# True Few-Shot Learning with Language Models

**Anonymous Author(s)**
Affiliation
Address
email

## Abstract

Pretrained language models (LMs) perform well on many tasks even when learning from a few examples, but prior work uses many held-out examples to tune various aspects of learning, such as hyperparameters, training objectives, and natural language templates ("prompts"). Here, we evaluate the few-shot ability of LMs when such held-out examples are unavailable, a setting we call *true few-shot learning*. We test two model selection criteria, cross-validation and minimum description length, for choosing LM prompts and hyperparameters in the true few-shot setting. On average, both marginally outperform random selection and greatly underperform selection based on held-out examples. Moreover, selection criteria often prefer models that perform significantly worse than randomly-selected ones. We find similar results even when taking into account our uncertainty in a model's true performance during selection, as well as when varying the amount of computation and number of examples used for selection. Overall, our findings suggest that prior work significantly overestimated the true few-shot ability of LMs given the difficulty of few-shot model selection.

## 1 Introduction

Major progress in language model (LM) pretraining has led to the idea that LMs can learn a new task using a small number of examples only, i.e., few-shot learning [1–3]. Few-shot learning overcomes many challenges with data-rich supervised learning: collecting labeled data is expensive, often requires experts, and scales poorly with the number of tasks. However, the few-shot performance of LMs is very sensitive to the textual task description ["prompt"; 3–6], order of training examples [6–8], decoding strategy [9, 10], and other hyperparameters [3, 5, 9, 11, 12], as well as the learning algorithm itself [3, 12]. Thus, effective model selection is crucial for obtaining good few-shot performance.

There are issues with how recent work approaches model selection in few-shot learning, however. Prior work uses large train or held-out sets with many examples to choose prompts [2, 12, 13] and hyperparameters [12]. Other work claims to use no validation set for hyperparameter selection [3, 11, 14] but does not describe how they design other aspects of their learning algorithm (e.g., training objectives). It is unlikely that no validation examples were used, given the sophisticated nature of the proposed algorithms. In this work, we examine if prior few-shot learning methods still perform well when using only the provided examples for model selection, a setting we term *true few-shot learning*.

We find that true few-shot model selection yields prompts that marginally outperform random selection and greatly underperform selection based on held-out examples. Our result shows that prior work may have greatly overestimated the few-shot ability of LMs. In other words, one reason that prompts are so effective ["worth many examples"; 15] is that they are often tuned using many examples. We evaluate two standard model selection criteria – cross-validation (CV) and minimum description length (MDL) – finding that both obtain only limited improvements over random selection and perform much worse than selection using held-out examples. For prompt selection, our observation holds for 9 LMs ranging over 3 orders of magnitude in size [1, 2, 16] on 3 classification tasks and 41

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

tasks in the LAMA benchmark [17]. For choosing hyperparameters, true few-shot selection causes performance to drop by 2-10% across 8 tasks for ADAPET [12], a state-of-the-art few-shot method. Furthermore, true few-shot model selection has high variance in performance; selected models often do much worse than randomly-chosen ones. We find similar results when varying the number of examples used, amount of computation, and conservativeness of our selection criterion. Altogether, our results suggest that model selection is a fundamental roadblock to true few-shot learning.

## 2 Can We Do Model Selection in Few-Shot Learning?

Prior work uses the phrase "few-shot learning" in multiple senses, raising questions about what it means to do few-shot learning. We categorize few-shot learning into three distinct settings, each of which assumes access to different data. Here, we formally disambiguate between these settings to help future work avoid inadvertently comparing few-shot methods that operate in different settings.

Consider the supervised learning scenario where we have a dataset of inputs $x_{1:N}$ and labels $y_{1:N}$, sampled from a distribution over datasets $D$. We aim to determine the learning algorithm $\mathcal{A}^* \in \mathcal{A}_1, \ldots, \mathcal{A}_A$ with the smallest generalization loss $\mathcal{L}$ at predicting $y$ given $x$ on unseen validation examples $D_{\text{val}} \sim D$ after learning on training examples $D_{\text{train}} \sim D$. We say that an algorithm $\mathcal{A}(D_{\text{train}}, R)$ maps a training dataset $D_{\text{train}}$ and various random factors $R$ that influence training to a function that predicts $y$ given $x$. $\mathcal{A}$ specifies, e.g., the model architecture, hyperparameters, and prompt. $R$ includes random factors that impact the results of a learning algorithm, such as parameter initialization and the order of training examples for online learning algorithms like stochastic gradient descent. We say that $\mathcal{A}$ obtains a generalization loss $\mathcal{L}(\mathcal{A}(D_{\text{train}}, R), D_{\text{val}})$ on a given validation set $D_{\text{val}}$. We aim to find the $\mathcal{A}^*$ that minimizes the expected loss across training and validation sets:

$$\text{EL}(\mathcal{A}, R) = \mathbb{E}_{D_{\text{train}}, D_{\text{val}}}\left[\mathcal{L}\Big(\mathcal{A}(D_{\text{train}}, R); D_{\text{val}}\Big)\right]$$

In *data-rich supervised learning*, $\text{EL}(\mathcal{A}, R)$ is usually evaluated with a single train-validation split $(D_{\text{train}}, D_{\text{val}})$. Since large $D_{\text{train}}$ and $D_{\text{val}}$ are not always available, the traditional few-shot setting evaluates $\text{EL}(\mathcal{A}, R)$ with many small $(D_{\text{train}}, D_{\text{val}})$ drawn from many, distinct distributions $D$ [see, e.g., work in meta-learning 18–21]. Each distribution $D$ is sampled from $D^*$, a distribution over distributions (e.g., of similar tasks), so we call this setting *multi-distribution few-shot learning*.

Recent work does not assume access to data from other distributions, performing few-shot learning using only a few examples from a single distribution to update a pretrained LM [2, 12]. These papers use a large validation set $D_{\text{val}}$ to tune the learning algorithm $\mathcal{A}$, a setting we term *tuned few-shot learning*. For example, Brown et al. [2] try prompts with different phrasings and numbers of training examples to improve the validation accuracy of GPT-3. Tam et al. [12] choose the early stopping iteration, prompt, and other model-specific hyperparameters based on validation performance. Tuned few-shot learning relies on many labeled examples, so we argue that tuned few-shot learning does not qualify as few-shot learning. If many validation examples are available, they could be incorporated into the training set and trained on using data-rich supervised learning. Tuned few-shot learning algorithms should be compared against data-rich supervised learning algorithms that use the same amount of total data $|D_{\text{train}}| + |D_{\text{val}}|$.

