with BM25; 4) Contextual Clue Sampling (CCS) (Liu et al., 2022): a concurrent work that uses a GAR-like generative model to perform beam search decoding, followed by filtering to obtain multiple expanded queries for performing multiple retrievals with BM25, and then fusion of the results; and 5) SEAL (Bevilacqua et al., 2022): an autoregressive search engine, proposing constrained decoding with the FM-index data structure that enables autoregressive models to retrieve passages.

3.4.4 Result: In-Domain Dataset

We first train and evaluate EAR on NQ and TriviaQA. In Table 3.2, we see that both EAR-RI and EAR-RD improve the performance of GAR significantly. EAR-RI improves the top-5/20/100 accuracy of GAR by 1-2 points, while EAR-RD improves the top-5 accuracy of GAR by 6-8 points, and the top-20 accuracy by 3-5 points on both datasets. Moreover, EAR-RD is significantly better than DPR except for the top-20 accuracy on NQ. These results show that it is possible for BM25 to beat dense retrieval with the help of an optimized process to generate high-quality query expansions. Additional qualitative studies in Appendix 3.7 provide further insight into how EAR works. We also report the results of *the best query from* GAR, which presents the potential performance upper bound that could be achieved by query reranking. It suggests that there is still room for EAR to improve if mechanisms for more effective query selection are developed. At the bottom of Table 3.2, we present the fusion retrieval results of combining EAR and DPR. EAR-RD+DPR outperforms the fusion results of BM25/GAR/CCS, showing the complementarity between EAR-RD and DPR.

Model	**WebQuestions**			**TREC**			**EntityQuestions**		
	Top-5	**Top-20**	**Top-100**	**Top-5**	**Top-20**	**Top-100**	**Top-5**	**Top-20**	**Top-100**
BM25	41.8	62.4	75.5	64.3	80.7	89.9	60.6	70.8	79.2
				In-Domain Supervised					
DPR[†]	62.8	74.3	82.2	66.6	81.7	89.9	-	-	-
				Transfer from NQ					
DPR[†]	52.7	68.8	78.3	74.1	85.9	92.1	38.1	49.7	63.2
GAR	50.0	66.0	79.0	70.9	83.9	92.4	59.7	71.0	79.8
EAR-RI	53.7	69.6	81.3	73.5	85.9	92.9	62.7	73.3	81.4
EAR-RD	**59.5**	**70.8**	**81.3**	**80.0**	**88.9**	**93.7**	**65.5**	**74.1**	**81.5**
GAR *best q.*	*78.9*	*85.4*	*90.3*	*93.1*	*95.5*	*97.1*	*78.6*	*85.2*	*90.9*
				Transfer from TriviaQA					
DPR[†]	**56.8**	**71.4**	**81.2**	78.8	87.9	**93.7**	51.2	62.7	74.6
GAR	45.5	61.8	76.7	71.5	84.0	91.5	58.2	68.9	78.7
EAR-RI	49.6	67.1	79.6	74.2	86.2	92.5	62.1	72.0	80.4
EAR-RD	54.5	68.0	79.7	**79.8**	**88.5**	93.1	**64.9**	**73.0**	**80.5**
GAR *best q.*	*78.4*	*84.6*	*89.3*	*92.5*	*95.2*	*96.8*	*79.1*	*85.9*	*91.8*

Table 3.4: Top-k retrieval accuracy on the test sets of three datasets for cross-dataset generalization settings. GAR *best q.* represents the performance upper bound we can achieve by selecting the best query according to the labels. Numbers in bold are the best scores for each setting. [†]Results are provided by Ram et al. (2022).

3.4.5 Result: Cross-Dataset Generalization

To better evaluate the robustness of these models for out-of-domain examples, we train our models only on NQ or TriviaQA, and then test them on WebQ, TREC, and EntityQs in a *zero-shot* manner. The results are shown in Table 3.4. We observe that when transferring from NQ or TriviaQA, DPR experiences a decline in performance compared to in-domain supervised training on WebQ.[2] GAR performs even worse than DPR on both WebQ and TREC. However, GAR performs better than DPR on EntityQs, which is designed to challenge dense retrieval by including many rare entities. Here we also present the performance of GAR *best query*. We see that although GAR transfers poorly on cross-domain datasets, it still has the ability to generate high-quality query expansions by random sampling. This provides an opportunity for EAR to improve performance. After adopting EAR, we see that EAR-RI improves the performance of GAR by 2-4 points for top-5/20 accuracy, and EAR-RD further boosts the performance of GAR by 5-10 points for top-5/20 accuracy. Overall, EAR-RD outperforms DPR except when transferring from TriviaQA to WebQ.

These results suggest that query reranking is a general technique that can work well even on out-of-domain examples, showing that *generating relevant contexts* (GAR) is largely dependent on the domains, while *judging which contexts may be more beneficial to retriever* is a more domain-agnostic skill.

3.4.6 Result: End-to-end QA with FiD

To fully understand whether EAR can benefit end-to-end QA systems, we further evaluate the exact match scores with Fusion-in-Decoder (FiD) (Izacard and Grave, 2021), a generative reader model trained from T5-large (Raffel et al., 2020). We take the FiD models that were pre-trained on NQ/TriviaQA and directly test on our retrieval results without further fine-tuning. The exact match scores using the top-100 retrieved passages as input to FiD is shown at the top of Table 3.5. We observe that EAR consistently outperforms previous work, including DPR, GAR, SEAL, and CCS, on both NQ and TriviaQA. Although these gains may appear relatively small, however, this is primarily due to FiD's ability to take the top-100 retrieved passages as input and generate answers using cross-attention across all passages. Thus, even with low-ranked answer passages (say the answer is in the 99th passage), it is still possible that FiD could produce correct answers.

As there are many methods where relatively smaller context windows compared

[2]The in-domain DPR performs poorly on TREC since it only has 1,125 training examples.

