Crosslingual Generalization through Multitask Finetuning

Niklas Muennighoff[1] **Thomas Wang**[1] **Lintang Sutawika**[2,3] **Adam Roberts**[4] **Stella Biderman**[3,5]
Teven Le Scao[1] **M Saiful Bari**[6] **Sheng Shen**[7] **Zheng-Xin Yong**[8] **Hailey Schoelkopf** [3,9]
Xiangru Tang [9] **Dragomir Radev**[9] **Alham Fikri Aji**[10*] **Khalid Almubarak**[11]
Samuel Albanie[12] **Zaid Alyafeai**[13] **Albert Webson**[8] **Edward Raff**[5] **Colin Raffel**[1]
[1]Hugging Face [2]Datasaur.ai [3]EleutherAI [4]Google Research, Brain Team [5]Booz Allen Hamilton
[6]Nanyang Technological University [7] UC Berkeley [8] Brown University
[9] Yale University [10] Amazon [11] PSAU [12] University of Cambridge [13] KFUPM
niklas@hf.co

AntNLP

Questions

- Task generalization on monolingual.
- Task generalization cross languages.

Models

- **BLOOMZ-P3 / mT0-P3:** Models finetuned on the English-only P3.
- **BLOOMZ / mT0:** Models finetuned on xP3, which consists of multilingual datasets with English prompts.
- **BLOOMZ-MT / mT0-MT:** Models finetuned on xP3mt, which consists of multilingual datasets with English and machine-translated prompts.

P3 and xP3

corpus

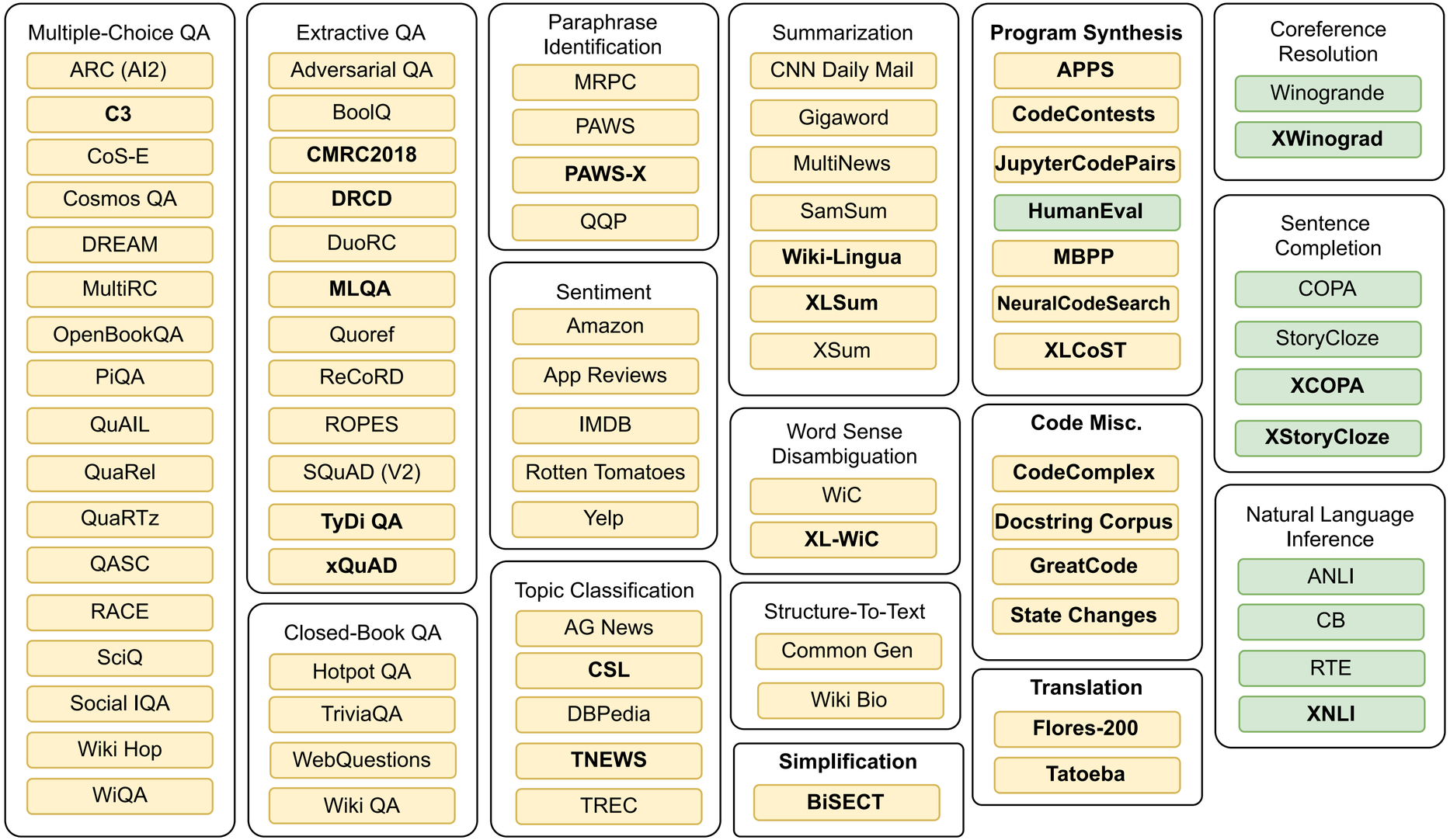


Figure 1: An overview of datasets in xP3. Datasets added to P3 in this work are marked **bold**. Yellow datasets are trained on. Green datasets are held out for evaluation.

P3, xP3 and xP3mt

English-only

Multilingual datasets with English prompts

Multilingual datasets with English and machine translated prompts.

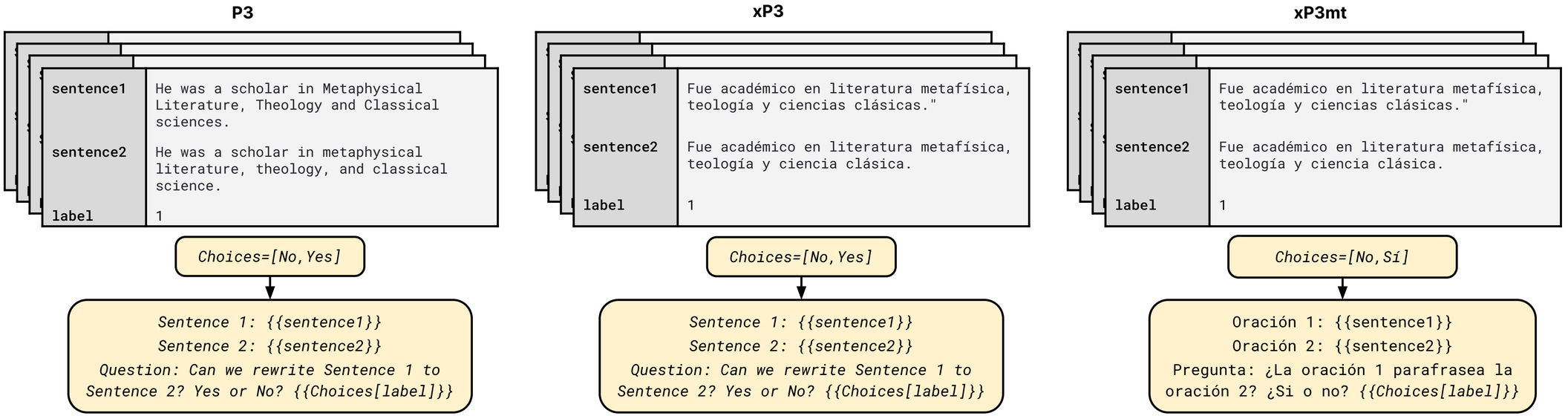


Figure 3: Comparison of dataset variants P3, xP3, and xP3mt on a sample from PAWS for P3 (Zhang et al., 2019) and PAWS-X (Yang et al., 2019) for xP3 and xP3mt. P3 pairs English datasets with English prompts, xP3 pairs multilingual datasets with English prompts and xP3mt pairs multilingual datasets with prompts machine-translated from English to match the dataset language. Expressions in curly brackets are replaced, e.g. for xP3mt the target shown as {{Choices[label]}}) becomes Sí.

