Enhancing the Generation of Predictions and Natural Language Explanations via Sparse Few-shot Fine-tuning and Prompting

Jesus Solano
ETH Zürich
jesus.solano@inf.ethz.ch

Mardhiyah Sanni
University of Edinburgh
m.o.sanni@sms.ed.ac.uk

Oana-Maria Camburu
University College London
o.camburu@ucl.ac.uk

Pasquale Minervini
University of Edinburgh
p.minervini@ed.ac.uk

Abstract

Generating natural language explanations (NLEs) for models' predictions have gained increasing interest, but it typically demands large datasets of human-written NLEs for ground-truth labels at training time, which can be costly and impractical. Recent works have shown promise in fine-tuning pre-trained language models (PLMs) in conjunction with prompt-based learning for few-shot scenarios. However, PLMs typically have billions of parameters, making full fine-tuning expensive. We introduce SPARSEFIT, a sparse few-shot fine-tuning strategy that leverages discrete prompts to jointly generate predictions and NLEs. Our experiments with T5 and Llama 2 across four datasets reveal that SPARSEFIT configurations that fine-tune only 6.8% of the model parameters achieve competitive performance for both task performance and NLE quality compared to full fine-tuning. Moreover, SPARSEFIT produces better results on average than other state-of-the-art Parameter-Efficient Fine-Tuning (PEFT) techniques.

1 Introduction

Neural networks have achieved great success [11, 6], but often lack human-intelligible explanations for their predictions, which is crucial for trustworthiness. These models usually lack human-intelligible explanations for their predictions, which are paramount for ensuring their trustworthiness. There is growing interest in building neural models that provide natural language explanations (NLEs), which are easy for humans to understand and more expressive than other types of explanations [48, 50]. However, these models require large datasets of human-written NLEs for training, which could be costly and time-consuming to collect. Few-shot learning of NLEs has emerged as a solution [32, 54], but current methods involve fine-tuning the *entire* model with a few examples, which is computationally expensive due to the large number of parameters in modern NLE models [41].

This paper investigates whether *sparse fine-tuning* (i.e. fine-tuning only a subset of parameters), in conjunction with prompt-based learning (i.e., textual instructions provided to a model [28]), can help in scenarios with limited availability of training instances with labels and NLEs. While sparse fine-tuning has been applied in Natural Language Processing (NLP) before [18, 29, 55], our work is the first to analyze sparse fine-tuning in the context of jointly generating predictions and NLEs. We extend the sparse fine-tuning strategy beyond only bias terms [55] to include all layers and pairs of layers in a language model.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

In this paper, we propose SPARSEFIT, an efficient few-shot prompt-based training regime for models generating both predictions and NLEs for their predictions. We experiment with SPARSEFIT on three pre-trained language models (PLMs) that have previously shown high performance on task performance and NLE generation, namely T5 [36], UNIFIEDQA [13] and LLAMA 2-7B [46]. We test our approach on four popular NLE datasets: e-SNLI [8], ECQA [1], SBIC [39], and ComVE [49], and evaluate both the task performance and the quality of the generated NLEs, the latter with both automatic metrics and human evaluation. Overall, SPARSEFIT shows competitive performance in few-shot learning settings with 48 training instances. For example, fine-tuning only the *Normalization Layer* together with the *Self-attention Query Layer*, which account for 6.84% of the model's parameters, consistently gave the best performance (penalized by the number of fine-tuned parameters) over all four datasets. Remarkably, SPARSEFIT outperforms the current state-of-the-art parameter-efficient fine-tuning (PEFT) models in terms of both task performance and quality of generated NLEs in two of the four datasets. Therefore, we conclude that few-shot sparse fine-tuning of PLMs can achieve results competitive with fine-tuning the entire model.

2 Related Work

Few-shot learning refers to training models with limited labeled data for a given task [16, 47].It has been effectively used in various applications, including image captioning [14], object classification [37], behavioral bio-metrics [44], graph node classification [40], and language modeling [47]. Large Language Models (LLMs) have shown impressive skills in few-shot learning [6, 11] due to their extensive pre-training corpora and the statistical capacity of the models [20].

Parameter-Efficient Fine-Tuning Using fine-tuning, LLMs have shown breakthrough language understanding and generation capabilities in a wide range of domains [36, 6, 11]. However, in NLP, the up-stream model (i.e., the model to be fine-tuned) is commonly a LLM with millions of parameters, which makes them computationally expensive to fine-tune. To address this, Parameter-efficient Fine-tuning (PEFT) methods have been developed, which fine-tune only a small set of the PLM's parameters (or an extra small set of parameters) to maintain performance while reducing computational cost. Some key PEFT methods include *Prefix-Tuning* [25], which adds a small task-specific vector to the input to adapt the frozen PLM to new downstream tasks; *LoRA* [19], which injects trainable low-rank matrices into the transformer architecture while keeping the pre-trained weights frozen, with *AdaLoRA* [56] extending this by adaptively allocating the rank budget during training; BitFit [55], which fine-tunes only the bias terms in each layer of a transformer-based PLM; and *(IA)*^{3} [27], which scales intermediate activations with learned vectors.

Explainability of Neural Models Several approaches have been proposed in the literature to bring a degree of explainability to the predictions of neural models, using different forms of explanations, such as (1) Feature-based explanations [38, 43, 53, 42], (2) Natural Language Explanations [8, 33, 23, 31], (3) Counterfactual explanations [2], and (4) Surrogate explanations [3]. In this paper, we focus on models that provide NLEs, i.e. free-form text stating the reasons behind a prediction. Being in natural language, NLEs should be easy to interpret by humans and more expressive than other types of explanations [50, 7, 21]. NLEs have been applied to several domains such as question answering [34], natural language inference [8], recommendation systems [10], reinforcement learning [15], medical imagining [23], visual-textual reasoning [17, 22, 31], and solving mathematical problems [26]. To enable neural models to generate accurate NLEs, the prevalent method involves annotating predictions with human-written explanations and training models to treat NLE generation as a sequence generation task [8]. Due to the high cost and time required to collect extensive datasets of human-written NLEs, methods such as [54] aims to fine-tune a PLM with a few NLEs but with abundant task-specific labels. More recently, [32] introduced the FEB benchmark for few-shot learning of NLEs and a prompt-based fine-tuning strategy, which we serves as the baseline in this study.

3 SPARSEFIT

We propose SPARSEFIT, an efficient few-shot NLE training strategy that fine-tunes only a subset of parameters in a LLM. SPARSEFIT is inspired by (1) FEB [32], who used fine-tuning and prompts to do few-shot learning of labels and NLEs; and (2) BitFit [55], who showed that fine-tuning only the

SparseFit		ComVE	ECQA	SBIC	e-SNLI	**Avg**
Baseline	Acc.	**80.5** ±4.5	57.6 ±2.6	**70.1** ±3.4	84.8 ±2.6	73.3 ±3.3
(100.00%)	nBERTs	**74.2** ±4.2	51.7 ±2.4	**67.8** ±3.3	76.9 ±2.5	67.7 ±3.1
Decoder	Acc.	67.3 ±6.0 ▽	58.5 ±2.6	66.8 ±3.1 ▽	86.6 ±1.7 ▽	69.8 ±3.4
(54.60%)	nBERTs	61.7 ±5.5 ▽	**52.3** ±2.4 ▽	64.7 ±2.7	**78.3** ±1.6 ▽	64.3 ±3.0
Encoder	Acc.	72.6 ±6.1 ▽	53.2 ±3.6 ▽	62.4 ±6.5 ▽	79.0 ±3.4 ▽	66.8 ±4.9
(40.95%)	nBERTs	67.1 ±5.7	47.2 ±3.2 ▽	58.7 ±6.5 ▽	72.4 ±3.2 ▽	61.3 ±4.6
Dense.wo	Acc.	61.3 ±4.4 ▽	56.1 ±2.1 ▽	62.4 ±2.6 ▽	84.0 ±1.9	65.9 ±2.8
(27.29%)	nBERTs	56.4 ±4.1 ▽	0.0 ±0.0 ▽	59.8 ±2.6 ▽	74.7 ±2.6 ▽	47.7 ±2.3
Self-attention (KQV)	Acc.	76.2 ±4.4 ▽	56.9 ±3.0	69.9 ±3.8	83.3 ±2.4 ▽	71.6 ±3.4
(20.47%)	nBERTs	70.3 ±4.0 ▽	50.2 ±2.7 ▽	67.4 ±3.9 ▽	76.1 ±2.2 ▽	66.0 ±3.2
LM head + Attention.Q	Acc.	74.8 ±5.0 ▽	55.4 ±2.7 ▽	67.1 ±5.2 ▽	82.8 ±3.0 ▽	70.0 ±4.0
(11.28%)	nBERTs	69.0 ±4.6	43.7 ±4.3 ▽	64.5 ±5.5	75.8 ±2.8 ▽	63.2 ±4.3
LM head	Acc.	15.6 ±1.3 ▽	58.9 ±2.3 ▽	0.2 ±0.2 ▽	**86.7** ±1.8 ▽	40.3 ±1.4
(4.46%)	nBERTs	0.0 ±0.0 ▽	0.0 ±0.0 ▽	0.2 ±0.2 ▽	0.0 ±0.0 ▽	0.0 ±0.0
LayerNorm + Attention.Q	Acc.	74.9 ±5.3 ▽	55.8 ±3.1 ▽	67.0 ±4.4 ▽	82.6 ±2.7 ▽	70.1 ±3.9
(6.84%)	nBERTs	69.0 ±4.8	45.9 ±3.7 ▽	64.3 ±4.7	75.6 ±2.5 ▽	63.7 ±3.9
Attention.K	Acc.	48.8 ±2.8 ▽	56.7 ±2.5 ▽	0.2 ±0.2 ▽	19.6 ±11.5 ▽	31.3 ±4.3
(6.82%)	nBERTs	19.4 ±10.0 ▽	0.0 ±0.0 ▽	0.1 ±0.2 ▽	0.2 ±0.3 ▽	4.9 ±2.6
Attention.Q	Acc.	74.8 ±5.1 ▽	55.5 ±3.2 ▽	66.9 ±4.6 ▽	82.8 ±2.6 ▽	70.0 ±3.8
(6.82%)	nBERTs	68.9 ±4.7	43.4 ±4.8 ▽	64.2 ±4.8	75.8 ±2.3 ▽	63.1 ±4.2
Attention.V	Acc.	55.5 ±3.0 ▽	53.1 ±2.8 ▽	30.1 ±10.2 ▽	84.2 ±2.0	55.7 ±4.5
(6.82%)	nBERTs	51.0 ±2.8 ▽	0.0 ±0.0 ▽	30.1 ±10.2 ▽	71.7 ±3.4 ▽	38.2 ±4.1
LayerNorm	Acc.	34.3 ±2.4 ▽	**59.0** ±2.4 ▽	0.3 ±0.3 ▽	86.6±1.8 ▽	45.0 ±1.7
(0.02%)	nBERTs	0.0 ±0.0 ▽	0.0 ±0.0 ▽	0.2 ±0.2 ▽	0.0 ±0.0 ▽	0.1 ±0.1

Table 1: Summary of top-performing SparseFit configurations for `T5-large`, evaluating them based on the **accuracy** metric and the normalized BERTScore (**nBERTs**) across 60 few-shot train-validation splits. The percentages of fine-tuned weights for each configuration are indicated in brackets. The highest metric for each dataset is highlighted in **bold**, the best performance without considering parameter count is in blue, and the optimal setting considering fine-tuned parameters is in green. The trade-off between parameters and performances was computed using $(1 - \%\text{params}) \times$ nBERTs). Significance was tested via mean t-test against baseline: $\triangledown$ represents a p-value lower than 10^{-2}.

bias terms in a PLM leads to competitive performance. We extend BitFit by exploring the fine-tuning of different components (i.e., layers or blocks) in the PLM's architecture. In particular, we study the self-rationalization performance after fine-tuning varios components in the transformer-baseed model, such as encoder and decoder blocks, language model head, self-attention layers, feed-forward networks, and normalization layers. This approach aims to identify fine-tuning guidelines for achieving competitive performance with minimal parameter updates. Notice that when fine-tuning any component, or pair of components, we freeze all other PLM's parameters and train the LM to conditionally generate a text in the form of *"[label] because [explanation]"*.

Encoder The T5 model's encoder consists of multiple transformer blocks (N), each containing self-attention, position-wise fully connected layers, and layer normalization, accounting for approximately 41% of the model's parameters. The number of blocks depends on the T5 variant (12 blocks for `T5-base`, 24 for `T5-large`, and 36 for `T5-3B`).

