The Curious Case of Absolute Position Embeddings

Koustuv Sinha[‡†*] **Amirhossein Kazemnejad** [‡*]
Siva Reddy[‡] **Joelle Pineau**[†‡] **Dieuwke Hupkes**[†] **Adina Williams**[†]
[‡] McGill University / Mila - Quebec AI; [†] Meta AI
{koustuv.sinha,amirhossein.kazemnejad}@mail.mcgill.ca

Abstract

Transformer language models encode the notion of word order using positional information. Most commonly, this positional information is represented by absolute position embeddings (APEs), that are learned from the pretraining data. However, in natural language, it is not *absolute* position that matters, but *relative position*, and the extent to which APEs can capture this type of information has not been investigated. In this work, we observe that models trained with APE over-rely on positional information to the point that they break-down when subjected to sentences with shifted position information. Specifically, when models are subjected to sentences starting from a non-zero position (excluding the effect of priming), they exhibit noticeably degraded performance on zero- to full-shot tasks, across a range of model families and model sizes. Our findings raise questions about the efficacy of APEs to model the relativity of position information, and invite further introspection on the sentence and word order processing strategies employed by these models.

1 Introduction

Recently, Transformer (Vaswani et al., 2017) language models (TLMs) have been widely used for natural language applications. Such models incorporate positional encodings: vectors encoding information about the order of words in context. Many models, such as RoBERTa (Liu et al., 2019), GPT3 (Brown et al., 2020) and OPT (Zhang et al., 2022), utilize *absolute* position embeddings (APEs) that directly encode absolute (linear) word order. APEs appear to contribute to the performance of such models; although when they are removed, some models become sensitive to ablative word scrambles (Sinha et al., 2021), while others work optimally (Haviv et al., 2022). Thus, what precisely APEs contribute remains unclear.

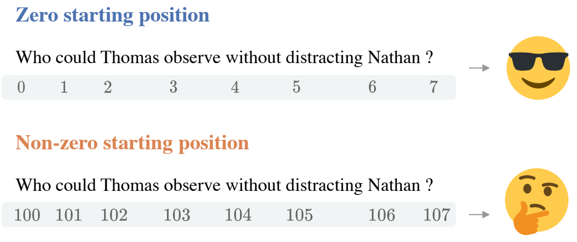
Zero starting position
Who could Thomas observe without distracting Nathan ?
0 1 2 3 4 5 6 7
Non-zero starting position
Who could Thomas observe without distracting Nathan ?
100 101 102 103 104 105 106 107

Figure 1: Transformer models with absolute positional embeddings have different representations for sentences starting from non-zero positions.

It is conceivable that APEs may enable the model to handle the relative distances between words. If models were somehow learning relative position information despite using *absolute* positional embeddings, we would expect sentence encodings to be the same in most cases, regardless of where they appear in the context window. For example, the meaning of “smoking kills” should be constant in “Kim said *smoking kills*” (positions 2–3) and “It was commonly believed by most adult Americans in the 90s that *smoking kills*” (positions 13–14), despite the fact that these words appear in different absolute positions. Given this, our central question is: do APEs enable the model to learn the relative distances between the words in a sentence?

Prior work has attempted to explore the consequences of APEs using probing methods (Wang et al., 2021). APEs have been found to not capture the meaning of absolute or relative positions (Wang and Chen, 2020). APEs have also been found to bias model output with positional artefacts (Luo et al., 2021), leading to better performance on token to position de-correlation (Ke et al., 2021). Haviv et al. (2022) even find that causal TLMs perform adequately even without an explicit APEs. However, a systematic study on relativity of positional encodings is still needed.

To better understand the relativity of absolute

[*] Equal contributions.

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 4449–4472
December 7-11, 2022 ©2022 Association for Computational Linguistics

position embeddings, we first need to ascertain the robustness of relative position understanding for a given input. TLMs are typically trained in a batch containing multiple sentences, with a limited sequence window size, which is typically much larger than an average sentence. We hypothesize that a systematic model should encode the same sentence equally throughout this context window. However, evaluating the encoding of a sentence starting from any position in this window in isolation is hard, as the representation of the sentence would depend on the prior context (Misra et al., 2020; Kassner and Schütze, 2020).

In this work, we subject models from several different architectures and sizes to *phase shifting*. In this paradigm, the sentences exposed to the model are provided contiguous position identifiers starting from a non-zero position (Figure 1). Such inspection allows us to gauge the model's sentence encodings on different positions, emulating sub-window sentence representation, while factoring out the influence of prior context. We investigate several zero shot, few shot and full shot tasks by shifting the start positions of the sentences. We observe the following:

- TLMs display different sub-window sentence representation capabilities, resulting in decreased zero shot task performance and variability in sentence perplexities.
- Autoregressive models, including the recently published OPT (Zhang et al., 2022), show erratic zero and few-shot performance on sub-window representations, highlighting the brittleness of in-context learning evaluation.
- Masked Language Models (MLMs) encode sentences in non-standard positions better than their autoregressive counterparts.
- During fine-tuning models suffer drastically on cross phase-shifted evaluation, suggesting position specific overfitting.

We aim to raise awareness about issues with APEs, which are still widely used in pre-training large language models. Our results highlight the severity of position shortcuts taken by the model during pre-training and fine-tuning, and imply that TLMs may have vastly varying sub-window sentence representation capability than previously assumed. We will release the code and analysis used in this work on Github. [1]

2 Approach

Position encodings used by TLMs come in three broad categories: fixed sinusoidal embeddings as proposed by Vaswani et al. (2017), absolute or learned popularized by BERT (Devlin et al., 2019) family of masked language models, and relative positions (Shaw et al., 2018) used by T5 (Raffel et al., 2020). Wang et al. (2021) presents a comprehensive overview of current encoding strategies.

Despite being an older method, absolute positional embeddings (APEs) are reportedly better than its relative counterparts on several tasks (Ravishankar et al., 2021), and are still used by majority of the large pre-trained TLMs, including the recently released OPT (Zhang et al., 2022). APEs compute token representation after adding the input token to the position embedding for the corresponding position: $x_i = \theta_W[w_i] + \theta_P[i]$, where, $\theta_W \in \mathbf{R}^{|V|\times d}$ is the token vocabulary of size $|V|$, embedding dimension d, and the absolute position embedding matrix $\theta_P \in \mathbf{R}^{|T|\times d}$, where T is the maximum context window size of the model. Now, a sentence $S = [w_1, w_2...w_n]$ containing n tokens, is mapped during inference to positions 1,2, ... n contiguously for all models.

TLMs offer various sizes of *context window*, which is the maximum sequence length in tokens it can train and infer on. Since this context window is usually larger than the average sentence length, multiple sentences can be packed together to “fill" the context window during pre-training. This allows TLMs to learn that sentences can start from various positions in their context window. If models trained with APEs do encode relativity of position, then the sentence representations should be roughly equal throughout the context window, regardless of their starting position.

2.1 Phase Shift Methodology

To understand the relativity of APEs, we examine the model performance under *phase shift* conditions. Phase shift[2] involves right-shifting the absolute positions of all tokens in the sentence by an equal distance k, such that the tokens are now

[1] https://github.com/kazemnejad/lm_pos_investigations

[2] More related to our work, Kiyono et al. (2021) train a Transformer model from scratch using shifted positional embeddings for machine translation, and observe improved performance in extrapolation and intrapolation setup.

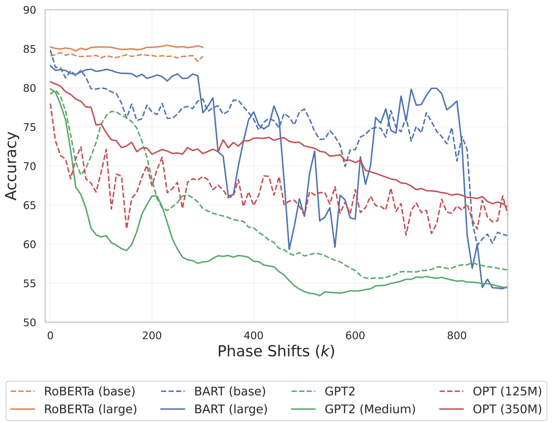

Figure 2: Acceptability Scores in BLiMP (Warstadt et al., 2020) dataset across different phase shifts. RoBERTa only supports context window of size $T = 512$, so we capped the scores to phase shift $k = 300$ to allow for sentences of maximum length in BLiMP to be evaluated.

mapped to new positions $1+k, 2+k, ..., n+k$, or $x_i = \theta_W[w_i] + \theta_P[i+k]$. As such, phase shifting changes only the absolute position, but preserves the relative distances between tokens in the a sentence. Theoretically, we can shift the positions within the context window as long as $k+n \leq T$. For example, given phase shift $k = 100$, and sentence length of n, we could have the following vector of position ids:

$$\vec{p} = [101, 102, 103, \ldots, n + 100]$$

While computing the task scores and perplexities of the models, we observed that all of the models exhibit poor task performance on phase shifts. Due to the non-shiftable nature of the `[CLS]` token in masked language models (MLMs), we first fix the position of `[CLS]` token to start position during phase shifting, which results in significantly improved performance for all models:

$$\vec{p} = [1, 102, 103, \ldots, n + 100]$$

Futhermore, we observed yet another marked improvement in task performance when we use *special tokens* in the beginning of the sentence: typically the end-of-sentence (`[EOS]`) token in case of MLM models (RoBERTa, BART). An explanation for this ambiguity in results is that typically when models are pre-trained, multiple sentences are packed together in the context window by delimiting the start of each sentence with an `[EOS]` token [3]. Thus, in all of our results, we opt with this configuration (adding an `[EOS]` token before the sentence) to ensure fairer evaluation for all model families. Concretely, the input to a model uses the following template [4]:

```
[CLS][EOS]<sentence>
```

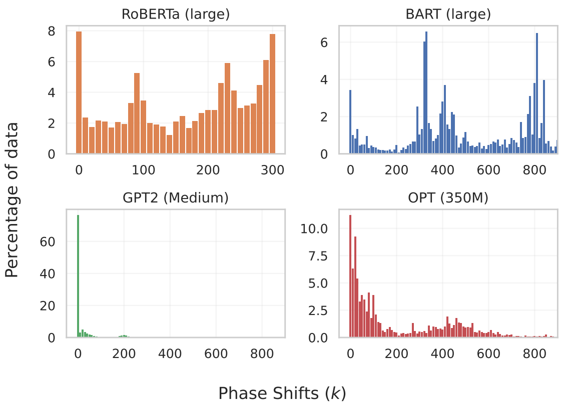

Figure 3: Distribution of sentences in BLiMP (Warstadt et al., 2020) having the lowest perplexities (i.e., are deemed most acceptable) for each phase shift.

3 Impact of phase shifts on grammatical acceptability

First, we investigate the impact of phase shifting on the model performance. We compute the perplexities of several publicly available models—RoBERTa (Liu et al., 2019), BART (Lewis et al., 2020), GPT2 (Radford et al., 2019) and OPT (Zhang et al., 2022)—to evaluate the grammatical acceptability capabilities of the model, using the BLiMP (Warstadt et al., 2020) benchmark.[5] We compute the task score by comparing grammatical and ungrammatical sentence perplexities, and applying the phase shift in increasing values of k to the sentences and models (Figure 2).

