

be defined as:

$$f_{\text{prompt}}(x_{\text{test}}) = \mathcal{I} \parallel_{i=1}^K \{ f_{\text{temp}}(x_i) \parallel f_{\text{verb}}(y_i) \} \\ \parallel f_{\text{temp}}(x_{\text{test}})$$

where \parallel denotes the string concatenation operator. The prompt can then be provided as input to the LLM $P(\cdot; \theta)$ to obtain the prediction $z_{\text{test}} = \arg \max_{z \in \mathcal{Z}} P(z | f_{\text{prompt}}(x_{\text{test}}); \theta)$, where \mathcal{Z} is the space of possible answers, which in all of our experiments is taken to be the entirety of the language as modeled by the LLM. We approximate the arg max by sampling from the probability distribution predicted by the LLM.

2.3.1 Multilingual Prompting Strategies

The choice of prompt significantly influences the performance of LLMs and these models have been shown to be brittle to simple prompting variations, such as the choice of prompt template and the training examples or even the ordering of examples (Zhao et al., 2021). For multilingual setups as highlighted in Lin et al. (2022a) and Shi et al. (2022), some additional variations to consider include, the choice of the language of the few-shot examples, the language of the prompt template, and the language of the test examples.

In this work, we evaluate models using three types of prompting strategies: **Monolingual Prompting**: In this setup, the k randomly selected examples are of the same language as the test examples. **Zero-Shot Cross-Lingual**: Here, we evaluate generative models’ zero-shot cross-lingual transfer ability during in-context learning. We use k -shot examples from a pivot language (always English in our experiments) which is different from the language of the test example. **Translate-Test**: In this setup also, the few-shot examples are sampled from English data. However, the test example itself is modified by translating it into English. We use Bing Translator to translate the test examples into English. We do not perform evaluations with Translate-Test prompting for QA and Sequence Labelling tasks where there is no trivial alignment between the labels in the translated text with native language text. To preserve costs, for GPT-4 we only run evaluations with the Monolingual prompting strategy except for a couple of datasets, which we explicitly discuss later in §3. Irrespective of the prompting strategy, we use the prompt templates written in English (see Appendix §A.7 for the impact of this choice).

Prompt Tuning. We use PromptSource (Bach et al., 2022) for a database of existing prompts to use for our experiments. In order to select the best prompt for a dataset (to appropriately measure the capabilities of these models), we evaluate the performance of available English templates on PromptSource on the English validation set and select the prompt that gives the best performance. This prompt template is then used to evaluate models on the test sets for all languages and prompt strategies. While it would be ideal to tune the prompts separately for each language, the scale of our experiments and the computational costs of these models make it prohibitive. We investigate the impact this choice has on our results in §4.1. We perform separate prompt-tuning for DV003 and GPT-3.5-Turbo models, and to keep the costs in check, we use the prompts obtained for the latter for GPT-4 as well. Final prompts selected are included in Appendix §A.4.

Choice of Few-Shot Examples. In all our experiments, we choose few-shot examples randomly from the training or validation set (depending on what’s available) of a dataset. For most datasets, we use 8 few-shot examples, excluding tasks with longer contexts like QA and summarization tasks where we use $k = 4$.

3 Results and Analysis

In this section, we analyze the results of our benchmarking exercise across tasks and languages. Broadly, we cover the comparison between the effectiveness of various prompting strategies §3.1 followed by the performance comparison of GPT-3.5 and GPT-4 models with appropriate baselines §3.2. We conclude with an examination of the factors that affects the performance of these models §3.3.

3.1 Comparing different prompting strategies

In Figure 2, we compare the performance of the three prompting strategies. We find that translate-test often improves the performance over the monolingual strategy, especially so in the case of DV003. We also find that for datasets, which include many low resource and non-latin script languages like IndicXNLI and XStoryCloze, the gains with translate-test are even more substantial for both the models. In Figure 3, we present the average (over different tasks) relative improvement by Translate-Test over Monolingual on GPT-3.5-Turbo for different languages and observe for languages like

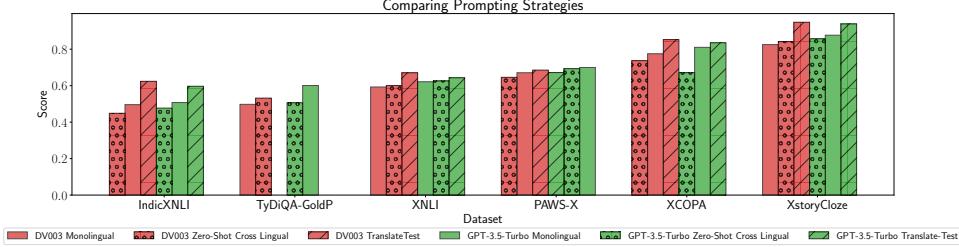


Figure 2: Comparing different prompting strategies discussed in §2.3.1 on DV003 and GPT-3.5-Turbo. The y -axis denotes the task-wise performance metric, e.g. Accuracy for XNLI and F1-Score for TyDiQA-GoldP. A list of metrics for all the tasks is provided in Table 1.