Model	**NQ**	**TriviaQA**
Top-100 passages as input to FiD		
DPR + Extractive	41.5	57.9
RAG	44.5	56.1
DPR + FiD	51.4	67.6
GAR + FiD	50.6	70.0
SEAL + FiD	50.7	-
CCS Liu et al. (2022) + FiD	51.7	70.8
EAR RI + FiD	51.4	71.2
EAR RD + FiD	**52.1**	**71.5**
Top-10 passages as input to FiD		
GAR + FiD	30.5	48.9
EAR RI + FiD	35.5	56.7
EAR RD + FiD	**39.6**	**60.0**

Table 3.5: End-to-end QA exact-match scores on the test sets of NQ and TriviaQA. Numbers for prior work are cited from Liu et al. (2022).

to FiD are used, especially when models are scaled up and cross-attention becomes much more expensive, improving retrieval accuracy for smaller k may be beneficial. For example, GPT-3 (Brown et al., 2020) only has a context window size of 2048, which can only support 10-20 passages as input. We explore this setting by selecting only the top-10 retrieved passages as input to FiD, and show the results at the bottom of Table 3.5. EAR achieve significant improvement over GAR, roughly 10% in exact match on both datasets, showing potential benefits for methods with limited context window size.

3.5 Query Reranking vs Passage Reranking

EAR shares similarities with passage reranking (PR). EAR reranks the queries before retrieving the passages, while PR reranks the retrieved list of passages after the retrieval process is completed. To better understand the relationship between EAR and PR, we implement a BERT-based passage reranker, following the method outlined in (Nogueira and Cho, 2019), to rerank the retrieval results of GAR. The implementation details can be found in Appendix B.2. From the experiments we aim to answer three questions: 1) Is EAR better than PR? 2) Are the contributions of EAR and PR complementary? Can their performance gains be aggregated if we apply both? 3) What are the extra advantages of EAR compared to PR?

Model	**Natural Questions**			**TriviaQA**		
	Top-5	**Top-20**	**Top-100**	**Top-5**	**Top-20**	**Top-100**
BM25	43.8	62.9	78.3	67.7	77.3	83.9
GAR	60.8	73.9	84.7	71.8	79.5	85.3
GAR + Passage Rerank (k=25/k=32)	68.8	75.7	84.7	77.6	81.0	85.3
GAR + Passage Rerank (k=100)	71.7	80.2	84.7	79.2	83.3	85.3
EAR-RD	69.3	78.6	**86.5**	77.6	82.1	**86.4**
EAR-RD + Passage Rerank (k=100)	**73.7**	**82.1**	**86.5**	**80.6**	**84.5**	**86.4**

Table 3.6: Top-k retrieval accuracy (%) on the Natural Questions and TriviaQA test sets for comparison of query reranking and passage reranking.

Is EAR better than PR? We focus on comparing EAR-RD with PR, as EAR-RI is limited by its input, being able to see only the short expanded queries. On the other hand, EAR-RD has access to the top-1 passage retrieved by each query candidate, providing it with the same level of information as PR. In Table 3.6, we first present the performance of PR when reranking the same number of passages as the average number of query candidates considered by EAR (25 for NQ; 32 for TriviaQA), which can be found in row 3. The result of EAR-RD (shown in row 5) is better than row 3, indicating that when considering the same amount of information as inputs, EAR-RD outperforms PR. However, when PR is able to rerank a larger number of passages, such as the top-100 passages shown in row 4, it achieves better performance than EAR-RD. This implies that EAR-RD is more effective when PR can only access to the same level of information.

Are EAR and PR complementary? We found that the effects of EAR-RD and PR can be effectively combined for even better performance. When applying PR on the retrieval results of EAR-RD (shown in row 6), we see a significant improvement compared to both row 4 and row 5. This suggests that the contributions of EAR-RD and PR are complementary: EAR strengthens first-pass retrieval by selecting good queries, while PR re-scores all the retrieved passages and generates an entirely

Model	**Top-5**	**Top-20**	**Top-100**
EAR-RI	63.2	76.4	85.9
EAR-RI holdout	63.6	76.3	86.0
EAR-RD	69.3	78.6	86.5
EAR-RD w/ DPR	65.7	78.7	86.3

Table 3.7: Top-k retrieval accuracy (%) on NQ for comparison of the two different training example construction methods and for EAR with dense retrievers.

new order for these passages. The distinction between these two mechanisms makes improvements accumulative and leads to superior results.

Extra advantages of EAR? An advantage of EAR is that it improves retrieval results beyond the top-k passages. In row 4, the top-100 accuracy cannot be improved by PR as it reranks within the top-100 passages. In contrast, the improvements provided by EAR are not limited to top-100 passages. As long as EAR selects good query expansions, it can improve the whole list of retrieved passages; we can see EAR-RD improves the top-100 accuracy of GAR from 84.7 to 86.5.

3.6 Discussion

3.6.1 Generating Training Examples with GAR

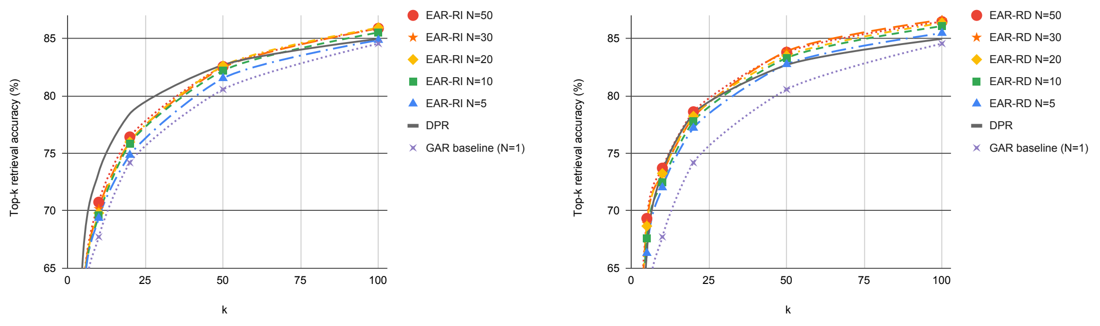
In Section 3.3.3, we discussed two methods to construct training examples for EAR. In our main experiments, we used T0-3B to randomly sample diverse query expansions. An alternative method was also explored, where we trained $K = 5$ different GAR models separately on $(K-1)$ training subsets, then randomly sampled from the hold-out sets. The performance of this method, as shown in Table 3.7 (EAR-RI holdout), is slightly better than using T0-3B, but the difference is less than 0.5 points on Top-5/20/100 accuracy. Therefore, we continue to use T0-3B to generate training data in our main experiments as it eliminates the need to train K different GAR models separately.

