

Cached Transformers: Improving Transformers with Differentiable Memory Cache

Zhaoyang Zhang¹, Wenqi Shao¹, Yixiao Ge², Xiaogang Wang¹, Jinwei Gu¹, Ping Luo³

¹The Chinese University of Hong Kong

²Tencent Inc.

³The University of Hong Kong

{zhaoyangzhang@link., wqshao@link., xgwang@ee., jwgu@}cuhk.edu.hk, geyixiao831@gmail.com, pluo@cs.hku.hk

Abstract

This work introduces a new Transformer model called Cached Transformer, which uses Gated Recurrent Cached (GRC) attention to extend the self-attention mechanism with a differentiable memory cache of tokens. GRC attention enables attending to both past and current tokens, increasing the receptive field of attention and allowing for exploring long-range dependencies. By utilizing a recurrent gating unit to continuously update the cache, our model achieves significant advancements in **six** language and vision tasks, including language modeling, machine translation, ListOPs, image classification, object detection, and instance segmentation. Furthermore, our approach surpasses previous memory-based techniques in tasks such as language modeling and displays the ability to be applied to a broader range of situations.

Introduction

The design of Transformer (Vaswani et al. 2017), a deep model stacking self-attention and feed-forward layers, has achieved remarkable progress in various tasks. Compared to the traditional deep models, a key characteristic of Transformer is the self-attention mechanism, which enables global receptive field and allows each token to access all the other tokens in a data batch, providing a flexible scheme to capture contextual representation (Vaswani et al. 2017; Dosovitskiy et al. 2021; Carion et al. 2020). Such paradigm is however in a complexity square to sequence length, thus not suitable to model long-term dependencies. In this work, we aim to extend the conventional transformer models using attention with a long-term token representation in a memory cache, which enables larger and longer receptive field at minimal additional computations.

Capturing long-range relationships between tokens and samples is crucial for various tasks due to several reasons. (i) In sequential data such as language sentences, there can exist dependencies between tokens that are far away from each other. For example, an event or character can be referred to from time to time across multiple paragraphs in an article. Failing to capture such dependencies can result in poor performance in natural language processing tasks. (ii) Modeling cross-sample relationships can also be useful for non-sequential data like images. For example, incorporating

Copyright © 2024, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

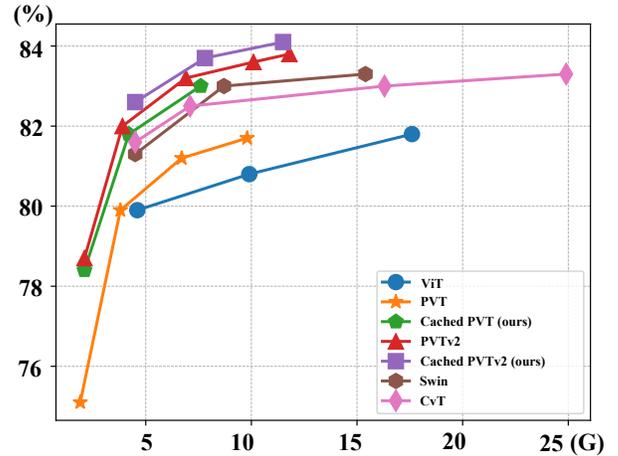


Figure 1: Performance-Complexity Curve: Top-1 accuracy (%) comparisons on ImageNet with respect to model capacity (FLOPs) using vision transformers (Parameter-efficiency curves). Curves of our cached models are consistently on top of their corresponding baselines (PVT and PVTv2), indicating the effectiveness of GRC-cached models considering both complexity and accuracy.

a memory module that stores prototypical feature representations can enable instance-invariant feature learning, leading to improved performance in vision tasks (Long et al. 2022; Deng et al. 2022). Furthermore, other studies (Wang et al. 2020b; Zhong et al. 2019) have demonstrated that using cross-batch memory to store previous embeddings can be beneficial for visual representation learning. (iii) Longer-range attention has also been shown to enhance the representation learning ability of models, as demonstrated in works like (Dai et al. 2019; Wu et al. 2022; Tay et al. 2021b).

However, longer dependency modeling makes computations more expensive. For example, the vanilla Transformer has $O(T^2)$ computational complexity in each attention module when handling a token sequence of length T . Although some works apply efficient alternatives, such as low-rank decomposition (Wang et al. 2020a; Zhu et al. 2021), block-based sparsification (Zaheer et al. 2020), and local sensitive hashing (Kitaev, Kaiser, and Levskaya 2020), they still have complexity linear to the token length ($O(T)$) and thus unable to efficiently capture sparse long-range dependency.

Another line of research (Wu et al. 2022) reduces the complexity of attention module by selecting top- k token pairs from a memory cache for the current tokens, but the cost of maintaining a huge cache of tokens for all layers is still significant. Hence, developing efficient and effective mechanisms for capturing long-range dependencies remains an active area of research.