We observe that the task performance of all models, except for RoBERTa, drastically suffers from phase shifting. Autoregressive models in particular display worse results. This is likely due to a mismatch of position information learned due to

[3] While this is not the case for GPT2, we also observed improved performance in some cases when we add a beginning of sentence (`[BOS]`) token to the sentence and add a special `[EOS]` token to delimit the start of a sentence.

[4] In cases where a model does not have the `[CLS]` token, we instead use `[BOS]`. If none of those are available, we replace it with `[EOS]` (so a total of two `[EOS]`'s will be prepended).

[5] We adopt the perplexity computation strategy for RoBERTa and BART from Salazar et al. (2020)

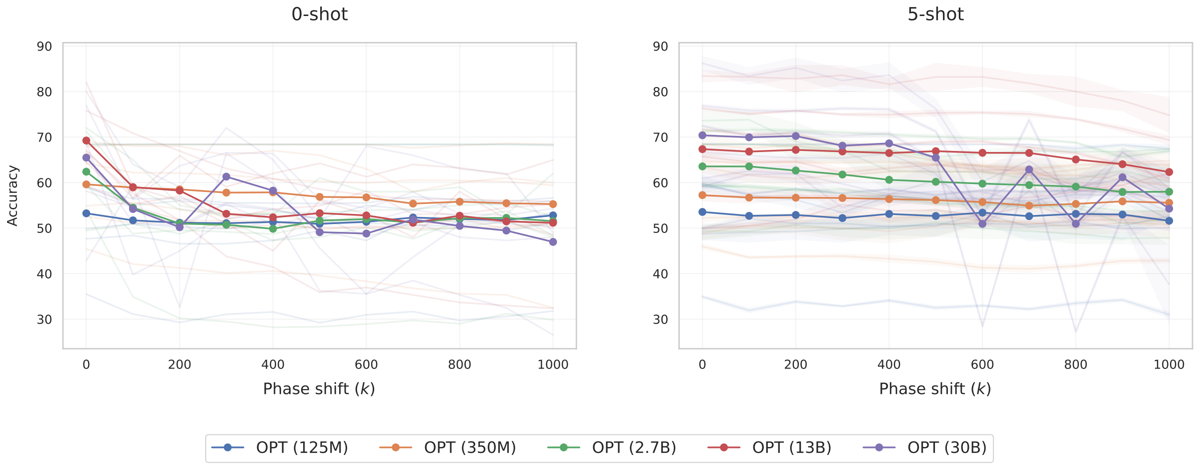

Figure 4: Aggregate performance of OPT family on six NLP tasks when various phase shifts are applied.

the causal language modelling objective vs the position information provided to the model during phase shift (Haviv et al., 2022). We also compare the perplexities of each sentence across different phase shifts and plot the frequency of sentences having the lowest perplexity in each k (Figure 3). We observe in GPT2 that more than 70% of the sentences have their best perplexity in $k = 0$, highlighting a severe zero-position bias. $\text{OPT}_{350\text{M}}$ has better sub-window sentence representation capacity than similarly sized GPT2, which is also evident from the acceptability results in Figure 2.

4 Impact of phase shifts on in-context learning

More recently, zero-shot and few-shot inference, commonly referred to as in-context learning, have become a de facto standard in evaluating pretrained language models (Brown et al., 2020). In this approach, the model's predictions are produced by conditioning it on certain prompts, such as instructions (zero-shot setting) or a few examples of input-output pairs (few-shot setup). In both cases, the model faces an extended input text, and we suspect it will be affected by deficiencies of APE. To evaluate this hypothesis, we employ an experimental setup similar to §3. Under zero-shot and five-shot inference regimes, we assess the model performance on standard NLP tasks when it is fed with inputs in increasing values of phase shifts. We choose OPT model family, because it is available in a wide range of sizes (125M to 30B parameters), allowing allows us to examine the behavior of APE at different scales. Moreover, our evaluations take into account four tasks reported in the original paper: Winogrande (Sakaguchi et al., 2020), COPA (Gordon et al., 2012), PIQA (Bisk et al., 2020), and ARC (Clark et al., 2018) as well as two classification datasets from GLUE benchmark (Wang et al., 2019): MRPC and RTE. We provide an aggregated view of the models' performance on all six accuracy-dominated benchmarks in Figure 4. The detailed plots for each task are in Appendix B.

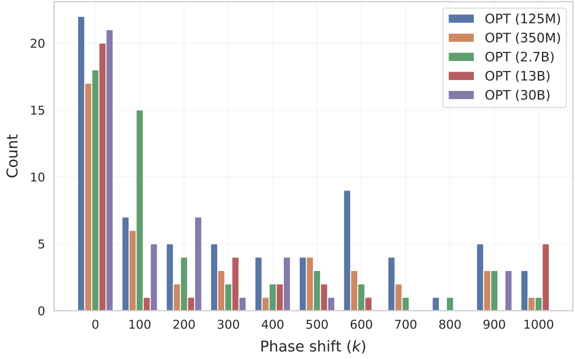

Figure 5: Distribution of prompts with best accuracy across all six tasks.

In most tasks, the performance deteriorates when the model process inputs in any other phase shift than zero, especially in zero-shot inference. More importantly, the model's performance is not always adversely affected by phase shifts. In fact, Figure 5 shows that non-zero starting positions result in the best accuracy for many prompts. This erratic performance is present in all model sizes, and scaling the number of parameters does not help. Furthermore, one can see larger models are more affected by shifted starting position, which suggests that absolute positional embedding might need more data or training as the number of parameters increases.

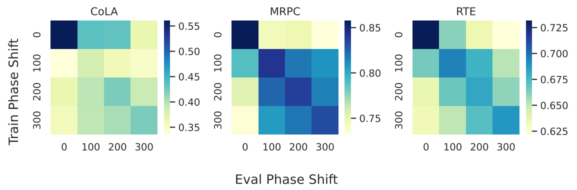

Figure 6: GLUE task heatmap with varying fine-tuning train and test phase shifts, averaged across all models. Darker colors represent better task performance.

5 Impact of phase-shifts on fine-tuning

Finally, we investigate the effect of phase shift in fine-tuning. We ask whether the models can generalize to out-of-phase sentences for a given task. We train RoBERTa, BART, GPT2 and OPT models on CoLA, RTE and MRPC tasks from the GLUE benchmark (Wang et al., 2019) and evaluate them on phase-shifts. We choose these three relatively small tasks in order to decrease the number of gradient updates to position embeddings during fine-tuning. We perform a cross-phase analysis by training and evaluating across different phase shifts ($k = 0, 100, 200, 300$) for all models on the same set of datasets, and show the averaged performance. We observe for all models, the task performance drops during out-of-phase evaluation (non-diagonals in Figure 6).

The drop in performance of evaluating out-of-phase sentences might just be simply attributed to overfitting on position information during fine-tuning. However, we observe that for all tasks, training and evaluating on the same phase-shift is worse when $k \neq 0$ (diagonals in Figure 6). Out-of-phase training appears to be worst for CoLA, which suffers drastically when fine-tuning on different phase shifts. These results highlight a potential task data bias with respect to different positions.

6 Conclusion

In this work, we investigate the abilities of APEs in encoding the relative positions of the tokens in an input. We observe that TLMs using APEs encode sentences differently based on the starting position of the sentence in the context window. This result has major implications in the way we perceive the sentence processing capabilities of TLMs. Specifically, we observe that the representation of the same sentence varies depending on where it is in the context window, such that it impacts zero shot, few shot and full shot task performance of sub-window sentences. Future work could leverage the start position in building robust and position-generalizable models. We hope our work can inform the community on the pitfalls of using APEs, and inspire development and adoption of alternative relative position embedding based approaches.

Limitations

Our work primarily focuses on evaluating the relative position encoding of APEs. We do not focus on the relative position embeddings (Shaw et al., 2018; Raffel et al., 2020) (RPE) as our method of phase-shift analysis is not applicable to those classes of models. RPEs employ a window based position information computation on the fly, which does not require it to store embeddings uniquely for each position. Thus, a phase shift in RPE would not change the sentence processing pipeline, as the model recomputes the position information based on the shifted window. Thus, we need different tools to study the relative position encoding of RPE than the one proposed in this paper.

We also acknowledge that our study is primarily focused on English language data from BLiMP and GLUE. It is likely the same results would hold in a multi-lingual model, however, since many languages are less word order inflexible than English, that should be investigated in a follow-up work.

Ethical Consideration

Our work aims at understanding the difference in sentence representation by shifting position information. In practice, this could yield un-intended results from a TLM deployed in production. Since we observe a large variation in results, we would advise for caution when deploying TLMs in sensitive real world applications, as the relative positioning of a given sentence might evoke different responses from the model. We hope our work can be useful to motivate the use of better positional encoding schemes in pre-training TLMs in future.

Acknowledgements

We would like to thank Kanishka Misra, Shagun Sodhani, Stephen Roller and Kushal Arora for their feedback on the initial versions of this draft. We are also grateful for anonymous reviewers' feedback. Siva Reddy acknowledges the support by the Facebook CIFAR AI Chair program.

References

Mikel Artetxe, Shruti Bhosale, Naman Goyal, Todor Mihaylov, Myle Ott, Sam Shleifer, Xi Victoria Lin, Jingfei Du, Srinivasan Iyer, Ramakanth Pasunuru, Giri Anantharaman, Xian Li, Shuohui Chen, Halil Akin, Mandeep Baines, Louis Martin, Xing Zhou, Punit Singh Koura, Brian O'Horo, Jeff Wang, Luke Zettlemoyer, Mona T. Diab, Zornitsa Kozareva, and Ves Stoyanov. 2021. Efficient large scale language modeling with mixtures of experts. *CoRR*, abs/2112.10684.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *CoRR*, abs/2004.05150.

Yonatan Bisk, Rowan Zellers, Ronan LeBras, Jianfeng Gao, and Yejin Choi. 2020. PIQA: reasoning about physical commonsense in natural language. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7432–7439. AAAI Press.

Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. 2021. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow. If you use this software, please cite it using these metadata.

Sidney Black, Stella Biderman, Eric Hallahan, Quentin Gregory Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, Michael Martin Pieler, USVSN Sai Prashanth, Shivanshu Purohit, Laria Reynolds, Jonathan Tow, Ben Wang, and Samuel Weinbach. 2022. GPT-neox-20b: An open-source autoregressive language model. In *Challenges & Perspectives in Creating Large Language Models*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual.*

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. Palm: Scaling language modeling with pathways.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the AI2 reasoning challenge. *CoRR*, abs/1803.05457.

Róbert Csordás, Kazuki Irie, and Juergen Schmidhuber. 2021. The devil is in the detail: Simple tricks improve systematic generalization of transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 619–634, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Leo Gao, Jonathan Tow, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Kyle McDonell, Niklas Muennighoff, Jason Phang, Laria Reynolds, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2021. A framework for few-shot language model evaluation.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. 2007. The third PASCAL recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 1–9, Prague. Association for Computational Linguistics.

Andrew Gordon, Zornitsa Kozareva, and Melissa Roemmele. 2012. SemEval-2012 task 7: Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 394–398, Montréal, Canada. Association for Computational Linguistics.

Adi Haviv, Ori Ram, Ofir Press, Peter Izsak, and Omer Levy. 2022. Transformer Language Models without Positional Encodings Still Learn Positional Information. *ArXiv preprint*, abs/2203.16634.

