SharPT: Shared Latent Space Prompt Tuning

Bo Pang Semih Yavuz Caiming Xiong Yingbo Zhou
Salesforce Research
{b.pang, syavuz, cxiong, yingbo.zhou}@salesforce.com

Abstract

Prompt tuning is an efficient method for adapting large language models, and Soft Prompt Transfer (SPoT) further narrows the gap between prompt tuning and full model tuning by transferring prompts learned from source tasks to target tasks. It is nevertheless difficult and expensive to identify the source task that provides optimal prompts. In this work, we propose to learn a shared latent space which captures a set of basis skills from a mixture of source tasks. Given an instance, its embedding queries the latent space, yielding a basis skill vector. This vector generates soft prompts, via a lightweight prompt generator, which modulates a frozen model. The latent space and prompt transformation are learned end-to-end by training on source tasks. Transfer learning from source tasks to a target task simply amounts to finetuning the prompt generator, accounting for roughly 0.3% parameters of the frozen backbone model, while the shared latent space is also frozen in finetuning. Our approach outperforms prior soft prompt methods by a significant margin on a variety of tasks such as NLI, sentence completion, QA, conference resolution, word sense disambiguation. We also find, on various model scales, our method achieves competitive performance compared to finetuning the full model.

1 Introduction

Adapting pre-trained large language models (LLMs) has advanced the progress in many NLP areas (Devlin et al., 2019; Raffel et al., 2020). This is typically done by finetuning all parameters of a model on a downstream task (i.e., ModelTuning). This approach is however expensive, especially given the growing sizes of SOTA LLMs.

This limitation motivates recent research on parameter-efficient methods which only tune a small amount of parameters (Houlsby et al., 2019; Brown et al., 2020; Karimi Mahabadi et al., 2021; Lester et al., 2021; Li and Liang, 2021; Hambardzumyan et al., 2021). Among them, a line of research focus on the methods that modulate a frozen LLM via prompts (Liu et al., 2021). Brown et al. (2020) showed that prepending an input text with a prompt, which typically consists of a task description and/or several examples, can effectively adapt a frozen GPT-3. This approach nevertheless underperforms ModelTuning and is sensitive to the choice of prompt wordings. Instead of actual text (or hard prompt), Lester et al. (2021) proposed PromptTuning, which prepends a soft prompt, consisting of k tunable tokens, to input text. The soft prompt can be optimized with gradient-based methods. PromptTuning achieves competitive performance to ModelTuning when the model size is large (e.g., over 10B parameters) but still underperforms with smaller models.

SPoT (Vu et al., 2022) improves over Prompt-

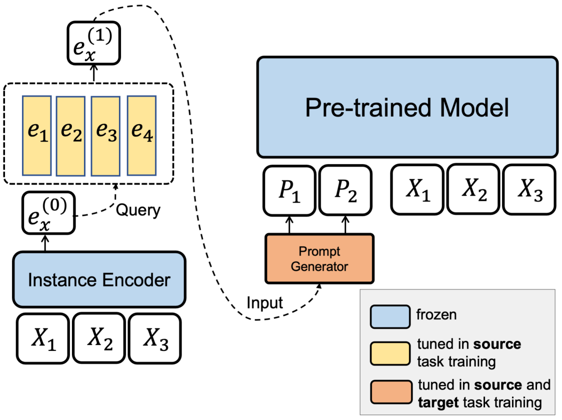

Figure 1: An illustration of SharPT. An instance, as illustrated by with three tokens $\{X_1, X_2, X_3\}$, is encoded by the instance encoder, giving $e_x^{(0)}$, and then queries the skill latent space, resulting in a skill vector $e_x^{(1)}$. The skill vector is transformed by a simple and lightweight prompt generator, outputting prompt tokens (e.g., $\{P_1, P_2\}$). They are prepended to the instance tokens and modulate the pre-trained frozen model. The instance encoder and the pre-trained model are frozen in all scenarios. The skill vectors are tuned in source task training and frozen in target task training. The prompt generator is tuned in both source task and target task training.

Findings of the Association for Computational Linguistics: EACL 2023, pages 1244–1250
May 2-6, 2023 ©2023 Association for Computational Linguistics

TUNING by leveraging knowledge from source tasks. They first learn a task-specific soft prompt for each task in a set of source tasks. Given a target task, they search over the set of source prompts and use the best one or some weighted combination to initialize the prompt for the target task and then tune the prompt. It further narrows the performance gap to MODELTUNING on smaller models. But it is complicated and expensive to identify the source task that provides optimal prompts.

