Parallel Context Windows for Large Language Models

Nir Ratner   Yoav Levine   Yonatan Belinkov   Ori Ram   Inbal Magar   Omri Abend
Ehud Karpas   Amnon Shashua   Kevin Leyton-Brown   Yoav Shoham
AI21 Labs
`nirr@ai21.com`

Abstract

When applied to processing long text, Large Language Models (LLMs) are limited by their context window. Existing efforts to address this limitation involve training specialized architectures, and cannot be easily applied to off-the-shelf LLMs. We present Parallel Context Windows (PCW), a method that alleviates the context window restriction for any off-the-shelf LLM *without further training*. The key to the approach is to carve a long context into chunks ("windows"), restrict the attention mechanism to apply only within each window, and re-use the positional embeddings across the windows. Our main results test the PCW approach on in-context learning with models that range in size between 750 million and 178 billion parameters, and show substantial improvements for tasks with diverse input and output spaces. We show additional benefits in other settings where long context windows may be beneficial: multi-hop questions and retrieval-augmented question answering with multiple retrieved documents. Our results highlight Parallel Context Windows as a promising method for applying off-the-shelf LLMs in a range of settings that require long text sequences. We make our code publicly available at `https://github.com/ai21labs/parallel-context-windows`.

1 Introduction

A key parameter of a Large Language Model (LLM) is its *context window*, the number of text tokens it can process in a forward pass. Current LLM architectures limit the context window size—typically up to several thousand tokens— because the global nature of the attention mechanism imposes computational costs quadratic in context length. This presents an obstacle to use cases where the LLM needs to process a lot of text, *e.g.*, tackling tasks that require long inputs (Tay et al., 2020; Shaham et al., 2022), considering large sets of retrieved documents for open-book question answering (Karpukhin et al., 2020; Levine et al., 2022a,b),
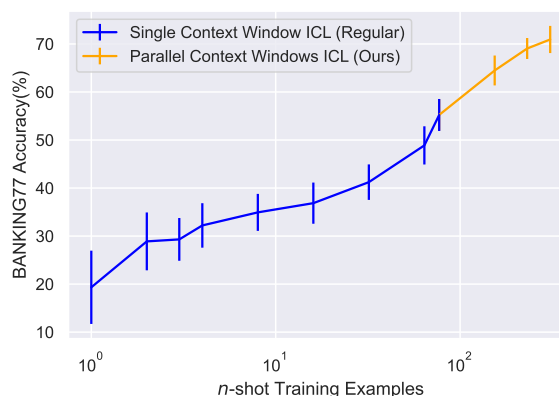

Figure 1: In-context learning (ICL) accuracy against n-shot training examples for the BANKING77 intent classification dataset (Casanueva et al., 2020) using the model Jurassic-1-Grande (17B). The blue line shows the improvement in performance as the context window is filled with examples; the orange line shows how our Parallel Context Windows method, which adds up to four times more training examples, provides a significant boost in performance. The error bars represent the standard deviation across multiple runs, as explained in Section 3.1.

or performing in-context learning (Brown et al., 2020) when the desired input-output relationship cannot be adequately characterized within the context window.

Previous work has addressed such obstacles by training dedicated architectures, *e.g.*, training sparse attention mechanisms for long inputs (Zaheer et al., 2020; Guo et al., 2021) and Fusion-in-Decoder readers for retrieved documents (Izacard and Grave, 2020). However, these architectures are often tailored to specific use cases, and they are often constrained in terms of their size as a trade-off, in order to facilitate long text consumption. It remains an open problem to find an effective way to allow off-the-shelf LLMs to process text longer than its original context window, *without dedicated training*.

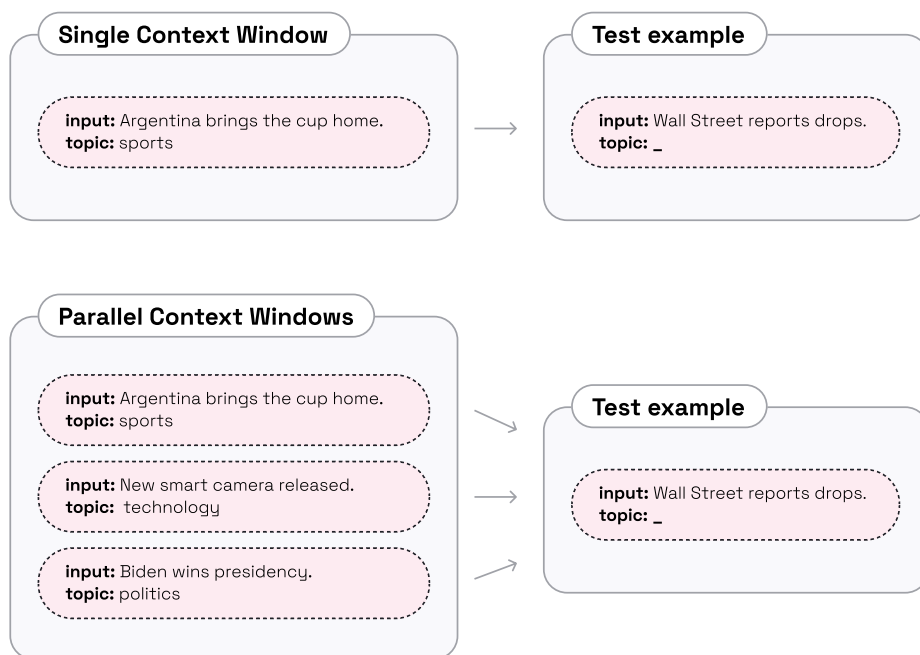In this paper, we introduce *Parallel Context Win-*

6383

Figure 2: An illustration of Parallel Context Windows (PCW) approach, exposing the LLM to text within multiple context windows during generation. Tokens inside each window attend only to the previous tokens *in their window*. Test example tokens attend to the tokens of all context windows.

dows (PCW), illustrated in Figure 2, a new approach for addressing this problem in any decoder-based LLM[1], and show its efficacy in several setups. PCW involves splitting long text into multiple parallel contexts, each equally accessible during output generation. Doing so consists of two simple *post-hoc* modifications to a pretrained LLM, neither of which requires any further training: (1) using sparse masking to allow each context window to attend only to itself, while still allowing the generated text to attend to all contexts simultaneously; and (2) reusing the model's learned positional embeddings within each parallel context window, sidestepping the problem of extrapolating positional embeddings and signaling to the model that each window is equally "close" to the generated tokens.

We conducted an in-depth investigation of the extent to which Parallel Context Windows can improve LLMs' ability to perform *in-context learning* (Brown et al., 2020): when a pretrained LLM is given an input sequence of concatenated "training" input–output pairs representing a task, followed by

a single "test" input, it is able to supply the corresponding test output with high accuracy. Crucially, in the setting of in-context learning, the context window limitation inherently caps the number of training examples that can be inserted before the test example. This significantly limits the applicability of in-context learning for tasks with long or highly diverse inputs or outputs.

We focus on these types of tasks, showing that Parallel Context Windows significantly aid in-context learning of two task families that tend to suffer from low in-context learning performance: classification tasks that have many classes and extractive question answering tasks. We experiment with Jurassic-1 models (Lieber et al., 2021) having between 7B and 178B parameters and GPT2 models (Radford et al., 2019) having between 750M and 1.5B parameters. Notably, using 3 Parallel Context Windows leads to average performance gains of 6.7, 7.3, and 7.9 points in the in-context learning scores of classification tasks with over 5 classes for Jurassic-1 models of sizes 7B, 17B, and 178B, respectively (see example in Figure 1). Our results show that Parallel Context Windows broadens the scope of tasks that can be learned via the
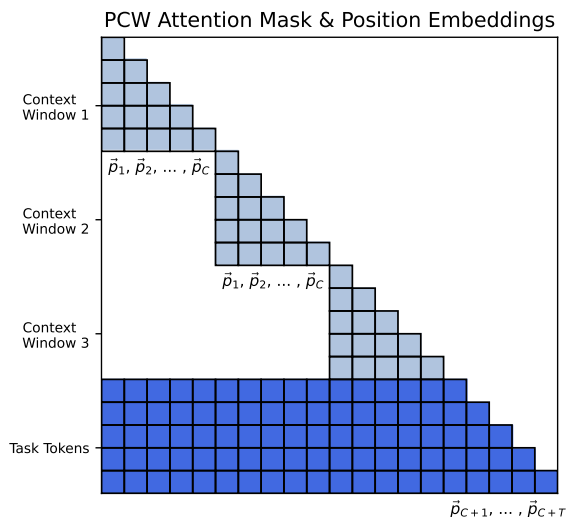
---

[1]We will use LLM to refer to decoder-only language models.

Figure 3: Attention pattern and positional embeddings assignment in PCW. The (i, j) cell in the matrix is colored iff the ith token can attend to the jth token. Each context window (in grey) attends to itself and is assigned positional embeddings $(\vec{p}_i)$ independently, thus re-using the positional vectors. Task tokens (in blue) attend to all the windows. PCW makes the attention matrix sparser, effectively parallelizing the processing of multiple windows.

popular setup of in-context learning, to tasks that require more training examples than permitted in current context sizes.