Model	RI	RD
Ear N=50	63.16	69.34
Ear N=30	63.02	68.86
Ear N=20	62.96	68.67
Ear N=10	62.60	67.62
Ear N=5	62.44	66.32
GAR baseline (N=1)	60.80	

Table 3.8: Top-5 accuracy on NQ with different N. N stands for the maximum number of query expansions considered by the query reranker.

3.6.2 Ear with Dense Retrievers

Ear is specifically optimized to work well with the BM25 retriever and hence its performance may be impacted when changing the retriever to DPR. As shown at the bottom of Table 3.7, when coupled with DPR, the top-5 accuracy of Ear-RD decreases, while the top-20/100 accuracy remains relatively unchanged. This suggests that Ear is heavily reliant on the retriever, and thus changing the retriever negatively impacts its performance. Making Ear work with DPR would require retraining with DPR retrieval results and significantly more compute. We leave this direction for future work.



(a) Ear-RI (b) Ear-RD

Figure 3-2: Top-k performance curves on NQ for Ear-RI and Ear-RD with a reduced candidate size N.

3.6.3 Reducing the Query Candidate Size

In our experiments, we generate 50 query expansions per question and then deduplicate the repeated ones. However, we can also limit the maximum query expansions considered by our reranker to trade-off between efficiency and performance. In Table 3.8 we show the top-5 accuracy of lowering the maximum candidate size N from 50 to 30/20/10/5. We observe that the performance drops gradually as N decreases. However, we still see improvement over GAR even when $N = 5$, showing that EAR still works with a small candidate size. We also show the curves of the top-k accuracy in Figure 3-2, where we observe a big gap between DPR (solid line) and GAR (dotted line with x mark). EAR-RI gradually eliminates the gap as N increase, while EAR-RD even matches DPR for $k < 50$ and outperforms DPR for $k \geq 50$ with a small $N = 5$.

3.7 Qualitative Study

Model	**answer**	**sentence**	**title**
Original Query		9.2	
GAR	13.3	38.8	32.3
EAR-RI	13.1	36.2	29.3
EAR-RD	13.2	38.2	28.8

Table 3.9: Lengths of the expanded queries in words for different methods on NQ test set.

In this section, we aim to investigate the differences between queries generated by GAR and EAR. We first look at the lengths of the expanded queries for GAR, EAR-RI, EAR-RD. In general, the lengths of queries from EAR are slightly shorter than that of GAR, but the trends are not very obvious. Thus, we conducted a qualitative study to analyze the differences between these queries.

As shown in Table 3.10, we provide three examples to demonstrate how our EAR method works. In the first example, the initial query only includes two keywords, “Deadpool” and “released,” that can match the answer passage. As a result, the

Model	Query [Answer = May 18, 2018]	Answer Rank
BM25	When is the next **Deadpool** movie being **released**?	77
GAR	When is the next **Deadpool** movie being **released**? Miller Brianna Hildebrand Jack Kesy Music by Tyler Bates Cinematography Jonathan Sela Edited by Dirk Westervelt...	>100
EAR-RI	When is the next **Deadpool** movie being **released**? Deadpool 2 is scheduled to be released on May 26, 2018 , with Leitch directing.	1
EAR-RD	When is the next **Deadpool** movie being **released**? The film is scheduled to be released on March 7, 2018 , in the United States .	1
	Answer Passage	
	"**Deadpool** 2" premiered at Leicester Square in London on May 10, 2018. It was **released** in the United States on May 18, 2018 , having been previously scheduled for release on June 1 of that year. Leitch ś initial cut of the film was around two hours and twelve minutes, ...	

Model	Query [Answer = Natasha Bharadwaj, Aman Gandotra]	Answer Rank
BM25	Who has won **India's next** super **star**?	96
GAR	Who has won **India's next** super **star**? The winner of the competition is 18 year-old Mahesh Manjrekar from Mumbai.	>100
EAR-RI	Who has won **India's next** super **star**? The winner of the Superstar Season 2018 is Siddharth Shukla.	1
EAR-RD	Who has won **India's next** super **star**? The winner of the Superstar Season 2018 is Siddharth Shukla.	1
	Answer Passage	
	India's Next Superstars (INS) is an Indian talent-search reality TV show, which premiered on **Star** Plus and is streamed on Hotstar. Karan Johar and Rohit Shetty are the judges for the show. Aman Gandotra and Natasha Bharadwaj were declared winners of the 2018 season ...	

Model	Query [Answer = Anthropomorphism, Pathetic fallacy, Hamartia, Personification]	Answer Rank
BM25	**Method** used by a **writer** to develop a character?	92
GAR	**Method** used by a **writer** to develop a character? Developing a character is a technique employed by writers in the creation of a narrative.	>100
EAR-RI	**Method** used by a **writer** to develop a character? Developing a character is the primary method employed by writers in the creation of a fictional character.	>100
EAR-RD	**Method** used by a **writer** to develop a character? Developing a character is a technique employed by writers in terms of establishing a persona and building a relationship between the reader and the character.	>100
	Answer Passage	
	The intensive journal **method** is a psychotherapeutic technique largely developed in 1966 at Drew University and popularized by Ira Progoff (1921-1998). It consists of a series of writing exercises using loose leaf notebook paper in a simple ring binder, divided into sections to helping accessing various areas of the **writer**'s life. These include a dialogue section for the personification of things, a "depth dimension" to aid in accessing the subconscious and other places for	

Table 3.10: Examples that show the difference between BM25/GAR/EAR-RI/EAR-RD. Words in blue are query expansions generated by GAR. **Bold words** are useful keywords from the original query. Words highlighted in green are useful keywords generated by GAR. **Answer Rank** shows the ranking of the answer passage in the retrieval results.

BM25 algorithm is unable to retrieve the correct passage within the top results until the 77th passage. The greedy decoding output for GAR also fails to retrieve the correct passage, as it includes many irrelevant name entities. However, both EAR-RI and EAR-RD are able to select useful outputs from GAR, which contain keywords such as "scheduled," "2018," "Leitch," and "in the United States." Although none of these keywords contains the real answer *May 18, 2018*, these keywords already provide enough lexical overlap with the answer passage, allowing BM25 to correctly retrieve the answer passage in the top-1 result.