Dieuwke Hupkes, Verna Dankers, Mathijs Mul, and Elia Bruni. 2020. Compositionality decomposed: How do neural networks generalise? (extended abstract). In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 5065–5069. International Joint Conferences on Artificial Intelligence Organization. Journal track.

Nora Kassner and Hinrich Schütze. 2020. Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7811–7818, Online. Association for Computational Linguistics.

Guolin Ke, Di He, and Tie-Yan Liu. 2021. Rethinking positional encoding in language pre-training. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Shun Kiyono, Sosuke Kobayashi, Jun Suzuki, and Kentaro Inui. 2021. SHAPE: Shifted absolute position embedding for transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3309–3321, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Brenden M. Lake and Marco Baroni. 2018. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *ICML*.

Hector J. Levesque, Ernest Davis, and L. Morgenstern. 2011. The winograd schema challenge. In *KR*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Ziyang Luo, Artur Kulmizev, and Xiaoxi Mao. 2021. Positional artefacts propagate through masked language model embeddings. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5312–5327, Online. Association for Computational Linguistics.

Brian W. Matthews. 1975. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et biophysica acta*, 405 2:442–51.

Kanishka Misra. 2022. minicons: Enabling flexible behavioral and representational analyses of transformer language models. *ArXiv preprint*, abs/2203.13112.

Kanishka Misra, Allyson Ettinger, and Julia Rayz. 2020. Exploring BERT's sensitivity to lexical cues using tests from semantic priming. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4625–4635, Online. Association for Computational Linguistics.

Santiago Ontanon, Joshua Ainslie, Zachary Fisher, and Vaclav Cvicek. 2022. Making transformers solve compositional tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3591–3607, Dublin, Ireland. Association for Computational Linguistics.

Ofir Press, Noah Smith, and Mike Lewis. 2022. Train short, test long: Attention with linear biases enables input length extrapolation. In *International Conference on Learning Representations*.

Ofir Press, Noah A. Smith, and Mike Lewis. 2021. Shortformer: Better language modeling using shorter inputs. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5493–5505, Online. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

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.

Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. 2021. Do vision transformers see like convolutional neural networks? *Advances in Neural Information Processing Systems*, 34:12116–12128.

Vinit Ravishankar, Andrey Kutuzov, Lilja Øvrelid, and Erik Velldal. 2021. Multilingual ELMo and the effects of corpus sampling. In *Proceedings of the 23rd Nordic Conference on Computational Linguistics (NoDaLiDa)*, pages 378–384, Reykjavik, Iceland (Online). Linköping University Electronic Press, Sweden.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Winogrande: An adversarial winograd schema challenge at scale. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8732–8740. AAAI Press.

Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff. 2020. Masked language model scoring. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2699–2712, Online. Association for Computational Linguistics.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 464–468, New Orleans, Louisiana. Association for Computational Linguistics.

Koustuv Sinha, Robin Jia, Dieuwke Hupkes, Joelle Pineau, Adina Williams, and Douwe Kiela. 2021. Masked language modeling and the distributional hypothesis: Order word matters pre-training for little. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2888–2913, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Ben Wang. 2021. Mesh-Transformer-JAX: Model-Parallel Implementation of Transformer Language Model with JAX. `https://github.com/kingoflolz/mesh-transformer-jax`.

Benyou Wang, Lifeng Shang, Christina Lioma, Xin Jiang, Hao Yang, Qun Liu, and Jakob Grue Simonsen. 2021. On position embeddings in BERT. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Yu-An Wang and Yun-Nung Chen. 2020. What do position embeddings learn? an empirical study of pre-trained language model positional encoding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6840–6849, Online. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. Opt: Open pre-trained transformer language models. *ArXiv*, abs/2205.01068.

A Experiment Details

A.1 Models

We used 11 publicly available pretrained language models in this work, ranging across different architecture families: Encoder, Sequence-to-Sequence, and Auto regressive models. All of them use absolute positional embeddings (APE) that is learned during pretraining. In §4, we follow the standard practice for in-context learning evaluation (Brown et al., 2020; Black et al., 2022; Gao et al., 2021) and use autoregressive models. In our initial experiments, we found GPT2 to have a similar behaviour to OPT models, and since the OPT models are available in a wider range of sizes, we primarily focus on them for these experiments. In fine-tuning (§5) and acceptability (§3) experiments, we assess all model families. However, because of the computational costs associated with these experiments, we opt for model variants with < 1B parameters. The details of all models can be found in Table 1. We use HuggingFace (Wolf et al., 2020) model hub to load, fine-tune train, and run infererence for all models.

A.2 Datasets

We use BLiMP (Warstadt et al., 2020) for the grammatical acceptability experiments in §3 as it is typically employed in a inference-only setting and does not require additional training. For §5, we take three tasks from the standard language understanding benchmark GLUE (Wang et al., 2019) which is often used for finetuning language models: MRPC, RTE, and COLA. In addition to these three tasks, we use four other datasets, COPA, PIQA, WinoGrande, and ARC, on which the OPT family have previously demonstrated good performance (Zhang et al., 2022). Table 2 shows the statistics of all datasets, and the following provides a brief description of them:

- **BLiMP** (Warstadt et al., 2020) is a challenge set designed to measures the model's ability to distinguish between acceptable and unacceptable English sentences. This benchmark consists of synthetic examples created based on expert-crafted grammars, where each instance comes with two versions: one acceptable and one unacceptable.

- **COPA** (Gordon et al., 2012) is an open-domain commonsense causal reasoning task, where the model is given a premise and must correctly identify its cause or effect. COPA consists of short hand-crafted sentences and is provided as a multi-choice task.

- **PIQA** (Bisk et al., 2020) is a physical commonsense benchmark dataset, challenging language models' idea of the physical world. Given a physical goal, a model must choose the most plausible solution between two choices. This benchmark is used in the multi-choice format.

- **WinoGrande** (Sakaguchi et al., 2020) is a commonsense reasoning benchmark based on the Winograd Schema Challenge (WSC) (Levesque et al., 2011) with increased hardness and scale. The dataset is provided as a pronoun resolution problem, where the model must recover an ambiguous pronoun in a given context.

- **ARC** (Clark et al., 2018) is collected from grade-school-level science questions commonly asked in exams. This question-answering dataset is provided in a multi-choice QA format suitable for evaluating pre-trained language models. We use the "easy" subset of this benchmark.

- **MRPC** (Dolan and Brockett, 2005) is a paraphrase identification dataset collected from online news websites and has become a standard benchmark in the NLP community. We follow the previous works and treat the data as a text classification task.

- **RTE** (Giampiccolo et al., 2007) is one of original subtasks in the GLUE benchmark and comprises textual entailment challenges. We follow the standard format and use Natural Language Inference (NLI) protocol for this dataset.

- **CoLA** (Warstadt et al., 2019) is a linguistic acceptability dataset, where each example is an English sentence annotated with a binary label showing whether it is a grammatical sentence. This is a text classification dataset and we follow the standard protocol and report Matthews correlation coefficient (Matthews, 1975).

Model	Type	Pretraining Objective	Context Size	First Position	# Layers	Hidden Size	# Params
RoBERTa family (Liu et al., 2019)							
RoBERTa$_{\text{BASE}}$	encoder-only	Masked Language Modeling	514	2	12	768	123M
RoBERTa$_{\text{LARGE}}$	encoder-only	Masked Language Modeling	514	2	24	1024	325M
BART family (Lewis et al., 2020)							
BART$_{\text{BASE}}$	encoder-decoder	Masked Language Modeling	1024	2	6	768	140M
BART$_{\text{LARGE}}$	encoder-decoder	Masked Language Modeling	1024	2	12	1024	400M
GPT2 family (Radford et al., 2019)							
GPT2	decoder-only	Next Token Prediction	1024	0	12	768	125M
GPT2$_{\text{MEDIUM}}$	decoder-only	Next Token Prediction	1024	0	24	1024	345M
OPT family (Zhang et al., 2022)							
OPT$_{\text{125M}}$	decoder-only	Next Token Prediction	2048	2	12	768	125M
OPT$_{\text{350M}}$	decoder-only	Next Token Prediction	2048	2	24	1024	350M
OPT$_{\text{2.7M}}$	decoder-only	Next Token Prediction	2048	2	32	2560	2.7B
OPT$_{\text{13B}}$	decoder-only	Next Token Prediction	2048	2	40	5120	13B
OPT$_{\text{30B}}$	decoder-only	Next Token Prediction	2048	2	48	7168	30B

Table 1: Details of the models we used in this paper.

Dataset	# Train	# Test/Validation
BliMP	-	67000
COPA	400	100
PIQA	16113	1838
WinoGrande	40398	1267
ARC (Easy)	2251	2376
MRPC	3668	408
RTE	2490	277
CoLA	8551	1043

Table 2: Dataset statistics we used in this work.

Parameter	Value
Learning rate	$\{0.0001, 0.0002, 0.0003\}$
Batch size	$\{16, 32\}$
# Train Epochs	10
Early Stopping	On
Early Stopping Tolerance	3
Optimizer	AdamW
Learning Rate Schedule	Linear
Weight Decay	0.0
Warm Up	6% of initial training steps

Table 3: Summary of hyperparamters used in finetuning experiments.

A.3 Grammatical acceptability

We use all 67 subsets (a total of 67K data instances) of BLiMP (Warstadt et al., 2020). A model achieves a score of 1 if it successfully assigns a lower perplexity to the grammatical version of each example. We report the average score across the entire dataset for starting positions that are shifted in the intervals of 10. The inputs are fed to the models in the format explained in §2.1. Recall that perplexities are ill-defined in case of Masked Language Models. Thus, we follow the formulation of Salazar et al. (2020) to compute a pseudo-perplexity for RoBERTa and BART. We adopt the Minicons (Misra, 2022) library to compute the perplexities, which provides a unified interface for models hosted in HuggingFace (Wolf et al., 2020).

A.4 Prompting

For evaluating zero-shot inference and in-context learning, we make use of EleutherAI Language Model Evaluation Harness (Gao et al., 2021), an open-source library that is used for evaluating autoregressive pretrained language models (Black et al., 2022). In the zero-shot setting, each example is converted to a prompt using task-specific templates. Then, the prompt is fed to the language model to elicit the answer. Similarly, in the few-shot setup, a prompt is created from the concatenation of few dataset examples base on the same template and are prepended as a context to validation instances. In our experiments, we use default templates provided by the EleutherAI Language Model Evaluation Harness, which can be found in Table 4. The task performance is computed over the validation set of due to the lack of public test sets, except for ARC, where we evaluate the models on the test set. We set the number of few-shots examples to be five and randomly sample them from the training set of each dataset. We report the few-shot results averaged over five random seeds. Note that feeding inputs to the models still follows the same protocol introduced in §2.1.