In this work, we evaluate the success of tuned few-shot learning methods when no large $D_{\text{val}}$ is available, a setting we term *true few-shot learning*. Formally, we aim to choose a learning algorithm $\mathcal{A}$ with low expected loss $\text{EL}(\mathcal{A}, R)$ using only a small training set $D_{\text{train}}$ drawn from a single distribution. Here, we must choose $\mathcal{A}$ by approximating $\text{EL}(\mathcal{A}, R)$, e.g., using cross-validation. Several papers claim to circumvent the need to estimate $\text{EL}(\mathcal{A}, R)$ by choosing hyperparameters based on an educated guess [3, 9, 14]. However, the proposed learning algorithms themselves are quite sophisticated, and it is unclear how they were designed if not by using validation performance. Other work chooses the learning algorithm and hyperparameters using one or multiple other datasets before evaluating on the target dataset [5, 11]. Such approaches fall under *multi-distribution few-shot learning* and cannot be directly compared to methods that attempt to perform true few-shot learning, even though prior work has made such comparisons [14].

In what follows, we describe two model selection criteria – cross-validation and minimum description length – which we use to evaluate tuned few-shot methods in the true few-shot setting.

### 2.1 Cross-validation

Cross-Validation (CV) [22–24] is one of the most widely used methods for estimating generalization loss [25]. CV has also been used in prior work on multi-distribution few-shot learning [26, 27]. CV randomly partitions $D_{\text{train}}$ into $K$ equally-sized folds $F(D_{\text{train}})_1, \ldots, F(D_{\text{train}})_K$ and evaluates the average loss on a validation fold $F(D_{\text{train}})_k$ when training on the remaining data $F(D_{\text{train}})_{\neg k}$:

$$\text{CV}(\mathcal{A}, R, F) = \mathbb{E}_{k \sim \text{Unif}(1,K)} \left[ \mathcal{L}\Big( \mathcal{A}(F(D_{\text{train}})_{\neg k}, R); F(D_{\text{train}})_k \Big) \right]$$

In this way, CV forms $K$ train-validation splits out of the pool of labeled examples. CV with one example per fold ($K = N$ folds) is commonly referred to as leave-one-out CV (LOOCV).

### 2.2 Minimum description length

We may also form train-validation splits in a different manner than CV, drawing inspiration from work on the Minimum Description Length (MDL) principle [28]. MDL can be estimated by evaluating the average loss on a fold $F(D)_k$ when training on the previous folds $F(D)_{1:k-1}$:

$$\text{MDL}(\mathcal{A}, R, F) = \mathbb{E}_{k \sim \text{Unif}(1,K)} \left[ \mathcal{L}\Big( \mathcal{A}(F(D_{\text{train}})_{1:k-1}, R); F(D_{\text{train}})_k \Big) \right]$$

This procedure is referred to as "online coding" [29, 30], as it evaluates the generalization loss of the algorithm as it learns "online" from more and more data.[1] There are other ways to evaluate MDL [see 31, for an overview]. We use online coding as it has been shown to be an effective way to estimate MDL, especially for deep learning methods [32].

MDL measures generalization because it evaluates how much a learning algorithm compresses the labels $y_{1:N}$ given the inputs $x_{1:N}$, and because better compression implies better generalization [33]. Recent work has used MDL to determine which learning algorithms are most effective at explaining the given data [Rissanen Data Analysis; 10, 34].

### 2.3 Variance matters

We evaluate the generalization loss of the algorithm chosen by CV (likewise for MDL):

$$\mathcal{L}(\mathcal{A}_{\text{CV}}(D_{\text{train}}, R), D_{\text{val}}), \qquad \text{where } \mathcal{A}_{\text{CV}} = \arg\min_{\mathcal{A}} \mathbb{E}_{R,F}[\text{CV}(\mathcal{A}, R, F)].$$

The above loss should be low in expectation, across different datasets $D_{\text{train}} \sim D$, $D_{\text{val}} \sim D$, and random factors $R, F$: $\mathbb{E}_{D_{\text{train}}, D_{\text{val}}, R, F}[\mathcal{L}(\mathcal{A}_{\text{CV}}(D_{\text{train}}, R), D_{\text{val}})]$. The loss should also be low in variance: $\mathbb{V}_{D_{\text{train}}, D_{\text{val}}, R, F}[\mathcal{L}(\mathcal{A}_{\text{CV}}(D_{\text{train}}, R), D_{\text{val}})]$. Low variance implies that CV/MDL *reliably* choose an algorithm that generalizes to $D_{\text{val}}$ when trained with a given $D_{\text{train}}$ and random factors $R, F$. Reliability is important for many practical or commercial applications where worst-case performance is important, such as image recognition [35, 36], dialogue systems [37, 38], and robotics [39, 40].

We also experiment with explicitly taking into account an algorithm's variance during model selection, choosing $\mathcal{A}_{\text{CV}}$ to minimize a conservative estimate of CV, $\text{CV}_\alpha(\mathcal{A})$, chosen such that the probability $\Pr_{R,F}[\text{CV}(\mathcal{A}, R, F) < \text{CV}_\alpha(\mathcal{A})]$ is high:

$$\text{CV}_\alpha(\mathcal{A}) = \mathbb{E}_{R,F}[\text{CV}(\mathcal{A}, R, F)] + \alpha \sqrt{\mathbb{V}_{R,F}[\text{CV}(\mathcal{A}, R, F)]}$$

where $\alpha$ is a hyperparameter set based on the desired probability. In particular, if $\text{CV}(\mathcal{A}, R, F)$ follows a normal distribution $\mathcal{N}$ when sampling $R, F$, then $\text{CV}(\mathcal{A}, R, F) \leq \text{CV}_\alpha(\mathcal{A})$ with probability $\int_{-\infty}^{\alpha} \mathcal{N}(\mu = 0, \sigma = 1)$ for a given $R, F$. $\text{CV}_\alpha(\mathcal{A})$ resembles the Watanabe Akaike Information Criterion [41], which estimates the generalization of a model trained with $\mathcal{A}$ using the expected loss from a model trained with $\mathcal{A}$ plus the variance in training loss across models trained with $\mathcal{A}$.

[1] Online coding formally computes a sum over $\mathcal{L}(.)$ rather than the expectation, which differs by a constant factor. The two are equivalent for our purposes (ranking $\mathcal{A}$).

### 2.4 Other model selection criteria

Prior work has developed other model selection criteria such as the Akaike Information Criterion [AIC; 42], Watanabe-Akaike Information Criterion [WAIC; 41], and Mallows' $C_p$ [43]. These methods often rely on assumptions or quantities that are not available in the context of deep learning (AIC, Mallows' $C_p$) or are approximations of LOOCV (WAIC). Since state-of-the-art few-shot learning methods tend to be based on deep learning, we focus on CV and MDL as our model selection criteria. In Appendix §C, we also test several other criteria that are applicable to deep learning methods.

Selection criteria can be optimized automatically, e.g. with bayesian optimization [44–46], evolutionary methods [45, 47, 48], reinforcement learning [49], or gradient descent [50–53]. Such methods aim to match the performance of exhaustive search, the optimal approach (used in our work).

