Generate to Understand for Representation

1st Changshang Xue*
Baidu Inc.
laohur@gmail.com

2nd Xiande Zhong
Baidu Inc.
zhongxiande@baidu.com

3rd Xiaoqing Liu
Baidu Inc.
liuxiaoqing02@baidu.com

***Abstract*— In recent years, a significant number of high-quality pretrained models have emerged, greatly impacting Natural Language Understanding (NLU), Natural Language Generation (NLG), and Text Representation tasks. Traditionally, these models are pretrained on custom domain corpora and finetuned for specific tasks, resulting in high costs related to GPU usage and labor. Unfortunately, recent trends in language modeling have shifted towards enhancing performance through scaling, further exacerbating the associated costs.**

Introducing GUR: a pretraining framework that combines language modeling and contrastive learning objectives in a single training step. We select similar text pairs based on their Longest Common Substring (LCS) from raw unlabeled documents and train the model using masked language modeling and unsupervised contrastive learning. The resulting model, GUR, achieves impressive results without any labeled training data, outperforming all other pretrained baselines as a retriever at the recall benchmark in a zero-shot setting. Additionally, GUR maintains its language modeling ability, as demonstrated in our ablation experiment. Our code is available at https://github.com/laohur/GUR.

***Keywords*-self-supervised pre-train; contrastive learning; language model; zero-shot learning; text representation; NLP; NLU; NLG; retrieval**

I. Introduction

Pre-training methods that learn directly from raw text have revolutionized NLP and related fields in recent years. Neural networks from the transformer [1] family are trained on large general corpora for self-supervised, task-agnostic objectives, such as autoregressive and masked language modeling. Subsequently, these networks are fine-tuned on a small amount of labeled data for various downstream tasks. This pretraining-finetuning pipeline has significantly enhanced the performance of numerous NLP tasks, including NLU, NLG, and text representation. However, the specific training datasets required for these tasks can be expensive or insufficient. Consequently, the costs of GPUs and labor soar for scaling these models.

Despite numerous efforts to study and improve the efficiency of language model pretraining, the majority of these efforts concentrate on specific aspects of the traditional framework. The development of "text-to-text" [2] as a standardized input-output interface has facilitated task-agnostic architectures for NLU and NLG tasks, although not as swiftly as NLU models. Distilled models [3] transfer essential knowledge from a larger teacher model to a smaller student model, thus accelerating online inference. Research in information retrieval and contrastive learning has demonstrated the potential for better performance in text representation without the need for labeled data.

In this work, we explore alternatives to the standard pretraining-finetuning paradigm, aiming for a more significant improvement in efficiency while reducing resource costs. Our pretraining scheme, which maintains performance levels, encompasses serving NLU, NLG, and text representation tasks in a zero-shot manner. We propose a simple, efficient, and fine-tune-free framework that is capable of understanding, generating, and representing (GUR) text in a zero-shot manner following unsupervised pretraining.

Our approach enhances the language model by incorporating a contrastive learning task. Given a large general corpus, two sentences extracted from a single document that share a sufficiently long common substring are considered relevant sentences. These sentences form a positive pair for the contrastive learning task, along with in-batch negatives. Subsequently, we pretrain the model using a masked language modeling objective in conjunction with an unsupervised contrastive learning objective.

The resulting model, GUR, demonstrates remarkable performance without the need for any labeled training data points. Notably, it significantly surpasses all other pretrained baselines as a retriever in the recall benchmark under a zero-shot setting and competes closely with BM25 [4], a robust sparse baseline. Moreover, GUR retains its language modeling capabilities, as shown in our ablation experiment. These modules can function separately for different online scenarios, offering increased speed. Once pretrained, GUR is capable of zero-shot learning across various contexts.

II. Related Work

A. NLP

Following the release of the Transformer, pretrained models experienced a surge in popularity. GPT [5] and BERT [6] achieved state-of-the-art (SOTA) results in NLG and NLU tasks. A typical NLP pipeline involves pretraining a large model from scratch or initializing it on a public corpus, followed by post-training on a custom domain corpus, and finally fine-tuning downstream tasks (NLU, NLG, and text representation) on labeled datasets. Some research attempts