A.5 Fine-tuning

We fine-tune all models on CoLA, RTE and MRPC tasks from the GLUE benchmark on different values of phase shift k, and evaluate across all pos-

Dataset		Template
COPA	Prompt	`<premise>` because/therefore `<possible-continuation>`
	Example	*The water in the teapot started to boil* therefore *the teapot whistled.*
PIQA	Prompt	Question: `<question>`\n Answer: `<possible-answer>`
	Example	Question: *How can I quickly clean my blender without washing?*\n Answer: *Put some ice, water, and a half cup of baking soda in the blender and puree for 3 min.*
WinoGrande	Prompt	`<context>` because `<replaced-pronoun>` `<continuation>`
	Example	*Angela was better suited to conduct the science experiment than Katrina* because *Katrina was less disciplined.*
ARC	Prompt	Question: `<question>`\n Answer: `<possible-answer>`
	Example	Question: *Amanda is learning about different adaptations of animals. Which is an example of a behavioral adaptation?*\n Answer: *migration of songbirds*
MRPC	Prompt	Sentence 1: `<sentence1>`\n Sentence 2: `<sentence2>`\n Question: Do both sentences mean the same thing?\n Answer: `<label>`
	Example	Sentence 1: *Inamed shares closed down nearly 12 percent on Nasdaq, where it was one of the top percentage losers.*\n Sentence 2: *Inamed shares dropped as much as about 16 percent on Nasdaq, where it was one of the top percentage losers.*\n Question: Do both sentences mean the same thing?\n Answer: *yes*
RTE	Prompt	`<premise>`\n Question: `<sentence2>`. True or False?\n Answer: `<label>`
	Example	*United States astronaut Sunita Williams, currently on board the International Space Station, has today broken the record for. . .* \n Question: *Anousheh Ansari paid to go in space.* True or False?\n Answer: *False*
CoLA	Prompt	`<sentence>`\n Question: Does this sentence make sense?\n Answer: `<label>`
	Example	*Brandon read every book that Megan did.*\n Question: Does this sentence make sense?\n Answer: *yes*

Table 4: Prompt templates used in EleutherAI Language Model Evaluation Harness library (Gao et al., 2021)

sible phase shifts. Since RoBERTa only supports 512 positions, and maximum sentence length in these datasets amount to 128, we train models upto $k = 300$. For each fine-tuning experiment, we first run a hyperparameter sweep varying learning rate $(0.0001, 0.0002, 0.0003)$ and training batch size $(16, 32)$ (amounting to 6 runs) with 6% warmup steps, similar to the setting by Liu et al. (2019). We also set the weight decay to zero in order to not harm the existing positional encodings which are not used during training. Table 3 summarizes all of the parameters. Finally, we choose the best hyperparams and repeat the experiment over five different seeds (42 to 46), and present an aggregate over the results. Table 5 lists the outcome of hyperparameters tuning.

In Figure 7, we further show the difference in fine-tuned models when trained on no phase shift $(k = 0)$ and evaluated on different phase shifts $(k = 100, 200, 300)$. In-line with our experimental results from §3, we observe worse generalization results from BART.

B Detailed results on phase shifting with prompts

We displayed a holistic view of zero-shot and five-shot experiments in Figure 4, covering the accuracies averaged over all six datasets. In this section, we now report and analyze the result of each dataset individually. Figure 9 and Figure 10 showcase models' performance in zero-shot and five-shot configurations. The same pattern can be seen across all model sizes in COPA, WinoGrande, PIQA, ARC (Easy), and RTE. Concretely, the zero-shot abilities of the models sharply decrease as we increase the starting position. Moreover, five-shot inference, typically referred to as in-context learning, is also subject to decreased performance, ranging from -2% to -40%. However, the degradation is not as severe as with zero-shot setting. Only MRPC exhibits stable phase shift performance, but even in this case, larger models are still adversely affected. Due to the exceptionally poor performance of OPT family on CoLA, we exclude these results from our analysis (Figure 10).

The erratic behaviour observed in majority of evaluated datasets makes it evident that models struggle to encode the relative distances of words as

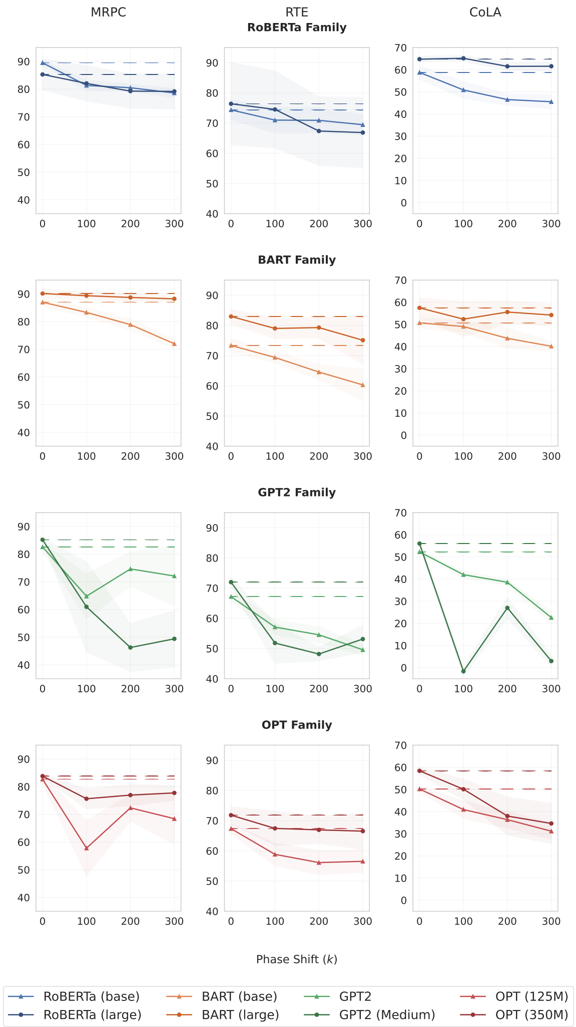

Figure 7: GLUE downstream task results on CoLA, RTE and MRPC. The dashed lines represent the model performance with no phase shifts. The shaded area show the standard deviation from five random seeds.

their understanding of inputs heavily change with various phase shifts. It is important to note that our findings demonstrate models' unstable functioning as opposed to solely highlighting their failure. Indeed, Figure 5 shows that one can extract better and improved accuracies with non-zero starting positions. Namely, $\text{OPT}_{30\text{B}}$ has the best zero-shot performance on phase shift $k = 300$ in the case of MRPC; the same pattern can also be observed in RTE five-shot for $\text{OPT}_{13\text{B}}$ on phase shift $k = 300$. Another noteworthy observation is that the performance drop is often a *non-monotonic* function of phase shifts. i.e., for some prompts, the model might be more accurate for $k = 1000$ than for $k = 0$. This observation suggests that some positional biases might be learned during pre-training and are well-captured by APE. So, increasing values of k in some occasions lands the model attentions in a "sweet spot" in the processing window, such that the model benefits from some positional biases learned during pre-training.

We observe the presence of erratic behavior across a fairly wide range of model sizes in the OPT family. Additionally, it can be seen that larger models are more prone to fail at encoding relative positions than their smaller counterparts. One possible explanation for this is that in order for the models to encode relative positional information, they need to view all combinations of words and sentences in every position. This coverage rarely occurs in natural data, resulting in data sparsity issues. Hence, models with a large number of parameters may require more data/training to learn the relative ordering of words.

C Variation of best perplexity across phase shifts

In this section, we investigate the perplexity of individual sentences from the BLiMP dataset across each phase shift for each model. We plot the distribution of sentences achieving lowest perplexity in each phase shift for the range of models in Figure 8. We observe several modes of phase shift for RoBERTa and BART models where they have the least perplexity on phase shifts other than the standard (zero position). In the case of GPT2 and OPT, the distribution is more skewed towards zero, indicating they almost always achieve the lowest perplexity in the zero position, i.e. when there is no phase shift.

D Code and reproducibility

For all of the experiments in this work, we used open-source libraries (Wolf et al., 2020; Gao et al., 2021; Misra, 2022) and models with publicly available checkpoints. The code to reproduce the results can be accessed from `https://github.com/kazemnejad/lm_pos_investigations`. Furthermore, Listing 1 provides a short, easy-to-use code snippet to modify starting position in HuggingFace models. (We will also release a singularity image with all dependencies to facilitate reproducibility.) We ran our experiments on a mix of NVIDIA A100 40G and NVIDIA RTX8000 48G GPUs. In particular, almost all experiments required only one of such GPUs. The exception was only in the prompting section, where the OPT$_{30B}$ model required two NVIDIA RTX8000 48G GPUs to fit the model and inputs of batch size 1.

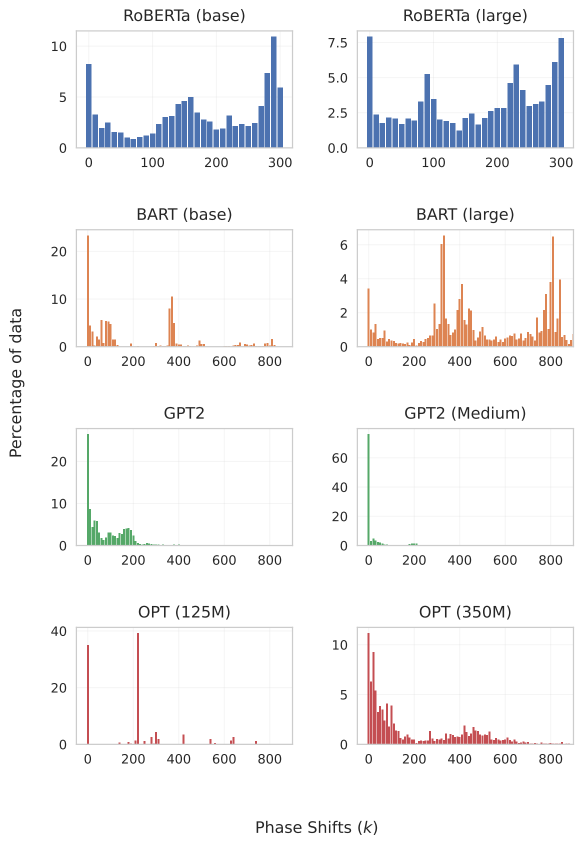

Figure 8: Distribution of sentences having the lowest perplexities for each phase shift

E Attention analysis

We further perform attention analysis on GPT2, RoBERTa and BART to visualize whether the model's attention pattern changes with phase shifts.

```
import torch
from transformers import AutoModelForCausalLM, AutoTokenizer

# Download and load the pretrained model
tokenizer = AutoTokenizer.from_pretrained("GPT2-medium")
model = AutoModelForCausalLM.from_pretrained("GPT2-medium")

text = "The capital of France is"
inputs = tokenizer(text, return_tensors="pt")

# Create unshifted position ids from the attention_mask, which
↪  is equivalent to
↪  torch.arange(inputs["input_ids"].shape[-1])
inputs["position_ids"] = inputs["attention_mask"].cumsum(-1)
↪  -1
print(inputs["position_ids"])
# >>> tensor([[0, 1, 2, 3, 4]])

output1 = model(**inputs, return_dict=True)
next_token_id = torch.argmax(output1.logits[0, -1])
print(tokenizer.decode(next_token_id))
# >>>  Paris

# Add special tokens
special_tokens = torch.LongTensor([tokenizer.bos_token_id,
↪  tokenizer.eos_token_id])
special_attention_mask = torch.LongTensor([1,1])
inputs['input_ids'] = torch.cat([special_tokens,
↪  inputs['input_ids'][0]]).unsqueeze(0)
inputs['attention_mask'] = torch.cat([special_attention_mask,
↪  inputs['attention_mask'][0]]).unsqueeze(0)

# Recompute position ids
inputs["position_ids"] = inputs["attention_mask"].cumsum(-1)
↪  -1

# Shift the position ids by 10
inputs["position_ids"] += 9
inputs["position_ids"][0, 0] = 0
print(inputs["position_ids"])
# >>> tensor([[ 0, 10, 11, 12, 13, 14, 15]])

output2 = model(**inputs, return_dict=True)
next_token_id = torch.argmax(output2.logits[0, -1])
print(tokenizer.decode(next_token_id))
# >>>  the
```

Listing 1: Python code example to shift the starting position of a sentence from $k = 0$ to $k = 10$.

