It’s Not Just Size That Matters

Small Language Models Are Also Few-Shot Learners

Paper by Timo Schick, Hinrich Schütze
Presented by Ayla Karakaş

Key Contributions

- One of the first papers to really focus on prompt engineering for transformer models

- Provides a method of training transformers to perform well in a few-shot setting with little data

- **Claim:** performance similar to GPT-3 can be obtained with LMs with several orders of magnitude fewer parameters

- **Goal:** environmentally sound NLP – reducing the amount of compute for few-shot learning

Prior Work:
Pattern-Exploiting Training (*PET*) & Iterative PET (*iPET*)

Exploiting Cloze Questions for Few Shot Text Classification and NLI
(Shick and Schütze, 2020)

Problems are difficult for most LMs to grasp from just a few examples:

- T_1: This was the best pizza I've ever had. ~~~~ **Label:** A
- T_2: You can get better sushi for half the price. ~~~ **Label:** B
- T_3: Pizza was average. Not worth the price. ~~~ **Label:** ???

Based on just the examples, how to infer the correct label for T_3?

Task descriptions help solving few-shot tasks.
It's much easier to assign label B if we specify the task is identifying whether the text is about prices.

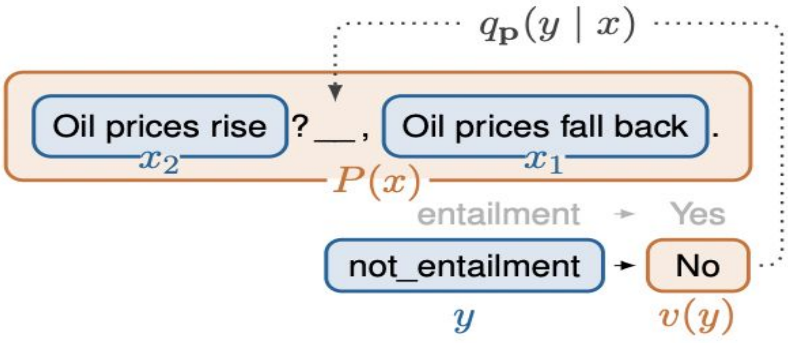
- Helps **distill the knowledge** of generative models into discriminative downstream tasks (e.g. sentiment analysis, natural language inference).

Patterns & Verbalizers

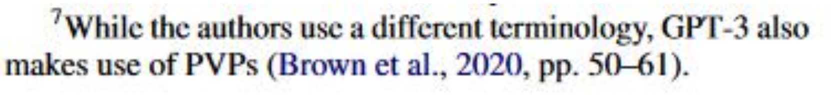
Pattern-verablizer pair (PVP) p = (P,v) consists of:

- a **pattern** function *P* which maps inputs to cloze questions containing a single mask
- a **verbalizer** function *v* that maps each output to a single token representing task-specific meaning in *P*

Aim: derive the probability of an output y being correct for input x from the probability of v(y) being “correct” at masked position in P(x).

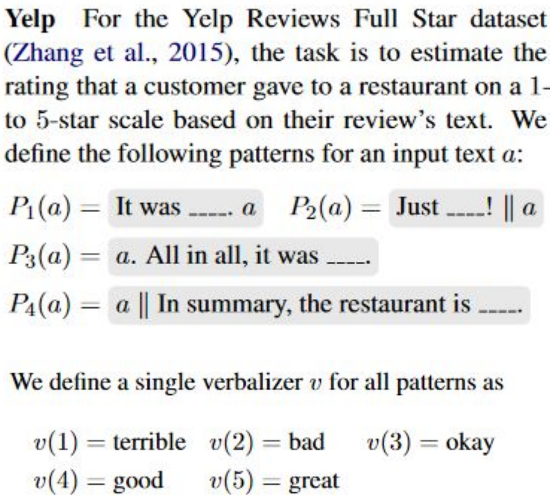


- Input x = (x_1,x_2) converted into a cloze question P(x).
- for each output y, q_p(y|x) comes from the probability of v(y) being a plausible choice for the masked position.



[7] While the authors use a different terminology, GPT-3 also makes use of PVPs (Brown et al., 2020, pp. 50–61).

Patterns & Verbalizers



Yelp For the Yelp Reviews Full Star dataset (Zhang et al., 2015), the task is to estimate the rating that a customer gave to a restaurant on a 1- to 5-star scale based on their review's text. We define the following patterns for an input text a:

$P_1(a) =$ It was ____. a $P_2(a) =$ Just ____! || a

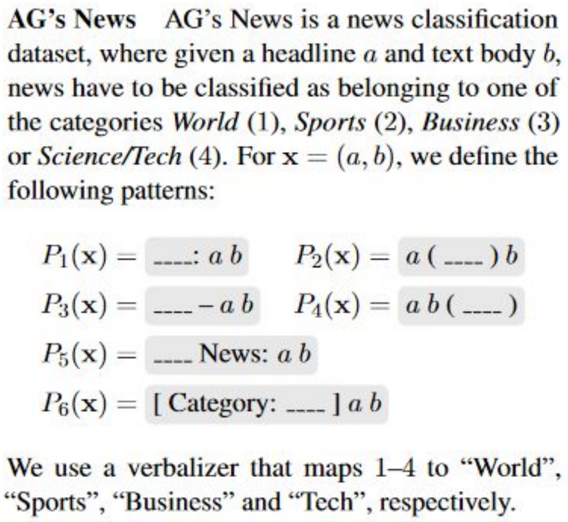
$P_3(a) =$ a. All in all, it was ____.