To address these issues, we propose a novel family of Transformer models called Cached Transformer, which has a Gated Recurrent Cache (GRC) that enables Transformers to access historical knowledge, as illustrated in Fig. 2. The GRC is implemented as a meta-learner that compresses the historical representation into embedding vectors and updates them adaptively with a gating mechanism, avoiding the need for a large memory cache. The GRC updates the past representation with a reset gate that suppresses historical caches and an update gate that further updates the suppressed caches using the current token sequences. This design allows the GRC to access previously seen knowledge in a computationally efficient way. Based on the GRC, we implement a semi-cached attention mechanism that attends to both the latent and current tokens.

We propose Cached Transformer with Gated Recurrent Cache (GRC) and make the following **contributions**, which make it more appealing than prior arts in several aspects.

- GRC is built on a general differentiable formulation and is compatible with various attention schemes, Transformer networks, and tasks. We demonstrate that GRC can be easily plugged into diverse Transformer-variants such as Transformer-XL (Dai et al. 2019), ViT (Dosovitskiy et al. 2021), PVT (Wang et al. 2021, 2022), Swin (Liu et al. 2021) Bigbird (Zaheer et al. 2020), and Reformer (Kitaev, Kaiser, and Levskaya 2020).
- GRC can cache all representations of arbitrary length recurrently, independent of sequence length, while existing cache-based methods can only capture recent tokens (Rae et al. 2019; Dai et al. 2019) or require KNN searching at each step (Wu et al. 2022).
- Besides efficiency, GRC surpasses previous memory-based methods (Dai et al. 2019; Burtsev et al. 2020; Bulatov, Kuratov, and Burtsev 2022) by a large margin on both vision (Table 2) and language tasks (Table 5).
- GRC yields consistent improvements not only in sequential data such as texts but also in spatial context such as image classification (Table 1) and object detection (Table 3). To our knowledge, existing works of Vision Transformers mainly focused on learning intra-sample tokens, while GRC is the first attempt to model cross-sample relationships by attending over inter-sample tokens, such as tokens from different independent images.
- We observe that models with GRC may attend more over the cache than the regular self-attention. We investigate this behavior in image classification and find that GRC can separate features into two parts, attending over caches yielding **instance-invariant** features, as well as attending over self, yielding **instance-specific** features (See in Fig. 4). This behavior is similar to that of a vector prototype (Caron et al. 2020), which enables cross-sample regularization to avoid overfitting.

Extensive experiments show that the Cached Transformer with GRC achieves promising results on various vision and language Transformer backbones. (i) **Language:** In the IWSLT14 De-En benchmark for machine translation, PreNormed Transformer+GRC yields 36.0 BLEU, outperforming the baselines by 0.5. In the challenging long-range-arena benchmark (Tay et al. 2021a), GRC improves state-of-the-art methods with different attention types including Reformer (Kitaev, Kaiser, and Levskaya 2020), Bigbird (Zaheer et al. 2020), and regular Transformer (Vaswani et al. 2017) consistently by up to 1.2% accuracy. (ii) **Vision:** For image classification on ImageNet (Krizhevsky, Sutskever, and Hinton 2012), we plug GRC into the recent vision transformers of different scales, such as ViT (Dosovitskiy et al. 2021), PVT (Wang et al. 2021), PVTv2 (Wang et al. 2022), Swin (Liu et al. 2021), and obtain up to 3.3% accuracy gain. As shown in Fig. 1, our cached model with PVTv2 backbone achieves superior performance considering both the model **complexity and accuracy**. We further evaluate GRC on the COCO (Lin et al. 2014) dataset for object detection and instance segmentation, where PVT+GRC can yield more than 4.0 box AP improvement.

Related Works

Cached Language Models. Cache models are effective in long-range modeling, and are firstly introduced by (Kupiec 1989; Kuhn and De Mori 1990) for speech recognition. In general, a cache model stores representations of the past, which are usually unigrams or key-value pairs for future computation. Transformer-XL (Dai et al. 2019) further applies this technique to transformers, where the cache stores previous key-value pairs in attentions from previous training steps. Many memory-based methods are explored following Transformer-XL: For instance, MT (Burtsev et al. 2020) and RMT (Bulatov, Kuratov, and Burtsev 2022) use extra memory tokens to store local and global information for different segments of inputs. (Rae et al. 2019) compress the tokens before they’re saved in the cache to reduce memories and computations. However, these methods often use cache in a fixed-length and first-in-first-out (FIFO) manner, which limits the amount of tokens that can be memorized in sequence. In contrast, our proposed GRC-based Cached Transformers learn to build the cache adaptively with a complexity that is independent of the attention range.

Vision Transformers. Vision transformers and their variants have recently achieved remarkable success in various vision tasks. The original Vision Transformer (ViT) model (Dosovitskiy et al. 2021) was the first to split images into patch sequences and feed them into transformer encoders. However, existing methods focus mainly on intra-sample tokens, whereas our proposed GRC enhances vision transformers by learning instance-invariant features via attending over inter-sample tokens. This allows GRC-based transformers to capture richer contextual information and achieve even better performance on vision tasks. For a more comprehensive understanding of the related literature, please refer to the Appendix.