## 3 True Few-Shot Prompt Selection

Recent work on LMs performs few-shot learning by providing training examples as input in the form of a natural language "prompt" [2, 3, 9]. For example, for a question-answering task, Brown et al. [2] prepend input examples with "READING COMPREHENSION ANSWER KEY" before providing them to GPT-3 (see Appendix Table 2 for more examples). They then have the LM complete the remaining words in the prompt, conditioning on earlier words (including various input examples), following the LM's pretraining objective (next word prediction). No parameter updates are involved. It is not obvious *a priori* which prompts will generalize well for a given LM, and there is high variance in how well different prompts generalize [3, 6], even between prompts with minor differences [e.g., one comma; 5]. Effective prompt selection is thus crucial for true few-shot learning.

### 3.1 Experimental setup

In what follows, we test on LAMA [17], a benchmark for retrieving facts with LMs, for which prior work has developed many strategies for designing prompts [4, 54–56]. LAMA evaluates the accuracy of LMs at choosing the correct target object for various (`subject`, `relation`, `object`) triples present in knowledge bases, such as (`Dante`, `born-in`, `Florence`). We use the "TREx" split, which consists of 41 relations (up to 1k examples each). Petroni et al. [17] design a prompt for each relation, which an LM completes to predict an answer (e.g., "The birthplace of Dante was _"). Some relations have multiple valid target entities, so LAMA evaluates how often one of the true answers matches the top-predicted token (out of 20k candidates). We only use examples from the LAMA-UnHelpfulNames subset [LAMA-UHN; 57] which filters out easy-to-guess examples (e.g., "The Apple Watch was created by _" with the answer *Apple*). We test the 5-shot accuracy of 9 popular LMs of various sizes: GPT-3 [175B, 13B, 6.7B, 2.7B parameter models; 2], GPT-2 [1.5B, 782M, 345M, 117M models; 2], and DistilGPT-2 [16], a distilled, 82M parameter version of GPT-2 117M.[2] The supplementary material contains the code to reproduce all results and plots in our paper.

**Prompts** To form our set of candidate prompts $\mathcal{A}_1, \ldots, \mathcal{A}_A$, we rely on LAMA as well as the LM Prompt And Query Archive [LPAQA; 4]. For each relation, we use the manually-written prompt from LAMA, plus LPAQA prompts formed by (1) paraphrasing the manual prompt with back-translation (2) mining from Wikipedia, and (3) paraphrasing the top mined prompt. For each relation, we use up to 16 prompts with a mean of 12 prompts (see Appendix §F.1 for details on the prompts we use).

**Computing CV and MDL** As the loss function $\mathcal{L}$, we use the negative log-likelihood (NLL) of the label given the input over all held-out examples $\sum_{(x,y)} -\log p(y|x)$. We use NLL as in prior work on MDL [32, 60, 10], to retain MDL's property as a measure of label compression. For CV, NLL avoids ties between prompts that arise from using accuracy in the context of such limited data (e.g., 5 examples). We use $K = N$ folds (where $N$ is the number of training examples) for both MDL and CV (here, LOOCV). Here, $N$-fold CV requires $N$ forward passes to evaluate the loss on each of the $N$ examples when conditioning on the $N - 1$ other examples. $N$-fold MDL can be computed using a

[2] We use OpenAI's API for GPT-3 (`https://beta.openai.com/`) and HuggingFace Transformers [58] via PyTorch [59] for GPT-2/DistilGPT-2. OpenAI does not disclose API model sizes, so following [6, 7], we assume that the 4 API models are the 4 largest from [2]. We will update our paper should OpenAI release model sizes.

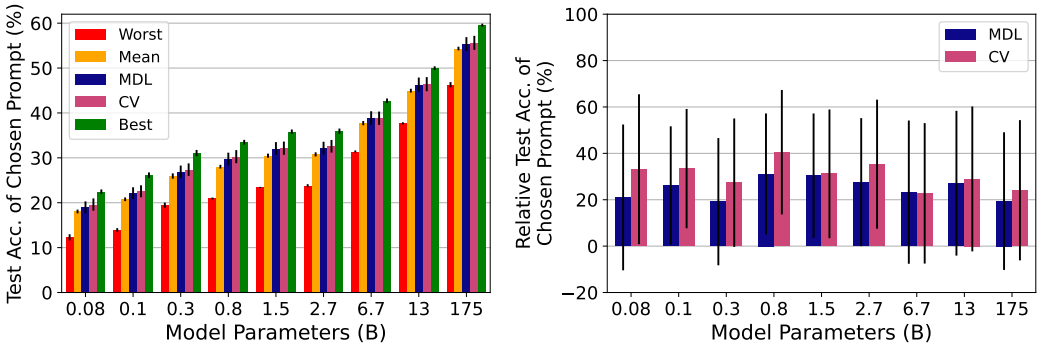


Figure 1: **Left**: LAMA-UHN accuracy of CV/MDL-chosen prompts vs. accuracy of the worst, average (randomly-selected), and best prompt (prior work). **Right**: The average accuracy gain from using CV/MDL-chosen prompts instead of randomly-chosen ones, relative to the gain from the best prompt. We plot mean/std. err. across 5 runs with different training sets. CV/MDL-chosen prompts obtain only small improvements over randomly-chosen ones and do far worse than the best prompts.

single LM forward pass to compute the loss on each example conditioned on the previous examples. This feature makes MDL more computationally efficient than CV.

**Marginalizing out example order** The order of training examples impacts the generalization of LMs [7], so we treat order as a random factor $R$ that we marginalize over to evaluate the generalization of a prompt $\mathcal{A}$. We compute the exact $\mathbb{E}_{R,F}[\text{CV}(\mathcal{A}, R, F)]$ and $\mathbb{E}_{R,F}[\text{MDL}(\mathcal{A}, R, F)]$ by averaging over all $N!$ training example orders. We use $N = 5$ examples to limit $N!$. We estimate the average test accuracy on $N! = 120$ examples in LAMA, excluding the training examples, by evaluating on one test example per permutation of training examples. We compute CV, MDL, and test accuracy with $N! = 120$ forward passes in total by appending a test example to each permutation of training examples, and we compute all selection criteria using the same set of $N! = 120$ forward passes to maximize comparability across different methods. We show the test accuracy from CV/MDL-chosen prompts, averaged over all relations. For comparison, we show the test accuracy of always choosing (1) the best prompt, chosen using held-out accuracy as in prior work, (2) the worst prompt, as a lower bound, and (3) random prompts (we show the mean accuracy over all prompts).

### 3.2 How well does prompt selection do in true few-shot learning?

Fig. 1 (left) shows the results; prompt selection obtains marginal improvements over random selection across model sizes ranging over 3 orders of magnitude. Prompts chosen by CV and MDL alike underperform the best prompt (chosen using held-out performance) by 5-7% absolute on average. In fact, prompts chosen based on held-out performance often outperform larger models whose prompts are chosen in a true few-shot manner. CV and MDL do tend to choose better-than-average prompts, but only close the gap between the average and best prompts by 20-40%, as shown in Fig. 1 (right). We find similar results for several other tasks (Appendix §B) and selection criteria (Appendix §C).