$P_4(a) =$ a || In summary, the restaurant is ____.

We define a single verbalizer v for all patterns as

$v(1) =$ terrible $v(2) =$ bad $v(3) =$ okay

$v(4) =$ good $v(5) =$ great



AG's News AG's News is a news classification dataset, where given a headline a and text body b, news have to be classified as belonging to one of the categories *World* (1), *Sports* (2), *Business* (3) or *Science/Tech* (4). For $\mathbf{x} = (a, b)$, we define the following patterns:

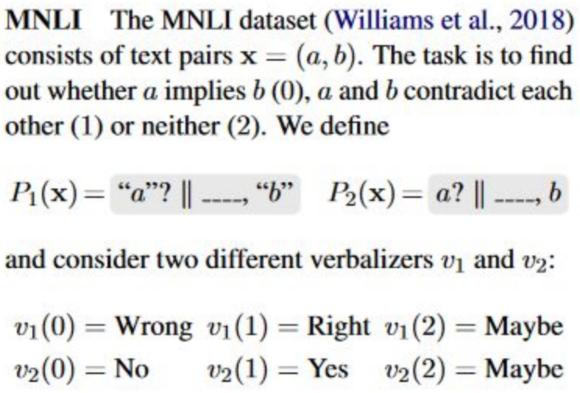
$P_1(\mathbf{x}) =$ ____: a b $P_2(\mathbf{x}) =$ a (____) b

$P_3(\mathbf{x}) =$ ____ – a b $P_4(\mathbf{x}) =$ a b (____)

$P_5(\mathbf{x}) =$ ____ News: a b

$P_6(\mathbf{x}) =$ [Category: ____] a b

We use a verbalizer that maps 1–4 to "World", "Sports", "Business" and "Tech", respectively.



MNLI The MNLI dataset (Williams et al., 2018) consists of text pairs $\mathbf{x} = (a, b)$. The task is to find out whether a implies b (0), a and b contradict each other (1) or neither (2). We define

$P_1(\mathbf{x}) =$ "a"? || ____, "b" $P_2(\mathbf{x}) =$ a? || ____, b

and consider two different verbalizers v_1 and v_2:

$v_1(0) =$ Wrong $v_1(1) =$ Right $v_1(2) =$ Maybe

$v_2(0) =$ No $v_2(1) =$ Yes $v_2(2) =$ Maybe

Pattern-Exploiting Training (PET)

- **Pattern-Exploiting Training (PET)**
  - semi-supervised training procedure

1. **reformulate input** examples as cloze-style phrases to help LMs understand the task
2. those phrases are used to **assign soft labels** to large set of unlabeled examples (distillation)
3. standard **supervised training** is used on resulting soft-labeled training set

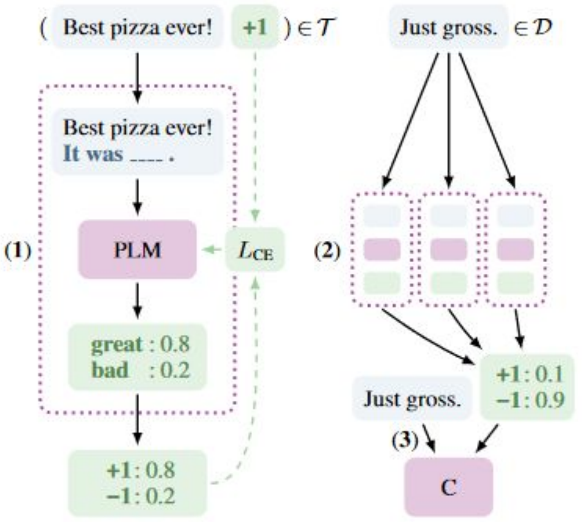


Figure 1: PET for sentiment classification. (1) A number of patterns encoding some form of task description are created to convert training examples to cloze questions; for each pattern, a pretrained language model is finetuned. (2) The ensemble of trained models annotates unlabeled data. (3) A classifier is trained on the resulting soft-labeled dataset.

Iterative PET (iPET)

Problem:

- Training set for the final model may contain many **mislabeled examples**
- The knowledge of **all individual models** is distilled into a single classifier
- Some patterns perform **much worse** than others.

Solution: Train several generations of models on datasets of increasing size.

Iterative PET (iPET)