Decoder The decoder accounts for roughly 54% of T5 model parameters and it includes an additional encoder-decoder attention layer that attends to the encoded input sequence.

LM Head: The language modeling head, responsible for text generation, makes up roughly 5% of the parameters and it is located on top of the decoder.

Attention Layer: The self-attention layers, with parameters for query, key, and value matrices are also explored. There are three types of parameters in the self-attention layer, namely, for computing the *query matrix* Q, the *key matrix* K, and the *value matrix* V. We propose to explore the fine-tuning of each self-attention matrix as a possible SparseFit configuration. We also explore fine-tuning the *entire Self-attention Layer* (Q, K, V). On average, the percentage of trainable parameters associated with each matrix accounts for roughly 6% of model parameters.

Layer Normalization: The Normalization Layer is intended to enhance training speed and represents about 0.2% of the parameters [4]. The T5 model includes two *Layer Normalization* components per block, one after the self-attention layer and one after the feed-forwards networks.

	FLOPS		ComVE	ECQA	SBIC	e-SNLI	Avg
SparseFit (Att.Q+LN)	**2.37e14**	Acc.	**74.86** ±5.27	55.81 ±3.12	**66.99** ±4.4	82.62 ±2.73	**70.07** ±3.88
(6.84%)		nBERTs	**69.02** ±4.83	45.88 ±3.72	**64.29** ±4.7	75.63 ±2.51	**63.7** ±3.94
AdaLoRA	1.48e15	Acc.	69.66 ±3.47 ▽	46.60 ±4.02 ▽	61.80 ±2.74 ▽	84.50 ±1.95 ▽	65.64 ±3.05
(1.15%)		nBERTs	64.06 ±3.19 ▽	41.22 ±3.65 ▽	58.91 ±2.86 ▽	**77.43** ±1.79 ▽	60.41 ±2.87
LoRA (Att.KQVO, Rank=4)	2.75e14	Acc.	68.96 ±3.68 ▽	39.04 ±4.06 ▽	62.66 ±3.46 ▽	84.05 ±1.81▽	63.68 ±3.25
(0.32%)		nBERTs	63.48 ±3.39 ▽	33.52 ±3.76 ▽	59.80 ±3.66 ▽	77.04 ±1.66 ▽	58.46 ±3.12
(IA)^3	2.74e14	Acc.	58.53 ±2.32 ▽	**59.14** ±2.36 ▽	48.08 ±0.81 ▽	**86.64** ±1.85 ▽	63.10 ±1.84
(0.07%)		nBERTs	53.87 ±2.15 ▽	**48.08** ±1.92 ▽	48.06 ±0.80 ▽	72.18 ±1.54 ▽	55.55 ±1.60

Table 2: Performance comparison between SparseFit and other PEFT strategies. We report the average and the standard deviation over the 60 few-shot train-validation splits for the **accuracy** metric and the normalized BERTScore (**nBERTs**). The highest metric for each dataset is highlighted in **bold**. Significance testing was assessed via mean t-test, with ▽ indicating a p-value lower than 10^{-2}.

4 Experiments

Datasets We follow the FEB benchmark for few-shot learning of NLEs [32] and consider four NLE datasets: e-SNLI for natural language inference [8], ECQA for multiple-choice question answering [1], ComVE for commonsense classification [49], and SBIC for offensiveness classification [39].

Few-shot Learning Data Splits Following the few-shot evaluation protocol used by [32], we conducted our experiments using the same 60 train-validation splits. Each experiment included 48 training examples and 350 validation examples. The number of training examples per label varied across datasets: e-SNLI had 16, ECQA 48, SBIC 24, and omVE 24; always totaling 48 training examples for all datasets.

Training Procedure Following [32], we fine-tune T5 [36] and UnifiedQA [24]. Depending on the setup, the gradients are activated for specific parameters (SparseFit) or the entire model (baseline). Additionally, we also adapted LoRA [19], AdaLoRA [56] and (IA)^3 [27]; to compare with other PEFT baselines. For SparseFit configurations, we fine-tuned each component (or pair) for 25 epochs with a batch size of 4 samples, using AdamW optimizer [30] with a fixed learning rate of 0.00003. Conditional text generation is used for inference on the validation set. Training and evaluation were run on an NVIDIA P100, and took 23.2min, on average.

Automatic Evaluation The evaluation considers the task accuracy and the quality of the generated NLEs. Following [32], we use the BERTScore [57] to evaluate the quality of the NLEs. [22] found that the BERTScore correlates the best with human evaluation in NLEs. We compute a **normalized BERTScore** that assigns a zero score to empty NLEs, or NLEs of incorrectly predicted samples. We report the averages and standard deviations of the accuracy and the normalized BERTScore over 60 train-validation splits for each fine-tuning configuration.

Human Evaluation In addition to the normalized BERTScore, we conducted a small-scale human evaluation to assess the quality of NLEs for the best-performing SparseFit configurations. We selected NLEs from the first 30 correctly predicted samples in each validation set (balanced to the number of classes). Our human evaluation framework follows those of [22, 32]. For the NLE quality assessment, each annotator is asked to answer: *"Does the explanation justify the answer?"* and select one of four possible answers: *yes*, *weak yes*, *weak no*, or *no*. Moreover, we also ask the annotators to identify the main shortcomings, if any, of the generated NLEs. As in [22], we compute a numerical quality score for NLEs by mapping the four answers as follows: yes $\rightarrow 1$, weak yes $\rightarrow \frac{2}{3}$, weak no $\rightarrow \frac{1}{3}$, and no $\rightarrow 0$; and averaging them per model.

4.1 Results

To evaluate SparseFit, we computed the task accuracy and the quality of the generated NLEs. Among the sixty-two (62) SparseFit possible configurations (including single layers plus pairs of layers), for space reasons, we present only the top configurations based on generalization properties. Comprehensive results for all configurations are shown in Appendix C.

		ComVE	ECQA	SBIC	e-SNLI	Avg
Full Fine-tuning	Acc.	63.71 ±9.14	11.14 ±3.14	**63.86** ±1.86	34.91 ±0.43	43.41 ±3.64
(100%)	nBERTs	55.93 ±9.16	9.46 ±2.79	**57.42** ±1.32	28.62 ±0.84	37.86 ±3.53
SparseFit (Att.Q+LN)	Acc.	**68.03** ±8.24	**24.53** ±3.34	57.90 ±1.70	**40.10** ±4.03	**47.64** ±4.33
(7.97%)	nBERTs	**58.67** ±7.20	**20.60** ±2.83	50.41 ±2.72	**34.32** ±3.50	**41.00** ±4.06
AdaLoRA	Acc.	64.23 ±2.86	13.04 ±2.09	57.29 ±1.86	38.15 ±4.22	43.18 ±2.76
(0.30%)	nBERTs	56.16 ±2.77	11.13 ±1.79	50.63 ±0.33	33.48 ±3.71	37.85 ±2.15

Table 3: Performance comparison between SparseFit and other PEFT strategies for **Llama 2-7B**. We report the average and the standard deviation over the 60 few-shot splits for the **accuracy** metric and the normalized BERTScore (**nBERTs**). The highest metric for each dataset is highlighted in **bold**. Significance testing was assessed via mean t-test, with $\triangledown$ indicating a p-value lower than 10^{-2}.

Task Performance We present in Table 1 the accuracy performance for various SparseFit settings for `T5-large`, highlighting that some SparseFit configurations with very few fine-tuned parameters outperform the baseline (i.e., full fine-tuning). For instance, fine-tuning the *Normalization Layer* (*LayerNorm*) (0.02% of the model's parameters) yields better task performance for two out of four datasets Furthermore, we consistently see that if two SparseFit configurations achieve good generalization results, combining them by jointly fine-tuning both components produces significantly better results than each configuration in isolation. Results for `T5-base` and `T5-3b` are shown in Appendix C. The results indicate that larger LMs consistently achieve higher task accuracy across all datasets, although the performance gap between `T5-large` and `T5-3b` is small (<7%) despite a substantial increase in trainable parameters (∼5x).

NLE Quality Recall that the LM is fine-tuned to conditionally generate a text in the form of *"[label] because [explanation]"*. Table 1 shows the normalized BERTScore for selected SparseFit settings as a proxy to evaluate how good the NLEs generated after the explanation token (i.e. *"because"*) is. Overall, it can be observed that SparseFit settings with few trainable parameters (<10%), such as the *Self-attention Query* (*Att.Q*), *LM Head + Attention Query* (*Att.Q+LMhead*), and *Layer Norm + Attention Query* (*Att.Q+LN*), perform competitively against the baseline. Moreover, we can see that the best quality of NLEs is achieved for SparseFit combinations of two or more types of components (e.g., *LayerNorm + Att.Q*). The performance gap between most of the SparseFit configurations and the baseline does not exceed 15% for all the datasets, even for very sparse fine-tuning strategies.

Unexpectedly, some SparseFit configurations with high task accuracy (e.g., *LayerNorm*) have a low normalized BERTScore due to limitations with generating the explanation token after the generated label token. We investigate more about this behavior in Section 4.2. Results for other T5 model sizes (i.e. `T5-base`, `T5-large` and `T5-3b`) are shown in Appendix C.1. We found that the normalized BERTScore consistently increases with the size of the LM. Remarkably, the best SparseFit configurations for `T5-large` also achieve the best performance when fine-tuning `T5-base`, but they are slightly different for `T5-3b`.

Other PEFT Baselines To compare SparseFit with other PEFT baselines, we also evaluated LoRA [19], AdaLoRA [56] and (IA)^3 [27] for NLEs. The comparison with other PEFT strategies (Table 2) shows that, on average, SparseFit outperforms the other strategies in downstream performance and NLE quality. While these PEFT methods tune less than 20% of the parameters updated by SparseFit, the quality of NLEs is considerably better for SparseFit for two out of four datasets. Additionally, SparseFit exhibits the lowest FLOPS, likely due to SparseFit neither introduces additional model parameters nor increases the model's architectural complexity.

Larger Language Models To assess the performance of SparseFit in both larger language models and different architectures we also conducted experiments on `Llama 2-7B`. Notice that SparseFit approach is applicable to any architecture (not only the `T5` encoder-decoder) since it focuses on identifying the optimal layer(s) to fine-tune, independent of the model's structure. Experiments on the larger decoder-only model `Llama 2-7B` show that SparseFit outperforms the best PEFT baseline (AdaLora) in terms of both predictive accuracy and NLE quality (See Table 3). Specifically, the best SparseFit have on average roughly 5% better NLE quality across all datasets.

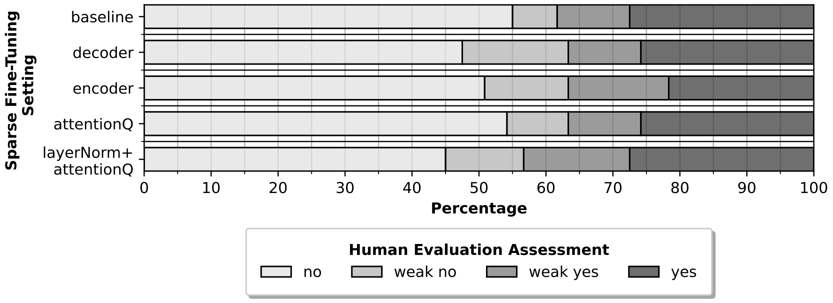

Figure 1: Illustration of plausibility score given by human annotators to the quality of the NLEs generated by different SPARSEFIT configurations. Annotators evaluated the explanations by answering the question: *"Does the explanation justify the answer?*

Human Evaluation					
	e-SNLI	ECQA	SBIC	ComVE	**Avg**
Full Fine-tuning	29.63 (0.43)	**41.92** (0.23)	54.44 ±0.7	21.67 (0.22)	36.91
AdaLora 1.15%	23.33 (0.34)	34.44 (0.26)	**61.11** (0.69)	23.34 (0.25)	35.55
SPARSEFIT Att.Q+LN	**38.27** (0.34)	31.31 (0.26)	58.89 (0.69)	**40.00** (0.25	**42.12**

Table 4: Average scores given by human annotators for the best performing SPARSEFIT and other PEFT baselines of `T5-large`. The best results are in **bold**. In brackets, we show the inter-annotator agreement score.