In this work, we propose a novel prompt-based transfer learning method, SHARPT (**Sh**ared Latent Space **P**rompt **T**uning). Figure 1 illustrates the general idea. SHARPT assumes a shared (discrete) latent space by all source and target tasks. We call each vector in the latent space as a *skill vector*, since we assume each one captures a basis NLP capacity or skill after training on the source tasks. Given an instance (from either a source task or a target task), an instance encoder embeds it into an instance vector, which is then used to query the latent space to find the nearest neighbor, yielding a skill vector for this instance. A lightweight prompt generator then generates soft prompts as a function of the selected skill vector. The soft prompts condition a frozen LLM. The latent space and prompt generator are learned end-to-end on a mixture of source tasks. In target task training, the latent space is frozen and only the prompt generator is tuned.

SHARPT retains the key advantage of prior prompt methods, parameter-efficiency. It only updates approximately 0.1% to 0.3% parameters compared to MODELTUNING. Different from prior methods, we add an instance encoder to encode each instance. The instance encoder is lightweight and frozen in all scenarios.

SHARPT and SPOT both exploit a generic idea, *leveraging knowledge shared across tasks*. The approaches to achieve this are however distinctly different. SPOT assumes *task-to-task transfer* based on *task-level prompts* and the knowledge is encoded in task prompts. It is not straightforward to identify a source prompt for a target task. They illustrated two approaches: (1) SPOT-Oracle and (2) SPOT-Retrieval. SPOT-Oracle involves using oracle test labels and expensive search (e.g., 48 times more expensive than regular prompt tuning in their experiments). In SPOT-Retrieval, they first tuned a task prompt for each source and target task independently and retrieved a prompt based on prompt similarity. Note that the retrieval tuning is only for searching a source prompt, which is in addition to final prompt tuning on the target task. In contrast, SHARPT assumes the knowledge is encoded in a *shared latent space* and utilizes *instance-level prompts*, which are generated based on latent vectors from the shared space. These designs make source-to-target transfer simple. We learn the shared latent space with all source tasks in a single training run. Also, the tuning on the target task only requires a single run. Given an instance from a target task, we use the instance embedding to identify a skill vector, learned from all source tasks, which is then transformed to soft prompts.

In summary, we design an instance-prompt-based method by learning a shared skill latent space. We apply SHARPT to a diverse set of tasks covering diverse domains and task categories. We find that our method outperforms prior prompt-based methods and matches full-model-tuning across model scales.

2 Method

Suppose we have a task with data $T = \{(\boldsymbol{x}, \boldsymbol{y})\}$ and a pre-trained LLM P_θ. MODELTUNING updates θ to minimize $\mathcal{L}(\theta) = -\log P_\theta(\boldsymbol{y}|\boldsymbol{x})$ [1]. PROMPTTUNING prepends to $\boldsymbol{x}$ a soft prompt, $\boldsymbol{p} \in \mathbb{R}^{L\times d}$, which has L vectors of size d. It then optimizes $\boldsymbol{p}$ by minimizing $\mathcal{L}(\boldsymbol{p}) = -\log P_\theta(\boldsymbol{y}|\boldsymbol{p}, \boldsymbol{x})$.

SHARPT assumes there exists a discrete latent space, consisting of a set of skill vectors $\boldsymbol{E} = \{\boldsymbol{e}_i \in \mathbb{R}^m\}_{i=1}^K$ with K vectors in total. The soft prompt is a simple transformation of one of the skill vectors $\boldsymbol{e}_i$, that is, $\boldsymbol{p} = f_\alpha(\boldsymbol{e}_i)$. The transformation or prompt generator (f_α) is a light-weight MLP.

$$\boldsymbol{e}_i' = \mathrm{Tanh}(W_1\boldsymbol{e}_i + b_1), \boldsymbol{p}_l = W_2(\boldsymbol{z}_l + \boldsymbol{e}_i') + b_2 \tag{1}$$

where $\boldsymbol{z}_l \in \mathbb{R}^d$ is the position embedding for the lth token (and randomly initialized in training) and $W_1 \in \mathbb{R}^{d\times m}$, $W_2 \in \mathbb{R}^{d\times d}$. Then we have the soft prompt $\boldsymbol{p} = \{\boldsymbol{p}_l\}_{l=1}^L$.

