How Many Data Points is a Prompt Worth?

Teven Le Scao
Hugging Face
teven@huggingface.co

Alexander M. Rush
Hugging Face
sasha@huggingface.co

Abstract

When fine-tuning pretrained models for classification, researchers either use a generic model *head* or a task-specific *prompt* for prediction. Proponents of prompting have argued that prompts provide a method for injecting task-specific guidance, which is beneficial in low-data regimes. We aim to quantify this benefit through rigorous testing of prompts in a fair setting: comparing prompted and head-based fine-tuning in equal conditions across many tasks and data sizes. By controlling for many sources of advantage, we find that prompting does indeed provide a benefit, and that this benefit can be quantified per task. Results show that prompting is often worth 100s of data points on average across classification tasks.

1 Introduction

The main paradigm for adapting pretrained models for classification (Radford, 2018; Dong et al., 2019; Devlin et al., 2018) is fine-tuning via an explicit classifier head. However, an alternative approach has arisen: adapting the pretrained language model directly as a predictor through autoregressive text generation (Radford et al., 2019) or completion of a cloze task (Trinh and Le, 2018). This method is notably used in T5 fine-tuning (Raffel et al., 2019) leading to state-of-the-art results on the SuperGLUE benchmark (Wang et al., 2019).

One argument made for classification by direct language generation is that it allows us to pick custom *prompts* for each task (McCann et al., 2018). While this approach can be used for zero-shot classification (Puri and Catanzaro, 2019) or priming (Brown et al., 2020), it can also be used in fine-tuning to provide extra task information to the classifier, especially in the low-data regime (Schick and Schütze, 2020a,b).

If this argument is indeed true, it is natural to ask how it impacts the sample efficiency of the model, or more directly, *how many data points is a prompt worth?* As with many low-data and pretraining-based problems, this question is complicated by the fine-tuning setup, training procedure, and prompts themselves. We attempt to isolate these variables through diverse prompts, multiple runs, and best practices in low-training data fine-tuning. We introduce a metric, the *average data advantage*, for quantifying the impact of a prompt in practice.

Our experiments find that the impact of task-targeted prompting can nicely be quantified in terms of direct training data, and that it varies over the nature of different tasks. On MNLI (Williams et al., 2018), we find that using a prompt contributes approximately 3500 data points. On SuperGLUE, it adds approximately 280 data points on RTE (Dagan et al., 2005) and up to 750 on BoolQ (Clark et al., 2019). In low- to medium-data settings, this advantage can be a real contribution to training a model.

2 Related Work

Prompting has been used both for zero-shot and fine-tuning based methods. Zero-shot approaches attempt to answer a task with a prompt without fine-tuning through generation (Radford et al., 2019). GPT3 (Brown et al., 2020) extends this approach to a supervised priming method by taking in training data as priming at inference time, so it can attend to them while answering. T5 (Raffel et al., 2019) and other sequence-to-sequence pretrained models use standard word-based fine-tuning with a marker prompt to answer classification tasks with strong empirical success. Our setting differs in that we are interested in using task-based prompts and fine-tuning, in-between the T5 and GPT2 setting.

Our setting most closely resembles PET (Schick and Schütze, 2020a,b), which claims that task-specific prompting helps transfer learning, espe-

Code available at https://github.com/TevenLeScao/pet

Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 2627–2636
June 6–11, 2021.

cially in the low-data regime. However, in order to reach the best possible results on SuperGLUE, PET introduces several other extensions: semi-supervision via additional pseudo-labeled data, ensembling models trained with several different prompts, and finally distilling the ensemble into a linear classifier rather than a language model. Our aim is to isolate the specific contributions of prompting within supervised fine-tuning.

Finally, recent papers have experimented with discovering prompts through automated processes tailored to the language model (Jiang et al., 2020; Schick et al., 2020). We limit ourselves to human-written prompts, as we are interested into whether prompting itself specifically adds information to the supervised task. It is an interesting question as to whether automatic prompts can have this same impact (relative to the training data they require).