Human Evaluation					
	e-SNLI	ECQA	SBIC	ComVE	**Avg**
Full Fine-tuning	17.78	38.89	47.78	40.00	36.11
AdaLora 1.18%	**50.00**	52.22	**71.11**	55.56	57.22
SPARSEFIT Att.Q+LN	41.11	**73.33**	68.89	**70.00**	**63.33**

Table 5: Average scores given by human annotators for the best performing SPARSEFIT and other PEFT baselines of `Llama-2-7B`. The best results are in **bold**.

Human Evaluation Table 4 present the distribution of the scores assigned by human annotators to the quality of NLEs generated by the best SPARSEFIT strategies and the top-performing PEFT baseline (AdaLora). The inter-annotator agreement was measured using the *Cohen'* κ metric [12]. Overall, SPARSEFIT-generated NLEs are significantly better than those of the baseline and AdaLoRA for 2 out of 4 tasks. For the other two tasks, AdaLoRA produces better NLEs by a very smaller margin. On average, the NLEs of SPARSEFIT are roughly 8% better than NLEs of AdaLoRA and 6% better than full fine-tuning NLEs. However, the human evaluation shows that the generated NLEs often failed to adequately explain the predictions, with roughly half of the NLEs that do not justify the answer (no matter what fine-tuning strategy is used). We detail the shortcomings and limitations of generated NLEs in Section 4.2. Finally, Table 5 shows the human evaluation results for the best SPARSEFIT configuration and other PEFT baselines when applied to `Llama2-7B`. On average, the NLEs of SPARSEFIT are roughly 6% better than NLEs of AdaLoRA and 21% better than full fine-tuning NLEs for `Llama2-7B`.

4.2 Discussion

Analysis of the Generated NLEs In Figure 2, we show a collection of examples of the generated NLEs by the baseline and the best performing SPARSEFIT configurations. As in previous works [8, 22, 32], we only show examples where the label was correctly predicted by the model since we do not expect a model that predicted a wrong label to generate a correct explanation.

NLE Shortcomings Figure 3 depicts the frequency histogram of shortcomings for the baseline and the top-performing SPARSEFIT strategies. It can be observed that the most common shortcomings are *Lack of explanation*, *Nonsensical*, and *Incomplete explanation*. For the best SPARSEFIT configuration (i.e. *Att.Q+LN*), *Incomplete explanation* is the most frequent shortcoming. A detailed breakdown of these shortcomings for each dataset is provided in Appendix C.4.

Inter-Annotator Agreement As shown in Table 4, the agreement between annotators is moderately low for the set of evaluated NLEs, with differing scores assigned to 181 out of 600 NLEs. The ECQA

Premise	2 woman walking down the street next to graffiti
Hypothesis	The 2 woman are running
Label	Contradiction
Explanation	You can't be running, if you are walking. Walking is different from running.
	Generated Explanation
Baseline	walking and running are different actions
Decoder	walking and running are different verbs
Att.Q	the 2 woman are walking not running
Att.Q + Head	walking and running are not the same
Att.Q + LN	walking is not running

Question	The teacher doesn't tolerate noise during a test in their what?
Answer Choices	A) Theatre B) Bowling C) Alley D) Factory E) Classroom
Answer	Classroom
Explanation	Test is done in classroom. A teacher when in classroom does not tolerate noise.
	Generated Explanation
Baseline	noise is harmful for students. classroom is a place where students can take test.
Decoder	classroom is a place where students are studying. there is no noise in classroom.
Att.Q	classroom is a place where people can learn. noise is a problem in classroom.
Att.Q + Head	noise is harmful to students
Att.Q + LN	classroom is a place where people can learn. noise is a distracting thing.

Post	@cmencarini: I dont trust girls ; cant trust these hoes
Label	Offensive
Explanation	This post implies that women are not honest. This post implies that women are not trustworthy. This post implies that women are all women are hoes.
	Generated Explanation
Baseline	this post implies that women aren't important
Decoder	this post implies that women are prostitutes
Att.Q	this post implies that women aren't trustworthy
Att.Q + Head	this post implies that women aren't trustworthy
Att.Q + LN	this post implies that women are dirty

Choice 1	A Latino can communicate with someone from Spain
Choice 2	French and Latinos can communicate easy
Label	Choice 2
Explanation	French can't communicate with Latinos because they don't speak the same language. French and Latinos belongs to two different countries with two different languages.
	Generated Explanation
Baseline	french and latinos are two different languages
Decoder	french and latinos are two different people and don't speak the same language
Att.Q	french is not a common language in latinamerica
Att.Q + Head	french and latinos cannot communicate easily.
Att.Q + LN	french and latinos cannot communicate easily

Figure 2: Examples of generated NLEs for e-SNLI (Green), ECQA (Blue), SBIC (Red), and ComVE (Yellow).

dataset exhibits the most significant differences (63), while the SBIC dataset is the most uniform, with 17 differences. The variation between annotators can result from three potential perceptual reasons: (1) *perceptual disagreement*, where annotators could not objectively identify the difference between two adjacent answers (i.e. *Weak Yes* vs *Weak No*). (2) *positionality disagreement*, which could alter how the annotators perceive the outcomes of the algorithm due to their race, gender, and other socioeconomic identity factors. and (3) *expectation disagreement*, which may cause an annotator to be more strict on the characteristics that make an explanation complete and accurate [5, 35]. An extensive collection of examples of perceptual disagreement, positionality disagreement, and expectation disagreement samples are in Appendix C.5.

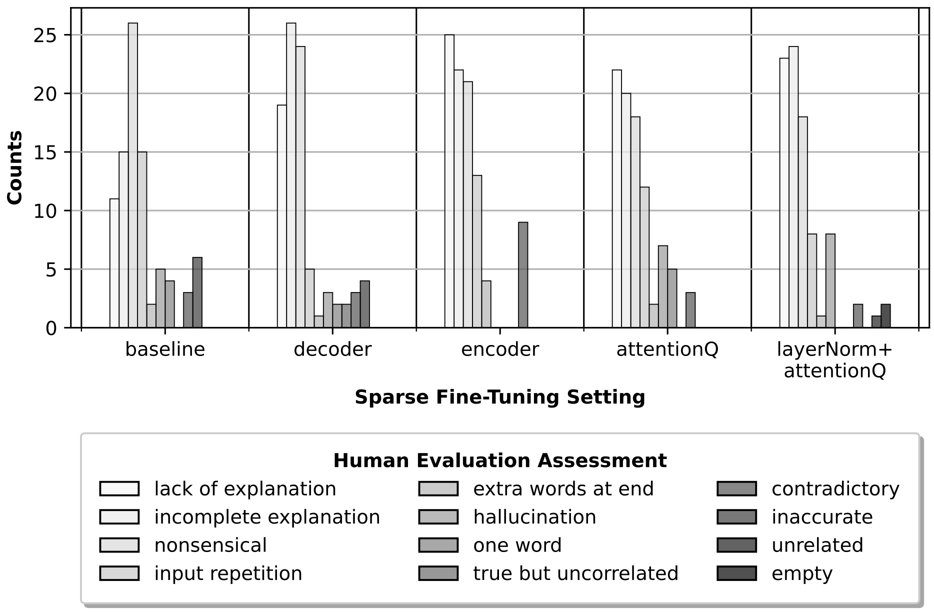

Figure 3: Histogram of the shortcomings of the generated NLEs for the baseline and the performing SparseFit configurations aggregated for all the datasets.

Generation of Empty NLEs Some SparseFit configurations, such as *LayerNorm*, exhibits high task performance but often generate empty NLEs, particularly for the e-SNLI and ECQA datasets. This discrepancy may arise because generating NLEs is inherently more complex than solving downstream tasks, potentially requiring fine-tuning more significant portions of the model parameters. Another explanation can be found by analyzing the pre-training tasks of the PLM. For example, T5 was pre-trained on the MNLI dataset [52], which is composed of NLI instances without NLEs, leading the model to predict labels without generating explanations when only a small subset of parameters is updated. Similar reasoning may be concluded for ECQA since UnifiedQA was pre-trained on CommonsenseQA [45], which is composed of samples with only the answer for the multiple-choice question.

5 Summary

We introduced SparseFit, a strategy that combines sparse fine-tuning with prompt-based learning to train NLE models in a few-shot setup. SparseFit consistently performs competitively while updating only a minimal subset of parameters (i.e. $\sim$6.8% for the *Self-attention Query + Layer Normalization*, configuration). We found that the sparse fine-tuning of `T5-large` consistently achieves better performance than fine-tuning `T5-base` and is slightly worse ($< 5\%$) than `T5-3b`, no matter the SparseFit strategy. Moreover, the best three configurations on `T5-base` are achieved by the same set of SparseFit configurations found for `T5-large`. Compared to other PEFT techniques, SparseFit produces better average predictive accuracy and NLE quality. Future work includes build upon SparseFit by, e.g. relying on soft prompts rather than hard prompts.

Limitations

Although generating natural language explanations is a fervid research area, there is still no guarantee that such explanations accurately reflect how the model works internally [51, 9]. For example, the fact that the generated explanation seems reasonable does not mean that the model does not rely on protected attributes and spurious correlations in the training data to produce its predictions. As such, we still recommend being careful to use self-explanatory models in production, as they can capture potentially harmful biases from the training data, even though these are not mentioned in the explanations.

References

[1] Shourya Aggarwal, Divyanshu Mandowara, Vishwajeet Agrawal, Dinesh Khandelwal, Parag Singla, and Dinesh Garg. Explanations for commonsenseqa: New dataset and models. In *Workshop on Commonsense Reasoning and Knowledge Bases*, 2021.

[2] Arjun R. Akula, Shuai Wang, and Song-Chun Zhu. Cocox: Generating conceptual and counterfactual explanations via fault-lines. In *AAAI*, pages 2594–2601. AAAI Press, 2020.

[3] Ahmed M. Alaa and Mihaela van der Schaar. Demystifying black-box models with symbolic metamodels. In *NeurIPS*, pages 11301–11311, 2019.

[4] Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *CoRR*, abs/1607.06450, 2016.

[5] Brian Bourke. Positionality: Reflecting on the research process. *The qualitative report*, 19(33):1–9, 2014.

[6] 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. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020.

[7] Oana-Maria Camburu, Eleonora Giunchiglia, Jakob Foerster, Thomas Lukasiewicz, and Phil Blunsom. The struggles of feature-based explanations: Shapley values vs. minimal sufficient subsets. In *AAAI 2021 Workshop on Explainable Agency in Artificial Intelligence*, 2021.

[8] Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. e-SNLI: Natural language inference with natural language explanations. *Advances in Neural Information Processing Systems*, 31, 2018.

[9] Oana-Maria Camburu, Brendan Shillingford, Pasquale Minervini, Thomas Lukasiewicz, and Phil Blunsom. Make up your mind! adversarial generation of inconsistent natural language explanations. In *ACL*, pages 4157–4165. Association for Computational Linguistics, 2020.

[10] Hanxiong Chen, Xu Chen, Shaoyun Shi, and Yongfeng Zhang. Generate natural language explanations for recommendation. *CoRR*, abs/2101.03392, 2021.

[11] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: Scaling language modeling with pathways. *CoRR*, abs/2204.02311, 2022.

[12] Jacob Cohen. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46, 1960.

[13] Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. Unified language model pre-training for natural language understanding and generation. *Advances in Neural Information Processing Systems*, 32, 2019.

[14] Xuanyi Dong, Linchao Zhu, De Zhang, Yi Yang, and Fei Wu. Fast parameter adaptation for few-shot image captioning and visual question answering. In *Proceedings of the 26th ACM international conference on Multimedia*, pages 54–62, 2018.

[15] Upol Ehsan, Brent Harrison, Larry Chan, and Mark O Riedl. Rationalization: A neural machine translation approach to generating natural language explanations. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, pages 81–87, 2018.

[16] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, volume 70 of *Proceedings of Machine Learning Research*, pages 1126–1135. PMLR, 2017.

[17] Lisa Anne Hendricks, Ronghang Hu, Trevor Darrell, and Zeynep Akata. Grounding visual explanations. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[18] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR, 2019.

[19] Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

[20] Gautier Izacard, Patrick S. H. Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. Few-shot learning with retrieval augmented language models. *CoRR*, abs/2208.03299, 2022.

[21] Harmanpreet Kaur, Harsha Nori, Samuel Jenkins, Rich Caruana, Hanna Wallach, and Jennifer Wortman Vaughan. Interpreting interpretability: Understanding data scientists' use of interpretability tools for machine learning. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, 2020.