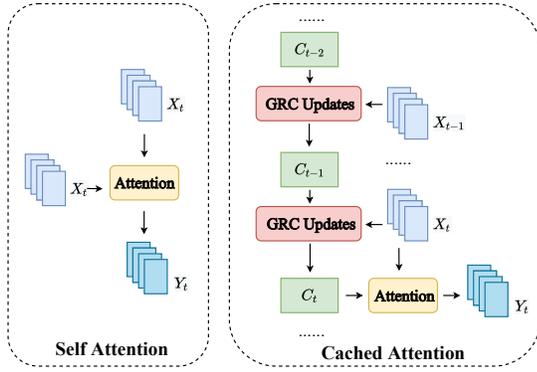


Figure 2: Comparisons of vanilla self-attention and cached attentions at training stage. The self-attention only attends to the token itself (X_t). While in cached attention, outputs at training step t (denoted by Y_t) are derived by attending over a Gated Recurrent Cache (GRC, i.e., C_t derived from historical tokens X_0 to X_t), and the current token (X_t).

Methodology

In this section, we first revisit the vanilla language and vision transformer models, then introduce implementation of Cached Transformers with Gated Recurrent Cache(GRC).

Vanilla Transformer

We begin with a brief review of the standard transformer architecture. The transformer model (Vaswani et al. 2017) is constructed by stacking multi-head self-attention blocks and feed-forward layers which is usually a two-layer linear transformation with activation. Each transformer block is fed with $T \times D$ input tokens, where T is the number of tokens and D represents the size of token embedding.

Self-attention mechanism. As shown in Fig.2, the self-attention module first projects each input X into Q (query), K (key), and V (value) using linear transformations. Typically, the self-attention is performed in a multi-head manner where the input will be divided into multiple heads for parallel computation. The output of the attention head h can be written as :

$$o_{self}^h = \text{softmax}(Q_h K_h^T / \sqrt{D/H}) V_h, \quad (1)$$

where o_{self}^h is the output of head h of the self-attention and H is the number of heads. The output from heads will be concatenated and then fed into another linear transformations with normalization and residual connections.

Limitations. As shown in Eqn.(1), the vanilla self-attention mechanism used in Transformers is highly sensitive to sequence length, with a computational complexity of $O(T^2)$ with respect to the sequence length T . This means that the computational cost grows rapidly as the sequence length increases, which limits the model’s ability to capture long-term relationships in the data. As a result, vanilla Transformers can only model relatively short sequences of tokens in language tasks, and it also makes it challenging to develop cross-task memory modules (Wang et al. 2020b; Zhong et al. 2019) in a attention-based way for vision tasks. Towards this issue, we introduce the proposed Cached Transformers, which provides a more flexible

paradigm for capturing long-term dependencies, leading to consistent improvements for both vision and language tasks.

Cached Transformer

To extend receptive fields of both language and vision transformers, in this section we will introduce our implementations of Cached Transformers, which maintains a continuous cache termed Gated Recurrent Cache (GRC) to support efficient long-term representation learning. The core idea is to hold token embedding as caches which can dynamically record historical samples according to their significance. The Cached Transformer will then gain additional capabilities to encode both the current and accumulated information by attending to the gathering of caches C and inputs X . Such an attention scheme is described as GRC-Attention, and the following parts present more details.

General implementations. The proposed Cached Transformers enable attending over caches on arbitrary multi-layers architectures accepting sequential inputs. Typically, the Cached Transformer models can be derived by replacing their self-attention blocks with the proposed GRC-Attention. Fig. 3 (b) gives overall illustrations of how the GRC-Attention is conducted.

Considering input sequence $X_t \in \mathbf{R}^{B \times T \times D}$, where B is the batch size and t denotes training steps, GRC-attention attends to both the memory cache and the current tokens. We formulate GRC-attention by

$$O^h = \sigma(\lambda_h) * o_{mem}^h + (1 - \sigma(\lambda_h)) * o_{self}^h, \quad (2)$$

where O^h and o_{mem}^h are the outputs of the GRC-attention and Cached attention (i.e., attention over memory cache) in the head h , respectively. o_{self}^h is the output of the self-attention in Eqn.(1). Moreover, in Eqn.(2), $\sigma(\cdot)$ is the sigmoid function and λ_h is a head-wise learnable ratio trading off self-attention and Cached attention ¹.