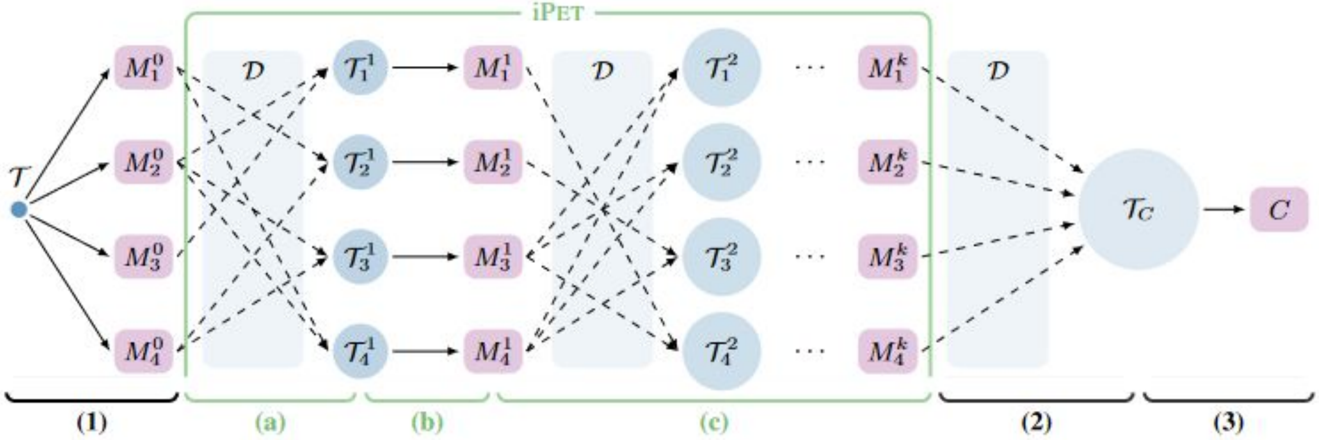
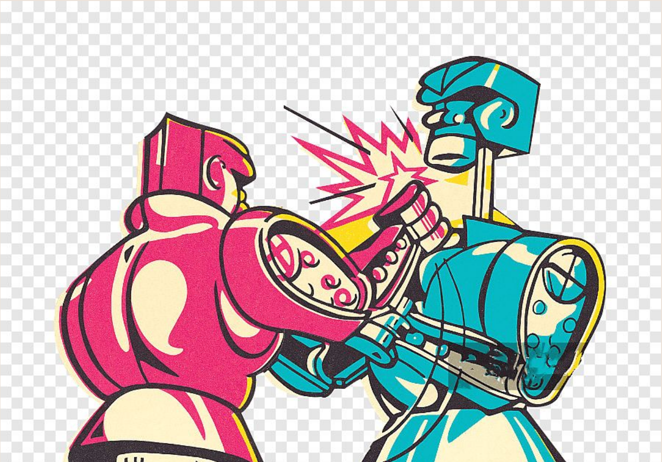


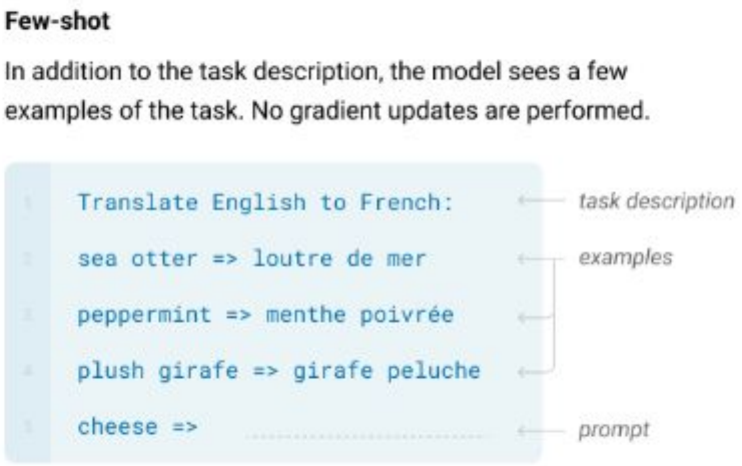
Figure 2: Schematic representation of PET (1-3) and iPET (a-c). (1) The initial training set is used to finetune an ensemble of PLMs. (a) For each model, a random subset of other models generates a new training set by labeling examples from $\mathcal{D}$. (b) A new set of PET models is trained using the larger, model-specific datasets. (c) The previous two steps are repeated k times, each time increasing the size of the generated training sets by a factor of d. (2) The final set of models is used to create a soft-labeled dataset $\mathcal{T}_C$. (3) A classifier C is trained on this dataset.

PET/iPET vs. GPT-3

(2021)



Recap: GPT-3 “Priming” (in-context learning)



Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

```
Translate English to French:
sea otter => loutre de mer
peppermint => menthe poivrée
plush girafe => girafe peluche
cheese => ..............................
```

task description
examples
prompt

- examples of the task are included in the description of the task (2048 tokens in GPT-3 context window, approx. 100 examples)

- no gradient updates

- **but** requires massive LM to work well

- most LMs can only support a context window of a few hundred tokens

Summary of the model

Underlying LM: ALBERT-xxlarge-v2

ALBERT = “A Lite BERT”

https://huggingface.co/albert-xxlarge-v2

- encoder only, like BERT
- 12 repeating layers
- 128 embedding dimension
- 4096 hidden dimension
- 64 attention heads
- 223M parameters

Plus final sequence classification head

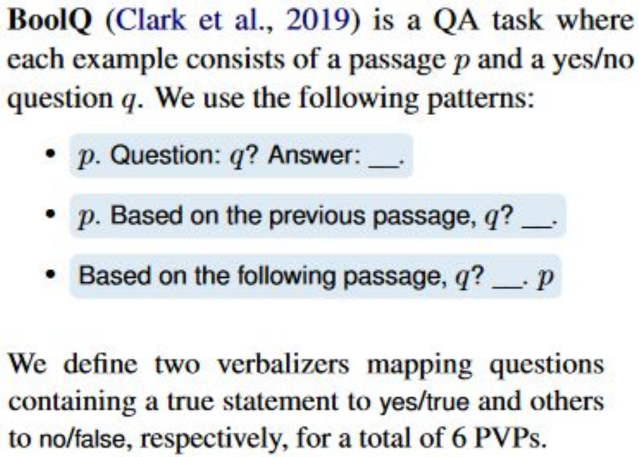
Pretraining objectives:

- Masked language modeling (MLM)
- Sentence Ordering Prediction (SOP)

Training data:

- BookCorpus (11,038 unpublished books, 800M words)
- English Wikipedia (2,500M words)
- raw texts (self-supervised)

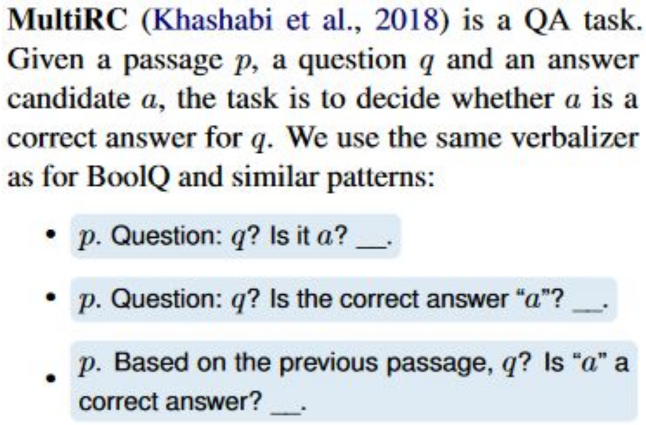
Tasks



BoolQ (Clark et al., 2019) is a QA task where each example consists of a passage p and a yes/no question q. We use the following patterns:

- p. Question: q? Answer: __.
- p. Based on the previous passage, q? __.
- Based on the following passage, q? __. p

We define two verbalizers mapping questions containing a true statement to yes/true and others to no/false, respectively, for a total of 6 PVPs.

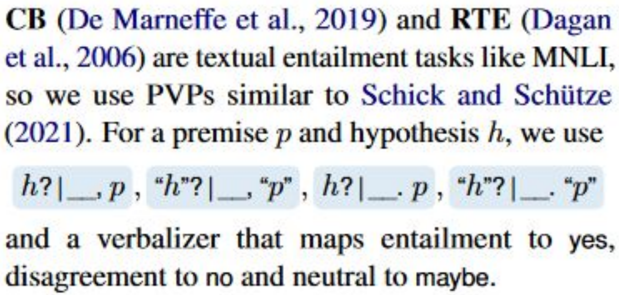


MultiRC (Khashabi et al., 2018) is a QA task. Given a passage p, a question q and an answer candidate a, the task is to decide whether a is a correct answer for q. We use the same verbalizer as for BoolQ and similar patterns:

- p. Question: q? Is it a? __.
- p. Question: q? Is the correct answer "a"? __.
- p. Based on the previous passage, q? Is "a" a correct answer? __.

← **QA**

Entailment →



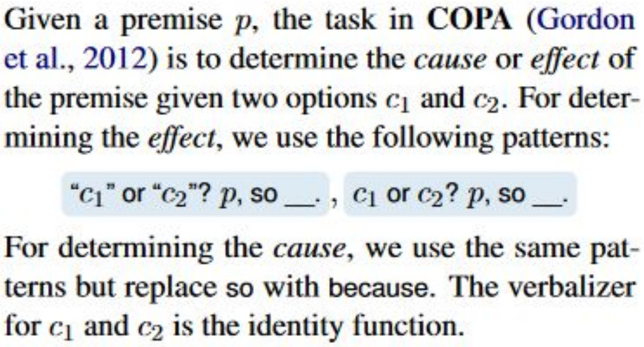
CB (De Marneffe et al., 2019) and **RTE** (Dagan et al., 2006) are textual entailment tasks like MNLI, so we use PVPs similar to Schick and Schütze (2021). For a premise p and hypothesis h, we use

h? | __, p , "h"? | __, "p" , h? | __. p , "h"? | __. "p"

and a verbalizer that maps entailment to yes, disagreement to no and neutral to maybe.

Tasks (cont’d)

Causal inference

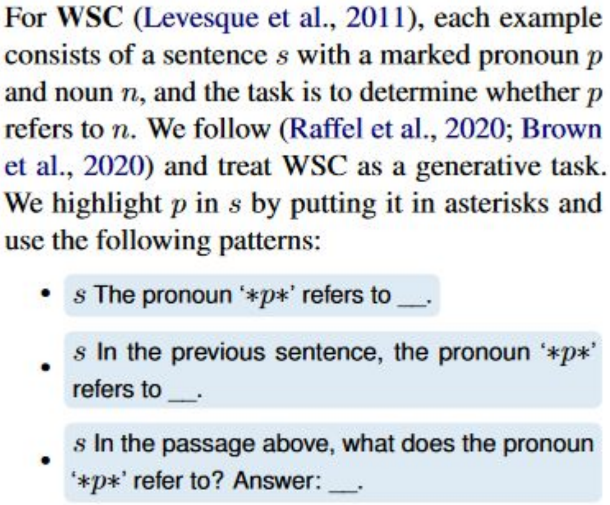


Given a premise p, the task in **COPA** (Gordon et al., 2012) is to determine the *cause* or *effect* of the premise given two options c_1 and c_2. For determining the *effect*, we use the following patterns:

“c_1” or “c_2”? p, so ___ , c_1 or c_2? p, so ___.

For determining the *cause*, we use the same patterns but replace so with because. The verbalizer for c_1 and c_2 is the identity function.