3 Comparison: Heads vs Prompts

Consider two transfer learning settings for text classification: *head-based*, where a generic head layer takes in pretrained representations to predict an output class; *prompt-based*, where a task-specific pattern string is designed to coax the model into producing a textual output corresponding to a given class. Both can be utilized for fine-tuning with supervised training data, but prompts further allow the user to customize patterns to help the model.

For the *prompt* model we follow the notation from PET (Schick and Schütze, 2020a) and decompose a prompt into a *pattern* and a *verbalizer*. The *pattern* turns the input text into a cloze task, i.e. a sequence with a masked token or tokens that need to be filled. Let us use as example an excerpt from SuperGLUE task BoolQ (Clark et al., 2019), in which the model must answer yes-or-no binary questions. In order to let a language model answer the question in *italics*, our pattern is in **bold** (Schick and Schütze, 2020b):

> "Posthumous marriage – Posthumous marriage (or necrogamy) is a marriage in which one of the participating members is deceased. It is legal in France and similar forms are practiced in Sudan and China. Since World War I, France has had hundreds of requests each year, of which many have been accepted. **Based on the previous passage, *can u marry a dead person in france ?* <MASK>**"

The masked word prediction is mapped to a *verbalizer* which produces a class. (here "Yes": True. "No": False[1]). Several *pattern-verbalizer pairs* (*PVPs*) could be used for a single task, differing either through the pattern, the verbalizer, or both. Fine-tuning is done by training the model to produce the correct verbalization. The loss is the cross-entropy loss between the correct answer and the distribution of probabilities amongst the tokens in the verbalizer. We re-use pattern choices from Schick and Schütze (2020b); examples are available in Appendix A.

4 Experimental Setting

We run all experiments with the same pretrained checkpoint, *roberta-large* (355M parameters) from RoBERTa (Liu et al., 2019), which we load from the *transformers* (Wolf et al., 2020) library.[2] In line with previous observations (McCoy et al., 2019; Dodge et al., 2020; Lee et al., 2020), head-based fine-tuning performance varies considerably. We follow recommendations of Mosbach et al. (2020) and Zhang et al. (2020) to train at a low learning rate (10^{-5}) for a large number of steps (always at least 250, possibly for over 100 epochs).

We perform our evaluation on SuperGLUE and MNLI (Williams et al., 2018). These datasets comprise a variety of tasks, all in English, including entailment (MNLI, RTE (Dagan et al., 2005), CB (de Marneffe et al., 2019)), multiple choice question answering (BoolQ (Clark et al., 2019), MultiRC (Khashabi et al., 2018)), and commonsense reasoning (WSC (Levesque et al., 2012), COPA (Roemmele et al., 2011), WiC (Pilehvar and Camacho-Collados, 2018)). We do not include ReCoRD (Zhang et al., 2018) in our comparisons as there is no head model to compare with, since it is already a cloze task. Data sizes range from 250 data points for CB to 392,702 for MNLI. As test data is not publicly available for SuperGLUE tasks, we set aside part of training (from 50 for CB, COPA and MultiRC to 500 for BoolQ) to use for development, and evaluate on their original validation sets. For MNLI, we use the available matched validation and test sets.

We compare models across a scale of available data, starting with 10 data points and increasing exponentially (as high-data performance tends to

[1] The correct answer here is, of course, *yes*. Originated in 1803 as Napoleon rose to power, this practice was mainly to the benefit of war widows.

[2] After experimenting with RoBERTa, AlBERT (Lan et al., 2019) and BERT (Devlin et al., 2018), we found *roberta-large* to have the most consistent performance.