To construct the triplet key, query and value for Cached attention, we choose a portion of X_t as input $\bar{X}_t \in \mathbf{R}^{B \times T \times D_m}$, which is derived by slicing X_t on channel dimension. Note that $D_m = rD^2$ indicates channels used for memorizing the past tokens embedding, where r is the caching ratio. With \bar{X}_t , the accumulated cache C_{t-1} will then be updated to C_t according to the GRC update rules as shown in Fig. 3. We describe the construction of GRC in Sec in detail. The Cached attention can be then conducted by using \bar{X}_t as queries and C_t as keys and values, written as:

$$o_{mem}^h = \text{softmax}(\bar{Q}_h \bar{K}_h^T / \sqrt{D_m/H}) \bar{V}_h, \quad (3)$$

where \bar{Q}_h , \bar{K}_h and \bar{V}_h are obtained by linear projections of h -th head of \bar{X}_t , C_t and C_t respectively.

Generalizations. Note that while we typically formulate Cached Transformer as a self-attention based model, it can also be an arbitrary transformer variant. In other words, the attention mechanism used to acquire o_{self}^h and o_{mem}^h in

¹All of the λ_h is initialized to be 0.

²At most cases we adopt $D_m = \frac{D}{2}$ to reduce the complexity of Cached attention , which means we choose half of the inputs to update caches

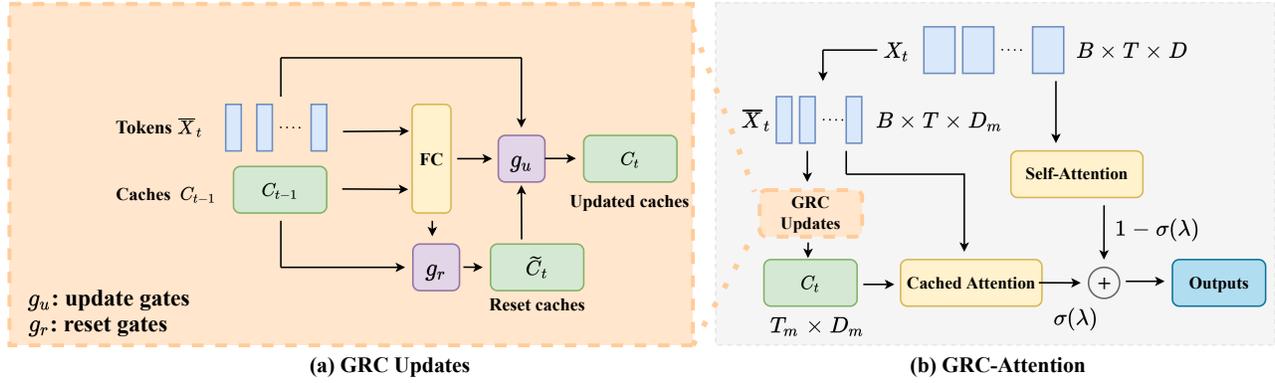


Figure 3: The illustration of proposed GRC-Attention in Cached Transformers. (a) Details of the updating process of Gated Recurrent Cache. The updated cache C_t is derived based on current tokens \bar{X}_t and cache of last step C_{t-1} . The reset gates g_r reset the previous cache C_{t-1} to reset cache \tilde{C}_t , and the update gates g_u controls the update intensity. (b) Overall pipeline of GRC-Attention. Inputs will attend over cache and themselves respectively, and the outputs are formulated as interpolation of the two attention results.

Eqn.(2) can be substituted by any other attention-like functions, such as sparse attentions (Zaheer et al. 2020) or local hashing (Kitaev, Kaiser, and Levskaya 2020). Further experiments will provide validations of Cached Transformers on several transformer variants.

Gated Recurrent Cache Update

This section describes the formulation and updating of proposed Gated Recurrent Cache (GRC).

Cache Initialization. The GRC is characterized to be fixed-length vectors $C_t \in \mathbf{R}^{T_m \times D_m}$. Unlike previous works that formulate cache to be tokens or words directly (Tu et al. 2018; Dai et al. 2019), GRC embeds historical tokens implicitly. By learning to embed arbitrary length samples into C_t , GRC allows traversing caches in constant time that is independent of the number of memorized tokens. The cache C_0 will be initialized to be T_m -length zero vectors before training, and then updated as depicted in Fig. 3(a).

Gating Mechanism. Inspired by gated RNNs (Cho et al. 2014), we adopt the gating mechanism to enable GRC to dynamically capture dependencies at different time scales. Specifically, the updating process of C_t is filtered by update gates g_u and reset gates g_r . Considering updating GRC at time step t , we first calculate the gates g_u and g_r :

$$g_u = \sigma(W_u[\bar{X}_t, C_{t-1}]) \text{ and } g_r = \sigma(W_r[\bar{X}_t, C_{t-1}]), \quad (4)$$

where σ denotes sigmoid function and $[\cdot, \cdot]$ concatenates tokens in channel dimension. For valid concatenation, \bar{X}_t is interpolated into a T_m -by- D_m token. The updated cache C_t is formulated by a linear interpolation as given by:

$$C_t = (1 - g_u)C_{t-1} + g_u\tilde{C}_t \text{ and } \tilde{C}_t = W_c[\bar{X}_t, g_r \odot C_{t-1}] \quad (5)$$

where \odot is element-wise multiplication. In above process, the update gates g_u decides how much current sample \bar{X}_t updates the cache and the reset gates g_r suppress the accumulated cache to forget unimportant components. Note that shape of the derived C_t is $B \times T_m \times D_m$ as X_t is involved, and we therefore average across the batch dimension to fit the cache size.