[22] Maxime Kayser, Oana-Maria Camburu, Leonard Salewski, Cornelius Emde, Virginie Do, Zeynep Akata, and Thomas Lukasiewicz. e-ViL: A dataset and benchmark for natural language explanations in vision-language tasks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1244–1254, 2021.

[23] Maxime Kayser, Cornelius Emde, Oana-Maria Camburu, Guy Parsons, Bartlomiej Papiez, and Thomas Lukasiewicz. Explaining chest x-ray pathologies in natural language. In Linwei Wang, Qi Dou, P. Thomas Fletcher, Stefanie Speidel, and Shuo Li, editors, *Medical Image Computing and Computer Assisted Intervention – MICCAI 2022*, pages 701–713, Cham, 2022. Springer Nature Switzerland.

[24] D. Khashabi, S. Min, T. Khot, A. Sabhwaral, O. Tafjord, P. Clark, and H. Hajishirzi. Unifiedqa: Crossing format boundaries with a single qa system. *EMNLP - findings*, 2020.

[25] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *ACL/IJCNLP (1)*, pages 4582–4597. Association for Computational Linguistics, 2021.

[26] Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *ACL (1)*, pages 158–167. Association for Computational Linguistics, 2017.

[27] Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin A Raffel. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *Advances in Neural Information Processing Systems*, 35:1950–1965, 2022.

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

[29] Robert L. Logan, Ivana Balazevic, Eric Wallace, Fabio Petroni, Sameer Singh, and Sebastian Riedel. Cutting down on prompts and parameters: Simple few-shot learning with language models. In *ACL (Findings)*, pages 2824–2835. Association for Computational Linguistics, 2022.

[30] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[31] Bodhisattwa Prasad Majumder, Oana-Maria Camburu, Thomas Lukasiewicz, and Julian Mcauley. Knowledge-grounded self-rationalization via extractive and natural language explanations. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 14786–14801. PMLR, 17–23 Jul 2022.

[32] Ana Marasovic, Iz Beltagy, Doug Downey, and Matthew Peters. Few-shot self-rationalization with natural language prompts. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 410–424, Seattle, United States, July 2022. Association for Computational Linguistics.

[33] Ana Marasović, Chandra Bhagavatula, Jae sung Park, Ronan Le Bras, Noah A Smith, and Yejin Choi. Natural language rationales with full-stack visual reasoning: From pixels to semantic frames to commonsense graphs. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2810–2829, 2020.

[34] Sharan Narang, Colin Raffel, Katherine Lee, Adam Roberts, Noah Fiedel, and Karishma Malkan. Wt5?! training text-to-text models to explain their predictions. *CoRR*, abs/2004.14546, 2020.

[35] Ana Niño. Machine translation in foreign language learning: Language learners' and tutors' perceptions of its advantages and disadvantages. *ReCALL*, 21(2):241–258, 2009.

[36] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

[37] Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B Tenenbaum, Hugo Larochelle, and Richard S Zemel. Meta-learning for semi-supervised few-shot classification. In *International Conference on Learning Representations*, 2018.

[38] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should I trust you?": Explaining the predictions of any classifier. In *HLT-NAACL Demos*, pages 97–101. The Association for Computational Linguistics, 2016.

[39] Maarten Sap, Saadia Gabriel, Lianhui Qin, Dan Jurafsky, Noah A. Smith, and Yejin Choi. Social bias frames: Reasoning about social and power implications of language. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5477–5490, Online, July 2020. Association for Computational Linguistics.

[40] Victor Garcia Satorras and Joan Bruna Estrach. Few-shot learning with graph neural networks. In *International Conference on Learning Representations*, 2018.

[41] Roy Schwartz, Jesse Dodge, Noah A. Smith, and Oren Etzioni. Green AI. *Commun. ACM*, 63(12):54–63, 2020.

[42] Lei Sha, Oana-Maria Camburu, and Thomas Lukasiewicz. Learning from the best: Rationalizing prediction by adversarial information calibration. In *Proceedings of the 35th Association for the Advancement of Artificial Intelligence (AAAI)*, 2021.

[43] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *ICML*, volume 70 of *Proceedings of Machine Learning Research*, pages 3145–3153. PMLR, 2017.

[44] Jesús Solano, Lizzy Tengana, Alejandra Castelblanco, Esteban Rivera, Christian Lopez, and Martın Ochoa. A few-shot practical behavioral biometrics model for login authentication in web applications. In *NDSS Workshop on Measurements, Attacks, and Defenses for the Web (MADWeb'20)*, 2020.

[45] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. 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 4149–4158, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[46] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[47] Oriol Vinyals, Charles Blundell, Tim Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In *NIPS*, pages 3630–3638, 2016.

[48] Eric Wallace, Jens Tuyls, Junlin Wang, Sanjay Subramanian, Matt Gardner, and Sameer Singh. Allennlp interpret: A framework for explaining predictions of NLP models. In *EMNLP/IJCNLP (3)*, pages 7–12. Association for Computational Linguistics, 2019.

[49] Cunxiang Wang, Shuailong Liang, Yue Zhang, Xiaonan Li, and Tian Gao. Does it make sense? and why? a pilot study for sense making and explanation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4020–4026, Florence, Italy, July 2019. Association for Computational Linguistics.

[50] Sarah Wiegreffe and Ana Marasovic. Teach me to explain: A review of datasets for explainable natural language processing. *35th Conference on Neural Information Processing Systems (NeurIPS) Track on Datasets and Benchmarks*, 2021.

[51] Sarah Wiegreffe, Ana Marasović, and Noah A. Smith. Measuring association between labels and free-text rationales. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10266–10284, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[52] Adina Williams, Nikita Nangia, and Samuel Bowman. 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, 2018.

[53] Jinsung Yoon, James Jordon, and Mihaela van der Schaar. INVASE: instance-wise variable selection using neural networks. In *ICLR (Poster)*. OpenReview.net, 2019.

[54] Yordan Yordanov, Vid Kocijan, Thomas Lukasiewicz, and Oana-Maria Camburu. Few-shot out-of-domain transfer learning of natural language explanations. *Findings of the Association for Computational Linguistics: EMNLP*, 2022.

[55] Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *ACL (2)*, pages 1–9. Association for Computational Linguistics, 2022.

[56] Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adaptive budget allocation for parameter-efficient fine-tuning. *arXiv preprint arXiv:2303.10512*, 2023.

[57] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*, 2019.

A SPARSEFIT Graphical Representation

In this paper, we propose an efficient few-shot prompt-based training regime for models generating both predictions and NLEs on top of the T5 language model. To have a better understanding of the active trainable parameters in each SPARSEFIT configuration, we illustrate in Figure 4 a graphical representation of the T5 architecture with active parameters colored for the *Layer Normalization* sparse fine-tuning. After freezing the rest of the model (gray-colored layers), the percentage of parameters that could potentially be updated in the *Layer Normalization* is 0.02% of the entire model. Considering that the UNIFIEDQA model's architecture is the same as the one in T5, the interpretation of active parameters holds for UNIFIEDQA.

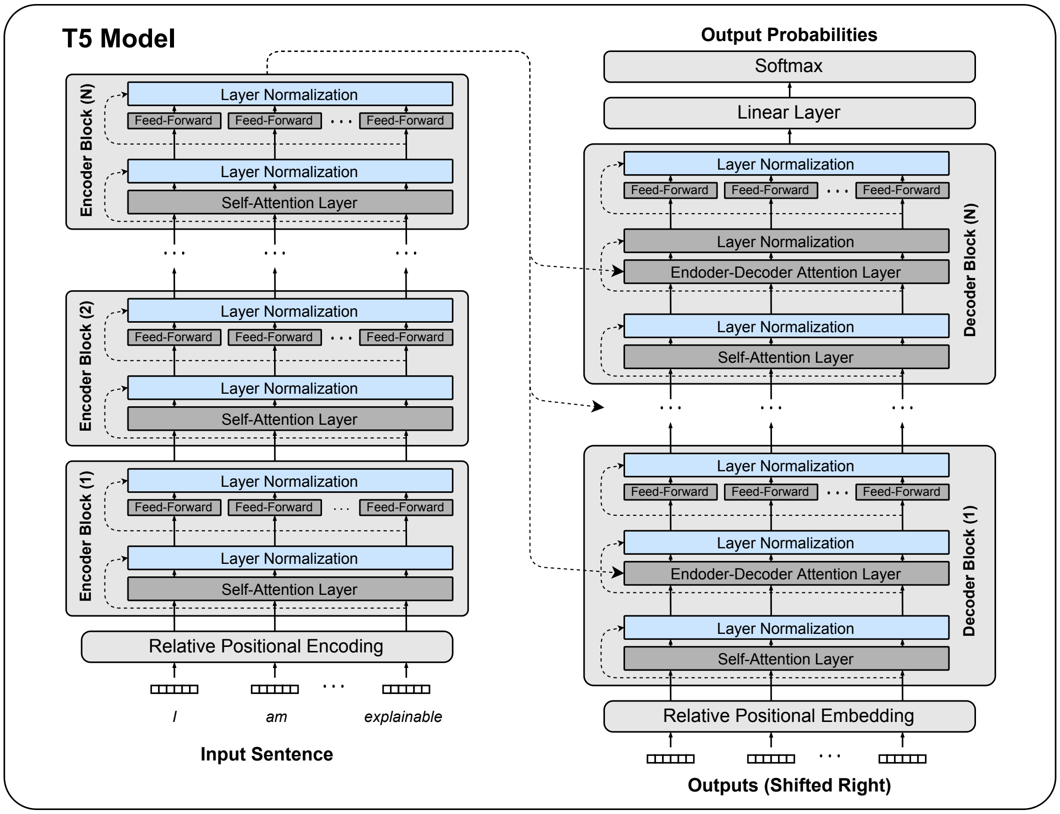

Figure 4: Illustration of the active trainable parameters in T5 when SPARSEFIT is performed over the layer normalization.

B Examples of Generated NLEs

This section shows a collection of examples of the generated NLEs by the baseline and the different sparse fine-tuning strategies considered in our approach. We show four examples for each dataset. Each example contains the generated NLE for the best performing SPARSEFIT configurations. As in previous works [8, 22, 32], we only show examples where the label was correctly predicted by the model (since we do not expect a model that predicted a wrong label to generate a correct NLE). Regarding the protocol for choosing the examples shown in this section, we have done a manual inspection of several possible examples, and we have chosen the more informative ones to conclude the strengths and the weaknesses of the generated NLEs. Notice that, due to the few-shot splits protocol (60 different train-validation splits), a single example could be predicted more than once for a single setup (i.e., the sample is in more than one validation set).

Premise	Two dogs play together on the bed.
Hypothesis	Dogs laying down on the floor, motionless.
Label	Contradiction
	Generated Explanation
Baseline	dogs can't be motionless and play together on the bed.
Decoder	playing together is different than lying down, motionless.
Encoder	dogs laying down on the floor are motionless.
Attention.Q	two dogs playing together on the bed are not laying down, motionless.
LM Head + Att.Q	two dogs playing together on the bed are not motionless.
LayNorm + Att.Q	two dogs playing together on the bed are not laying down, motionless.
Self-Attention	two dogs playing together on the bed are not motionless.

Premise	Man walking barefoot on the beach on what seems to be a very cloudy day
Hypothesis	It is going to rain soon.g
Label	Neutral
	Generated Explanation
Baseline	it does not look like it is going to rain soon
Decoder	not all cloudy means it is raining
Encoder	it does not mean it is going to rain soon
Attention.Q	not all clouds are bad
LM Head + Att.Q	not all cloudy days are going to rain soon
LayNorm + Att.Q	it does not mean it is going to rain soon
Self-Attention	not all cloudy days are going to rain soon

Premise	Number 13 kicks a soccer ball towards the goal during children's soccer game
Hypothesis	A player trying to score in a soccer game
Label	Entailment
	Generated Explanation
Baseline	number 13 is a player
Decoder	number 13 is a player trying to score
Encoder	number 13 is a player trying to score
Attention.Q	number 13 is a player
LM Head + Att.Q	number 13 is a player trying to score
LayNorm + Att.Q	number 13 is a player
Self-Attention	kicking a soccer ball towards the goal is similar to trying to score

Figure 5: Examples of generated NLEs for e-SNLI.