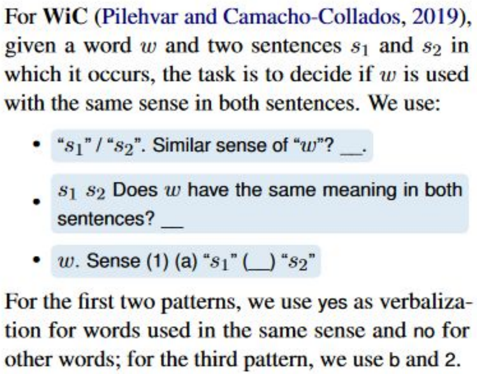
Pronoun resolution



For **WSC** (Levesque et al., 2011), each example consists of a sentence s with a marked pronoun p and noun n, and the task is to determine whether p refers to n. We follow (Raffel et al., 2020; Brown et al., 2020) and treat WSC as a generative task. We highlight p in s by putting it in asterisks and use the following patterns:

- s The pronoun ‘$*p*$’ refers to ___.
- s In the previous sentence, the pronoun ‘$*p*$’ refers to ___.
- s In the passage above, what does the pronoun ‘$*p*$’ refer to? Answer: ___.

Word sense



For **WiC** (Pilehvar and Camacho-Collados, 2019), given a word w and two sentences s_1 and s_2 in which it occurs, the task is to decide if w is used with the same sense in both sentences. We use:

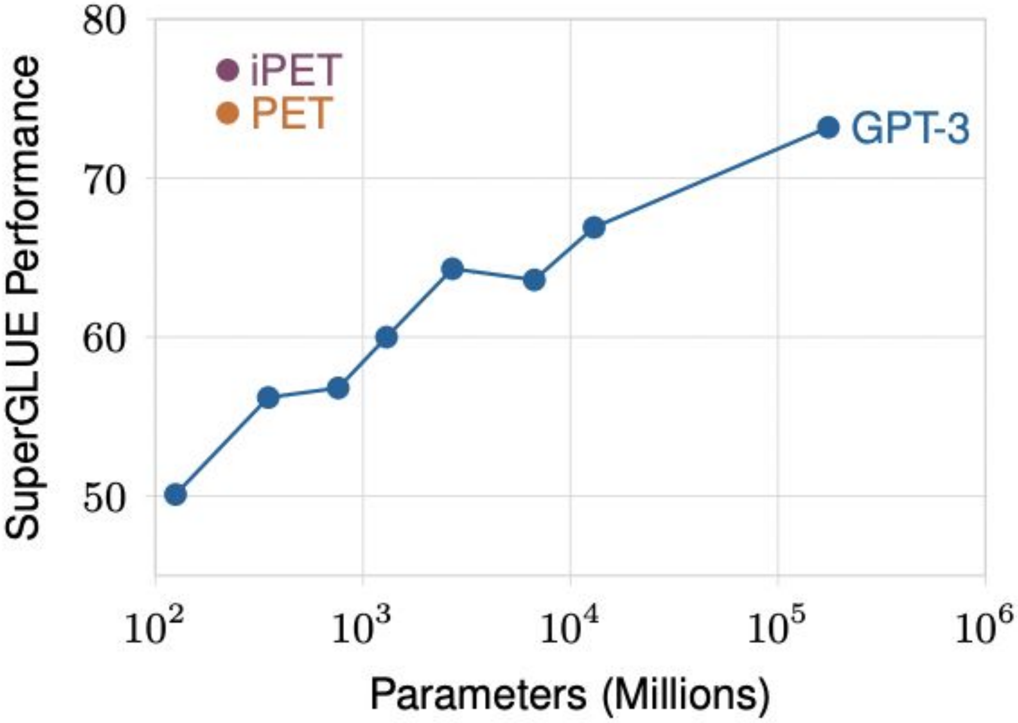
- “s_1” / “s_2”. Similar sense of “w”? ___.
- s_1 s_2 Does w have the same meaning in both sentences? ___
- w. Sense (1) (a) “s_1” (___) “s_2”

For the first two patterns, we use yes as verbalization for words used in the same sense and no for other words; for the third pattern, we use b and 2.