We further explore the applicability of PCW to two other settings that may benefit from the integration of several documents. One is multi-hop question answering, where the different pieces of information are shown in different windows. We show that in some cases parallel reading is beneficial, through a test case on the HotpotQA benchmark (Yang et al., 2018). The other setting is retrieval-augmented question answering, where we show that reading several retrieved documents in parallel is advantageous, through a test case on the Natural Questions benchmark (Kwiatkowski et al., 2019).

Overall, we provide clear evidence that, without any further training, Parallel Context Windows can make a large amount of text accessible to an off-the-shelf LLM during decoding. We thus see promise in further investigation of Parallel Context Windows for applying off-the-shelf LLMs in other applications that require such capabilities, such as tackling tasks with long inputs.

2 Parallel Context Windows

This section provides the details of our Parallel Context Windows method. The high-level idea of PCW is to insert a long input sequence into multiple replicas of the LLM's original context window, and to allow for a small amount of tokens at the end of the sequence to attend to all of the context windows simultaneously. We design PCW so that the modifications made to the off-the-shelf LLM are minimal, such that processing long contexts remains effective even without further training of the LLM. A side advantage is that the LLM modifications required for PCW are quite simple to implement. Specifically, PCW applies two modifications to two mechanisms in common autoregressive LLMs: the positional embeddings (Section 2.1) and the attention mask (Section 2.2). Figure 3 illustrates both changes.

2.1 Positional Embeddings Modification

Denoting the LLM's original context window size by N and the Transformer's input representation dimension by d, Transformer-based LLMs receive information regarding the input text ordering via a set of N positional embeddings $\{\vec{p}_i \in \mathbb{R}^d\}_{i=1}^N$, by adding $\vec{p}_i$ to the input token embedding in position i.

We conceptually divide the tokens at the input of the LLM into *context tokens* and *task tokens*. The context tokens are inputs that assist the LLM with a given task, such as in-context examples, or relevant retrieved documents. Task tokens refer to the input of the test example, *e.g.*, a sentence to be classified or a question.

When considering a task that requires T task tokens to formulate, the fact that there are only N trained positional embeddings implies that effectively only $C = N - T$ input tokens can be processed as context.[2] In order to implement PCW, we expand the number of processable context tokens by a factor of B such that the overall input sequence can include $B \cdot C + T$ tokens. In order to allow LLMs to process this long sequence of text, we assign one of N learned positional embedding vectors to location $i \in \{1, \ldots, B \cdot C + T\}$ by the

---

[2]In the case of absolute positional embeddings this is a hard restriction; for relative positional embeddings, processing more tokens entails degradation (Press et al., 2021).

following mapping (depicted in Figure 3):

$$\vec{p}_i^{PCW} = \begin{cases} \vec{p}_{(i-1 \mod C)+1} & 1 \leq i \leq BC \\ \vec{p}_{i-(B-1)C} & BC < i \leq BC + T \end{cases} \quad (1)$$

In words, via this mapping, the model effectively identifies B replicas of the first C original positional embeddings, and the T task tokens retain the last T positional embeddings, now seeing these B replicas as context in their near past. We refer to these replicas of the positional embeddings as *context window replicas*. Notably, while the above re-use of the positional embeddings assigns meaningful positions to all tokens within the longer input sequence, the memory cost of this expansion is quadratic, and moreover, the model was not trained to have two tokens in the same position attend to each other. To address these, we describe below a modification to the LLM's attention mechanism.

2.2 Attention Mask Modification

We impose a restriction on the attention mechanism which implies that tokens within each context window replica perform autoregressive attention to other tokens in their context window replica, and do not attend to tokens in other context window replicas. In contrast, the task tokens attend to context tokens within all context window replicas.

In the above setting of context window size N, we represent attention restrictions by attention mask scores $a_{ii'} \in \{0, 1\}$ for $i, i' \in [N] := \{1, \ldots, N\}$. If $a_{ii'} = 0$ then for any Transformer layer in the LLM, tokens in input location i cannot attend to tokens in input location i', and if $a_{ii'} = 1$ they can. In common autoregressive LLMs, a token can only attend to tokens that precede it, which following the above notation is translated into $a_{ii'} = 1$ if $1 \leq i' \leq i \leq N$ and $a_{ii'} = 0$ otherwise.

For the case of PCW, the B parallel context windows include tokens in positions $i \in [C]$, and are identified with an index $b \in [B]$. The T task tokens are not parallelized, and are located in positions $i \in \{C+1, \ldots, C+T = N\}$. For completeness of the notation, we will assign a dummy context window index $b = B+1$ to the T task tokens. We add a second index to the attention scores: $a_{ii'}^{bb'} \in \{0, 1\}$ for $i, i' \in [N]$ and $b, b' \in [B]$. Similarly to the above, if $a_{ii'}^{b,b'} = 0$ then for any Transformer layer in the LLM, tokens in input location i and context window b cannot attend to tokens in input location i' and context window b', and if $a_{ii'}^{b,b'} = 1$ they can.

With the above notation in place, the following restriction implies that context tokens perform autoregressive attention within each context window replica (illustrated in Figure 3):

$$a_{ii'}^{b,b'} = \begin{cases} 1, & \text{if } 1 \leq i' \leq i \leq C \text{ and } b = b' \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

The following implies that the T task tokens attend to all tokens in all B context windows (for $i > C$):

$$a_{ii'}^{B+1,b'} = \begin{cases} 1, & \text{if } 1 \leq i' \leq i \leq N, \ b' \in [B+1] \\ 0, & \text{otherwise} \end{cases} \quad (3)$$

The above attention masks allow the model to attend to B times more context when decoding the output, while keeping the computational cost linear in the number of parallel contexts B. Overall, for both the above PCW modifications, assigning $B = 1$ corresponds to the vanilla LLM mechanism.

3 PCW for In-Context Learning

3.1 Experimental Setup

We apply the PCW method in the setting of in-context learning (ICL): we distribute the in-context training examples among the multiple context window replicas, thus allowing the test example to attend to more training examples. For each experiment, we report the performance with regular ICL, using the maximum number of examples that fit in a model's context window (n_{max}). For our PCW method, given B parallel windows, we effectively use $B \times n_{max}$ training examples. The n_{max} used for each dataset and model can be found in Table 9. Unless stated otherwise, we report results with $B = 3$ in the main paper, and discuss the choice of B in Appendix C. Since training examples vary in length, we allocate in-context examples into the parallel windows in a manner that balances the windows' lengths.[3] The test example (corresponding to the T task tokens in Section 2) receives the positional embedding that immediately follows the longest context window.

Training and test sets The performance of in-context learning was shown to significantly vary with the choice of training examples (Zhao et al., 2021). We followed past work (Zhao et al., 2021;

---

[3]Within each window, positional embeddings are assigned sequentially starting from 1. See Appendix A for a discussion.

Lu et al., 2021), randomly sampling 30 sets of training examples from the full training set. We report the mean and standard deviation of performance metrics across these samples. When comparing PCW method with standard ICL, statistically significant differences according to a t-test (p-value < 0.05) are marked with [*]. To allow for an extensive set of experiments, we followed prior work and randomly subsampled the test sets to contain at most 250 examples (Zhao et al., 2021; Lu et al., 2021; Han et al., 2022).

Models We experiment with 5 LMs of varying sizes: GPT2-Large (0.75B parameters) and GPT2-XL (1.5B) (Radford et al., 2019); and three Juarassic-1 (J1) models (Lieber et al., 2021): Large (7.5B), Grande (17B), and Jumbo (178B). Due to its massive size, we reduced the number of sampled training sets and the test set size for J1-Jumbo to 15 and 125, respectively.

Datasets Our main focus is classification, and we experiment with 15 different datasets in this category, listed in Appendix B. Many of these datasets are used in prior work on in-context learning (Zhao et al., 2021; Lu et al., 2021; Han et al., 2022). We additionally experiment with several datasets with a high number of output classes (up to 150), to examine how well our approach works in this setting. To classify an example in the in-context learning setup, we assign the label using restrictive greedy decoding (see Appendix A). We also experiment with another type of tasks, information extraction, and test 4 datasets with a subset of the models (J1-Large and J1-Grande). For these tasks we use greedy decoding at temperature 0 (as in Zhao et al. (2021)). For further information about the decoding and formats used for the different types of datasets, see Appendices A and B.

3.2 Classification Tasks Results

PCW enables in-context learning with a large number of classes. Table 1 shows the results on various classification tasks, organized by the number of classes. With a small number of output classes (≤ 5), we find small or insignificant differences between PCW and vanilla ICL on J1-Large (7.5B), while with J1-Grande (17B) and J1-Jumbo (178B), PCW is superior in the majority of cases. However, many of these differences are not statistically significant.