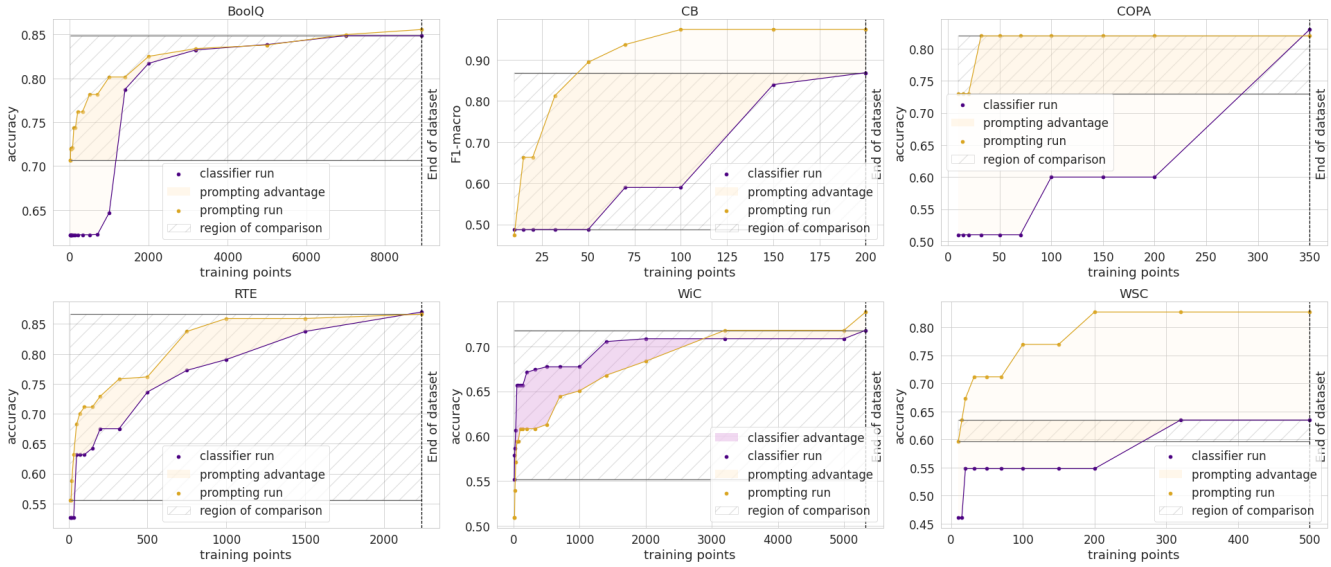

Figure 1: Prompting vs head (classifier) performance across data scales, up to the full dataset, for six SuperGLUE tasks. Compares the best prompt and head performance at each level of training data across 4 runs. Highlighted region shows the accuracy difference of the models. Cross-hatch region highlights the lowest- and highest- accuracy matched region in the curves. The highlighted area in this region is used to estimate the data advantage.

saturate) to the full dataset. For example, for MultiRC, which has 969 data points initially, we start by reserving 50 data points for development. This leaves us with 919 training points, and we train models with 10, 15, 20, 32, 50, 70, 100, 150, 200, 320, 500, 750, and 919 training points. We run every experiment 4 times in order to reduce variance, for a total of 1892 training runs across all tasks. At every point, we report the best performance that has been achieved at that amount of data or lower. Full graphs are presented in Appendix B.

5 Results

Figure 1 shows the main results comparing head- and prompt-based fine-tuning with the best-performing pattern on that task. Prompting enjoys a substantial advantage on every task, except for WiC as is reported in previous results (Schick and Schütze, 2020b). Both approaches improve with more training data, but prompting remains better by a varying amount. Many tasks in SuperGLUE have relatively few data points, but we also see an advantage in large datasets like BoolQ and MNLI.

To quantify how many data points the prompt is worth, we first isolate the y-axis band of the lowest- and highest- accuracy where the two curves match in accuracy.[3] The horizontal line at these points represents the advantage of prompting. We then take the integral in this region, i.e. area between the linearly-interpolated curves[4], divided by the height of the band. The area has the dimension of a quantity of data points times the metric unit, so dividing by the performance range yields a # of data points advantage. As low data training is sensitive to noise, in addition to following best training practices we run several different experiments for each x-point. We use a bootstrapping approach to estimate confidence over these runs. Specifically, we hold out one of the 4 head runs and 4 prompt runs (16 combinations total), and compute the standard deviation of those outcomes.

