Testing the Relation of Politeness to Performance in Pre-Trained Large Language Models

Uri Haran
Student at Tel-Aviv University
Natural Language Processing 2023B
uriharan@mail,tau.ac.il

Abstract

The advent of Pre-Trained Large Language Models has witnessed a remarkable surge in both their performance and utilization. With models now boasting hundreds of billions of parameters, their training has become a formidable challenge in terms of cost and resources. Consequently, alternative techniques have emerged to enhance these models' capabilities without the need for extensive retraining. One such approach, known as Prefix-Tuning, involves the attachment of additional context to the model's input to elicit improved performance. This document presents a modular methodology for scrutinizing the impact of diverse prefixes on any generic language model and dataset, with a specific focus on assessing the influence of politeness levels. By conducting a series of experiments that involve appending both polite and impolite prefixes to the same dataset and employing identical model configurations, this study aims to elucidate the role of politeness in shaping the output quality of these language models.

1 Introduction

In recent years, the world of Natural Language Processing has witnessed a seismic shift, driven by the convergence of abundant computing resources and vast repositories of web-scraped data. These developments have empowered Large Language Models to grow to colossal proportions, granting them access to an unprecedented wealth of examples from which to learn. Pre-training, in which a model is trained on a general, easily automated task, allows models to then be Fine-tuned, and converge quickly to useful results when trained on new tasks. However, as model sizes continue increasing, even this could become prohibitively expensive demand-wise. This is due to the difficulty in computing back-propagation over the model's gradients. As such, many techniques to improve model performance without the use of back-propagation were developed, such as Chain-of-Thought, Few-Shot, and Prefix-Tuning. Notably, the process of Prefix-Tuning, previously done manually by humans, was recently automated (Vos et al., 2022). As the models grew, they exhibit an increasing ability to mimic human behaviour, yet by doing so artifacts of human expression find their way into the model, including elements such as toxicity and bias. Thus, the question arises into whether such biases can be exploited to improve the baseline performance of models. This paper introduces a modular methodology to evaluating prefixes used on models, and uses this methodology to conduct basic testing over the effect of prefixes with a different level of politeness.

2 Methods

In order to achieve a systematic, modular methodology for testing general-purpose input generation to data models, a combined data flow is created. The intent of the data flow is to evaluate Pre-trained models "out-of-the-box" - without any Fine-tuning. To achieve generality, the pipeline utilizes a generic structure for data: each item to be evaluated by the model is constructed from a generator into a set of "Query" - the input question, "Context" - information which the input question requires or is assisted by, and "Answer" - the golden answer provided by the generator. Following that, a text arranger turns the "Query" and "Context", and creates from them a "Text" - the input on which the model is evaluated on. This stage is the one focused on within this paper, as it provides a hook in which different prefixes may be used. Note, however, that this may be expanded to test other methods, such as Chain-of-Thought or Few-Shot. Then, models are loaded into a generic "model" and "tokenizer" pair, to be passed the Texts for inference to result in Outputs. In the end, the Outputs and the Answers are passed to a data comparator, to be evaluated.

In the course of my research, i have used this

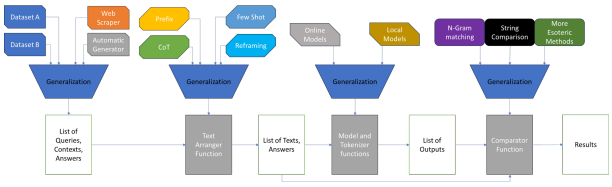

Figure 1: Illustration of the Evaluation Pipeline Methodology

methodology to test the effects of different prefixes, engineered to be of differing politeness levels, evaluated on several different datasets and using several models. The prefixes were designed by selecting from politeness features from (Yeomans et al., 2018), to generate increasing levels of politeness or impoliteness by adding the appropriate features. Split into 6, the politeness levels are, from most impolite to most polite:

- -3: "Answer the following, <Profanity>:" or "Answer the <Profanity> following:"
- -2: "Answer:"
- -1: "Answer the following:"
- 1: "Please answer the following:"
- 2: "Could you please answer the following:"
- 3: "<Hello>, Could you please answer the following:"

Where words in <> are randomly chosen from a matching list. Swear words or degrading names for <Profanity>, and variations of "Hello" (such as "Hey", "Hi there", etc).

The datasets used for evaluating the effects of the different levels of politeness are as follows:

1. Two Review-Based Datasets:
   (a) ChatGPT App Reviews - Reviews containing their title, text, and 1-5 rating.
   (b) McDonald's Store Reviews - Akin to the previous dataset, Except there is no title to the review - only text and rating.
2. MathQA - multiple-choice math questions.
3. Sentiment & Emotions Labelled Tweets - Tweets analyzed to obtain both their sentiment and an associated emotion.
4. A piece of self-written code to generate random boolean expressions and their truth value.

3 Results

Testing was performed on FLAN-T5 small (Raffel et al., 2020). Once from each dataset, except for the Labelled Tweets (once for the sentiments, another for the emotions), 2,000 Questions were generated, then those too long to feed the model were discarded, before evaluating the model's performance. The results (in percentage of success) are shown in tables 1 and 2.

