Extending English Large Language Models to New Languages

A Survey

Anoop Kunchukuttan

Microsoft Translator

AI4Bharat

Version 2 (6th August 2024)

Outline

- Introduction to LLMs
- The Multilingual LLM Challenge
- Extending English LLMs
  - Vocabulary Expansion
  - Continued Pre-training
  - Instruction Tuning
- Summary & Future Directions

If you find this survey useful, please cite it in your work

```
@online{kunchukuttan2024extendllm,
	author = {{Anoop Kunchukuttan}},
	title = {Extending English Large Language Models to New Languages: A Survey},
	url = {https://anoopkunchukuttan.gitlab.io/publications/presentations/extend_en_llms_aug2024.pdf},
	date = {6th August 2024},
	urldate = {6th August 2024}
}
```

BIBLIOGRAPHY *(identify paper by PAPER_KEY on the slides)*

What are Large Language Models?

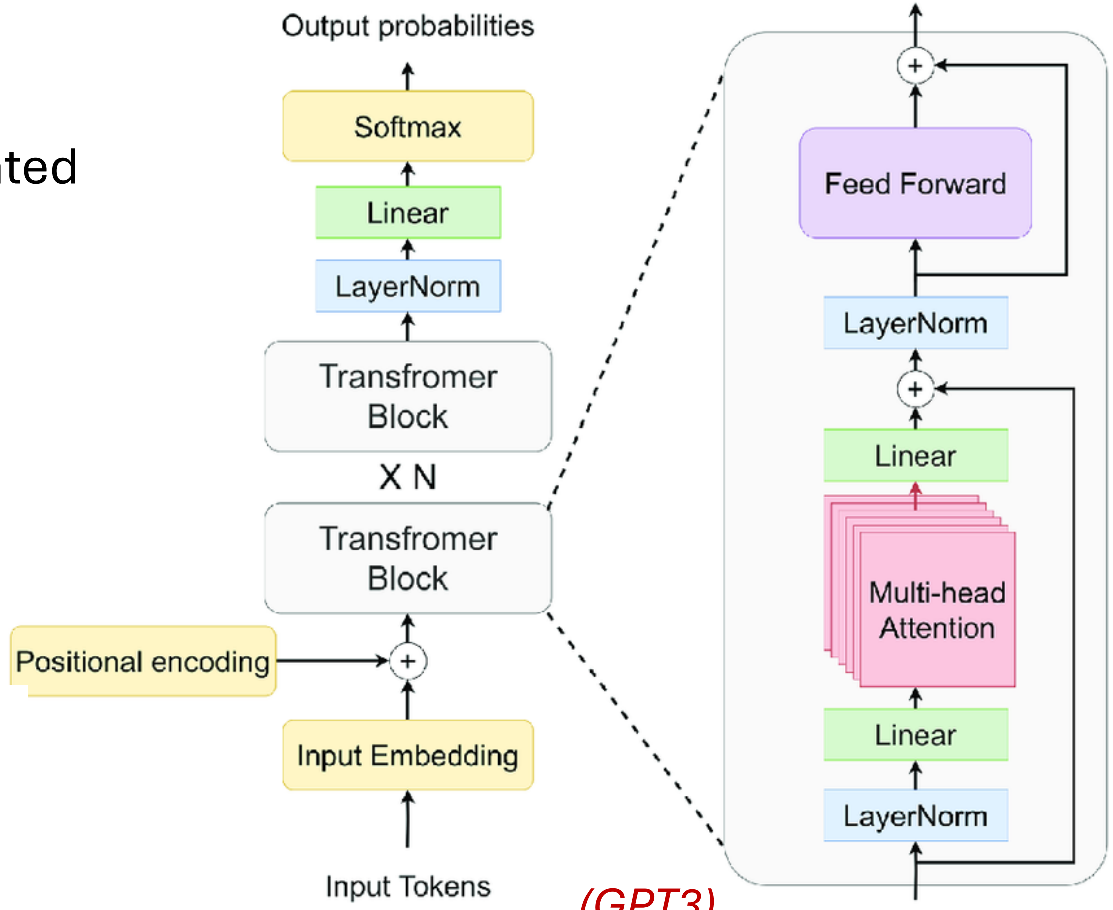
- Typically, transformer decoder models
- They generate text by looking at only previously generated text *(auto-regressive)*
- Trained on a self-supervised task
  - Next word prediction task
  - Large amount of text data
  - Large Models
- In-context learning capability

Instruction Tell me the sentiment of this review

Example The movie begins ….. The plot is engaging, thoroughly enjoyable.
The movie is great

Oh, how can such a fine cast produce such a terrible performance….. A total waste of time.
The movie is pathetic

Prompt It is just a rehash of old movies
The movie is <MASK>



(GPT3)

(InstructGPT)

- Finetuning on (relatively) small supervised and preference data to align instructions and values

Current LLMs *vs.* older generation (BERT/BART/XLM-R)

Current

- Every task is just text completion
- Decoder-only (NLU and NLG)
- In-context learning & Instruction Tuning
- Causal LM training objectives
- Large model size (GPT3: 175B params)
- Trained on large corpora (15T tokens LLama3)

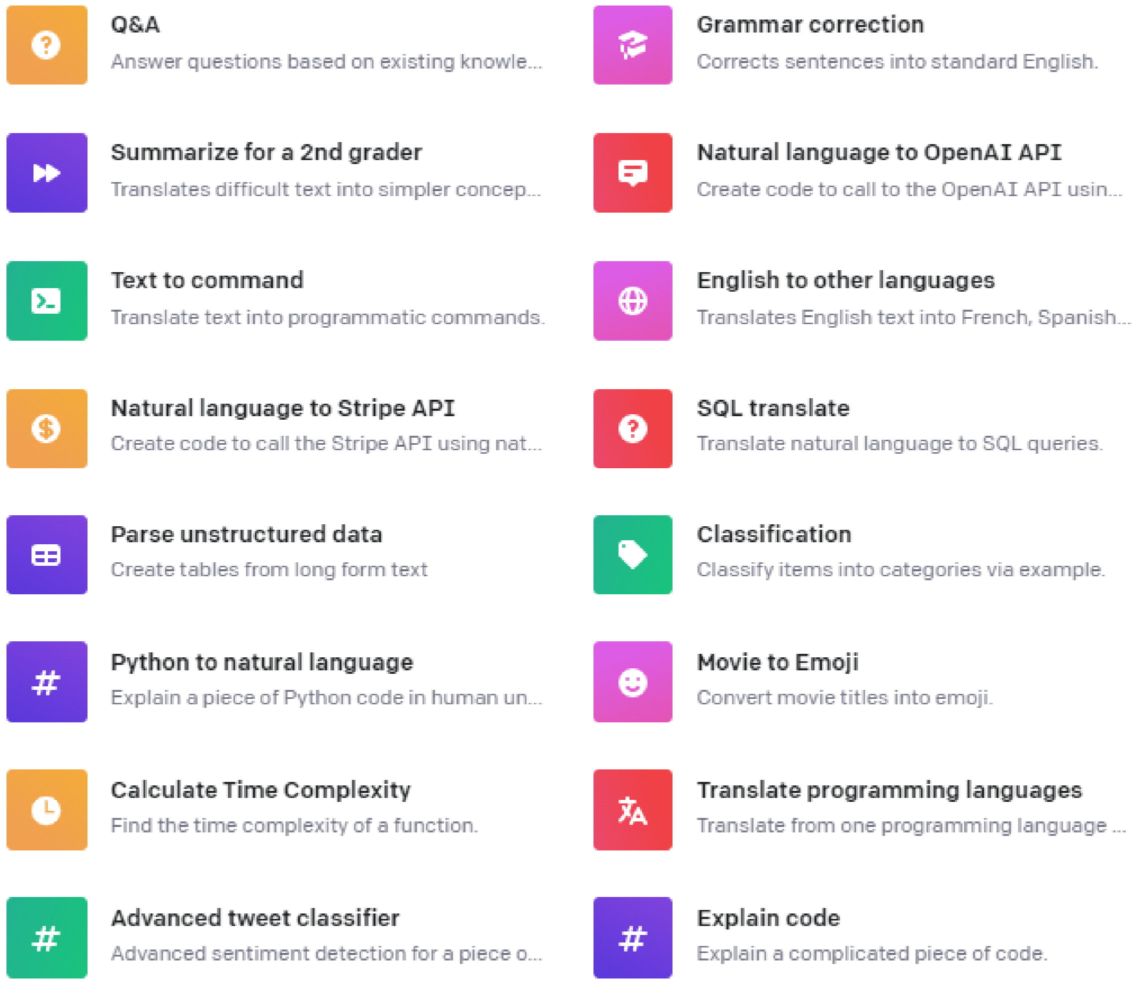
Old Generation

- Classification/text generation
- Encoder-only (NLU), Enc-dec (NLG)
- Per-task finetuning
- Denoising objectives (MLM, DAE)
- Small models (largest mT5: ~13B)
- Modest amount of data (~BERT: 137B tokens)

NLU: natural language understanding
NLG: natural language generation

What makes LLMs exciting?

- Inherent knowledge
- Multi-task abilities and task-composition
- Little task-specific training required
- Help in writing , creativity, brainstorming, data generation, explanation, *etc*.



(SparksOfAGI) https://arxiv.org/abs/2303.12712 https://www.youtube.com/watch?v=qbIk7-JPB2c

Maths

Prompt: Can you write a proof that there are infinitely many primes, with every line that rhymes?

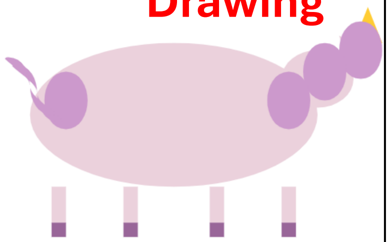
GPT-4:

```
Yes, I think I can,
Though it might take a clever plan.
I'll start by noting Euclid's proof,
Which shows that primes aren't just aloof.
Assume we have a finite list,
Of primes, and that none have been missed.
Multiply them all together,
And add one, just to be clever.
The result will be a number,
That has no prime factors, oh wonder!
But every number has a prime,
That divides it, it's just a matter of time.
So we've found a contradiction,
And our finite list needs eviction.
There must be infinitely many primes,
And that's the end of my rhyming lines.s
```

Drawing

Prompt: Draw a unicorn in TiKZ.

GPT-4: [Produces LaTeX compiling to following picture.]



Prompt: Can you generate a pyplot for the following data: $x = [1, 3, 5, 6, 8]$, $y1 = [2, 3, 5, 18, 1]$, $y2 = [3, 5, 6, 8, 1]$, $y3 = [5, 1, 2, 3, 4]$, $y4 = [9, 7, 2, 3, 1]$. I want $y1, y2$ to be in the same plot, but $y3$ is in another plot next to that plot, $y4$ is in below. I want the legend of $y1$ to be "bob", $y2$ to be "alice", $y3$ to be "bilbo", $y4$ to be "allie". I want the x-axis to be labeled with "time" and y axis to be labeled with "money". I want to add a 10 %-40% random error bar to each curve, through all times (including non-integers). I want smoothed curves to show the plot, and smoothed error bar. Do not use linear interpolation, use smooth interpolation! I want to also add some small zig-zag to the smoothed curve to make it look more real. I want to put a baseline as the mean of each line. I want to put a pie chart below indicating the fraction of the four people in each time step. I also want an animation to show how the fractions are changing in the pie chart in **continuous time**. Interpolate the missing fractions! I also want vertical line animation in other three plots to match the pie chart. I want the fanciest plot. Please add as many fancy things as possible.

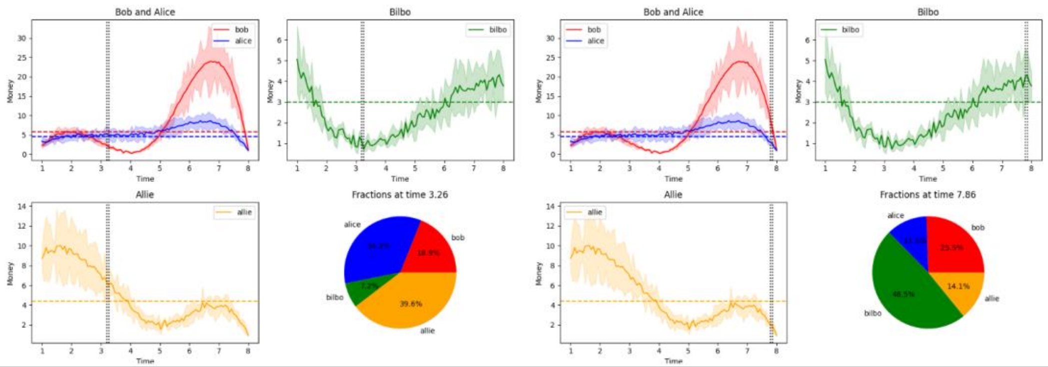
Here is a possible code to generate a pyplot for the following data, with some comments to explain the steps:

Plotting

Prompt: Can you generate a pyplot for the following data: $x = [1, 3, 5, 6, 8]$, $y1 = [2, 3, 5, 18, 1]$, $y2 = [3, 5, 6, 8, 1]$, $y3 = [5, 1, 2, 3, 4]$, $y4 = [9, 7, 2, 3, 1]$. I want $y1, y2$ to be in the same plot, but $y3$ is in another plot next to that plot, $y4$ is in below. I want the legend of $y1$ to be "bob", $y2$ to be "alice", $y3$ to be "bilbo", $y4$ to be "allie". I want the x-axis to be labeled with "time" and y axis to be labeled with "money". I want to add a 10 %-40% random error bar to each curve, through all times (including non-integers). I want smoothed curves to show the plot, and smoothed error bar. Do not use linear interpolation, use smooth interpolation! I want to also add some small zig-zag to the smoothed curve to make it look more real. I want to put a baseline as the mean of each line. I want to put a pie chart below indicating the fraction of the four people in each time step. I also want an animation to show how the fractions are changing in the pie chart in **continuous time**. Interpolate the missing fractions! I also want vertical line animation in other three plots to match the pie chart. I want the fanciest plot. Please add as many fancy things as possible.