Following the experimental protocol of Raghu et al. (2021), we first collect a summary of attention weights computed with token distances for each token-pair in a sentence. This summary metric is then further normalized for sentence length. The values of this metric show whether the attention is local (low values)—focused on small token distances—or global (high values)—i.e. focused on the whole sentence.

We compute this attention summary metric on a sample of 5000 sentences drawn from the BLiMP

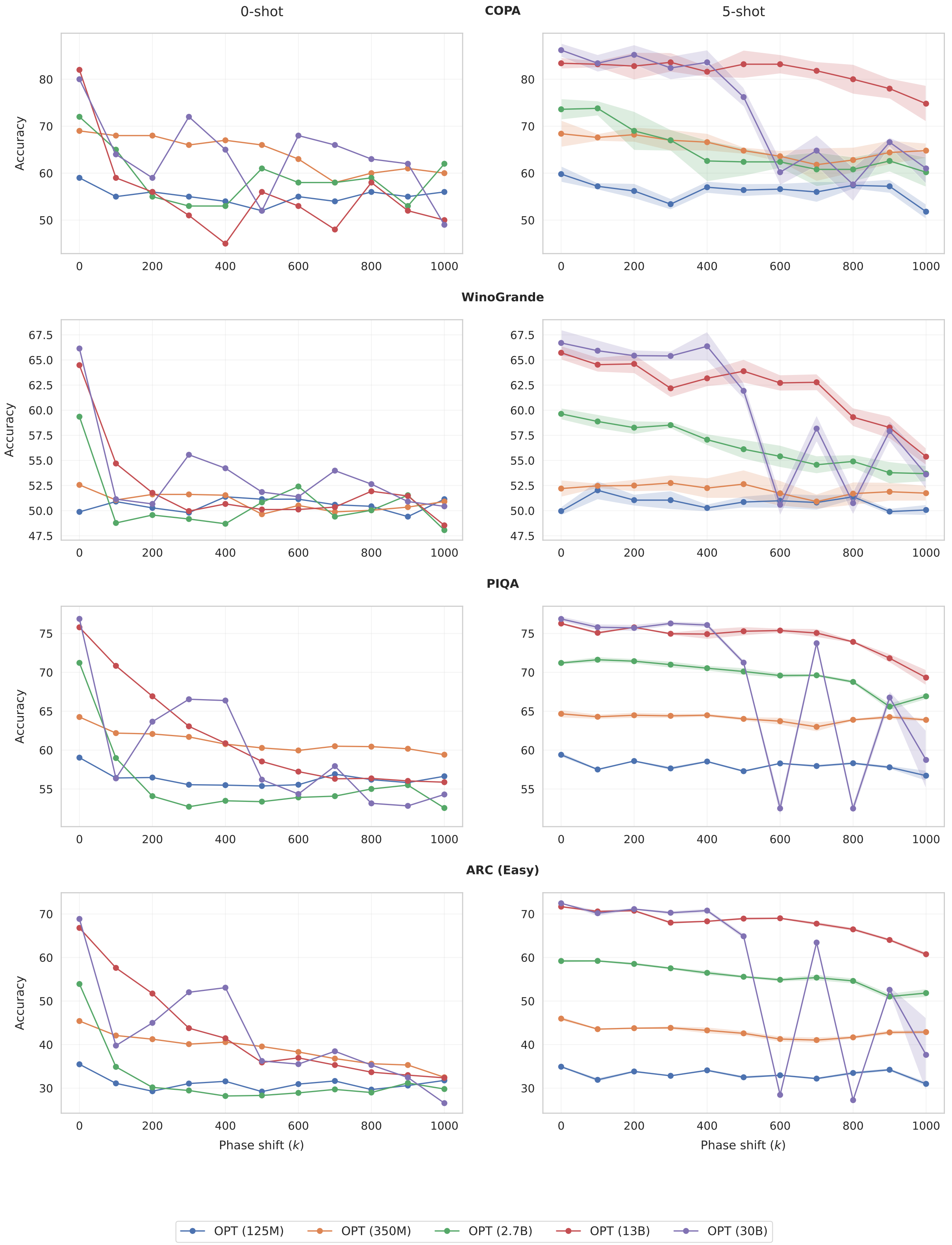

Figure 9: Zero-shot and Few-shot performance of OPT family with various phase shifts for each individual dataset (Part 1)

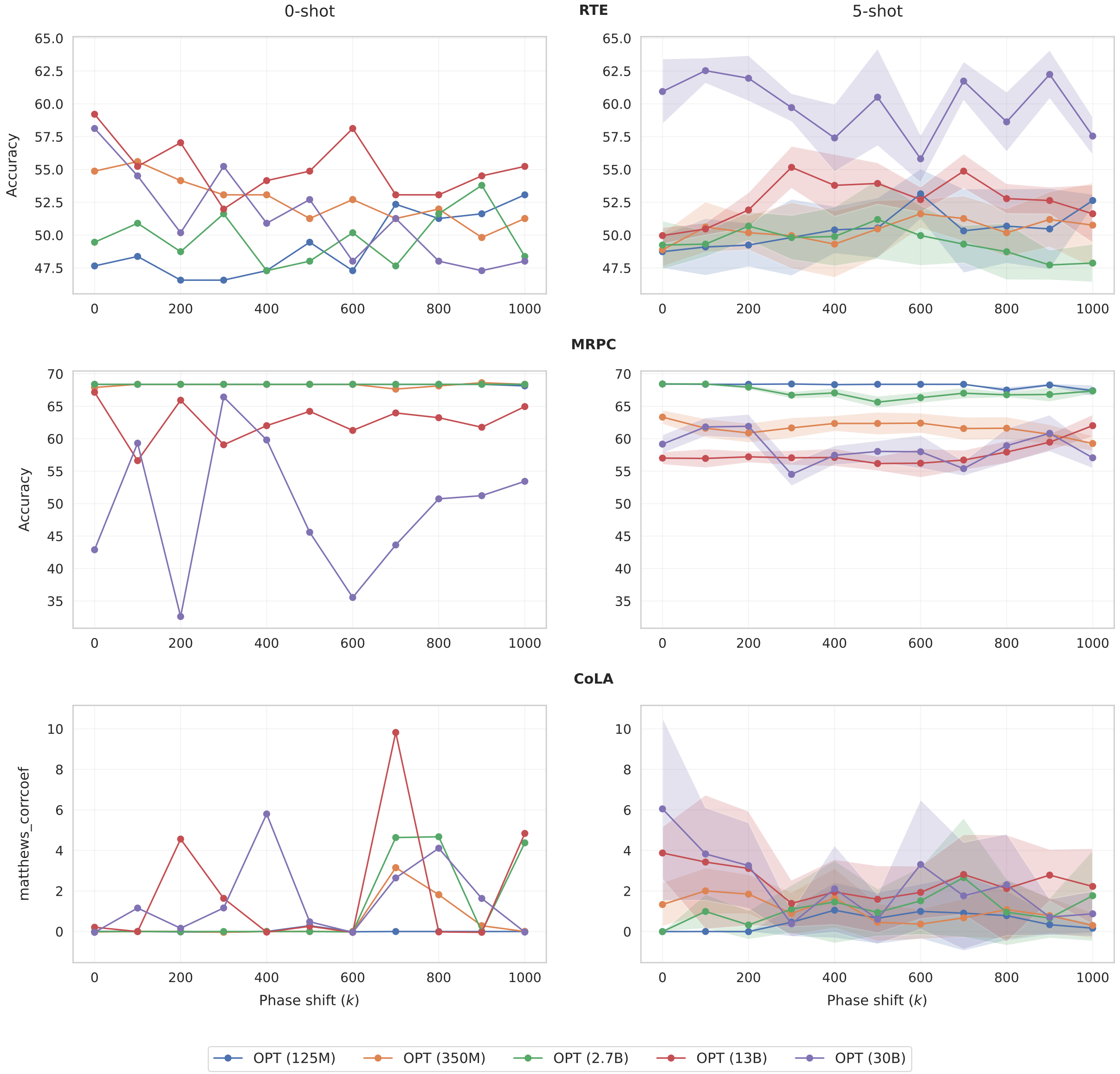

Figure 10: Zero-shot and Few-shot performance of OPT family with various phase shifts for each individual dataset (Part 2)

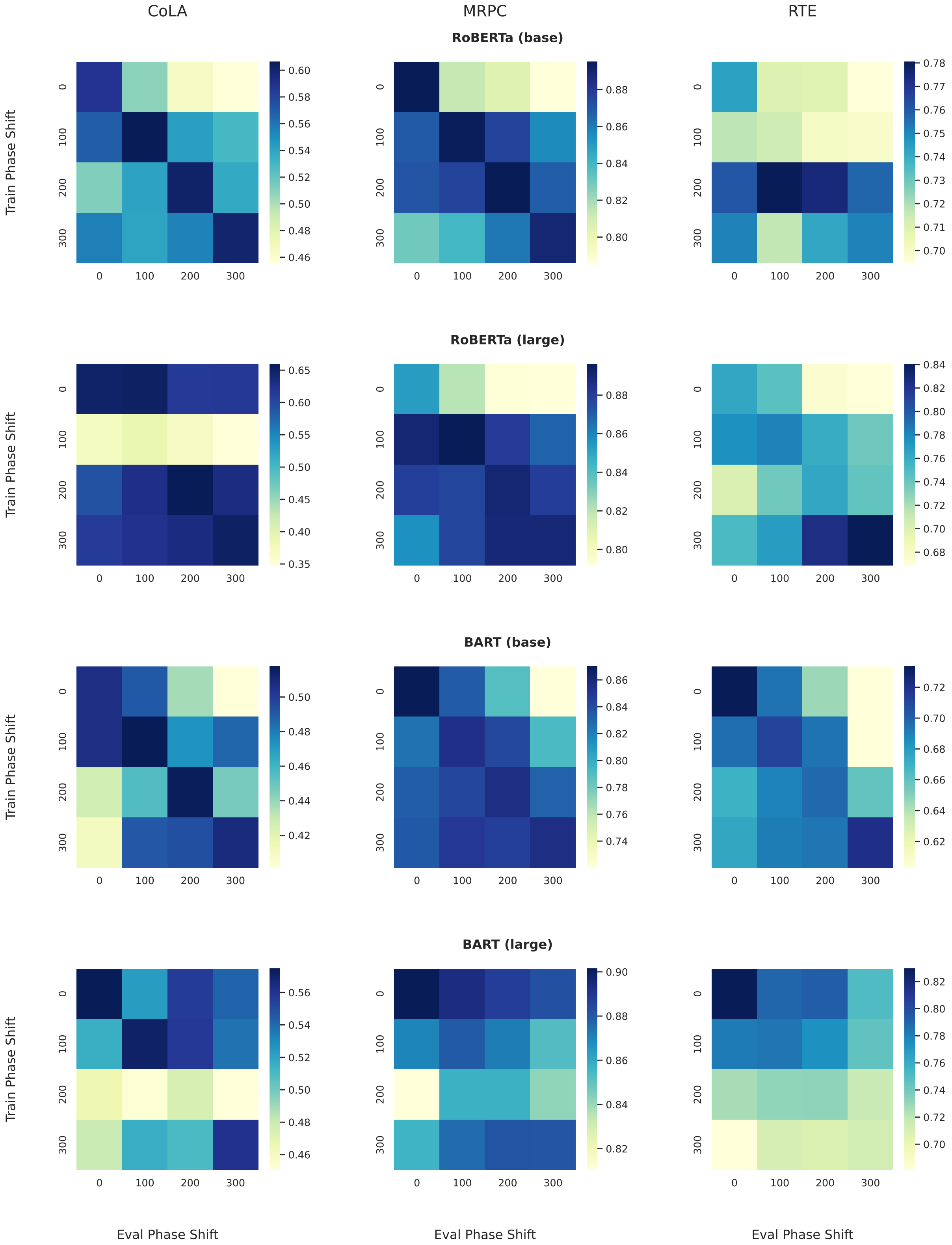

Figure 11: Individual heatmap for each GLUE task and model with varying train (fine-tune) and test phase. (Part 1)