Architecture	Top-1 (%)	Top-5 (%)	Δ Top-1 (%)
ViT-S	79.9	95.0	-
ViT-S (Cached)	81.3	95.5	+ 1.4
PVT-T	75.1	92.3	-
PVT-T (Cached)	78.4	94.2	+ 3.3
PVT-S	79.9	95.0	-
PVT-S (Cached)	81.8	95.9	+ 1.9
PVT-M	81.2	95.7	-
PVT-M (Cached)	83.0	96.4	+ 1.8
Swin-T	81.2	95.5	-
Swin-T (Cached)	82.1	95.9	+ 0.9
PVTv2-B2	82.0	95.9	-
PVTv2-B2 (Cached)	82.6	96.2	+ 0.6
PVTv2-B	83.2	96.3	-
PVTv2-B3 (Cached)	83.7	96.4	+ 0.5
PVTv2-B4	83.6	96.3	-
PVTv2-B4 (Cached)	84.1	96.6	+ 0.5

Table 1: Performance of various Cached Transformers evaluated on ImageNet. "(Cached)" indicates models implemented with the proposed GRC-Attention. Top-1 / Top-5 / Δ Top-1 denotes top-1 accuracy / top-5 accuracy / top-1 accuracy difference respectively. The cached models outperform their corresponding baselines consistently.

Experiments

This section extensively evaluates the effectiveness of the proposed Cached Transformer and Gated Recurrent Cache (GRC) in both vision and language tasks, including language modeling on WikiText-103, Long Listsops of Long Range Arena (Tay et al. 2021a), machine translation on IWSLT14 (Cettolo et al. 2014) / IWSLT15 (Cettolo et al. 2015), image classification on ImageNet (Krizhevsky, Sutskever, and Hinton 2012), and object detection and instance segmentation on COCO2017 (Lin et al. 2014). In addition, as the cached models are newly introduced to vision transformers, we also perform thorough discussions on the role of the proposed caches and their significance. All of the experiments are conducted on Tesla V100 GPUs.

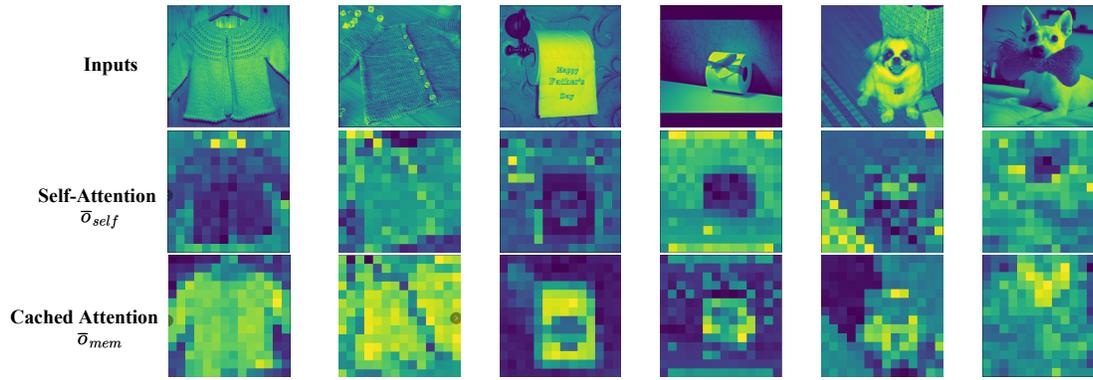


Figure 4: Visualizations of averaged features output from self-attention and cached attention, which is obtained by feeding images of ImageNet validation sets to trained cached ViT-S. The results are obtained by averaging features over channel (and head) dimension. Both \bar{o}_{self} and \bar{o}_{mem} are unflattened to 14×14 for better comparisons. Dark pixels mean small values.

Image Classification

Experiments Setup. We first evaluate our methods on Imagenet-1k for image classification. We implement our GRC-Attention as a general pytorch module which maintains fixed-length buffers as cache. In image classification task, we set the cache ratio r to be 0.5 and keep cache length T_m equal to the length of image patches T . For fair comparisons, we directly replace the self-attention layers in corresponding transformers with our GRC-Attention module without varying the architecture and hyperparameters. To maintain spatial token structures, we add positional encodings to our proposed GRC-Attention like other vision transformers. Both the baselines and their cached counterparts are trained with 224×224 size inputs using 16 GPUs. To fully validate the proposed cache mechanism, we evaluate GRC-Attention on four recent vision transformers including: ViTs (Dosovitskiy et al. 2021), PVT (Wang et al. 2021), Swin-Transformer (Liu et al. 2021) and PVT-v2 (Wang et al. 2022). Without bells and whistles, all of the training settings for cached models are kept consistent with the original baselines including data augmentation, optimizer type, learning rates and training epochs.