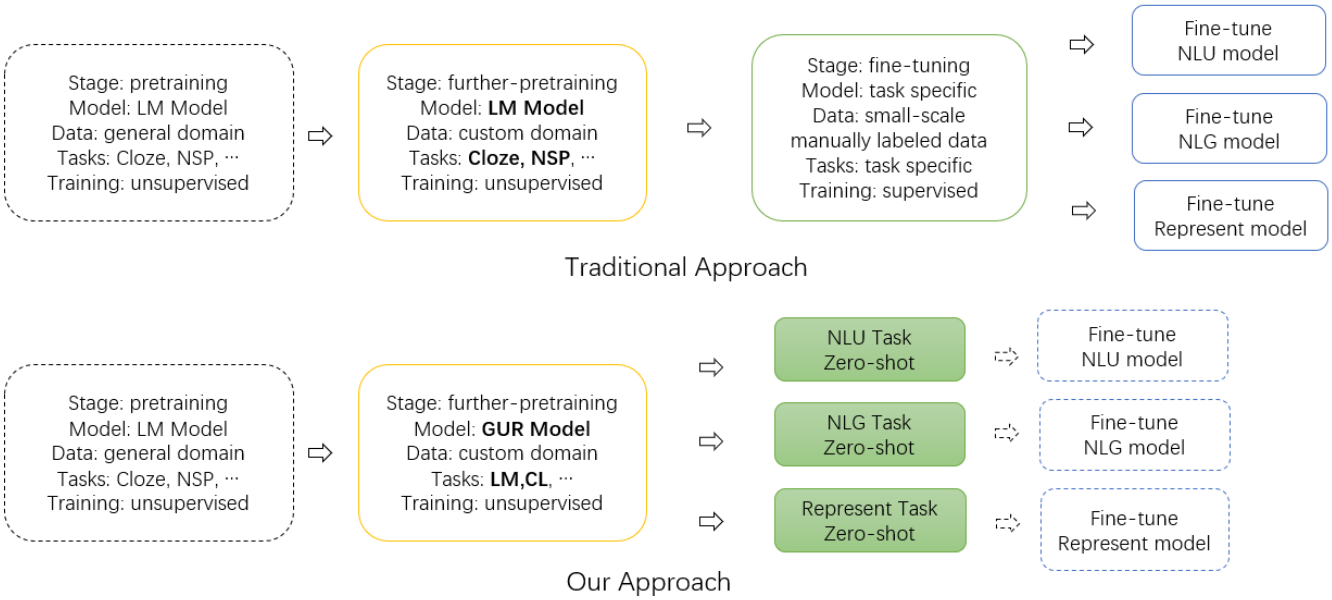

Figure 1. Comparison between the traditional pretraining-finetuning paradigm and our proposed framework GUR: Instead of domain-adaptive further-pretraining using only the LM objective and finetuning on various downstream tasks, we post-pretrain the model with both LM and CL objectives on a custom domain and employ zero-shot learning for NLU, NLG, and recall tasks. The model "GUR-CL" follows the traditional approach without the CL objective. The other models utilize the second approach. Without additional resources, the "GUR-FULL" model maintains the same LM capabilities as the traditional approach model while acquiring the ability for text representation using our approach. All models are initialized from a pretrained LM on a general corpus, minimizing costs.

to simplify this pipeline, with unifying NLP tasks as "text-to-text" yielding notable performance improvements. The primary factor driving these advancements is an increase in model size. Even large models can function as few-shot or zero-shot learners [7].

Certain studies [8] demonstrate that tailoring a pretrained model to the domain of a target task remains beneficial. [9] employs task data as queries to retrieve a small subset of the general corpus and jointly optimizes both the task objective and the language modeling objective from scratch.

Prompt [10] circumvents the need to fine-tune large models by using prompts to leverage pretrained knowledge. One approach trains a large model [11] and subsequently distills it into a smaller model for more efficient inference. Elastic [12] enables early exit for simple samples, thus saving prediction time.

B. Retrieval

Retrieval is typically perceived as a complex system. BM25 [4] serves as a powerful and straightforward sparse retrieval model. Recently, pretraining representation models and interaction models have been incorporated into dense retrieval models [13]. Some research focuses on identifying similar samples, such as the Inverse Cloze Task (ICT) [14], and employs contrastive learning to enhance text representation [15].

C. Representation

The objective of representation learning [16] is to develop an embedding space wherein similar examples are positioned closely together, while dissimilar ones remain distant [17]. In contrastive learning, the learning process is formulated as a classification problem, taking into account both similar and dissimilar candidates.

Contrastive learning can be employed in both supervised and unsupervised settings. In the context of unsupervised data, contrastive learning has emerged as one of the most potent approaches in self-supervised learning. [18] utilizes in-batch negative samples, while [19] and [20] offer text argument techniques. Some research, such as [21] and [22], employs sophisticated methods to capture more accurate semantic similarity between related sentences.

III. Methodology

A. Pretrain Task

We incorporate two tasks during pretraining: LM and contrastive learning. Our total loss, denoted by 1, comprises both LM (Language Modeling) loss and CL (Contrastive Learning) loss, as shown below. The variable α serves to balance the weight between LM Loss and CL Loss.

$$Total\ Loss = LM\ Loss + \alpha\, CL\ Loss \tag{1}$$

In accordance with BERT [6], we employ MLM (Masked Language Modeling) as our LM objective. In addition to

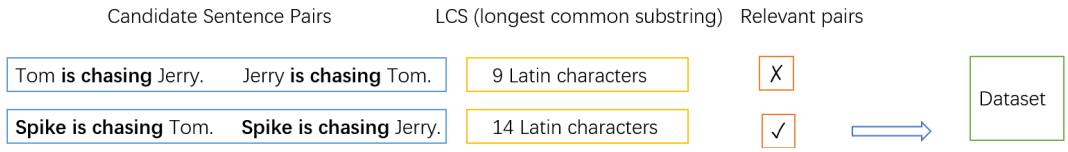

Figure 2. An example of our dataset building approach: For each document, we enumerate sentence pairs as candidates. We then calculate the Longest Common Substring (LCS) for every pair. Pairs with an LCS longer than the threshold are selected as similar texts. We split the article "Tom and Jerry" into sentences and create pairs from every two distinct sentences as candidate similar text pairs. The LCS of "Tom is chasing Jerry." and "Jerry is chasing Tom." is "**is chasing**", which contains only 9 Latin characters and falls short of our threshold. As a result, they are disregarded as a dissimilar pair. The substring "**Spike is chasing**" found in the sentences "Spike is chasing Tom." and "Spike is chasing Jerry." serves as the LCS, which consists of 14 Latin characters and is included in our training dataset as a relevant pair.

the LM task, we also optimize the representation learning objective using a contrastive learning approach. We adopt the same n-pair / InfoNCE [23] loss as CLIP, albeit with a fixed temperature of 0.1. This method is slightly different from Normalized Temperature-scaled Cross-Entropy (NT-Xent) [24] and is adaptive when comparing embeddings from different aspects while consuming less memory.