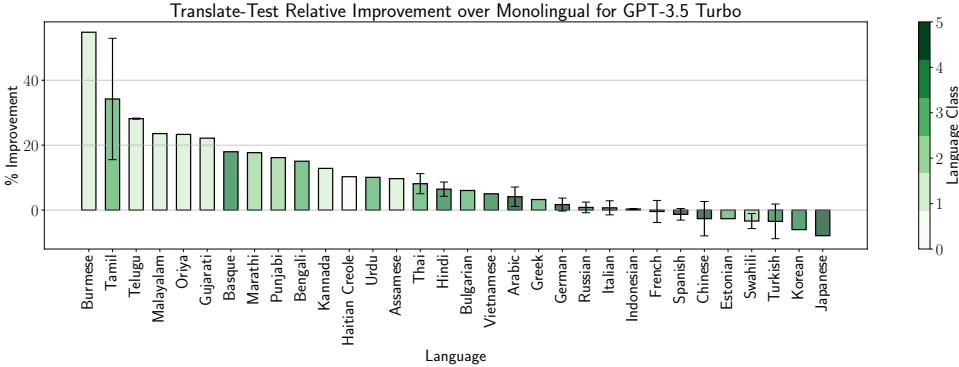


Figure 3: Relative percentage improvement over Monolingual prompting when using Translate-Test for GPT-3.5-Turbo. The bars are color-coded based on the class taxonomy provided in (Joshi et al., 2020)

Burmese, Tamil, and Telugu the relative improvement can be $> 30\%$! In general, we see that for low-resource languages, the translate-test results in substantial improvement in performance, while for high-resource languages the two perform similarly. While we do not evaluate GPT-4 Translate-Test exhaustively for all tasks, we do run the tests for XStoryCloze and XCOPA datasets. Based on these two, we observe that GPT-4’s Monolingual prompting performance is often much more on-par with Translate-Test and many times even better. However, for low-resource languages we again see Translate-Test to perform much better. e.g., in XStoryCloze GPT-4’s accuracy on Burmese is 77.6% vs 93.2% for Monolingual and Translate-Test respectively (Figures 10b and 10d in Appendix).

Note that while Translate-Test substantially improves performance on low-resource languages, compared to the performance of these models in English, the gap even after Translate-Test is significantly high. For example, using translate-test with GPT-3.5-Turbo for Urdu in XNLI results in 54% accuracy compared to 49.1% for monolingual. However, this contrasts with the 76.2% accuracy that the same model achieves in English.

Zero-Shot Cross-Lingual prompting for DV003

often performs on par with Monolingual but for GPT-3.5-Turbo, there is a drop in performance, especially so for tasks like XCOPA which have some extremely low resource languages: Quechua and Haitian Creole. For these languages, we observed that when provided few-shot examples in English, GPT-3.5-Turbo would often resort to predicting outputs like *"I'm sorry, but the premise is not in a language that I understand."*. However, by providing examples in the language, we are able to ground the model to these languages and we almost never observe such predictions in that case.

3.2 Comparing different models

The aggregated results comparing different models and prompting strategies are provided in Table 1 and Table 7 (for Indic Datasets). Excluding the commonsense reasoning tasks XCOPA and XStoryCloze, the OpenAI models generally lag behind the fine-tuned baseline TULRv6 for most tasks often by a significant margin and often are only slightly better than some of the smaller fine-tuned multilingual models i.e. mBERT and mT5-base. Between OpenAI models and BLOOMZ, the former models tend to outperform the latter (despite having a larger proportion of multilingual