Given $\boldsymbol{x}$, we infer its skill vector by (1) embedding it via a frozen instance encoder (e.g., SimCSE BERT-base), which yields $\boldsymbol{e}_x^{(0)}$; (2) querying $\boldsymbol{E}$ to find the nearest neighbour. Formally, that is,

$$\boldsymbol{e}_x^{(1)} = \boldsymbol{e}_k, \quad k = \arg\min_{i\in[K]} \left\| \boldsymbol{e}_x^{(0)} - \boldsymbol{e}_i \right\|_2. \tag{2}$$

For a target task, our method is then trained with the following loss,

$$\mathcal{L}(\alpha) = -\log P_\theta(\boldsymbol{y}|f_\alpha(\boldsymbol{e}_k), \boldsymbol{x}). \tag{3}$$

[1] Summation over the data is omitted for notation clarity.

In target task training aforementioned, $\boldsymbol{E}$ is known and fixed. We next specify how to learn it from source tasks. Suppose we have N source tasks, $\{T_j^{(s)}\}_{j=1}^N$. We simply mix all tasks together, $T^{(s)} = \bigcup_{j=1}^{N} T_j^{(s)}$. Given $x \in T^{(s)}$ and its embedding $\boldsymbol{e}_x^{(0)}$. $\boldsymbol{E}$ is learned with the following loss,

$$\mathcal{L}(\boldsymbol{E}) = \left\| \mathrm{sg}(\boldsymbol{e}_x^{(0)}) - \boldsymbol{e}_k \right\|_2, \tag{4}$$

where sg() is a stop gradient operator and $\boldsymbol{e}_k$ is defined in Equation (2). The overall loss in source task learning is,

$$\mathcal{L}(\alpha, \boldsymbol{E}) = \mathcal{L}(\alpha) + \mathcal{L}(\boldsymbol{E}) \tag{5}$$

In summary, the forward pass for training on source and target tasks are exactly the same (also see Figure 1). The only difference is the loss function, Equation 5 (source) versus Equation 3 (target).

3 Experiments

High-to-Low Resource Transfer In this setting, the target tasks are low-resource tasks (less than 10K training examples), while the source tasks are high-resource tasks. It consists of 25 tasks in total. There are 15 source tasks (e.g., DocNLI, DROP) and 10 target asks (e.g., BoolQ, ColA). Please see Appendix A for the complete list or Table 1 for the target tasks. We keep the setting to be almost the same as a major experiment in Vu et al. (2022) for a fair comparison, with the exception that we exclude C4 from the source task since it is a much larger dataset than other tasks. Excluding C4 does not affect SPoT performance since it does not provide an optimal source prompt for any target task.

Transfer across Different Task Categories We here investigate the transferability from datasets in some task categories to datasets in other held-out task categories. Following Sanh et al. (2022), we assume datasets in each category measures a general NLP ability, and use the same taxonomy defined in Sanh et al. (2022). The source tasks include (1) QA tasks: ReCoRD, SQuAD, DROP, MultiRC, and RACE; (2) sentiment analysis tasks: Yelp-2 and SST-2; (3) a paraphrase detection task: QQP; (4) a semantic similarity task: CXC. The target tasks include (1) a sentence completion task: COPA; (2) NLI tasks: CB and RTE; (3) a coreference resolution tasks: WSC; (4) a word sense disambiguation task: WiC.

Training Details As in prior works (Raffel et al., 2020; Lester et al., 2021), all datasets are converted to a text-to-text format. All experiments are conducted with T5-base-LM-adapted as the backbone unless stated otherwise. We use a SimCSE (Gao et al., 2021) model (BERT-base) as the instance encoder. Since the instance encoder is always frozen, we can pre-compute the embeddings of all instances and only keep the embeddings. However, we find that memory and time saved in this approach is negligible [2]. In source task training, the model (skill latent space and prompt generator) is simply tuned on the mixture of all source tasks for each setting. The model is tuned for 80K steps. In learning and testing on target tasks, we closely follow the procedure in Vu et al. (2022). The model is tuned for 100K on each target task. We save a checkpoint every 500 steps and report results on the checkpoint with the highest validation performance. The prompt generator generates 64 soft tokens. The following hyperparameters are shared in all target and source task training: learning rate (0.3), the number of warmup steps (4000), optimizer (Adam).

4 Results

High-to-Low Resource Transfer The results are shown in Table 1. We first compare our method, SHARPT, to methods with comparable compute- and parameter-efficiency, PROMPTTUNING and SPoT-Retrieval. Our method has a clear improvement over the two methods across most tasks and on the average performance. We next compare SHARPT with much more expensive methods, SPoT-Oracle and MODELTUNING. Note that SPoT-Oracle is significantly more expensive than our method since it tunes on each target task with each possible task prompt (e.g., it requires roughly 48 times more training time), and utilizes oracle labels. While being much more efficient, SHARPT matches or outperforms SPoT-Oracle. Also, our method performance is on par with the MODELTUNING performance which requires to tune the entire model. These results indicate SHARPT is an efficient and competitive approach.