Our PCW method shines in classification tasks with a large number of output classes. With more
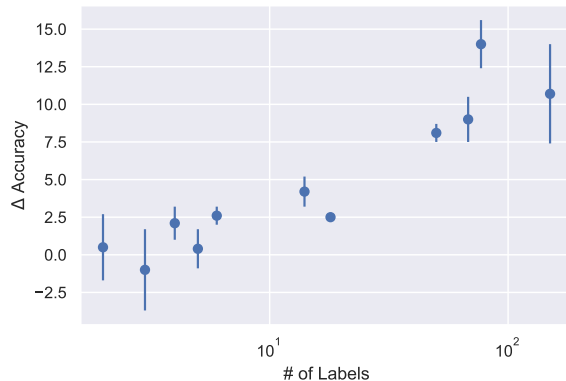

Figure 4: Average gains of PCW vs. the # of labels. Each data point represents the average gain across all datasets and J1 models. There is a a strong positive correlation between the number of unique labels and the gains from PCW.

than 5 classes, PCW statistically significantly outperforms ICL in nearly all models and datasets. The average improvement across these datasets is 6.7, 7.3, and 7.9 for J1-Large, J1-Grande, and J1-Jumbo. Evidently, *the larger the model, the greater the benefit from our method*. This positive scaling behavior of PCW stands in contrast to prior work attempting to improve ICL (Zhao et al., 2021; Lu et al., 2021; Han et al., 2022), where improvements to 178B-scale models were smaller than improvements observed in smaller models.

In Table 5 (Appendix D.1), we report results with GPT-2 models. Although they are smaller than J1 models, we find consistent statistically significant improvements with GPT2-XL (1.5B parameters) in almost all datasets. With GPT2-Large (0.75B), we find improvements in the majority of datasets.

PCW improves with more classes. To examine the relation between the number of output classes and the performance of PCW, we compute the difference between PCW and ICL in each experiment, and average over all datasets (and models) having the same number of classes. As Figure 4 shows, there is a strong positive correlation between the number of classes and the improvement brought about by PCW (Pearson correlation $r = 0.93$ between the log-number of classes and the average improvement; the slope is 3.02). For datasets with dozens of unique labels—specifically Banking77 (Casanueva et al., 2020), NLU Intent (Xingkun Liu and Rieser, 2019), and CLINIC150 (Larson et al., 2019)—we observe improvements of 10–15 points in most cases. Importantly, prior in-context learn-

		J1- Large (7.5B)		J1-Grande (17B)		J1-Jumbo (178B)	
Dataset	# Labels	ICL	PCW	ICL	PCW	ICL	PCW
SST-2	2	$93.5_{1.6}$	$\mathbf{93.8_{1.1}}$	$95.2_{1.1}$	$\mathbf{95.6_{0.5}}$	$96.5_{1.4}$	$\mathbf{97.0_{1.5}}$
CR	2	$\mathbf{93.9_{0.7}}$	$\mathbf{93.9_{0.7}}$	$93.6_{0.8}$	$\mathbf{93.8_{0.8}}$	$\mathbf{93.6_{1.5}}$	$93.1_{1.0}$
RTE	2	$\mathbf{58.3_{3.8}}$	$58.1_{3.7}$	$61.2_{5.1}$	$\mathbf{62.2_{3.0}}$	$63.9_{5.0}$	$\mathbf{66.0_{4.1}}$
Subj	2	$\mathbf{84.1_{7.7}}^{*}$	$79.1_{7.2}$	$93.0_{2.5}$	$\mathbf{95.3_{1.2}}^{*}$	$89.1_{5.3}$	$\mathbf{93.6_{2.1}}^{*}$
CB	3	$\mathbf{65.2_{8.0}}$	$61.2_{8.2}$	$75.0_{8.1}$	$\mathbf{75.7_{6.0}}$	$76.2_{4.3}$	$\mathbf{76.6_{3.5}}$
AGNews	4	$79.8_{3.6}$	$\mathbf{81.5_{2.1}}^{*}$	$81.4_{3.0}$	$\mathbf{82.7_{2.1}}$	$82.5_{3.8}$	$\mathbf{85.9_{1.7}}^{*}$
SST-5	5	$45.5_{3.9}$	$\mathbf{47.4_{2.9}}^{*}$	$51.6_{3.4}$	$\mathbf{53.8_{2.2}}^{*}$	$\mathbf{55.4_{2.8}}$	$55.1_{3.9}$
YELP	5	$56.2_{3.8}$	$\mathbf{56.3_{5.1}}$	$\mathbf{66.2_{2.2}}$	$65.6_{2.0}$	$\mathbf{66.3_{4.1}}$	$65.4_{2.6}$
TREC	6	$87.0_{4.5}$	$\mathbf{89.4_{3.2}}^{*}$	$86.5_{3.8}$	$\mathbf{88.7_{3.4}}^{*}$	$87.1_{5.7}$	$\mathbf{90.4_{3.1}}$
DBPedia	14	$93.2_{3.0}$	$\mathbf{96.2_{1.5}}^{*}$	$92.5_{3.3}$	$\mathbf{97.3_{1.6}}^{*}$	$91.7_{4.4}$	$\mathbf{96.5_{2.3}}^{*}$
NLU Scenario	18	$81.9_{2.2}$	$\mathbf{84.2_{1.5}}^{*}$	$86.1_{2.1}$	$\mathbf{88.8_{1.1}}^{*}$	$85.4_{2.9}$	$\mathbf{87.8_{1.6}}^{*}$
TREC Fine	50	$60.5_{6.9}$	$\mathbf{68.8_{3.4}}^{*}$	$63.3_{6.0}$	$\mathbf{71.8_{4.6}}^{*}$	$71.4_{5.7}$	$\mathbf{78.7_{3.6}}^{*}$
NLU Intent	68	$69.7_{3.3}$	$\mathbf{79.7_{1.9}}^{*}$	$72.1_{3.1}$	$\mathbf{81.9_{1.6}}^{*}$	$74.3_{3.4}$	$\mathbf{81.6_{2.9}}^{*}$
BANKING77	77	$51.0_{3.4}$	$\mathbf{63.5_{2.7}}^{*}$	$55.2_{3.3}$	$\mathbf{69.1_{2.2}}^{*}$	$55.3_{3.5}$	$\mathbf{70.9_{3.1}}^{*}$
CLINIC150	150	$67.3_{2.7}$	$\mathbf{75.4_{1.7}}^{*}$	$68.9_{2.5}$	$\mathbf{78.6_{1.8}}^{*}$	$65.7_{5.0}$	$\mathbf{79.9_{2.1}}^{*}$

Table 1: Accuracy results (in %) for J1-Large, J1-Grande, and J1-Jumbo models with regular ICL in comparison with using PCW with $B = 3$ prompts. The best results for each model and dataset are boldfaced, and '*' is used to indicate that the boldfaced result is statistically better (t-test with p-value < 0.05).

ing work has not considered datasets with such a larger number of classes, perhaps due to the standard limitation of the context window size.[4] We note that in GPT-2 models (Table 5, Appendix D.1) we do not see a significant correlation between PCW improvements and the number of classes, but these smaller models tend to struggle with very large numbers of classes.

Comparing results for datasets with different numbers of output classes may be confounded by other factors, such as differences in domain, style, or genre. To isolate such effects, we compare results with two datasets, each having both fine-grained and coarse-grained labels: (1) The TREC dataset (Li and Roth, 2002), which has 6 coarse-grained and 50 fine-grained classes. (2) NLU (Xingkun Liu and Rieser, 2019),[5] which has 18 scenarios and 68 intents. From Table 1, we see that PCW outperforms standard ICL by 2.6 and 8.1 points on TREC coarse-grained and fine-grained classification, respectively. Similarly, on NLU coarse- and fine-grained classification, we see average improvements of 2.5 and 9.0 points, respectively. We conclude that our approach shines especially well when dealing with a large number of output classes.

PCW makes in-context learning more stable. A known limitation of in-context learning is high variance across examples and sensitivity to aspects like the order of examples (Lu et al., 2021). Encouragingly, we find that PCW reduces such variance: We observe average std values of 3.1, 2.3, and 2.6 for J1-Large, J1-Grande, and J1-Jumbo with PCW, compared to 3.9, 3.4, and 3.9 in standard ICL.

3.3 Information Extraction Results

Table 2 shows the results of ICL and PCW on information extraction datasets with tasks like airline name extraction or extractive question answering. These tasks can be considered as classification tasks with an extremely large number of classes, potentially the entire vocabulary or phrases from the vocabulary. Our approach consistently improves results with both J1-Large and J1-Grande, resulting in statistically significant improvements in almost all cases. We also observe smaller standard deviations with PCW compared to ICL.

It is worth noting that prior work has not experimented much with information extraction in an in-context learning setting. Zhao et al. (2021) reported results with several datasets, but not with extractive question-answering. Our approach seems to allow in-context learning in such cases as well.

Finally, we tested two multiple-choice QA tasks: OpenBookQA (Mihaylov et al., 2018) and StoryCloze (Mostafazadeh et al., 2017). With our larger model, J1-Grande, PCW leads to a signifi-

Dataset	J1-Large (7.5B)		J1-Grande (17B)	
	ICL	PCW	ICL	PCW
ATIS	$85.6_{5.3}$	$\mathbf{89.0_{3.0}}^{*}$	$88.0_{4.6}$	$\mathbf{91.7_{3.1}}^{*}$
MIT Movies	$67.9_{2.7}$	$\mathbf{70.3_{2.5}}^{*}$	$69.0_{3.9}$	$\mathbf{69.3_{3.3}}$
SQuAD	$79.2_{2.1}$	$\mathbf{80.5_{1.4}}^{*}$	$83.8_{2.5}$	$\mathbf{85.1_{1.4}}^{*}$
adversarialQA	$43.0_{2.2}$	$\mathbf{44.6_{1.5}}^{*}$	$46.4_{2.0}$	$\mathbf{47.4_{1.8}}$

Table 2: PCW improves information extraction (ATIS and MIT Movies are measured with EM, SQuAd and adversarialQA with F1).

cant improvement in OpenBookQA and does not significantly improve or worsen over ICL in other cases. Details and results of the experiment can be found in Appendix D.2.