Given a batch of N (text, text) pairs, GUR is trained to predict which of the $N \times N$ possible (text, text) pairings across a batch actually occurred. Each sample in a batch has one relevant sample and $N-1$ irrelevant samples. To achieve this, GUR learns a text embedding space by jointly training a shared text encoder to maximize the cosine similarity of the text pair embeddings for the N real pairs in the batch, while minimizing the cosine similarity of the embeddings for the $N * (N-1)$ incorrect pairings. We optimize a symmetric cross-entropy loss over these similarity scores. For a sentence i in a positive relevant pair (i, I), the CL Loss is defined as:

$$L_i = -log \frac{\exp\left(sim(h_i, h_I)/\tau\right)}{\sum_{j=1}^{N} \exp\left(sim(h_i, h_j)/\tau\right)} \tag{2}$$

where τ is the temperature of softmax operation as a hyperparameter.

[25] demonstrates that models often target relevance and semantic textual similarity. For instance, "Tom is chasing Jerry" is relevant to "Jerry is chasing Tom", but their semantics are not equivalent. For a general purpose, we define our contrastive learning objective as relevance. It is preferable to obtain a relevance vector directly and measure semantic textual similarity during fine-tuning tasks.

B. Corpus

For pretraining tasks, we exclusively utilize unlabeled data. Few unsupervised methods can compete with BM25. Spider (Span-based unsupervised dense retriever) [15] surpasses BM25 by generating numerous query-document pairs from documents. However, its complexity is not ideal for pretraining tasks and results in excessive text fragmentation. Therefore, we simplify this method during pretraining.

As illustrated in III-A, the dataset is generated using the following steps:

(1) Divide the document into sentences.

(2) Enumerate all unutilized sentence pairs in a document and generate their normalized versions.

(3) Compute the Longest Common Substring (LCS) of the normalized versions using the https://github.com/laohur/SuffixAutomatonSuffixAutomaton; a pair with a sufficiently long LCS is considered similar.

(4) Incorporate similar pairs into the dataset.

(5) If a sentence's length exceeds the maximum length, it will be randomly cropped [26] in the dataloader.

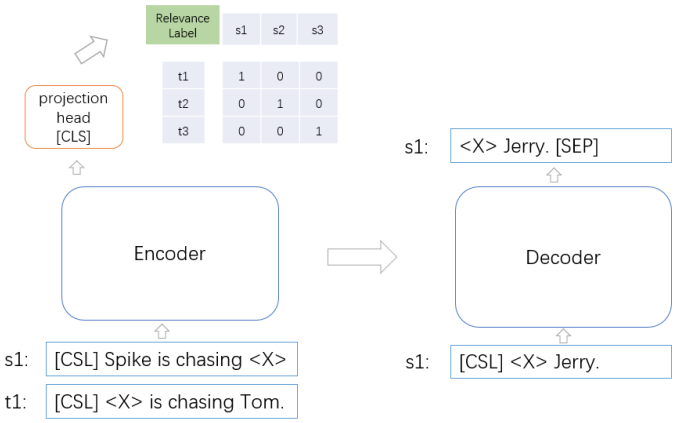

Figure 3. The architecture of our approach. The GUR model, which employs an encoder and a decoder from the Transformer, incorporates a Projection Head to map the sentence embedding to a vector. The related pair $s1$ and $t1$ are flattened in the dataset and subsequently masked as input for T5. The GUR model is designed to jointly train a language model like T5 and predict accurate pairings for a batch of training examples. During testing, the learned text encoder synthesizes a zero-shot classifier by embedding the names of the target dataset's classes.

C. Model Architecture

The scale law suggests that larger models typically exhibit better performance. Our objective is to achieve comparable performance while enhancing training and inference speed. The framework should facilitate Natural Language Understanding (NLU), Natural Language Generation (NLG), and text representation tasks with a zero-overhead abstraction mechanism. Consequently, our framework comprises a Transformer encoder, a Transformer decoder, and a projection head for representing sentences as vectors. These

components can be employed independently for various tasks during both training and inference stages.

Our model, as depicted in III-B, is based on T5 for a shorter decoding length. To enhance inference speed, we refrain from formatting all tasks into text-to-text, as in the original T5 style. Instead, we utilize only the T5 Encoder for processing NLU tasks. Some studies ([28], EncT5 [29]) suggest that the encoder plays a more crucial role in NLU tasks than the decoder. Thus, employing just the encoder in a Transformer model is both sufficient and expedient.

A pre-trained model is adaptable for various downstream tasks, thanks to extensive training datasets. We introduce a projection head above the base encoder to represent sentences as fixed-dimensional vectors. As [24] suggests, the contrastive task is trained using a non-linear projection head. We incorporate a non-linear projection head to represent a sentence as a fixed-dimensional vector. To expedite text representation and avoid conflicts between multiple objectives, unlike BART, the projection head is positioned above the encoder rather than the decoder. This configuration allows the tensor to flow solely through the encoder during inference.