ROOTS

corpus

Figure 3: Graphical overview of the ROOTS corpus. **Left:** A treemap plot of the language families of all 46 natural languages where surface is proportional to the number of bytes. Indo-European and Sino-Tibetan families overwhelm the plot with a combined total of 1321.89 GB. The thin orange surface represents 18GB of Indonesian data and the green rectangle 0.4GB constituting the Niger-Congo language family subset. **Right:** A waffle plot of the distribution of the 13 programming languages by size, where one square represents approximately 200MB.

Language	ISO-639-3	catalog-ref	Genus	Family	Macroarea	Size in Bytes
Akan	aka	ak	Kwa	Niger-Congo	Africa	70,1554
Arabic	arb	ar	Semitic	Afro-Asiatic	Eurasia	74,854,900,600
Assamese	asm	as	Indic	Indo-European	Eurasia	291,522,098
Bambara	bam	bm	Western Mande	Mande	Africa	391,747
Basque	eus	eu	Basque	Basque	Eurasia	2,360,470,848
Bengali	ben	bn	Indic	Indo-European	Eurasia	18,606,823,104
Catalan	cat	ca	Romance	Indo-European	Eurasia	17,792,493,289
Chichewa	nya	ny	Bantoid	Niger-Congo	Africa	1,187,405
chiShona	sna	sn	Bantoid	Niger-Congo	Africa	6,638,639
Chitumbuka	tum	tum	Bantoid	Niger-Congo	Africa	170,360
English	eng	en	Germanic	Indo-European	Eurasia	484,953,009,124
Fon	fon	fon	Kwa	Niger-Congo	Africa	2,478,546
French	fra	fr	Romance	Indo-European	Eurasia	208,242,620,434
Gujarati	guj	gu	Indic	Indo-European	Eurasia	1,199,986,460
Hindi	hin	hi	Indic	Indo-European	Eurasia	24,622,119,985
Igbo	ibo	ig	Igboid	Niger-Congo	Africa	14078,521
Indonesian	ind	id	Malayo-Sumbawan	Austronesian	Papunesia	19,972,325,222
isiXhosa	xho	xh	Bantoid	Niger-Congo	Africa	14,304,074
isiZulu	zul	zu	Bantoid	Niger-Congo	Africa	8,511,561
Kannada	kan	kn	Southern Dravidian	Dravidian	Eurasia	2,098,453,560
Kikuyu	kik	ki	Bantoid	Niger-Congo	Africa	359,615
Kinyarwanda	kin	rw	Bantoid	Niger-Congo	Africa	40,428,299
Kirundi	run	rn	Bantoid	Niger-Congo	Africa	3,272,550
Lingala	lin	ln	Bantoid	Niger-Congo	Africa	1,650,804
Luganda	lug	lg	Bantoid	Niger-Congo	Africa	4,568,367
Malayalam	mal	ml	Southern Dravidian	Dravidian	Eurasia	3,662,571,498
Marathi	mar	mr	Indic	Indo-European	Eurasia	1,775,483,122
Nepali	nep	ne	Indic	Indo-European	Eurasia	2,551,307,393
Northern Sotho	nso	nso	Bantoid	Niger-Congo	Africa	1,764,506
Odia	ori	or	Indic	Indo-European	Eurasia	1,157,100,133
Portuguese	por	pt	Romance	Indo-European	Eurasia	79,277,543,375
Punjabi	pan	pa	Indic	Indo-European	Eurasia	1,572,109,752
Sesotho	sot	st	Bantoid	Niger-Congo	Africa	751,034
Setswana	tsn	tn	Bantoid	Niger-Congo	Africa	1,502,200
Simplified Chinese	—	zhs	Chinese	Sino-Tibetan	Eurasia	261,019,433,892
Spanish	spa	es	Romance	Indo-European	Eurasia	175,098,365,045
Swahili	swh	sw	Bantoid	Niger-Congo	Africa	236,482,543
Tamil	tam	ta	Southern Dravidian	Dravidian	Eurasia	7,989,206,220
Telugu	tel	te	South-Central Dravidian	Dravidian	Eurasia	2993407,159
Traditional Chinese	—	zht	Chinese	Sino-Tibetan	Eurasia	762,489,150
Twi	twi	tw	Kwa	Niger-Congo	Africa	1,265,041
Urdu	urd	ur	Indic	Indo-European	Eurasia	2,781,329,959
Vietnamese	vie	vi	Viet-Muong	Austro-Asiatic	Eurasia	43,709,279,959
Wolof	wol	wo	Wolof	Niger-Congo	Africa	3,606,973
Xitsonga	tso	ts	Bantoid	Niger-Congo	Africa	707,634
Yoruba	yor	yo	Defoid	Niger-Congo	Africa	89,695,835
Programming Languages	—	—	—	—		174,700,245,772

Table 1: Linguistic makeup of the ROOTS corpus.

Language composition

In xP3, Twi and others are represented solely as a translation task using data from Flores200 (NLLB Team et al., 2022).

No Language Left Behind: Scaling Human-Centered Machine Translation

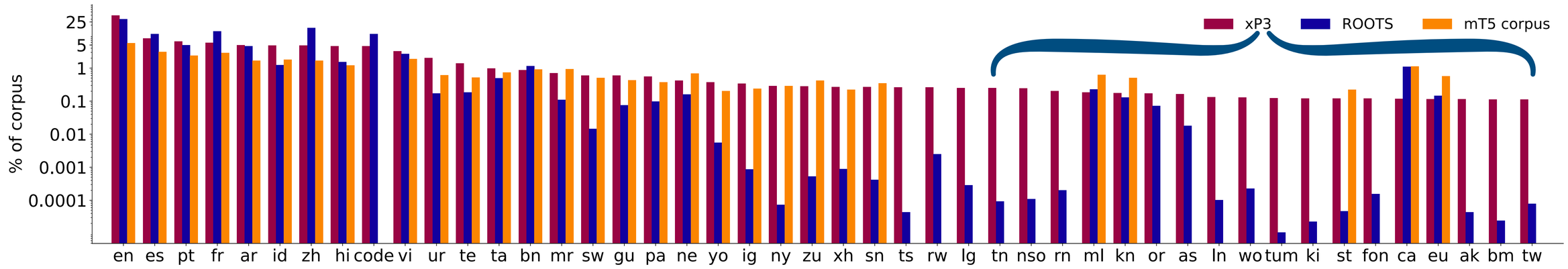
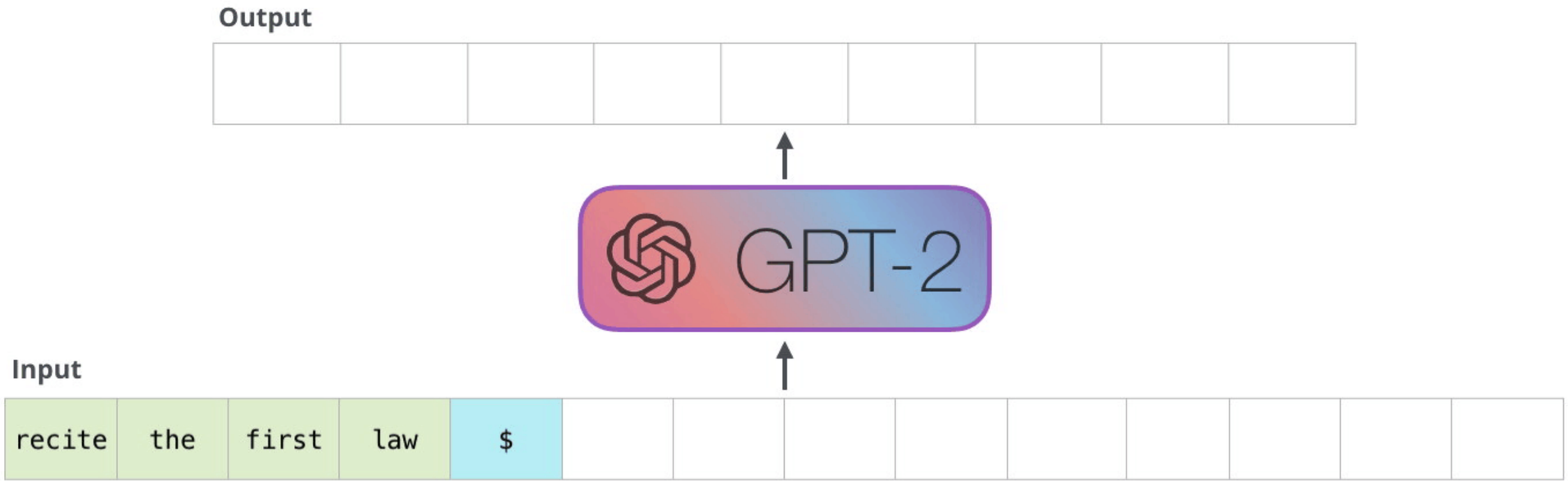


Figure 2: Language composition of xP3, ROOTS, and the corpus of mT5. All ROOTS and xP3 languages are depicted. The mT5 corpus covers additional languages that are not included in the graph.