Fig. 2 (left) shows that CV/MDL struggle to choose the prompt with the highest test accuracy. Poor top-prompt selection is especially prevalent for larger models like GPT-3 175B that have spurred interest in prompt design (only 21% accuracy for CV vs. 9% for random chance). In Appendix §A, we show that CV/MDL-chosen prompts differ from the best prompts not only in accuracy but also in how effectively they transfer to models of different sizes. Altogether, our results show that optimal prompt selection is difficult in the true few-shot setting, and that prior work overestimated the ability of LMs by using held-out examples for prompt selection.

### 3.3 How reliably does prompt selection improve over the average prompt?

If the average improvement from prompt selection is small, can we at least obtain an improvement with high probability for a given task and train set? Fig. 1 (left) shows that the worst prompts do far worse than average, so it would be useful if prompt selection helped to avoid the worst prompts. We

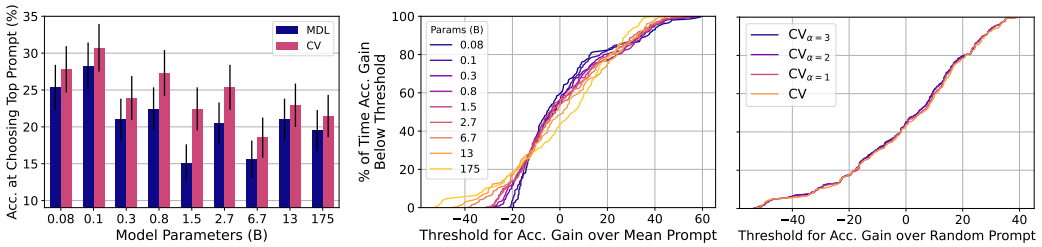


Figure 2: **Left**: CV/MDL have low accuracy at choosing the best prompt (mean/std. err. across 5 runs with different train sets). **Middle**: The chance of various accuracy gains over the average prompt, when using prompts chosen by CV, and (**Right**) conservative estimates of CV that also minimize variance in CV; CV often chooses worse-than-average prompts, even with conservative selection.

examine the probability with which prompt selection obtains various accuracy gains over the average prompt and show results in Fig. 2 (middle) for CV (and similar results in Appendix §D for MDL).

CV/MDL-chosen prompts show high variance in test accuracy relative to the average prompt. For most model sizes (.1B-6.7B), the chance of improving over the average, randomly-chosen prompt is only $\sim$56% for CV and $\sim$55% for MDL. The performance of prompt selection forms a long-tailed distribution; there is a $\sim$27% chance that prompt selection causes an accuracy drop of $\sim$13% for all model sizes and CV/MDL alike. Furthermore, the tails grow heavier as model size increases. For the largest model (GPT-3 175B), CV/MDL-chosen prompts sometimes do far worse than average, e.g., 40% worse, 5% of the time. Our results suggest a troubling trend: as models grow bigger and generalize better, our ability to reliably choose good prompts degrades. One possible explanation is that larger models have the capacity to draw more complex decision boundaries, requiring more examples to estimate the true expected loss on unseen examples; we may need to scale validation sets along with model size. In Appendix §B, we show similar results for several other tasks. Overall, the limited average-case gains from prompt selection cannot be expected with any reasonable confidence in the true few-shot setting, a problem that will only become worse with larger models.

### 3.4 Can we increase the likelihood of improved performance from prompt selection?

As we have shown, CV and MDL do not reliably choose better-than-average prompts. Here, we explore the extent to which we can reduce the variance in generalization by explicitly preferring prompts with low variance (§2.3). For the largest model (GPT-3 175B), we choose prompts based on a conservative estimate of generalization loss, $\text{CV}_\alpha$ (§2.3). We show the test accuracy for the prompt chosen with various levels of confidence $\alpha \in \{1, 2, 3\}$ and with CV ($\alpha = 0$).

As shown in Fig. 2 (right), all $\alpha$ lead to a similar distribution of performance gain as CV. For example, CV outperforms the average prompt 50% of the time vs. 51% for $\alpha = 2$. These results suggest that it is non-trivial to choose prompts that reliably perform better than random selection, even when explicitly minimizing variance in generalization.

### 3.5 Does prompt selection improve with more labeled examples?

The poor performance of prompt selection methods may be due to using such a small number of labeled examples. As the number of labeled examples increases, we expect prompt selection methods to improve. Thus, true few-shot prompt selection may be possible with a few dozen examples (though it is not always possible to use more examples, due to limits on input length for LMs like GPT). We therefore examine the test accuracy of CV/MDL-chosen prompts as we use an increasing number of labeled examples $N \in \{5, 10, 15, 20, 30, 40\}$. For $N \geq 10$, it is not feasible to marginalize over all possible training example permutations, so we randomly sample 120 permutations (to match $N = 5$) such that each example occurs the same number of times in each position (i.e., to use each example as the held-out CV fold the same number of times). We run the experiment for $\leq$6.7B parameter models, since it is prohibitively costly to run with larger models via the OpenAI API.

As shown in Fig. 3, there is no consistent trend in the performance of prompt selection, both in terms of task performance (left) and in terms of accuracy at choosing the highest accuracy prompt (right).

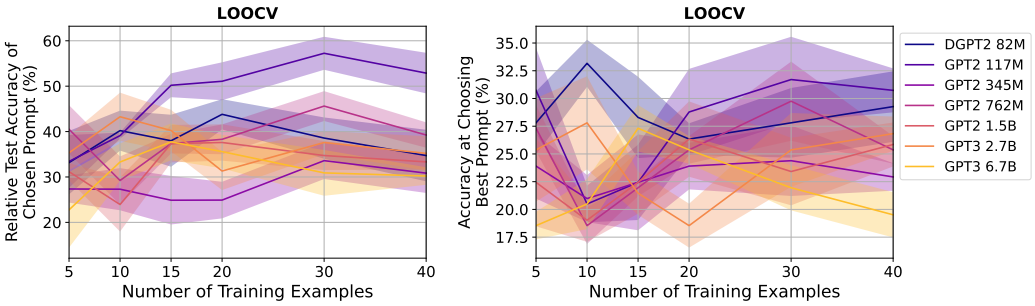


Figure 3: Increasing the number of examples up to 40 does not clearly improve CV in terms of (**Left**) accuracy gain over the average prompt (scaled to 0), relative to the best one (scaled to 100) or (**Right**) accuracy at choosing the best prompt. Mean/std. err. on LAMA over 5 runs (varying train sets).