Classification Results. Table 1 reports overall performance of cached transformers on corresponding baselines. As shown, transformers implemented with GRC-Attention consistently outperform their no-cache counterparts by yielding significantly higher accuracy, demonstrating the effectiveness of our proposed caching mechanism. For instance, by enabling cache, PVT-Tiny can achieve 78.4% top-1 accuracy and 94.2% top-5 accuracy, surpassing the original PVT-Tiny by 3.3% and 1.9% respectively. Moreover, even for the recent stronger backbone PVTv2, our proposed cached mechanism can still keep > 0.5 top-1 improvements.

Complexity Analysis. In current settings where cache ratio $r = 0.5$, replacing all the attention layers with GRC-Attention will cost approximately an extra 10% – 15% FLOPs and Params. Considering the performance improvements, the extra computations are acceptable (See in Fig. 1) and more efficient than increasing the depth and width of models.

Significance of Cached Attention. To verify that the above performance gains mainly come from attending over caches, we analyze the contribution of o_{mem} by visualizing the learnable attention ratio $\sigma(\lambda^h)$. Please be reminded that in Eq 2, outputs of GRC-Attention is derived by interpolating outputs of cached attention o_{mem}^h and self-attention o_{self}^h according to $\sigma(\lambda^h)$. Hence, $\sigma(\lambda^h)$ can be used to represent the relative significance of o_{mem}^h and o_{self}^h . Fig. 5 depicts the learned $\sigma(\lambda^h)$ for each head respect to layers in ViT-S, PVT-Tiny and PVT-Small. As we can see, for more than half of the layers, $\sigma(\lambda^h)$ is larger than 0.5, denoting that outputs of those layers are highly dependent on the cached attention. Besides, we also notice an interesting fact that the models always prefer more cached attention except for the last several layers. This makes us curious about the roles of cached attention: what is the feature that models actually learn by attending over caches? The following paragraph answers this question.

Roles of Cached Attention. We investigate the function of GRC-Attention by visualizing their interior feature maps. We choose the middle layers of cached ViT-S, averaging the outputs from self-attention o_{self} and cached attention (o_{mem}) across the head and channel dimension, and then normalizing them into $[0, 1]$. The corresponding results are denoting as \bar{o}_{self} and \bar{o}_{mem} , respectively. Fig. 4 provides visualizations of \bar{o}_{self} and \bar{o}_{mem} obtained by feedings images of ImageNet validation sets to trained cached ViT-S. As \bar{o}_{self} and \bar{o}_{mem} are sequences of patches, they are unflattened to 14×14 shape for better comparison. From Fig. 4 we can see, features derived by the above two attentions are visually complementary. In GRC-Attention, o_{mem} is derived by attending over the proposed cache (GRC) containing compressive representations of historical samples, and thus being adept in recognizing **public** and frequently showing-up patches of this **class**. While for o_{self} from self-attention branch, it can focus on finding out more private and **characteristic** features of current **instance**.

With above postulates, we can attempt to explain the regularity of $\sigma(\lambda^h)$ in Fig. 5: employing more o_{mem} (larger $\sigma(\lambda^h)$) in former layers can help the network to distinguish

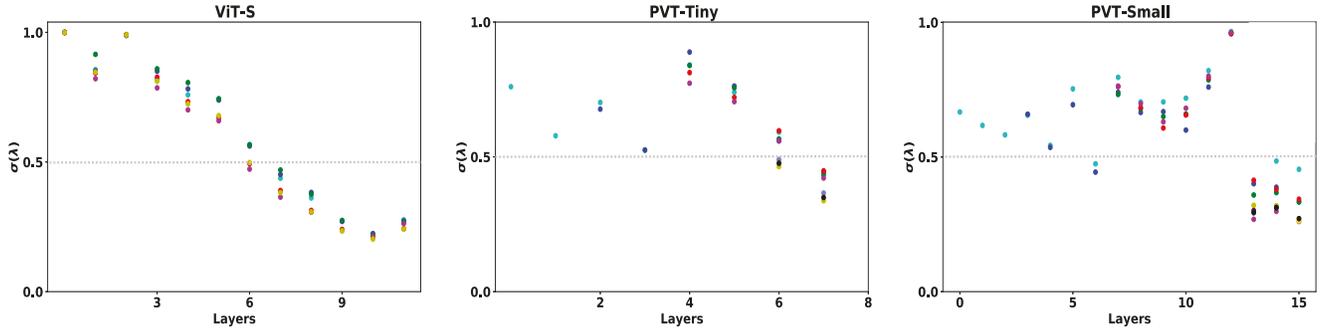


Figure 5: Visualizations of learned $\sigma(\lambda^h)$ for each head with respect to layer number (from shallow to deep) in different models: ViT-S, PVT-Tiny and PVT-Small. Note that the ViT-S has 6 heads for all the layers, while PVT-Tiny and PVT-Small adopt a progressive head strategy where head numbers increase from 1 to 8 gradually. Circles with different colors denote those different heads. $\sigma(\lambda^h)$ controls the interpolation ratio of cached attention outputs o_{mem} which reflects head-wise contributions of cached attention to the final outputs. Note that $\sigma(\lambda^h) > 0.5$ means cached attention contributes more than self-attention. As shown, in all of the three models, $\sigma(\lambda^h) > 0.5$ holds for more than half of the GRC-Attention layers, deducing that the model outputs are significantly dependent on the cache.