IV. Experiments

A. pre-training

We first pretrain a small-sized model, as demonstrated in I. GUR-Small comprises 8 encoder layers and 8 decoder layers and is initialized using the small-sized model available at https://huggingface.co/IDEA-CCNL/Randeng-T5-Char-57M-MultiTask-ChineseHugging Face. We incorporate a projection head after the encoder to encode sentence representations. The projection head maps the "[CLS]" token representation from the final encoder layer output to a 128-dimensional vector. The weight of the Contrastive Learning (CL) Loss, α, is set to 1, ensuring that the CL Loss and Masked Language Model (MLM) Loss values remain reasonably close.

Table I
Pre-training Setting

	GUR-Small
optimizer	AdamW
lr scheduler	constant with warmup
learning rate	1e-4
masking rate	15%
α	1
temperature	0.1
model dim	512
vector dim	128
encoder layers	8
decoder layers	8
projection token	[CLS]
tokenizer	WordPiece

Owing to resource constraints, we choose to sample a masked sentence span for the model input rather than employing more complex text augmentation methods. We utilize the Pairwise Ranking Objective for model representation instead of the Listwise Ranking Objective.

```
# Hump Geometric Distribution
distribution = [p ** abs(i - mode) for i in range(
    lower, upper + 1)]
distribution = [x / (sum(distribution)) for x in
    distribution]
# keep lower=1, upper=10, mean=3.8 as SpanBERT
# set p=0.66 as a hyper parameter, mode=3 as T5
```

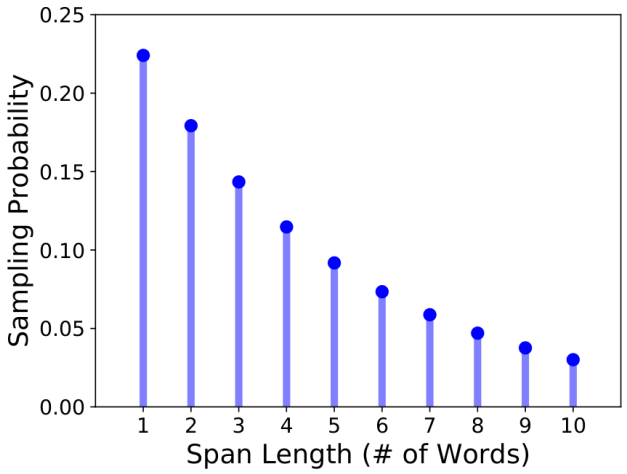

Figure 4. SpanBERT [30] samples random span lengths from a geometric distribution $L \sim \text{Geo}(p = 0.2)$ clipped at $L_{max} = 10$.

The masking rate is set at 15%. SpanBERT [30] employs a geometric distribution for its masking distribution, which is skewed towards shorter spans. However, many of our sentences contain no more than ten words. The default masking strategy of SpanBERT generates numerous one-token mask spans, causing the masking rate to potentially deviate from the target masking rate. To address this issue, we sample our masking distribution from a geometric progression with a peak, tailored for our fragmented sentences. This approach proves to be more robust for short sentences.

$$P(k; p; mode) = Normalize(p^{abs(k-mode)}) \tag{3}$$

In the **Hump Geometric Distribution**, the probability decreases as the distance from the peak increases, as illustrated in 3. This strategy favors generating longer spans compared to the geometric distribution, while maintaining the same expected value.

Our corpus comprises Wikipedia https://dumps.wikimedia.org/dumps acquired using https://github.com/laohur/wiki2txtwiki2txt, CSL [31], and custom documents. Notably, our texts are often of low quality, with some documents containing only a single high-quality sentence in the title amidst the document content. For each document, we sequentially enumerate

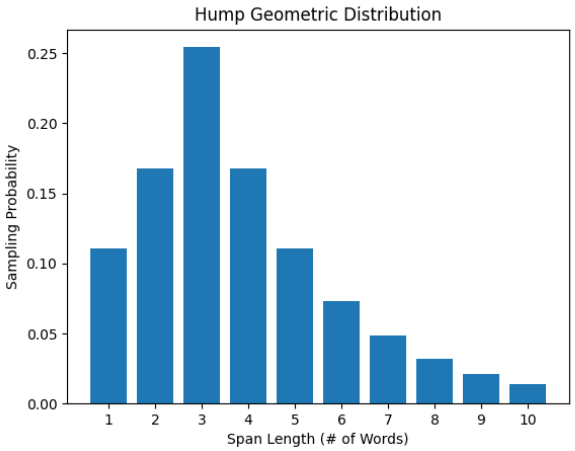

Figure 5. We sample random span lengths from a hump geometric distribution $L \sim$ HumpGeo(p = 0.66, $mode$=3) and clip them at $L_{max} = 10$. This approach is more robust for short sentences.

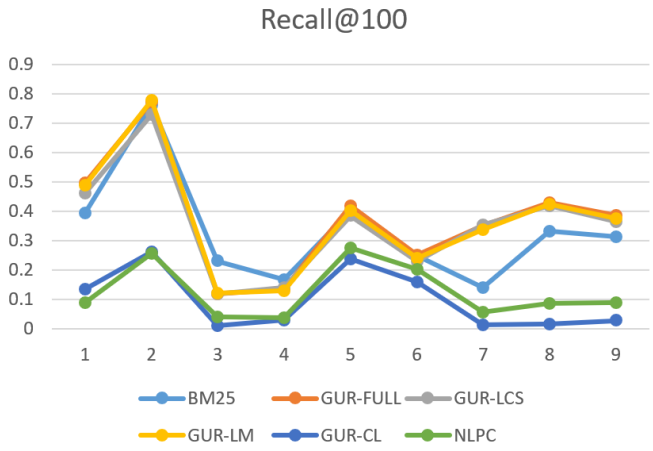

Figure 6. The recall@100 performance of the models on retrieval tasks. The task score curves among the models are nearly parallel. Only BM25 stands out in task3 and underperforms in task7, task8, and task9, due to term hits or mismatches.

every content sentence in relation to the title, forming potential similar text pairs. Ultimately, the large dataset is deduplicated and shuffled using bigsort, accessible at https://github.com/laohur/bigsortGitHub, and read in a stream to minimize resource requirements.