4 PCW for Question Answering

In this section, we explore potential usages of PCW in other settings than in-context learning. Specifically, we examined two question-answering settings where PCW is expected to help aggregate information from multiple texts. Firstly, we consider the case of question answering based on retrieved documents. Secondly, we experiment with multi-hop reasoning, where the model is required to utilize more than one text while answering a question. Importantly, while in Section 3 the parallel context windows were used for processing training examples for ICL, in this section the windows are used for parallel processing of documents related to the test example.

4.1 Retrieval Based Question Answering

Setup We first experiment with Natural Questions (NQ, Kwiatkowski et al., 2019) in an open-book question-answering retrieval setting: Given a question and a set of candidate documents, that may or may not contain the evidence for the question, a model needs to generate a free-text answer.

In the single context window setting (the baseline), we followed the few-shot setup defined by Lazaridou et al. (2022): For each question, we retrieved evidence documents from Wikipedia, using a BM25 sparse retriever (Robertson et al., 2009). We then prompted the model with in-context training examples of the related task of extracting the answer from a gold evidence document, and concatenated the test question and $N \in \{1, 2, 4, 6, 8, 10\}$ evidence documents[6]. To fully utilize the context

---
[6]Notably, Lazaridou et al. (2022) used this apparatus only with $N = 1$, while we experiment with different values of N.
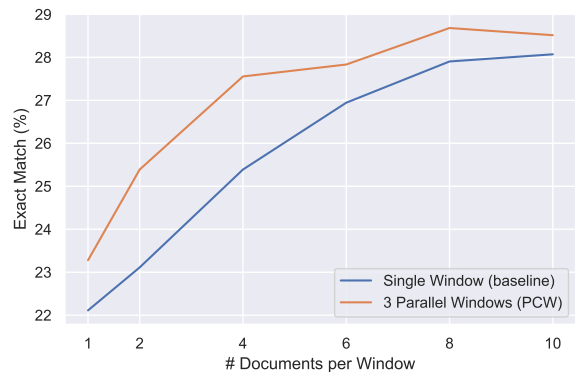

Figure 5: NQ results for J1 Grande—EM against number of documents in a single window.

Type	J1-Large (7.5B)		J1-Grande (17B)	
	Seq	PCW	Seq	PCW
Comparison	15.3	**21.5**	20.9	**28.7**
Bridge	**21.6**	16.5	**27.1**	24.0

Table 3: Zero-shot Exact Match (EM) results on Hot-potQA with Sequential and Parallel processing (PCW approach).

window size, we "padded" the prompt with as much in-context training examples as possible.

For PCW, we followed the setup of a single window while taking advantage of the method's natural ability of parallelization: We increased the number of retrieved documents per question, and divided them between windows. *E.g.*, for $N = 1$ and 3 parallel context windows ($B = 3$), PCW processes $B \times N = 3$ retrieved documents (1 per each window), thus effectively increasing the chance that the correct answer span will be shown to the model in one of the retrieved documents. The metric we used was the standard Exact Match (EM). We refer to Appendix A for more details.

Results Figure 5 shows the results for J1-Grande, when using PCW compared to the baseline, as a function of the number of candidate documents in a single window. In all cases, PCW performs better than the baseline, demonstrating the benefit of parallel processing of candidate documents. As we increase the number of available retrieved documents, we see an increase in performance for both approaches. Similar trend can be seen for J1-Large (see Figure 6 in Appendix). Naturally, the performance of this task depends on the probability of retrieving the correct answer. The latter increases in PCW setting, when the number of processed documents is multiplied by $B = 3$.

4.2 Multi-hop Question Answering

Setup Finally, we experiment with HotpotQA (Yang et al., 2018), which requires multi-hop reasoning. Given a question and 10 evidence documents (2 gold and 8 distractors), answering the question requires reasoning over both gold documents. HotpotQA includes two question types[7]: (a) Questions that refer to a *bridge* entity. For example, to answer the question "when was the singer of Radiohead born?", one needs to reason that the singer is "Thom Yorke" (the *bridge* entity) and then find his birthday. (b) Questions that rely on a *comparison* between two entities. For example: "Who has played for more NBA teams, Michael Jordan or Kobe Bryant?". As a baseline, we provide all of the evidences in a random, sequential manner. For PCW, we use 5 windows, with 2 evidences in each window. Since the 10 evidences filled most of the context window of J1 models, we work in a zero-shot setting. The evaluation metric is the standard Exact Match (EM).

Results Table 3 shows the results. We break down the results according to the bridge and comparison question types. Interestingly, PCW helps with comparison questions, improving performance over the baseline in both J1-Large and J1-Grande while degrading the performance on bridge questions. This disparate behavior can be explained by the kind of processing required to answer the two types of questions. In comparison questions, the model can extract the necessary information from the two gold texts independently, making them suitable for PCW. For example, to know who played for more NBA teams, the LM needs to extract the number of NBA teams Jordan played for from one text, while extracting the number of NBA teams Bryant played for from another independent text. In contrast, to answer a bridge question, the processing of each text is conditioned on the other text: When reading a sentence about Thom Yorke's birthplace, we already need to know that Yorke is the Radiohead singer, if we wish to then be able to answer the above question. This makes PCW less suitable for these types of tasks in its current form, and we leave it as an open direction for how to encode sequential relation between windows (perhaps by some further training).

---

[7]Examples of questions taken from Yang et al. (2018)

5 Related Work

5.1 In-Context Learning

In-context learning has been the subject of extensive research since it was first introduced by Brown et al. (2020). For instance, Zhao et al. (2021) showed that LMs are often miscalibrated. Zhao et al. (2021) and Han et al. (2022) explored ways to overcome this issue by different calibration methods. Lu et al. (2021) observed that few-shot performance varies significantly depending on the order of examples in the prompt, and proposed a protocol for finding better permutations. Min et al. (2021) proposed a noisy channel approach to boost few-shot performance. Our framework is orthogonal and thus complementary to these methods, as we are mainly focused on how to increase the number of examples shown to the model. Our approach is also more general as it seamlessly supports generative tasks as well.

5.2 Expanding the Context Window

The issue of a limited context window has been the focus of many studies that tried to alleviate the memory footprint of self-attention. One line of work (Zaheer et al., 2020; Guo et al., 2021, *inter alia*) suggested using sparse attention to overcome this difficulty. Press et al. (2022) proposed to encode positional information via relative factors added to attention weights, instead of absolute positional encoding. Despite the impressive extrapolation abilities of Press et al. (2022), the self-attention cost of such models remains quadratic, making inference for longer prompts slow and expensive. Ivgi et al. (2022) suggest SLED, an encoder–decoder model for long texts, which encodes short overlapping chunks of the input text, and fuses the information in the decoder, *a-la* Fusion-in-Decoder (Izacard and Grave, 2020). Similarly to our approach, both Izacard and Grave (2020) and Ivgi et al. (2022) employ off-the-shelf architectures, but those methods require further training. Among all mentioned methods, our work is the first that utilizes existing LLMs for longer inputs *without any further training*.

In concurrent work, Hao et al. (2022) suggest using multiple context windows, while scaling the context tokens' attention weights. We show that large gains can be made without scaling the attention weights, and we demonstrate particularly large gains for tasks with diverse output spaces. Moreover, they focus on LLMs with non-learned

positional encoding (sinusoidal, Vaswani et al. 2017 and ALIBI, Press et al. 2022) and only show results in the ICL setting. In contrast, we show that PCW is effective for more common LLMs that have learned positional embeddings, and show that PCW obtains gains both in ICL and in document retrieval settings.

6 Conclusion and Future Work

In recent years, a multitude of successful approaches have been proposed for allowing Transformer-based language models to leverage large amounts of text during inference, leading to a variety of dedicated architectures. In parallel, however, the mainstream LLM production line of new models with "regular"—up to several thousand tokens—context window sizes enjoys faster progress in the form of scaling, innovation, and data updating.

This paper introduced *Parallel Context Windows* (PCW): A simple approach for allowing any off-the-shelf LLM to broaden the scope of text it can access during inference. We showed the effectiveness of PCW in the framework of in-context learning, where access to a context that is larger by a factor of B implies learning from B times more training examples. Our results show that PCW is more effective than the vanilla single context window approach for in-context learning over a broad set of multi-class classification tasks, suggesting that PCW could improve in-context learning in tasks with diverse input or output spaces. We also showed promising signals for applying PCW for multiple retrieved document reading.

Two key directions of future work strike us as particularly promising. First, by demonstrating that an off-the-shelf LLM can attend to substantially larger quantities of text via PCW, our results motivate further investigation of the PCW method in other settings in which it would be desirable to apply mainstream LLMs over long text sequences. Second, though our results suggest that PCW is effective without further training, we believe that further (short) training of an LLM with parallel context windows could further enhance the abilities demonstrated in this work.