Here is a possible code to generate a pyplot for the following data, with some comments to explain the steps:

GPT-4: [We give two snapshots from the animation resulting from the produced code]



Explanations

GPT-4:

```
This is a conversation between a user and an
artificial intelligence agent that follows
instructions or answers questions. The AI
always follows the instructions correctly.
---
User: Can you compose a short tune (say four
to eight bars) using ABC notation?
AI: There are many possible ways to compose
a short tune in abc notation, but here is
one example:
X:1
T:Example Tune
M:4/4
L:1/8
K:C
|:G2AB c2BA | G2AB c2c2 | edcB c2BA | G4-
G2z2 :|
|:c2de f2ed | c2de f2f2 | g2fe d2cB | c4-
c2z2 :|
This tune has two four-bar phrases, each
repeated once, forming an A part and a B
part. The key is C major.
```

GPT-4:

```
User: After the first bar G2AB c2BA, you
chose G to start the next bar. Give an
example of a change to the notes in the
first bar that would have caused you to
pick D instead of G as the first note in the
second bar.
AI:One possible change to the notes in the
first bar that would have caused me to pick
D instead of G as the first note in the sec
ond bar is:

|:G2AB c2Bc | ...

This change creates a stronger cadence on
the C note at the end of the first bar,
which is the fifth degree of the scale. This
suggests a modulation to the dominant key of
G major, which has D as its fifth degree.
Therefore, starting the second bar with D
would create a smooth transition to the new
key and imply a harmonic progression of V-I.
```

Understanding Programs

Prompt:

```c
struct mystruct_A { char a; int b; char c; } x;
struct mystruct_B { int b; char a; char c; } y;

#include <stdio.h>
int main(void) {
    printf("Size of x: %zu bytes\n", sizeof(x));
    printf("Size of y: %zu bytes\n", sizeof(y));
    return 0;
}

Possible Output:
```

GPT-4:

```
There are different ways to implement and align structures in memory,
so the exact output may vary depending on the compiler, platform, and
options used. However, one possible output is:

Size of x: 12 bytes
Size of y: 8 bytes

This output assumes that the compiler uses 4-byte alignment for
structures, meaning that each structure member starts at an
offset that is a multiple of 4 bytes from the beginning of the
structure. This may result in some padding bytes being inserted
between or after the members to maintain alignment. For example,
in x, there may be 3 padding bytes after a and 3 padding bytes after c,
while in y, there may be no padding at all. The size of a structure is
the sum of the sizes of its members plus any padding bytes.
```

(SparksOfAGI)

Explosion of LLMs … but mostly limited to English

ANTHROP\C

OpenAI

Gemini

MISTRAL AI_

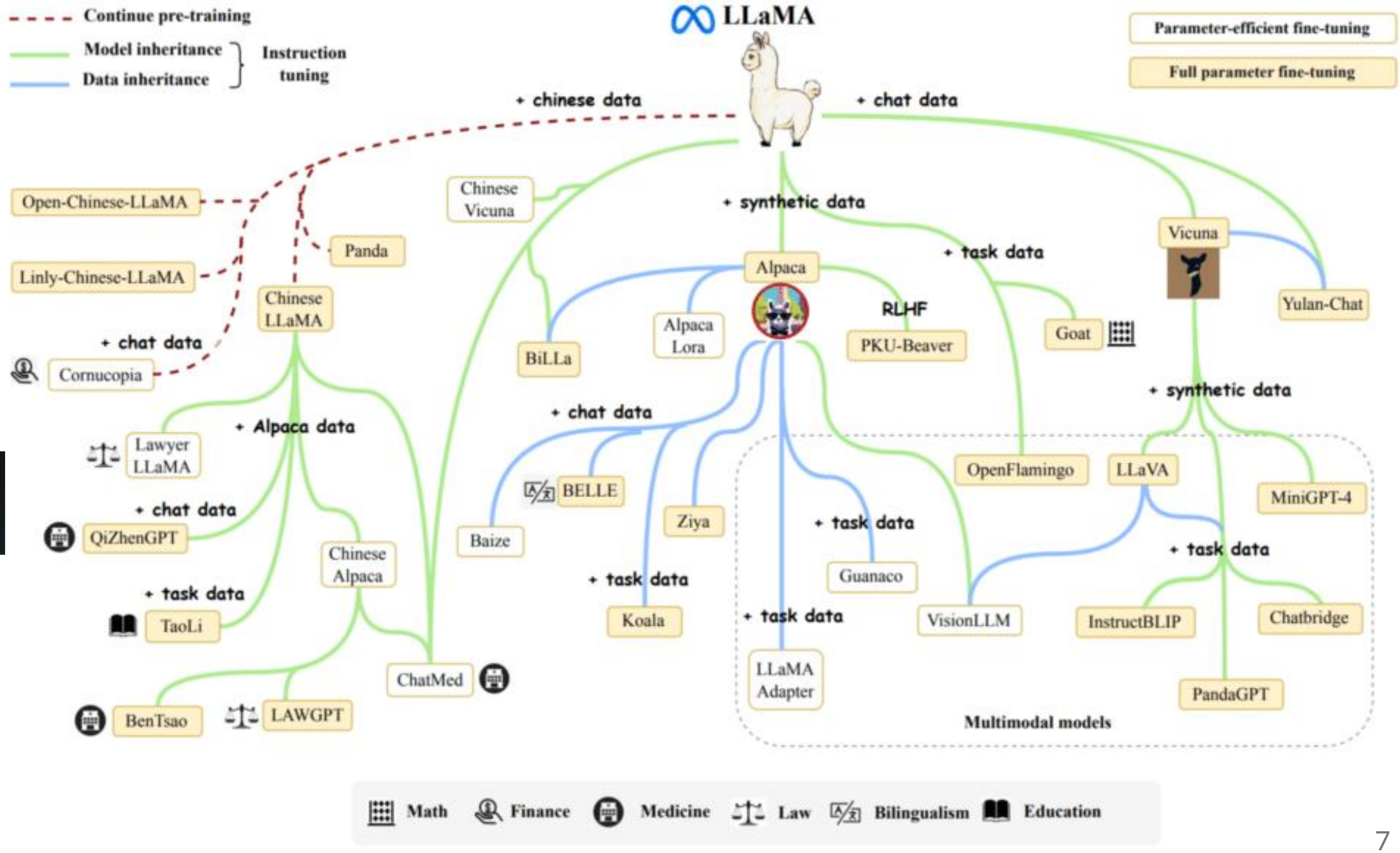
cohere

LLaMA

gpt-neo

Gemma

Phi - 3



Benefits of LLMs are mostly limited to English

Results on XNLI

Language	Cat.	ChatGPT (en)	ChatGPT (spc)
English	H	70.2	70.2
Russian	H	60.8	45.4
German	H	64.5	51.1
Chinese	H	58.2	35.5
French	H	64.8	42.2
Spanish	H	65.8	47.4
Vietnamese	H	55.4	44.8
Turkish	M	57.1	37.1
Arabic	M	55.3	22.3
Greek	M	55.9	54.5
Thai	M	44.7	11.5
Bulgarian	M	59.7	44.6
Hindi	M	48.8	5.6
Urdu	L	43.7	6.3
Swahili	X	50.3	40.8

Language	Cat.	ChatGPT(en) EM	ChatGPT(en) F1
English	H	56.0	74.9
Russian	H	30.2	49.1
German	H	45.9	65.8
Chinese	H	37.1	42.3
Spanish	H	41.8	65.8
Vietnamese	H	36.1	57.3
Turkish	M	34.5	56.4
Arabic	M	32.0	50.3
Greek	M	29.7	45.0
Thai	M	31.2	43.4
Hindi	M	17.5	37.8
Average		35.6	53.5

Results on Xquad QnA

	#langs.	avg. chrF	avg. BLEU
ChatGPT (0-shot)	203	32.3	16.7
ChatGPT (5-shot)	203	33.1	17.3
GPT-4	20	44.6	24.6
NLLB	201	45.3	27.1
Google	115	**52.2**	**34.6**

Performance on translation averaged across languages

Lang.	ChatGPT BLEU	ChatGPT chrF	NLLB BLEU	NLLB chrF
srp_Cyrl	1.36	3.26	**43.4**	**59.7**
kon_Latn	0.94	8.50	**18.9**	**45.3**
tso_Latn	2.92	15.0	**26.7**	**50.0**
kac_Latn	0.04	2.95	**14.3**	**37.5**
nso_Latn	3.69	16.7	**26.5**	**50.8**
jpn_Jpan	**28.4**	**32.9**	20.1	27.9
nno_Latn	**37.1**	**58.7**	33.4	53.6
zho_Hans	**36.3**	**31.0**	26.6	22.8
zho_Hant	**26.0**	**24.4**	12.4	14.0
acm_Arab	**28.2**	**44.7**	11.8	31.9

Performance on translation High vs low resource

Results on X-CSQA

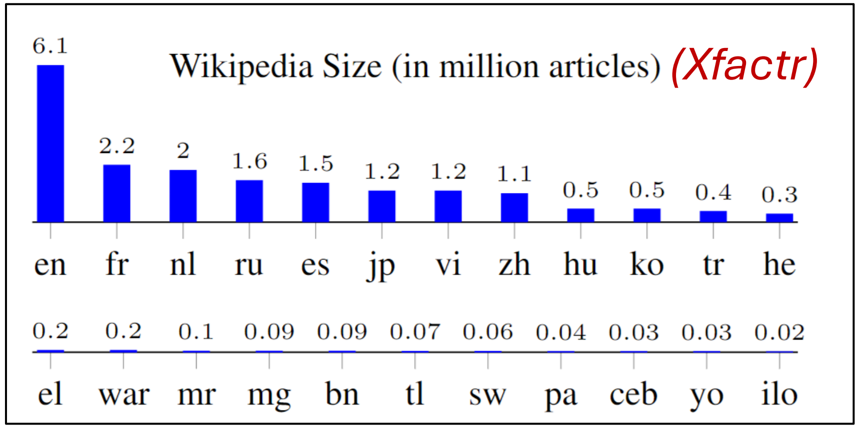
Language	Code	Cat.	ChatGPT (en)	ChatGPT (tgt)
English	en	H	75.0	75.0
Russian	ru	H	50.2	53.5
German	de	H	52.6	61.0
Chinese	zh	H	50.2	42.5
Japanese	jp	H	41.9	43.0
French	fr	H	50.5	61.7
Spanish	es	H	53.3	62.5
Italy	it	H	50.6	55.9
Dutch	nl	H	52.9	60.4
Polish	pl	H	35.2	51.1
Portugese	pt	H	49.5	59.2
Vietnamese	vi	H	42.3	47.9
Arabic	ar	M	49.4	47.3
Hindi	hi	M	41.1	38.6
Urdu	ur	L	34.7	24.5
Swahili	sw	X	35.6	46.6
Average			47.8	51.9

- Significant gap between English and other languages on multiple tasks
- High-resource and Latin script languages can give good performance on GPT
- Poor performance on low-resource languages
- Translate-test is a strong baseline
- Open-source models lag behind GPT models ➔ they are very English heavy

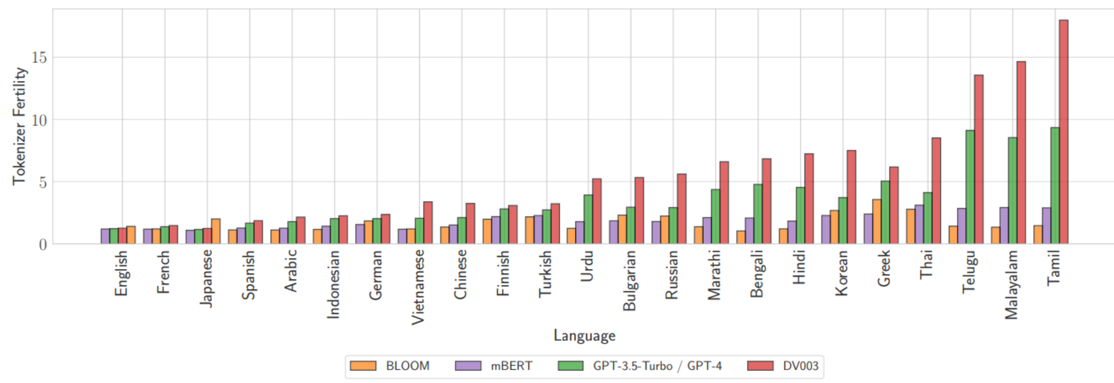
(BUFFET, MEGA, ChatGptMT,ChatGptMLing)

Why do LLMs lag behind for other languages?

- Lack of
  - Pre-training data
  - Token representation
  - Instruction tuning data
  - Human preference data
- Inability to transfer from English
- Limitations of Translate-Test



Most LLMs trained on <10% non-English data



Fertility → number of tokens per word
High fertility → low-efficiency, suboptimal representations

(BUFFET, MEGA, ChatGptMT)

Do English LLMs have some inherent multilingual capabilities?

Yes, to some extent …

Why? – during training they might have been exposed to some non-English data

- Documents with multiple languages
- Incorrect LID
- Increasingly some representation of non-English data e.g. Gemma2, LLama3

How good are the multilingual capabilities?

- Might be ok at language understanding *e.g. classification, sentiment analysis*
- Bad at generation
- Better on Latin script languages
- Languages with better pre-training representation perform better

How do English LLM achieve multilingual capabilities?