To expedite the pre-training phase and augment the number of training steps, we divide the dataset based on sentence length. Sentences with a length of $<$ 64 (lcs $>=$ 2 Hanzi) are trained using a sequence length of 32. Sentences with a length of $>=$ 64 (lcs $>=$ 3 Hanzi) are trained using a sequence length of 128. The batch size varies according to the sequence length. Additionally, we incorporate a prompt task called "document2title" for the generation task.

For the ablation study, we pre-trained four models under different conditions. The "GUR-FULL" or "GUR" model integrates all the methods mentioned earlier, serving as our foundational model. The "GUR-LCS" model processes candidate pairs without LCS filtering, which is akin to ICT but less random in this experiment. The "GUR-LM" model omits the LM task, while the "GUR-CL" model removes the CL task.

V. Result

A. retrieval

We employ recall@100 and MRR@10 scores in retrieval tasks to evaluate the text representation results. In addition to the "GUR" models, BM25 and NLPC (http://nlpc.baidu.com/platform/demo/wordemb) also participate in these retrieval tasks without any fine-tuning. We simply obtain sentence (dense or sparse) embeddings from the model outputs and use similar candidates as retrieval results. Although not mandatory, we extract the GUR Encoder and Projector to create GurForSequenceRepresentation, which

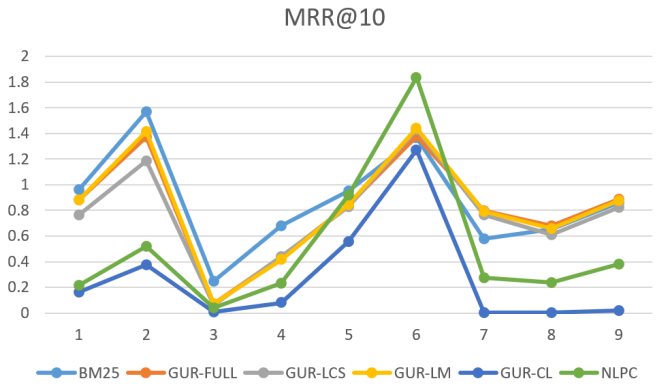

Figure 7. The MRR@10 performance of the models across retrieval tasks. Some tasks are either extremely challenging or easy for all models.

embeds the text of "GUR" models for reduced GPU memory consumption.

Table V-A displays the recall performance across different tasks. The "GUR-FULL," "GUR-LM," and "GUR-LCS" models perform almost identically and outperform the others. In this experiment, our LM and CL tasks exhibit no conflict. Since we generate similar pairs by coupling document titles with sentences in the document content, the datasets used in the "GUR-FULL" and "GUR-LCS" models exhibit slight differences. The "BM25" model excels in task3 but lags in task7, task8, and task9 due to term hits or mismatches. The "GUR-CL" and "NLPC" models obtain the lowest scores because the "GUR-CL" model is trained without the CL task, and the "NLPC" model relies on Word2Vec [32], [33].

We also employ MRR@10 V-A to more accurately measure text representation. The MRR@10 score curves for

these models exhibit minor differences compared to the recall@100 score curves. The difficulty between recall and MRR varies across some tasks. Certain tasks are challenging to recall but easier to rank. The sparse model "BM25" performs best in most tasks due to precise term hits. The "GUR-FULL" and "GUR-CL" models closely follow. The "GUR-LCS" model experiences a slight decline in some tasks. The simpler "NLPC" model achieves the highest score in a few tasks.

In the retrieval benchmark, we observed that the LM task and CL task can be trained concurrently, even for a small-sized model. No single model excels in both recall (GUR) and ranking (BM25) performance.

B. NLU

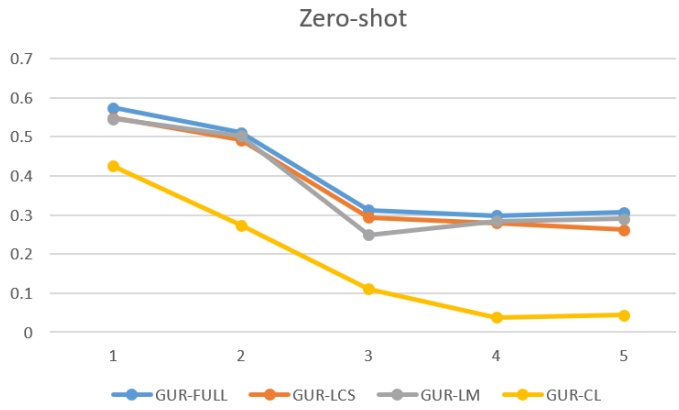

Figure 8. Multi-classification ACC results on zero-shot tasks. Each task comprises tens or hundreds of classes. Most labels consist of one or two terms. The models encode both the samples and the labels into vectors, subsequently searching for the nearest label for each sample.