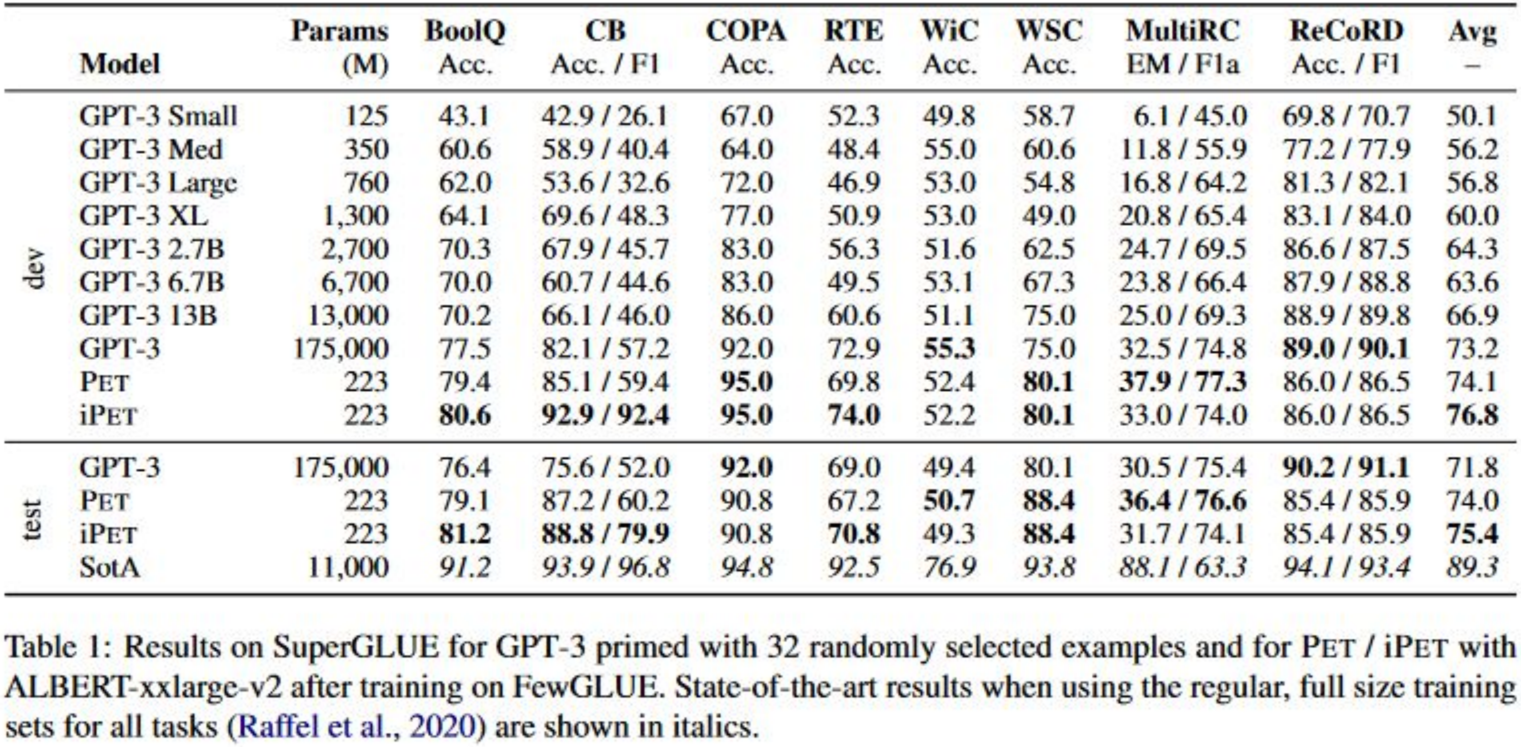
ALBERT + PET/iPET outperforms GPT-3 on SuperGLUE

https://super.gluebenchmark.com/leaderboard/

- 32 training examples
- 0.1% of parameters compared to GPT-3 (223M vs. 175B)
- Several hours on a single GPU



