# LM-BFF-MS: Improving Few-Shot Fine-tuning of Language Models based on Multiple Soft Demonstration Memory

**Eunhwan Park**[†], **Donghyeon Jeon**[‡], **Seonhoon Kim**[‡],
**Inho Kang**[‡], **Seung-Hoon Na**[†*]
[†]Jeonbuk National University, [‡]NAVER Corporation
{judepark, nash}@jbnu.ac.kr
{donghyeon.jeon, seonhoon.kim, once.ihkang}@navercorp.com

## Abstract

LM-BFF (Gao et al., 2021) achieves significant few-shot performance by using auto-generated prompts and adding demonstrations similar to an input example. To improve the approach of LM-BFF, this paper proposes **LM-BFF-MS**—**b**etter **f**ew-shot **f**ine-tuning of **l**anguage **m**odels with **m**ultiple **s**oft demonstrations by making its further extensions, which include 1) prompts with *multiple demonstrations* based on automatic generation of multiple label words; and 2) *soft demonstration memory* which consists of multiple sequences of *globally shared* word embeddings for a similar context. Experiments conducted on eight NLP tasks show that LM-BFF-MS leads to improvements over LM-BFF on five tasks, particularly achieving 94.0 and 90.4 on SST-2 and MRPC, respectively[1].

## 1 Introduction

The GPT-3 model (Brown et al., 2020) has achieved remarkable few-shot performance on natural language understanding tasks given a *natural language prompt* and $|K|$ labeled samples as *demonstrations* in the inputs without updating the model's weights. However, the GPT-3 model consists of 175B parameters, making it challenging to perform task-specific fine-tuning, which is often required in real-world applications.

To enable task-specific fine-tuning, *prompt-based few-shot fine-tuning* has been widely studied to encourage the few-shot capabilities of pre-trained language models (PLMs) equipped with label-specific *verbalizers* and *prompts* that are compatible with language models (Schick and Schütze, 2021a,b). Prompt-based fine-tuning reformulates downstream tasks as a masked language modeling problem, where a token (*label word*) is generated on a given prompt with a task-specific *template*. However, constructing optimal prompts requires domain expertise and the use of manual prompts can be suboptimal (Webson and Pavlick, 2021; Lu et al., 2021; Zhao et al., 2021).

Among the various methods of prompt-based fine-tuning, this study is based on the LM-BFF method (Gao et al., 2021), which uses a demonstration-aware prompt where a demonstration is produced by unmasking the example prompt in contexts similar to the input, inspired by the findings from the GPT-3 model (Brown et al., 2020). With demonstration-aware prompts, the LM-BFF outperforms the conventional fine-tuning approach and GPT-3's in-context learning. To improve LM-BFF, we propose **LM-BFF-MS**, **b**etter **f**ew-shot **f**ine-tuning of **l**anguage **m**odels with **m**ultiple **s**oft demonstration memory, based on the following two extensions:

1. **Prompts with multiple demonstrations**. While LM-BFF uses single demonstration[2], our model uses *multiple* demonstrations with different *label phrases*, where each demonstration is constructed per label phrase. Given that label phrases are semantically related or similar, it is expected that resulting demonstrations indirectly augment the vocabulary of the verbalizer with label phrases[3].

2. **Soft demonstration memory based on multiple sequences of word embeddings.** Unlike LM-BFF, which directly uses a sequence of hard tokens in the demonstration, inspired by the *soft prompts* of Lester et al. (2021), we replace them with a sequence of soft vectors as a proper context for each label phrase, where soft vectors are *globally shared soft examples* for each label phrase but are not sensitive to

[*]Corresponding author
[1]Our implementation is publicly available at https://github.com/judepark96/LM-BFF-MS
[2]Note that LM-BFF also explored sampling multiple demonstrations per label, but did not observe any improvement.
[3]Here, it is assumed that the set of label words is different from the set of label phrases.



*Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*
*Volume 2: Short Papers*, pages 310 - 317

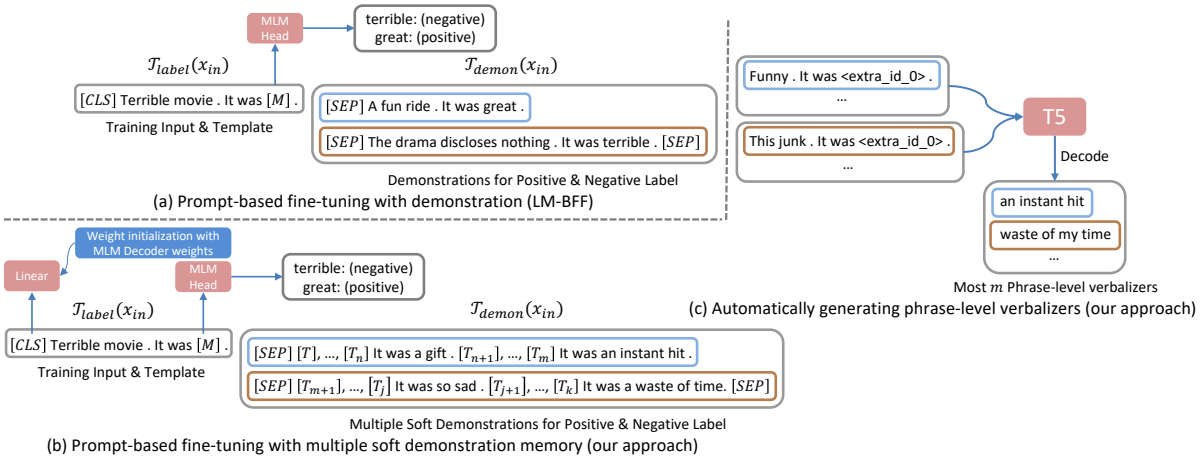



Figure 1: An illustration of (a) the prompt of LM-BFF (Prompt-based fine-tuning with demonstration), comparing to that of (b) our proposed LM-BFF-MS (Prompt-based fine-tuning with Multiple soft demonstration memory) in Section 3. The subfigure (c) shows the span-corrupted input and output of T5 used for automatic generation of phrase-level verbalizers as in Section 3.2. $[M]$, $[T_k]$, <extra_id_0>, blue and brown colored square box referred as mask and soft token, sentinel token of T5, positive, negative, respectively.

an input context. In our approach, soft vectors are considered as *automatically* generated demonstration that matches well for each label phrase, capturing the common context for the corresponding phrase. To train the soft demonstration memory effectively, we further introduce an auxiliary task, named *next demonstrations prediction* (NDP) task, inspired by NSP-BERT (Sun et al., 2021).

Following the previous setting of the LM-BFF, the experimental results on eight NLP datasets show that the proposed LM-BFF-MS leads to a better and more stable few-shot performance compared to the previous models. The contributions of this study are summarized as follows:

- We propose prompts with multiple soft demonstration memory based on the automatic generation of multiple label phrases and the use of soft demonstration memory that is armed with an auxiliary NDP task.
- We present promising results of the proposed method on eight NLP tasks by showing improved results on some datasets, particularly achieving state-of-the-art performance on SST-2 and MRPC.

## 2 Related Work

*Prompt-based few-shot fine-tuning*, which finetunes based on few-shot examples under a prompting setting, has been widely studied for moderately sized PLMs such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019). For example, PET reformulates downstream tasks as a masked language modeling problem and performs gradient-based fine-tuning (Schick and Schütze, 2021a,b). AutoPrompt creates appropriate prompts for a set of discrete tokens using a gradient-guided search (Shin et al., 2020). *Null Prompts*—simple concatenations of the inputs and `[MASK]` token—achieve a free of prompt engineering (Logan et al., 2021). Instead of using hard prompts, there have also been works of using continuous vectors of prompt tokens, called *soft prompting*[4], including the work of Lester et al. (2021), which proposes *soft prompts* composed of learnable continuous embeddings while freezing the weight of PLMs; and Gu et al. (2021) proposes pre-training prompts by adding soft prompts into the pre-training stage to obtain a better initialization. The *demonstration-aware prompt* has also been explored by (Gao et al., 2021) with their proposed LM-BFF, where a demonstration is constructed by unmasking the masked prompt on a similar input example.