For the second example, the original query only contains three keywords "India's," "next," and "star" that can match the answer passage, so BM25 with the original query cannot retrieve the correct passage within the top retrieved results until the 96th passage. For GAR, the greedy decoding output for GAR is also not effective, as it is a misleading answer and only includes one useful keyword "winner" and thus cannot retrieve the correct passage within the top-100 results. For EAR-RI and EAR-RD, they are able to select a sentence that, while not containing the correct answer "Natasha Bharadwaj" or "Aman Gandotra," does include useful keywords such as "winner," "Superstar," "Season," and "2018." These keywords provide enough lexical overlap with the answer passage, allowing EAR-RI and EAR-RD to correctly retrieve the answer passage in the top-1 result.

The third example presents a challenging scenario. The initial query only includes two common keywords, "method" and "writer," which makes it difficult to match the answer passage. While BM25 is able to correctly retrieve the answer at the 92nd passage, the generated query expansions are not helpful and instead are misleading, resulting in GAR and EAR-RI/EAR-RD all unable to retrieve the correct passage within the top-100 results due to the distracting query expansions. This example illustrates the importance of the GAR generators. If all of the generated query expansions are not useful, EAR is unable to improve the results.

3.8 Computational Cost and Latency

We report the latency of DPR, GAR, and EAR in Table 3.11. Inference details can be found in Appendix B.3.

Model	Build Index	Query Expand	Query Rerank	Retrieval	Index Size	Top-5 (NQ)
DPR	3.5hr	-	-	22.4s	64GB	68.3
+HNSW	8.5hr	-	-	0.04s	142GB	68.0
BM25	0.5hr	-	-	0.15s	2.4GB	43.8
GAR	0.5hr	0.58s	-	0.56s	2.4GB	60.8
EAR-RI	0.5hr	1.29s	0.04s	0.50s	2.4GB	63.2
EAR-RD	0.5hr	1.29s	0.84s	0.54s	2.4GB	69.3

Table 3.11: Latency per query for DPR/BM25/GAR/EAR.

Dense Retrieval We first generate DPR document embeddings on 4 GPUs for ~3.5 hours on 21M documents. Standard indexing takes ~10 minutes with a 64GB index size. Indexing with the more advanced Hierarchical Navigable Small World (HNSW) (Malkov and Yashunin, 2018) takes ~5 hours and results in a huge index size of 142GB. For retrieval, standard indexing takes 22.3s per query, while the highly optimized HNSW can shorten it to 0.04s per query.

Sparse Retrieval For BM25 with Pyserini, indexing only takes 0.5 hours, with a very small index size of 2.4GB. Retrieval for BM25 takes 0.15s per query. For GAR, it needs an extra 0.58s to generate the query expansions, and retrieval time is 0.56s. For EAR, it needs 1.29s to batch sample 50 query expansions. EAR-RI only takes 0.04s to rerank queries. EAR-RD needs extra time to retrieve the top-1 passages for each expansion, which takes an extra 0.70s, and then run the actual reranking process, taking 0.14s, giving a total of 0.84s for query reranking. For retrieval, the time needed for both EAR-RI and EAR-RD is similar to GAR.

To conclude, EAR inherits the advantage of BM25: *fast indexing time* and *small index size*. This makes it possible to index large collections of documents in a relatively short amount of time, which is important for tasks where documents are frequently added or updated. The main cost for EAR is the time for sampling query expansions. However, this can potentially be reduced by speed-up toolkits that optimize the inference time of transformers, such as FasterTransformer [3] (3.8~13× speedup for decoding) or FastSeq (Yan et al., 2021a; 7.7× speedup for BART decoding). Moreover, we can leverage model distillation (Shleifer and Rush, 2020) and quantization (Li

[3]https://github.com/nvidia/fastertransformer

et al., 2022) for transformers. We leave these directions for future work.

3.9 Related Work

Query Expansion and Reformulation Traditionally, query expansion methods based on pseudo relevance feedback utilize relevant context without external resources to expand queries (Rocchio, 1971; Jaleel et al., 2004; Lv and Zhai, 2010; Yu et al., 2021). Recent studies attempt to reformulate queries using generative models, relying on external resources such as search sessions (Yu et al., 2020) or conversational contexts (Lin et al., 2020; Vakulenko et al., 2021), or involve sample-inefficient reinforcement learning training (Nogueira and Cho, 2017). More recently, GAR (Mao et al., 2021b) explored the use PLMs for query expansion instead of external resources. A concurrent study (Liu et al., 2022) generates multiple expansions with beam search and filters and fuses the results, but EAR is aware of the BM25 retriever and could select more promising query expansions and run fewer BM25 retrievals.

Retrieval for OpenQA Sparse retrieval with lexical features such as BM25 was first explored for OpenQA (Chen et al., 2017). Dense retrieval methods were shown to outperform sparse methods (Karpukhin et al., 2020; Guu et al., 2020), while requiring large amounts of annotated data and much more compute. Although powerful, dense retrievers often fall short in the scenarios of 1) requiring lexically exact matching for rare entities (Sciavolino et al., 2021) and 2) out-of-domain generalization (Reddy et al., 2021). For 1), (Luan et al., 2021b) proposed a sparse-dense hybrid model, and (Chen et al., 2021) trained a dense retriever to imitate a sparse one. For 2), (Ram et al., 2022) created a pre-training task for dense retrievers to improve zero-shot retrieval and out-of-domain generalization. Another recent line of research explores passage reranking with PLMs to improve performance for both sparse and dense methods. (Nogueira and Cho, 2019) first explored BERT-based supervised rerankers for standard retrieval tasks and (Mao et al., 2021c) proposed reranking by reader predictions without any training. (Sachan et al., 2022) attempt to use an LLM directly as the reranker, but it requires huge amounts of computation at inference time and underperforms fine-tuned rerankers.

3.10 Limitations

Since EAR largely relies on GAR generators, the performance of the method is closely tied to the quality of the generator used. We have attempted to use large language models such as T0-3B without fine-tuning as a replacement for the GAR generator during testing, but the performance decreases. The main reason is that the quality of query expansions generated by T0-3B is too diverse, which means EAR has a higher chance to select a poor quality expansion. In contrast, the output quality of GAR is more stable. We may need a more complex mechanism that can exclude poor quality query expansion if we want to directly use the query expansions generated by T0-3B during inference.

EAR has demonstrated a strong generalization ability to out-of-domain data, but the method may still face challenges when transferring to other languages without any supervised QA data, which GAR and EAR are trained on. Although challenging, we are still trying to train the EAR system without supervised QA data.