We report these quantities for every task in Table 1 as *Average advantage*. For almost all the tasks, we see that prompting gives a substantial advantage in terms of data efficiency, adding the equivalent of hundreds of data points on average.

6 Analysis

Impact of Pattern vs Verbalizer The intuition of prompts is that they introduce a task description in natural language, even with few training points. To better understand the zero-shot versus adaptive nature of prompts, we consider a *null verbalizer*, a control with a verbalizer that cannot yield semantic information without training. For every task that requires filling in one word (which excludes

[3] We assume asymptotically the two curves would match, but are limited by data.

[4] In areas where the head model is better, if any, get subtracted from the total.

	Average Advantage (# Training Points)							
	MNLI	BoolQ	CB	COPA	MultiRC*	RTE	WiC	WSC
P vs H	3506 ± 536	752 ± 46	90 ± 2	288 ± 242	384 ± 378	282 ± 34	-424 ± 74	281 ± 137
P vs N	150 ± 252	299 ± 81	78 ± 2	-	74 ± 56	404 ± 68	-354 ± 166	-
N vs H	3355 ± 612	453 ± 90	12 ± 1	-	309 ± 320	-122 ± 62	-70 ± 160	-

Table 1: Average prompting advantage in number of data points for MNLI & SuperGLUE tasks. *P* denotes the prompt model, *H* the head model. On average across performance levels, an MNLI prompt model yields the results of an MNLI head model trained with 3500 additional data points. Confidence levels are based on a multiple random runs (see text). *N* indicates a null-verbalizer prompting task that replaces the verbalizer with a non-sensical mapping. *The comparison band of MultiRC is too small as the head baseline fails to learn beyond majority class; we use the full region for a lower-bound result.

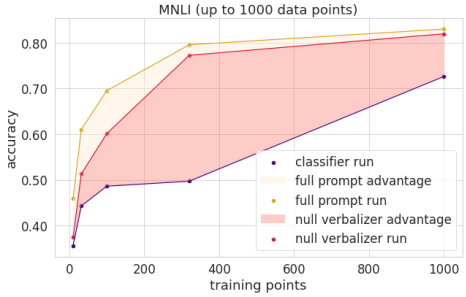

Figure 2: Comparison of full prompt and null verbalizer advantage on MNLI at lower data scales.

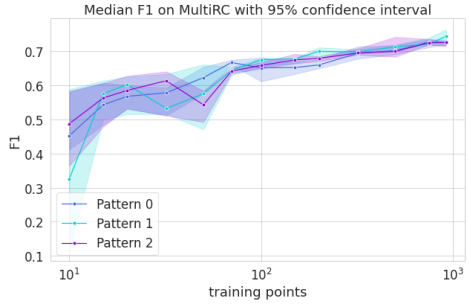

Figure 3: Median performance on MultiRC across runs for three prompts. Differences are inconsistent and eclipsed by the variance within one prompt's runs.

the more free-form COPA and WSC), we replace the verbalizers, for example, "yes", "no", "maybe", "right" or "wrong", with random first names.

Table 1 shows the advantage of the standard prompt over the null verbalizer to estimate this control. We see that for small data tasks such as CB, the null verbalizer removes much of the benefits of prompting. However, with more training data, the model seems to adapt the verbalizer while still gaining the inductive bias benefits of the pattern. Figure 2 showcases this dynamic on MNLI. This result further shows that prompting yields data efficiency even if it is not directly analogous to the generation process of training.

Impact of Different Prompts If the prompt acts as a description of the task, one would expect different valid descriptions to vary in their benefits. In order to compare the different prompts we used on each task, we chart the median performance for each of them under different runs. In nearly every experiment, we find that the confidence intervals of those curves largely overlap, implying that prompt choice is not a dominant hyperparameter, i.e. the variance across random seeds usually outweighs the possible benefits of prompt choice. One exception is the low-data regime of BoolQ, where one of the prompts enjoys a significant few-shot advantage over the others. We plot this curve for MultiRC in Figure 3 and the rest in Appendix C.