Unlike LM-BFF, which uses a single demonstration per label, our work uses '*multiple*' demonstrations that are provided for automatically generated label phrases. In addition, inspired by the method of soft prompting, we use '*soft*' demonstration memory based on globally shared soft vectors for prompt tokens, without using hard tokens of the similar context.

[4]Here, the soft prompting method refers to the methods of using unknown prompt-specific token embedding or hidden representations at prompt positions.

| Model | SST-2 (acc) | MR (acc) | Subj (acc) | MRPC (F1) |
|---|---|---|---|---|
| Majority† | 50.9 | 50.0 | 50.0 | 81.2 |
| Prompt-based zero-shot‡ | 83.6 | 80.8 | 51.4 | 61.9 |
| "GPT-3" in-context learning | 84.8 (1.3) | 80.5 (1.7) | 53.6 (0.8) | 45.7 (6.0) |
| Fine-tuning | 81.4 (3.8) | 76.9 (5.9) | 90.8 (1.8) | 76.6 (2.5) |
| LM-BFF (man) + demonstration | 92.6 (0.5) | 86.6 (2.2) | 92.3 (0.8) | 77.8 (2.0) |
| DART | 93.5 (0.5) | 88.2 (1.0) | 90.7 (1.4) | 78.3 (4.5) |
| LM-BFF-MS | **94.0 (0.3)** | **88.3 (0.5)** | **92.7 (0.3)** | **80.4 (1.3)** |
| Fine-tuning (full)† | *95.0* | *90.8* | *97.0* | *91.4* |
| **Model** | **MNLI** (acc) | **SNLI** (acc) | **CR** (acc) | **MPQA** (F1) |
| Majority† | 32.7 | 33.8 | 50.0 | 50.0 |
| Prompt-based zero-shot‡ | 50.8 | 49.5 | 79.5 | 67.6 |
| "GPT-3" in-context learning | 52.0 (0.7) | 47.1 (0.6) | 87.4 (0.8) | 63.8 (2.1) |
| Fine-tuning | 45.8 (6.4) | 48.4 (4.8) | 75.8 (3.2) | 72.0 (3.8) |
| LM-BFF (man) + demonstration | **70.7 (1.3)** | **79.7 (1.5)** | 90.2 (1.2) | 87.0 (1.1) |
| DART | 67.5 (2.6) | 75.8 (1.6) | **91.8 (0.5)** | - |
| LM-BFF-MS | 68.2 (3.3) | 73.9 (3.0) | 90.8 (1.7) | **87.9 (0.4)** |
| Fine-tuning (full)† | *89.8* | *92.6* | *89.4* | *87.8* |

Table 1: The main results with RoBERTa-large. †: the full training set is used. ‡: no training examples are used. Otherwise, we use $K = 16$ (# examples per class). The mean (and standard deviation) performance over five different splits is reported. Majority: majority class "GPT-3" in-context learning: using the in-context learning proposed in (Brown et al., 2020) with RoBERTa-large (no parameter updates); man: manual prompt; LM-BFF & DART: the performance in (Gao et al., 2021; Zhang et al., 2021) is reported. full: fine-tuning using full training set.

## 3 Fine-tuning with Multiple Soft Demonstration Memory

### 3.1 Background

Following on from (Gao et al., 2021), suppose that the input sentences $x_{in} = x_1$ and $x_{in} = (x_1, x_2)$ are presented for single-sentence and sentence-pair tasks, respectively. The template $\mathcal{T}$ is defined as $(\mathcal{T}_{label}, \mathcal{T}_{demon})$, where $\mathcal{T}_{label}$ is the template used to generate the *main* prompt for input $x_{in}$ and $\mathcal{T}_{demon}$ is the additional template to generate demonstrations of input $x_{in}$. For example, $\mathcal{T}_{label}(x_{in})$ for a single sentence task is given as:

$$\mathcal{T}_{label}(x_{in}) = \texttt{[CLS]}\ x_1\ \text{It was}\ \texttt{[MASK]}\ .\ \texttt{[SEP]}$$

We use $\mathcal{T}_{label}$ with the manually designed templates of (Gao et al., 2021)[5].