Transfer across Different Task Categories The results are shown in Table 2. Our method outperforms both PROMPTTUNING and SPoT methods.

[2] For instance, removing the instance encoder in training (by pre-computing the instance embeddings) does not allow a larger batch size compared to including the instance encoder.

	BoolQ	CB	CoLA	COPA	CR	MRPC	RTE	STS-B	WiC	WSC	Average
ModelTuning	**81.4**	94.0	51.1	**71.2**	94.1	87.5	**81.5**	89.4	68.3	80.8	79.9
SPoT-Oracle	77.6	**97.0**	55.6	69.3	93.9	88.7	74.7	**90.0**	**70.2**	77.2	79.4
PromptTuning	73.0	92.7	52.9	56.7	93.5	86.1	68.7	88.1	63.6	71.5	74.7
SPoT-Retrieval	74.2	95.4	54.8	58.3	93.6	88.4	71.6	**90.0**	66.7	72.9	76.6
SHARPT	78.9	94.6	**58.2**	67.0	**94.5**	**89.7**	79.4	89.1	68.8	**81.6**	**80.2**

Table 1: Results on the high-to-low transfer learning setting. Methods in the upper panel are significantly more expensive than those in the lower panel. The best performance is in **bold**, and the second best is underlined.

	COPA	CB	RTE	WSC	WiC
ModelTuning	**71.2**	94.0	**81.5**	**80.8**	68.3
SPoT-Oracle	63.0	92.9	72.0	77.2	**70.2**
PromptTuning	56.7	92.7	68.7	71.5	63.6
SPoT-Retrieval	61.2	89.4	71.4	73.6	66.7
SHARPT	65.0	**94.6**	79.4	79.0	69.8

Table 2: Results on transferring across task categories.

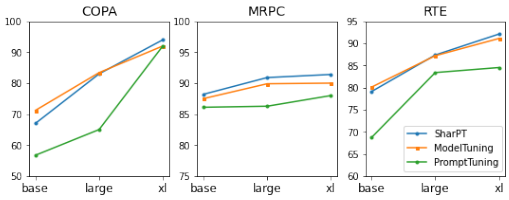

Figure 2: Results on models of different sizes.

	BoolQ	CB	CoLA	COPA
SHARPT	78.9	94.6	58.2	67.0
No Source Task Training	64.3	89.3	10.3	58.0
No Latent Space	67.9	82.4	17.6	61.0

Table 3: Ablation results.

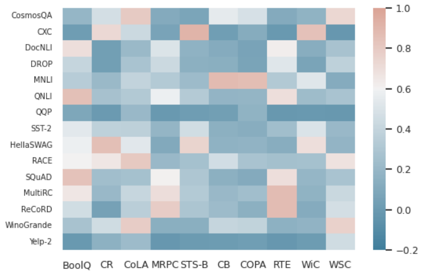

Figure 3: A heatmap of task relations based on skill vector usage of each task.

The improvement over SPOT methods is larger in this setting than in the high-to-low transfer setting. This might be because SPOT relies more on knowledge shared by tasks in the same category, while SHARPT learns a *shared latent space across all source tasks* and is more suitable to leverage knowledge shared across datasets of different categories.

Across Model Scales In the experiments above, we show that our method can close the performance gap between full model tuning and prompt-based methods on a mid-sized model, T5-base (220M). Here conducts experiments with larger models, T5-large (800M) and T5-xl (3B), and compare SHARPT to MODELTUNING and PROMPTTUNING. As shown in Figure 2, SHARPT matches or slightly outperforms MODELTUNING under the three model scales. Our method also shows considerable improvements over PROMPTTUNING.

Ablations We ablate two key components of SHARPT: (1) training on source tasks; (2) skill latent space that captures shared knowledge. See the results in Table 3. Clearly, knowledge learned from source tasks and encoded in the latent space is critical for target task performance.

Task Relations We investigate if the latent space captures source and target task relations to allow knowledge transfer. Each instance queries the latent space and selects one latent skill. We convert this selection to a one-hot vector and treat it as an instance encoding. A task representation is the average of instance encodings in the task. The cosine similarity between two task representations is computed as their relation. The relations between source and target tasks are visualized in Figure 3. It seems that more complicated source tasks such as QA and NLI tasks transfer more knowledge to target tasks via the skill latent space.

5 Conclusion

We introduce SHARPT, which learns a shared latent space which captures a set of basis NLP capacities from a mixture of source tasks. Target instance queries this space to retrieve a skill vector, which then generates prompt tokens to condition a frozen LLM. Our approach outperforms prior soft prompt methods by a significant margin on a variety of tasks. Our method also matches full-model-tuning across model scales.