- *Do LLMs think in English?*
- *Do LLM use English as a pivot for decision making?*

Bottom layers: Feature learning

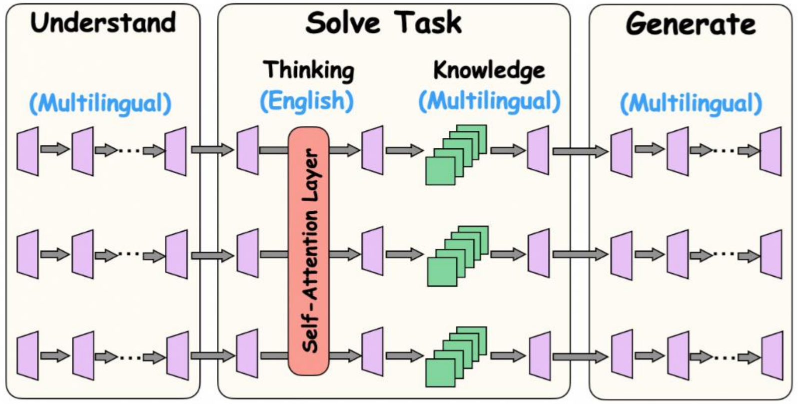
Middle layers: Concept mapping to language tokens (with English bias), task solving

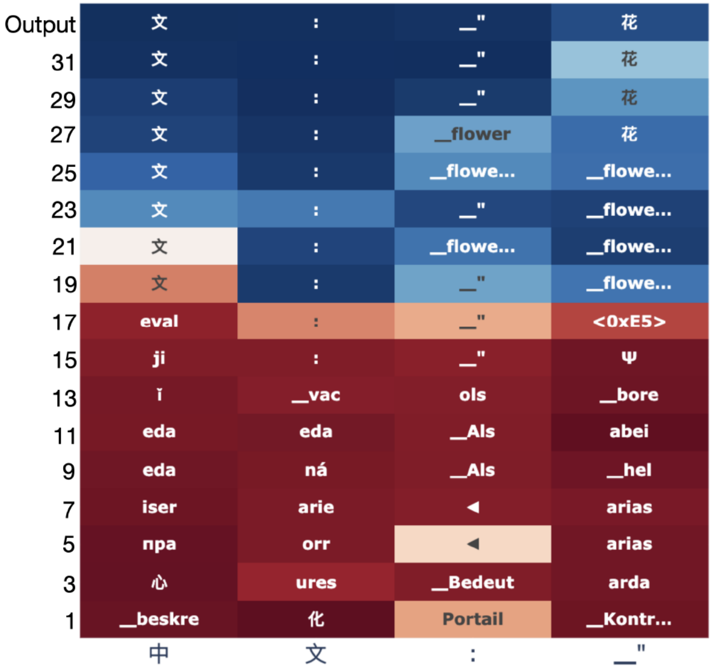
Top layers: Language generation in target language

There are language-specific neurons (mainly concentrated in the top and bottom layers)

The central question in building multilingual LLM is to bring representations of English and other languages closer to achieve good cross-lingual transfer

(LmaLatent,PNLD,LSP,SharingNeurons)





Open-source Multilingual LLM Efforts

Trained from scratch: *BLOOM, mGPT, PolyLM, EAGLE, mT0, XGLM, AYA*

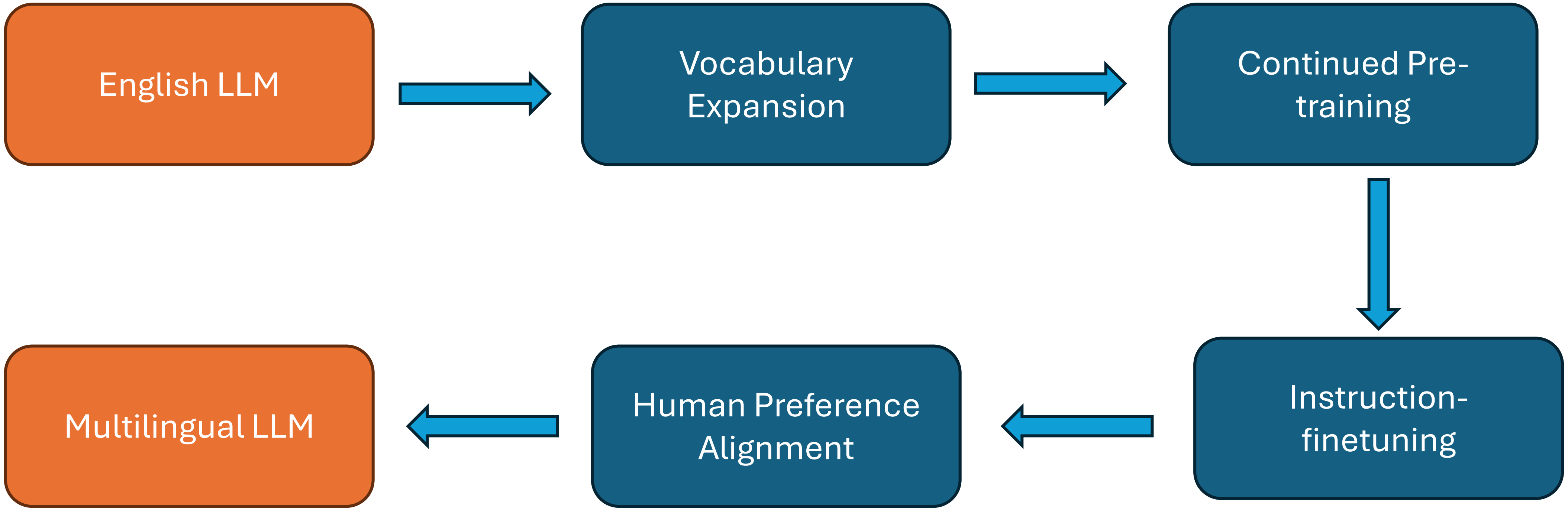
- English representation is lesser compared to models like Llama, Gemma, Mistral ➔ limited English capabilities
- Cannot expect good non-English capabilities either
- Large-scale compute needed for training

Focus of this survey

Extending English LLMs: *ChineseLLama, OpenHathi, SeaLLM, ALMA, RomanSetu*

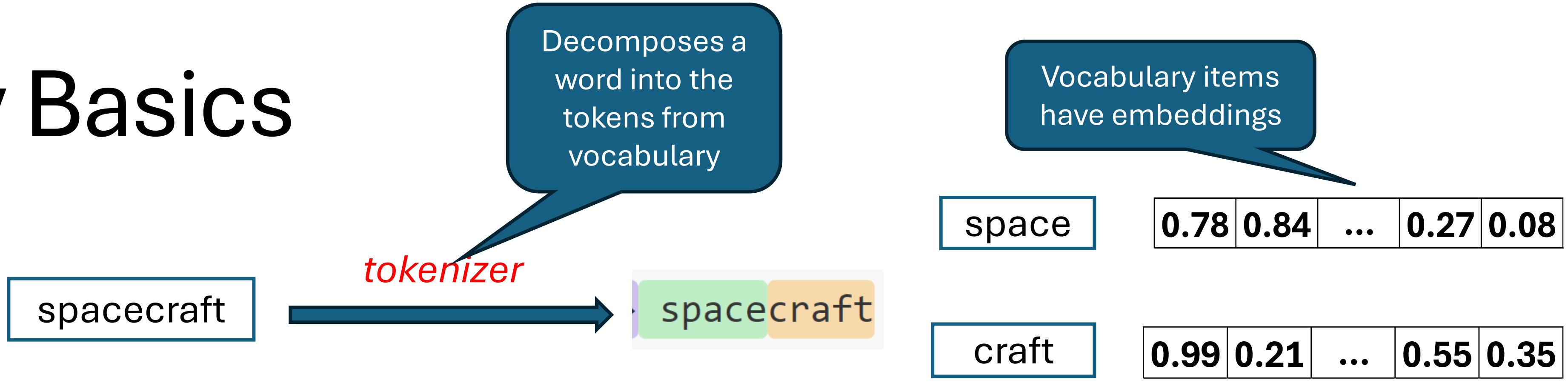
- Strong English capabilities of base LLMs
- Less compute-requirements
- English LLMs are at the cutting edge with regular updates

Extending English LLMs to Non-English Languages



Vocabulary Expansion

Vocabulary Basics



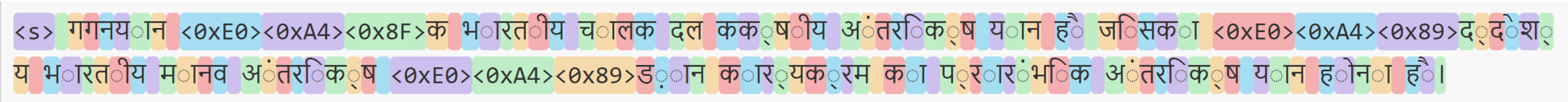
<s> Gaganyaan is an Indian crewed orbital spacecraft intended to be the formative spacecraft of the Indian Human Spaceflight Programme.

<u>Vocabulary</u>: Set of tokens (basic I/O units)

LLM Vocabulary Properties

- Finite vocabulary size
- Subword units: basic units are smaller than words
- Open vocabulary: all words can be defined as concatenation of subwords

What if vocabulary is under-represented?



Fertility = Average number of tokens per word

Unknown characters (BPE-based vocab)	UNK vocab item
Fallback to known characters (BPE-based vocab)	High Fertility
Fallback to bytes (Byte BPE-based vocab)	Even Higher Fertility

High fertility →

More memory consumption

More decoding time

Limit on longest processable sequence

Addressing Vocabulary issues

Status-quo (use suboptimal vocab)

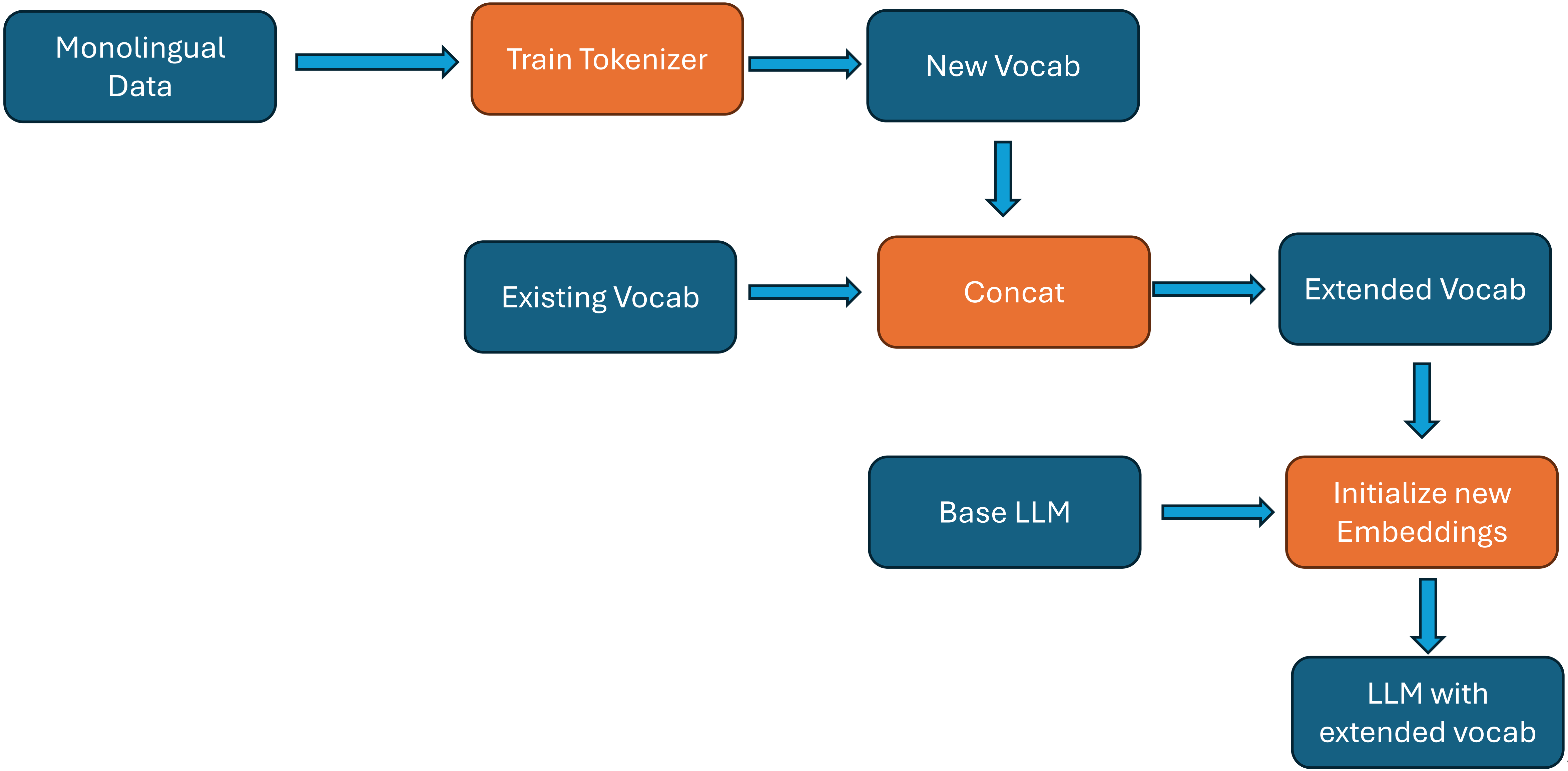
- High fertility
- Increased sequence length
  - Increased inference time
  - Limit on max sequence length
- Inferior token representation
- Lesser pre-training required

Extending Vocabulary

- Low Fertility
- Reasonable sequence length
  - Decreased inference time
  - Longer sequences possible
- Increased softmax computation
- More pre-training required

Some evidence seems to suggest that extending vocabulary needs a lot of pre-training to align languages (0.5B tokens vs 30B tokens) *(LmaByndEng)*

How to extend tokenizer vocabulary?