To define $\mathcal{T}_{demon}$, suppose that $\mathcal{V}$ and $\mathcal{Y}$ are the vocabulary and label space, respectively. Let $\mathcal{M}_{wo}\colon \mathcal{Y} \to \mathcal{V}$ and $\mathcal{M}_{ph}^{(1)}, \cdots, \mathcal{M}_{ph}^{(m)}\colon \mathcal{Y} \to \mathcal{V}^*$ be mapping functions that convert a label into individual words and phrases, called *word*-level and *phrase*-level functions, respectively. For example, $\mathcal{M}_{wo}(pos)$ = "great", $\mathcal{M}_{wo}(neg)$ = "terrible", $\mathcal{M}_{ph}(pos)$ = "a gift", $\mathcal{M}_{ph}(neg)$ = "a total waste of my time". Let $\mathfrak{M}$ be the set of $m$ phrase-level mapping functions, that is, $\mathfrak{M} = \{\mathcal{M}_{ph}^{(1)}, \cdots, \mathcal{M}_{ph}^{(m)}\}$. Given $\mathcal{M}_{ph} \in \mathfrak{M}$, $\tilde{\mathcal{T}}_{demon}(x_{in}, y, \mathcal{M}_{ph})$ is defined as the *unmasked* sequence of $\mathcal{T}_{label}(x_{in})$ by placing $\mathcal{M}_{ph}(y)$ instead of the `[MASK]` token; $\tilde{\mathcal{T}}_{demon}(x_{in}, y, \mathcal{M}_{ph})$ is obtained by first applying $\mathcal{T}_{label}$ to $x_{in}$ to produce $\mathcal{T}_{label}(x_{in})$ and then replacing `[MASK]` with $\mathcal{M}_{ph}(y)$. For example, given $x_{in} = x_1, y = neg, \mathcal{M}_{ph}(neg)$ = "so sad", $\tilde{\mathcal{T}}_{demon}(x_{in}, y, \mathcal{M}_{ph})$ is then obtained as: "$x_1$ It was so sad . `[SEP]`".

Now, suppose that $\mathcal{N}(x_{in}, y)$ is the set of training examples similar to $x_{in}$ labeled with $y$. Then $\mathcal{T}_{demon}$ is defined as follows:

$$\mathcal{T}_{demon}(x_{in}) = \bigoplus_{\substack{\tilde{\mathcal{M}}_{ph} \in \mathfrak{M}, \\ x \in \mathcal{N}(x_{in}, y), \\ y \in |\mathcal{Y}|}} \tilde{\mathcal{T}}_{demon}(x, y, \mathcal{M}_{ph}) \quad (1)$$

where $\oplus$ denotes the concatenation operator.

Finally, $x_{in}$ is converted to its prompted version $x_{prompt} = \mathcal{T}_{label}(x_{in}) \oplus \mathcal{T}_{demon}(x_{in})$, which is used as the input for the prompt-based few-shot fine-tuning.

Note that this setting includes the LM-BFF as a specific case with $|\mathcal{N}(x_{in}, y)| = 1$ per label and $\mathfrak{M} = \{\mathcal{M}_{wo}\}$ in Eq. (1), which refers to a *single* demonstration setting.

[5]We use Table 1 of (Gao et al., 2021) for $\mathcal{T}_{label}$ as described in Table 5. Note that we do not use auto-generated templates and label words.

### 3.2 Automatically Generating Phrase-Level Verbalizers

In contrast to the LM-BFF, we employ *phrase*-level mapping function, as it is theorized that it would enable a better representation of the demonstration than a word-level mapping function. The remaining part describes how to obtain the $m$ phrase-level mapping functions in Eq. (1), $\mathcal{M}_{ph}^{(j)} \in \mathfrak{M}$.

To this end, we use T5 to generate label phrases using a properly designed span-corrupted input in the reverse manner of (Gao et al., 2021) which exploits T5 to automatically generate templates. The input for T5's encoder is merely the prompted sequence $\mathcal{T}_{label}(x_{in})$, however, with `[MASK]` as the span-corrupted token, the decoder then fills in the placeholders, removes duplicated results, and chooses the top $m$ most likely generated sequences for phrase-level mapping functions of the corresponding label as described in Figure 1 (c). Our generation results are shown in Table 4.