3.11 Chapter Summary

In this chapter, we propose EAR, which couples GAR and BM25 together with a query reranker to unlock the potential of sparse retrievers. EAR significantly outperforms DPR while inheriting the advantages of BM25: *fast indexing time* and *small index size* compared to the compute-heavy DPR. The cross-dataset evaluation also shows that EAR is very good at generalizing to out-of-domain examples. Furthermore, we demonstrate that contributions of EAR and passage reranking are complementary, and using both methods together leads to superior results. Overall, EAR is a promising alternative to existing dense retrieval models, providing a new way to achieve high performance with less computing resources.

Chapter 4

Conclusions and Future Work

4.1 Conclusion

This thesis presents two contributions to the field of natural language processing and information retrieval. Chapter 2 introduces DiffCSE, a novel approach to unsupervised sentence embedding that leverages MLM-based word replacement awareness as the auxiliary task. DiffCSE demonstrates superior performance in semantic textual similarity and transfer tasks, outperforming current state-of-the-art methods. Its application to dense paragraph retrieval for open-domain question answering further highlights its generalizability and potential for application to a wider range of domains.

Chapter 3 proposes Ear, a model that enhances Gar and BM25 with a query reranker, improving the capabilities of sparse retrievers. Ear excels in fast training and inference as well as small index size while maintaining high performance, particularly in cross-dataset generalization. This model offers a computationally efficient alternative to existing dense retrieval models, proving that high performance can be achieved with fewer resources.

Although DiffCSE and Ear provide innovations from different perspectives of information retrieval, i.e. dense retrieval vs sparse retrieval, both DiffCSE and Ear are built based on the idea of "contrasting". DiffCSE enables the model to contrast and compare similar sentences edited by the MLM model, so as to enhance the model's ability to distinguish the subtle differences between sentences. Ear trains the model to contrast the different query expansions generated by an LM, enabling the model to select the high-quality expansions out of the bad ones.

4.2 Future Work

For future research, several paths can be further explored:

- **DiffCSE in other scenarios:** The idea of DiffCSE can be extended to other domains or modalities. For example, Zhou et al. (2023) recently proposed ECL-SR, a DiffCSE-like architecture that also contains a random replacement generator and a conditional discriminator, but operates on top of the user interaction history data. ECL-SR has been proven to be effective in improving sequential recommendation models. Besides sequential recommendation, we believe that DiffCSE also has the potential to be leveraged in other modalities such as vision and speech. The discretized image or audio representations (Mao et al., 2021a; Hsu et al., 2021) allow the discrete operations on the input modalities, making it easier to apply the same pipeline of a random replacement generator with a conditional discriminator, to improve the quality of the instance-level representations in these modalities.

- **Scaling Ear:** The Ear models are based on BART-large (406M) (Lewis et al., 2020) generators and DeBERTa-V3-base (86M) (He et al., 2021) rerankers. With the recent advancements in large-scale LLMs like LLaMA with up to 70B parameters (Touvron et al., 2023a,b), we can potentially leverage these LLMs' internal knowledge to improve the effectiveness of Ear. A straightforward direction will be fine-tuning LLaMA models to be the generator and the reranker.

- **Unsupervised Ear:** Unsupervised retrieval models have been explored in dense retrieval settings (Izacard et al., 2022; Ram et al., 2022), but they are still under-explored in the query expansion scenarios. A promising direction will be training language models to generate query expansions without using the labeled QA datasets. For example, we can design unsupervised methods to extract pseudo-positive pairs (Ram et al., 2022) of query-document from unlabeled corpora. If such a method can be applied to extract large-scale datasets for unsupervised EAR, it is possible to scale up the unsupervised EAR model and try to achieve the performance of supervised EAR.

- **EAR retrieval with multiple queries:** In the setting of EAR, we use the top-reranked single query for retrieval. However, we can also utilize the top N reranked queries for retrieving documents, and then combine the retrieved documents by second-stage passage reranking. By leveraging the different query expansions, it is more likely to cover different aspects of the information relevant to the query, resulting in better retrieval accuracy.

Appendix A

Experiment Details for Chapter 2

A.1 Training Details

We use a single NVIDIA 2080Ti GPU for each experiment. The averaged running time for DiffCSE is 3-6 hours. We use grid-search of batch size $\in \{64, 128\}$ learning rate $\in$ {2e-6, 3e-6, 5e-6, 7e-6, 1e-5} and masking ratio $\in \{0.15, 0.20, 0.30, 0.40\}$ and $\lambda \in \{0.1, 0.05, 0.01, 0.005, 0.001\}$. The temperature τ in SimCSE is set to 0.05 for all the experiments. During the training process, we save the checkpoint with the highest score on the STS-B development set. And then we use STS-B development set to find the best hyperparameters (listed in Table A.1) for STS task; we use the averaged score of the development sets of 7 transfer tasks to find the best hyperparameters (listed in Table A.2) for transfer tasks. All numbers in Table 2.1 and Table 2.2 are from a single run.

hyperparam	**BERT$_{\text{base}}$**	**RoBERTa$_{\text{base}}$**
learning rate	7e-6	1e-5
masking ratio	0.30	0.20
λ	0.005	0.005
training epochs	2	2
batch size	64	64

Table A.1: The main hyperparameters in STS tasks.

During testing, we follow SimCSE to discard the MLP projector and only use the `[CLS]` output to extract the sentence embeddings.

hyperparam	BERT$_{\text{base}}$	RoBERTa$_{\text{base}}$
learning rate	2e-6	3e-6
masking ratio	0.15	0.15
λ	0.05	0.05
training epochs	2	2
batch size	64	128

Table A.2: The main hyperparameters in transfer tasks.

Method	BERT$_{\text{base}}$	RoBERTa$_{\text{base}}$
SimCSE	110M	125M
DiffCSE (train)	220M	250M
DiffCSE (test)	110M	125M

Table A.3: The number of parameters used in our models.

The numbers of model parameters for BERT$_{\text{base}}$ and RoBERTa$_{\text{base}}$ are listed in Table A.3. Note that in training time DiffCSE needs two BERT models to work together (sentence encoder + discriminator), but in testing time we only need the sentence encoder, so the model size is the same as the SimCSE model.