Limitations

We presented Parallel Context Windows (PCW), a simple approach that alleviates context window restrictions for any off-the-shelf LLM, without ad-ditional training. We showed the potential of this method on a variety of models and datasets. With that, our method does have some limitations.

The number of context windows has a limit, and needs to be predetermined. Similarly to vanilla in-context learning, the number of examples to include in the prompt must be selected beforehand. For PCW, it is also required to select the number of context windows, B. In this paper, most of the results are for $B = 3$. We experiment in Appendix C with the choice of B. The results are task dependent, but at a high level we find that there are diminishing returns around B in the range of 5 to 7. We leave further investigation of how to effectively benefit from more windows for future work.

Not effective for all types of tasks. As discussed in Section 3, PCW shows impressive gains in ICL for tasks such as multi-class tasks classification as well as information extraction. However, for some tasks, PCW does not improve performance. This might indicate that some tasks are not suited for parallel processing. Section 4.2 demonstrated that PCW is more suitable for cases where the input text could be divided into few independent inputs, but it remains an open question as to whether tasks, such as long text generation, would benefit from PCW.

7 Acknowledgements

References

Neel Alex, Eli Lifland, Lewis Tunstall, Abhishek Thakur, Pegah Maham, C. Jess Riedel, Emmie Hine, Carolyn Ashurst, Paul Sedille, Alexis Carlier, Michael Noetel, and Andreas Stuhlmüller. 2021. Raft: A real-world few-shot text classification benchmark.

Max Bartolo, Alastair Roberts, Johannes Welbl, Sebastian Riedel, and Pontus Stenetorp. 2020. Beat the ai: Investigating adversarial human annotation for reading comprehension. *Transactions of the Association for Computational Linguistics*, 8:662–678.

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.

Iñigo Casanueva, Tadas Temcinas, Daniela Gerz, Matthew Henderson, and Ivan Vulic. 2020. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on NLP for ConvAI - ACL 2020*. Data available at https://github.com/PolyAI-LDN/task-specific-datasets.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The pascal recognising textual entailment challenge. In *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*, pages 177–190, Berlin, Heidelberg. Springer Berlin Heidelberg.

Marie-Catherine de Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The commitmentbank: Investigating projection in naturally occurring discourse.

Xiaowen Ding, Bing Liu, and Philip Yu. 2008. A holistic lexicon-based approach to opinion mining. pages 231–240.

Mandy Guo, Joshua Ainslie, David Uthus, Santiago Ontanon, Jianmo Ni, Yun-Hsuan Sung, and Yinfei Yang. 2021. Longt5: Efficient text-to-text transformer for long sequences.

Zhixiong Han, Yaru Hao, Li Dong, Yutao Sun, and Furu Wei. 2022. Prototypical calibration for few-shot learning of language models.

Yaru Hao, Yutao Sun, Li Dong, Zhixiong Han, Yuxian Gu, and Furu Wei. 2022. Structured prompting: Scaling in-context learning to 1,000 examples.

Charles T. Hemphill, John J. Godfrey, and George R. Doddington. 1990. The ATIS spoken language systems pilot corpus. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27,1990*.

Maor Ivgi, Uri Shaham, and Jonathan Berant. 2022. Efficient long-text understanding with short-text models.

Gautier Izacard and Edouard Grave. 2020. Leveraging passage retrieval with generative models for open domain question answering. *arXiv preprint arXiv:2007.01282*.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. 2019. An evaluation dataset for intent classification and out-of-scope prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Angeliki Lazaridou, Elena Gribovskaya, Wojciech Stokowiec, and Nikolai Grigorev. 2022. Internet-augmented language models through few-shot prompting for open-domain question answering.

Yoav Levine, Itay Dalmedigos, Ori Ram, Yoel Zeldes, Daniel Jannai, Dor Muhlgay, Yoni Osin, Opher Lieber, Barak Lenz, Shai Shalev-Shwartz, et al. 2022a. Standing on the shoulders of giant frozen language models. *arXiv preprint arXiv:2204.10019*.

Yoav Levine, Ori Ram, Daniel Jannai, Barak Lenz, Shai Shalev-Shwartz, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2022b. Huge frozen language models as readers for open-domain question answering. In *ICML 2022 Workshop on Knowledge Retrieval and Language Models*.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander Rush, and Thomas Wolf. 2021. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Opher Lieber, Or Sharir, Barak Lenz, and Yoav Shoham. 2021. Jurassic-1: Technical details and evaluation. *White Paper. AI21 Labs*.

Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: A Python toolkit for reproducible

information retrieval research with sparse and dense representations. In *Proceedings of the 44th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2021)*, pages 2356–2362.

Jingjing Liu, D. Scott Cyphers, Panupong Pasupat, Ian McGraw, and James R. Glass. 2012. A conversational movie search system based on conditional random fields. In *Interspeech*.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2021. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *EMNLP*.

Sewon Min, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2021. Noisy channel language model prompting for few-shot text classification.

Nasrin Mostafazadeh, Michael Roth, Annie Louis, Nathanael Chambers, and James Allen. 2017. Lsdsem 2017 shared task: The story cloze test. In *Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics*, pages 46–51.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. *CoRR*, cs.CL/0409058.

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. Train short, test long: Attention with linear biases enables input length extrapolation. *arXiv preprint arXiv:2108.12409*.

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

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. *arXiv e-prints*, page arXiv:1606.05250.

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Uri Shaham, Elad Segal, Maor Ivgi, Avia Efrat, Ori Yoran, Adi Haviv, Ankit Gupta, Wenhan Xiong, Mor Geva, Jonathan Berant, et al. 2022. Scrolls: Standardized comparison over long language sequences. *arXiv preprint arXiv:2201.03533*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. 2020. Long range arena: A benchmark for efficient transformers. *arXiv preprint arXiv:2011.04006*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Pawel Swietojanski Xingkun Liu, Arash Eshghi and Verena Rieser. 2019. Benchmarking natural language understanding services for building conversational agents. In *Proceedings of the Tenth International Workshop on Spoken Dialogue Systems Technology (IWSDS)*, pages xxx–xxx, Ortigia, Siracusa (SR), Italy. Springer.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big bird: Transformers for longer sequences.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015a. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015b. Character-level convolutional networks for text classification. In *NIPS*.

Tony Z. Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models.

A Experimental Details

A.1 PCW Implementation Details

Handling context windows of various lengths Section 2 thoroughly describes PCW method for cases where each window has the same number of tokens. Throughout all our experiments, this was rarely the case. We considered two variations of

PCW to handle these cases. The first was whether to use left or right indentation of the windows, meaning whether all of the windows should begin or end in the same position id. To avoid any discontinuity in the assignment of position ids, it is also possible to pad the windows with some dummy tokens (*e.g.*, new line). Left indentation was found to be the most preferred option in ICL setting, while padding did not appear to be significant. For that reason, and considering the simplicity of this solution, we chose to use left indentation in all of our experiments. It is important to note that in the PCW implementation, all the windows and the task tokens attend to a single shared BOS token. We found that having multiple BOS tokens negatively affected our results.

Splitting the inputs into windows For the experiments described in Section 3, we assigned an equal number of n_{max} samples per window, and only attempted to balance the lengths of the windows by greedily switching long and short samples between windows. n_{max} was calculated according to the following formula:

$$n_{max} = \lfloor \frac{N - T_{max}}{D_{90}} \rfloor \quad (4)$$

where N is the context window size, T_{max} is the length of longest test sample and D_{90} is the 90th percentile of the train samples' lengths. To avoid unwanted effects due to outliers, we removed the longest percentile of train and test samples.

In the experiments described in Section 4.1, we divided the documents according to the retriever's ranking, so that the last document in each window would have the highest ranking in the window. It should be noted that the training examples were not parallelized. The same randomly chosen examples were used for both baseline and PCW, and new examples were drawn for each test sample. For the experiment described in Section 4.2, the division between windows was random.

A.2 Evaluation Details

Classification A common way to evaluate models in the in-context learning setup is to iterate over all possible labels for each test sample and check which label receives the highest probability according to the LM. This approach is problematic where a large number of classes is present, especially when some class names are split into multiple tokens. To save computational costs, we

implemented constrained greedy decoding, at each step allowing only tokens that could result in a valid label. It is important to acknowledge that this evaluation method could result in slightly different performance for both the ICL baseline and for the PCW approach. However, since most of the labels only contained few tokens in both J1's & GPT's tokenizers, and the first token is usually quite indicative to the nature of the label, this effect should be minor.

Information extraction The LMs' predictions for the information extraction tasks were generated with greedy decoding at temperature 0, similar to Zhao et al. (2021). We used Exact Match (EM) or F1 as the metric of choice for the extraction tasks.

Computational cost As discussed in the beginning of this appendix, we used restrictive decoding for the majority of the experiments in the paper. This usage greatly reduced the computational cost of our experiments: Most classification tasks were preformed in 1-4 GPU hours for all models (besides experiments with J1-Jumbo, which lasted roughly 10-50 GPU hours per experiment). The experiments described in Section 3.3 and Section 4 took up to 20 GPU hours each.