Limitations

Although our method is much simpler than SPoT, PROMPTTUNING is still arguably the simplest method for adapting LLMs to downstream tasks. It would be a fruitful research direction to design transfer learning approaches that retain (or even improve) our method's performance and meanwhile further simplify our method, getting closer to the simplicity of PROMPTTUNING.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D 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 Ziegler, Jeffrey Wu, Clemens Winter, Chris 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. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. 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.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. 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 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In *Proceedings of the 1st International Conference on Machine Learning Challenges: Evaluating Predictive Uncertainty Visual Object Classification, and Recognizing Textual Entailment (MLCW 2005)*, page 177–190.

Marie-Catherine De Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The CommitmentBank: Investigating projection in naturally occurring discourse. In *Proceedings of Sinn und Bedeutung 23 (SuB 2018)*, volume 23, pages 107–124.

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 (IWP 2005)*.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. 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 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Karen Hambardzumyan, Hrant Khachatrian, and Jonathan May. 2021. WARP: Word-level Adversarial ReProgramming. 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 4921–4933, Online. Association for Computational Linguistics.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Cosmos QA: Machine reading comprehension with contextual commonsense reasoning. 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 2391–2401, Hong Kong, China. Association for Computational Linguistics.

Shankar Iyer, Nikhil Dandekar, and Kornél Csernai. 2017. First Quora Dataset Release: Question pairs.

Rabeeh Karimi Mahabadi, Sebastian Ruder, Mostafa Dehghani, and James Henderson. 2021. Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks. In *Annual Meeting of the Association for Computational Linguistics*.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. 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 252–262, New Orleans, Louisiana. Association for Computational Linguistics.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, Copenhagen, Denmark. Association for Computational Linguistics.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hector J. Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *Proceedings of the Thirteenth International Conference on Principles of Knowledge Representation and Reasoning (KR 2012)*, page 552–561.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. 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 4582–4597, Online. Association for Computational Linguistics.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pretrain, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Zarana Parekh, Jason Baldridge, Daniel Cer, Austin Waters, and Yinfei Yang. 2021. Crisscrossed captions: Extended intramodal and intermodal semantic similarity judgments for MS-COCO. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2855–2870, Online. Association for Computational Linguistics.

Mohammad Taher Pilehvar and Jose Camacho-Collados. 2019. WiC: the word-in-context dataset for evaluating context-sensitive meaning representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL 2019)*, pages 1267–1273.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. 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, Austin, Texas. Association for Computational Linguistics.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S Gordon. 2011. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *Proceedings of the 25th AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning (AAAI Spring Symposium 2011)*.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Teven Le Scao, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M Rush. 2022. Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*.

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.

Tu Vu, Brian Lester, Noah Constant, Rami Al-Rfou', and Daniel Cer. 2022. SPoT: Better frozen model adaptation through soft prompt transfer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5039–5059, Dublin, Ireland. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics (TACL 2019)*, 7:625–641.

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.

Wenpeng Yin, Dragomir Radev, and Caiming Xiong. 2021. DocNLI: A large-scale dataset for document-level natural language inference. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4913–4922, Online. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Sheng Zhang, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme. 2018. Record: Bridging the gap between human and machine commonsense reading comprehension. *arXiv preprint arXiv:1810.12885*.

A Source and Target Tasks in the High-to-Low Resource Transfer Setting

The source tasks include DocNLI (Yin et al., 2021), Yelp-2 (Wang et al., 2018), MNLI (Williams et al., 2018), QQP (Iyer et al., 2017), QNLI (Wang et al., 2018), ReCoRD (Zhang et al., 2018), CXC (Parekh et al., 2021), SQuAD (Rajpurkar et al., 2016), DROP (Dua et al., 2019), SST-2 (Socher et al., 2013), WinoGrande (Sakaguchi et al., 2021), HellaSWAG (Zellers et al., 2019), MultiRC (Khashabi et al., 2018), CosmosQA (Huang et al., 2019), RACE (Lai et al., 2017).

The target tasks include BoolQ (Clark et al., 2019), CB (De Marneffe et al., 2019), CoLA (Warstadt et al., 2019), COPA (Roemmele et al., 2011), CR (De Marneffe et al., 2019), MRPC (Dolan and Brockett, 2005), RTE (Dagan et al., 2005), STS-B (Cer et al., 2017), WiC (Pilehvar and Camacho-Collados, 2019), WSC (Levesque et al., 2012).