Model	Classification				Question Answering			Sequence Labelling	Summarization	
	XNLI	PAWS-X	XCOPA	XStoryCloze	XQuAD	TyDiQA-GoldP	MLQA			
Metrics	Acc.	Acc.	Acc.	Acc.	F1 / EM	F1 / EM	F1 / EM	F1	F1	ROUGE-L
<i>Fine-tuned Baselines</i>										
mBERT	65.4	81.9	56.1	×	64.5 / 49.4	59.7 / 43.9	61.4 / 44.2	71.9	62.2	×
mT5-Base	75.4	86.4	49.9	×	67.0 / 49.0	57.2 / 41.2	64.6 / 45.0	-	55.7	28.1 [†]
XLM-R Large	79.2	86.4	69.2	×	76.6 / 60.8	65.1 / 45.0	71.6 / 53.2	76.2	65.2	×
TuLRv6 - XXL	88.8 [†]	93.2 [†]	82.2 [†]	×	86 / 72.9 [†]	84.6 / 73.8 [†]	81 / 63.9 [†]	83.0 [†]	84.7 [†]	×
<i>Prompt-Based Baselines</i>										
BLOOMZ	54.2	(82.2) [‡]	60.4	76.2	(70.7 / 58.8) [‡]	(75.2 / 63.2) [‡]	-	-	-	-
<i>Open AI Models</i>										
text-davinci-003	59.27	67.08	75.2	74.7	40.5 / 28.0	49.7 / 38.3	44.0 / 28.8	-	-	-
text-davinci-003 (TT)	67.0	68.5	83.8	94.8	×	×	54.9 / 34.6	×	×	-
gpt-3.5-turbo	62.1	70.0	79.1	87.7	60.4 / 38.2	60.1 / 38.4	56.1 / 32.8	60.2 [‡]	40.3	18.8
gpt-3.5-turbo (TT)	64.3	67.2	81.9	93.8	×	×	46.3 / 27.0	×	×	16.0*
gpt-4-32k	75.4 [‡]	73.0	89.7 [‡]	96.5 [‡]	68.3 / 46.6	71.5 / 50.9	67.2 / 43.3 [‡]	66.6 [‡]	55.5 [‡]	19.7 [‡]

Table 1: Average performance across languages in each of the different datasets included in MEGA. TT suffix refers to the translate-test prompting strategy discussed in Section 2.3.1, without any suffix we refer to the monolingual strategy by default (except for XQuAD and IndicQA where it refers to cross-lingual setup). Numbers in **bold** with [†] symbol indicate best performing Fine-tuned model and the ones with [‡] refer to the best prompt-based generative model. The best overall numbers are underlined. For BLOOMZ the values in parenthesis indicate that the model was fine-tuned on the task during multi-task training. Missing values corresponding to the ‘x’ symbol denote experiments that were not applicable and the ones with ‘-’ were the ones deprioritized due to limited compute. gpt-3.5-turbo (TT) on XL-Sum was only evaluated on 29 languages which are supported by Bing Translator.

pre-training data), except for datasets like PAWS-X, XQUAD, and TyDiQA-GoldP, where BLOOMZ performs better. However, it must be noted that all these three datasets were present in the multi-task fine-tuning stage for BLOOMZ, especially for XQUAD and TyDiQA-GoldP for which the validation data that we use for evaluation is also likely to be included in the fine-tuning data³.

Between the OpenAI models, generally DV003 and GPT-3.5-Turbo perform on par, with Translate-Test performance of DV003 being generally better than GPT-3.5-Turbo, and the other way around for Monolingual performance. However, we do observe a notable exception to this, which is for the QA tasks where GPT-3.5-Turbo performs substantially better than DV003, especially so for IndicQA. We attribute this to the fact that in order to fit the prompt in the 4096 context size for DV003, we had to resort to retrieve-then prompt strategy and imperfect retrieval for low-resource languages leads to worse performance. Please check §A.5 of Appendix for more details on this. For GPT-4 on the other hand, we consistently observe substantial improvements, with it being *Pareto Optimal* (Choudhury and Deshpande, 2021) compared to the two GPT-3.5 models for all datasets with an exception of XL-Sum, where for some languages GPT-3.5-Turbo performs better. For the detailed

results spanning all models, tasks, and languages, please refer to Appendix §A.8.

3.3 Factors Explaining Performance Trends

In this section, we try to understand what factors influence our observed trends in multilingual LLM capabilities. We begin by investigating the *Fertility* of the tokenizers used by different models, which is defined as the average number of sub-words produced per tokenized word (higher means worse quality), as that has been shown to critically impact the downstream task performance of pre-trained multilingual models (Rust et al., 2021). In Figure 4, we plot the tokenizer fertility of different models. We observe that the tokenizers for the OpenAI models are substantially worse for low-resource, non-latin script languages: where the fertility for languages like Malayalam and Tamil is so high (~ 10) that the tokenizer essentially operates as a byte-level tokenizer for these languages. Note that this means that for low-resource languages, substantially larger number of tokens are needed to encode the inputs as well as for generation, which results in a significant additional API costs. Ahia et al. (2023) discusses how this phenomenon leads to large socio-economic disparities for speakers of underrepresented languages. We study if these discrepancies in the tokenizer’s quality across languages have any effect on the performance. As can be seen in Figure 5, for six tasks we observe statis-

³Note that this can be a possibility for OpenAI models as well and we discuss this in more detail in §4.2.