Initialization of New Embeddings

Sampling from Random (Normal) Distribution

Simple
Changes existing vocab's probability distribution
Large convergence time

Average of Existing Embeddings

Limited change in existing vocab's distribution
Large convergence time [AveInit]

Weighted Average of Existing Embeddings

Limited change in existing vocab's distribution
Initializations like WESCHEL, OFA, FOCUS, ConstrainedW2V

WESCHEL uses similarities between vocab items across languages to decide weights; this improves convergence rates

Hypernetworks for learning embeddings

Learn a hypernetwork that can predict embeddings for any tokenizer, enabling zero-shot tokenizer transfer

Average Initialization

[AveInit,ConstrainedW2V, ExpandChoices]

Limitations of initialization from (Normal) Random distribution

- Incorrect generation in existing language
  - Large KL-divergence between pre- and post-expansion LMs for existing vocabulary
- No reason for fast convergence

A simple solution: Initialize new tokens to average of embeddings of existing tokens

- Low KL-divergence between pre- and post-expansion LMs for existing vocabulary
- Greedy decoding with prefix of existing tokens will result in output from existing tokens
- **A general result**: the above applies if new embeddings are in the convex hull of existing embeddings

A practical solution: We want to avoid all new embeddings been initialized to same value

- Add small random noise to the average embeddings

Initial drop in task performance on CPT, but performance recovers with increase in training data

Strong baseline

However, this method does to give any solution to improve convergence in continued pre-training

Weighted Average Initialization [WESCHEL]

- Target token embeddings as weighted average of source token embeddings
- Token weights based on source-target token similarities based on external static pre-trained word embeddings

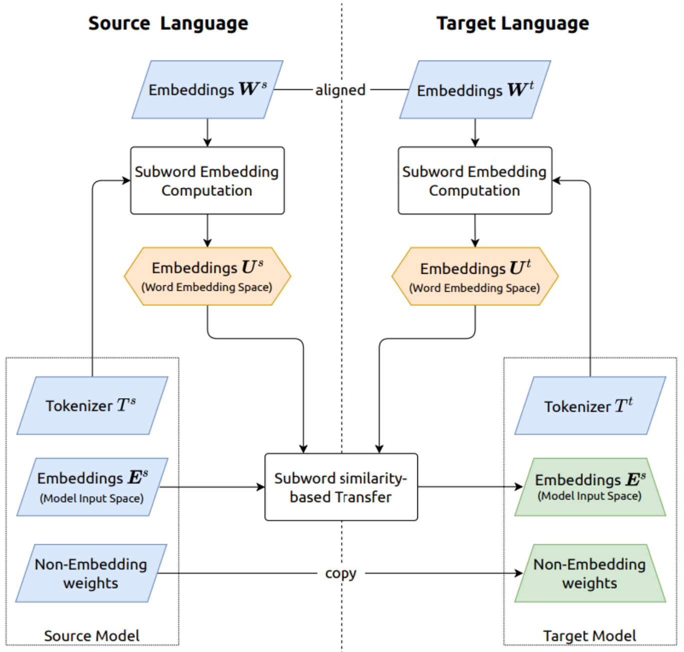
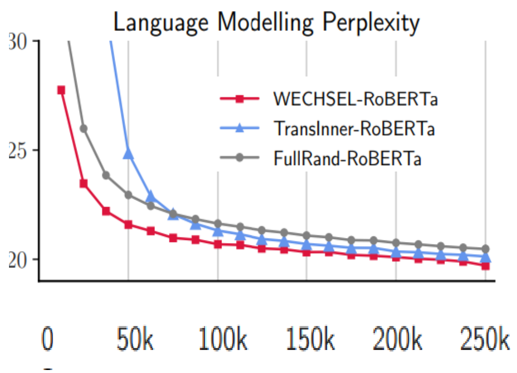


Figure 1: Summary of our WECHSEL method. We show inputs, intermediate results and outputs.

$$e_x^t = \frac{\sum_{y \in \mathcal{J}_x} \exp\left(s_{x,y}/\tau\right) \cdot e_y^s}{\sum_{y' \in \mathcal{J}_x} \exp\left(s_{x,y'}/\tau\right)}$$



Model	Score@0			Score@25k			Score@250k		
	NLI	NER	Avg	NLI	NER	Avg	NLI	NER	Avg
WECHSEL-RoBERTa	78.25	86.93	82.59	81.63	90.26	85.95	**82.43**	**90.88**	**86.65**
TransInner-RoBERTa	60.86	69.57	65.21	65.49	83.82	74.66	81.75	90.34	86.04
FullRand-RoBERTa	55.71	70.79	63.25	69.02	84.24	76.63	75.28	89.30	82.29
XLM-$\mathrm{R}_{\mathrm{Base}}$ *(Final)*	79.25	89.48	84.37						

Continued Pre-training

Faster convergence vs. baselines for

- LM perplexity
- Downstream performance

Results for small LMs → embeddings contribute a large % of parameters

Will we see such convergence improvements for Large LMs?

More Methods and Findings

Extensions of WESCHEL

OFA (One-for-All): *multilingual vocabulary, need to handle large vocab (OFA)*

- Reduce embedding dimension (inspired from ALBERT)
- Source embedding factorization with SVD for dimensionality reduction
  - Co-ordinates: language-dependent
  - Primitives: language-independent
- Projection of source co-ordinates to target co-ordinates like WESCHEL

FOCUS: *Target token embeddings as weighted average of* ***overlapping*** *source token embeddings (FOCUS)*

Constrained Word2Vec [ConstrainedW2V]

A simple approach to learn embeddings for new tokens in the convex hull of existing tokens

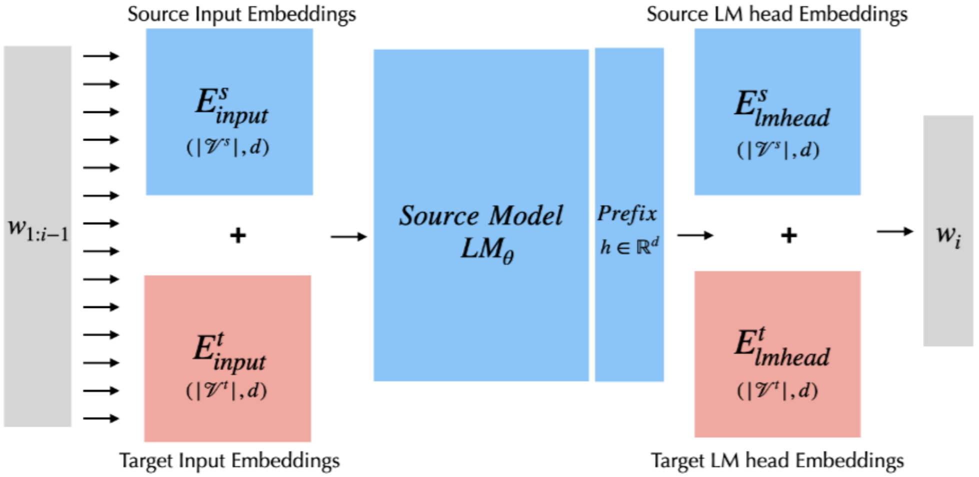
Formulate learning new token embeddings as a Word2Vec problem with the following constrains:

- Embeddings of existing tokens are not updated during word2vec training
- Embeddings of new tokens are strictly expressed as convex combination of existing tokens (so just averaging weights are learnt)

To ensure cross-lingual mapping of word embeddings, bilingual dictionaries are used in word2vec training

- Dictionary entries (w_s, w_t) are simply serialized as a sentence "$w_s\ w_t$" for word2vec training

New token embeddings are learnt based on context as well as similarity to existing embeddings



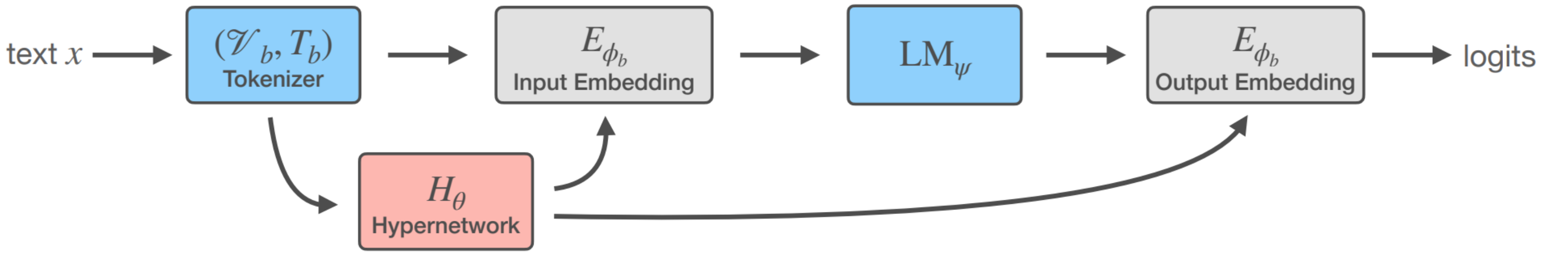
	LLaMA2							
	MT		XNLI		QA		XLSUM	
	En-X	X-En	en	avg	en	avg	en	avg
CW2V	**17.0**	**27.3**	60.4	**38.1**	**77.7**	**35.8**	**0.6**	**0.4**
OFA	11.2	16.2	60.4	37.1	76.0	26.0	0.6	0.3
Multivariate	11.1	16.1	60.4	37.2	77.5	28.7	0.5	0.2
Univariate	11.1	16.0	60.4	37.2	77.4	28.7	0.5	0.3
Mean	11.1	16.2	**60.5**	37.2	77.4	28.7	0.5	0.3
Random	0.0	0.0	33.3	33.3	0.0	0.0	0.0	0.0

CW2V is competitive or better than other sophisticated initialization approaches

Zero-shot Tokenizer Transfer *(ZSTT)*

Can we learn a function that can predict the embedding for any given tokenizer for a fixed language model?

Learn this function once, and then use it to predict embeddings for any new tokenizer



Tokenizer data for training is synthetically generated by considering all possible frequent tokenizations of a string

The Hypernetwork generates the target embeddings for the new tokenizer

Gold Target embeddings are not explicitly defined, but are ones which minimize the language modeling loss of the LM under considering

End-to-end training: learn embeddings which actually improve language modeling

ZSTT performs better than other approaches on the XNLI task and other tasks as well

	ar	**bg**	**de**	**el**	**en**	**es**	**fr**	**hi**	**ru**	**sw**	**tr**	**ur**	**vi**	**Avg.**
original	68.9	75.6	74.7	73.7	82.3	76.9	76.8	68.4	72.9	63.5	72.2	64.7	73.1	72.6
Lexical	58.7	63.1	65.3	61.7	72.8	68.4	66.7	61.8	62.3	51.8	58.5	60.0	72.0	63.3
FVT	63.9	70.3	70.9	67.4	79.0	73.9	71.9	65.7	67.8	57.1	66.3	61.7	72.9	68.4
OFA	57.3	64.2	67.3	62.8	73.6	68.6	68.4	61.8	63.1	54.8	59.7	59.3	72.3	64.1
FOCUS	64.8	71.0	71.6	67.7	79.6	74.4	72.6	64.5	68.1	55.7	67.3	61.9	72.6	68.6
ours	**67.9**	**73.9**	**74.1**	**71.4**	**81.1**	**76.2**	**74.7**	**67.7**	**70.7**	**62.3**	**68.7**	**63.2**	**73.9**	**71.2**
Δaccuracy	-1%	-2%	-1%	-2%	-1%	-1%	-2%	-1%	-2%	-1%	-3%	-2%	+1%	-1%
Δlength	-22%	-14%	-13%	-23%	-9%	-11%	-12%	-13%	-13%	-19%	-15%	-9%	-3%	-14%

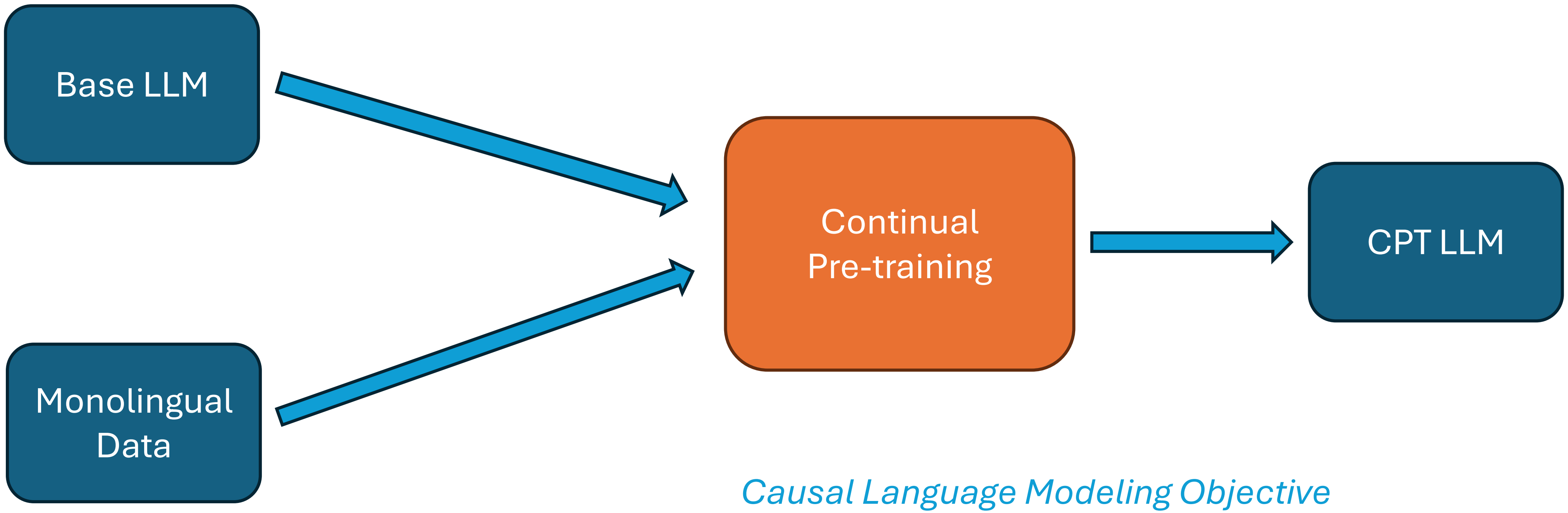
Summary & Recommendations

- **Vocab expansion reduces fertility and improves efficiency**
- **Is vocabulary expansion better than relying to initial sub-optimal vocab?**
  - Initial drop in results for vocab expansion before recovery
  - Vocab expansion might require lot of pre-training for alignment

- **Can we do better than random initialization?**
  - Embeddings which initialize new tokens based on similarity with older embeddings do better
  - Simple methods like averaging, constrained W2V are sufficient
  - Faster convergence
  - Slightly better downstream performance
  - Results mostly for smaller LMs and decoder LMs

- **Will vocabulary extension lead to lower performance on English?**
  - If initialized embeddings are in convex hull, greedy decoding results does not change

Continual Pre-training



Causal Language Modeling Objective

$$p(\mathbf{x}) = p(x_1, x_2, ..., x_T) = \prod_{t=1}^{T} p(x_t|\mathbf{x}_{<t})$$

Train on document-level data

Finetuning on long, coherent sequences helps model learn and correlate different pieces of knowledge

To avoid forgetting English competence and knowledge

- Include English in the pre-training data
- Finetune-only small number of adapter parameters
  (ChineseLLama, OpenHathi)

What properties do we want to see in multilingual corpora?