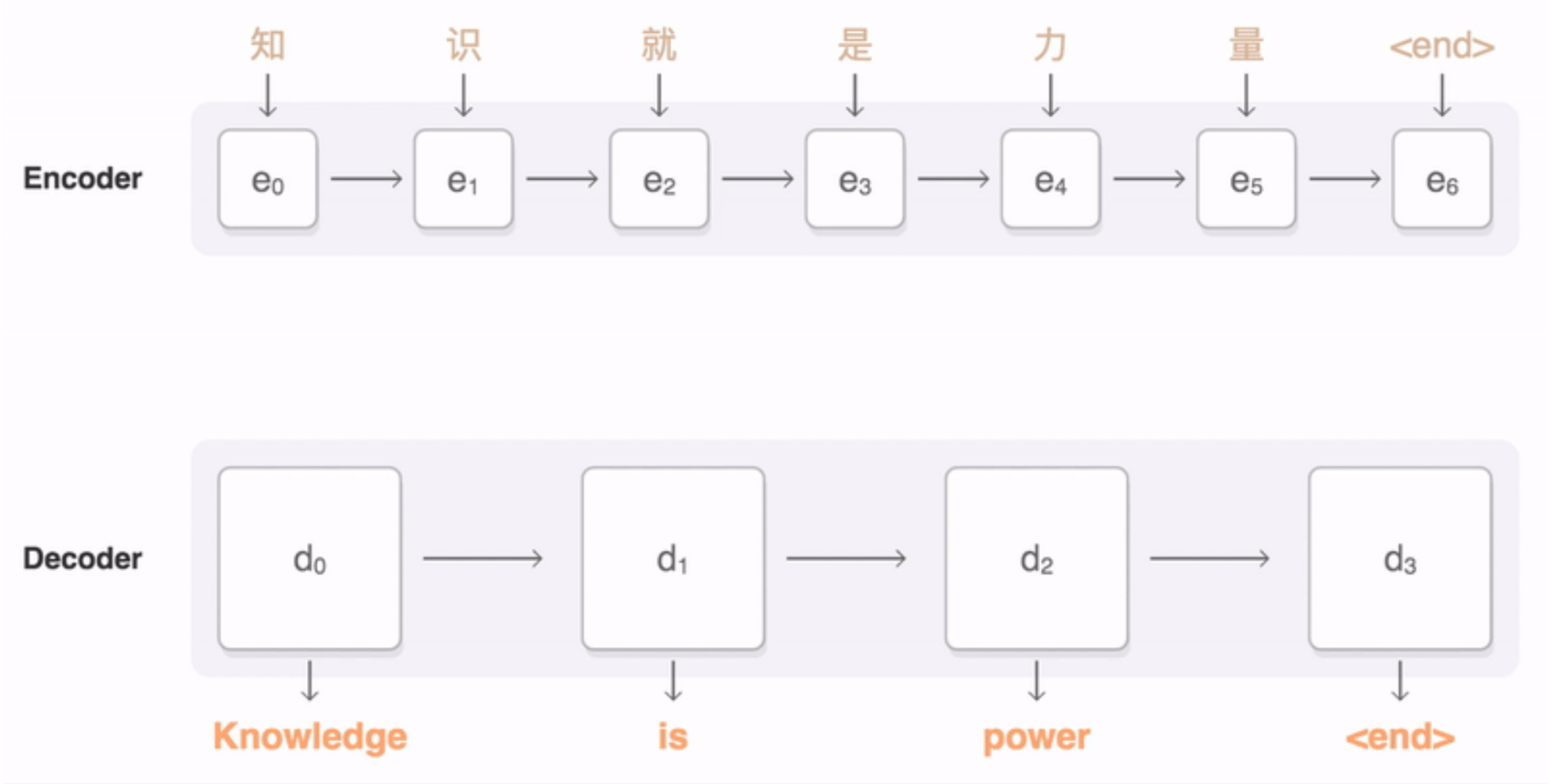
How to do instruction tuning?

Decoder only (GPT-x, BLOOM)



How to do instruction tuning?

Encoder-Decoder (T5, BART)



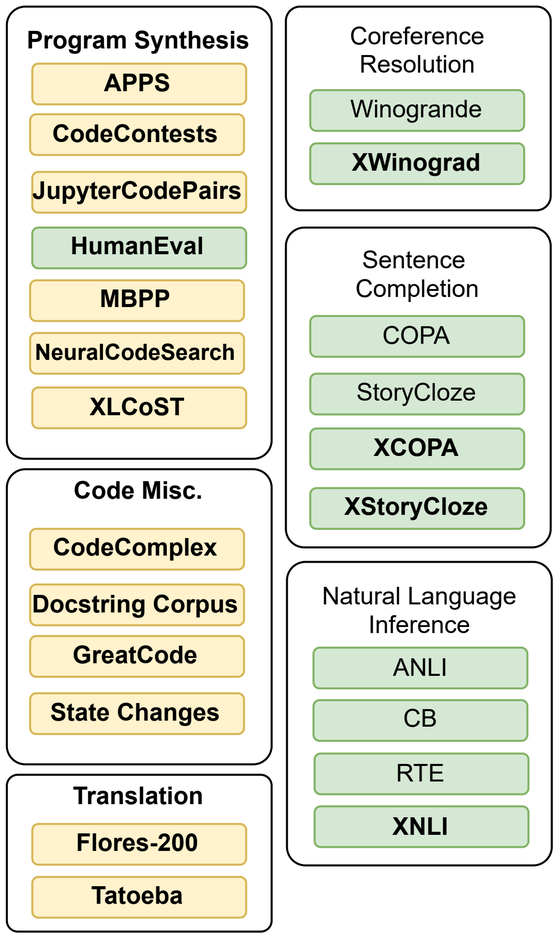
How to do instruction tuning?

1. Loss only being computed on target tokens

2. Downscale the loss of each token by the length of the target it belongs to

3. Skip samples longer than 2048 tokens and use packing to train efficiently on multiple samples at a time

How to evaluate?



1. Compute the log-likelihood of each possible completion and select the highest scoring option.
2. Select 5 prompts at random from PromptSource and use them for all language splits of the dataset.
3. Report the median of the 5 prompts for results per language split.
4. Do not tune prompts based on performance on validation data.

PromptSource

hf.co

Original Task?
True

Choices in template?
True

Metrics
Accuracy

Prompt Languages
en (English)

Answer Choices

```
World politics ||| Sports ||| Business ||| Science and technology
```

Jinja template

Input template

```
{{text}}
Is this a piece of news regarding {{"world politics, sports, business, or science and technology"}}?
```

Target template

```
{{answer_choices[label] }}
```


```
{
  "text" :
  "Wall St. Bears Claw Back Into the Black (Reuters)
  Reuters - Short-sellers, Wall Street's
  dwindling\band of ultra-cynics, are seeing green
  again."
  "label" : 2
}
```


Input

Wall St. Bears Claw Back Into the Black (Reuters) Reuter again.
Is this a piece of news regarding world politics, sports

Target

Business

Promptsource (Read only) 🌸 - Prompted dataset viewer

Dataset

ag_news

Split

train

No of prompts created for `ag_news` : 7

Prompt name

recommend

Select the example index (Size = 120000)

0 − +

Dataset Schema


```
{
  "text" : "string"
  "label" : [
    0 : "World"
    1 : "Sports"
    2 : "Business"
    3 : "Sci/Tech"
  ]
}
```

HumanEval

A set of 164 hand-written programming problems.