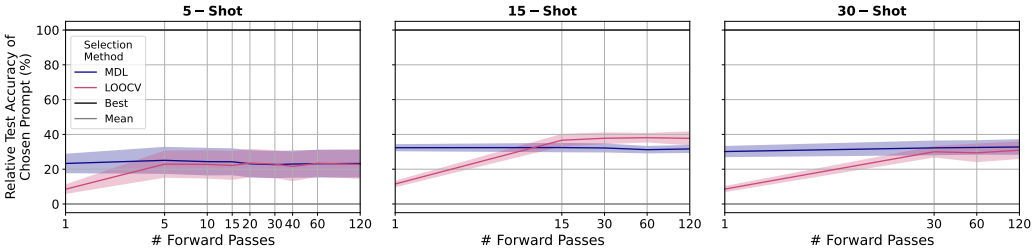


Figure 4: For $N \in \{5, 10, 30\}$ -shot learning, increasing the compute used to estimate CV/MDL does not notably improve the accuracy of chosen prompts beyond a certain point (1 forward pass for MDL, $N$ forward passes for CV). Mean/std. err. across 5 runs for GPT-3 6.7B.

Even in higher-data regimes (40 examples), CV/MDL struggle to choose effective prompts and do not consistently, across model sizes, perform better than choosing examples based on 5 examples. Our findings are surprising, because the true-few shot setting is where prompt design has been thought most promising, due to the scarcity of training data [15]. However, the true few-shot setting is also one in which prompt selection is hardest, greatly undermining the potential value of prompts.

### 3.6 Does prompt selection improve with more computation?

In the preceding sections, we computed $\mathbb{E}_{R,F}[\text{CV}(\mathcal{A}, R, F)]$ using a fixed number of samples for $R$. Can we improve prompt selection by using more samples, at the cost of increased computation? To answer this question, we vary the number of samples of $R$ (and thus LM forward passes) used to compute the above expectation and choose prompts as described in §2.3. To estimate CV with a single forward pass, we sample a single fold $k$ (here, a single example) and evaluate accuracy on fold $k$ when conditioning the LM on all others folds. Fig. 4 shows the results for $N \in \{5, 15, 30\}$ training examples using the largest model from §3.5 (GPT-3 6.7B).

Computation is not the bottleneck in prompt selection, as test accuracy roughly plateaus after one forward pass for MDL and $N$ forward passes for CV. This observation holds across $N$, as well as all models with <6.7B parameters (omitted for space). Our results suggest that true few-shot prompt selection is fundamentally limited by the number of examples available.

## 4 True Few-Shot Hyperparameter Selection

Having shown that true few-shot prompt selection is difficult, we now study the effectiveness of model selection in the context of hyperparameter selection more generally. We examine the ADAPET model [12], as it is open-source[3] and currently the top-performing few-shot model according to SuperGLUE [61], a standard NLP benchmark. ADAPET finetunes the pretrained $\text{ALBERT}_{\text{xxlarge-v2}}$

[3] `https://github.com/rrmenon10/ADAPET`

| | **BoolQ** Acc | **CB** Acc/F1 | **COPA** Acc | **RTE** Acc | **WiC** Acc | **WSC** Acc | **MultiRC** EM/F1 | **ReCoRD** EM/F1 | **Avg** |
|---|---|---|---|---|---|---|---|---|---|
| **Worst** | $75.0_{4.8}$ | $79.5_{2.3}$/$67.3_{7.8}$ | $76.8_{2.2}$ | $63.2_{4.0}$ | $49.0_{1.3}$ | $77.2_{1.8}$ | $38.5_{7.4}$/$80.0_{2.9}$ | $76.2_{1.8}$/$86.5_{1.2}$ | $69.4_{1.5}$ |
| **Mean** | $79.0_{1.5}$ | $85.9_{2.3}$/$74.5_{11.0}$ | $81.1_{2.9}$ | $70.8_{2.5}$ | $51.5_{1.8}$ | $82.5_{2.7}$ | $44.2_{6.6}$/$82.3_{2.7}$ | $78.3_{1.3}$/$87.8_{0.8}$ | $73.9_{1.2}$ |
| **MDL** | $76.5_{5.8}$ | $85.7_{5.6}$/$74.8_{13.4}$ | $82.0_{2.9}$ | $70.4_{8.5}$ | $52.2_{3.0}$ | $82.0_{3.1}$ | $39.7_{8.1}$/$80.6_{3.2}$ | $78.9_{0.7}$/$88.2_{0.4}$ | $73.4_{2.8}$ |
| **CV** | $78.9_{2.4}$ | $83.9_{5.3}$/$69.2_{10.3}$ | $80.5_{3.3}$ | $68.7_{7.0}$ | $51.1_{1.6}$ | $83.1_{2.6}$ | $41.9_{7.2}$/$81.4_{3.1}$ | $78.7_{1.6}$/$88.1_{1.0}$ | $73.0_{2.1}$ |
| **Best** | $80.9_{1.0}$ | $89.8_{3.1}$/$79.8_{13.4}$ | $84.8_{4.5}$ | $76.7_{1.8}$ | $54.1_{2.3}$ | $86.6_{1.8}$ | $46.8_{6.9}$/$83.4_{2.9}$ | $80.4_{1.1}$/$89.2_{0.7}$ | $77.2_{0.9}$ |
| **ADAPET** [12] | 80.3 | 89.3 / 86.8 | 89.0 | 76.5 | 54.4 | 81.7 | 39.2 / 80.1 | 85.4 / 92.1 | 77.3 |
| **iPET** [9] | 80.6 | 92.9 / 92.4 | 95.0 | 74.0 | 52.2 | 80.1 | 33.0 / 74.0 | 86.0 / 86.5 | 76.8 |
| **PET** [9] | 79.4 | 85.1 / 59.4 | 95.0 | 69.8 | 52.4 | 80.1 | 37.9 / 77.3 | 86.0 / 86.5 | 74.1 |
| **GPT-3** [2] | 77.5 | 82.1 / 57.2 | 92.0 | 72.9 | 55.3 | 75.0 | 32.5 / 74.8 | 89.0 / 90.1 | 73.2 |

Table 1: ADAPET results on SuperGLUE validation when choosing early stopping checkpoint and masked LM rate using CV/MDL vs. the worst/mean/best hyperparameters chosen with validation ($\text{mean}_{\text{std. dev.}}$ over four 32-shot train sets). On all tasks, CV/MDL-chosen hyperparameters perform similar to or worse than average, and several points below the best hyperparameters.

LM [62] to (1) classify each label as correct/incorrect given the input and (2) to predict randomly masked out input tokens given the label and unmasked input tokens, similar to Masked LM [63]. ADAPET was developed in the context of tuned few-shot learning, as ADAPET's hyperparameters were chosen using validation examples. We examine how ADAPET does in the true few-shot setting.

