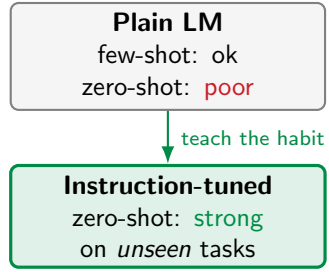
Great at examples, clumsy at instructions

A large LM like GPT-3 is a strong **few-shot** learner: show it a couple of examples and it plays along. But hand it a bare **zero-shot** instruction – "Does this premise entail this hypothesis?" – and it often flubs it.

Why? Next-token pretraining never rewarded "*do what this instruction says*." The model saw text, not tasks. (Same framing as InstructGPT, LLM-10.)

FLAN's question: can we make a model good at **brand-new tasks with zero examples**, just by teaching it the *habit* of following instructions?

Matched pair to watch: **LIMA (LLM-4)** answers "curate 1,000 examples"; FLAN answers "**scale up** the number of tasks."



Outline

Teach the habit, not the tasks

You are not teaching 60 tasks. You are teaching one skill: read an instruction, do what it asks.

Intuition: the new hire who learns to take a ticket

Picture a support agent on their first week. They handle password resets, refund requests, shipping questions – **60 kinds of ticket**. Then a **61st kind** they have never seen lands in the queue, and they handle it fine.

Why? They did not memorize 60 scripts. They learned the **format of the job**: *read the request, figure out what is being asked, produce the answer.*

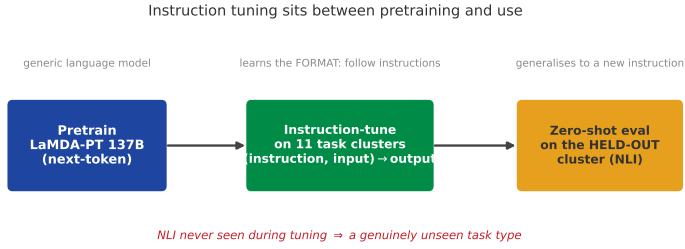
> FLAN's bet in one line: finetune on enough tasks phrased as instructions, and the model stops learning *those tasks* and starts learning **"follow an instruction"** – a skill that transfers to instructions it never trained on.

This is the opposite failure mode from overfitting: the win is *generalization to new task types*, not memorizing the training tasks.

Instruction tuning, defined

Instruction tuning sits between pretraining and use: a round of **multitask finetuning** where every example is reformatted as

$$(\text{instruction},\ \text{input}) \longrightarrow \text{output}.$$



It borrows from both worlds: the **supervision** of pretrain–finetune and the **instruction interface** of prompting. The payoff is measured on a **held-out cluster** the model was never tuned on.

One task, two framings

Take an NLI example. Instruction tuning is just **rewriting** it as a natural-language instruction – then training on many such rewrites.

Raw dataset row

```
premise: a dog runs in the park
hypothesis: an animal moves
label: entailment
```

The model just learns this *schema* – it never learns to *obey* a request.

Rewritten as an instruction

"A dog runs in the park." Using only this, is "an animal moves" **true**, **false**, or **undetermined**?
$\rightarrow$ **true**

$\sim$10 templates like this per dataset, across 62 datasets.

Same data, rephrased. Do this across *many* task types and the model learns the meta-skill "**read the instruction, do what it asks**" – which then transfers to instructions it never saw.

The method

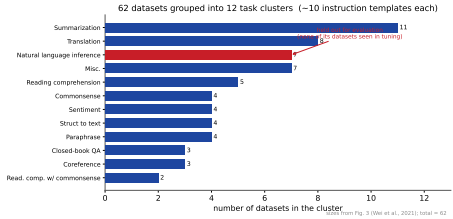
62 datasets, 12 clusters, hand-written templates – and one strict rule for “unseen.”

62 datasets, 12 clusters, ∼10 templates each

The data recipe:

- **62** public NLP datasets, grouped into **12 task clusters** (NLI, QA, summarization, translation, sentiment, . . .).
- Each dataset gets ∼**10 hand-written instruction templates**; up to **3** "turn the task around" (e.g. *generate* a review from its label).
- Base model: **137B LaMDA-PT** (dense, decoder-only, LM-only pretrained).

No new labels are collected. Existing datasets are re-*verbalized* as natural-language instructions.



The rigorous eval: hold out the whole cluster

How do you prove "zero-shot on an unseen task" when you finetuned on 60 tasks? FLAN uses a **conservative** definition of unseen.