B   Datasets Information

B.1   Overview

We used 15 different datasets for our classification experiments: SST-2 (Socher et al., 2013), CR (Ding et al., 2008), RTE (Dagan et al., 2006), Subj (Pang and Lee, 2004), CB (de Marneffe et al., 2019), AGNews (Zhang et al., 2015b), SST-5 (Socher et al., 2013), YELP (Zhang et al., 2015a),TREC (Li and Roth, 2002), DBPedia (Zhang et al., 2015a), NLU (Xingkun Liu and Rieser, 2019), BANKING77 (Casanueva et al., 2020) and CLINIC150 (Larson et al., 2019). TREC and NLU datasets were used with both fine and coarse grained labels. The different formats used in all of tasks, as well as the values of n_{max} for both J1 and GPT2 models, can be found in Table 9. We have also used 6 more datasets from extraction and multiple-choice domains, which were only evaluated with J1 models:

- ATIS airlines (Zhao et al., 2021); $n_{max} = 67$.

- MIT Movie Genre (Zhao et al., 2021); $n_{max} = 54$.

- SQuAD (Rajpurkar et al., 2016); $n_{max} = 8$.

- adversarialQA(Bartolo et al., 2020); $n_{max} = 8$.

- OpenBookQA (Mihaylov et al., 2018); $n_{max} = 87$.

- StoryCloze(Mostafazadeh et al., 2017); $n_{max} = 44$.

For Section 4 we used Natural Questions (Kwiatkowski et al., 2019) and HotpotQA (Yang et al., 2018) datasets. All datasets were evaluated with the standard test set or validation set in the absence of a public test set. As described in Section 3, we subsampled all test sets for the ICL experiments. In Natural Questions dataset, we used half of the test set (its original size was 3610 samples) to speed up evaluation. We used the full HotpotQA validation set, containing 7405 samples. The datasets are all in English.

The majority of the datasets can be found in the Huggingface Datasets package (Lhoest et al., 2021), apart from the information extraction tasks ATIS airlines (Hemphill et al., 1990) and MIT movie genre (Liu et al., 2012), which were taken from Zhao et al. (2021), and Natural Questions (Kwiatkowski et al., 2019) which was loaded and incorporated with retrieved documents using Pyserini (Lin et al., 2021). Since loading the training set via Pyserini is not currently a built-in option, we used the validation set of Natural Questions as an effective train set. We found this decision reasonable since we only used the training set for few-shot prompting, and we did not optimize any parameters using the validation set.

We have tried our best to track the licenses of the datasets used in this work. The license information that we have managed to find is as follows: SST-2, RTE, SST-5, NLU Scenario, NLU Intent, BANK-ING77 and SQuAD—CC-BY 4.0, adversarialQA—CC-BY-SA 4.0, DBPedia—CC-BY-SA 3.0.

B.2 Preprocessing and Formatting

In all ICL experiments, we used only pairs of inputs and expected outputs, without any instructions. For the classification datasets, we mainly used formats found in Lu et al. (2021) when applicable. For extraction and multi-choice datasets, we used the formats from Brown et al. (2020). We generated new formats for classification datasets with dozens of labels, which are rarely used in few-shot

setting. The formats were based on wordings and labels used in HuggingFace, with minor modifications to make the formats more similar to natural language (*e.g.*, replacing '_' with spaces in label names). Details of the classification prompts can be found in Table 9. Experiments from Section 4 were formatted similarly to the work done by (Lazaridou et al., 2022). Their prompts formats are presented in Table 10.

C The Effect of the Number of Context Windows on Performance

When using PCW for ICL, the number of parallel context windows (B) affects the number of in-context training examples. We used $B = 2, 3, 4$ in preliminary experiments, and saw that for classification tasks, the optimal choice of B depends on the number of unique labels in the task. We observed that the performance for tasks with a high number of classes was improved when we increased B, while the optimal choice of B for tasks with few classes tended to be 1 or 2 (See Tables 6,7 and 8). For simplicity, We chose to display results for $B = 3$ in all of the main experiments.

Nevertheless, we were curious to see how far we could push the number of parallel context windows before the model stopped benefiting from them. We used a representative subset of three datasets with a varying number of classes, and increased B from 1 to 8. The number of training sets and the size of test set for those experiments were set on 15 and 125 respectively.

As seen in Figure 7, when the number of context windows is increased, datasets with a large number of classes, such as AGNews and DBPedia (with 4 and 14 labels, respectively), continue to improve (with a convergence at around $B = 6$). Hence, PCW can achieve even greater improvements by optimizing B per dataset. Increasing the number of context windows, however, seemed to harm the performance of SST-2.

Identifying which tasks benefit from large parallel data processing would be an interesting research direction in the future. For now, we recommend choosing an optimal B on the development set (if available) for best results. In the absence of a development set, a conservative choice, such as $B = 3$, may be beneficial. It is possible to investigate the behaviour of PCW with a larger number of windows, but we find it irrelevant for most practical cases of ICL, where an extremely large number of
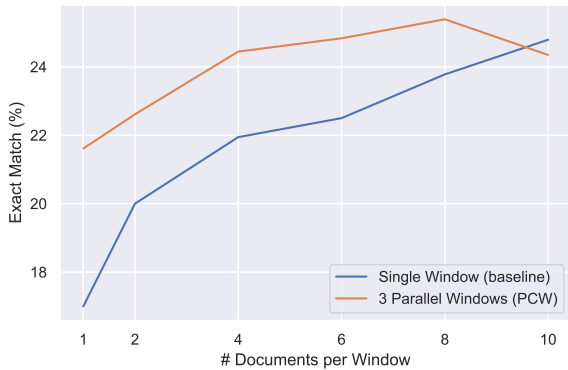
Figure 6: NQ results for J1 Large—EM against number of documents in a single window.

training samples would allow finetuning a model. We leave exploring this issue for future work.

D Additional Results

D.1 Replication Experiments with GPT2

Table 5 presents replication of the results shown in Table 1 for GPT2-Large and GPT2-XL models. A qualitative inspection of errors in these experiments suggested that vanilla ICL fails more in examples where the test label appears earlier in the prompt. Since PCW allows more context windows, it more often shows a training example with the test label towards the end of one of the windows. We evaluated GPT2-Large performance on the AGNews dataset and discovered that PCW shows a training example with the test label in a closer location to the test example 62% of the time. In those cases, PCW outperforms ICL by 19.4%, compared to a margin of roughly 10% for the entire test set. This analysis suggests that PCW provides a solution to the recency bias noted by Zhao et al. (2021).

D.2 Multiple-Choice QA

In addition to our in-depth investigation of PCW, we have experimented with two multiple-choice QA datasets OpenBookQA (Mihaylov et al., 2018) and StoryCloze (Mostafazadeh et al., 2017) under ICL setting. We formatted and evaluated the tasks as in Brown et al. (2020), by providing few-shot examples with the correct completion followed by an example of context only, and comparing the average per-token LM likelihood of each possible completion. We did not use the calibration from Brown et al. (2020). We used the same setup as described in Section 3.1, with the exception of reducing the number of sampled training sets and

Dataset	J1-Large (7.5B)		J1-Grande (17B)	
	ICL	PCW	ICL	PCW
OpenBookQA	$46.0_{1.5}$	$\mathbf{46.6_{1.0}}$	$51.6_{2.2}$	$\mathbf{54.2^*_{1.7}}$
StoryCloze	$84.2_{1.0}$	$\mathbf{84.3_{0.8}}$	$84.7_{0.9}$	$84.6_{1.0}$

Table 4: Results for J1-Large and J1-Grande models for Multiple Choice datasets.

the test set size used for J1-Grande in the Open-BookQA experiment to 15 and 125, respectively.

The results shown in Table 4 show that increasing the number of examples of in-context training under the PCW setting improved the performance of J1-Grande in the OpenBookQA task, but did not significantly affect the other scenarios. Based on this observation, it seems that PCW has the potential of providing gains for multiple-choice tasks in specific scenarios, but further analysis should be made based on more datasets to better understand it. We leave this for future work.