Question	What does a person who likes school also do?
Answer Choices	A) Fever B) Play dumb C) Read text D) Catch cold E) Enjoy learning
Answer	Enjoy learning
	Generated Explanation
Baseline	A school is a place where one can learn. it is a place where one can get knowledge and skills
Decoder	a person who likes school also does. a person who likes school also does
Encoder	enjoy learning is also a part of life of a person. a person who likes school also does enjoy learning
Attention.Q	school is a place where people learn. a person who likes school also likes to learn.
LM Head + Att.Q	school is a place where people learn. a person who likes school also does things like studying.
LayNorm + Att.Q	school is a place where people learn. a person who likes school also does things like studying.
Self-Attention	a person who likes school also does things like studying.

Question	When someone is watching a film that is a documentary, what are they hoping to gain?
Answer Choices	A) Like B) Insight C) Fear D) Being entertained E) Awareness
Answer	Insight
	Generated Explanation
Baseline	insight is the ability to gain knowledge through observation. when someone watches a documentary, they are hoping to gain insight.
Decoder	insight is the ability to gain knowledge through observation. people often seek out information through watching a documentary.
Encoder	insight is the ability to gain knowledge through observation.
Attention.Q	insight is a kind of knowledge that is gained through observation

Post	Watching conservacucks slobber all over the Western Wall makes me ill.
Label	Offensive
	Generated Explanation
Baseline	this post implies that southerners are lazy.
Decoder	this post implies that conservacucks are slobbering
Encoder	this post implies that conservacucks are disgusting
Attention.Q	this post implies that conservacucks are slobbering all over the western wall.
LM Head + Att.Q	tthis post implies that conservacucks are slobbering all over the western wall.
LayNorm + Att.Q	this post makes fun of conservacucks
Self-Attention	this post implies that conservacucks are dirty.

Post	Whats the difference between a training gun and a police officer? A training gun shoots blanks, an officer shoots blacks
Label	Offensive
	Generated Explanation
Baseline	this post implies that black people are criminals
Decoder	this post implies that blacks are the only color that will be killed
Encoder	this post implies that blacks are the victims of police brutality
Attention.Q	this post implies that blacks are more dangerous
LM Head + Att.Q	this post implies that blacks are more dangerous
LayNorm + Att.Q	this post implies that black people are criminals
Self-Attention	this post implies that blacks are inferior

Post	What's the worst than a nerd being Basketball Player A Nigga Becoming a Doctor
Label	Offensive
	Generated Explanation
Baseline	this post implies that black folks are less than white
Decoder	this post trivializes harm to victims
Encoder	this post implies that blacks are worse than nerd
Attention.Q	this post implies that blacks are better than nerds
LM Head + Att.Q	this post implies that blacks are inferior
LayNorm + Att.Q	this post implies that blacks are inferior
Self-Attention	this post implies that blacks are the only people who shoot training guns

Figure 7: Examples of generated NLEs for SBIC.

Choice 1	He used a towel to dry off after his shower.
Choice 2	He used an apple to dry off after his shower.
Label	Choice 2
	Generated Explanation
Baseline	apple is a fruit, so it cannot dry off.
Decoder	apple is a fruit and does not dry off
Encoder	an apple is not a very absorbent material
Attention.Q	an apple doesn't dry off as well as a towel
LM Head + Att.Q	an apple is a fruit and it doesn't dry off.
LayNorm + Att.Q	an apple is a fruit and it's not a towel, so it's not drying off as well.
Self-Attention	apples are a fruit, and they don't dry well on their own.

Choice 1	He walked his fish
Choice 2	He walked his dog
Label	Choice 1
	Generated Explanation
Baseline	fish have no legs
Decoder	fish cannot walk on water
Encoder	fish cannot walk
Attention.Q	fish are not able to walk
LM Head + Att.Q	fish don't walk
LayNorm + Att.Q	fish cannot walk
Self-Attention	fish cannot be walked

Choice 1	A ball is square and can roll
Choice 2	A ball is round and can roll
Label	Choice 1
	Generated Explanation
Baseline	a square ball cannot roll
Decoder	square balls are not rollable
Encoder	a ball is square cannot roll
Attention.Q	balls are round and can roll
LM Head + Att.Q	a ball is round and can roll.
LayNorm + Att.Q	a square ball cannot roll
Self-Attention	a ball can roll only in a round shape

Figure 8: Examples of generated NLEs for ComVE.

C SparseFit Full Results

This section shows the results in terms of task accuracy, and NLEs quality all configurations of SparseFit and for different model sizes (i.e. `T5-base`, `T5-large` and `T5-3b`). For each metric, we also break down the results by dataset.

C.1 Task Performance

Figure 9 depicts the distribution of the accuracy score for SparseFit configurations trained on top of `T5-large`. It can be observed that several SparseFit configurations exhibit similar performance as the baseline, particularly for ECQA and E-SNLI. The SparseFit configurations with the best task performance are *Decoder*, *Self-Attention KQV*, *Self-attention Query*, and *Layer Normalization*. Remarkably, the SparseFit configurations do not show a higher variance than the baseline across the 60 train-validation splits (inter-quartile range). Figure 10 depicts the distribution of the accuracy score for SparseFit configurations trained on top of `T5-3b`. It can be observed that all SparseFit configurations outperform the baseline. However, the best performance for `T5-3b` is achieved by the sparse fine-tuning of the *Self-attention Value Layer*. The results for `T5-base` can be observed in the breakdown done for each dataset.

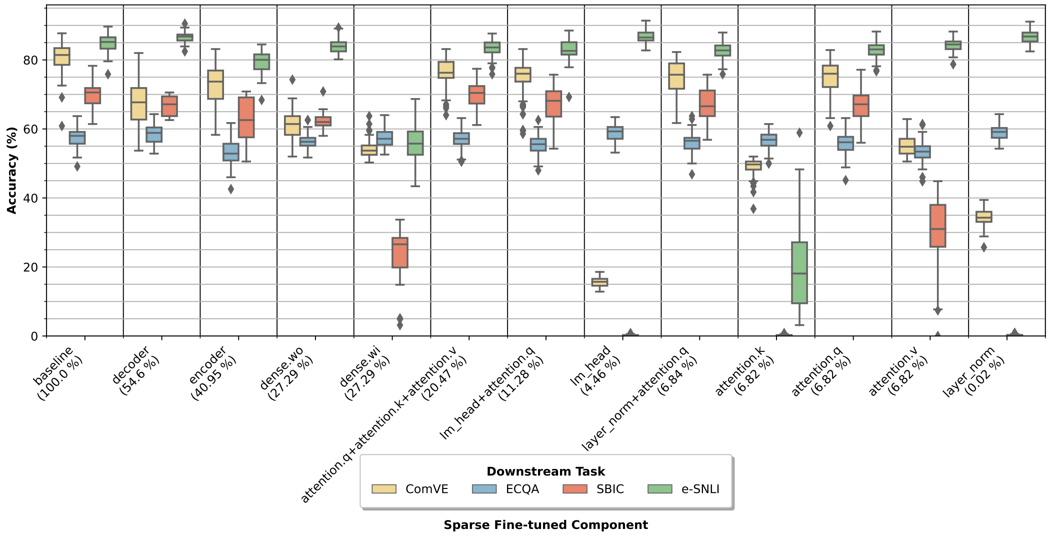

Figure 9: Distribution of the **accuracy** scores for different SPARSEFIT configurations for `T5-large`. The percentage of parameters fine-tuned for each configuration is shown between brackets.

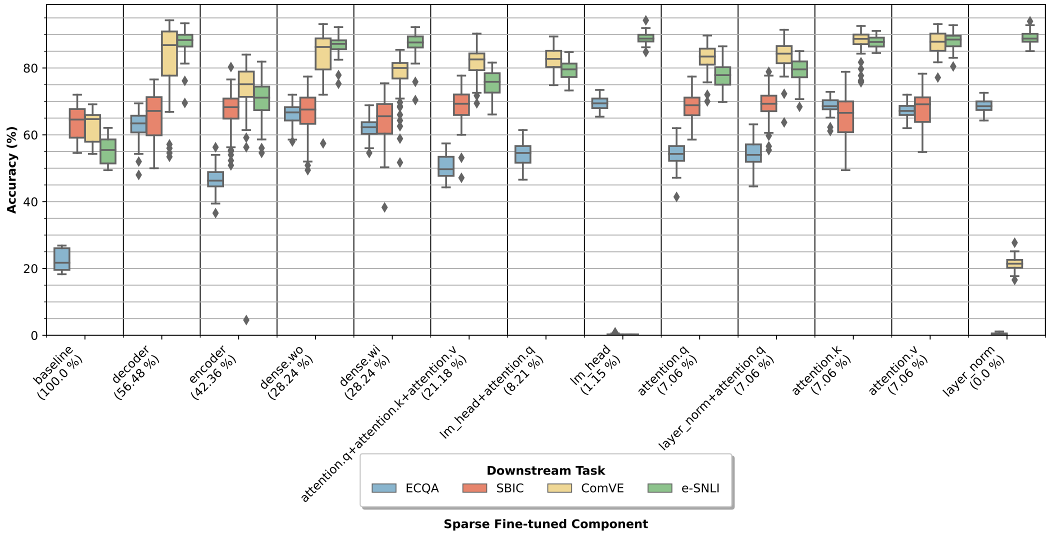

Figure 10: Distribution of the **accuracy** scores for different SPARSEFIT configurations for `T5-3b`. The percentage of parameters fine-tuned for each configuration is shown between brackets.

SPARSEFIT		ComVE	ECQA	SBIC	e-SNLI	**Avg**
LayerNorm + Attention.Q	Acc.	53.22 ±3.67 ▽	39.35 ±2.31 ▽	62.11 ±5.04 ▽	72.63 ±2.87 ▽	56.83 ±3.47
`T5-base`	nBERTs	48.77 ±3.37 ▽	0.0 ±0.0 ▽	59.45 ±5.47 ▽	64.81 ±3.13 ▽	43.26 ±2.99
LayerNorm + Attention.Q	Acc.	74.9 ±5.3 ▽	55.8 ±3.1 ▽	67.0 ±4.4 ▽	82.6 ±2.7 ▽	70.1 ±3.9
`T5-large`	nBERTs	69.0 ±4.8	45.9 ±3.7 ▽	64.3 ±4.7	75.6 ±2.5 ▽	63.7 ±3.9
LayerNorm + Attention.Q	Acc.	83.27 ±4.52 ▽	54.13 ±3.86 ▽	**68.87** ±4.86 ▽	79.16 ±3.72 ▽	71.36 ±4.24
`T5-3B`	nBERTs	75.83 ±4.14 ▽	48.31 ±3.46 ▽	65.86 ±5.07 ▽	71.27 ±3.44 ▽	65.32 ±4.03

Table 6: Summary of best performing SPARSEFIT configurations for *LayerNorm + Attention*. We report the average and the standard deviation over the 60 few-shot train-validation splits for the **accuracy** metric and the normalized BERTScore (**nBERTs**). In brackets are the percentages of fine-tuned weights for each SPARSEFIT configuration. We show in **bold** the setting with the highest metric for each dataset, in blue the highest performance among SPARSEFIT without considering the number of parameters, and in green the best-performing setting after considering the percentage of fine-tuned parameters. The trade-off between parameters and performances was computed using $(1 - \%\text{params})\times$ nBERTs). Significance testing was assessed via mean t-test compared with the baseline: ▽ represents a p-value lower than 10^{-2}.

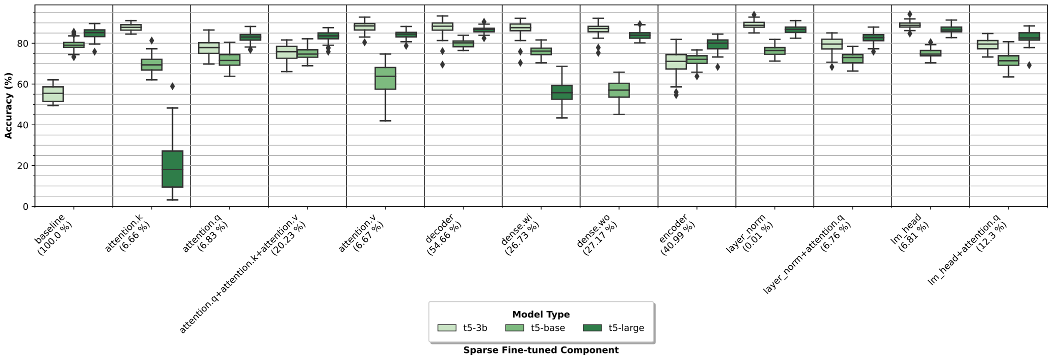

Figure 11: Distribution of the accuracies for different settings of SPARSEFIT for the **e-SNLI** dataset. For each model, the variation represents the overall performance in each of the 60 train-validation splits. The percentage of parameters fine-tuned for each setup is depicted in brackets below the name of each configuration.