- To score cluster c (say NLI), **drop every dataset in c's cluster** from finetuning, tune on the rest, then evaluate on c.
- So "unseen NLI" means *no* NLI dataset – not just a different NLI dataset – was ever seen. Each evaluated cluster uses its **own** checkpoint.

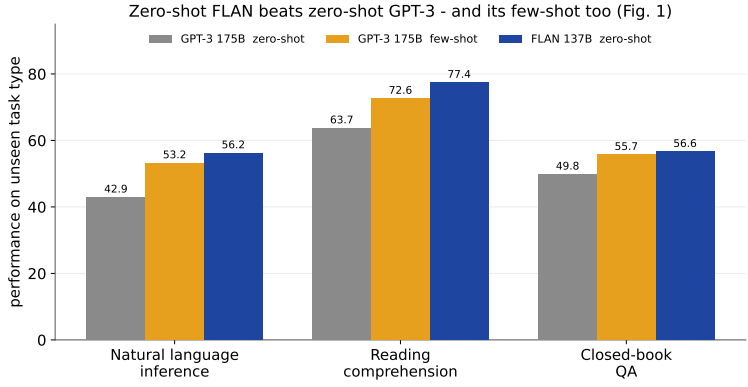
This held-out-**cluster** design (stricter than held-out-dataset) is what makes the zero-shot claim honest: the model truly never met that task type.

Cost of the honesty: you train a *separate* model per held-out cluster. The number in a results table is really an average over several tuned models.

Does it work?

Zero-shot FLAN vs a 175B model that is 1.3x larger.

FLAN 137B beats GPT-3 175B – zero-shot, and often few-shot



Zero-shot FLAN outperforms **zero-shot** GPT-3 on **20 of 25 datasets**, and beats **few-shot** GPT-3 on **10** – by a large margin on ANLI, RTE, BoolQ, AI2-ARC, OpenbookQA, and StoryCloze. All from a *smaller* base model.

The honest caveat: it does not help everything

Instruction tuning does **not** help tasks that are *already* phrased like the LM objective. Commonsense reasoning and coreference posed as **sentence completions** ("finish this sentence") are basically next-token prediction already.

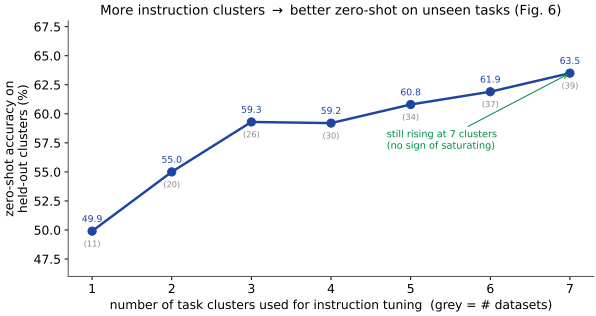
- On **7** such commonsense / coreference tasks, FLAN beat the untuned LaMDA-PT on only **3 of 7**.
- There the instruction is **redundant** – it adds nothing the model was not already doing, so there is no habit to gain.

This sharpens the thesis: the win comes from the **instructions**, not from "more data." Where the instruction carries real information (NLI, QA, translation), FLAN surges; where it is redundant, it does not.

What actually makes it work

Three knobs: number of clusters, model scale, and the instructions themselves.

Knob 1: more task clusters → better generalization



Hold out NLI, closed-book QA, and commonsense; add the other clusters to finetuning **one at a time**.

- Held-out zero-shot accuracy **keeps climbing** with each cluster added.
- At 7 clusters it **still has not saturated** – more diversity would likely help further.

The driver is instruction **diversity**: many *kinds* of task teach "follow instructions" better than more data of one kind.

Knob 2: predict first – does this help a *small* model?

We tune models of **422M, 2B, 8B, 68B, 137B** the same way. Instruction tuning clearly helps the 137B. **Before the next slide: what happens to the 422M model?**

(a) Helps, just less – smaller model, smaller boost.

(b) No effect – too small to notice.

(c) It **hurts** – the small model gets *worse* at unseen tasks.

Knob 2: predict first – does this help a *small* model?