Dataset	# Labels	GPT2-Large(0.75B)		GPT2-XL(1.5B)	
		ICL	PCW	ICL	PCW
SST-2	2	$80.5_{11.4}$	$\mathbf{85.5_{5.7}}$*	$90.7_{3.8}$	$\mathbf{93.0_{2.1}}$*
CR	2	$81.3_{6.3}$	$\mathbf{83.8_{4.7}}$	$79.4_{6.0}$	$\mathbf{82.9_{3.4}}$*
RTE	2	$53.0_{2.5}$	$\mathbf{53.5_{1.9}}$	$55.4_{2.7}$	$\mathbf{55.5_{2.0}}$
Subj	2	$67.4_{12.3}$	$\mathbf{69.5_{11.8}}$	$68.0_{10.8}$	$\mathbf{68.6_{6.7}}$
CB	3	$\mathbf{45.3_{4.7}}$	$44.4_{4.2}$	$\mathbf{53.5_{9.2}}$	$51.9_{7.0}$
AGNews	4	$61.9_{12.9}$	$\mathbf{72.5_{7.0}}$*	$68.0_{12.4}$	$\mathbf{80.0_{3.5}}$*
SST-5	5	$41.1_{4.6}$	$\mathbf{44.7_{4.4}}$*	$37.1_{7.9}$	$\mathbf{43.3_{5.9}}$*
TREC	6	$55.6_{8.3}$	$\mathbf{57.7_{4.9}}$	$48.4_{4.7}$	$\mathbf{48.6_{2.6}}$
DBPedia	14	$63.1_{18.9}$	$\mathbf{80.7_{5.3}}$*	$77.2_{10.5}$	$\mathbf{87.3_{3.9}}$*
NLU Scenario	18	$\mathbf{37.0_{6.1}}$*	$31.4_{3.7}$	$47.5_{8.0}$	$\mathbf{52.9_{6.1}}$*
TREC Fine	50	$30.3_{7.8}$	$\mathbf{33.6_{4.0}}$*	$36.8_{6.4}$	$\mathbf{39.5_{2.8}}$*
NLU Intent	68	$24.3_{5.4}$	$\mathbf{28.1_{4.4}}$*	$30.2_{5.2}$	$\mathbf{38.9_{4.5}}$*
BANKING77	77	$\mathbf{29.3_{5.3}}$	$28.5_{4.0}$	$30.9_{4.0}$	$\mathbf{33.7_{3.2}}$*
CLINIC150	150	$44.2_{3.2}$	$\mathbf{45.4_{1.8}}$	$46.9_{2.5}$	$\mathbf{48.7_{1.9}}$*

Table 5: Accuracy results for GPT2-Large and GPT2-XL models with regular ICL in comparison with using PCW with $B = 3$ prompts. The results mirror the results found in Table 1 and use the same format, with the exception of YELP dataset that had $n_{max} = 0$ for GPT2 models.

Dataset	# Labels	ICL	PCW ($B = 2$)	PCW ($B = 3$)	PCW ($B = 4$)
SST-2	2	$93.5_{1.6}$	$\mathbf{94.1_{1.3}}$	$93.8_{1.1}$	$94.0_{1.1}$
CR	2	$\mathbf{93.9_{0.7}}$	$93.8_{0.7}$	$\mathbf{93.9_{0.7}}$	$92.9_{1.5}$
RTE	2	$58.3_{3.8}$	$\mathbf{59.4_{3.9}}$	$58.1_{3.7}$	$57.9_{4.3}$
Subj	2	$\mathbf{84.1_{7.7}}$	$81.9_{7.5}$	$79.1_{7.2}$	$77.7_{7.0}$
CB	3	$\mathbf{65.2_{8.0}}$	$59.9_{7.7}$	$61.2_{8.2}$	$56.8_{5.4}$
AGNews	4	$79.8_{3.6}$	$80.9_{2.4}$	$81.5_{2.1}$	$\mathbf{81.9_{1.9}}$
SST-5	5	$45.5_{3.9}$	$46.3_{3.9}$	$\mathbf{47.4_{2.9}}$	$46.1_{2.8}$
YELP	5	$56.2_{3.8}$	$\mathbf{56.8_{3.4}}$	$56.3_{5.1}$	$54.8_{3.1}$
TREC	6	$87.0_{4.5}$	$88.8_{3.4}$	$89.4_{3.2}$	$\mathbf{89.7_{3.0}}$
DBPedia	14	$93.2_{3.0}$	$95.1_{2.3}$	$96.2_{1.5}$	$\mathbf{96.4_{1.3}}$
NLU Scenario	18	$81.9_{2.2}$	$83.4_{1.7}$	$84.2_{1.5}$	$\mathbf{84.6_{1.4}}$
TREC Fine	50	$60.5_{6.9}$	$65.2_{3.8}$	$\mathbf{68.8_{3.4}}$	$68.8_{3.2}$
NLU Intent	68	$69.7_{3.3}$	$77.7_{2.1}$	$79.7_{1.9}$	$\mathbf{80.9_{1.9}}$
BANKING77	77	$51.0_{3.4}$	$58.7_{3.3}$	$63.5_{2.7}$	$\mathbf{65.8_{2.5}}$
CLINIC150	150	$67.3_{2.7}$	$74.4_{2.5}$	$75.4_{1.7}$	$\mathbf{78.1_{2.1}}$

Table 6: Results for different choices of B for J1-Large model. The best result for each dataset is boldfaced.

Dataset	# Labels	ICL	PCW ($B = 2$)	PCW ($B = 3$)	PCW ($B = 4$)
SST-2	2	$95.2_{1.1}$	$95.4_{0.7}$	$\mathbf{95.6_{0.5}}$	$\mathbf{95.6_{0.3}}$
CR	2	$93.6_{0.8}$	$\mathbf{93.9_{0.9}}$	$93.8_{0.8}$	$\mathbf{93.9_{0.7}}$
RTE	2	$61.2_{5.1}$	$\mathbf{64.2_{2.7}}$	$62.2_{3.0}$	$62.4_{3.4}$
Subj	2	$93.0_{2.5}$	$94.6_{1.3}$	$95.3_{1.2}$	$\mathbf{95.7_{1.0}}$
CB	3	$75.0_{8.1}$	$74.7_{8.3}$	$\mathbf{75.7_{6.0}}$	$73.0_{5.6}$
AGNews	4	$81.4_{3.0}$	$82.1_{2.4}$	$\mathbf{82.7_{2.1}}$	$82.6_{2.0}$
SST-5	5	$51.6_{3.4}$	$53.6_{2.9}$	$53.8_{2.2}$	$\mathbf{53.9_{2.0}}$
YELP	5	$66.2_{2.2}$	$\mathbf{66.6_{1.7}}$	$65.6_{2.0}$	$65.5_{1.9}$
TREC	6	$86.5_{3.8}$	$88.1_{4.0}$	$88.7_{3.4}$	$\mathbf{89.2_{4.5}}$
DBPedia	14	$92.5_{3.3}$	$95.8_{2.7}$	$97.3_{1.6}$	$\mathbf{97.9_{1.3}}$
NLU Scenario	18	$86.1_{2.1}$	$88.4_{1.4}$	$88.8_{1.1}$	$\mathbf{89.2_{1.2}}$
TREC Fine	50	$63.3_{6.0}$	$67.7_{4.3}$	$\mathbf{71.8_{4.6}}$	$71.2_{4.8}$
NLU Intent	68	$72.1_{3.1}$	$79.7_{2.4}$	$81.9_{1.6}$	$\mathbf{83.3_{1.6}}$
BANKING77	77	$55.2_{3.3}$	$64.5_{3.1}$	$69.1_{2.2}$	$\mathbf{70.9_{2.8}}$
CLINIC150	150	$68.9_{2.5}$	$76.5_{2.5}$	$78.6_{1.8}$	$\mathbf{80.2_{2.6}}$

Table 7: Results for different choices of B for J1-Grande model. The best result for each dataset is boldfaced.

Dataset	# Labels	ICL	PCW ($B = 2$)	PCW ($B = 3$)
SST-2	2	$96.5_{1.4}$	$\mathbf{97.8_{0.9}}$	$97.0_{1.5}$
CR	2	$93.6_{1.5}$	$\mathbf{93.9_{1.0}}$	$93.1_{1.0}$
RTE	2	$63.9_{5.0}$	$65.2_{3.9}$	$\mathbf{66.0_{4.1}}$
Subj	2	$89.1_{5.3}$	$91.6_{3.0}$	$\mathbf{93.6_{2.1}}$
CB	3	$76.2_{4.3}$	$76.2_{7.1}$	$\mathbf{76.6_{3.5}}$
AGNews	4	$82.5_{3.8}$	$84.9_{1.7}$	$\mathbf{85.9_{1.7}}$
SST-5	5	$55.4_{2.8}$	$\mathbf{55.6_{3.2}}$	$55.1_{3.9}$
YELP	5	$66.3_{4.1}$	$\mathbf{68.3_{2.5}}$	$65.4_{2.6}$
TREC	6	$87.1_{5.7}$	$89.1_{3.0}$	$\mathbf{90.4_{3.1}}$
DBPedia	14	$91.7_{4.4}$	$96.2_{2.6}$	$\mathbf{96.5_{2.3}}$
NLU Scenario	18	$85.4_{2.9}$	$87.1_{1.8}$	$\mathbf{87.8_{1.6}}$
TREC Fine	50	$71.4_{5.7}$	$77.5_{2.4}$	$\mathbf{78.7_{3.6}}$
NLU Intent	68	$74.3_{3.4}$	$80.3_{2.5}$	$\mathbf{81.6_{2.9}}$
BANKING77	77	$55.3_{3.5}$	$65.9_{3.9}$	$\mathbf{70.9_{3.1}}$
CLINIC150	150	$65.7_{5.0}$	$74.8_{4.2}$	$\mathbf{79.9_{2.1}}$