Projector with BatchNorm In Section 2.5, we mention that we use a projector with BatchNorm as the final layer of our model. Here we provided the PyTorch code for its structure:

```
class ProjectionMLP(nn.Module):
    def __init__(self, hidden_size):
        super().__init__()
        in_dim = hidden_size
        middle_dim = hidden_size * 2
        out_dim = hidden_size
        self.net = nn.Sequential(
        nn.Linear(in_dim, middle_dim, bias=False),
        nn.BatchNorm1d(middle_dim),
        nn.ReLU(inplace=True),
        nn.Linear(middle_dim, out_dim, bias=False),
        nn.BatchNorm1d(out_dim, affine=False))
```

Appendix B

Experiment Details for Chapter 3

B.1 Training Details

For the training set, we use T0-3B [1] to randomly sample 50 query expansions per query. For the dev set and test set, we use the three GAR generators (answer/sentence/title), which are BART-large seq2seq models (Lewis et al., 2020) to generate 50 query expansions per query. We use the DeBERTa V3 base model[2], which has 86M parameters that are the same as BERT-base (Devlin et al., 2019), as EAR-RI and EAR-RD rerankers. For the implementation of rerankers, we reference the implementation of SimCLS (Liu and Liu, 2021)[3], which also does reranking for sequences. We start from the code of SimCLS and change the loss function to our ranking loss $\mathcal{L}_{\text{Rank}}$. During training, we use the dev set generated from three GAR generators to pick the best checkpoints, resulting in three different reranker models corresponding to the answer/sentence/title generators.

The ranges we search for our hyperparameters are shown in Table B.1. Each training example in our dataset contains 50 sequences (generated by T0-3B). To prevent memory issues of GPU, we used gradient accumulation to simulate a batch size of 4 or 8, which effectively consists of 200 or 400 sequences, respectively.

The training time on a single NVIDIA V100 GPU is around 12 hours for EAR-RI

[1] `https://huggingface.co/bigscience/T0_3B`
[2] `https://huggingface.co/microsoft/deberta-v3-base`
[3] `https://github.com/yixinL7/SimCLS`

and 1 to 2 days for EAR-RD. The best hyperparameters according to the dev set are shown in Table B.2. However, in our experiments, the variance between different hyperparameters is actually quite small.

Hyperparams	**Range**
`MAX_RANK`	[101, 250]
Batch size	[4, 8]
Learning rate	[2e-3, 5e-3]
Epochs (EAR-RI)	2
Epochs (EAR-RD)	3
Max length (EAR-RI)	64
Max length (EAR-RD)	256

Table B.1: The range for hyperparameter search. The definition of `MAX_RANK` is shown in Section 3.3.1.

Hyperparams	**answer**	**sentence**	**title**
NQ: EAR-RI			
`MAX_RANK`	101	250	101
Batch size	8	8	4
Learning rate	5e-3	2e-3	2e-3
NQ: EAR-RD			
`MAX_RANK`	101	101	250
Batch size	4	4	8
Learning rate	2e-3	2e-3	2e-3
TriviaQA: EAR-RI			
`MAX_RANK`	101	101	101
Batch size	8	8	8
Learning rate	2e-3	2e-3	2e-3
TriviaQA: EAR-RD			
`MAX_RANK`	101	101	101
Batch size	8	8	8
Learning rate	2e-3	2e-3	2e-3

Table B.2: The best hyperparameters for NQ and TriviaQA dev sets.

B.2 Passage Reranking Details

For the implementation of a BERT-based passage reranker, we generally follow the setting of Nogueira and Cho (2019) for training. We separately fine-tuned two `bert-base-uncased` models on the NQ training set and the TriviaQA training set. Each pre-trained BERT model is fine-tuned for reranking using cross-entropy loss on the binary classification head on top of the hidden state corresponding to the [CLS] token. We use the top-10 outputs of BM25 ran on the training sets as the training examples, which contains both positive and negative examples. We fine-tune the models using 2 GPUs with mixed precision (fp16) with a batch size of 128 for 3 epochs. AdamW (Loshchilov and Hutter, 2018) is used for optimization with a learning rate of 5e-5, linear warmup over the first 10k steps and linear decay afterwards, and a weight decay of 0.01.

B.3 Inference Details

For inference of GAR retrieval results, we follow GAR to retrieve with three queries generated by three context generators (answer/sentence/title), and then fuse the three retrieved passages lists in the order of sentence, answer, title. In other words, given the three retrieved lists of passages: $(a_1, a_2, ..., a_{100})$, $(s_1, s_2, ..., s_{100})$, $(t_1, t_2, ..., t_{100})$, we fuse the results as $(s_1, a_1, t_1, s_2, a_2, t_2, ..., s_{33}, a_{33}, t_{33}, s_{34})$. We skip all the duplicated passages that exist twice during the fusion process.

For EAR, we use the same pipeline of GAR, while the only difference is that instead of greedy decoding, now the three generators of GAR can do random sampling, and three different query rerankers (answer/sentence/title) are applied to select the best queries. After that, the pipeline to obtain retrieval results is exactly the same as GAR.

To fairly compare the latency of these methods, we run the 3610 queries in NQ test set one-by-one without batching (batch size = 1) and compute the average the latency per query, where document encoding, query expansion and reranking are run on NVIDIA RTX A5000 GPUs and indexing and retrieval are run on fifty Intel Xeon Gold 5318Y CPUs @ 2.10GHz, for both FAISS (Johnson et al., 2019) (DPR) and Pyserini (Lin et al., 2021) (BM25).

DPR document indexing We used 4 GPUs to encode 21M Wikipedia passages in parallel with mixed precision (fp16), which takes around 3.5 hours.

Gar and Ear For inference of Gar and Ear, answer/sentence/title generators/r-erankers are run in parallel on three GPUs.

FiD We take the public checkpoints of FiD[4], which are trained from T5-Large (**?**) with NQ/TriviaQA, to directly evaluate the end-to-end QA performance.

[4]`https://github.com/facebookresearch/FiD`

Bibliography

Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2016. https://doi.org/10.18653/v1/S16-1081 SemEval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 497–511, San Diego, California. Association for Computational Linguistics.

Petr Baudiš and Jan Šedivỳ. 2015. Modeling of the question answering task in the yodaqa system. In *International Conference of the cross-language evaluation Forum for European languages*, pages 222–228. Springer.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. https://aclanthology.org/D13-1160 Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, Seattle, Washington, USA. Association for Computational Linguistics.