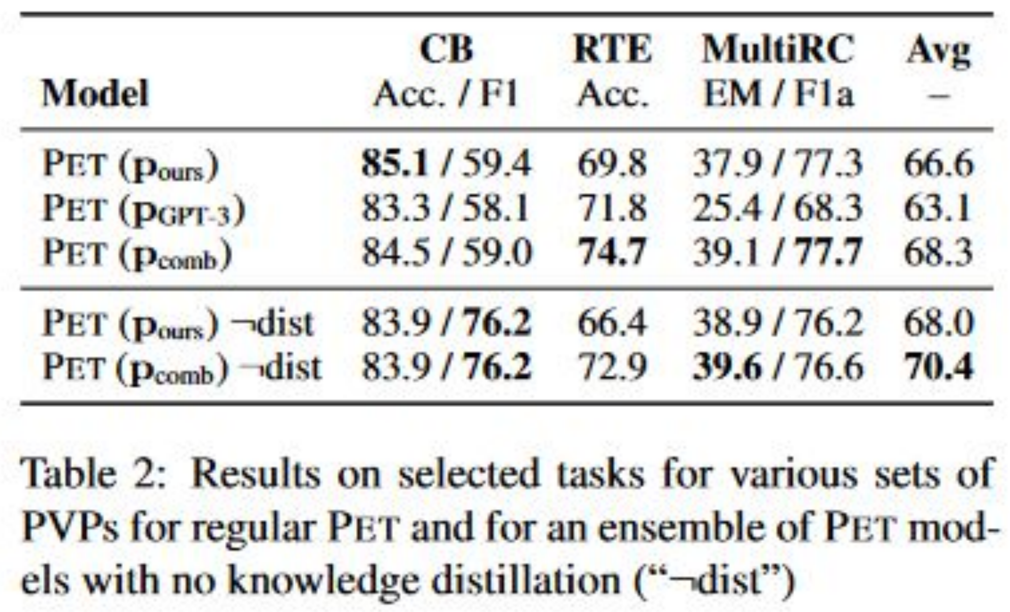
ALBERT+(i)PET



	Model	Params (M)	BoolQ Acc.	CB Acc. / F1	COPA Acc.	RTE Acc.	WiC Acc.	WSC Acc.	MultiRC EM / F1a	ReCoRD Acc. / F1	Avg –
dev	GPT-3 Small	125	43.1	42.9 / 26.1	67.0	52.3	49.8	58.7	6.1 / 45.0	69.8 / 70.7	50.1
	GPT-3 Med	350	60.6	58.9 / 40.4	64.0	48.4	55.0	60.6	11.8 / 55.9	77.2 / 77.9	56.2
	GPT-3 Large	760	62.0	53.6 / 32.6	72.0	46.9	53.0	54.8	16.8 / 64.2	81.3 / 82.1	56.8
	GPT-3 XL	1,300	64.1	69.6 / 48.3	77.0	50.9	53.0	49.0	20.8 / 65.4	83.1 / 84.0	60.0
	GPT-3 2.7B	2,700	70.3	67.9 / 45.7	83.0	56.3	51.6	62.5	24.7 / 69.5	86.6 / 87.5	64.3
	GPT-3 6.7B	6,700	70.0	60.7 / 44.6	83.0	49.5	53.1	67.3	23.8 / 66.4	87.9 / 88.8	63.6
	GPT-3 13B	13,000	70.2	66.1 / 46.0	86.0	60.6	51.1	75.0	25.0 / 69.3	88.9 / 89.8	66.9
	GPT-3	175,000	77.5	82.1 / 57.2	92.0	72.9	**55.3**	75.0	32.5 / 74.8	**89.0 / 90.1**	73.2
	PET	223	79.4	85.1 / 59.4	**95.0**	69.8	52.4	**80.1**	**37.9 / 77.3**	86.0 / 86.5	74.1
	iPET	223	**80.6**	**92.9 / 92.4**	**95.0**	**74.0**	52.2	**80.1**	33.0 / 74.0	86.0 / 86.5	**76.8**
test	GPT-3	175,000	76.4	75.6 / 52.0	**92.0**	69.0	49.4	80.1	30.5 / 75.4	**90.2 / 91.1**	71.8
	PET	223	79.1	87.2 / 60.2	90.8	67.2	**50.7**	**88.4**	**36.4 / 76.6**	85.4 / 85.9	74.0
	iPET	223	**81.2**	**88.8 / 79.9**	90.8	**70.8**	49.3	**88.4**	31.7 / 74.1	85.4 / 85.9	**75.4**
	SotA	11,000	*91.2*	*93.9 / 96.8*	*94.8*	*92.5*	*76.9*	*93.8*	*88.1 / 63.3*	*94.1 / 93.4*	*89.3*

Table 1: Results on SuperGLUE for GPT-3 primed with 32 randomly selected examples and for PET / iPET with ALBERT-xxlarge-v2 after training on FewGLUE. State-of-the-art results when using the regular, full size training sets for all tasks (Raffel et al., 2020) are shown in italics.

Results without distillation

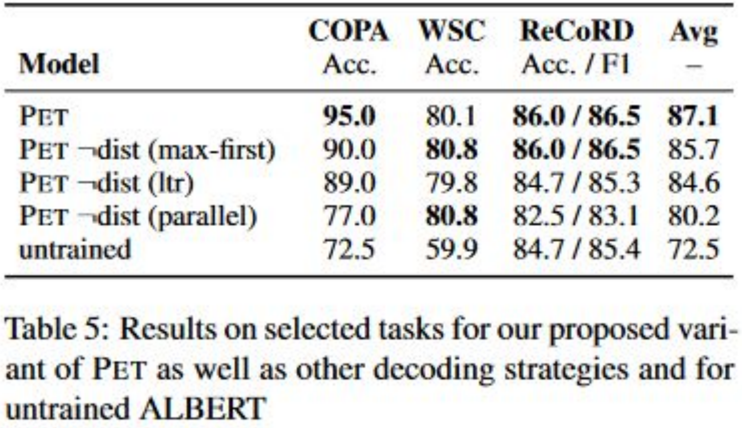


Model	CB Acc. / F1	RTE Acc.	MultiRC EM / F1a	Avg –
PET ($\mathbf{p}_{\text{ours}}$)	**85.1** / 59.4	69.8	37.9 / 77.3	66.6
PET ($\mathbf{p}_{\text{GPT-3}}$)	83.3 / 58.1	71.8	25.4 / 68.3	63.1
PET ($\mathbf{p}_{\text{comb}}$)	84.5 / 59.0	**74.7**	39.1 / **77.7**	68.3
PET ($\mathbf{p}_{\text{ours}}$) ¬dist	83.9 / **76.2**	66.4	38.9 / 76.2	68.0
PET ($\mathbf{p}_{\text{comb}}$) ¬dist	83.9 / **76.2**	72.9	**39.6** / 76.6	**70.4**

Table 2: Results on selected tasks for various sets of PVPs for regular PET and for an ensemble of PET models with no knowledge distillation ("¬dist")

PET with multiple masks

- **max-first:** decoding strategy of predicting tokens in order of probability
- **ltr:** left-to-right decoding
- **parallel:** decoding all tokens simultaneously
- **untrained:** untrained ALBERT



Model	COPA Acc.	WSC Acc.	ReCoRD Acc. / F1	Avg –
PET	**95.0**	80.1	**86.0 / 86.5**	**87.1**
PET ¬dist (max-first)	90.0	**80.8**	**86.0 / 86.5**	85.7
PET ¬dist (ltr)	89.0	79.8	84.7 / 85.3	84.6
PET ¬dist (parallel)	77.0	**80.8**	82.5 / 83.1	80.2
untrained	72.5	59.9	84.7 / 85.4	72.5

Table 5: Results on selected tasks for our proposed variant of PET as well as other decoding strategies and for untrained ALBERT