Politeness	MathQA	Sentiment	Emotion
-3	21.45	68.32	32.10
-2	21.61	69.32	35.20
-1	**21.64**	69.38	35.87
1	21.59	69.33	**37.78**
2	21.19	**69.51**	34.50
3	21.06	69.49	35.11

Table 1: Results, part 1

Politeness	McD Rev.	Cgpt Rev.	Boolean
-3	17.36	45.25	8.9
-2	17.50	42.47	12
-1	18.63	45.39	**12.2**
1	**19.77**	46.63	7.25
2	19.50	**47.18**	9.35
3	17.82	46.31	5.1

Table 2: Results, part 2

However, several of those results are definitely eyebrow-raising. Notably, the McDonald's Review's and the Boolean String's result are below random chance. For the reviews, we would assume that if the model randomly guessed it would achieve an accuracy of 20%, and for the boolean strings an accuracy of 50%. Checking the data, the model appears to output answers which are not part of the available solution set tested against. In the Reviews, the answer set is the numbers 1 through 5, yet the models outputs answers such as "0", or possible sentences which could be found in reviews, such as "Great place to go for a quick meal.", "Five Stars", or "Definitely recommend!", even if those answers are not actually present in the original review. In the case of the boolean strings, alongside the "FALSE" and "TRUE" answers, the model outputs answers such as "(FALSE)", "(TRUE)", "(NOT (FALSE))", "(NOT (TRUE))", which are regarded as incorrect due to the answers being compared with string-matching.

Also of note is that several of the results are not statistically significant: The sentiment analysis only achieves a Z-Score[1] of 0.811 between the two furthest distributions, and concatenating all the polite queries in comparison to all the total impolite queries worsens it to a Z-Score of 0.5138. In comparison, Being impolite, as a matter of fact, is surprisingly helpful for evaluating boolean queries (altough whether this is due to reducing artifacting or evaluating better was not tested), achieving a Z-Score of 7.2248 between the combined polite and impolite queries, corresponding to a p-Value of less than 0.00001. The Z-Scores between the distributions can be seen in table 3, in which the Z-Scores between the best and worst performing politeness levels, and the total polite and impolite queries, are shown. A score is considered statistically significant if the Z-Score is above the threshold matching a one-tailed, $p<0.05$, which is approximately 1.65.

Dataset	best-worst	polite-impolite
MathQA	0.4166	0.3569
Sentiment	0.811	0.5138
Emotion	3.6936	1.6028
McD Rev.	1.9016	1.6453
Cgpt Rev.	2.7445	2.3545
Boolean	7.9872	7.2248

Table 3: Z-Values Table

As the result in with the biggest difference in Z-Scores between the best-worst and polite-impolite, in order to obtain a more complete result, I chose to run the Emotions part again, on an additional 4,000 rows of the dataset, and concatenate the results. The percentage of success over those 6000 rows are:

Politeness	Emotion
-3	33.28
-2	35.62
-1	35.94
1	**37.46**
2	34.74
3	34.59

Table 4: Second Run table

Which improved the best-worst Z-Score to 4.7063, yet the polite-impolite remained below the significance threshold at 1.3173. Also of note is that the politeness levels 2 and 3 performed worse than -1 and -2 (Z-Score: 1.7527).

4 Conclusions

From this work several conclusions can be drawn. First, from the emotion classification and chatgpt reviews estimation, we can conclude that the level of politeness can create a noticable effect, to the order of several percent of success over different prefixes - even merely adding "Please" improves the emotion classification by 1.9% and the chatgpt reviews estimation by 1.24%. Secondly, with respect to the sentiment analysis, as a type of task which the model was trained on, it appears that (aside from the use of profanity), the model achieves similar results regardless of the prefix, indicating that prefix politeness is less relevant over tasks which the model was trained on. Thirdly, with respect to the mathematics questions - the model performs barely any better in the task then random chance, and again the change in prefix creates no significant change - indicating that the level of politeness does not affect whether the model achieves success over a task, only how good it is over tasks which it can perform. Lastly, it is important to note that being overly polite also effects the model negatively - in all datasets the politeness level 2 achieved better performance then the level 3.

5 Further Study

Due to failing to successfully run the code over the university's servers, I was restricted to using Google Colaboratory's free resources. Despite my attempts at applying methods to reduce memory usage, I could not use extremely large models without crashing the instance - the best I managed was Flan-T5 base, which was significantly slower, and I determined it would not be able to produce a statistically significant amount of data in the time I had available. This work was made to be tested on a larger set of models, and the difference in effect of politeness with respect to model size has not been determined here. Similarly, the method i used to determine success in evaluation - string matching - was incapable of checking the evaluation of various tasks, such as summarising or translation. It also was incapable of handling Chain-of-Thought, which is why it was not used in this paper. As such, other evaluation methods may be incorporated to

[1]Calculated with the online calculator at https://www.socscistatistics.com/tests/ztest/default2.aspx

test those other tasks, as well as to check whether the effect of politeness changes when incorporated alongside CoF. The last avenue of future study to be done is about the results of the boolean strings, which appear to improve by being less polite, and raise the question of whether being impolite somehow reduces artifacting and constrains the model's outputs to the examples specified.

6 Code

The code and the results generated for the project are available on github, and may be freely used by others for further researching the effects of various query manipulation techniques.

7 References

References

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

David Vos, Till Döhmen, and Sebastian Schelter. 2022. Towards parameter-efficient automation of data wrangling tasks with prefix-tuning. In *NeurIPS 2022 First Table Representation Workshop*.

Michael Yeomans, Alejandro Kantor, and Dustin Tingley. 2018. The politeness Package: Detecting Politeness in Natural Language. *The R Journal*, 10(2):489–502.