Each problem includes a function signature, docstring, body, and several unit tests, with an average of 7.7 tests per problem.

```python
def incr_list(l: list):
    """Return list with elements incremented by 1.
    >>> incr_list([1, 2, 3])
    [2, 3, 4]
    >>> incr_list([5, 3, 5, 2, 3, 3, 9, 0, 123])
    [6, 4, 6, 3, 4, 4, 10, 1, 124]
    """
    return [i + 1 for i in l]
```

```python
def solution(lst):
    """Given a non-empty list of integers, return the sum of all of the odd elements
    that are in even positions.

    Examples
    solution([5, 8, 7, 1]) ==>12
    solution([3, 3, 3, 3, 3]) ==>9
    solution([30, 13, 24, 321]) ==>0
    """
    return sum(lst[i] for i in range(0,len(lst)) if i % 2 == 0 and lst[i] % 2 == 1)
```

```python
def encode_cyclic(s: str):
    """
    returns encoded string by cycling groups of three characters.
    """
    # split string to groups. Each of length 3.
    groups = [s[(3 * i):min((3 * i + 3), len(s))] for i in range((len(s) + 2) // 3)]
    # cycle elements in each group. Unless group has fewer elements than 3.
    groups = [(group[1:] + group[0]) if len(group) == 3 else group for group in groups]
    return "".join(groups)

def decode_cyclic(s: str):
    """
    takes as input string encoded with encode_cyclic function. Returns decoded string.
    """
    # split string to groups. Each of length 3.
    groups = [s[(3 * i):min((3 * i + 3), len(s))] for i in range((len(s) + 2) // 3)]
    # cycle elements in each group.
    groups = [(group[-1] + group[:-1]) if len(group) == 3 else group for group in groups]
    return "".join(groups)
```

Zero-shot Generalization

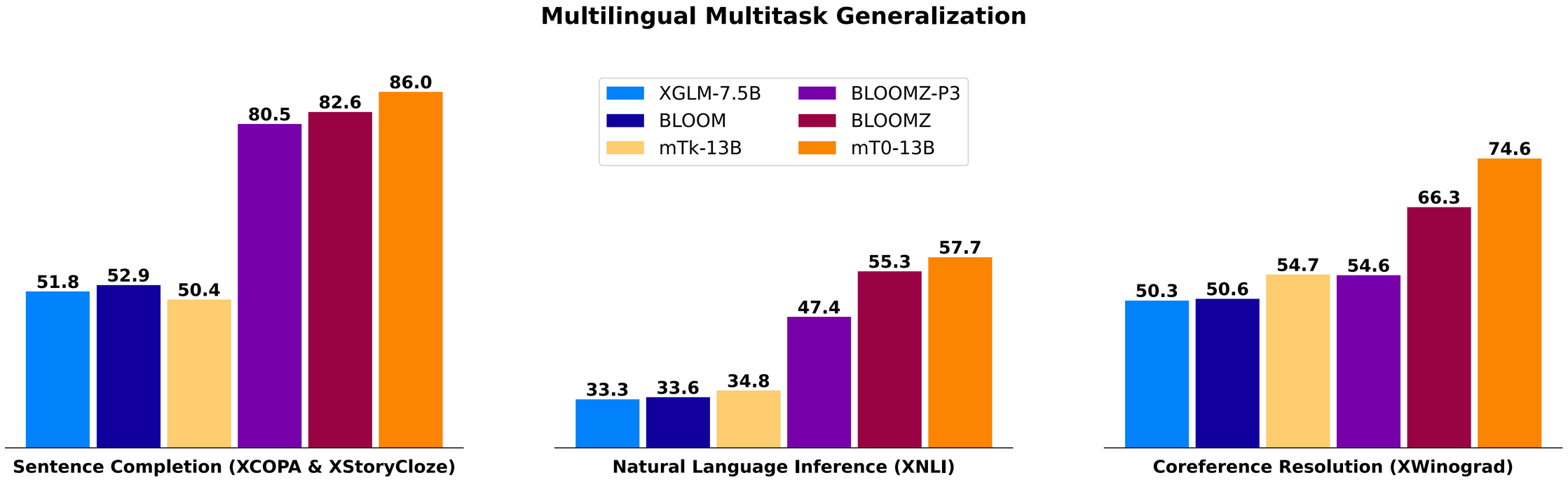


Figure 4: Zero-shot multilingual task generalization with English prompts. BLOOM models have 176 billion parameters. Scores are the language average for each task. Appendix §B breaks down performance by language.