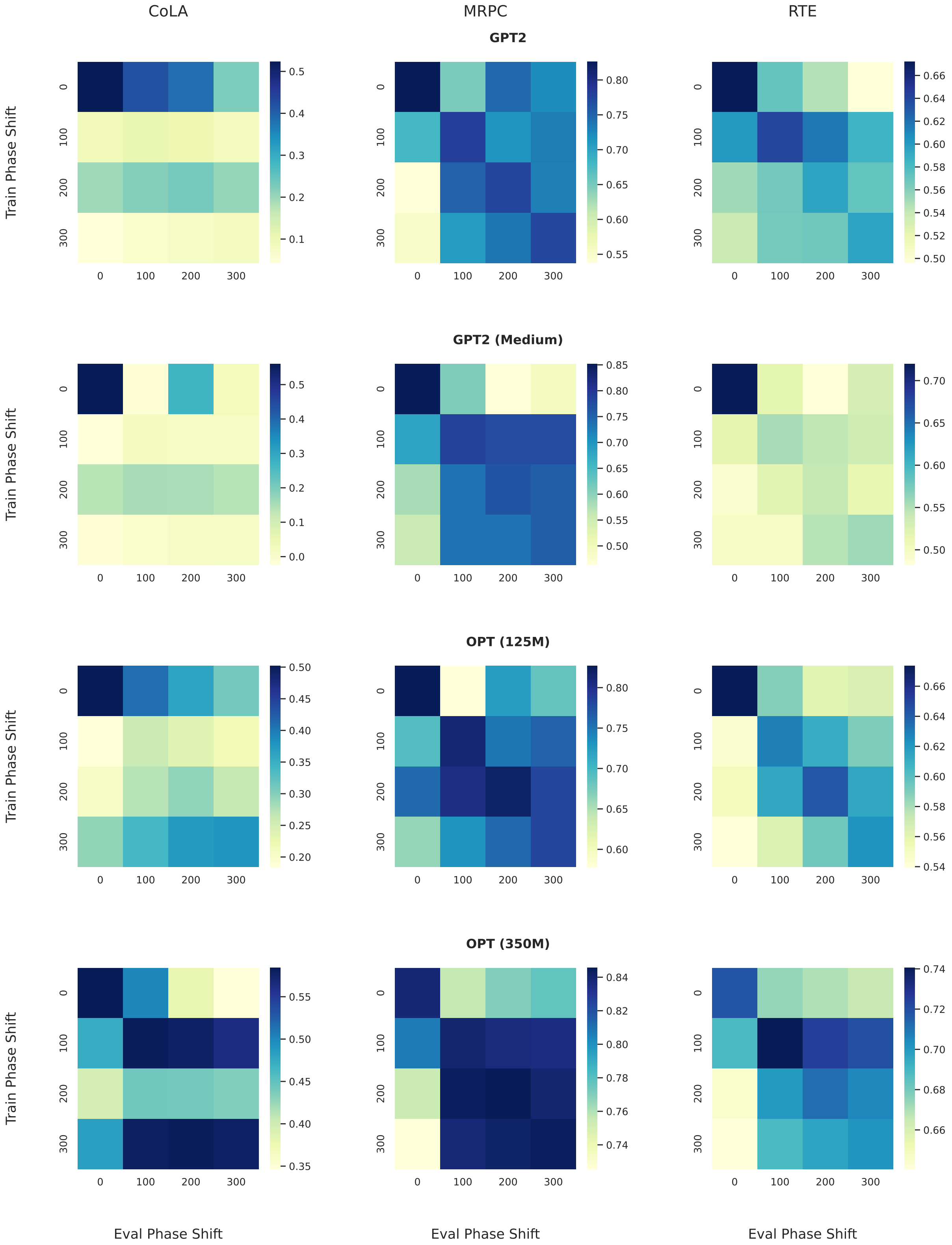

Figure 12: Individual heatmap for each GLUE task and model with varying train (fine-tune) and test phase. (Part 2)

	Phase shifts							
	$k = 0$		$k = 100$		$k = 200$		$k = 300$	
Model	Learning Rate	Batch Size	Learning Rate	Batch Size	Learning Rate	Batch Size	Learning Rate	Batch Size
				CoLA				
RoBERTa$_{\text{BASE}}$	0.00002	32	0.00002	16	0.00002	16	0.00002	16
RoBERTa$_{\text{LARGE}}$	0.00003	32	0.00003	32	0.00001	32	0.00002	16
BART$_{\text{BASE}}$	0.00002	32	0.00003	16	0.00002	16	0.00002	32
BART$_{\text{LARGE}}$	0.00002	16	0.00003	32	0.00003	16	0.00003	32
GPT2	0.00002	16	0.00003	32	0.00003	16	0.00003	16
GPT2$_{\text{MEDIUM}}$	0.00002	32	0.00001	16	0.00003	16	0.00003	16
OPT$_{\text{125M}}$	0.00002	16	0.00001	16	0.00001	32	0.00001	16
OPT$_{\text{350M}}$	0.00001	16	0.00001	32	0.00002	32	0.00001	16
				MRPC				
RoBERTa$_{\text{BASE}}$	0.00002	32	0.00003	16	0.00003	32	0.00001	32
RoBERTa$_{\text{LARGE}}$	0.00002	32	0.00001	16	0.00002	32	0.00002	16
BART$_{\text{BASE}}$	0.00001	16	0.00003	32	0.00002	16	0.00003	16
BART$_{\text{LARGE}}$	0.00002	16	0.00003	16	0.00002	16	0.00003	16
GPT2	0.00002	16	0.00003	16	0.00002	16	0.00003	16
GPT2$_{\text{MEDIUM}}$	0.00002	16	0.00003	16	0.00003	16	0.00003	16
OPT$_{\text{125M}}$	0.00003	16	0.00002	32	0.00002	16	0.00003	32
OPT$_{\text{350M}}$	0.00003	32	0.00001	16	0.00001	32	0.00001	32
				RTE				
RoBERTa$_{\text{BASE}}$	0.00002	16	0.00003	16	0.00002	16	0.00002	16
RoBERTa$_{\text{LARGE}}$	0.00003	32	0.00001	32	0.00003	32	0.00001	32
BART$_{\text{BASE}}$	0.00003	16	0.00003	32	0.00002	32	0.00003	16
BART$_{\text{LARGE}}$	0.00003	32	0.00003	16	0.00002	16	0.00003	16
GPT2	0.00001	16	0.00003	16	0.00003	16	0.00003	16
GPT2$_{\text{MEDIUM}}$	0.00002	16	0.00003	16	0.00001	16	0.00002	32
OPT$_{\text{125M}}$	0.00003	16	0.00001	32	0.00001	16	0.00001	32
OPT$_{\text{350M}}$	0.00001	16	0.00001	16	0.00001	32	0.00001	16

Table 5: Result of hyperparamter sweep for finetuning experiments.

dataset (Warstadt et al., 2020). We then plot the summary values per layer and sort according to the values for each attention head, as per Raghu et al. (2021). The idea is to discover whether this attention summary metric is drastically different under different phase shift conditions.

We do observe drastic differences in attention patterns in all layers for GPT2 (Figure 13) and GPT2-Medium (Figure 14). Comparing this with of RoBERTa (base) (Figure 15) and RoBERTa (large) (Figure 16), we can corroborate our findings from §3—RoBERTa is much more robust to phase shifts. Consequently, BART (Figure 17 and Figure 18) also displays differences in attention patterns, but they are not as drastic as GPT2.

F Extended Related Work

Positional encoding has been always an important part of the Transformer architecture, and since it original introduction different variants of it have been deployed by pretrained models (see Table 6 for a summary of positional encoding used by some of popular state-of-the-art models.)

Positional encodings have garnered a niche community over the past several years. Wang and Chen (2020) investigate whether position embeddings learn the meaning of positions and how do they affect the learnability for different downstream tasks. Wang et al. (2021) explore different positional encodings and establish monotonicity, translation and symmetry properties of different methods, including APEs. They also report that learned APE's demonstrate superior performance for text classification, further adding to the evidence APE's enable exploitation of positional biases. Luo et al. (2021) report that masked language model embeddings consists of positional artefacts which bias the model output. More related to our work, Kiyono et al. (2021) train a Transformer model from scratch using shifted positional embeddings for machine translation, and observe improved performance in extrapolation and intrapolation setup. Haviv et al. (2022) reports a surprising finding that autoregressive Transformer models trained without explicit positional information still perform on-par with their counterparts having access to positional information. This result is attributed to the causal attention structure induced by the autoregressive training only, as this effect is not observed with masked language models, as highlighted by both Haviv et al. (2022) and Sinha et al. (2021). Ke et al. (2021) proposes a novel technique to de-correlate the position encodings and token embeddings, and achieve better downstream performance than baselines. Ravishankar et al. (2021) find relative positional encoding does not improve over APE in