Metric sensitivity We treat each metric linearly in calculating advantage; alternatively, we could reparameterize the y axis for each task. This choice does not have a consistent effect for or against prompting. For example, emphasizing gains close to convergence increases prompting advantage on CB and MNLI but decreases it on COPA or BoolQ.

7 Conclusion

We investigate prompting through a systematic study of its data advantage. Across tasks, prompting consistently yields a varying improvement throughout the training process. Analysis shows that prompting is mostly robust to pattern choice, and can even learn without an informative verbalizer. On large datasets, prompting is similarly helpful in terms of data points, although they are less beneficial in performance. In future work, we hope to study the mechanism and training dynamics of the prompting benefits.

8 Impact statement

Significant compute resources were used to run this paper's experiments. A single experiment (defined as one model run, at one data level, on one task) was quite light-weight, taking usually a little under an hour on a single Nvidia V100. However, as we computed a little under two thousand runs, this adds up to about 1800 GPU hours, to which one must add around 400 GPU hours of prototyping and hyper-parameter searching. Those 2200 GPU hours would usually have necessitated the release of about 400kg of CO2, about 40% of a transatlantic flight for a single passenger, in the country where we ran the experiments, although we used a carbon-neutral cloud compute provider.

The main benefit of prompting, rather than compute efficiency, is data efficiency. Although we ran all of our experiments on English, we hope that this property will be especially helpful in low-resource language applications. In a sense, a practitioner could then remedy the lack of task-specific data in their language by introducing information through a prompt. However, this comes with the inherent risk of introducing human biases into the model. Prompt completion also suffers from biases already present within the language model (Sheng et al., 2019). This could cause a prompted model to repeat those biases in classification, especially in the few-shot setting where prompting mostly relies on the pretrained model.

9 Acknowledgments

We thank Steven Cao and Joe Davison for the discussions about prompting that initially spurred this paper. We further thank Timo Schick for making the code for PET available and for discussions about performance replication. We lastly thank Canwen Xu, Yacine Jernite, Victor Sanh, Dimitri Lozeve and Antoine Ogier for their help with the figures and writing of this draft.

References

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.

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. *CoRR*, abs/1905.10044.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. pages 177–190.

Marie-Catherine de Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The commitmentbank: Investigating projection in naturally occurring discourse. *Proceedings of Sinn und Bedeutung*, 23(2):107–124.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. 2020. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know?

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, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 252–262. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A lite BERT for self-supervised learning of language representations. *CoRR*, abs/1909.11942.

Cheolhyoung Lee, Kyunghyun Cho, and Wanmo Kang. 2020. Mixout: Effective regularization to finetune large-scale pretrained language models.

Hector J. Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Thirteenth International Conference, KR 2012, Rome, Italy, June 10-14, 2012*. AAAI Press.

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. *CoRR*, abs/1907.11692.

Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. 2018. The natural language decathlon: Multitask learning as question answering.

R. Thomas McCoy, Junghyun Min, and Tal Linzen. 2019. Berts of a feather do not generalize together: Large variability in generalization across models with similar test set performance.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2020. On the stability of fine-tuning bert: Misconceptions, explanations, and strong baselines.

Mohammad Taher Pilehvar and José Camacho-Collados. 2018. Wic: 10, 000 example pairs for evaluating context-sensitive representations. *CoRR*, abs/1808.09121.

Raul Puri and Bryan Catanzaro. 2019. Zero-shot text classification with generative language models. *CoRR*, abs/1912.10165.

Alec Radford. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S. Gordon. 2011. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *Logical Formalizations of Commonsense Reasoning, Papers from the 2011 AAAI Spring Symposium, Technical Report SS-11-06, Stanford, California, USA, March 21-23, 2011*. AAAI.

Timo Schick, Helmut Schmid, and Hinrich Schütze. 2020. Automatically identifying words that can serve as labels for few-shot text classification.

Timo Schick and Hinrich Schütze. 2020a. Exploiting cloze questions for few shot text classification and natural language inference.

Timo Schick and Hinrich Schütze. 2020b. It's not just size that matters: Small language models are also few-shot learners.