English results

Task generalization on monolingual

1. Fine-tuning on xP3 outperforms P3
2. Multilingual mT0 is stronger than monolingual T0 on English.

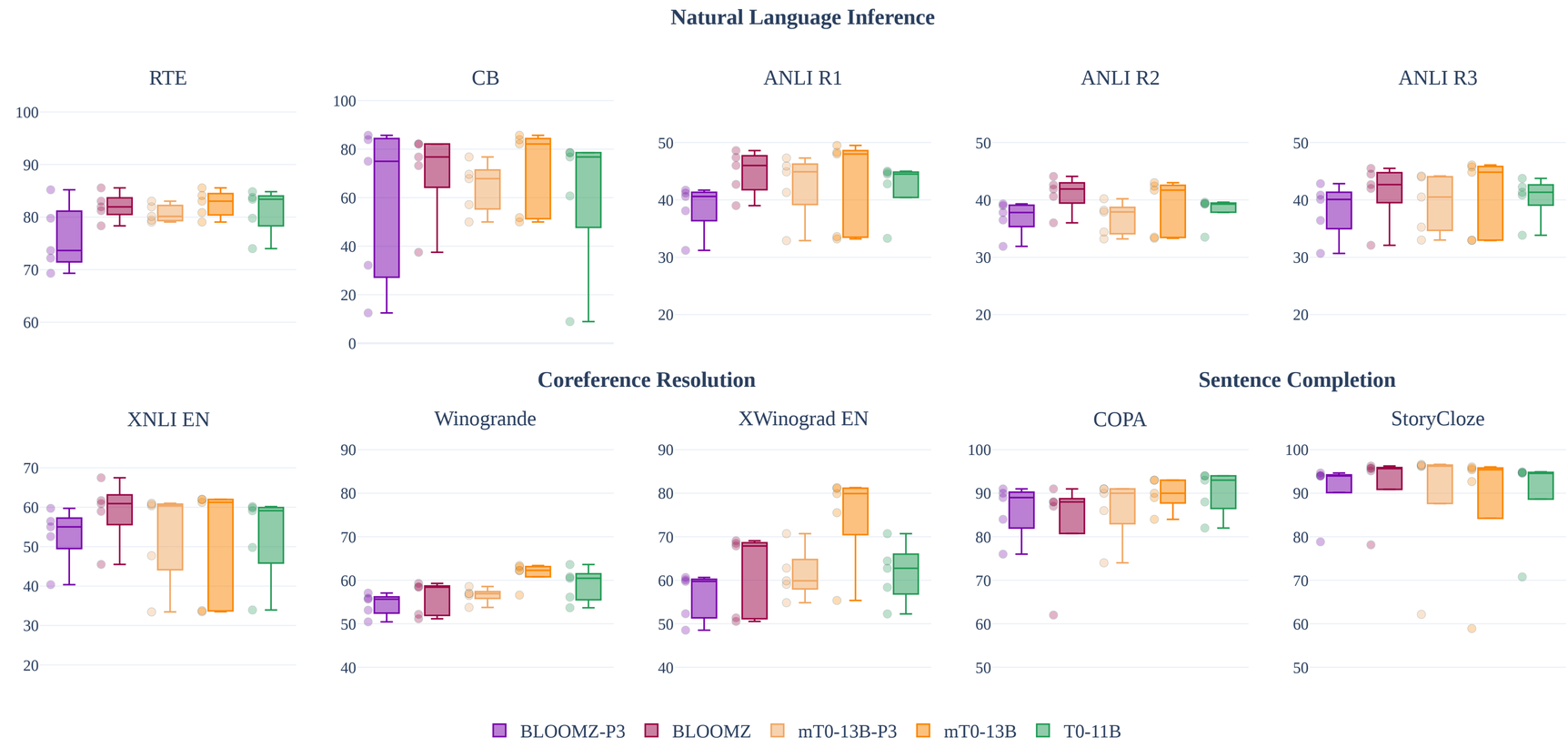
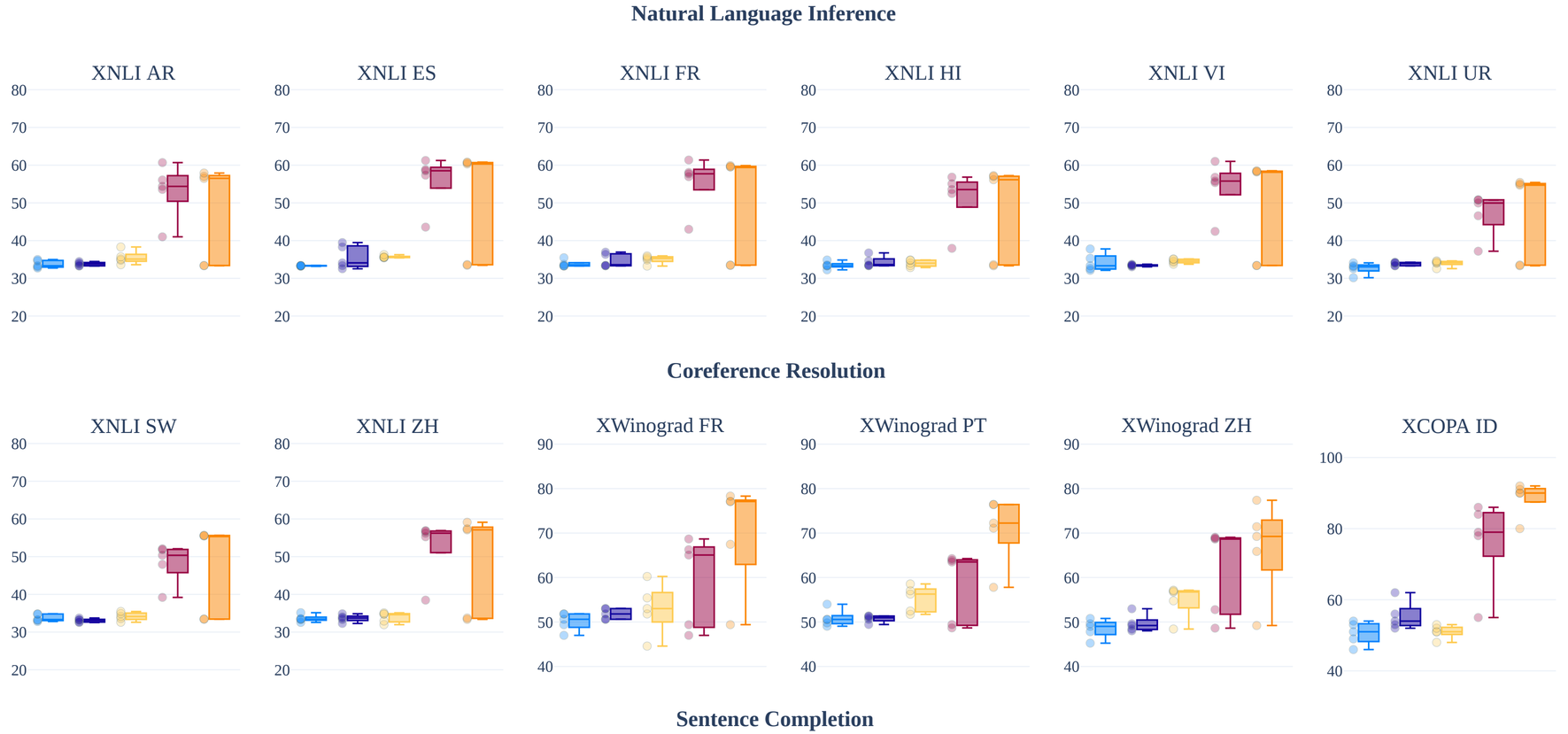


Figure 9: Zero-shot English task generalization. Each dot represents performance on one English evaluation prompt.

Multilingual results

Task generalization cross languages



1. Performance by prompt varies substantially highlighting that prompt engineering may still be necessary after MTF.

2. mT0 consistently outperforms BLOOMZ on Swahili (SW), possibly due to it being a larger part of its pre-training corpus

Multilingual results

Task generalization cross languages

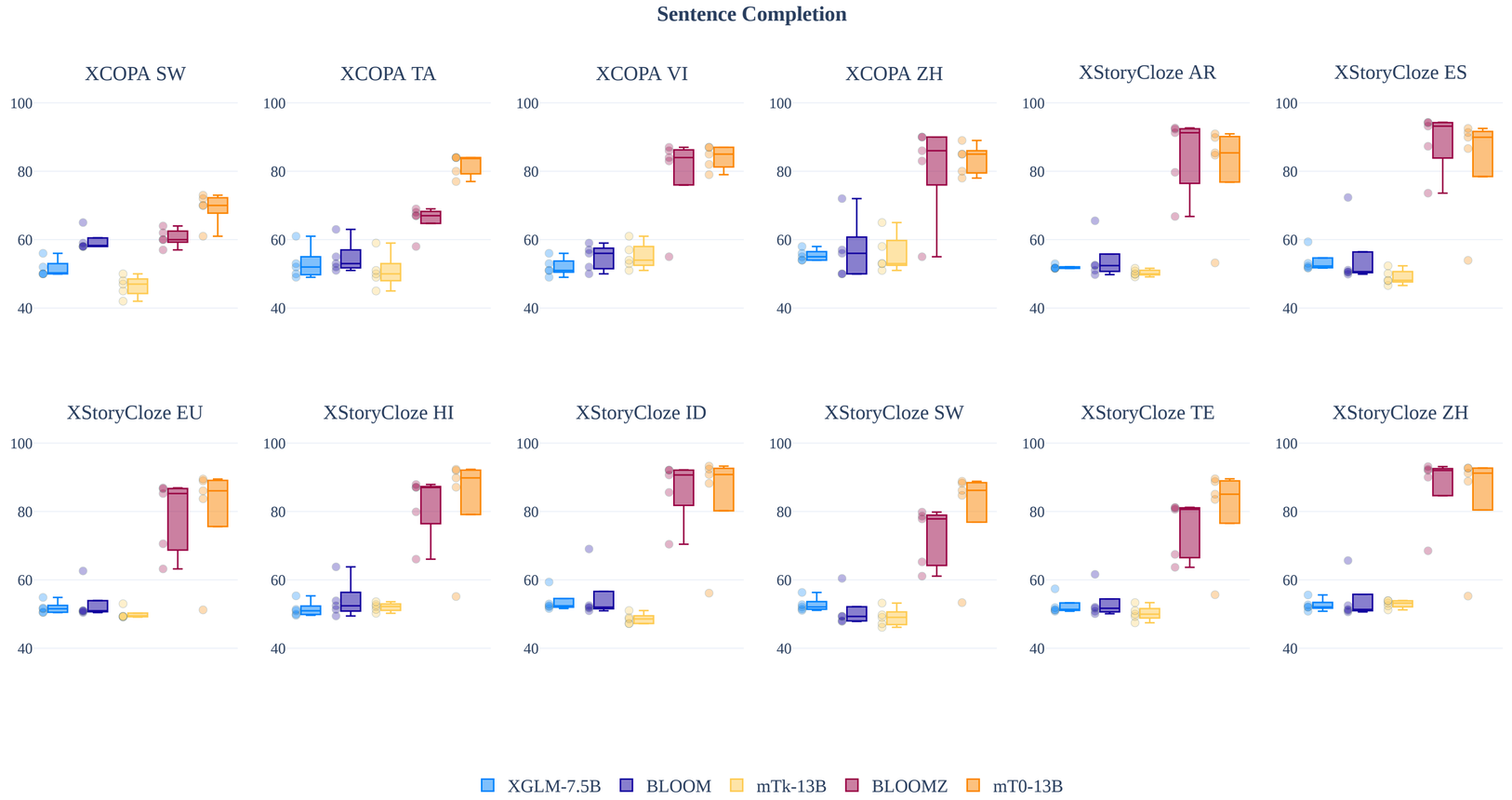


Figure 10: Zero-shot multilingual task generalization on languages seen during pretraining and finetuning. Each dot represents performance on one English evaluation prompt.