### 3.3 Soft Demonstration Memory

Different from LM-BFF which explicitly finds similar training examples $\mathcal{N}(x_{in}, y)$, 'soft demonstration memory' is used, which consists of *globally shared soft examples* as demonstrations, assuming that each demonstration uses $n$ *soft* tokens for a sentence as $[T_1] \cdots [T_n]$. Under a soft demonstration memory, $\mathcal{T}_{demon}(x_{in})$ is obtained using Eq. (1), but using the following definition of $\mathcal{N}(x_{in}, y)$[6]:

$$\mathcal{N}(x_{in}, y) = \left\{ [T_1^{(k)}] \cdots [T_n^{(k)}] \right\}_{k=1}^{m}$$

In single-sentence tasks, the soft demonstration memory maintains a total of $m \cdot |\mathcal{Y}|$ sentences each of which consists of $n$ soft tokens, where the set of $m$ sentences corresponds to each label. For example, when $n = 10$, $m = 5$, and $|\mathcal{Y}| = 2$, the total size of the global memory is 100.

To create $m$ demonstrations, all examples of global memory are chosen without requiring a sample of similar examples. An illustration of the incorporated soft demonstration memory is described shown in Figure 1 (b).

### 3.4 Next Demonstration Prediction Task

To obtain a better representation of soft demonstration memory, we introduce the NDP task, which predicts whether positive (or negative) examples in $\mathcal{T}_{demon}(x_{in})$ are correctly matched with a positive (or negative) label word for the prompted input $\mathcal{T}_{label}(x_{in})$.

To be more specific, the NDP task trains $P_{\text{NDP}}(y|x_{prompt}) = \texttt{softmax}(W_{\texttt{[MLM]}} h_{\texttt{[CLS]}} + b)$, where $W_{\texttt{[MLM]}} \in \mathbb{R}^{|\mathcal{Y}| \times d}$ are the output embedding weights of the label words in an MLM decoder. Finally, given a few-shot example $(x_{in}, y)$, the training objective is defined as:

$$\mathcal{L} = \text{CE}\left(P(\texttt{[MASK]} = \mathcal{M}_{wo}(y)|x_{prompt})\right) + \lambda \cdot \text{CE}\left(P_{\text{NDP}}(y|x_{prompt})\right)$$

where CE is the cross-entropy loss function and $\lambda$ is the hyper-parameter. Section 4.3 presents the effect of using the NDP loss compared to that without it.

[6]That is, the soft demonstration memory is a set of $|\mathcal{Y}|$ global memories each of which consists of $m$ demonstrations.

| Dataset | Model | K=16 |
|---|---|---|
| **SST-2** | Soft Prompting | 92.7 (0.5) |
| | LM-BFF-MS | **94.0 (0.3)** |

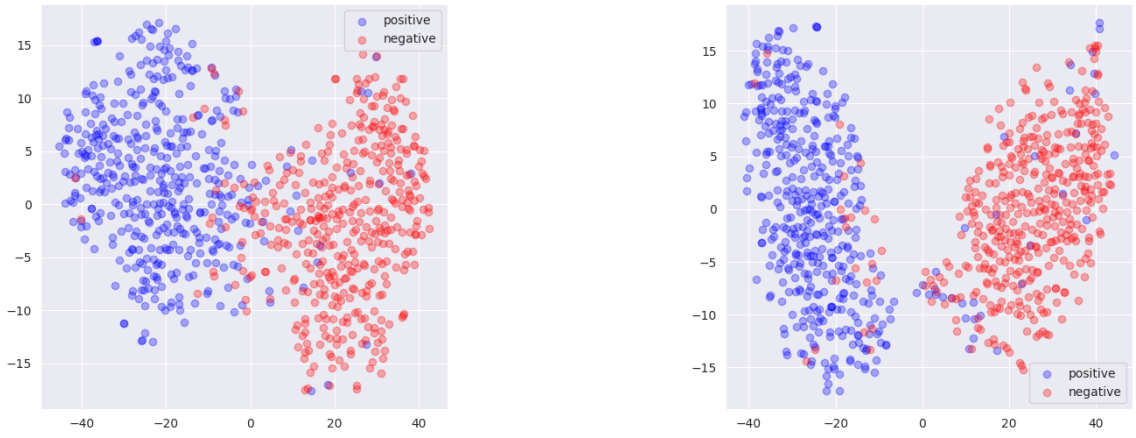
Table 2: Few-shot performance comparison with *soft prompting* (Lester et al., 2021) and our approach.