Figure 11 depicts the box plot with the distribution of the accuracy scores on e-SNLI for the 60 train-validation splits for different SPARSEFIT configurations and the two pre-trained LM sizes. Overall, for e-SNLI, the task performance increases with the size of the model for most of the sparse fine-tuning configurations. Moreover, the interquartile range is considerably smaller when the model size increases (i.e., `T5-large` scores are less spread than the ones for `T5-base`). The highest median score was achieved by the fine-tuning of the *Layer Normalization* in `T5-large`, followed very closely by the fine-tuning of the *LM head* and the *Decoder* in `T5-large`. The combination of components (i.e. Layer Norm + Self-attention Query) performed very closely to the best-performing settings.

For the ECQA dataset, Figure 12 shows the box plot with the accuracy scores for different SPARSEFIT setups. It can be observed that the performance of the larger LM (i.e., `T5-large`) is consistently better than `T5-base`. Overall, the accuracy is fairly similar for all the SPARSEFIT configurations for a given LM size, with an average of 58% and 42% for `T5-large` and `T5-base`, respectively. Note that the random guess accuracy is 20% for the ECQA dataset, since there are 5 possible answer choices. The highest accuracy was achieved by the fine-tuning of the *Decoder* in `T5-large`, followed very closely by the fine-tuning of the *Layer Normalization* and *LM Head*. The combination of components achieves a slightly lower performance than single components for the task prediction. Surprisingly, for ECQA, the variability for a given combination of configuration-model (i.e. each box) is higher for `T5-large` than for `T5-base`. Moreover, the fine-tuning of the *Encoder* for `T5-base` gives worse results in comparison with all the other configurations. Besides the setting where only the *Encoder* is fine-tuned for `T5-base`, the highest observed range in ECQA is roughly 14%.

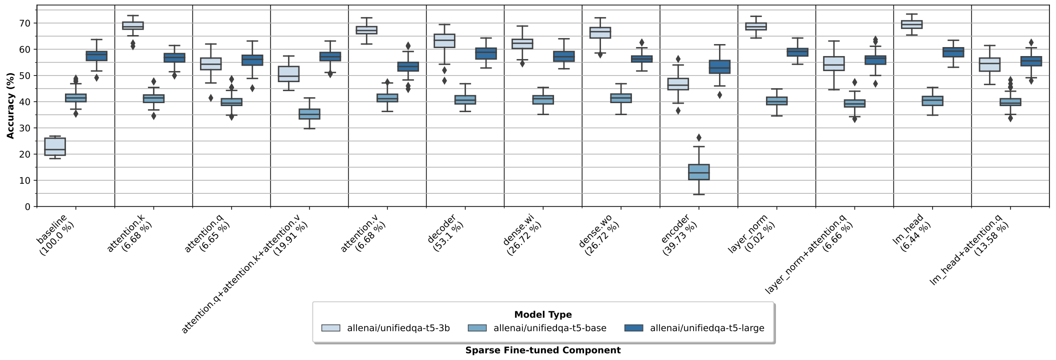

Figure 12: Distribution of the accuracy scores for different SPARSEFIT settings for the **ECQA** dataset. For each model, the variation represents the overall performance in each of the 60 train-validation splits. The percentage of parameters fine-tuned for each setup is depicted in brackets below the name of each configuration.todoUpdate plot with t5-3b results

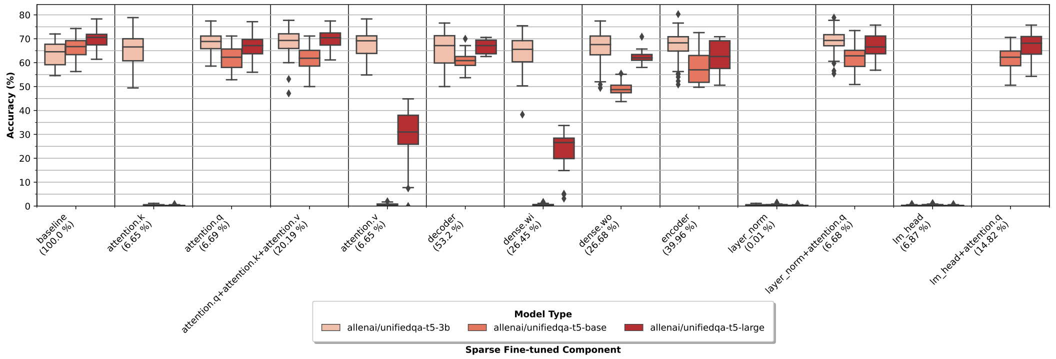

Figure 13: Distribution of the accuracy scores for different settings of SPARSEFIT for the **SBIC** dataset. For each model, the variation represents the overall performance in each of the 60 train-validation splits. The percentage of parameters fine-tuned for each setup is depicted in brackets below the name of each configuration.

For the SBIC dataset, Figure 13 depicts the box-plot with the dispersion of accuracy scores for `T5-base` and `T5-large`. Recall that for the SBIC dataset, we fine-tune the `UnifiedQA` variant of T5. In general, it can be seen that the accuracy score surges when the model size is increased; thus, the best accuracy scores for a given sparse fine-tuning setup are found for the `T5-large`. The best median accuracy performance is achieved by the baseline. However, the difference in the median scores between the best and the second and third best-ranked configurations (i.e. *Self-attention Layer* and *Layer Normalization* + *Self-attention Query*, respectively) are less than 3%. The maximum variance among scores for the 3 best-performing SPARSEFIT configurations is roughly 15%. Furthermore, it can be observed that for many very sparse fine-tuning configurations, the accuracy score is close to or equal to zero. Even though the performance of a random model is 50%, an accuracy of 0% is feasible in our scenario as the model could generate different words from the ones expected as labels. In this regard, the accuracy scores of zero are a consequence of the fact that, after the conditional generation, the model generates neither *“offensive"* nor *“non-offensive"* for any sample in the validation set. Notice that this phenomenon is particularly happening when only a small fraction of weights is fine-tuned.

For the ComVE dataset, we show in Figure 14 the accuracy for the 60 different train-validation splits for different SPARSEFIT settings and model sizes. It can be seen that the best-performing setting in terms of accuracy is the baseline for `UNIFIEDQA-T5-large`. (i.e. *Self-attention Layer* and *Layer Normalization* + *Self-attention Query* fine-tuning are the second and third best performing, respectively. Overall, the fine-tuning of the *Normalization Layer* leads to the worst task performance. Moreover, it can be observed that the performance increases with the size of

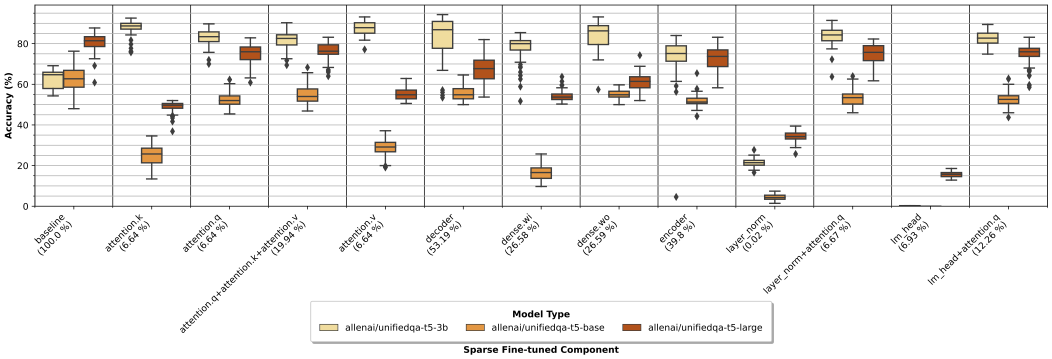

Figure 14: Distribution of the accuracy scores for different settings of SPARSEFIT for the **ComVE** dataset. For each model, the variation represents the overall performance in each of the 60 train-validation splits. The percentage of parameters fine-tuned for each setup is depicted in brackets below the name of each configuration.

the model, thus `UNIFIEDQA-T5-large` always performs better than `UNIFIEDQA-T5-base` for all the fine-tuning configurations. The smallest gap in performance between model sizes (`UNIFIEDQA-T5-large` vs. `UNIFIEDQA-T5-base`) happens for the fine-tuning of the *Dense Layer*. Conversely, the maximum spread in performance (i.e. the difference between the best and the worst split) is around 21% for models trained using the `UNIFIEDQA-T5-large` architecture.

C.2 Explanation Generation Performance

?? shows the box-plot with the normalized BERTscores for different SPARSEFIT setups fine-tuned on top of `T5-large`. In addition to explained in the main text, it can be seen that combinations of components lead to less variance in the score achieved for the 60 train-test splits (see the interquartile range). Furthermore, Table 7 shows the performance summary for the downstream performance and the NLEs quality for `T5-3b`. It can be observed that the *Attention Value Layer* achieves the best performance on average. We highlight that SPARSEFIT outperforms the baseline (i.e. full fine-tuning) for all datasets.

For e-SNLI, Figure 16 shows the normalized BERTscore over the 60 few-shot learning splits for different SPARSEFIT configurations. Overall, for every sparse fine-tuning setting, the BERTscore is consistently higher for the largest PLM (i.e. `T5-large`). However, the gap in performance is smaller for the best-performing sparse fine-tuning configurations. For instance, the difference in the average normalized BERTscore values between `T5-large` and `T5-base` for the best performing SPARSEFIT (i.e., *Decoder*) is roughly 5% while for the worst performing configuration is around 68%. The first five best-performing SPARSEFIT configurations for `T5-large` are *Decoder*, *Baseline*, *Self-attention KVQ*, *Layer Normalization* + *Self-attention Q*, and *Self-attention Values*. Note that the normalized BERTscore is zero for some sparse fine-tuning configurations (e.g., *Layer Normalization*). This is mostly happening when the sparse fine-tuning is applied to small models (i.e., `T5-base`). The fact that the BERTscore is zero for a given configuration for all the samples in a split implies that the generated NLEs are always empty. We explore the reasons behind this phenomenon in Section 4.2

For the ECQA dataset, we show in Figure 17 the spread of the normalized BERTscore for all SPARSEFIT configurations. Without exception, the largest model (`T5-large`) outperforms the `T5-base` models for every setting. Remarkably, for ECQA, many sparse fine-tuning configurations lead to the generation of empty explanations. Particularly, only the fine-tuning of the *Baseline*, the *Decoder*, and the *Encoder* are able to consistently generate non-empty explanations no matter the size of the model. Among the configurations that generate non-empty explanations, the best normalized BERTscores are achieved by the *Decoder* sparse fine-tuning, followed by the *Baseline* and *Encoder Blocks* fine-tuning. Note that for all of these configurations, the interquartile range is smaller than 6% regardless of the model size. Moreover, the fine-tuning of *Self-attention Query* achieves competitive results for `T5-large` but zero BERTscore for `T5-base`.