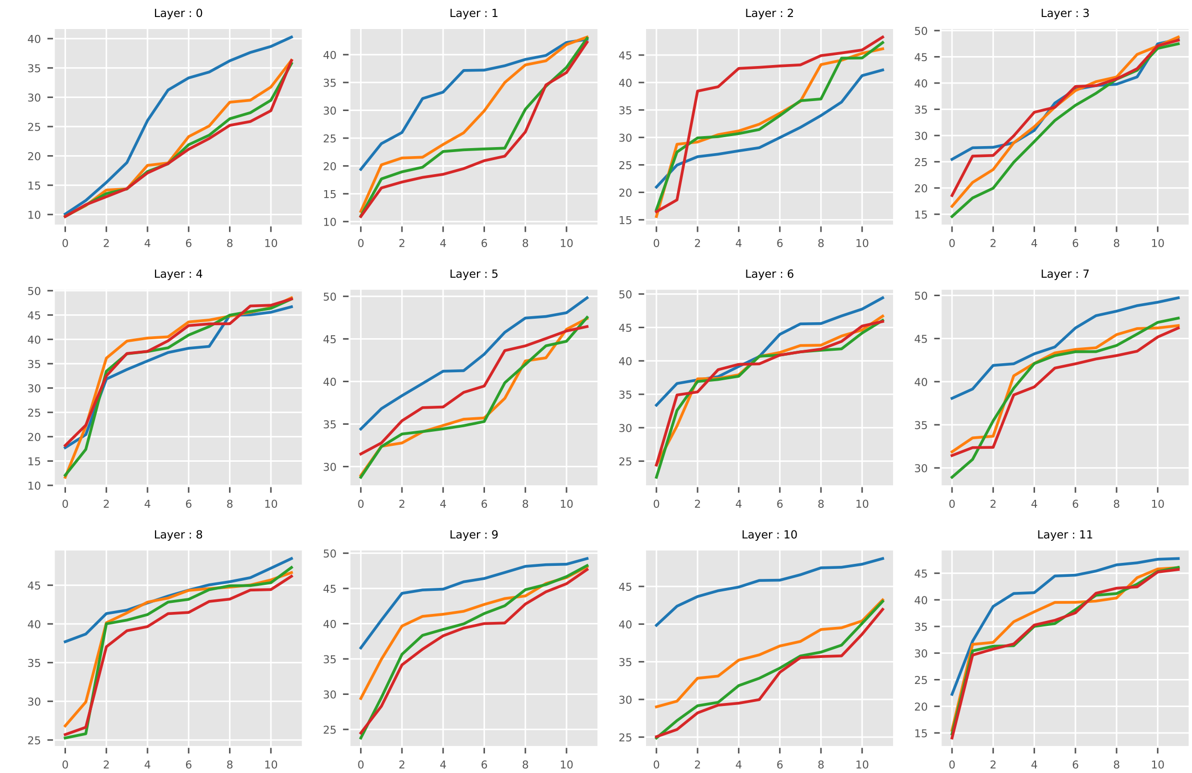

Figure 13: Attention globality distributions of GPT2 across different heads (sorted according to value) and averaged over all layers and 5000 data points. Blue curve stands for the no phase shift condition, and orange, green and red curves represent $k = 100, 200$ and 300 respectively.

multi-lingual setting.

On the other hand, multiple works have shown the advantage of explicit relative positional encoding for length extrapolation. Csordás et al. (2021) show Transformers equipped with variants of relative positional encoding (Dai et al., 2019; Shaw et al., 2018) significantly outperform their absolute counterparts when it comes to length generalization. In the same line of work, Ontanon et al. (2022) also find that for numerous synthetic benchmarks, the best extrapolation performance can only be obtained by relative positional encoding. Press et al. (2022) take the experiments beyond synthetic datasets and show that APE's struggle in generalization to longer sequence of natural language. All of these amount to the evidence that points to APE's as one of the potential reasons Transformers are known to fail in length generalization and productivity (Hupkes et al., 2020; Lake and Baroni, 2018). Although the benefits of using explicit relative positional bias is mentioned in various works, they typically come at the cost of slowing the training down: (Press et al., 2022) report that training T5 (which uses a relative variant of positional encoding) is almost twice as slow as training a model with sinusoidal absolute embedding. Thus, the gained runtime efficiency allows longer training of the APE model, which in turn enables the further extrapolation capabilities. These works suggest that we have a lot left to explore about positional encoding and highlight the fact that the consequences of particular choices is still an open field of ongoing research.

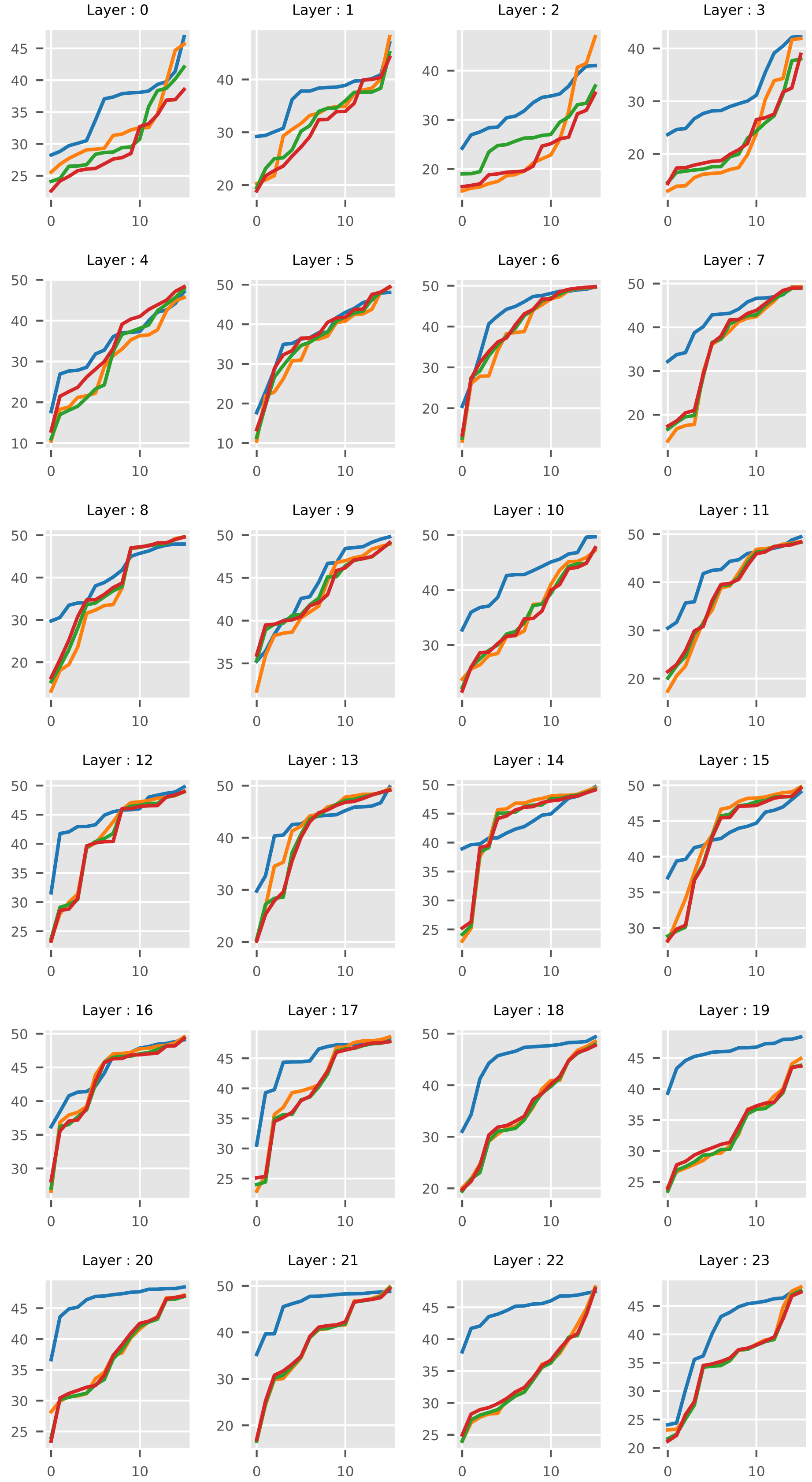

Figure 14: Attention globality distributions of GPT2-Medium across different heads (sorted according to value) and averaged over all layers and 5000 data points. Blue curve stands for the no phase shift condition, and orange, green and red curves represent $k = 100, 200$ and 300 respectively.

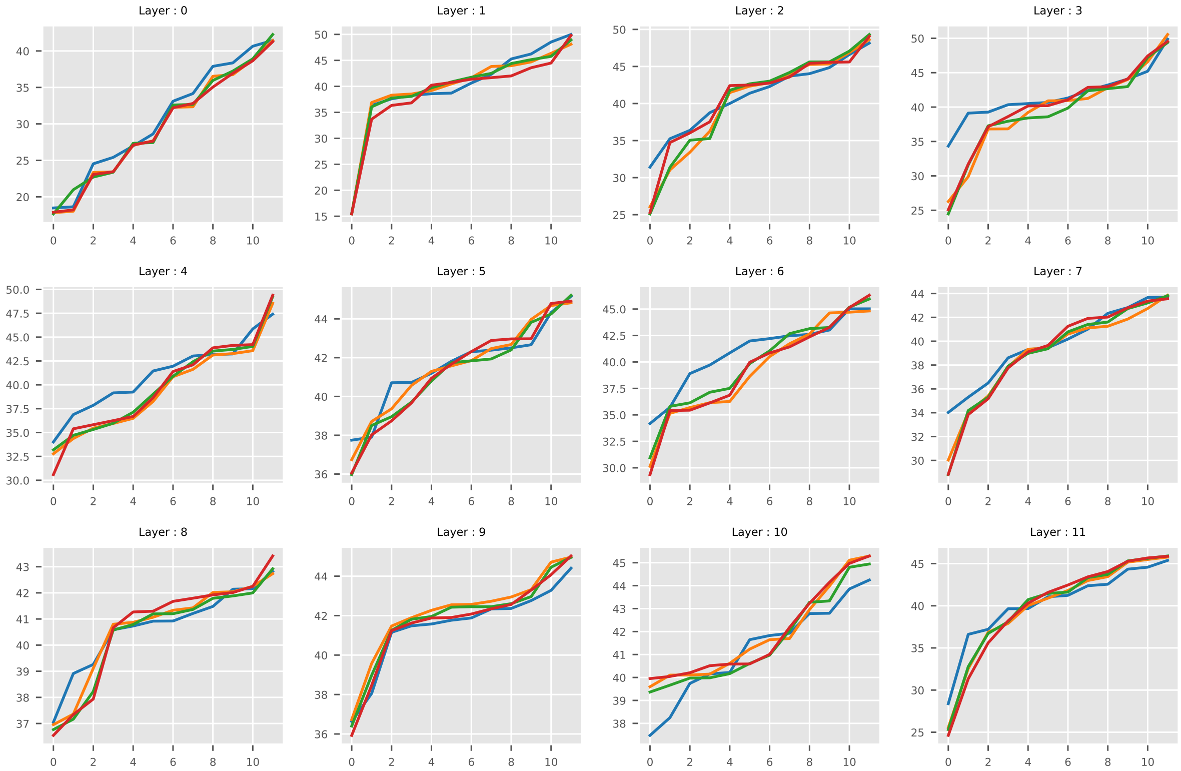

Figure 15: Attention globality distributions of RoBERTa (base) across different heads (sorted according to value) and averaged over all layers and 5000 data points. Blue curve stands for the no phase shift condition, and orange, green and red curves represent $k = 100, 200$ and 300 respectively.

Name	Release Year	Positional Encoding Type
BERT (Devlin et al., 2019)	2019	Learned Absolute
RoBERTa (Liu et al., 2019)	2019	Learned Absolute
GPT2 (Radford et al., 2019)	2019	Learned Absolute
BART (Lewis et al., 2020)	2020	Learned Absolute
LongFormer (Beltagy et al., 2020)	2020	Learned Absolute
T5 (Raffel et al., 2020)	2020	Relative Learned Bias
GPT3 (Brown et al., 2020)	2020	Learned Absolute
GPT-Neo (Black et al., 2021)	2021	Learned Absolute
Fairseq-Dense (Artetxe et al., 2021)	2021	Fixed Absolute
ShortFormer (Press et al., 2021)	2021	Fixed Absolute
GPT-J (Wang, 2021)	2021	Rotary
GPT-NeoX (Black et al., 2022)	2022	Rotary
OPT (Zhang et al., 2022)	2022	Learned Absolute
PaLM (Chowdhery et al., 2022)	2022	Rotary

Table 6: Positional encoding of commonly used pretrained language models.