Michele Bevilacqua, Giuseppe Ottaviano, Patrick Lewis, Scott Yih, Sebastian Riedel, and Fabio Petroni. 2022. https://openreview.net/forum?id=Z4kZxAjg8Y Autoregressive search engines: Generating substrings as document identifiers. In *Advances in Neural Information Processing Systems*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Fredrik Carlsson, Amaru Cuba Gyllensten, Evangelia Gogoulou, Erik Ylipää Hellqvist, and Magnus Sahlgren. 2021. https://openreview.net/forum?id=Ov$_sMNau - PFSemanticre - tuningwithcontrastivetension$.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. https://doi.org/10.18653/v1/S17-2001 SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. 2018. Universal sentence encoder for english. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 169–174.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. https://doi.org/10.18653/v1/P17-1171 Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada. Association for Computational Linguistics.

Danqi Chen and Wen-tau Yih. 2020. https://doi.org/10.18653/v1/2020.acl-tutorials.8 Open-domain question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, pages 34–37, Online. Association for Computational Linguistics.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Xilun Chen, Kushal Lakhotia, Barlas Oğuz, Anchit Gupta, Patrick Lewis, Stan Peshterliev, Yashar Mehdad, Sonal Gupta, and Wen-tau Yih. 2021. Salient phrase aware dense retrieval: Can a dense retriever imitate a sparse one? *arXiv preprint arXiv:2110.06918.*

Xinlei Chen and Kaiming He. 2021. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15750–15758.

Yung-Sung Chuang, Rumen Dangovski, Hongyin Luo, Yang Zhang, Shiyu Chang, Marin Soljačić, Shang-Wen Li, Scott Yih, Yoon Kim, and James Glass. 2022. Diffcse: Difference-based contrastive learning for sentence embeddings. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4207–4218.

Yung-Sung Chuang, Wei Fang, Shang-Wen Li, Wen-tau Yih, and James Glass. 2023. https://doi.org/10.18653/v1/2023.findings-acl.768 Expand, rerank, and retrieve: Query reranking for open-domain question answering. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 12131–12147, Toronto, Canada. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555.*

Alexis Conneau and Douwe Kiela. 2018. https://www.aclweb.org/anthology/L18-1269 SentEval: An evaluation toolkit for universal sentence representations.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680.

Rumen Dangovski, Li Jing, Charlotte Loh, Seungwook Han, Akash Srivastava, Brian Cheung, Pulkit Agrawal, and Marin Soljačić. 2021. Equivariant contrastive learning. *arXiv preprint arXiv:2111.00899*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. https://doi.org/10.18653/v1/N19-1423 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, Minneapolis, Minnesota. Association for Computational Linguistics.

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*.

John M Giorgi, Osvald Nitski, Gary D Bader, and Bo Wang. 2020. Declutr: Deep contrastive learning for unsupervised textual representations. *arXiv preprint arXiv:2006.03659*.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, koray kavukcuoglu, Remi Munos, and Michal Valko. 2020. https://proceedings.neurips.cc/paper/2020/file/f3ada80d5c4ee70142b17b8192b2958e-Paper.pdf Bootstrap your own latent - a new approach to self-supervised learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 21271–21284. Curran Associates, Inc.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*.

Felix Hill, Kyunghyun Cho, and Anna Korhonen. 2016. Learning distributed representations of sentences from unlabelled data. *arXiv preprint arXiv:1602.03483*.

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. 2021. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3451–3460.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177.

Tianyu Hua, Wenxiao Wang, Zihui Xue, Sucheng Ren, Yue Wang, and Hang Zhao. 2021. On feature decorrelation in self-supervised learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9598–9608.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised dense information retrieval with contrastive learning. *Transactions on Machine Learning Research.*

Gautier Izacard and Edouard Grave. 2021. https://doi.org/10.18653/v1/2021.eacl-main.74 Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 874–880, Online. Association for Computational Linguistics.

Nasreen Abdul Jaleel, James Allan, W. Bruce Croft, Fernando Diaz, Leah S. Larkey, Xiaoyan Li, Mark D. Smucker, and Courtney Wade. 2004. Umass at trec 2004: Novelty and hard. In *Text Retrieval Conference.*

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. https://doi.org/10.18653/v1/P17-1147 TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. https://doi.org/10.18653/v1/2020.emnlp-main.550 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, Online. Association for Computational Linguistics.

Taeuk Kim, Kang Min Yoo, and Sang-goo Lee. 2021. Self-guided contrastive learning for bert sentence representations. *arXiv preprint arXiv:2106.07345*.

Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. 2015. https://papers.nips.cc/paper/2015/hash/f442d33fa06832082290ad8544a8da27-Abstract.html Skip-thought vectors. pages 3294–3302.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. https://doi.org/10.18653/v1/P19-1612 Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096, Florence, Italy. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. https://doi.org/10.18653/v1/2020.acl-main.703 BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. https://doi.org/10.18653/v1/2020.emnlp-main.733 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, Online. Association for Computational Linguistics.

Zheng Li, Zijian Wang, Ming Tan, Ramesh Nallapati, Parminder Bhatia, Andrew Arnold, Bing Xiang, and Dan Roth. 2022. Dq-bart: Efficient sequence-to-sequence model via joint distillation and quantization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 203–211.

Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: A Python toolkit for reproducible information retrieval research with sparse and dense representations. In *Proceedings of the 44th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2021)*, pages 2356–2362.

Sheng-Chieh Lin, Jheng-Hong Yang, Rodrigo Nogueira, Ming-Feng Tsai, Chuan-Ju Wang, and Jimmy J. Lin. 2020. Query reformulation using query history for passage retrieval in conversational search. *ArXiv*, abs/2005.02230.

Linqing Liu, Minghan Li, Jimmy Lin, Sebastian Riedel, and Pontus Stenetorp. 2022. Query expansion using contextual clue sampling with language models. *arXiv preprint arXiv:2210.07093.*

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019a. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692.*

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692.*

Yixin Liu and Pengfei Liu. 2021. https://doi.org/10.18653/v1/2021.acl-short.135 SimCLS: A simple framework for contrastive learning of abstractive summarization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 1065–1072, Online. Association for Computational Linguistics.

Lajanugen Logeswaran and Honglak Lee. 2018. https://openreview.net/forum?id=rJvJXZb0W An efficient framework for learning sentence representations.

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

Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins. 2021a. https://doi.org/10.1162/tacl$_{a0}$0369$Sparse, dense, andattentionalrepresentationsfortextretrieval.T$
$329 - -345$.

Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins. 2021b. Sparse, dense, and attentional representations for text retrieval. *Transactions of the Association for Computational Linguistics*, 9:329–345.

Yuanhua Lv and ChengXiang Zhai. 2010. https://doi.org/10.1145/1835449.1835546 Positional relevance model for pseudo-relevance feedback. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '10, page 579–586, New York, NY, USA. Association for Computing Machinery.

Yu A Malkov and Dmitry A Yashunin. 2018. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence*, 42(4):824–836.

Chengzhi Mao, Lu Jiang, Mostafa Dehghani, Carl Vondrick, Rahul Sukthankar, and Irfan Essa. 2021a. Discrete representations strengthen vision transformer robustness. In *International Conference on Learning Representations.*

Yuning Mao, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han, and Weizhu Chen. 2021b. https://doi.org/10.18653/v1/2021.acl-long.316 Generation-augmented retrieval for open-domain question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4089–4100, Online. Association for Computational Linguistics.

Yuning Mao, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han, and Weizhu Chen. 2021c. https://doi.org/10.18653/v1/2021.findings-acl.29 Reader-guided passage reranking for open-domain question answering. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 344–350, Online. Association for Computational Linguistics.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, Roberto Zamparelli, et al. 2014. A sick cure for the evaluation of compositional distributional semantic models. In *Lrec*, pages 216–223. Reykjavik.

Yu Meng, Chenyan Xiong, Payal Bajaj, Saurabh Tiwary, Paul Bennett, Jiawei Han, and Xia Song. 2021. Coco-lm: Correcting and contrasting text sequences for language model pretraining. *arXiv preprint arXiv:2102.08473*.

Rodrigo Nogueira and Kyunghyun Cho. 2017. https://doi.org/10.18653/v1/D17-1061 Task-oriented query reformulation with reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 574–583, Copenhagen, Denmark. Association for Computational Linguistics.

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. *arXiv preprint cs/0409058*.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 115–124.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. http://jmlr.org/papers/v21/20-074.html Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392.

Ori Ram, Gal Shachaf, Omer Levy, Jonathan Berant, and Amir Globerson. 2022. https://doi.org/10.18653/v1/2022.naacl-main.193 Learning to retrieve passages without supervision. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2687–2700, Seattle, United States. Association for Computational Linguistics.

Revanth Gangi Reddy, Vikas Yadav, Md Arafat Sultan, Martin Franz, Vittorio Castelli, Heng Ji, and Avirup Sil. 2021. Towards robust neural retrieval models with synthetic pre-training. *arXiv preprint arXiv:2104.07800*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. 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 3982–3992.

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

J. J. Rocchio. 1971. Relevance feedback in information retrieval. In G. Salton, editor, *The Smart retrieval system - experiments in automatic document processing*, pages 313–323. Englewood Cliffs, NJ: Prentice-Hall.

Devendra Sachan, Mike Lewis, Mandar Joshi, Armen Aghajanyan, Wen-tau Yih, Joelle Pineau, and Luke Zettlemoyer. 2022. https://aclanthology.org/2022.emnlp-main.249 Improving passage retrieval with zero-shot question generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3781–3797, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

G. Salton, A. Wong, and C. S. Yang. 1975. https://doi.org/10.1145/361219.361220 A vector space model for automatic indexing. *Commun. ACM*, 18(11):613–620.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, et al. 2021.

Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*.

Christopher Sciavolino, Zexuan Zhong, Jinhyuk Lee, and Danqi Chen. 2021. https://doi.org/10.18653/v1/2021.emnlp-main.496 Simple entity-centric questions challenge dense retrievers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6138–6148, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Sam Shleifer and Alexander M Rush. 2020. Pre-trained summarization distillation. *arXiv preprint arXiv:2010.13002*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

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

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: On the importance of pre-training compact models. *arXiv preprint arXiv:1908.08962*.

Svitlana Vakulenko, Shayne Longpre, Zhucheng Tu, and Raviteja Anantha. 2021. Question rewriting for conversational question answering. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pages 355–363.

Ellen M Voorhees and Dawn M Tice. 2000. Building a question answering test collection. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 200–207.

Ellen M Voorhees et al. 1999. The trec-8 question answering track report. In *Trec*, volume 99, pages 77–82. Citeseer.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*.

Tongzhou Wang and Phillip Isola. 2020. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pages 9929–9939. PMLR.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*, 39(2):165–210.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. https://doi.org/10.18653/v1/2020.emnlp-demos.6 Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Yu Yan, Fei Hu, Jiusheng Chen, Nikhil Bhendawade, Ting Ye, Yeyun Gong, Nan Duan, Desheng Cui, Bingyu Chi, and Ruofei Zhang. 2021a. Fastseq: Make sequence generation faster. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 218–226.

Yuanmeng Yan, Rumei Li, Sirui Wang, Fuzheng Zhang, Wei Wu, and Weiran Xu. 2021b. Consert: A contrastive framework for self-supervised sentence representation transfer. *arXiv preprint arXiv:2105.11741*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5754–5764.

Ziyi Yang, Yinfei Yang, Daniel Cer, Jax Law, and Eric Darve. 2020. Universal sentence representation learning with conditional masked language model. *arXiv preprint arXiv:2012.14388*.

HongChien Yu, Chenyan Xiong, and Jamie Callan. 2021. https://doi.org/10.1145/3459637.3482124 Improving query representations for dense retrieval with pseudo relevance feedback. In *Proceedings of the 30th ACM International Conference on Information &; Knowledge Management*, CIKM '21, page 3592–3596, New York, NY, USA. Association for Computing Machinery.

Shi Yu, Jiahua Liu, Jingqin Yang, Chenyan Xiong, Paul Bennett, Jianfeng Gao, and Zhiyuan Liu. 2020. https://doi.org/10.1145/3397271.3401323 Few-shot generative conversational query rewriting. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, page 1933–1936, New York, NY, USA. Association for Computing Machinery.

Yan Zhang, Ruidan He, Zuozhu Liu, Kwan Hui Lim, and Lidong Bing. 2020. An unsupervised sentence embedding method by mutual information maximization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1601–1610.

Peilin Zhou, Jingqi Gao, Yueqi Xie, Qichen Ye, Yining Hua, Jaeboum Kim, Shoujin Wang, and Sunghun Kim. 2023. Equivariant contrastive learning for sequential recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems*, pages 129–140.