We choose two hyperparameters in a true few-shot manner: the early stopping checkpoint and fraction of words masked for the masked LM objective. ADAPET performs $T = 1000$ gradient updates on batches of 16 examples and chooses the checkpoint at $T \in \{250, 500, 750, 1000\}$ with the highest validation accuracy. ADAPET also chooses the best masking fraction $M \in \{0.075, 0.10, 0.105, 0.15\}$. Following ADAPET, we evaluate on SuperGLUE, a suite of 8 NLP tasks. SuperGLUE consists of four question-answering tasks (BoolQ, COPA, MultiRC, ReCoRD), a coreference resolution task (WSC), two entailment detection tasks (RTE, CV), and a commonsense reasoning task (WiC); see Appendix §F.2 for task details. We use CV/MDL to choose $T$ and $M$ (out of 16 total combinations) and then train a model on the full dataset with the chosen $T$ and $M$. We use FewGLUE [9], the 32-example subset of SuperGLUE used in prior work on few-shot learning. We also use 3 other 32-example subsets that we randomly sample from SuperGLUE, to estimate variance in performance across training sets. ADAPET uses a prompt during fine-tuning, choosing the prompt based on validation examples. To avoid using validation-tuned prompts, we use the first prompt for every task as the authors do for ablation studies. Since training ADAPET is expensive, we evaluate CV/MDL with $K = 8$ folds.[4] We show results in Table 1.

**Results** Across all SuperGLUE tasks, CV/MDL hyperparameter selection performs similar to or worse than average (randomly-chosen) hyperparameters and several points worse than the best hyperparameters. In the true few-shot setting, the average SuperGLUE performance of ADAPET drops below that of earlier methods (PET and iPET), highlighting how the use of validation examples can give the false appearance of progress in few-shot learning. On MultiRC, CV/MDL choose hyperparameters that give similar performance to the worst hyperparameters, another indication that model selection methods do not consistently prevent worst-case behavior in the true few-shot setting. Preliminary analysis in Appendix §E suggests that choosing better-than-average hyperparameters requires several thousand examples. Overall, our results indicate that it is not just prompt selection but model selection in general that is challenging in very low-data regimes.

## 5 Conclusion and Future Work

Our work shows that it is challenging to make even the most basic decisions about few-shot learning algorithms using only a few labeled examples. Instead, it may be more promising to make additional assumptions. The meta-learning setting assumes access to data from many other tasks in order to perform learning and model selection [20, 21, 64, 65]. Transfer learning and multitask learning assume access to data that is directly related to the task with limited data [66–69]. Data augmentation techniques assume there is a viable way to create more data from limited data [70–73]. Other approaches assume unlabeled data and develop unsupervised model selection techniques [74–76].

[4] See Appendix §F.4 for details on how we evaluate MDL on different SuperGLUE tasks.

When labeled data is cheap, the simplest approach is to assume more examples for validation—in which case we might be better off training on the additional examples. Unless we make such assumptions explicit, we cannot make meaningful comparisons between few-shot learning algorithms. We find the above avenues to be more promising future directions than true few-shot learning given the challenge of model selection.

Inspired by prior work [77, 78], we offer recommendations for future work in true few-shot learning:

- Report all hyperparameters (prompts) considered and the hyperparameter selection criteria.
- Include validation examples in the number of examples used by a few-shot learning algorithm. Validation examples include all examples used to decide on any aspect of learning: hyperparameters, prompts, training objectives, decoding strategies, model architecture, etc.
- Once you have decided on the learning algorithm, submit your model for test evaluation directly, without first evaluating on validation. Report the total number of test evaluations conducted (ideally, just one). Use the validation set only after test evaluation for any ablations you report, to avoid making decisions about your algorithm with the validation set.
- Do not rely on hyperparameters from prior work that were tuned using validation examples for the same benchmark (e.g., SuperGLUE), to avoid indirectly benefiting from validation examples. Instead, re-tune such hyperparameters using only the given few-shot examples.

The above protocols are strict but mimic how a true few-shot learning algorithm would be used in a real, low-data setting. To ensure researchers comply with such strict protocols, future benchmarks may need to keep large test sets private while releasing only a few labeled examples.

Given our negative results on true few-shot learning, a major question remains: is it possible to select models in a true zero-shot setting? Prior work uses LMs for zero-shot learning by choosing an arbitrary prompt [17, 79] which requires no data but is suboptimal [4]. Other efforts try multiple prompts and choose between them via trial and error alongside manual evaluation [1], effectively leveraging human supervision. CLIP [13] achieves high zero-shot accuracy on ImageNet after extensively tuning prompts and label names using ImageNet's training set (1.28M examples), as we noticed from the open-source code.[5] The authors report a 5% accuracy gain from tuning prompts alone, but the training examples used for tuning are not available in true zero-shot learning. Without any labeled data, the problem of model selection is even more challenging than in the true few-shot case. Overall, our work provides guidance for future work in few-shot learning by clarifying the assumptions made by the true few-shot setting and empirically demonstrates that model selection is a major roadblock to true few-shot learning.

## 6 Limitations and Broader Impact

We facilitate fair comparisons between few-shot methods in future work by disambiguating between three few-shot settings: multi-distribution, tuned, and true few-shot learning. We highlight that one setting, tuned few-shot learning, gives up the practical advantage of few-shot learning by using many labeled examples. Furthermore, we show that several tuned few-shot learning algorithms work significantly worse in the true few-shot setting, without tuning on many examples. While we explored CV, MDL, and several other criteria (Appendix §C), it is possible that effective true few-shot model selection is possible using other criteria (§2.4) or heuristics not explored here. In this event, our work will have discouraged work on a few-shot learning setting with applications to low-data settings, e.g., that involve low-resource languages or expert annotation. Overall, however, we believe our work will redirect future work to few-shot settings with more practical applications.

We show that it is hard to detect when a small input change hurts an LM's generalization, even when the change appears reasonable to human readers. We argue that practitioners will benefit from knowing such limitations, but they may also be discouraged from deploying LMs in many useful contexts, such as question-answering, hate speech detection, automatic translation, and commercial dialogue systems. Our findings may also encourage adversaries to target LM-based applications and highlight which models are most susceptible to attack (e.g., larger models). By shedding light on the shortcomings of (few-shot) LMs, we hope to spur future work to address these shortcomings.

[5] `https://github.com/openai/CLIP/blob/main/notebooks/Prompt_Engineering_for_ImageNet.ipynb`

# References


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

[2] 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. Language models are few-shot learners, 2020.

[3] Timo Schick and Hinrich Schütze. Exploiting cloze questions for few-shot text classification and natural language inference. *Computing Research Repository*, arXiv:2001.07676, 2020.

[4] Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. How can we know what language models know? *TACL*, 8:423–438, 2020.

[5] Tianyu Gao, Adam Fisch, and Danqi Chen. Making pre-trained language models better few-shot learners. *arXiv preprint arXiv:2012.15723*, 2020.

[6] Tony Z. Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. Calibrate before use: Improving few-shot performance of language models, 2021.

[7] Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity, 2021.

[8] Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, L. Carin, and W. Chen. What makes good in-context examples for gpt-3? *arXiv*, abs/2101.06804, 2021.

[9] Timo Schick and Hinrich Schütze. It's not just size that matters: Small language models are also few-shot learners. *Computing Research Repository*, arXiv:2009.07118, 2020.

[10] Ethan Perez, Douwe Kiela, and Kyunghyun Cho. Rissanen data analysis: Examining dataset characteristics via description length. In *ICML*, 2021.