- Large-scale, Document-level Datasets
- High Quality Documents
- Wide coverage of topics
- Representation of culture-specific data, native literature
- Capture data in different modalities and genres
- Data to Help Cross-lingual transfer with English

✓ Large-scale, Document-level Datasets

? High Quality Documents

? Wide coverage of topics

Multilingual corpora like mC4, CC-100, CulturaX are good starting points

✗ Representation of culture-specific data, native literature

✗ Capture data in different modalities and genres

✗ Data to Help Cross-lingual transfer with English

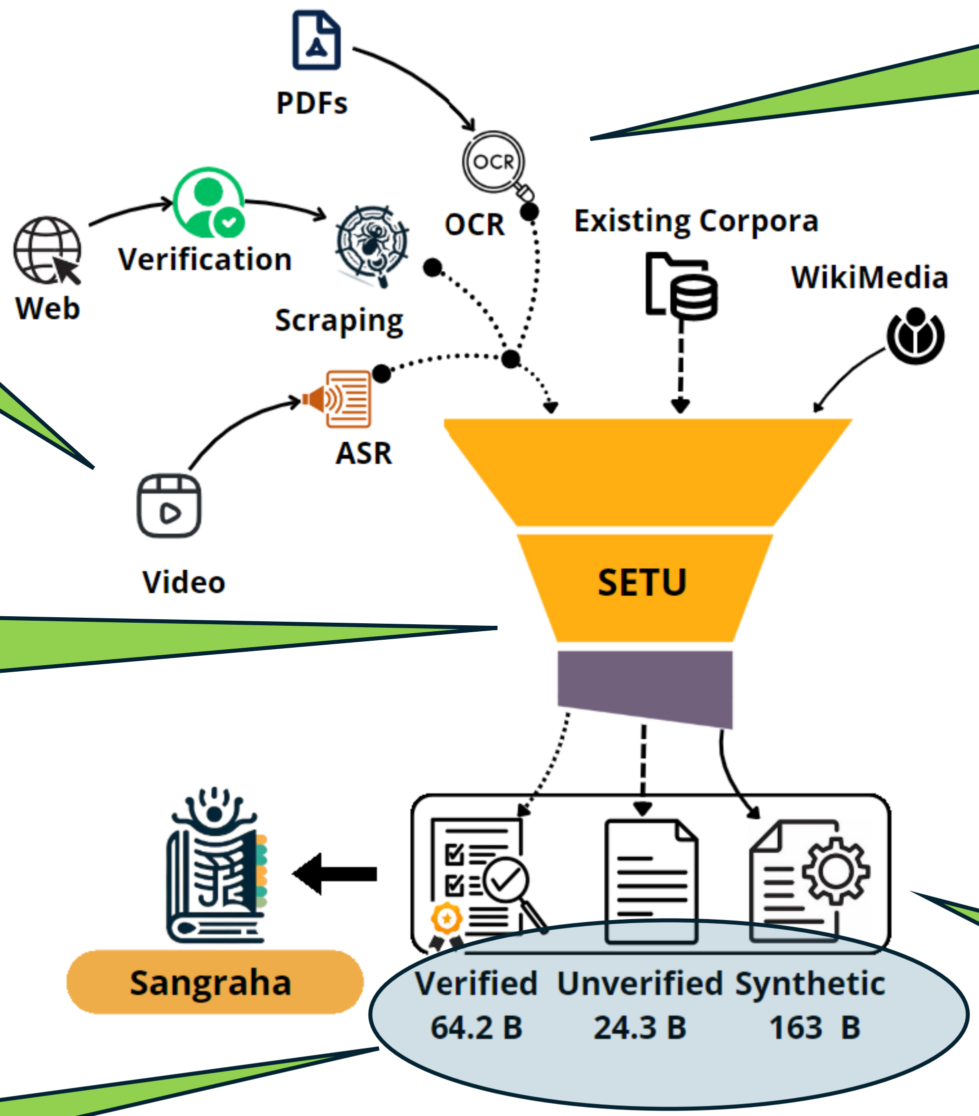
Build custom language (group) specific collections to address gaps

SANGRAHA

Creating high-quality pre-training data at scale

Data from different modalities – audio/OCR

OCR to unlock diverse regional and cultural content in PDFs



Filtering pipeline to ensure high-quality

High –quality sources

Translations and Transliterations to enable cross-lingual transfer

(IndicLLMSuite)

Why do continual pre-training?

Language competence/fluency in target language

	L(0)	L(10k)	L(100k)	L(1M)
Chinese	10.151	8.697	6.634	5.249

Perplexity reduces with increase in pre-training corpus size

(LmaByndEng)

Improve alignment b/w English and target language

Language	Base LLM	After CPT
Gujarati	0.39	**0.46**
Hindi	0.40	**0.44**
Marathi	0.44	**0.48**

Cosine similarities between English and target languages increases with CPT

(RomanSetu)

Provide required knowledge in target language for better understanding

- LMs better at using in-language knowledge *vs.* cross-lingual transfer (Xfactr,MLAMA)
- Incorporate cultural-specific knowledge capture in native language corpora only

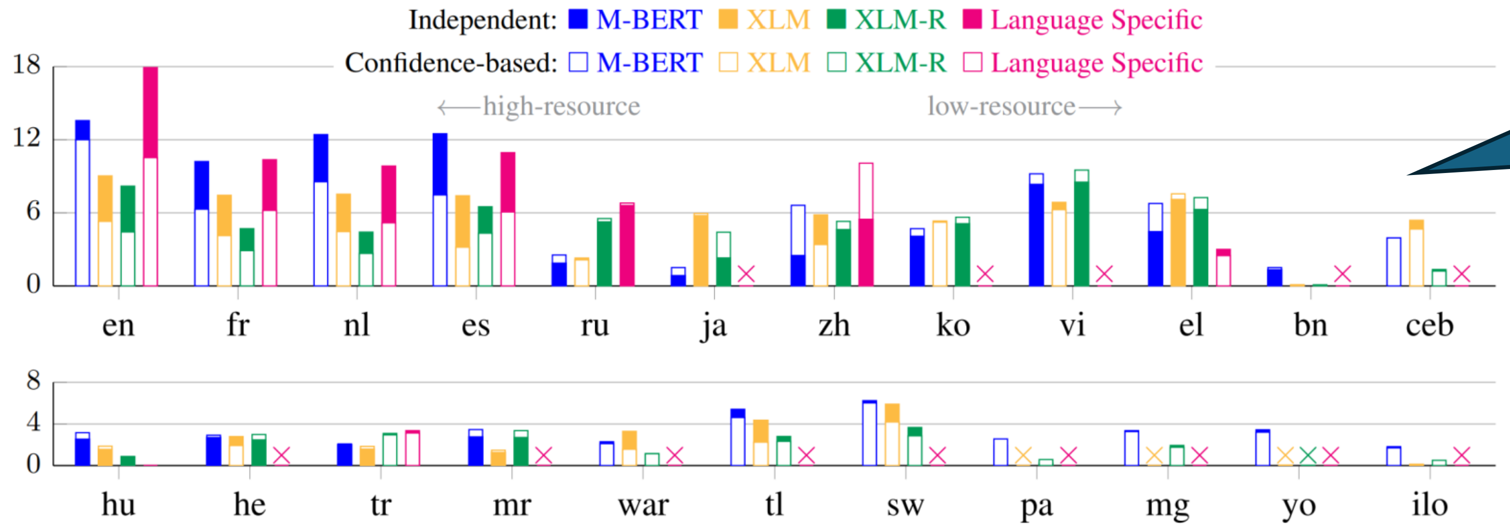
Most multilingual models can't transfer knowledge in English to other languages

Knowledge Probing Task ➔ Predict missing tokens which capture model's knowledge

fact	⟨Bloomberg L.P., founded_in, New York⟩
en prompt	[X] was founded in [Y].

es sentence: Bloomberg L.P. fue fundada en ⟨mask⟩ $\times 1 \sim 5$.

	prediction	#tokens	confidence
es outputs	2012	1	-1.90
	Nueva York	2	-0.61
	EE. UU	3	-1.82
	Chicago, Estados Unidos	4	-3.58
	2012 Bloomberg L.P	5	-3.06



English accuracy for knowledge probing is much higher than other langugaes

Results on Knowledge Probing task shows that non-English languages don't have enough data

(Xfactr)

Improving Cross-lingual Transfer in Pre-training

- Using Parallel/Translated Data
- Using Romanized Representation

Why?

- Help improve cross-lingual alignment
- Make knowledge available in English in the target languages
- Help translation task

Using Parallel/Translated Data

Using parallel data *(Tower, Palm2, PolyLM, OpenHathi, MTDataPretrain)*

- Train on document/paragraph pairs ➔ very little availability
- Train on sentence pairs ➔ modest availability depending on language pair
- MT Data modestly useful for NLU (results on encoder LMs) *(PrimerPMLM)*
  - More investigation needed

Using Machine Translated data *(IndicMonoDoc)*

Use off-the-shelf MT data to generate target language data at scale ➔ needs a decent MT model

- Model training includes translated documents
- Some evidence to show that translated documents can achieve performance close to pre-training with native language documents

Need better to understand impact of translation quality

Using Parallel/Translated Data (1)

Using human-written parallel data *(Tower, Palm2, PolyLM, OpenHathi, MTDataPretrain)*

- Train on document/paragraph pairs ➔ very little availability
- Train on sentence pairs ➔ modest availability depending on language pair

Useful for translation task *(Tower, OpenHathi, InciBiling)*

No systematic results on utility of parallel data in pre-training

Previous work

- Encoder-only models & NLU tasks ➔ parallel data has limited utility *(PrimerPMLM)*
- Encoder-decoder models & NLG tasks ➔ don't know

Using Parallel/Translated Data (2)

Using Machine Translated data (IndicMonoDoc)

Use off-the-shelf MT data to generate target language data at scale

➔ needs a decent MT model

- Model training includes machine translated documents
- Pre-training on translated documents slightly inferior to original documents
  - Translation quality filtering + using small original data makes result comparable
- For small LMs, synthetic data might outperform original data

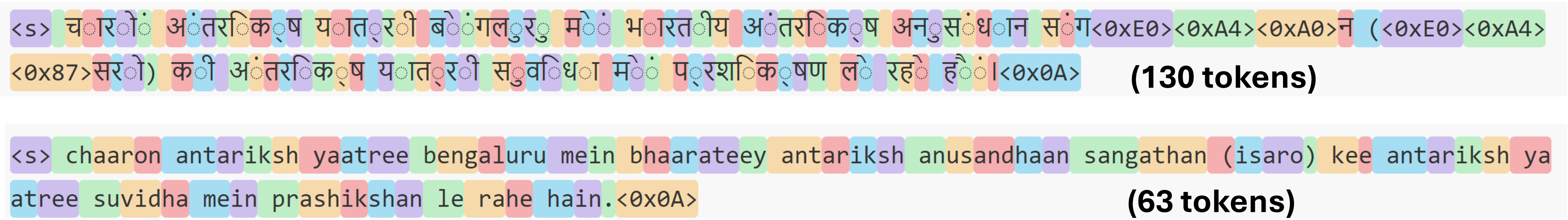
(a) Results on Hindi

Model	NLU						NLG				
	iXNLI	bbc-a	iitp-mr	iitp-pr	midas	Avg.	Headline Gen.	Sentence Summ.	Question Gen.	Wikibio	Avg.
HI-clean	73.61	81.75	72.58	79.73	80.34	77.60	27.54	23.64	24.84	52.16	32.04
syn-HI_en-unfiltered	72.87	77.92	64.36	76.22	79.91	74.26	**27.29**	22.93	24.22	**50.14**	**31.14**
syn-HI_en-unfiltered+10%	74.63	78.36	67.75	77.46	80.17	75.67	-	-	-	-	-
syn-HI_en-filtered	**74.75**	**81.06**	69.03	78.58	79.73	76.63	27.15	**23.10**	**24.41**	49.88	31.13
syn-HI_en-filtered+10%	74.49	80.94	**71.61**	**79.92**	**80.64**	**77.52**	-	-	-	-	-

Romanized Representation *(RomanSetu)*

Challenges with non-Latin script languages

- High-fertility/data loss for under-represented vocab
- Poor representation quality
- Vocab extension requires lot of pre-training *(Lai et al . 2023)*



Pre-train on romanized corpora

- Natural transliteration
- Fixed Romanization schemes

Language	N	R
Gujarati	18.44	**3.39**
Hindi	7.36	**2.98**
Malayalam	12.85	**5.04**
Marathi	8.91	**3.64**
Tamil	12.11	**4.89**