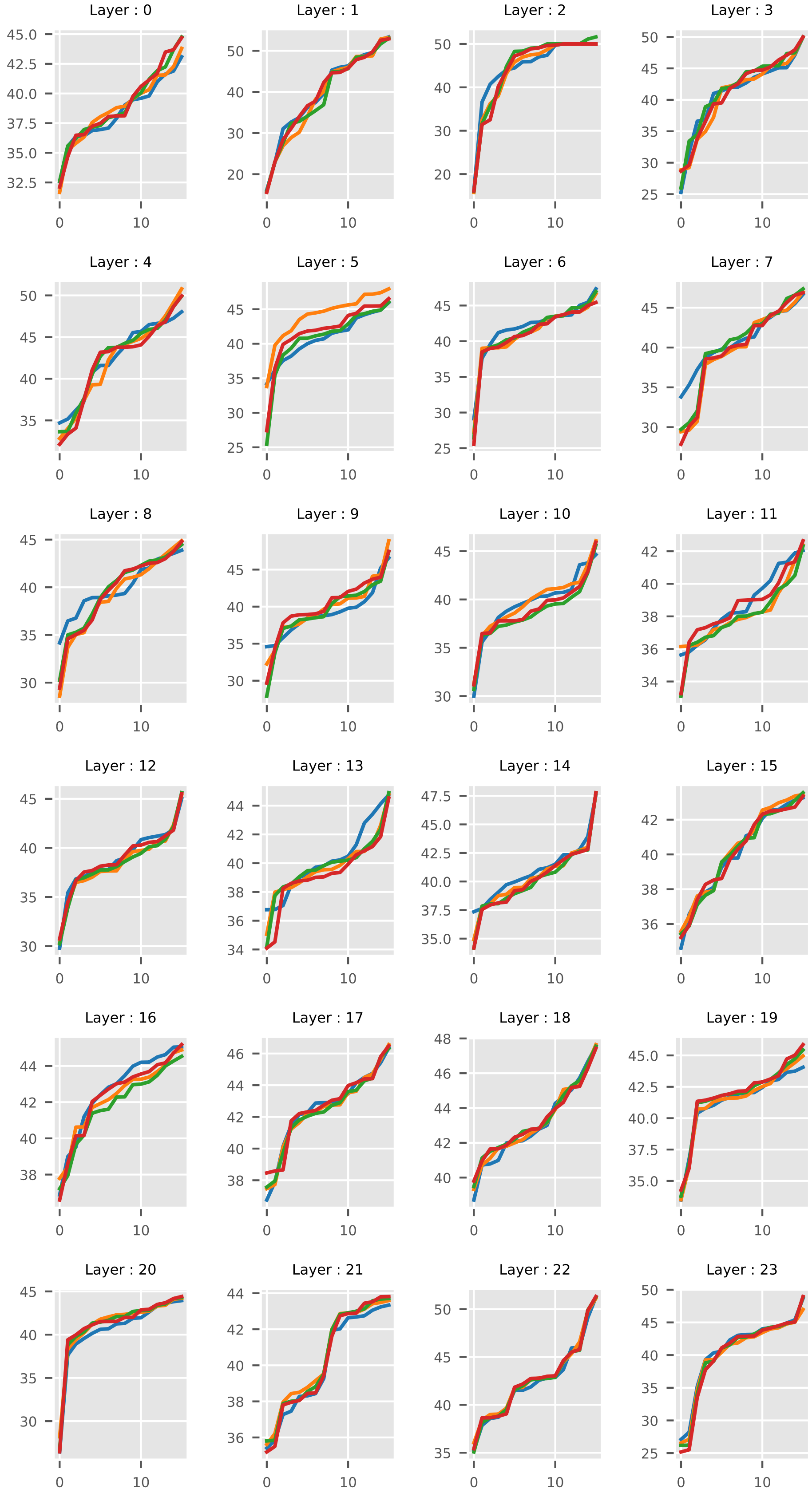

Figure 16: Attention globality distributions of RoBERTa (large) across different heads (sorted according to value) and averaged over all layers and 5000 data points. Blue curve stands for the no phase shift condition, and orange, green and red curves represent $k = 100, 200$ and 300 respectively.

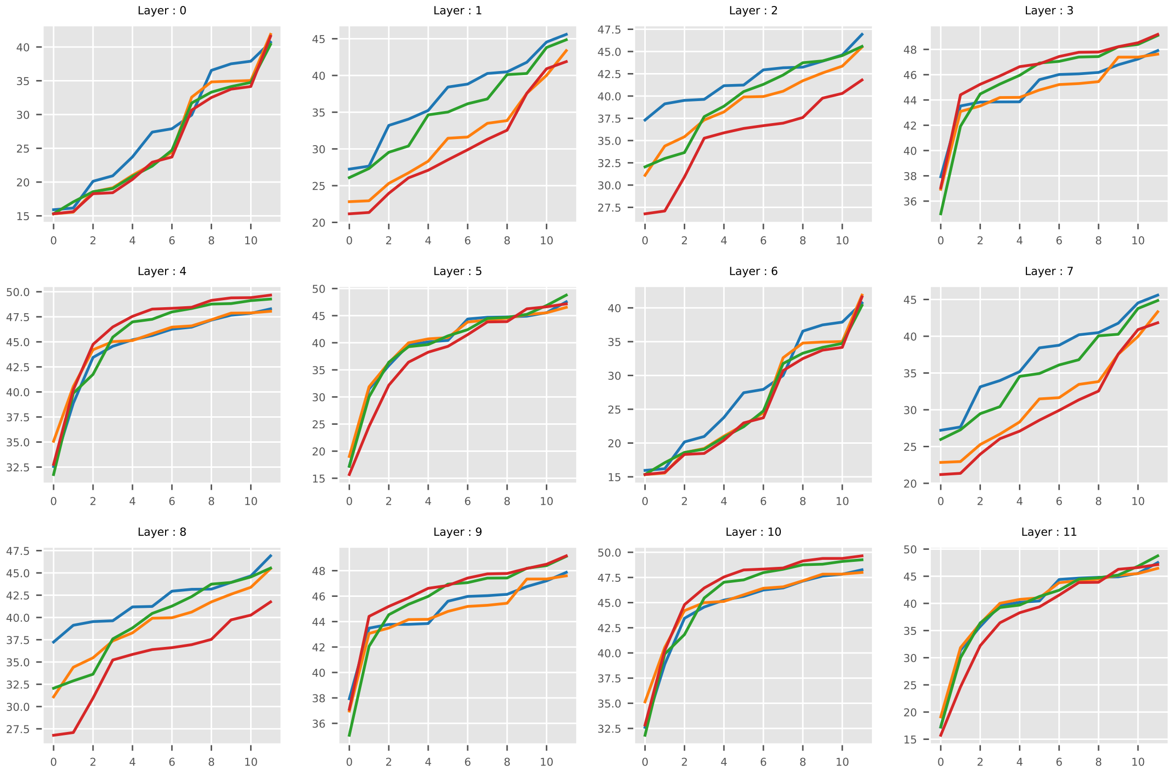

Figure 17: Attention globality distributions of BART (base) across different heads (sorted according to value) and averaged over all layers and 5000 data points. Blue curve stands for the no phase shift condition, and orange, green and red curves represent $k = 100, 200$ and 300 respectively.

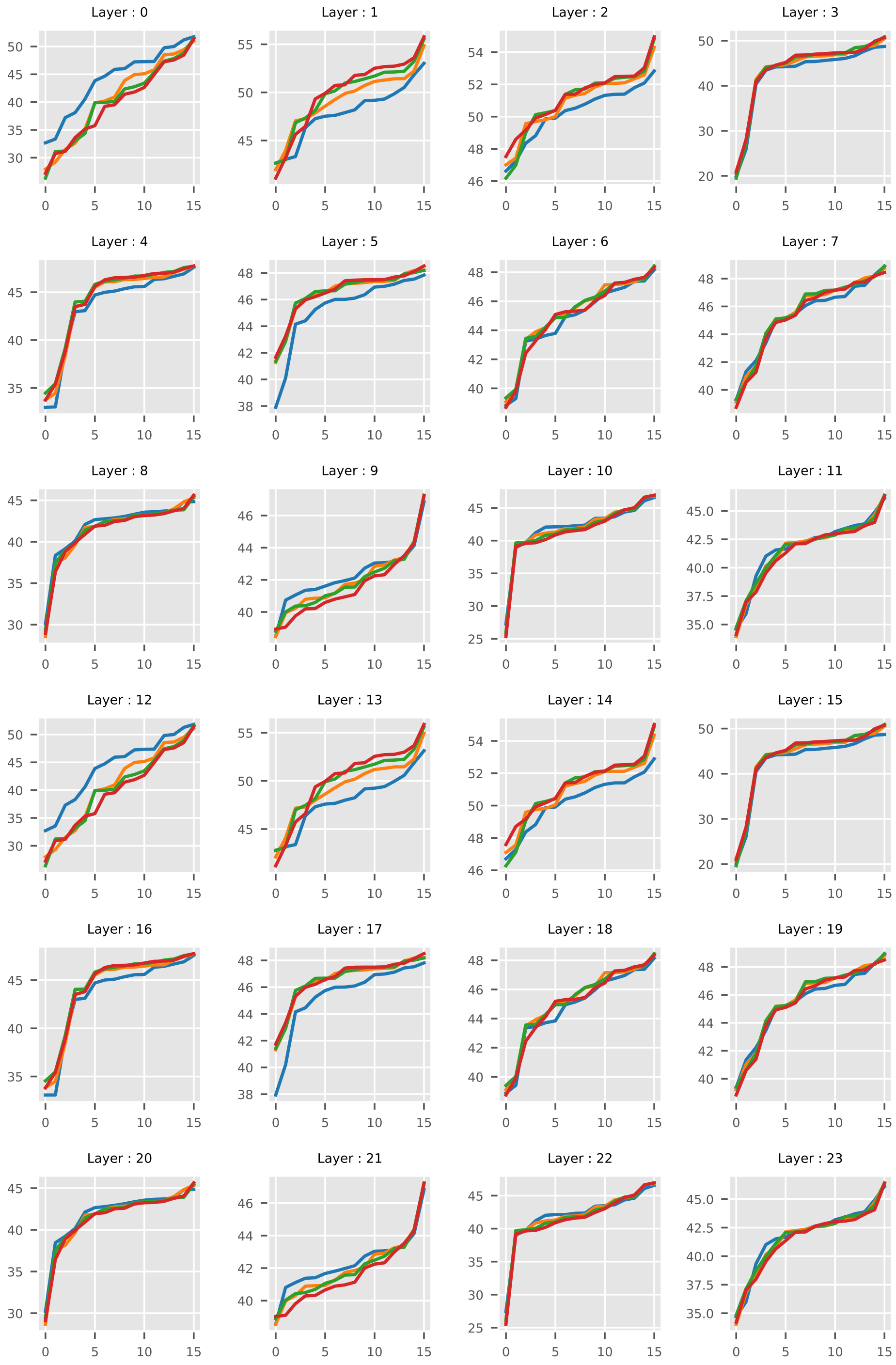

Figure 18: Attention globality distributions of BART (large) across different heads (sorted according to value) and averaged over all layers and 5000 data points. Blue curve stands for the no phase shift condition, and orange, green and red curves represent $k = 100, 200$ and 300 respectively.