Table 8: Results for different choices of B for J1-Jumbo model. The best result for each dataset is boldfaced. For computational considerations, we have only attempted to use $B = 2$ and $B = 3$.
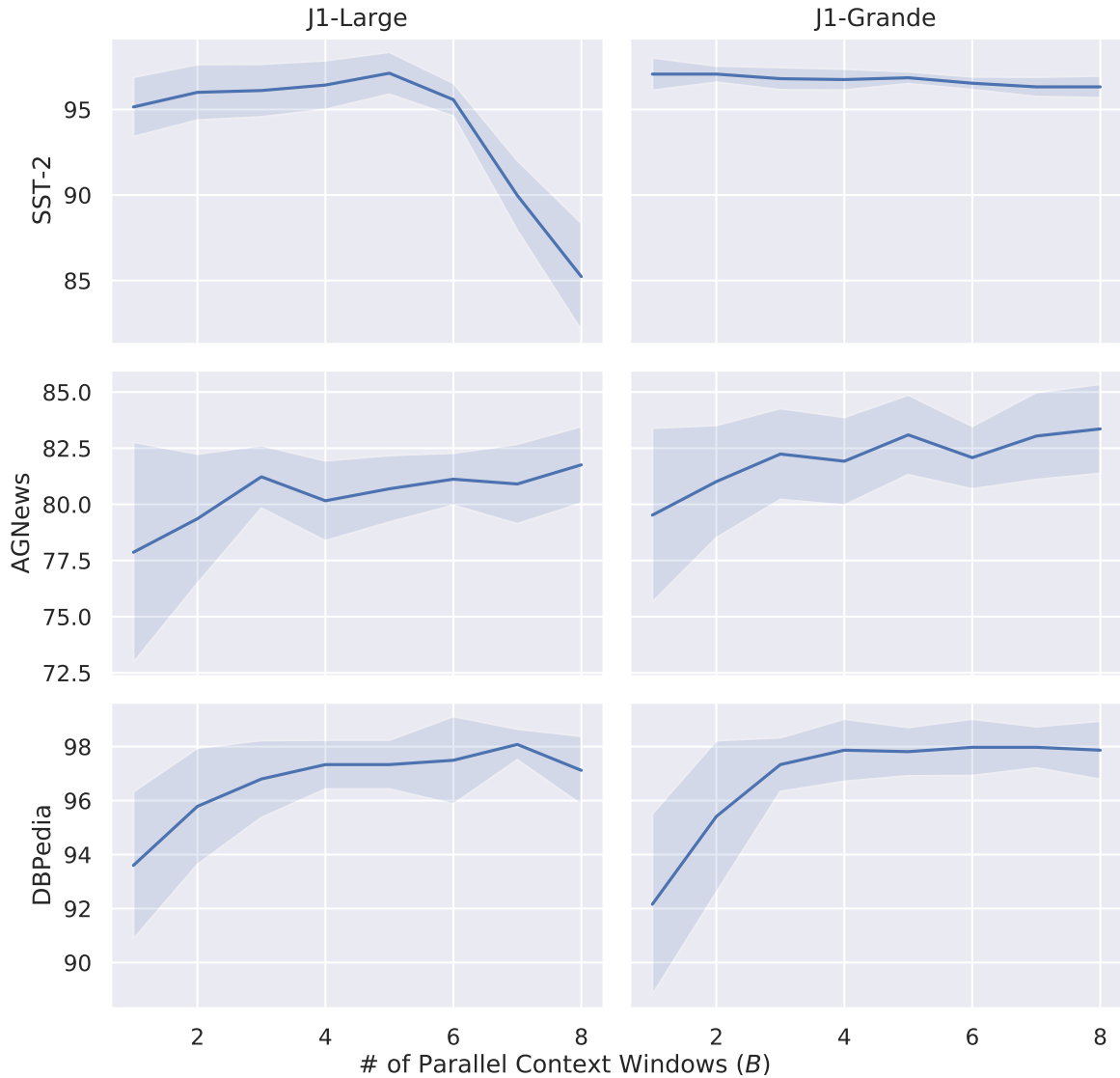

Figure 7: Multi-class tasks benefit from increased context windows, but simpler tasks with fewer classes may suffer from a decrease in performance.

Dataset	n_{max} J1	n_{max} GPT2	Prompt Example	Labels
SST-2	68	27	Review: , the power of this movie is undeniable . Sentiment: positive	[negative, positive]
CR	54	21	premise: Review: in fact , i liked it so much after using it with my son who is now 2 years old , that i bought one for our new baby ' s room Sentiment: positive	[negative, positive]
RTE	17	5	premise: The 10-day-old "test-tube" baby elephant born at Colchester Zoo has found a name, thanks to the internet! hypothesis: baby elephant born prediction: True	[True, False]
Subj	42	18	Input: they follow him to las vegas , where he is ostensibly doing " research " for the next season , but is actually pursuing a dream to become a dancer in a vegas show . Type: objective	[objective, subjective]
CB	19	5	premise: Paula could not help herself. It was just the way she was. Others might say they hated her and mean it. hypothesis: others hated Paula prediction: true	[true, false, neither]
AGNews	30	11	input: Citigroup faces regulatory probe The UK's Financial Services Authority launches a formal investigation into Citigroup's "unusual trading activity". type: business	[world, sports, business, technology]
SST-5	51	20	Review: it 's just a little too self-satisfied . Sentiment: okay	[terrible, bad, okay, good, great]
YELP	5	0	review: Good modern atmosphere and delicious cupcakes. stars: 3	[1, 2, 3, 4, 5]
TREC	88	38	Question: When was the first Barbie produced ? Type: numeric	[abbreviation, entity, description, human, location, numeric]
DBPedia	21	7	input: The Bstanu River is a tributary of the Râul Mare in Romania. type: nature	[company, school, artist, athlete, politics, transportation, building, nature, village, animal, plant, album, film, book]
NLU Scenario	112	43	utterance: you have got the answer right. scenario: general	[lists, weather, general, cooking, email, alarm, datetime, calendar, social, transport, iot, recommendation, takeaway, play, music, qa, news, audio]
TREC Fine	84	37	Question: What dropped 1 , 313 feet in 1980 ? Type: entity other	[abbreviation abbreviation, abbreviation expansion, entity animal, entity body, entity color, entity creation, entity currency, entity disease, entity event, entity food...]
NLU Intent	101	43	utterance: please read out the tasks from the list for today intent: lists query	[alarm query, alarm remove, alarm set, audio volume down, audio volume mute, audio volume other, audio volume up, calendar query, calendar remove, calendar set...]
BANK-ING77	77	27	query: Card payment didn't go through. intent: declined card payment	[activate my card, age limit, apple pay or google pay, atm support, automatic top up, balance not updated after bank transfer, balance not updated after cheque or cash deposit...]
CLINIC150	101	39	utterance: i would like to look up my credit score please intent: credit score	[restaurant reviews, nutrition info, account blocked, oil change how, time, weather, redeem rewards, interest rate, gas type...]

Table 9: Table of classification datasets with their used prompts and the n_{max} for both GPT2 and J1 tokenizers. For readability, we truncated the list of labels for some of the multi-label tasks.

Task	Prompt
Natural Questions (NQ)	Title: We Bought a Zoo Evidence: We Bought a Zoo We Bought a Zoo is a 2011 American family... The film also stars Scarlett Johansson, Maggie Elizabeth Jones... Question: who is the little girl on we bought a zoo? Answer: Maggie Elizabeth Jones Title: Vaal River Evidence: ...The river flows west into the Grootdraai Dam near Standerton, Mpumalanga. On its course to the Vaal Dam in Vereeniging... Question: where does the vaal dam get its water from? Answer: Vaal River == Title: San Juan River (Colorado River tributary) Evidence: in the San Juan Mountains has often been diminished by warming winter temperatures.. == Title: olorad Evidence: drained by the Colorado River. The South Park of Colorado is the region of the headwaters of the South Platte River... == Title: San Juan River (Colorado River tributary) Evidence: ...Colorado at the confluence of its East and West Forks. Both forks originate above elevations of in the eastern San Juan Mountains in the San Juan National Forest... == Question: where are the san juan mountains in new mexico? Answer:
HotpotQA	Evidences: == The 2009 Singapore Grand Prix (formally the 2009 Formula 1 SingTel Singapore Grand Prix) was a Formula One motor race held at the Marina Bay Street Circuit in Singapore on 27 September 2009... == Catharina Felser (born October 2, 1982) is a German race car driver born in Siegburg... == ...Sergio Pérez, the only other Mexican to finish on the podium, currently races with Sahara Force India F1 Team . == Sergio Pérez Mendoza (; born 26 January 1990) also known as "Checo" Pérez, is a Mexican racing driver, currently driving for Force India. == Question: Which other Mexican Formula One race car driver has held the podium besides the Force India driver born in 1990? Answer:

Table 10: Prompt formats for Natural Questions (NQ) and HotpotQA. The prompts were manually shortened for readability.

A For every submission:

☑ A1. Did you describe the limitations of your work?
Section Limitations

☒ A2. Did you discuss any potential risks of your work?
We see no reasonable direct potential risk

☑ A3. Do the abstract and introduction summarize the paper's main claims?
See abstract and section 1.

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

See sections 3-4.

☑ B1. Did you cite the creators of artifacts you used?
See sections 3,4 and appendix B.

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
See appendix B.

☒ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
All the artifacts were used for research purposes, similarly to how they were used in many other papers.

☒ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
The datasets used are quite common and are publicly available.

☑ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
See appendix B

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
See appendix B.

C ☑ Did you run computational experiments?

See Sections 3-4 and appendices C-D.

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
See section 3 and appendix A.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
See sections 3,4 and appendices.

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
See sections 3 and 4.

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
See appendix B.

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.