Romanized fertility more than 2x lower than native script fertility

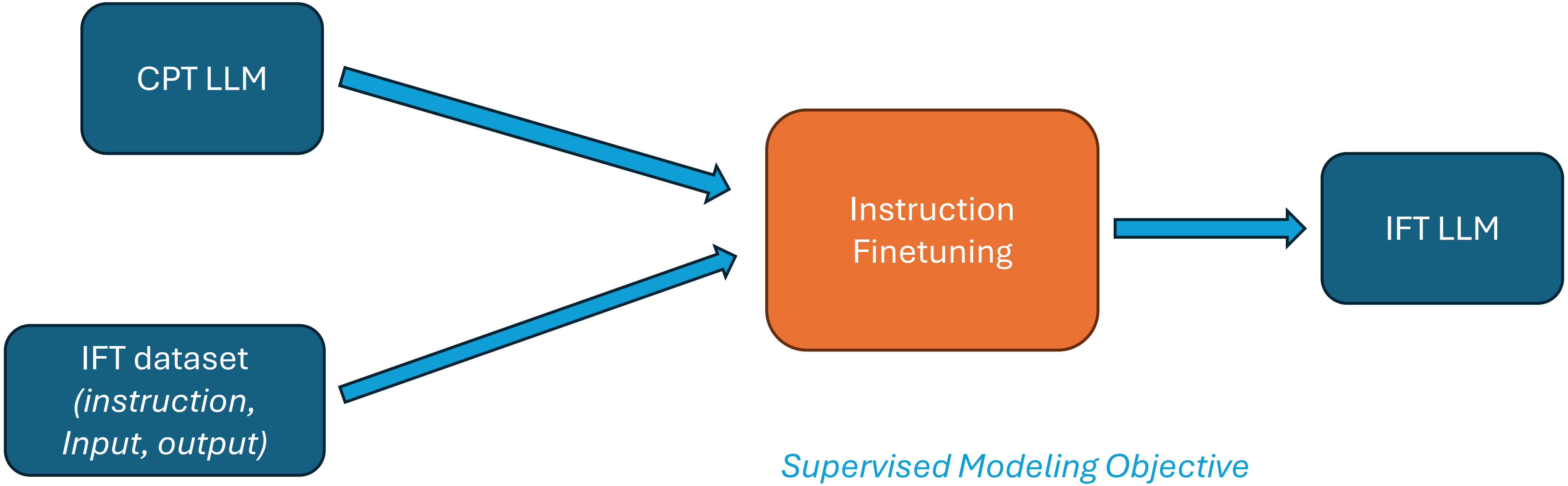
Language	E - N	E - R
Gujarati	0.39	0.47
Hindi	0.40	0.50
Malayalam	0.40	0.46
Marathi	0.44	0.48
Tamil	0.44	0.43

Romanized representations are better aligned to English than native script representations

Summary and Recommendations

- Amount of data used for CPT
  - Modest amount of data (hundreds of millions of tokens) for language competency and modest cross-lingual transfer
  - Lot of translated data required for knowledge transfer? – Avenue for research
- Does parallel data improve cross-lingual transfer?
  - Improves translation quality
  - Improving cross-lingual transfer, use of translated data requires further research
- Data augmentation methods like romanization, code-switching are helpful
- Drop in English task performance
  - Mitigation: Significant ratio of English, use adapters for CPT
  - Is retaining English performance critical to cross-lingual transfer?

Instruction Tuning



Supervised Modeling Objective

$$\ell_{\mathrm{CE}}(\mathbf{y},\hat{\mathbf{y}}) = -\sum_{j=1}^{|\mathcal{V}|} y_j \log(\hat{y}_j) \qquad \mathcal{L}_{\mathrm{SFT}} = \frac{1}{N}\sum_{i=1}^{N} \ell_{\mathrm{CE}}(\mathbf{y}_i, \mathcal{M}_\theta(\mathbf{x}_i))$$

Train on in-language IFT dataset

Sources of IFT dataset
Quality and diversity of IFT dataset

To retain English task performance

- Include English in the IFT training

Instruction Tuning Tasks

Variety of tasks/objectives to improve non-English performance

Generating IFT Data

- English Data IFT
- In-language IFT with Machine Translated Data
- Locally/Culturally relevant IFT data

Auxiliary Tasks

- Parallel Data
- Monolingual Data

Transforming IFT Datasets

- Romanized IFT Data
- Cross-lingual Thought Data
- Cross-lingual IFT Data
- Code-switched IFT Data

Let's look at these tasks in detail

Using English IFT Dataset

- Instruction tune the model on English instruction dataset
- Evaluate on non-English data ➔ Zero-shot cross-lingual evaluation
- Instruction tuning on English important to retain English capabilities

Using Machine Translated IFT Dataset

- Translate English instruction tuning datasets into the language
- Fine-tune model on translated dataset

Task	BeleBele QA	MKQA	XL-Sum
	Accuracy	*F1*	*Rouge-L*
English IFT	45.58	36.48	8.42
Language IFT	**48.28**	**37.95**	**15.87**

Average performance across many languages; src: SDRRL

Instruction tuning on translated data outperforms English instruction-tuning

Creating Translated IFT Data

Choice of Translation Engine

- Off-the-shelf NMT systems (Airavat): higher quality, particularly for low-resource
- GPT (Okapi): can do translation taking the entire context of input/output
- Hybrid Approach (LlmByndEng): Do one of the above depending on language's translation quality

	#langs.	avg. chrF	avg. BLEU
ChatGPT (0-shot)	203	32.3	16.7
ChatGPT (5-shot)	203	33.1	17.3
GPT-4	20	44.6	24.6
NLLB	201	45.3	27.1
Google	115	**52.2**	**34.6**

Comparison of various translation engines
Sentence-level
(ChatGptMT)

Model	**Human (General/Discourse)**				
	News	Social	Fiction	Q&A	**Ave.**
Google	1.9/2.0	1.2/1.3	2.1/2.4	1.5/1.5	1.7/1.8
DeepL	2.2/2.2	1.3/1.1	2.4/2.6	1.6/1.5	1.9/1.9
Tencent	2.3/2.2	1.5/1.5	2.6/2.8	1.8/1.7	2.1/2.1
GPT-3.5	2.8/2.8	2.5/2.7	**2.8**/**2.9**	2.9/2.9	2.8/2.8
GPT-4	**3.3**/**3.4**	**2.9**/**2.9**	2.6/2.8	**3.1**/**3.2**	**3.0**/**3.1**

Comparison of various translation engines
Document-level
(ChatGptMT)

Creating Translated IFT Data (2)

What to Translate

- <u>Instruction, Input, Output</u> (Okapi, Airavat, xLLama, SDRRL)

- <u>Input, Output</u> (BLOOMZ)
  - English instruction is a common usecase
  - Models are good at English Instruction following

Quality Filtering

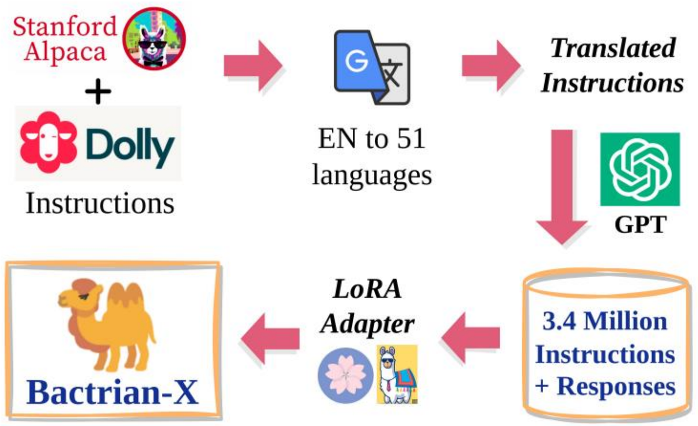
High quality examples are important for instruction tuning

- Use an MT evaluation metric like COMET-QE to identify bad translations
- Rule-based filters to avoid code examples, etc. that are difficult to translate

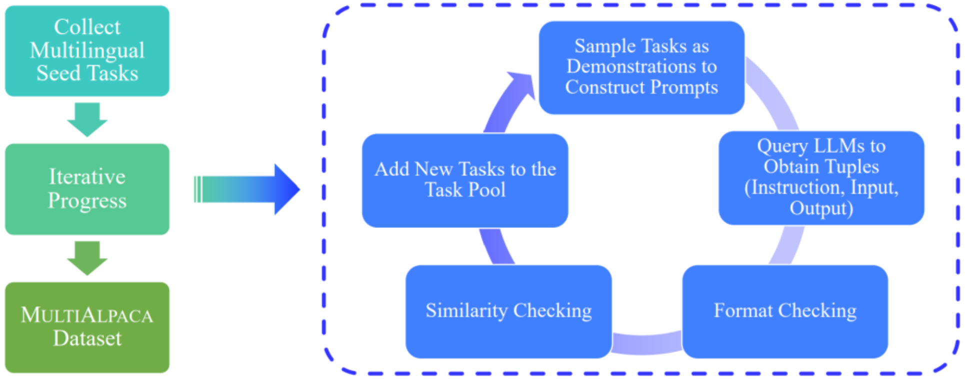
Creating Translated IFT Data (3)

What to Translate

- <u>Instruction, Input</u> (BactrianX)
  - Give translated Instruction & Input
  - Generate response using GPT in the target language
  - Language/culture-specific examples



- <u>Seed Instructions</u> (PolyLM,SeaLLM)
  - Generates the entire examples from strong LLM like GPT in target language
  - Language/culture specific examples, but quality/diversity might be issue

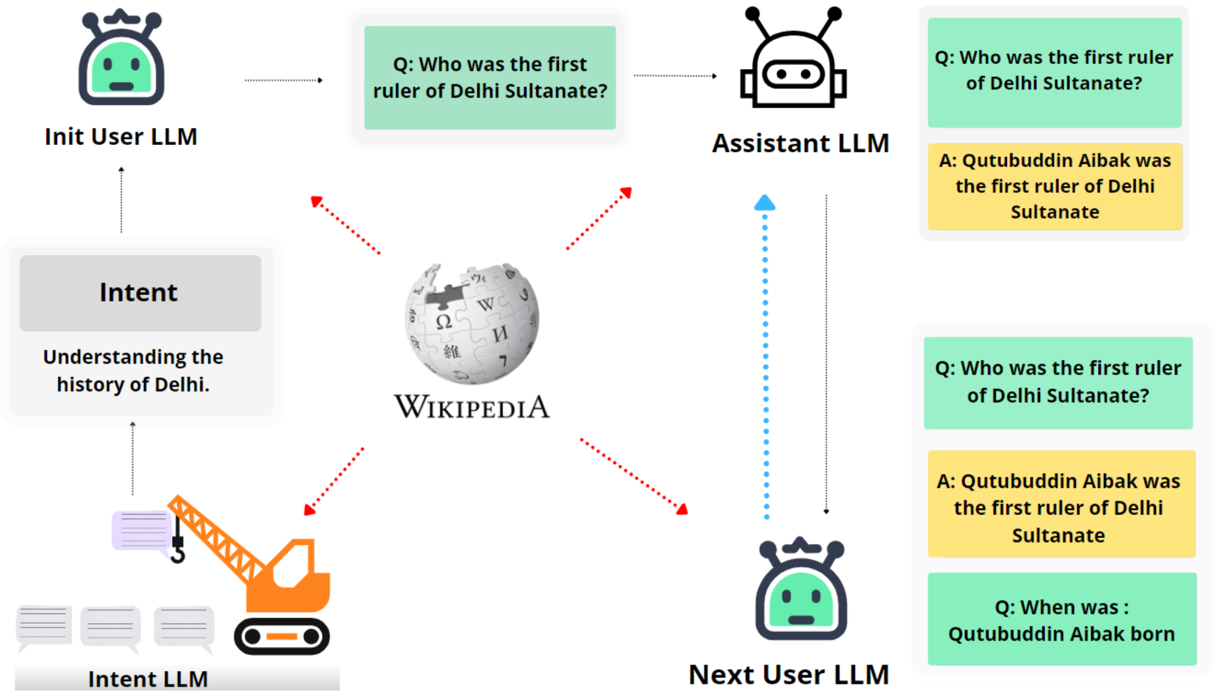


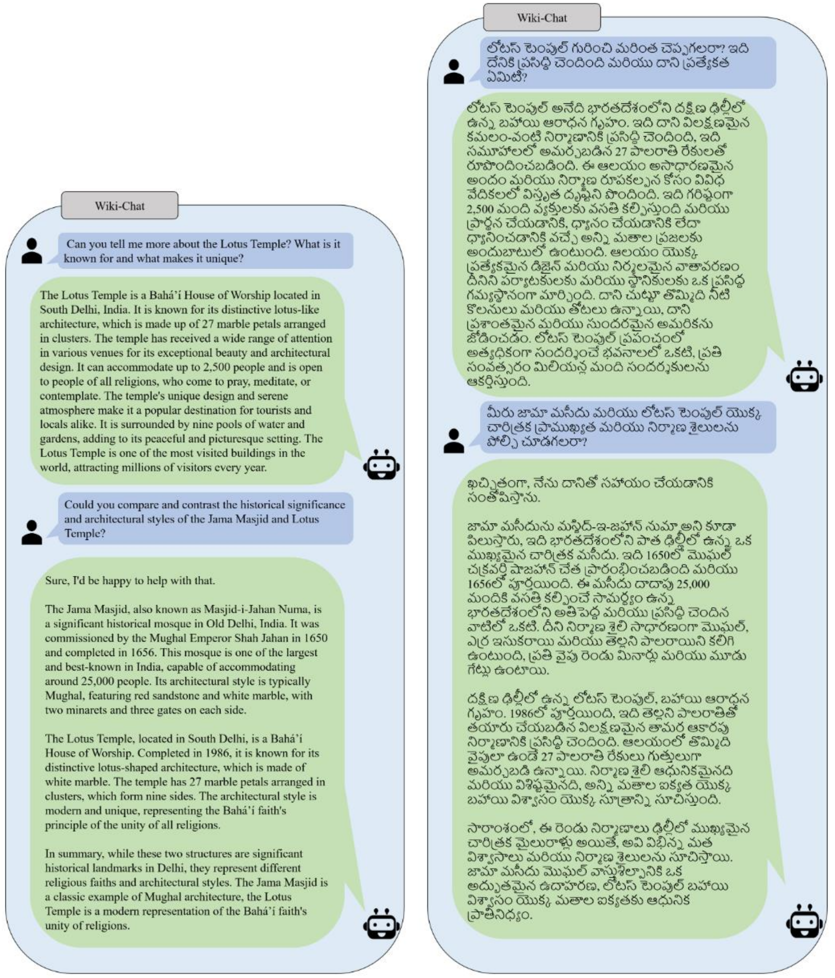
But we are just translating existing datasets – they might not be reflective of topics relevant to the language users

Creativity is the limit

Generating Culturally Relevant IFT Data

Use multiple English LLMs along with Wikipedia context to simulate conversations on topics of interest