Emily Sheng, Kai-Wei Chang, Premkumar Natarajan, and Nanyun Peng. 2019. The woman worked as a babysitter: On biases in language generation. *CoRR*, abs/1909.01326.

Trieu H. Trinh and Quoc V. Le. 2018. A simple method for commonsense reasoning.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. *CoRR*, abs/1905.00537.

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. 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. Huggingface's transformers: State-of-the-art natural language processing.

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.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q. Weinberger, and Yoav Artzi. 2020. Revisiting few-sample bert fine-tuning.

A Choice of prompts

We use a subset of prompts from (Schick and Schütze, 2020b).

- For entailment tasks (**CB**, **MNLI**, **RTE**) given a premise p and hypothesis h:

  h? | <MASK>, p

  "h"? | <MASK>. "p"

  with "yes" as a verbalizer for entailment, "no" for contradiction, "maybe" for neutrality.

- For **BoolQ**, given a passage p and question q:

  p. Question: q? Answer: <MASK>.

  p. Based on the previous passage, q? <MASK>.

  Based on the following passage, q? <MASK>. p

  with "yes" or "no" as verbalizers for True and False.

- For **COPA**, given an effect e and possible causes c_1 and c_2:

  "c_1" or "c_2"? e, so <MASK> .

  c_1 or c_2? e, so <MASK>.

  and a cause c and possible effects e_1 and e_2:

  "e_1" or "e_2"? <MASK>, because c.

  e_1 or e_2? <MASK>, because c.

  and the verbalizer is the identity function.

- For **MultiRC**, given a passage p, question q and answer a, we estimate whether a is a proper answer with:

  p. Question: q? Is it a? <MASK>.

  p. Based on the previous passage, q? Is the correct answer a? <MASK>.

  p. Question: q? Is the correct answer a? <MASK>.

- For **WiC**, given two sentences s_1 and s_2 and a word w, we classify whether w was used in the same sense.

  "s_1" / "s_2". Similar sense of "w"? <MASK>.

  s_1 s_2 Does w have the same meaning in both sentences? <MASK>.

  With "yes" and "no" as verbalizers.

- For **WSC**, given a sentence s with a marked pronoun *p* and noun n:

  s The pronoun '*p*' refers to <MASK>.

  s In the previous sentence, the pronoun '*p*'refers to <MASK>.

  s In the passage above, what does the pronoun'*p*' refer to? Answer: <MASK>.

  With the identity function as a verbalizer.

B Influence of the reporting method over runs

We chose to report the **accumulated maximum** performance across runs for every model. This means that if the maximum performance over random seeds is smaller than a maximum previously attained with less data points, we use the previous value. This appendix presents results with the **maximum** and **mean** at every point to condense several runs instead. Using either maximum is equivalent; using the mean, however, can make results vary significantly, as the distribution of outcomes is heavily left-skewed, or even bimodal, with poor-performance outliers.

	Average Advantage (accumulated maximum reporting)							
	MNLI	BoolQ	CB	COPA	MultiRC*	RTE	WiC	WSC
P vs H	3506 ± 536	752 ± 46	90 ± 2	288 ± 242	384 ± 378	282 ± 34	-424 ± 74	281 ± 137
P vs N	150 ± 252	299 ± 81	78 ± 2	-	74 ± 56	404 ± 68	-354 ± 166	-
N vs H	3355 ± 612	453 ± 90	12 ± 1	-	309 ± 320	-122 ± 62	-70 ± 160	-

Table 2: Average prompting advantage in number of data points for MNLI & SuperGLUE tasks with accmax reporting. *P* denotes the prompt model, *H* the head model. On average across performance levels, an MNLI prompt model yields the results of an MNLI head model trained with 3500 additional data points. Confidence levels are based on a multiple random runs (see text). *N* indicates a null-verbalizer prompting task that replaces the verbalizer with a non-sensical mapping. *The comparison band of MultiRC is too small as the head baseline fails to learn beyond majority class; we use the full region for a lower-bound result.