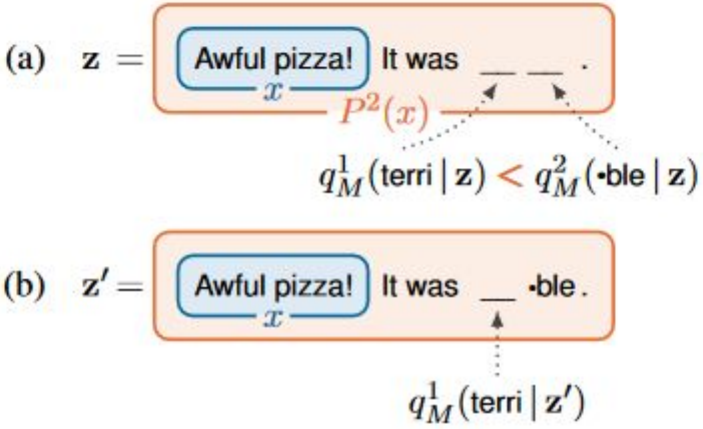
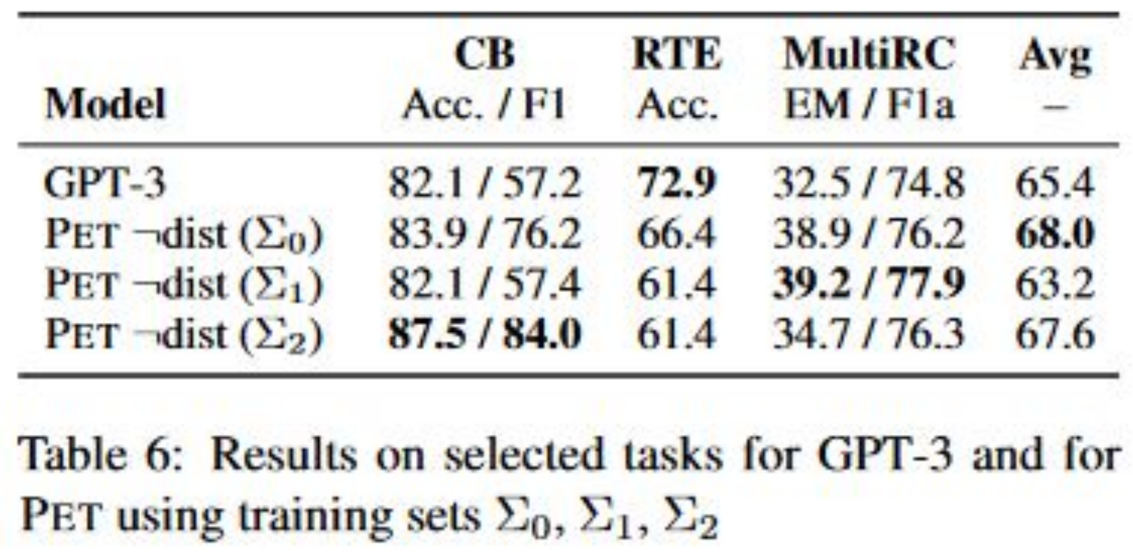

(a) $\mathbf{z} =$
Awful pizza! It was __ __ .
x
$P^2(x)$
$q_M^1(\text{terri} \mid \mathbf{z}) < q_M^2(\text{•ble} \mid \mathbf{z})$
(b) $\mathbf{z}' =$
Awful pizza! It was __ •ble .
x
$q_M^1(\text{terri} \mid \mathbf{z}')$

Figure 3: Inference for a verbalization consisting of the two tokens terri and •ble. (a) We first compute the probability of each token at its position in the cloze question $P^2(x)$ and identify the token with the highest probability. (b) We insert this token into the cloze question and compute the probability of the remaining token.

Variance of Labeled Examples



Model	CB Acc. / F1	RTE Acc.	MultiRC EM / F1a	Avg –
GPT-3	82.1 / 57.2	**72.9**	32.5 / 74.8	65.4
PET ¬dist (Σ_0)	83.9 / 76.2	66.4	38.9 / 76.2	**68.0**
PET ¬dist (Σ_1)	82.1 / 57.4	61.4	**39.2 / 77.9**	63.2
PET ¬dist (Σ_2)	**87.5 / 84.0**	61.4	34.7 / 76.3	67.6

Table 6: Results on selected tasks for GPT-3 and for PET using training sets Σ_0, Σ_1, Σ_2

Caveats

- GPT-3 few-shot learning is a demonstration of its capabilities at inference time.
    - GPT-3 was designed for language modeling, not few-shot learning, so it is a bit of an unfair comparison.

- Unlabeled data is easier to obtain than labeled data, but task-specific unlabeled data can still be hard to get
    - ADAPET https://aclanthology.org/2021.emnlp-main.407.pdf
        - in PET only label tokens (e.g. “yes”, “no”) get gradient updates
        - removes the distillation steps by introducing more fine-grained losses across the whole vocabulary, giving the model more chances to adapt to the task

Thank you :)

Questions

- Is this really few-shot learning if gradients are being updated?

- For iPET: Isn’t this method at risk of cascading failure if in some iteration a model generates mislabeled data?
    - The authors sort-of address this:
      “To avoid training future generations on mislabeled data, we prefer examples for which the ensemble of models is confident in its prediction. The underlying intuition is that even without calibration, examples for which labels are predicted with high confidence are typically more likely to be classified correctly (Guo et al., 2017).”
    - To what extent can we rely on this confidence?