(a) English (b) Telugu

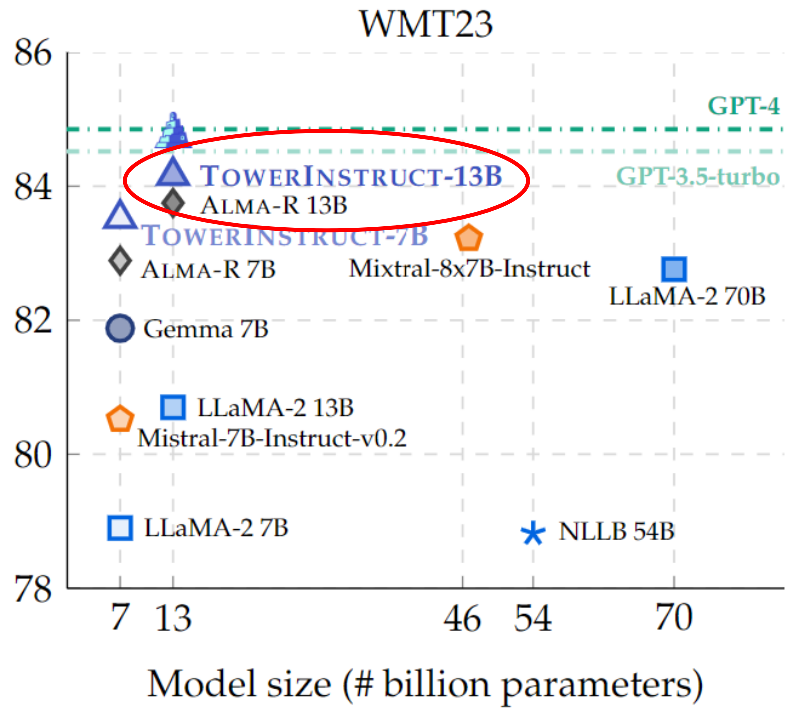
Translate the conversations into English

(IndicLLMSuite)

Using Parallel Data

Translation is a *special* task for multilingual IFT models

- Teaches the model to translate
- Aligns English and language representations better
- Improves performance on other downstream tasks
- Parallel data and translated IFT data both help



Results for translation task

Instruction Data	XQUAD (exact match)	MLQA (exact match)	mLAMA (exact match)	XLSum (Rouge-1)
Alpaca-En	31.8	26.7	5.3	9.0
Alpaca-En+En-Zh	34.3	38.0	5.8	27.1
Alpaca-En+Alpaca-Zh	51.7	48.0	21.9	25.5
Alpaca-En+Alpaca-Zh+En-Zh	54.9	51.8	30.4	28.3

Results on Chinese for various Tasks

Instruction-tuning Data	Ar	Hi	Vi	Zh
Alpaca-En	16.1	13.7	34.1	26.7
Alpaca-En+En-Zh	33.6	35.1	42.2	38.0
Alpaca-En+Alpaca-Zh	33.1	35.1	50.1	48.0
Alpaca-En+Alpaca-Zh+En-Zh	37.0	42.3	50.8	51.8

Results for other languages on MLQA

(xLlama,PLUG,TOWER,ALMA-R)

X-llama paper discussions: https://openreview.net/forum?id=CaP3CByuLp&noteId=reoX9bUPvQ

Using Monolingual Data

- "Translationese IFT Data" ➔ output language might not be fluent and high-quality
- Expose model to monolingual target language data during IFT
- Incorporate a task that helps model generate fluent output in target language

Task 1: Standard next-word prediction (CLM)

Switch between IFT and CLM objective in mini-batches

Task 2: Sentence Completion Task

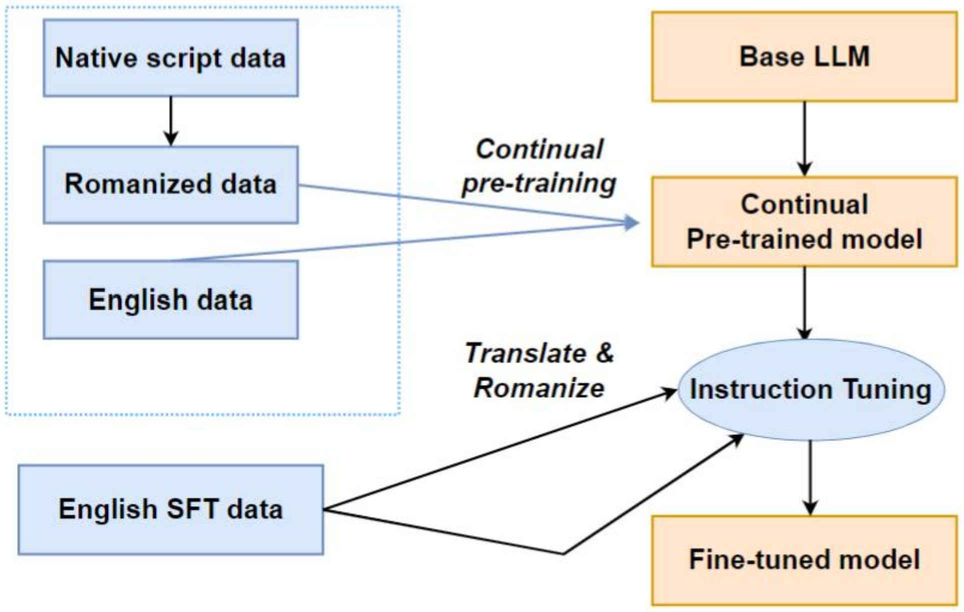
Only IFT objective required

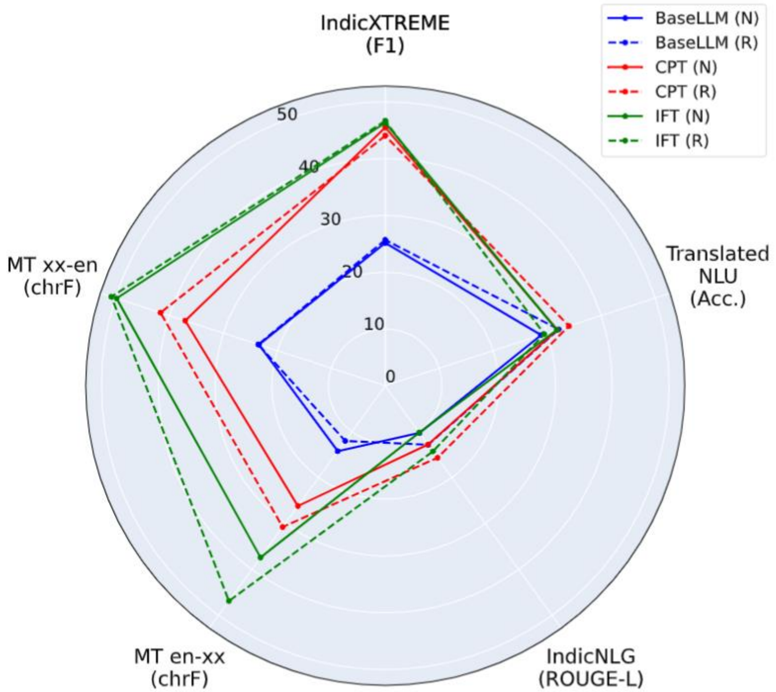
Question: Complete the following sentence in *Indonesian* according to its context.
Sang rubah cokelat cepat
Answer: *Sang rubah cokelat cepat melompati anjing malas.*

(SDRRL)

Romanized Representation

Just like pre-training, use romanized representation for IFT too





- *Continual Pre-training with romanized data is crucial*
- *NLG task performance improves with romanized data*
- *NLU task performance is on par, though more efficient*

(RomanSetu)

Cross-Lingual Thought Prompting (XLT)

Ask the model to implicitly think in a different language

Model asked to generate intermediate English artifacts

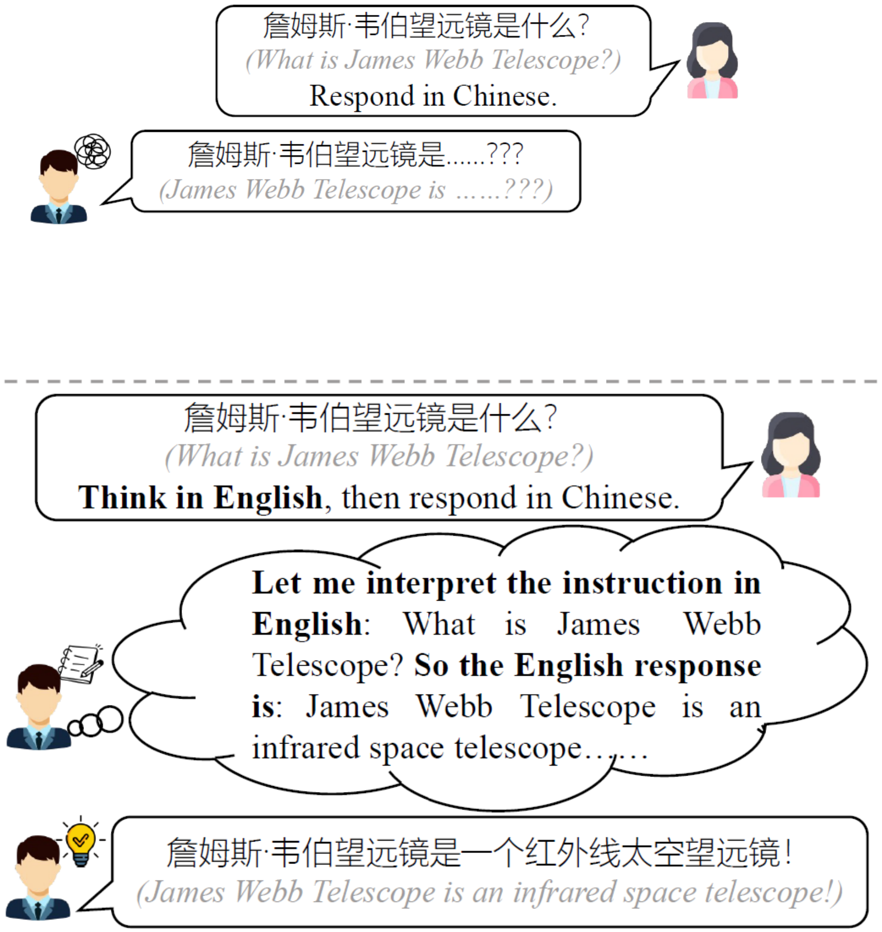
No explicit translation call

The LLM itself is used as a translator **implicitly**

Monolingual QA

Monolingual QA with 'thinking in English' aka

Cross-lingual Thought Prompting



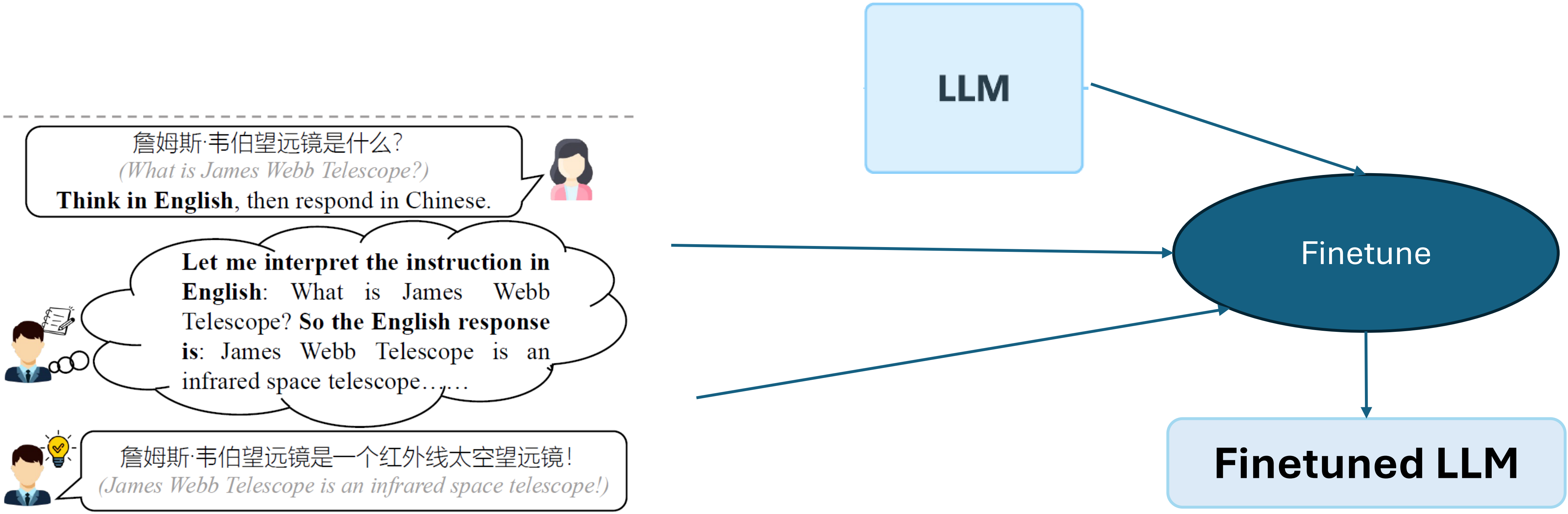
✓ **Multiple inferences are avoided**

✓ **Input in original language is available to LLM**

✗ **Increased token length for model, Reduces possible input token size**

(XLT,PLUG)

TaCo: Instruction tuning with Cross-Lingual Thought data



Limitations

- Reduced maximum sequence length
- Increased latency

	Standard IFT	TaCo
Nepali	39.3	88.1
Persian	46.5	83.9

Evaluation on translated Vicuna Benchmark with GPT4

The PLUG paper also shows improvements (discussed later)

(TaCo,PLUG)

Cross-Lingual Instruction Tuning

	English	*Non-English*
In-language instruction sets	(Q_{en}, A_{en})	(Q_{hi}, A_{hi})
Cross-language instruction sets	(Q_{hi}, A_{en})	(Q_{en}, A_{hi})

Jointly optimize all these objectives

Q_{hi}: मैं अपने समय प्रबंधन कौशल को कैसे सुधार सकता हूँ? मुझे 5 संक्षिप्त बिंदु दीजिये।

A_{en}:
1. Prioritize tasks based on importance and urgency.
2. Set specific goals with deadlines (SMART goals).
3. Use time management tools like calendars and apps.
4. Minimize distractions to maintain focus.
5. Regularly review and adjust your schedule for better efficiency.