1. Performance by prompt varies substantially highlighting that prompt engineering may still be necessary after MTF.

2. mT0 consistently outperforms BLOOMZ on Swahili (SW), possibly due to it being a larger part of its pre-training corpus

Task Generalization on Unseen language

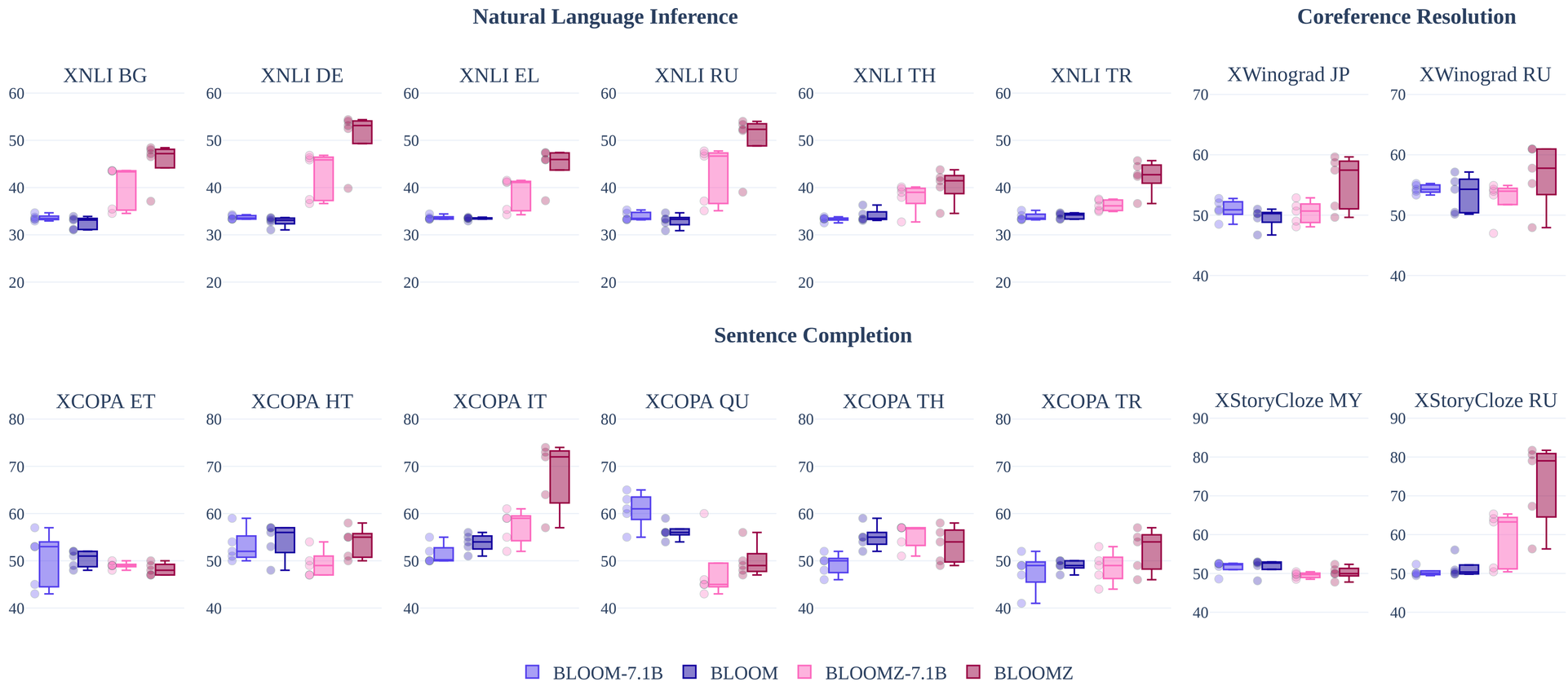


Figure 5: Zero-shot task and language generalization using English prompts on tasks and languages not intentionally seen during pretraining nor finetuning. Language codes are ISO 639-1, except for JP (Japanese).

Not intentionally seen

- It shows that ROOTS contains unintentionally collected languages, such as Burmese (my: 0.00003%), Thai (th: 0.006%), Turkish (tr: 0.03%), Greek (el: 0.03%), Russian (ru: 0.03%), Bulgarian (bg: 0.05%), Estonian (et: 0.06%), Haitian (ht: 0.12%), German (de: 0.21%), Italian (it: 0.28%) and Japanese (ja: 0.54%).

- These “unseen” languages only have small sentence proportions in our subsample compared to English (en: 46.23%), French (fr: 15.73%) and Spanish (es: 13.38%).

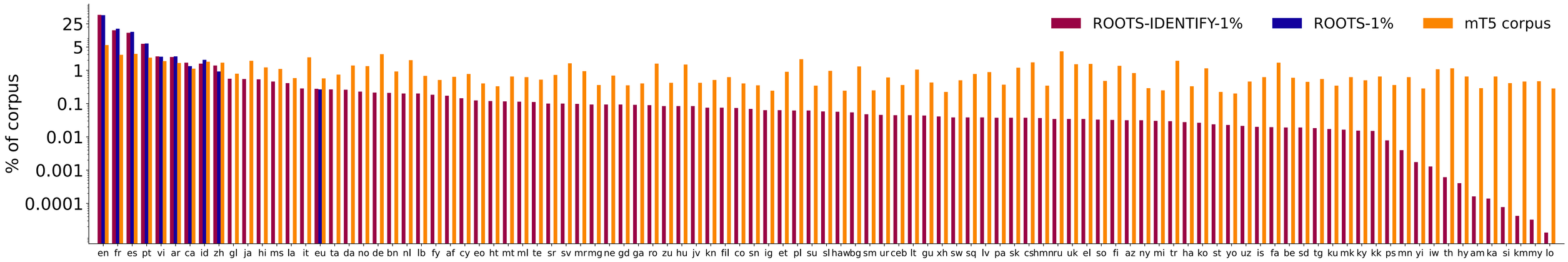


Figure 11: Language composition of ROOTS-IDENTIFY-1%, ROOTS-1% and the mT5 corpus. All mT5 languages are depicted. ROOTS-1% is a random 1% sample of ROOTS with its assigned meta-languages. ROOTS-IDENTIFY-1% are the actual languages in ROOTS-1% re-identified using `cld3`.

Multilingual prompting

Task	Prompt	Average accuracy			
		BLOOMZ	BLOOMZ-MT	mT0-13B	mT0-13B-MT
XNLI	EN	**53.58**	49.74	48.43	**51.52**
	MT	37.87	**42.03**	39.83	**42.64**
	HT	41.13	**44.55**	45.19	**47.03**
XCOPA	EN	75.5	**75.75**	**84.45**	81.6
	MT	71.95	**74.25**	**82.9**	81.1
XStoryCloze	EN	**84.42**	84.07	82.52	**82.58**
	MT	84.37	**85.31**	**84.01**	83.31
XWinograd	EN	**60.07**	59.15	70.49	**73.24**
	MT	58.48	**60.14**	66.89	**72.33**

Table 1: Comparison between EN (English), MT (machine-translated) and HT (human-translated) prompts for 176B BLOOMZ and 13B mT0 models finetuned on either only English or English and machine-translated multilingual prompts (-MT).