## 4 Experiments

The implementation details are provided in Appendix B. For a fair comparison, the same manual prompts for $\mathcal{T}_{label}$ in LM-BFF and LM-BFF-MS are used.

### 4.1 Main Results

As shown in Table 1, it is noticed that the proposed approach achieves a better and stable few-shot performance than the prior methods and the LM-BFF on five tasks. In particular, LM-BFF-MS achieves state-of-the-art performance on SST-2 and MRPC tasks, with 94.0 and 80.4, respectively. Moreover, it is observed that the performance variation of LM-BFF-MS are mostly lower than that of the prior methods except for the MNLI, SNLI, and CR tasks, implying that our approach is more stable than the existing models. On the other hand, LM-BFF-MS is weaker than LM-BFF on SNLI, although it shows comparable results to DART. We believe that the effect of global demonstration memory is task sensitive, suggesting that the *local* method of sampling similar demonstrations as in LM-BFF often needs to be employed for some tasks or specific input sentences.




(a) [MASK] w/o NDP Task

(b) [MASK] w/ NDP Task

Figure 2: A visualization of representation of [MASK] tokens.

### 4.2 Soft Demonstration Memory vs. Soft Prompting

To validate the use of soft demonstration memory, instead of inserting soft vectors into the demonstration parts. We further evaluate the soft prompting of (Lester et al., 2021) by prepending $p$ soft vectors to the main template $\mathcal{T}_{label}(x_{in})$, where $p$ is the length of the additional soft prompt[7].

Table 2 compares soft prompting with LM-BFF-MS on SST-2 and shows that LM-BFF-MS outperforms soft prompting under the setting of the same length of soft token. The results confirm that the gain of soft demonstration memory is not merely obtained by using additional parameters of soft vectors, but by effectively modeling the demonstration-aware context.

### 4.3 The Effect of Using Next Demonstration Prediction Task

| Method | SST-2 (acc) |
|---|---|
| LM-BFF (man) + demonstration | 92.6 (0.5) |
| DART | 93.5 (0.5) |
| LM-BFF-MS | **94.0 (0.3)** |
| −next demonstration prediction | 93.4 (0.4) |

Table 3: Ablation study for NDP on SST-2 dataset.

To examine whether the use of the NDP task is indeed effective in LM-BFF-MS, Table 3 compares results of LM-BFF-MS with and without the NDP task on the SST-2 dataset. As shown in Table 3, LM-BFF-MS with NDP loss shows improved performance compared to that without NDP loss, providing positive evidence for our motivating hypothesis that the use of the NDP loss is helpful in enhancing the representation of soft demonstration memory.

To further analyze the effect of auxiliary NDP task, Figure 2 visualizes the representation of [MASK] tokens on SST-2 using $t$-SNE (van der Maaten and Hinton, 2008) compared, with and without the NDP task. As shown in Figure 2a and 2b, the representation learned using the NDP task is more discriminative than that learned without the NDP task, suggesting that the NDP task provides an effective additional loss for learning representations of soft demonstration memory.

## 5 Conclusion

This study proposed LM-BFF-MS—manual prompts with *multiple soft demonstration memory* based on the automatic generation of multiple label words and an auxiliary NDP task. Experiments showed that the proposed method outperforms prior works on five tasks: SST-2, MR, Subj, MRPC, and MPQA. Extending our work to a large soft demonstration memory and a combination of local and global memory is valuable for future investigations.

## Acknowledgements

We would like to thank all anonymous reviewers for their valuable comments and suggestions.

[7]Here, $p$ is fixed to be the same as the number of tokens used in the multiple demonstrations of the LM-BFF-MS.

## References


Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 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, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. 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 3816–3830, Online. Association for Computational Linguistics.

Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. 2021. PPT: pre-trained prompt tuning for few-shot learning. *CoRR*, abs/2109.04332.