SparseFit		ComVE	ECQA	SBIC	e-SNLI	**Avg**
Baseline	Acc.	62.48 ±6.03	22.39 ±3.61	63.55 ±6.59	55.3 ±4.98	50.93 ±5.3
(100.00%)	nBERTs	55.55 ±5.6	19.73 ±3.22	61.21 ±6.79	49.25 ±4.36	46.44 ±4.99
Decoder	Acc.	83.67 ±10.12 ▽	62.62 ±4.16 ▽	65.59 ±7.51	87.48 ±4.02 ▽	74.84 ±6.45
(54.60%)	nBERTs	74.66 ±9.02 ▽	55.31 ±3.65 ▽	62.72 ±7.66	77.92 ±3.7 ▽	67.65 ±6.01
Encoder	Acc.	73.14 ±11.24 ▽	46.23 ±3.96 ▽	66.81 ±6.43	70.34 ±5.79 ▽	64.13 ±6.86
(40.95%)	nBERTs	66.7 ±10.28 ▽	41.46 ±3.56 ▽	64.45 ±6.7	63.79 ±5.28 ▽	59.1 ±6.46
Dense.wo	Acc.	83.91 ±6.54 ▽	66.21 ±3.12 ▽	66.64 ±6.46	86.85 ±3.0 ▽	75.9 ±4.78
(27.29%)	nBERTs	76.1 ±6.04 ▽	59.12 ±2.76 ▽	63.87 ±6.51	78.24 ±2.78 ▽	69.33 ±4.52
Dense.wi	Acc.	77.6 ±6.63 ▽	62.12 ±2.75 ▽	63.99 ±7.4	87.31 ±3.6 ▽	72.76 ±5.1
(27.29%)	nBERTs	70.21 ±6.04 ▽	55.12 ±2.44 ▽	61.05 ±7.43	78.24 ±3.28 ▽	66.16 ±4.8
Attention KQV	Acc.	81.73 ±4.14 ▽	50.24 ±3.48 ▽	68.84 ±5.37 ▽	75.3 ±3.78 ▽	69.03 ±4.19
(20.47%)	nBERTs	74.27 ±3.84 ▽	44.79 ±3.06 ▽	**66.12** ±5.46 ▽	67.67 ±3.36 ▽	63.21 ±3.93
LM head + Attention.Q	Acc.	82.59 ±3.37 ▽	54.28 ±3.57 ▽	0.0 ±0.0	79.33 ±2.94 ▽	72.07 ±3.29
(11.28%)	nBERTs	75.2 ±3.0 ▽	48.42 ±3.17 ▽	0.0 ±0.0	71.52 ±2.74 ▽	65.05 ±2.97
LM head	Acc.	0.09 ±0.13 ▽	**69.43** ±1.88 ▽	0.23 ±0.22 ▽	**89.04** ±1.63 ▽	39.7 ±0.96
(4.46%)	nBERTs	0.0 ±0.0 ▽	0.0 ±0.0 ▽	0.19 ±0.18 ▽	0.0 ±0.0 ▽	0.05 ±0.04
LayerNorm + Attention.Q	Acc.	83.27 ±4.52 ▽	54.13 ±3.86 ▽	**68.87** ±4.86 ▽	79.16 ±3.72 ▽	71.36 ±4.24
(6.84%)	nBERTs	75.83 ±4.14 ▽	48.31 ±3.46 ▽	65.86 ±5.07 ▽	71.27 ±3.44 ▽	65.32 ±4.03
Attention.Q	Acc.	83.09 ±4.15 ▽	54.39 ±3.66 ▽	68.44 ±4.44 ▽	77.88 ±3.66 ▽	70.95 ±3.98
(6.82%)	nBERTs	75.65 ±3.76 ▽	48.56 ±3.24 ▽	65.4 ±4.68 ▽	70.23 ±3.41 ▽	64.96 ±3.77
Attention.K	Acc.	87.7 ±3.83 ▽	68.74 ±2.29 ▽	65.48 ±6.26	87.8 ±1.83 ▽	77.43 ±3.55
(6.82%)	nBERTs	**80.01** ±3.52 ▽	**61.25** ±2.07 ▽	62.41 ±6.5	79.55 ±1.62 ▽	70.8 ±3.43
Attention.V	Acc.	**87.72** ±3.16 ▽	67.22 ±2.14 ▽	68.11 ±5.19 ▽	88.17 ±2.38 ▽	**77.81** ±3.22
(6.82%)	nBERTs	79.87 ±2.92 ▽	60.12 ±1.9 ▽	65.67 ±5.09 ▽	**79.63** ±2.26 ▽	**71.32** ±3.04
LayerNorm	Acc.	21.37 ±2.06 ▽	68.71 ±1.89 ▽	0.29 ±0.27 ▽	88.91 ±1.74 ▽	44.82 ±1.49
(0.02%)	nBERTs	0.0 ±0.0 ▽	0.0 ±0.0 ▽	0.24 ±0.22 ▽	0.0 ±0.0 ▽	0.06 ±0.06

Table 7: Summary of best performing SparseFit configurations for `T5-3B`. We report the average and the standard deviation over the 60 few-shot train-validation splits for the **accuracy** metric and the normalized BERTScore (**nBERTs**). In brackets are the percentages of fine-tuned weights for each SparseFit configuration. We show in **bold** the setting with the highest metric for each dataset. Significance testing was assessed via mean t-test in comparison with the baseline: ▽ represents a p-value lower than 10^{-2}.

Figure 18 shows the normalized BERTscore results for the SBIC dataset. Recall that for the SBIC dataset, we fine-tune the `UnifiedQA` variant of T5. As expected, the model size contributes to better performance. Consequently, the BERTscore is higher for the `T5-large` model for every sparse fine-tuning configuration. The best BERTscore median is achieved by the *Baseline* in combination with the `UNIFIEDQA-T5-large`, with a metric value of $\approx 68\%$. The second and third best-performing setups are the *Decoder* and the *Encoder*, respectively. Moreover, the fine-tuning of layers such as the *Normalization Layer* or *Self-attention Layer* results in the generation of text that does not contain the explanation token "because", thus the BERTscore is close to zero for those configurations.

We depict in Figure 19 the variation of the normalized BERTscore metric over the 60 different train-validation splits for the SparseFit configurations. Recall that for ComVE dataset, we fine-tune the `UnifiedQA` variant of T5. Overall, the BERTscore is substantially higher for `T5-large`. The best BERTscore for `T5-large` is obtained by the *Baseline* fine-tuning, with a median score of 75% for the 60 different seeds. Similar behavior can be seen for `T5-base`, where *Baseline* is also the setting with the best explanations (from the perspective of the automatic metric). The second and third best sparse fine-tuning setups are the *Self-attention Query* and *Baseline*, respectively. Notice that the difference in the median between the *Baseline* and the *Encoder* is around 3%. Moreover, the variance among the different splits for a given sparse fine-tuning setting is on average higher than for the *Baseline*. Remarkably, the sparse fine-tuning over the *Normalization Layer* was the only setting that obtained a zero BERTscore for the ComVE dataset.

C.3 Other PEFT Baselines

In order to make our approach comparable in the number of parameters, we test LoRa [19] using higher ranks. Table 8 shows the performance of LoRA for different rank sizes. Notice that average performance, in terms of accuracy and NLE quality, do not increase when the rank is increased.

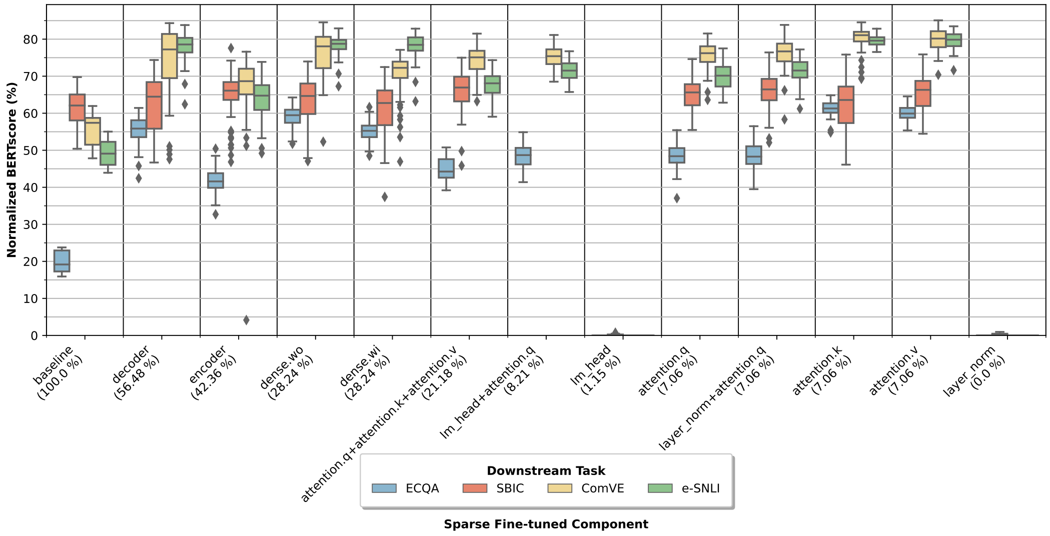

Figure 15: Distribution of the **normalized BERTScore** for different SPARSEFIT settings of sparse fine-tuning for `T5-3b`. The percentage of fine-tuned parameters is shown between brackets.

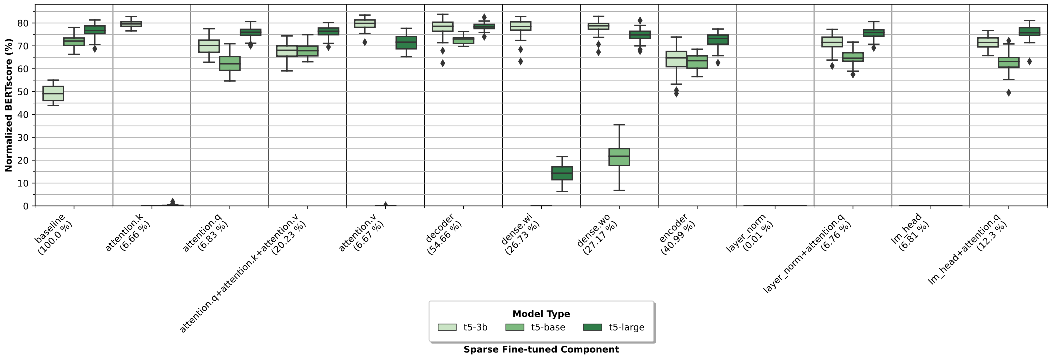

Figure 16: Distribution of the **normalized BERTscore** for different settings of sparse fine-tuning for the **e-SNLI** dataset. For each model, the box represents the overall performance over the 60 train-validation splits. The percentage of parameters fine-tuned for each setup is depicted in brackets below the name of each configuration.

C.4 Explanations Shortcomings per Dataset

Given the diverse nature of the studied datasets, we perform an individual analysis for each dataset in order to find the particular deficiencies and traits of the explanations by dataset. Figure 21 shows a set of histograms with the assessment of the annotators on shortcomings for the e-SNLI dataset. It can be seen that the *Nonsensical* category is consistently the most common no matter what fine-tuning strategy was used. Below, the reader can find two examples of *Nonsensical* explanations generated by the *Baseline* and the *Decoder* strategy, respectively.

In addition to this, *Input Repetition* is the second most common shortcoming for e-SNLI. A regular pattern found in the generated explanations is the repetition of a sub-string of the hypothesis as the predicted explanation, which happens for around 17% of the generated explanations. Below, the reader can see an example of input repetition found in the e-SNLI dataset.

We depict in Figure 24 a set of histograms with the number of times that a shortcoming category happens for different fine-tuning strategies for ECQA. Predominantly, *Incomplete Explanation* is the main weakness of generated NLEs. Notice that for this dataset, the answers are not generally shared by different samples (i.e., the possible labels for a sample are not always the same as in the other

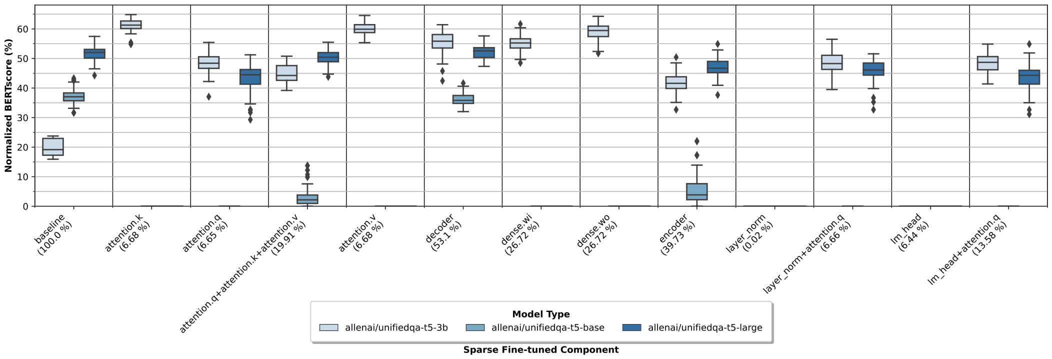

Figure 17: Distribution of the **normalized BERTscore** for different settings of sparse fine-tuning for the **ECQA** dataset. For each model, the box represents the overall performance over the 60 train-validation splits. The percentage of parameters fine-tuned for each setup is depicted in brackets below the name of each configuration.