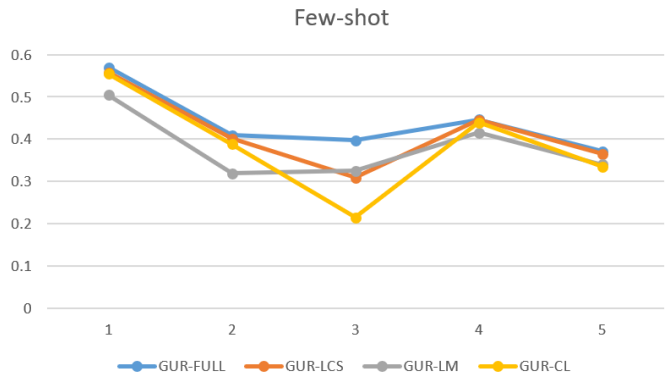

Figure 9. Multi-classification ACC results on few-shot tasks. Each task comprises tens or hundreds of classes. In the training dataset, each class contains 10 samples. All models are trained under the same conditions. The varying scores indicate the models' ability to handle a limited number of samples.

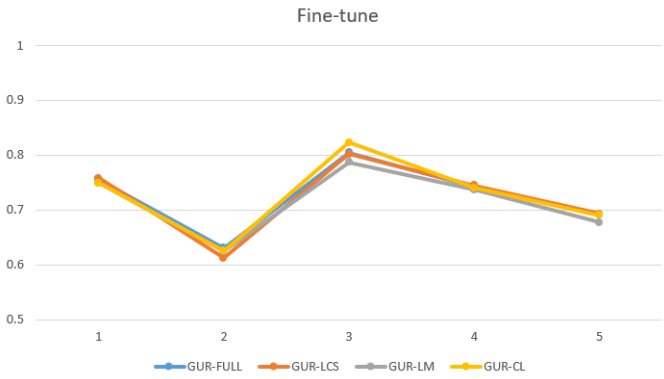

Figure 10. Multi-classification ACC results on fine-tuning tasks. Each task comprises tens or hundreds of classes. In the training dataset, each class contains 100 samples. All models are trained under the same conditions. Nearly all models achieve the same score after sufficient fine-tuning. The scores across the tasks indicate the difficulty level of each task.

We assemble several custom text classification tasks to evaluate the models in zero-shot, few-shot, and fine-tuning scenarios. Each task comprises tens or hundreds of classes.

In each zero-shot task, every sample is associated with a meaningful text label. The models transform queries and labels into vectors and identify the nearest label among hundreds of options for each query in a zero-shot manner. We simply extract the GUR Encoder to create GurForSequenceClassification, akin to BertForSequenceClassification, for classifying queries during the fine-tuning process. As depicted in V-B, the "GUR-FULL" and "GUR-LM" models achieve the best results, with the "GUR-LCS" model closely following. Naturally, the "GUR-CL" model, which lacks CL task pre-training, falls significantly behind. These results demonstrate that the LM and CL tasks can simultaneously contribute to the pre-trained model.

The majority of models achieve comparable scores after undergoing sufficient training in identical environments, as illustrated in Figure V-B. These scores reflect the inherent complexity of the tasks. As the task difficulty varies, so does the gap between fine-tuning and zero-shot performance. Considering a limited number of samples in the training dataset, Figure V-B demonstrates the effectiveness of models in few-shot scenarios. Among all tasks, the "GUR-FULL" model emerges as the most outstanding performer.

C. NLG

We assess the Natural Language Generation (NLG) capabilities of models in a zero-shot query expansion task using two distinct approaches for generating prompts. Initially, custom keywords are employed to retrieve queries, which are then masked, excluding the keywords, to create auto prompts. In contrast, artificial prompts are composed by human writers. These prompts are fed into the models, and the generated outcomes are subsequently analyzed and

Table II

NATURAL LANGUAGE GENERATION RESULTS FOR NLG TASKS: TRANSLATED FROM CHINESE TO ENGLISH. THE MODELS GENERATE THE MASKED SPANS WITHIN THE PROMPTS, REPLACING "<[MASK]>" WITH THE GENERATED SPAN. DUE TO POOR QUALITY AND EXCESSIVE LENGTH, THE "GUR-LM" RESULTS ARE DISREGARDED. MODELS PRETRAINED WITH THE LM TASK EXHIBIT SIMILAR AND FLUENT GENERATED OUTCOMES. BY MASKING RANDOM OPTIONAL WORDS, WE TRANSFORM ARTIFICIAL PROMPTS INTO AUTO PROMPTS. THE RESULTING SENTENCES FROM AUTO PROMPTS DISPLAY GREATER DIVERSITY COMPARED TO THOSE DERIVED FROM THE ARTIFICIAL PROMPTS.