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*, KDD '04, page 168–177, New York, NY, USA. Association for Computing Machinery.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *CoRR*, abs/2104.08691.

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

Robert Logan, Ivana Balažević, Eric Wallace, Fabio Petroni, Sameer Singh, and Sebastian Riedel. 2021. Cutting down on prompts and parameters: Simple few-shot learning with language models.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2021. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. *CoRR*, abs/2104.08786.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 271–278, Barcelona, Spain.

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, Ann Arbor, Michigan. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Timo Schick and Hinrich Schütze. 2021a. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021b. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2339–2352, Online. Association for Computational Linguistics.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew 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, Seattle, Washington, USA. Association for Computational Linguistics.

Yi Sun, Yu Zheng, Chao Hao, and Hangping Qiu. 2021. NSP-BERT: A prompt-based zero-shot learner through an original pre-training task-next sentence prediction. *CoRR*, abs/2109.03564.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605.

Albert Webson and Ellie Pavlick. 2021. Do prompt-based models really understand the meaning of their prompts?

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*, 39(2-3).

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi 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 M. Rush. 2020. 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.

Ningyu Zhang, Luoqiu Li, Xiang Chen, Shumin Deng, Zhen Bi, Chuanqi Tan, Fei Huang, and Huajun Chen. 2021. Differentiable prompt makes pre-trained language models better few-shot learners. *CoRR*, abs/2108.13161.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 12697–12706. PMLR.

# A Limitation

The main contribution of this work is the *multiple soft demonstration memory*, however, the number of available demonstrations is bounded by the maximum input length. Furthermore, unlike the GPT-3 model, the maximum input length of PLMs is usually 512, which is not sufficient to deal with more difficult tasks such as SNLI and MNLI. As shown in Table 1, despite the effectiveness of LM-BFF-MS, it shows a lower few-shot performance than previous studies on SNLI and MNLI. We believe that this is strongly related to automatic phrase-level generation and is bounded by the maximum input length. We leave this topic as a subject for future work.

# B Implementation Details

## B.1 Datasets & Setting

We used the following datasets—SNLI (Bowman et al., 2015), MNLI (Williams et al., 2018), SST-2 (Socher et al., 2013) MRPC (Dolan and Brockett, 2005), MR (Pang and Lee, 2005), CR (Hu and Liu, 2004), MPQA (Wiebe et al., 2005), and Subj (Pang and Lee, 2004). This study followed the same experimental setting from LM-BFF (Gao et al., 2021).

## B.2 Implementation

This proposed approach was implemented using PyTorch (Paszke et al., 2019) and HuggingFace Transformers (Wolf et al., 2020). Experiments were conducted with Nvidia Quadro RTX 8000 GPU. All optimizations were performed using the AdamW optimizer with a linear warm-up of the learning rate. The warmup proportion is 0.6. The gradients are clipped if their norms exceed 1.0.

A T5-large and beam search (e.g., beam width: 30) were used to generate phrase-level verbalizers automatically in a zero-shot manner.

## B.3 Multiple Soft Demonstration Memory Setting

**SST-2, MR, CR, Subj, MRPC, MPQA**

- Length of soft tokens: $n = 10$
- Number of target label: $|\mathcal{Y}| = 2$
- Demonstrations per label: $m = 5$
- Total: $|T| = n \cdot m \cdot |\mathcal{Y}| = 100$