We tune models of **422M, 2B, 8B, 68B, 137B** the same way. Instruction tuning clearly helps the 137B. **Before the next slide: what happens to the 422M model?**

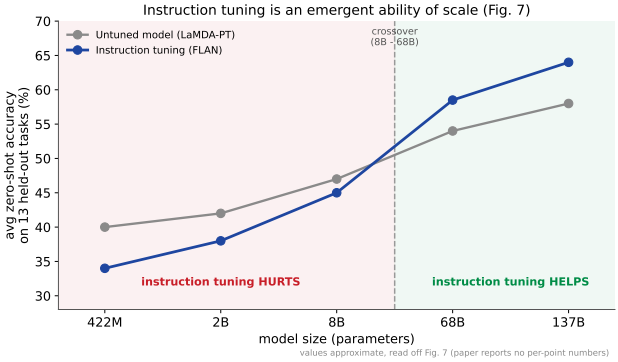
(a) Helps, just less – smaller model, smaller boost.

(b) No effect – too small to notice.

(c) It **hurts** – the small model gets *worse* at unseen tasks.

Answer: **(c)**. Below a threshold, instruction tuning **hurts** zero-shot generalization. The benefit is an **emergent ability of scale** – the same shape as chain-of-thought's emergence (LLM-14).

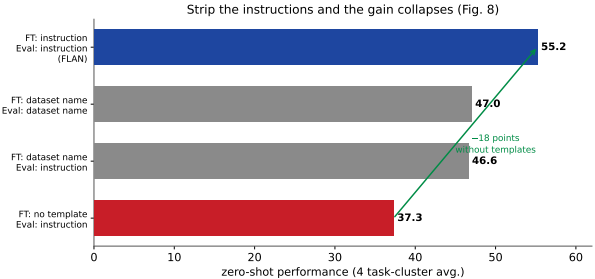
Knob 2: instruction tuning is emergent with scale



- **68B and 137B:** instruction tuning **helps** – big gains on held-out tasks.
- **8B and below:** it **hurts** – the tuned model does *worse* than the untuned one.
- The **crossover is between 8B and 68B**.

Likely why: a small model spends its *whole* capacity just fitting the ∼40 fine-tuning tasks, leaving nothing for a transferable "follow instructions" skill. A big model fits them *and* has room to spare.

Knob 3: are the instructions actually necessary?



Maybe the gain is just **multitask exposure** and the instructions are decoration? Test it: finetune on the same datasets but **strip the templates**.

- **FLAN** (real instructions): **55.2**
- Dataset-name tag only: **46.6 / 47.0**
- **No template** (plain input→output): **37.3**

Remove the instructions and the gain **collapses**. It is the *instructions*, not merely multitask finetuning, that teaches the skill.

Wrapping up

Two theses about SFT data, and where FLAN sits in the post-training story.

FLAN vs LIMA: the matched pair

Both take a pretrained LM and do **supervised finetuning** to make it usable. They disagree on *what data*.

	FLAN (LLM-18)	**LIMA (LLM-4)**
Thesis	scale up the *tasks*	curate a few *examples*
Data	62 datasets, 12 clusters	1,000 hand-picked prompts
Goal	zero-shot on **unseen task types**	helpful, well-formatted answers
Bet	instruction **diversity**	example **quality**

Not a contradiction: FLAN maximizes *task coverage* for cross-task generalization; LIMA argues the *format/quality* of a small set is enough once the base model is strong. Different questions, different answers.

Recap

- **Instruction tuning** = multitask finetuning where every example is phrased as an instruction: (instruction, input) $\rightarrow$ output. It teaches a transferable **"follow instructions"** skill.
- **Setup**: 137B LaMDA-PT, **62 datasets / 12 clusters**, $\sim$10 templates each; "unseen" means the **whole cluster** is held out.
- **Result**: zero-shot FLAN $>$ zero-shot GPT-3 175B on **20/25 datasets**, $>$ few-shot GPT-3 on **10**. But it does *not* help tasks already close to the LM objective (**3 of 7** commonsense/coreference).
- **What matters**: many clusters (diversity), **scale** (helps $\geq$68B, **hurts** $\leq$8B), and the **instructions themselves** (strip them, gain collapses: 55.2 $\rightarrow$ 37.3).

Next: *[LLM-4] LIMA* is the deliberate counterpoint – quality over quantity. And FLAN's "instructions as SFT data" is the engine behind step 1 of *[LLM-10] InstructGPT* and modern post-training. (Follow-up, do not conflate: *FLAN-T5* / "Scaling Instruction-Finetuned LMs," 2022 – same idea, bigger and with CoT data.)