Auto Prompts	**Artificial Prompts**
GUR-FULL	
Chicken Manure <Organic Fertilizer>Separator Principle	What brand of <solid-liquid>separator is of good quality
Ground flat iron anticorrosion <paint>	What is flat iron <angle steel>
Standard height of welding platform <how much>	<Resistance>What is the welding platform?
Dump truck transportation of stone <what is>?	Specific introduction about dump trucks
How much can you drive a long auger in <Line Plus>	Which <straight>auger is easy to use
What is composite PVC sports floor <material>	<Outdoor>What is the sports floor
The front room of the elevator is <whether there>air supply outlet	What is the vent made of
What does the top support of seawater <pt is>mean?	Where is jacking <mainly>used
Vehicle-mounted aerial work platform	Introduce <what is>aerial work platform
<Greenhouse>Can black mulch be used?	<Black>black mulch <choice>what brand is good
GUR-LCS	
Principle of chicken manure <solid-liquid>separator	What brand of <oil-water>separator is of good quality
Ground flat iron anti-corrosion <wood>	What is the flat iron <model number>
Standard height of welding platform <is more>	<Rail>What is the welding platform?
How to transport stone by dump truck <method>?	Specific introduction about dump trucks
How much can a long auger run in <under water>	Which <spiral>auger is easy to use
What is composite PVC sports floor <material>	<Outdoor>What is the sports floor
The front room of the elevator is <whether there>air supply outlet	What is the vent made of
What does the top support of sea water <yes>mean?	Where is jacking <mainly>used
Vehicle-mounted aerial work platform	Introduce <outdoor>aerial work platform
<Vegetable field>Can black mulch be used?	<PE>black mulch <machine selection>what brand is good
GUR-CL	
Principle of chicken manure <dry and wet>separator	What brand of <solid-liquid>separator is of good quality
Ground flat iron anticorrosion <treatment>	What is a flat iron <cross bar>
Standard height of welding platform <how much>	<Car>What is the welding platform?
Self-dumping car to transport stone <does it>?	Specific introduction about dump trucks
How much can a long auger drill run in <Jiangxi>	Which <crawler>auger is easy to use
What is composite PVC sports floor <Material>	<Rubber>What does sports flooring do?
The <what>air supply vent is in the front room of the elevator	What is the vent made of
What does the top support of sea water <yes>mean?	Where is jacking <mainly>used
Vehicle-mounted aerial work platform	Introduce <those>aerial work platforms
<Potatoes>Can black mulch be used?	<wheat>black mulch <material>which brand is good

compared.

Table II displays samples of both auto prompts (left column) and artificial prompts (right column). Upon reviewing the results in an artificial setting, it appears that models pretrained with the LM task exhibit similar performance. The majority of samples are suitable for query expansion, and results derived from artificial prompts demonstrate greater controllability. Interestingly, auto prompts, generated through random masking, perform comparably to artificial prompts while showcasing increased diversity.

VI. CONCLUSION

This study presents an alternative approach to the traditional pretraining-finetuning paradigm, with the goal of significantly enhancing efficiency at a reduced cost. We introduce a straightforward, effective, and finetune-free framework that can understand, generate, and represent (GUR) text in a zero-shot manner following unsupervised pretraining. Our pretraining method relies on instances of similar text pairs selected based on their longest common substring (LCS) from unannotated documents. This approach combines masked language modeling with an unsupervised contrastive learning task. Experimental results demonstrate that GUR achieves comparable performance to pre-trained language models (PLMs) in natural language understanding (NLU) and natural language generation (NLG) tasks while outperforming BM25 in recall tasks as a dense retriever in a zero-shot setting. The model architecture has been tailored for inference optimization across diverse scenarios by leveraging individual modules.

VII. LIMITATION

Due to resource constraints, our study has only been able to conduct limited experiments under specific conditions using a custom corpus. We have tested the framework with verified settings and restricted resources. A more comprehensive investigation would involve pretraining from scratch, benchmarking on general tasks, large-scale evaluations, and exploring more accurate and efficient sentence representations and contrastive approaches. Some of these investigations are currently underway. Our aim is to contribute to a more democratic pretraining style for AI models through this work.

References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[2] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of Machine Learning Research*, vol. 21, no. 140, pp. 1–67, 2020. [Online]. Available: http://jmlr.org/papers/v21/20-074.html

[3] C. Buciluundefined, R. Caruana, and A. Niculescu-Mizil, "Model compression," in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '06. New York, NY, USA: Association for Computing Machinery, 2006, p. 535–541. [Online]. Available: https://doi.org/10.1145/1150402.1150464

[4] S. Robertson and H. Zaragoza, "The probabilistic relevance framework: Bm25 and beyond," *Found. Trends Inf. Retr.*, vol. 3, no. 4, p. 333–389, apr 2009. [Online]. Available: https://doi.org/10.1561/1500000019

[5] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever *et al.*, "Improving language understanding by generative pre-training," 2018.

[6] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "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)*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186. [Online]. Available: https://aclanthology.org/N19-1423

[7] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language Models are Few-Shot Learners," *arXiv e-prints*, p. arXiv:2005.14165, May 2020.

[8] S. Gururangan, A. Marasović, S. Swayamdipta, K. Lo, I. Beltagy, D. Downey, and N. A. Smith, "Don't Stop Pretraining: Adapt Language Models to Domains and Tasks," *arXiv e-prints*, p. arXiv:2004.10964, Apr. 2020.

[9] X. Yao, Y. Zheng, X. Yang, and Z. Yang, "NLP From Scratch Without Large-Scale Pretraining: A Simple and Efficient Framework," *arXiv e-prints*, p. arXiv:2111.04130, Nov. 2021.

[10] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, "Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing," *arXiv e-prints*, p. arXiv:2107.13586, Jul. 2021.

[11] Z. Li, E. Wallace, S. Shen, K. Lin, K. Keutzer, D. Klein, and J. E. Gonzalez, "Train large, then compress: Rethinking model size for efficient training and inference of transformers," in *Proceedings of the 37th International Conference on Machine Learning*, ser. ICML'20. JMLR.org, 2020.