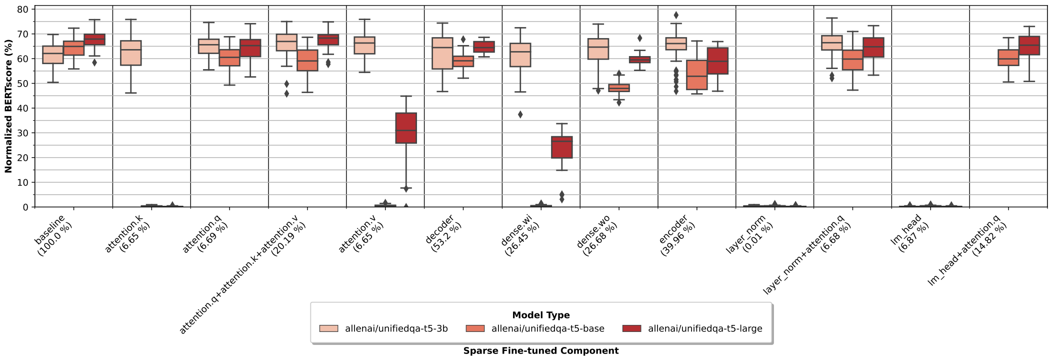

Figure 18: Distribution of the **normalized BERTscore** for different settings of sparse fine-tuning for the **SBIC** dataset. For each model, the box represents the overall performance over the 60 train-validation splits. The percentage of parameters fine-tuned for each setup is depicted in brackets below the name of each configuration.

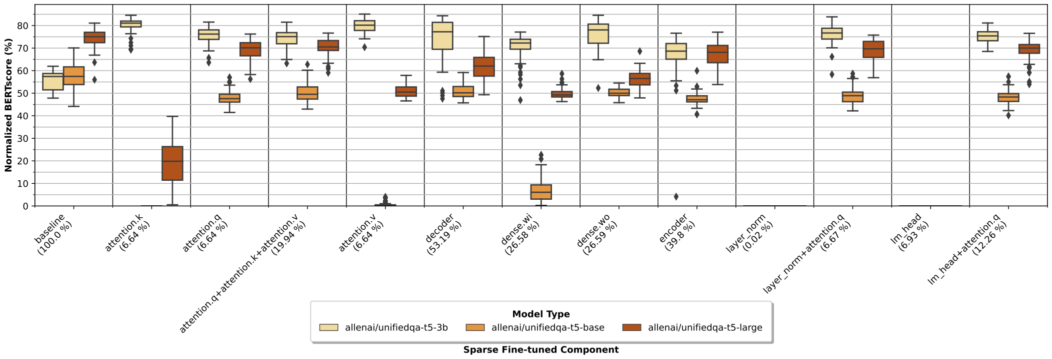

Figure 19: Distribution of the **normalized BERTscore** for different settings of sparse fine-tuning for the **ComVE** dataset. The baseline model represents the work done by [32], where all the parameters of the LM were fine-tuned. For each model, the box represents the overall performance over the 60 train-validation splits. The percentage of parameters fine-tuned for each setup is depicted in brackets below the name of each configuration.

datasets). This causes the generated explanations to be vague and incomplete. Below, the reader

PEFT Strategy	Rank Size	Percentage Parameters		**ComVE**	**ECQA**	**SBIC**	**e-SNLI**	**Avg**
LoRA	8	0.32%	Acc.	67.64 ±3.37	39.59 ±3.82	63.42 ±3.46	84.15 ±2.0	63.7 ±3.16
			nBERTs	61.24 ±3.09	1.55 ±1.26	60.93 ±3.45	76.41 ±1.82	50.03 ±2.4
	16	0.63%	Acc.	67.94 ±3.4	39.41 ±3.58	63.26 ±3.36	84.26 ±1.88	63.72 ±3.06
			nBERTs	61.51 ±3.11	1.44 ±1.31	60.78 ±3.49	76.5 ±1.71	50.06 ±2.41
	32	1.26%	Acc.	67.79 ±3.75	39.74 ±3.85	63.5 ±3.28	84.27 ±1.9	63.82 ±3.2
			nBERTs	61.36 ±3.43	1.36 ±1.18	61.01 ±3.36	76.51 ±1.73	50.06 ±2.43
	64	2.49%	Acc.	67.65 ±3.77	43.44 ±3.54	63.78 ±3.15	84.25 ±1.91	64.86 ±3.11
			nBERTs	61.31 ±3.42	0.32 ±0.40	61.10 ±3.31	76.54 ±1.73	50.01 ±2.10
	128	4.86%	Acc.	67.77 ±3.73	43.51 ±3.57	63.57 ±3.16	84.26 ±1.92	64.78 ±3.1
			nBERTs	61.36 ±3.41	0.33 ±0.41	61.06 ±3.29	76.49 ±1.75	49.81 ±2.22

Table 8: Accuracy and NLE quality metrics for different rank sizes in LoRA. We report the average and the standard deviation over the 60 few-shot train-validation splits for the **accuracy** metric and the normalized BERTScore (**nBERTs**).

Premise	A poor family is leaving their home with only a few belongings
Hypothesis	A man eats a chalupa
Label	Contradiction
Explanation	A family consists of two or more people, not just one man. **Generated Explanation** “a man who eats a chalupa also has to be poor"
Human Score	No
Reason	Nonsensical

Premise	A man in red pants skiing down a slope
Hypothesis	An Olympic skier skiing.
Label	Entailment
Explanation	WE have no idea if the man is an olympic skier or not. **Generated Explanation** “we don’t know what he is doing"
Human Score	No
Reason	Nonsensical

Figure 20: Examples of **Non-sensical** NLEs generated for e-SNLI.

can see 3 examples of *Incomplete Explanation* generated by the *Baseline*, *Decoder*, and *Encoder* fine-tuning strategy, respectively.

Figure 26 shows a set of histograms with the assessment done by the annotators about the most common shortcomings. Different from other datasets, there is no singular shortcoming that dominates the results for all the fine-tuning setups. The most common mistakes among all the explanations in the dataset are: *Inaccurate*, *Nonsensical*, and *Incomplete Explanation*. Below, the reader can find an example for the *Incomplete Explanation* shortcoming for the *Decoder* fine-tuning.

We have depicted in Figure 27 a series of histograms with the frequency of possible shortcomings given by human annotators to the evaluated explanations. It can be seen that annotators consider that the *Lack of explanation*, *Nonsensical*, and *Incomplete Explanation* are the most relevant categories to explain the weaknesses of the generated explanations.

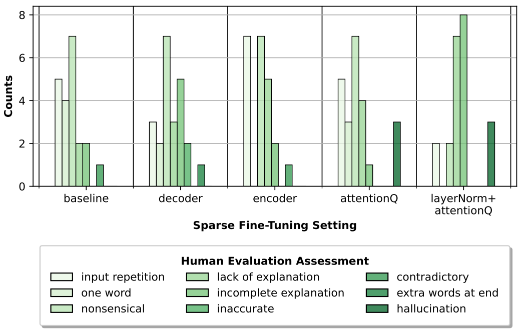

Figure 21: Histogram of the occurrences of the main shortcomings of the generated explanations for the baseline and the two best performing sparse fine-tuning setup for the **e-SNLI** dataset.

Premise	girl in uniform running through the water fountain gushing water from the tiles
Hypothesis	A girl is running through the water
Label	Entailment
Explanation	“If a water fountain is gushing water, that implies that there is water, and therefore the girl is running through water" **Generated Explanation** “a girl in uniform running through the water fountain."
Human Score	No
Reason	Input Repetition

Figure 22: Examples of **Input-Repetition** NLEs generated for e-SNLI.

C.5 Inter-annotator Agreement

We show in Figure 29 an example of perceptual disagreement where the annotators assigned the same plausibility reason but a different score. Furthermore, Figure 30 shows an example of expectation disagreement where human evaluators assigned a opposite score for the given explanation.

Question	Where is the ticket booth often outside the building?
Choices	train station — stadium — auditorium — subway
Label	auditorium
Explanation	Ticket booth of an auditorium is often outside the building **Generated Explanation** “auditorium is a large venue with lots of people. the ticket booth is often outside the building"
Human Score	Weak Yes
Reason	Incomplete Explanation

Question	What happens to the person that is well known for frequent farting?
Choices	social ostracism — evil — bad smell — stinking
Label	social ostracism
Explanation	Ticket booth of an auditorium is often outside the building **Generated Explanation** “Social ostracism is act of ignoring and excluding individuals. Frequent farting can make you ignored by people around you as frequent farting makes them uncomfortable with you"
Human Score	Weak Yes
Reason	Incomplete Explanation

Figure 23: Examples of **Incomplete Explanation** NLEs generated for ECQA.

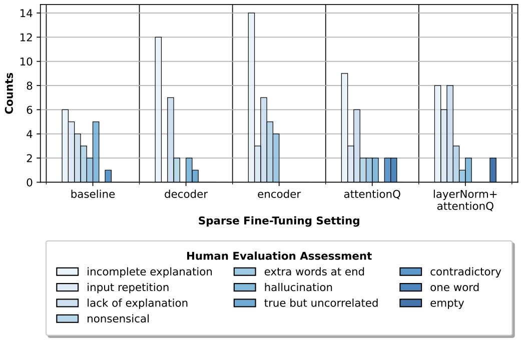

Figure 24: Histogram of the occurrences of the main shortcomings of the generated explanations for the baseline and the two best performing sparse fine-tuning setup for the **ECQA** dataset.

Post	The normie "vibe" argument is horseshit Anytime a normie says that you give off misogynistic vibes even if you don’t outright say "I hate women" they are lying.
Label	Offensive
Explanation	this post is a personal attack
	Generated Explanation “this post implies that women are inferior"
Human Score	Weak No
Reason	Incomplete Explanation

Figure 25: Examples of **Incomplete Explanation** NLEs generated for SBIC.

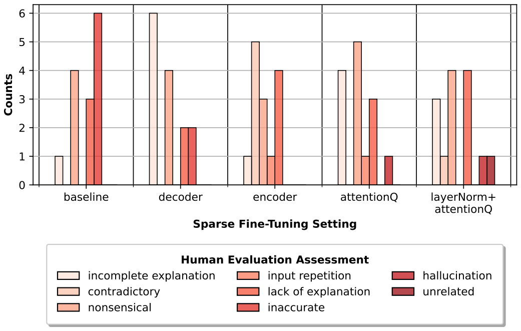

Figure 26: Histogram of the occurrences of the most common explanation shortcomings for the baseline and the two best performing sparse fine-tuning setup for the **SBIC** dataset.

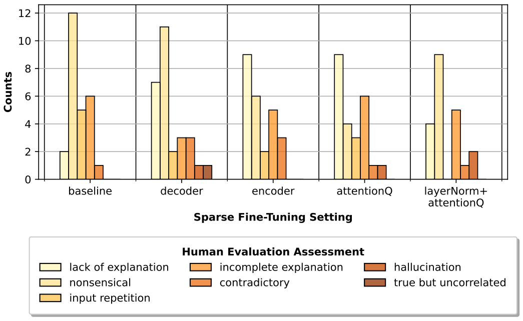

Figure 27: Histogram of the occurrences of the most common explanation shortcomings for the baseline and the two best performing sparse fine-tuning setup for the **ComVE** dataset.

Question	What is the best way to release energy?
Choices	yell — think — exercise — rest — work off
Label	exercise
Explanation	By doing exercise, one can release energy. The best way of releasing energy is exercise. **Generated Explanation** “exercise releases energy. to release energy, one must yell or yell loudly. to exercise, one must exercise vigorously. to work off, one must work off the energy."

	Annotator 1	**Annotator 2**
Human Score	No	Weak Yes
Reason	Hallucination	Hallucination

Figure 28: Example of annotator expectation disagreement in our study for the ECQA dataset.

Question	Teddy liked learning languages. He helped him with what?
Choices	problems — frustration — confidence — better communication — sadness
Label	better communication
Explanation	Better communication is defined as verbal speech or other methods of relaying information that get a point across. He helped him with better communication. **Generated Explanation** he helped him with better communication.

	Annotator 1	**Annotator 2**
Human Score	Weak No	Weak Yes
Reason	Lack of Explanation	Lack of Explanation

Figure 29: Example of annotator perceptual disagreement in our study for the ECQA dataset.

Question	What is the best way to release energy?
Choices	yell — think — exercise — rest — work off
Label	exercise
Explanation	By doing exercise, one can release energy. The best way of releasing energy is exercise. **Generated Explanation** “exercise releases energy. to release energy, one must yell or yell loudly. to exercise, one must exercise vigorously. to work off, one must work off the energy."

	Annotator 1	**Annotator 2**
Human Score	No	Weak Yes
Reason	Hallucination	Hallucination

Figure 30: Example of annotator expectation disagreement in our study for the ECQA dataset.