Scaling

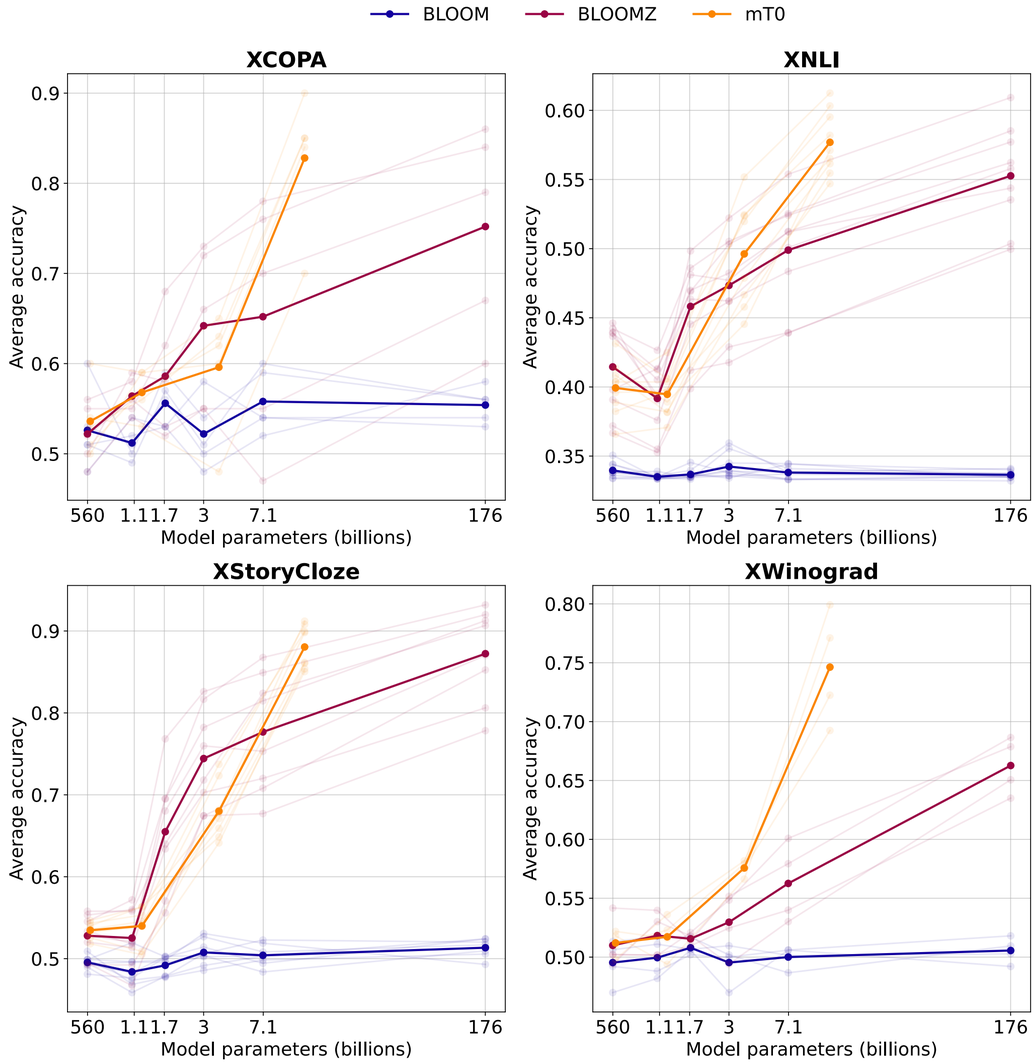


Figure 6: Aggregate performance vs. size. Transparent lines correspond to individual languages, while thick lines are average accuracy scores.

Generation tasks

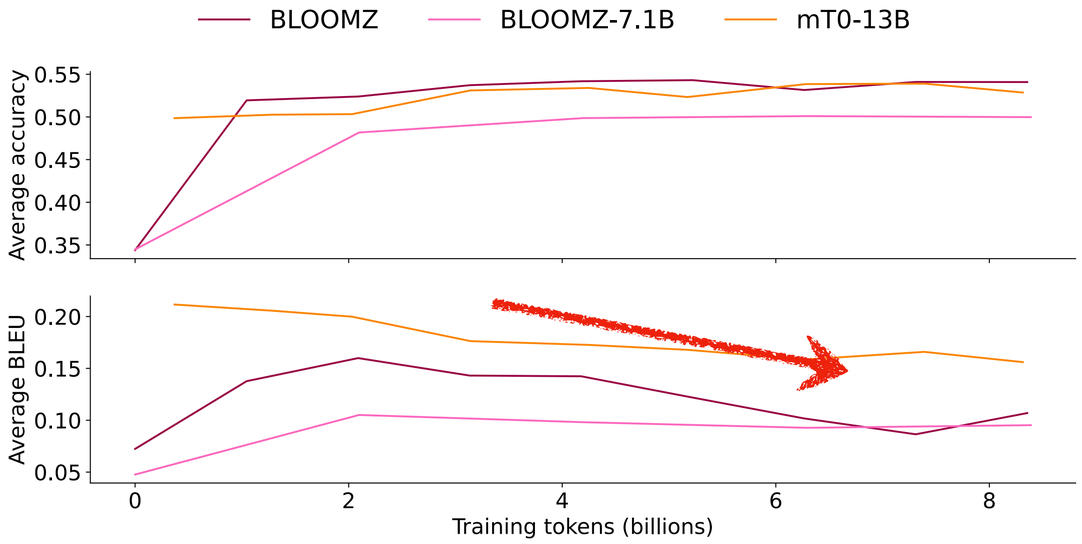


Figure 7: Validation performance during training on natural language understanding (NLU) and natural language generation (NLG) tasks. The former are scored using accuracy and the latter using BLEU (Papineni et al., 2002). The NLG tasks measured are translation and summarization.

	Pass@k		
	$k = 1$	$k = 10$	$k = 100$
GPT-Neo 1.3B	4.79%	7.47%	16.30%
GPT-Neo 2.7B	6.41%	11.27%	21.37%
GPT-J 6B	11.62%	15.74%	27.74%
GPT-NeoX 20B	15.4%	25.6%	41.2%
Codex-300M	13.17%	20.37%	36.27%
Codex-679M	16.22%	25.7%	40.95%
Codex-2.5B	21.36%	35.42%	59.5%
Codex-12B	28.81%	46.81%	72.31%
BLOOM-560M	0.82%	3.02%	5.91%
BLOOM-1.1B	2.48%	5.93%	9.62%
BLOOM-1.7B	4.03%	7.45%	12.75%
BLOOM-3B	6.48%	11.35%	20.43%
BLOOM-7.1B	7.73%	17.38%	29.47%
BLOOM	15.52%	32.20%	55.45%
BLOOMZ-560M	2.18 %	4.11%	9.00%
BLOOMZ-1.1B	2.63%	6.22%	11.68%
BLOOMZ-1.7B	4.38%	8.73%	16.09%
BLOOMZ-3B	6.29%	11.94%	19.06%
BLOOMZ-7.1B	8.06%	15.03%	27.49%
BLOOMZ	12.06%	26.53%	48.44%
BLOOMZ-P3	6.13%	11.79%	18.73%

Table 2: Code continuation on HumanEval. Non-BLOOM results come from prior work (Chen et al., 2021; Fried et al., 2022). Codex is a language model finetuned on code, while the GPT models (Black et al., 2021; Wang and Komatsuzaki, 2021; Black et al., 2022) are trained on a mix of code and text like BLOOM.