**MNLI**

- Length of soft tokens: $n = 20$
- Number of target label: $|\mathcal{Y}| = 3$

| Task | label | Phrase-level Verbalizers |
|---|---|---|
| **SST-2** | negative<br>positive | [well worth the effort, a total waste of my time, a real treat to watch, so sad, a cultural revolution]<br>[an instant hit, a gift, entertaining on an inferior level, an unforgettable experience, a thriller with an edge] |
| **MR** | negative<br>positive | [delicious, time, ms, the right thing to do, a very sad movie]<br>[refreshing, the best film of the year, well worth the money, such a shame, released on friday] |
| **Subj** | subjective<br>objective | [not a documentary, godard at his best, a funny film, not a great movie, not one of them]<br>[the story of dr, not an easy task, a great site, not a film to be missed, not a great film] |
| **MRPC** | not_equivalent<br>equivalent | [For the three-month daily average, According to the Washington Post, This is not unanticipated, At midday Monday, ? In the 1990s]<br>[On Friday, Yesterday, JERUSALEM, Hi, Today] |
| **MNLI** | contradiction<br>entailment<br>neutral | [In an interview with CNN, we hope, California Rural Justice Consortium, Realigning, In this simulation]<br>[For more information, He's nice, Ueno, I got good results, Exercise Bicycle]<br>[At the University of Georgia, million, Firstly, Arthur Schlesinger, I was embarrassed] |
| **SNLI** | contradiction<br>entailment<br>neutral | [Car in garage, Florida Marlins, The pipe is black, In the boat, , The man is painting.]<br>[Uncle Henry, At a trailer, Hippie is walking on foot, On a dusty path, In this kitchen]<br>[According to locals, Playing with a ball, ", A group of people are walking", Mountains in the background] |
| **CR** | negative<br>positive | [a complete waste of time, working fine for me, not working on my other phone, supposed to work, the same for me]<br>[exactly what i was looking for, a lot of fun to use, a great day, a pleasure to work with you, the perfect phone for me] |
| **MPQA** | negative<br>positive | [ful, here, China, good, customers]<br>[trade, know , transparent, -tuned, and values] |

Table 4: Automatic generation for phrase-level verbalizers $\mathcal{M}_{ph}$ used in our experiments.

| Task | Template | Label words |
|---|---|---|
| SST-2 | It was `[MASK]` . | positive: great, negative: terrible |
| MR | It was `[MASK]` . | positive: great, negative: terrible |
| CR | It was `[MASK]` . | positive: great, negative: terrible |
| MPQA | It was `[MASK]` . | positive: great, negative: terrible |
| Subj | This is `[MASK]` . | subjective: subjective, objective: objective |
| MNLI | ? `[MASK]` , | entailment: Yes, netural: Maybe, contradiction: No |
| SNLI | ? `[MASK]` , | entailment: Yes, netural: Maybe, contradiction: No |
| MRPC | ? `[MASK]` , | equivalent: Yes, not_equivalent: No |

Table 5: Manual templates and label words $\mathcal{M}_{wo}$ that we used in our experiments from LM-BFF (Gao et al., 2021).

- Demonstrations per label: $m = 1$
- Total: $|T| = n \cdot m \cdot |\mathcal{Y}| = 60$

**SNLI**

- Length of soft tokens: $n = 10$
- Number of target label: $|\mathcal{Y}| = 3$
- Demonstrations per label: $m = 1$
- Total: $|T| = n \cdot m \cdot |\mathcal{Y}| = 30$

### B.4 Training Example

Suppose that we train SST-2 dataset following setting: $n = 2, |\mathcal{Y}| = 2$, and $m = 2$. Then $x_{prompt}$ is formed as follows:

$$\begin{aligned} x_{prompt} = \texttt{[CLS]}\ x_1\ \text{It was}\ \texttt{[MASK]}\ .\ \texttt{[SEP]} \\ \texttt{[}T_1\texttt{]}\ \texttt{[}T_2\texttt{]}\ \text{It was an instant hit} \\ \texttt{[}T_3\texttt{]}\ \texttt{[}T_4\texttt{]}\ \text{It was a gift}\ \texttt{[SEP]} \\ \texttt{[}T_5\texttt{]}\ \texttt{[}T_6\texttt{]}\ \text{It was well worth the effort} \\ \texttt{[}T_7\texttt{]}\ \texttt{[}T_8\texttt{]}\ \text{It was a total waste of my time}\ \texttt{[SEP]} \end{aligned}$$

where $x_1$, $\texttt{[}T_1\texttt{]}, \cdots \texttt{[}T_4\texttt{]}$, $\texttt{[}T_5\texttt{]}, \cdots \texttt{[}T_8\texttt{]}$ are the input sentence and multiple soft demonstration memory for positive and negative labels, respectively. In this case, $W_{\texttt{[MLM]}} \in \mathbb{R}^{2\times d}$ is the output embedding weights of label words (e.g., positive: 'great', negative: 'terrible') in a MLM Decoder for the NDP task.