Cross-lingual instruction help the model to align languages and think in English

Avoids intermediate English generation of Cross-lingual Thought Prompting

(SDDRL,XCOT,CIT)

Code-switched Instruction Tuning

Code-switching: alternating between two or more languages in conversation

Generate code-switched instruction by replacing some source words with their target language translations

Q_{hi}: मैं अपने time प्रबंधन skill को कैसे सुधार सकता हूँ? मुझे 5 brief बिंदु दीजिये।

A_{en}:
1. महत्व और तात्कालिकता के आधार पर कार्यों को प्राथमिकता दें।
2. समय सीमा (स्मार्ट लक्ष्य) के साथ विशिष्ट लक्ष्य निर्धारित करें।
3. कैलेंडर और ऐप्स जैसे समय प्रबंधन टूल का उपयोग करें।
4. फोकस बनाए रखने के लिए विकर्षणों को कम करें।
5. बेहतर दक्षता के लिए नियमित रूप से अपने शेड्यूल की समीक्षा करें और उसे समायोजित करें।

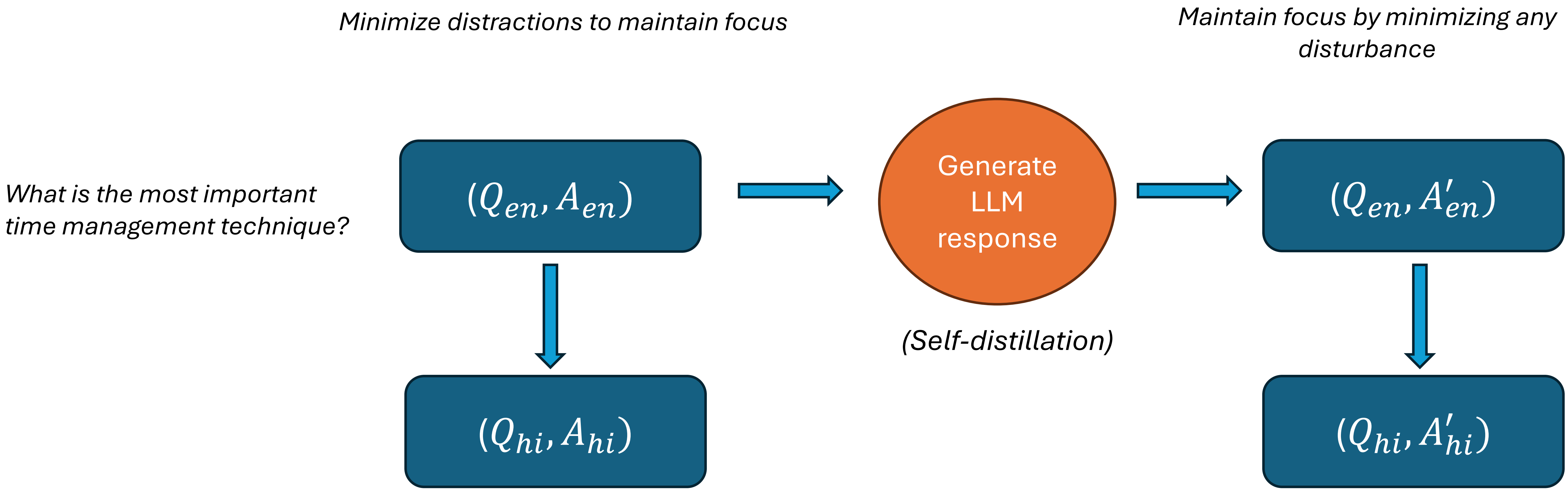
Code-switching help the model to align languages better at sub-sentence level

Some optimizations to improve models further

- Retain English instructions, code-switch only the inputs and outputs
- Do code-switching during pre-training as well

(COMMIT,SDDRL,XCOT)

Self-Distillation from Resource Rich Language

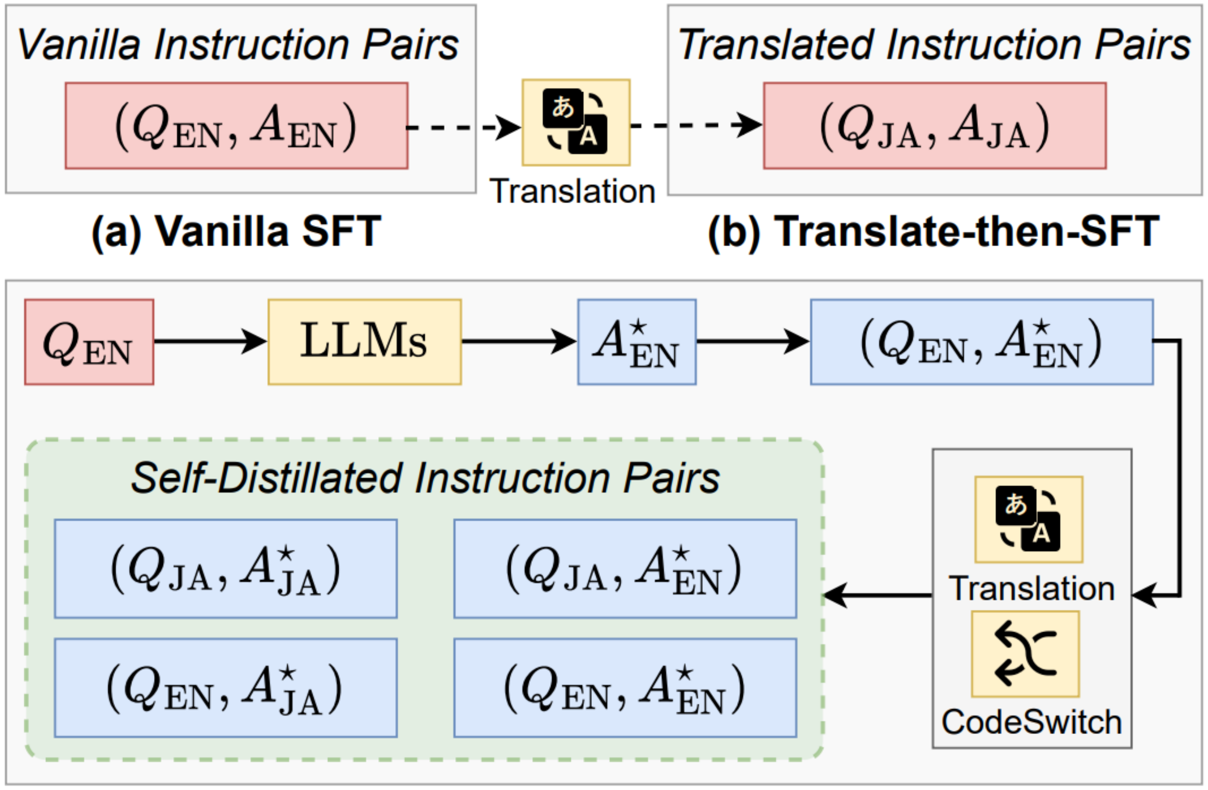


Using the model's own responses can help uses the model's own representation space better

(SDDRL,LMPpl)

Putting it all together

(SDDRL)



(c) Self-Distillation from Resource-Rich Languages (Ours)

+

- Cross-lingual Instructions
- Translation Task
- Sentence Completion Task

(these tasks are added so model preserves native language competency)

Zhang et al. Enhancing Multilingual Capabilities of Large Language Models through Self-Distillation from Resource-Rich Languages. 2024

Summary Results

(SDDRL)

	BELE.	XL-SUM	FLORES	MKQA	AVG.
Performance on Target Language					
SFT	42.24	16.48	18.45	38.86	29.01
T-SFT	42.77	15.32	16.59	43.40	29.52
CIT	42.53	15.75	20.49	43.70	30.62
XCOT	41.19	15.79	17.21	42.04	29.06
SDRRL	**43.67**	**17.89**	**25.86**	**44.63**	**33.01**
Performance on English Language					
SFT	60.19	15.25	28.49	39.62	35.89
T-SFT	58.70	15.63	23.72	37.43	33.87
CIT	58.66	15.42	18.31	36.67	32.27
XCOT	57.73	14.90	23.96	37.94	33.63
SDRRL	**60.67**	**16.24**	**29.47**	**40.32**	**36.68**

(*This summary is for the SeaLLM backbone LLM, results in main paper are for LLama*)

SFT: FT on English data
T-SFT: source and target translated
CIT: target translated
XCOT: source translated + source code-switching

Bringing together all these objectives and data augmentations:

- **Helps improve overall response quality across multiple tasks**
- **Retains English performance**

Ablation Studies (1)

(SDDRL)

		NLU Avg.		NLG Avg.	
		Tar.	Eng	Tar.	Eng
1	Full Method	**50.58**	**66.29**	**28.24**	**31.69**
2	- $\mathcal{D}_{\mathrm{TL}}$ and $\mathcal{D}_{\mathrm{LT}}$	49.56	65.93	26.15	30.55
3	- $\mathcal{D}_{\mathrm{synth}} + \mathcal{D}$	48.59	65.10	25.16	30.10
4	- $\mathcal{D}_{\mathrm{mt}}$ and $\mathcal{D}_{\mathrm{comp}}$	<u>50.41</u>	<u>66.01</u>	26.61	30.19
5	- Code Switching	50.37	65.94	<u>27.13</u>	<u>30.69</u>
6	Only $\mathcal{D}_{\mathrm{mt}}$ and $\mathcal{D}_{\mathrm{comp}}$	41.25	61.61	17.89	22.28

Table 6: Ablation study. Average scores of target language (Tar.) and English (Eng) on natural language understanding task (NLU, including BELEBELE) and natural language generation tasks (NLG, including FLORES, XL-SUM ROUGE-L, and MKQA) are reported.

- Using the LLMs own responses is a very useful method to improve cross-lingual transfer
- The MT and sentence completion tasks are very useful
- The cross-lingual instruction tuning tasks are also complementary
- Code-switching (on input side) has modest benefits

Ablation Studies (2)

(PLUG)

Training Method Comparison	Chinese			Korean			Italian			Spanish		
	Win%	Loss%	Δ%	Win%	Loss%	Δ%	Win%	Loss%	Δ%	Win%	Loss%	Δ%
English-Centric Foundation LLM: LLaMA-2-13B												
PLUG *vs.* Pivot-Only	70.9	19.1	+51.8	76.5	12.7	+63.9	67.6	17.8	+49.8	64.0	20.9	+43.1
PLUG *vs.* Mono. Response	58.0	25.2	+32.8	64.1	19.9	+44.2	50.3	25.8	+24.5	53.0	27.6	+25.5
PLUG *vs.* Mono.+Translation	53.0	28.0	+25.1	62.7	20.1	+42.6	50.1	26.6	+23.5	51.3	25.6	+25.7
PLUG *vs.* Mono.+Code-Switch	50.2	31.6	+18.6	55.2	25.6	+29.6	46.2	30.9	+15.3	48.4	29.9	+18.5

PLUG: Thinking in pivot language
Pivot-only: IFT On pivot language
Mono-Response: IFT on pivot and target language

Mono + Translation: add translation task to Mono-Response
Mono + Code-Switch: add cross-lingual instruction tuning to Mono-Response

Evaluation with GPT4

- Including Translation task is useful
- Training on cross-lingual thought data is most effective
- Cross-lingual instruction tuning is the best next, closes gap on cross-lingual thought data

Summary and Recommendations

- Machine Translation is the dominant method to create IFT data
    - Use English LLMs to generate culture/region-specific data before translation
- Improve alignment between English and other languages using methods like cross-lingual instruction tuning, romanized/code-switched data
- Machine Translation is an important task in the multilingual IFT mix

Summary

- Rapid Advances in Multilingual LLMs
- Extending strong English LLMs to other languages is an effective and efficient direction
- Vocabulary expansion to support new languages and make LLMs efficient, but challenges in achieving convergence
- Continual pre-training important to improve language competence
- Lot of work on aligning languages in the instruction tuning stage

Future Directions

Modeling/Training

- Improving cross-lingual transfer
- Use of synthetic data
- Better "thinking" in English
- Composing Task and Language skills efficiently
- Small Multilingual models
- Multilingual Preference Optimization

Data/Resources

- Scalable evaluation methods for multilingual LLMs
- Creation of multilingual benchmarks
- Collection of large-scale culture-specific text corpora

Thanks

If you find this work useful, please cite it in your work

@online{kunchukuttan2024extendllm,

author = {{Anoop Kunchukuttan}},

title = {Extending English Large Language Models to New Languages: A Survey},

url = {https://anoopkunchukuttan.gitlab.io/publications/presentations/extend_en_llms_apr2024.pdf},

date = {6th August 2024},

urldate = {6th August 2024}

}

Github Page: https://github.com/anoopkunchukuttan/multilingual_extend_llm

Acknowledgments

Based on work, explorations and discussions with many colleagues ...

Raj Dabre, Ratish Puduppully, Jay Gala, Thanmay Jayakumar, Jaavid Aktar Husain, Aswanth Kumar, Mohammed Safi Ur Rahman Khan, Mitesh Khapra, Priyam Mehta, Diptesh Kanojia, Rudra Murthy, Nandini Mundra, Aditya Nanda Kishore, Sumanth Doddapaneni, Rupesh Mehta, Manish Gupta, Diksha Golait

Contact: anoop.kunchukuttan@gmail.com

Home Page: https://anoopkunchukuttan.gitlab.io

Multilingual Pre-training Corpora

- MADLAD-400
- CulturaX
- ROOTS
- mC4
- OSCAR
- CC100
- Glot500-c
- Sangraha
- SEA-LION-PILE

Notable Projects on Extending English LLMs

- BLOOM+1
- ChineseLLama
- Bactrian-X
- Okapi
- SeaLLM
- TOWER
- ALMA and ALMA-R
- AceGPT