Model	No cache	Attention-based cache	GRC
ViT-S	79.9	80.0	81.3
PVT-T	75.1	74.8	78.4
PVT-S	79.9	79.6	81.8

Table 2: Performance (Top-1 Accuracy) comparisons of cached models using GRC and attention-based

Architecture	AP ^b	AP ₅₀ ^b	AP ₇₅ ^b	AP ^m	AP ₅₀ ^m	AP ₇₅ ^m
PVT-Tiny	36.7	59.2	39.3	35.1	56.7	37.3
+ Cached	41.0	63.4	44.8	38.3	60.4	41.1
PVT-Small	40.4	62.9	43.8	36.3	60.1	40.3
+ Cached	44.5	67.1	48.6	41.0	64.0	44.1
PVT-Medium	42.0	64.4	45.6	39.0	61.6	42.1
+ Cached	46.6	68.2	51.0	42.3	65.3	45.5

Table 3: Object detection and instance segmentation performance on COCO *val2017* following Mask R-CNN 1× settings.

this instance coarsely, and employing more o_{self} (smaller $\sigma(\lambda^h)$) enable the model to make fine-grained decision.

Cross-sample regularization. The above paragraph also shows that our proposed cache performs similarly to vector prototypes (Caron et al. 2020), storing public features of the same class implicitly and allowing models to classify inputs with both the public and characteristic representations. In such a way, the predictions are not only dependent on the current inputs but also on related cached samples, thus providing a cross-sample regularization to avoid overfitting.

GRC v.s. other memory-based methods. We perform further ablations to compare GRC and attention-based memory for image classification in ImageNet-1k. We deploy Transformer-XL-style caches to Vision Transformers (including ViT-S, PVT-Tiny and PVT-Small) and compare them to corresponding GRC-cached models. As shown in Table 2, GRC-cached models consistently outperform their attention-based cache and no-cache counterparts. Besides, it can be noted that the attention-based cache can hardly improve the model performance.

Object Detection and Instance Segmentation.

Experiments Setup. We further assess the generalization of our GRC-Attention on object detection / instance segmentation track using COCO2017 dataset (Lin et al. 2014). The models are trained on the COCO *train2017* (118k images) and evaluated on *val2017* (5k images). We use the cached PVT as backbone and adopt the Mask R-CNN detector (He et al. 2017) to verify the effectiveness of GRC-Attention. The standard COCO metrics of Average Precision (AP) for bounding box detection (AP_{bb}) and instance segmentation (AP_m) are used to evaluate our methods. All of the training settings and hyperparameters are kept the same as PVT original implementation (Wang et al. 2021), and all of the involved models are trained for 12 epochs using 8 GPUs. For both the cached PVT and baselines, backbones are firstly pretrained on ImageNet and then fine-tuned for detection.

Results. As shown in Table 3, when using Mask R-CNN for object detection, the cached PVTs significantly outperform their baselines. For example, the AP of cached PVT-Medium is 4.6 (46.6 vs. 42.0) points better than its no-cache counterparts. Similar results can also be found in instance segmentation results, where cached PVT-Medium achieves 3.3 higher AP^m (39.0 vs. 42.3). These results demonstrate the generalizability of the proposed caching mechanism.

Language Modeling

Experimental Setup In this work, we conduct experiments to compare the performance of Gated Recurrent Cache (GRC) with Transformer-XL (Dai et al. 2019) on a language modeling task using the WikiText-103 benchmark. To implement GRC-cached language models, we use the publicly available fairseq framework and follow the default memory-based Transformer-XL configurations as our baselines, including model architecture and training settings. To ensure a fair comparison, we compare GRC-cached models with two other memory-based methods, Memory Transformer (MT) (Burtsev et al. 2020) and Recurrent Memory Transformer (RMT) (Bulatov, Kuratov, and Burtsev 2022). We implement GRC-cached models by replacing the

Architecture	IWSLT14			IWSLT15		
	De-En	Es-En	En-Fr	De-En	En-Vi	Cs-En
Transformer	35.5	41.4	41.5	36.1	29.8	28.8
Transformer (GRC-cached)	36.0(+ 0.5)	41.8(+ 0.4)	41.7(+ 0.2)	36.3(+ 0.2)	30.2(+ 0.4)	29.4(+ 0.6)

Table 4: Neural machine translation results using Pre-Norm Transformers in terms of BLEU scores.