	Average Advantage (maximum reporting)							
	MNLI	BoolQ	CB	COPA	MultiRC*	RTE	WiC	WSC
P vs H	3506 ± 536	737 ± 53	86 ± 1	226 ± 189	448 ± 314	218 ± 24	-133 ± 65	297 ± 448
P vs N	150 ± 252	249 ± 80	75 ± 3	-	77 ± 64	331 ± 64	-341 ± 187	-
N vs H	3355 ± 612	488 ± 90	12 ± 2	-	371 ± 305	-113 ± 61	209 ± 176	-

Table 3: Average prompting advantage in number of data points for MNLI & SuperGLUE tasks with maximum reporting.

	Average Advantage (mean reporting)							
	MNLI	BoolQ	CB	COPA	MultiRC*	RTE	WiC	WSC
P vs H	4526 ± 266	1139 ± 285	81 ± 2	292 ± 212	399 ± 77	72 ± 71	447 ± 190	-490 ± 402
P vs N	185 ± 307	514 ± 287	80 ± 2	-	58 ± 2	621 ± 215	10 ± 104	-
N vs H	4341 ± 347	625 ± 402	1 ± 2	-	342 ± 78	-549 ± 203	437 ± 189	-

Table 4: Average prompting advantage in number of data points for MNLI & SuperGLUE tasks with mean reporting.