[11] Timo Schick and H. Schutze. Few-shot text generation with pattern-exploiting training. *arXiv*, abs/2012.11926, 2020.

[12] Derek Tam, Rakesh R Menon, Mohit Bansal, Shashank Srivastava, and Colin Raffel. Improving and simplifying pattern exploiting training. *arxiv preprint arXiv:2103.11955*, 2021.

[13] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021.

[14] Sinong Wang, Han Fang, Madian Khabsa, Hanzi Mao, and Hao Ma. Entailment as few-shot learner, 2021.

[15] Teven Le Scao and Alexander M. Rush. How many data points is a prompt worth?, 2021.

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

[17] Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. Language models as knowledge bases? In *EMNLP*, pages 2463–2473, Hong Kong, China, November 2019. ACL.

[18] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, koray kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *NeuRIPS*, volume 29. Curran Associates, Inc., 2016.

[19] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *NeuRIPS*, volume 30. Curran Associates, Inc., 2017.

[20] S. Ravi and H. Larochelle. Optimization as a model for few-shot learning. In *ICLR*, 2017.

[21] Ke Li and Jitendra Malik. Learning to optimize. *arXiv*, abs/1606.01885, 2017.

[22] David M. Allen. The relationship between variable selection and data agumentation and a method for prediction. *Technometrics*, 16(1):125–127, 1974.

[23] M. Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society. Series A (Methodological)*, 36:111–133, 1974.

[24] Seymour Geisser. The predictive sample reuse method with applications. *Journal of the American Statistical Association*, 70(350):320–328, 1975.

[25] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[26] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, volume 70 of *ICML'17*, pages 1126–1135. JMLR.org, 2017.

[27] Aravind Rajeswaran, Chelsea Finn, Sham M Kakade, and Sergey Levine. Meta-learning with implicit gradients. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *NeurIPS*, volume 32. Curran Associates, Inc., 2019.

[28] J. Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465 – 471, 1978.

[29] J. Rissanen. Universal coding, information, prediction, and estimation. *IEEE Transactions on Information Theory*, 30(4):629–636, 1984.

[30] A. P. Dawid. Present position and potential developments: Some personal views: Statistical theory: The prequential approach. *Journal of the Royal Statistical Society. Series A (General)*, 147(2):278–292, 1984.

[31] Peter Grünwald. A tutorial introduction to the minimum description length principle. *CoRR*, math.ST/0406077, 06 2004.

[32] Léonard Blier and Yann Ollivier. The description length of deep learning models. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *NeuRIPS*, volume 31, pages 2216–2226. Curran Associates, Inc., 2018.

[33] Alselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K. Warmuth. Occam's razor. *Inf. Process. Lett.*, 24(6):377–380, April 1987.

[34] Koustuv Sinha, Robin Jia, Dieuwke Hupkes, Joelle Pineau, Adina Williams, and Douwe Kiela. Masked language modeling and the distributional hypothesis: Order word matters pre-training for little. *CoRR*, abs/2104.06644, 2021.

[35] P. J. Phillips, Hyeonjoon Moon, S. A. Rizvi, and P. J. Rauss. The feret evaluation methodology for face-recognition algorithms. *TPAMI*, 22(10):1090–1104, 2000.

[36] Joy Buolamwini and Timnit Gebru. Gender shades: Intersectional accuracy disparities in commercial gender classification. In Sorelle A. Friedler and Christo Wilson, editors, *Fairness, Accountability and Transparency*, volume 81 of *PMLR*, pages 77–91, New York, NY, USA, 23–24 Feb 2018. PMLR.

[37] Peter Henderson, Koustuv Sinha, Nicolas Angelard-Gontier, Nan Rosemary Ke, Genevieve Fried, Ryan Lowe, and Joelle Pineau. Ethical challenges in data-driven dialogue systems. In *AAAI/ACM Conference on AI, Ethics, and Society*, AIES '18, page 123–129, New York, NY, USA, 2018. Association for Computing Machinery.

[38] Chandra Khatri, Behnam Hedayatnia, Anu Venkatesh, Jeff Nunn, Yi Pan, Qing Liu, Han Song, Anna Gottardi, Sanjeev Kwatra, Sanju Pancholi, Ming Cheng, Qinglang Chen, Lauren Stubel, Karthik Gopalakrishnan, Kate Bland, Raefer Gabriel, Arindam Mandal, Dilek Hakkani-Tür, Gene Hwang, Nate Michel, Eric King, and Rohit Prasad. Advancing the state of the art in open domain dialog systems through the alexa prize. *CoRR*, abs/1812.10757, 2018.

[39] Javier García, Fern, and o Fernández. A comprehensive survey on safe reinforcement learning. *JMLR*, 16(42):1437–1480, 2015.

[40] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul F. Christiano, John Schulman, and Dan Mané. Concrete problems in AI safety. *CoRR*, abs/1606.06565, 2016.

[41] Sumio Watanabe. Asymptotic equivalence of bayes cross validation and widely applicable information criterion in singular learning theory. *JMLR*, 11(116):3571–3594, 2010.

[42] H. Akaike. A new look at the statistical model identification. *TACON*, 19(6):716–723, 1974.

[43] C. L. Mallows. Some comments on cp. *Technometrics*, 15(4):661–675, 1973.

[44] Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In Carlos A. Coello Coello, editor, *Learning and Intelligent Optimization*, pages 507–523, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[45] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *NeuRIPS*, volume 24. Curran Associates, Inc., 2011.

[46] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *NeuRIPS*, volume 25. Curran Associates, Inc., 2012.

[47] Risto Miikkulainen, Jason Liang, Elliot Meyerson, Aditya Rawal, Daniel Fink, Olivier Francon, Bala Raju, Hormoz Shahrzad, Arshak Navruzyan, Nigel Duffy, and Babak Hodjat. Chapter 15 - evolving deep neural networks. In Robert Kozma, Cesare Alippi, Yoonsuck Choe, and Francesco Carlo Morabito, editors, *Artificial Intelligence in the Age of Neural Networks and Brain Computing*, pages 293–312. Academic Press, 2019.

[48] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V. Le. Regularized evolution for image classifier architecture search. *AAAI*, 33(01):4780–4789, Jul. 2019.

[49] Barret Zoph and Quoc V. Le. Neural architecture search with reinforcement learning. In *ICLR*. OpenReview.net, 2017.

[50] J. Larsen, L.K. Hansen, C. Svarer, and M. Ohlsson. Design and regularization of neural networks: the optimal use of a validation set. In *Neural Networks for Signal Processing VI. IEEE Signal Processing Society Workshop*, pages 62–71, 1996.

[51] Yoshua Bengio. Gradient-Based Optimization of Hyperparameters. *Neural Computation*, 12(8):1889–1900, 08 2000.

[52] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee. Choosing multiple parameters for support vector machines. *Machine Learning*, 46:131–159, 2004.