[12] X. Liu, T. Sun, J. He, J. Wu, L. Wu, X. Zhang, H. Jiang, Z. Cao, X. Huang, and X. Qiu, "Towards efficient NLP: A standard evaluation and a strong baseline," in *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Seattle, United States: Association for Computational Linguistics, Jul. 2022, pp. 3288–3303. [Online]. Available: https://aclanthology.org/2022.naacl-main.240

[13] Y. Fan, X. Xie, Y. Cai, J. Chen, X. Ma, X. Li, R. Zhang, and J. Guo, "Pre-training Methods in Information Retrieval," *arXiv e-prints*, p. arXiv:2111.13853, Nov. 2021.

[14] K. Lee, M.-W. Chang, and K. Toutanova, "Latent retrieval for weakly supervised open domain question answering," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 6086–6096. [Online]. Available: https://aclanthology.org/P19-1612

[15] O. Ram, G. Shachaf, O. Levy, J. Berant, and A. Globerson, "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*. Seattle, United States: Association for Computational Linguistics, Jul. 2022, pp. 2687–2700. [Online]. Available: https://aclanthology.org/2022.naacl-main.193

[16] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[17] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 2, 2006, pp. 1735–1742.

[18] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning Transferable Visual Models From Natural Language Supervision," *arXiv e-prints*, p. arXiv:2103.00020, Feb. 2021.

[19] Y. Yan, R. Li, S. Wang, F. Zhang, W. Wu, and W. Xu, "ConSERT: A contrastive framework for self-supervised sentence representation transfer," 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)*. Online: Association for Computational Linguistics, Aug. 2021, pp. 5065–5075. [Online]. Available: https://aclanthology.org/2021.acl-long.393

[20] T. Gao, X. Yao, and D. Chen, "SimCSE: Simple contrastive learning of sentence embeddings," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language*

Processing. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 6894–6910. [Online]. Available: https://aclanthology.org/2021.emnlp-main.552

[21] Y. Seonwoo, G. Wang, C. Seo, S. Choudhary, J. Li, X. Li, P. Xu, S. Park, and A. Oh, "Ranking-enhanced unsupervised sentence representation learning," 2023. [Online]. Available: https://openreview.net/forum?id=g77JafrHWyy

[22] J. Liu, J. Liu, Q. Wang, J. Wang, W. Wu, D. Zhao, and R. Yan, "RankCSE: Unsupervised sentence representations learning via learning to rank," 2023. [Online]. Available: https://openreview.net/forum?id=y_sZyxuuFh3

[23] A. van den Oord, Y. Li, and O. Vinyals, "Representation Learning with Contrastive Predictive Coding," *arXiv e-prints*, p. arXiv:1807.03748, Jul. 2018.

[24] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," *arXiv preprint arXiv:2002.05709*, 2020.

[25] A. Neelakantan, T. Xu, R. Puri, A. Radford, J. M. Han, J. Tworek, Q. Yuan, N. Tezak, J. W. Kim, C. Hallacy, J. Heidecke, P. Shyam, B. Power, T. Eloundou Nekoul, G. Sastry, G. Krueger, D. Schnurr, F. Petroski Such, K. Hsu, M. Thompson, T. Khan, T. Sherbakov, J. Jang, P. Welinder, and L. Weng, "Text and Code Embeddings by Contrastive Pre-Training," *arXiv e-prints*, p. arXiv:2201.10005, Jan. 2022.

[26] G. Izacard, M. Caron, L. Hosseini, S. Riedel, P. Bojanowski, A. Joulin, and E. Grave, "Unsupervised dense information retrieval with contrastive learning," *Transactions on Machine Learning Research*, 2022. [Online]. Available: https://openreview.net/forum?id=jKN1pXi7b0

[27] Y. Tay, M. Dehghani, J. Rao, W. Fedus, S. Abnar, H. W. Chung, S. Narang, D. Yogatama, A. Vaswani, and D. Metzler, "Scale efficiently: Insights from pretraining and finetuning transformers," in *International Conference on Learning Representations*, 2022. [Online]. Available: https://openreview.net/forum?id=f2OYVDyfIB

[28] Y. Shao, Z. Geng, Y. Liu, J. Dai, H. Yan, F. Yang, L. Zhe, H. Bao, and X. Qiu, "CPT: A Pre-Trained Unbalanced Transformer for Both Chinese Language Understanding and Generation," *arXiv e-prints*, p. arXiv:2109.05729, Sep. 2021.

[29] F. Liu, T. Huang, S. Lyu, S. Shakeri, H. Yu, and J. Li, "EncT5: A Framework for Fine-tuning T5 as Non-autoregressive Models," *arXiv e-prints*, p. arXiv:2110.08426, Oct. 2021.

[30] M. Joshi, D. Chen, Y. Liu, D. S. Weld, L. Zettlemoyer, and O. Levy, "SpanBERT: Improving pre-training by representing and predicting spans," *arXiv preprint arXiv:1907.10529*, 2019.

[31] Y. Li, Y. Zhang, Z. Zhao, L. Shen, W. Liu, W. Mao, and H. Zhang, "CSL: A large-scale Chinese scientific literature dataset," in *Proceedings of the 29th International Conference on Computational Linguistics*. Gyeongju, Republic of Korea: International Committee on Computational Linguistics, Oct. 2022, pp. 3917–3923. [Online]. Available: https://aclanthology.org/2022.coling-1.344

[32] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," *arXiv e-prints*, p. arXiv:1301.3781, Jan. 2013.

[33] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," *arXiv e-prints*, p. arXiv:1310.4546, Oct. 2013.