C Curves on all tasks

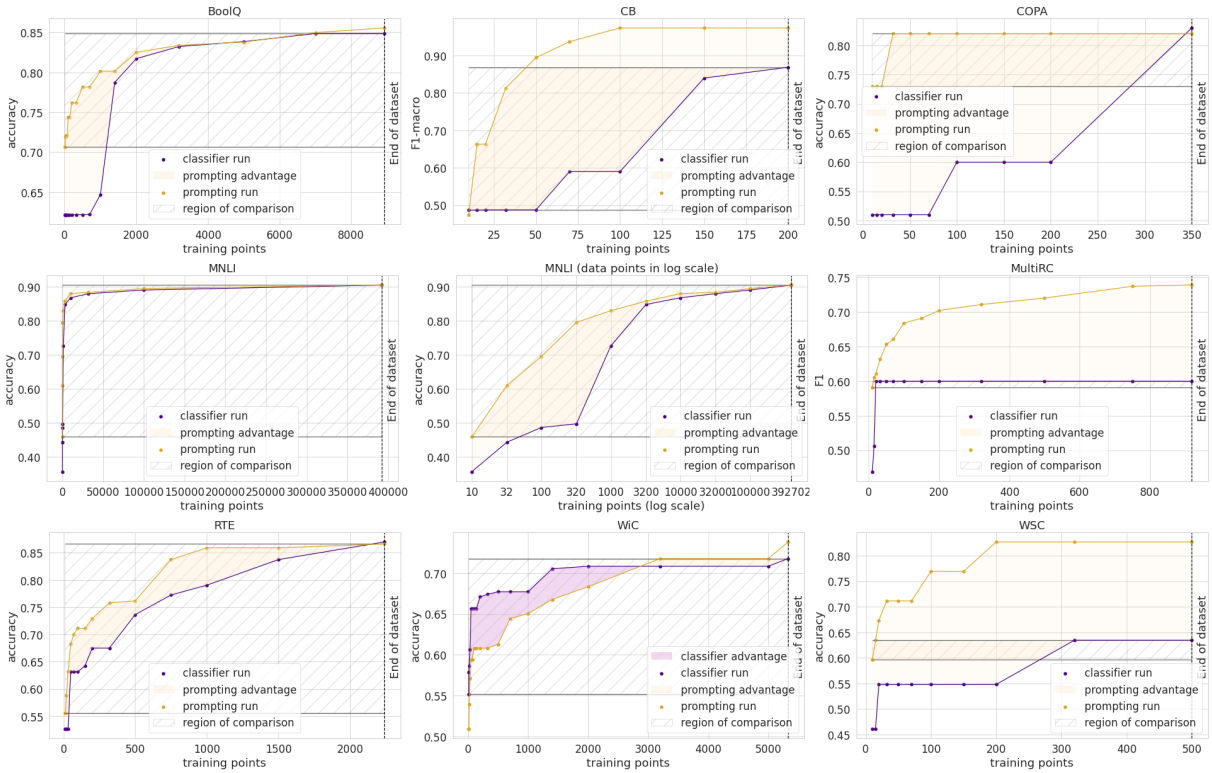

Figure 4: Prompting vs head (classifier) performance across data scales, up to the full dataset, for seven SuperGLUE tasks & MNLI. Compares the best prompt and head performance at each level of training data across 4 runs. Highlighted region shows the accuracy difference of the models. Cross-hatch region highlights the lowest- and highest- accuracy matched region in the curves. The highlighted area in this region is used to estimate the data advantage from prompting.

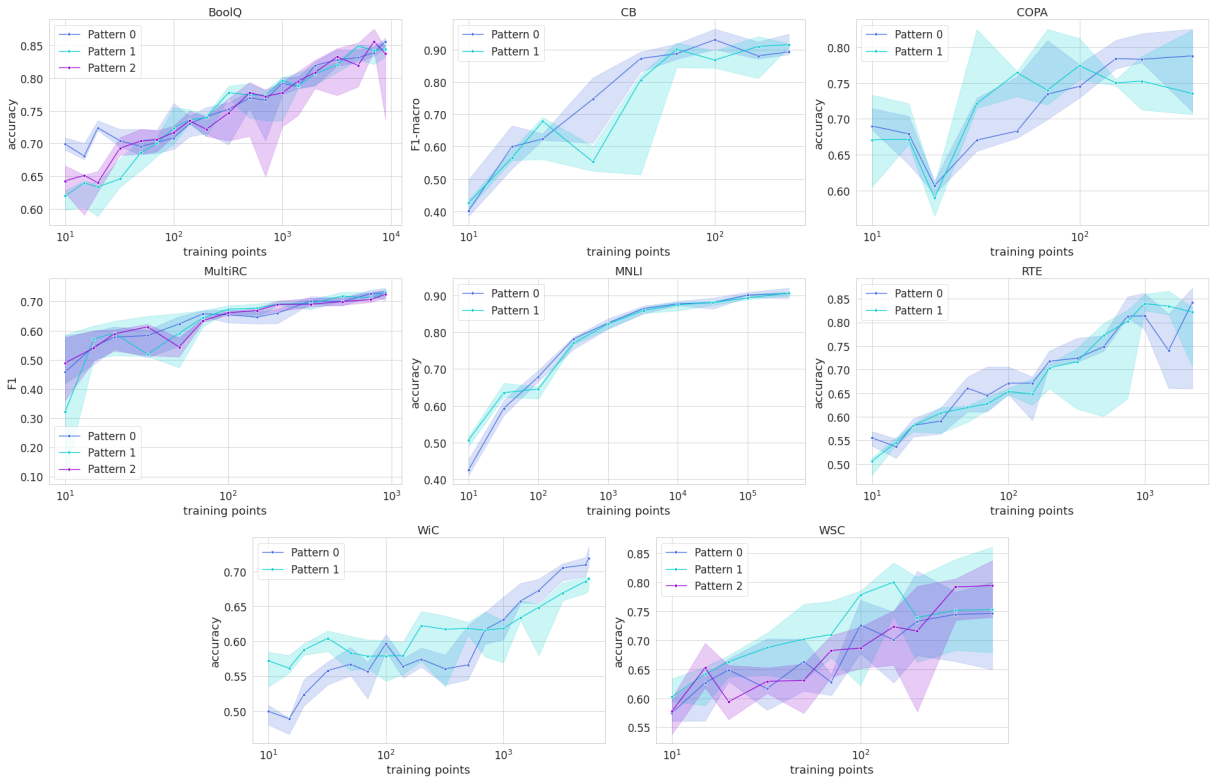

Figure 5: Median performance across runs for each prompt on every task.