Code generation

```python
from typing import List


def has_close_elements(numbers: List[float], threshold: float) -> bool:
    """ Check if in given list of numbers, are any two numbers closer to each other than
    given threshold.
    >>> has_close_elements([1.0, 2.0, 3.0], 0.5)
    False
    >>> has_close_elements([1.0, 2.8, 3.0, 4.0, 5.0, 2.0], 0.3)
    True
    """
    if len(numbers) <= 1:
        return False
    numbers = sorted(numbers)
    i = 0
    while i < len(numbers) - 1:
        # Get distance between current and next elements
        distance = numbers[i] - numbers[i + 1]

        # Check if distance is closer than threshold
        if distance < threshold:
            return True
        i += 1
    return False
```

(a) BLOOM

```python
from typing import List


def has_close_elements(numbers: List[float], threshold: float) -> bool:
    """ Check if in given list of numbers, are any two numbers closer to each other than
    given threshold.
    >>> has_close_elements([1.0, 2.0, 3.0], 0.5)
    False
    >>> has_close_elements([1.0, 2.8, 3.0, 4.0, 5.0, 2.0], 0.3)
    True
    """
    numbers = sorted(numbers)
    return any(numbers[i + 1] - numbers[i] < threshold for i in range(len(numbers) - 1))
```

(b) BLOOMZ

Figure 12: Code generations of BLOOM and BLOOMZ on HumanEval. The model is prompted to generate after the final """. The generation is stopped after an end-of-sequence token or a return statement followed by a newline.

Data (→)	HumanEval generations		Targets of xP3
	BLOOM	BLOOMZ	code datasets
Average characters	247	144	531
Average Python comments (#)	0.69	0.04	0.85

Table 4: Number of characters and comments for generations and targets in the finetuning corpus.

Effect of language proportions

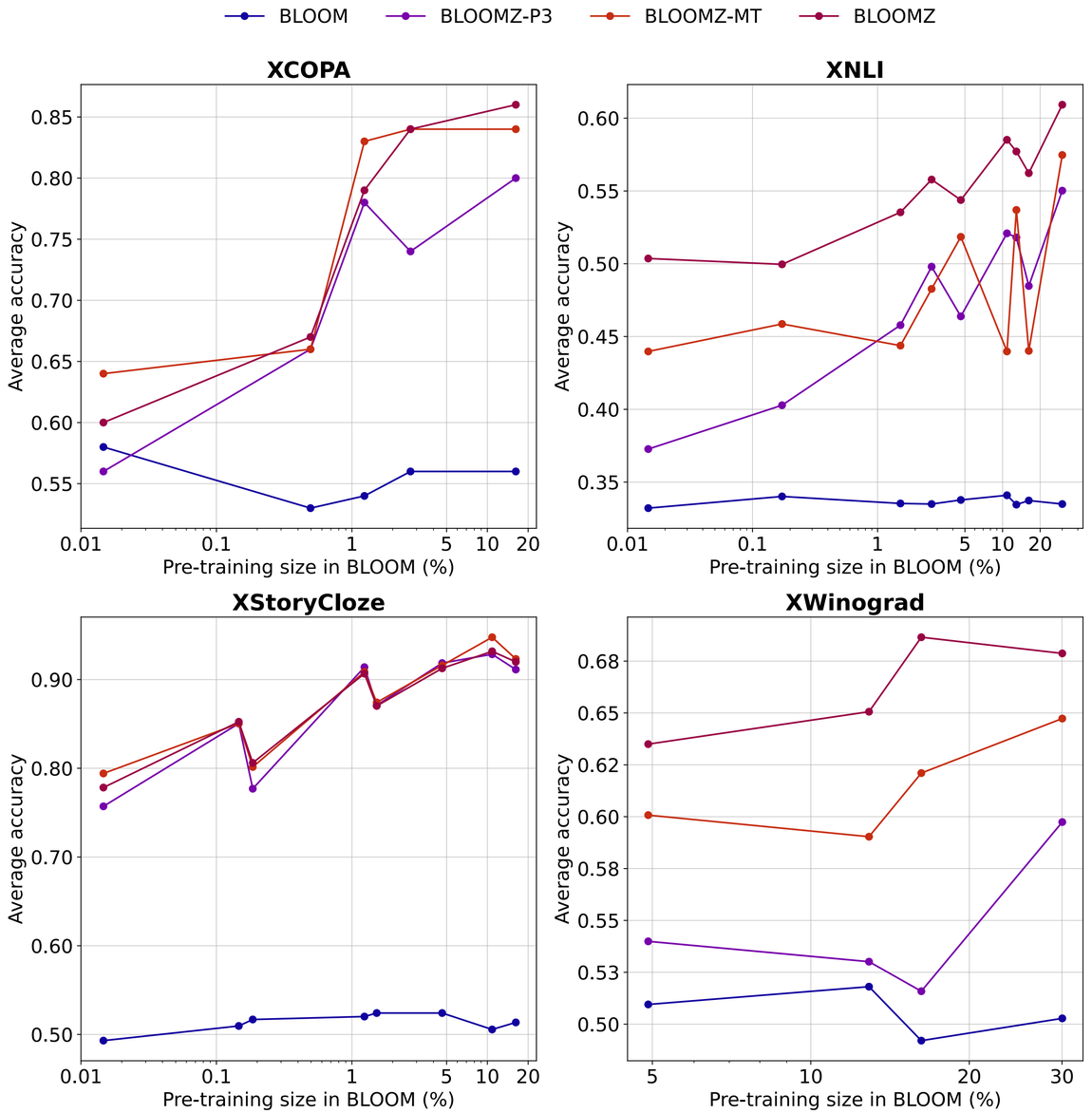


Figure 8: Performance across languages by size in the BLOOM pretraining corpus, ROOTS.

Timeline of Public Instruction Tuning Collections

		Model Details				Data Collection & Training Details			
Release	Collection	Model	Base	Size	Public?	Prompt Types	Tasks in Flan	# Exs	Methods
2020 05	UnifiedQA	UnifiedQA	RoBerta	110-340M	P	ZS	46 / 46	750k	
2021 04	CrossFit	BART-CrossFit	BART	140M	NP	FS	115 / 159	71.M	
2021 04	Natural Inst v1.0	Gen. BART	BART	140M	NP	ZS / FS	61 / 61	620k	+ Detailed k-shot Prompts
2021 09	Flan 2021	Flan-LaMDA	LaMDA	137B	NP	ZS / FS	62 / 62	4.4M	+ Template Variety
2021 10	P3	T0, T0+, T0++	T5-LM	3-11B	P	ZS	62 / 62	12M	+ Template Variety + Input Inversion
2021 10	MetaICL	MetaICL	GPT-2	770M	P	FS	100 / 142	3.5M	+ Input Inversion + Noisy Channel Opt
2021 11	ExMix	ExT5	T5	220M-11B	NP	ZS	72 / 107	500k	+ With Pretraining
2022 04	Super-Natural Inst.	*Tk*-Instruct	T5-LM, mT5	11-13B	P	ZS / FS	1556 / 1613	5M	+ Detailed k-shot Prompts + Multilingual
2022 10	GLM	GLM-130B	GLM	130B	P	FS	65 / 77	12M	+ With Pretraining + Bilingual (en, zh-cn)
2022 11	xP3	BLOOMz, mT0	BLOOM, mT5	13-176B	P	ZS	53 / 71	81M	+ Massively Multilingual
2022 12	Unnatural Inst.†	T5-LM-Unnat. Inst.	T5-LM	11B	NP	ZS	~20 / 117	64k	+ Synthetic Data
2022 12	Self-Instruct†	GPT-3 Self Inst.	GPT-3	175B	NP	ZS	Unknown	82k	+ Synthetic Data + Knowledge Distillation
2022 12	OPT-IML Bench†	OPT-IML	OPT	30-175B	P	ZS + FS CoT	~2067 / 2207	18M	+ Template Variety + Input Inversion + Multilingual
2022 10	Flan 2022 (ours)	Flan-T5, Flan-PaLM	T5-LM, PaLM	10M-540B	P NP	ZS + FS CoT	1836	15M	+ Template Variety + Input Inversion + Multilingual

Figure 2: A **Timeline of Public Instruction Tuning Collections** specifies the collection release date, detailed information on the finetuned models (the base model, their size, and whether the model itself is Public (P) or Not Public (NP)), what prompt specification they were trained for (zero-shot, few-shot, or Chain-of-Thought), the number of tasks contained in the Flan 2022 Collection (released with this work), and core methodological contributions in each work.
Note that the number of tasks and of examples vary under different assumptions and so are estimates. For instance, the definition of "task" and "task category" vary by work, and are not easily simplified to one ontology. The reported counts for the number of tasks are reported using task definitions from the respective works.
† indicates concurrent work.