[53] Hanxiao Liu, Karen Simonyan, and Yiming Yang. DARTS: Differentiable architecture search. In *ICLR*, 2019.

[54] Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *EMNLP*, pages 4222–4235, Online, November 2020. ACL.

[55] Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. Gpt understands, too, 2021.

[56] Zexuan Zhong, Dan Friedman, and Danqi Chen. Factual probing is [MASK]: learning vs. learning to recall. *CoRR*, abs/2104.05240, 2021.

[57] Nina Poerner, Ulli Waltinger, and Hinrich Schütze. E-BERT: Efficient-yet-effective entity embeddings for BERT. In *Findings of EMNLP*, pages 803–818, Online, November 2020. ACL.

[58] 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. Transformers: State-of-the-art natural language processing. In *EMNLP: System Demonstrations*, pages 38–45, Online, October 2020. ACL.

[59] 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. 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, *NeuRIPS*, pages 8024–8035. Curran Associates, Inc., 2019.

[60] Elena Voita and Ivan Titov. Information-theoretic probing with minimum description length. In *EMNLP*, pages 183–196, Online, November 2020. ACL.

[61] Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Superglue: A stickier benchmark for general-purpose language understanding systems. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *NeuRIPS*, volume 32. Curran Associates, Inc., 2019.

[62] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. In *ICLR*, 2020.

[63] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, pages 4171–4186, Minneapolis, Minnesota, June 2019. ACL.

[64] Eleni Triantafillou, Tyler Zhu, Vincent Dumoulin, Pascal Lamblin, Utku Evci, Kelvin Xu, Ross Goroshin, Carles Gelada, Kevin Swersky, Pierre-Antoine Manzagol, and Hugo Larochelle. Meta-dataset: A dataset of datasets for learning to learn from few examples. In *ICLR*, 2020.

[65] Qinyuan Ye, Bill Yuchen Lin, and Xiang Ren. Crossfit: A few-shot learning challenge for cross-task generalization in NLP. *CoRR*, abs/2104.08835, 2021.

[66] Rich Caruana. Learning many related tasks at the same time with backpropagation. In G. Tesauro, D. Touretzky, and T. Leen, editors, *NeuRIPS*, volume 7. MIT Press, 1995.

[67] Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, July 1997.

[68] Jason Phang, Thibault Févry, and Samuel R. Bowman. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *CoRR*, abs/1811.01088, 2018.

[69] Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. Multi-task deep neural networks for natural language understanding. In *ACL*, pages 4487–4496, Florence, Italy, July 2019. ACL.

[70] Vid Kocijan, Ana-Maria Cretu, Oana-Maria Camburu, Yordan Yordanov, and Thomas Lukasiewicz. A surprisingly robust trick for the Winograd schema challenge. In *ACL*, pages 4837–4842, Florence, Italy, July 2019. ACL.

[71] Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. Unsupervised data augmentation for consistency training. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *NeuRIPS*, volume 33, pages 6256–6268. Curran Associates, Inc., 2020.

[72] Jiaao Chen, Zichao Yang, and Diyi Yang. MixText: Linguistically-informed interpolation of hidden space for semi-supervised text classification. In *ACL*, pages 2147–2157, Online, July 2020. ACL.

[73] Yiben Yang, Chaitanya Malaviya, Jared Fernandez, Swabha Swayamdipta, Ronan Le Bras, Ji-Ping Wang, Chandra Bhagavatula, Yejin Choi, and Doug Downey. Generative data augmentation for commonsense reasoning. In *Findings of EMNLP*, pages 1008–1025, Online, November 2020. ACL.

[74] Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. Unsupervised neural machine translation. In *ICLR*, 2018.

[75] Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. Unsupervised machine translation using monolingual corpora only. In *ICLR*, 2018.

[76] Ethan Perez, Patrick Lewis, Wen-tau Yih, Kyunghyun Cho, and Douwe Kiela. Unsupervised question decomposition for question answering. In *EMNLP*, pages 8864–8880, Online, November 2020. ACL.

[77] Avital Oliver, Augustus Odena, Colin Raffel, Ekin D. Cubuk, and Ian J. Goodfellow. Realistic evaluation of deep semi-supervised learning algorithms. *CoRR*, abs/1804.09170, 2018.

[78] Jesse Dodge, Suchin Gururangan, Dallas Card, Roy Schwartz, and Noah A. Smith. Show your work: Improved reporting of experimental results. In *EMNLP*, pages 2185–2194, Hong Kong, China, November 2019. ACL.

[79] Allyson Ettinger. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *TACL*, 8:34–48, 2020.

[80] Aki Vehtari, Andrew Gelman, and Jonah Gabry. Practical bayesian model evaluation using leave-one-out cross-validation and waic. *Statistics and Computing*, 27(5):1413–1432, September 2017.

[81] Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah A. Smith. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *CoRR*, abs/2002.06305, 2020.

[82] Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *NAACL*, pages 2924–2936, Minneapolis, Minnesota, June 2019. ACL.

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

[84] Ido Dagan, Oren Glickman, and Bernardo Magnini. The pascal recognising textual entailment challenge. In Joaquin Quiñonero-Candela, Ido Dagan, Bernardo Magnini, and Florence d'Alché Buc, editors, *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*, pages 177–190, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[85] Roy Bar Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. The second PASCAL recognising textual entailment challenge. In *Second PASCAL Challenges Workshop on Recognising Textual Entailment*, 2006.

[86] Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. The third pascal recognizing textual entailment challenge. In *ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, RTE '07, page 1–9, USA, 2007. ACL.

[87] Luisa Bentivogli, Ido Dagan, Hoa Trang Dang, Danilo Giampiccolo, and Bernardo Magnini. The fifth pascal recognizing textual entailment challenge. In *TAC*, 2009.

[88] Mohammad Taher Pilehvar and Jose Camacho-Collados. WiC: The word-in-context dataset for evaluating context-sensitive meaning representations. In *NAACL*. ACL, 2019.

[89] Hector J. Levesque, Ernest Davis, and Leora Morgenstern. The winograd schema challenge. In *KR*, KR'12, page 552–561. AAAI Press, 2012.

[90] Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *NAACL*. ACL, 2018.

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

# Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See §6.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See §6.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] Included in the supplementary material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] We specify these details throughout our paper, since our work aims to investigate the importance of such details. We describe our data splitting procedure formally in §2, and dataset-specific details in §3 (especially §3.1) and §4.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] We show all plots and tables with error bars, except where it is non-trivial to plot error bars (the heatmaps in Fig. 5) or when we directly plot the distribution over errors (Fig. 2 middle/right).
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Appendix §F.5.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See §3.1, §B and Appendix §F for citations for the dataset, prompts, and models we use for experiments in §3. See §4 and Appendix §F for citations for the dataset and model we use for experiments in §4.
   (b) Did you mention the license of the assets? [Yes] See Appendix §F.3.
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] We include the code to reproduce all experiments as supplementary material.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] We only use publicly-released datasets that allow for use in a research context.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] See Appendix §F.3.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]