Architecture	No cache	MT	RMT	GRC
Transformer-XL _{base}	24.0	23.99	23.95	22.9
Transformer-XL _{large}	18.3	-	-	17.9

Table 5: Comparison of performance(Test PPL) for GRC and other Memory-based methods (Burtsev et al. 2020; Bulatov, Kuratov, and Burtsev 2022) on WikiText-103. The smaller is better. GRC outperform Transformer-XL and previous memory-based methods for language modeling by a large margin of 1.1 PPL.

caching scheme with the GRC approach while keeping all data augmentation and hyper-parameters unchanged for a more fair comparison.

Comparison to Other Memory-Based Methods We present the performance of GRC-cached models compared to Transformer-XL baselines and other memory-based methods in Table 5. The results show that GRC-cached models outperform Transformer-XL and other memory-based methods in terms of perplexity on both base and large-scale models. For instance, GRC-cached Transformer-XL_{base} achieves up to 1.1 lower PPL compared to the baseline Transformer-XL and 1.05 lower PPL to the RMT, demonstrating the superiority of GRC over previous memory-based Transformer methods.

Long Range Arena

Experiments Setup. We extensively conduct experiments on recently proposed Long Range Arena (LRA) benchmarks (Tay et al. 2021a) to validate our proposed methods under the long-context scenario. To demonstrate the long-range sequence modeling capability of GRC-Attention and the corresponding cache mechanism, we choose the challenging Long ListOps task in LRA, which is a longer variation of ListOps task (Nangia and Bowman 2018) with up to 2k length sequences and considerably difficult. In this task, we also extend GRC-Attention to efficient attention variants by replacing the self-attention function (See section). Concretely, we compare GRC-Attention to their no-cache counterparts on baselines including Transformer (Vaswani et al. 2017), BigBird (Zaheer et al. 2020) and Reformer (Kitaev, Kaiser, and Levskaya 2020). For those efficient attentions like BigBird and Reformer, we only import gated recurrent cache and maintain their inner attention function unchanged. All of the experiments are under default settings in (Tay et al. 2021a).

Results. Table 6 reports Long ListOps results. As shown, cached models consistently outperform their baselines (including the SOTA methods Reformer) significantly. For instance, by employing GRC, BigBird model can achieve 1.39 higher accuracy. These results show the long-range sequence modeling ability of GRC as well as its generalizability to other attention variants.

Architecture	baseline	GRC-cached	Δ
Transformer	36.23	37.40	+ 1.17
BigBird	36.06	37.45	+ 1.39
Reformer	37.27	37.85	+ 0.58

Table 6: Results on Long ListOPs task in LRA in terms of accuracy. The "cached" column indicates cached models whose attention layers are implemented as generalized GRC-Attention. Δ denotes the difference between proposed cached models and baselines.

Neural Machine Translation

Experiments Setups. We experiment our methods on widely used public datasets IWSLT14 and IWSLT15. Multiple language sources³ are included to fully verify effectiveness of the proposed GRC, and models are trained for each track individually. We adopt the Pre-Norm Transformer settings in (Wang et al. 2019) and implement the models using *fairseq-py* (Ott et al. 2019) framework. Following (Wang et al. 2019; Ott et al. 2019), we generally increase the learning rates by 2 and average the last 10 checkpoints for inference. We employ the proposed GRC-cached models by replacing all attention modules of transformer encoder layers with GRC-Attention. The cache length T_m is set to be 64 for all cached models. All the transformers in this task are using six encoder layers and six decoder layers. For a fair comparison, both the baselines and cached models are trained under identical settings.

Results. We use BLEU (Papineni et al. 2002) as evaluation metrics and compare GRC cached transformers to their baselines in Table 4. It can be seen that consistent improvements can be reached by applying GRC-Attention to baselines. For tracks like IWSLT14 De-En and IWSLT15 Cs-En, the increments can achieve 0.5/0.6 points, which is actually significant for these tasks.

Discussion

We introduce Cached Transformer with Gated Recurrent Cache (GRC), a simple extension to Transformer-based models that significantly increases the length of attention context by allowing access to historical states through a gating mechanism. GRC embeds previous tokens, whether they are close or distant, as fixed-length vectors, without complexity dependence on the number of cached tokens. Consequently, GRC model token dependencies over a broader range of input, resulting in improved accuracy and performance across diverse Transformers-variants with different architectures and attention functions, on a variety of vision and language tasks.

³IWSLT14: German-English(De-En), Spanish-English(Es-En) and English-French(En-Fr), IWSLT15: German-English(De-En), English-Vietnamese(En-Vi) and Czech-English(Cs-En)

References

- Bulatov, A.; Kuratov, Y.; and Burtsev, M. 2022. Recurrent memory transformer. *Advances in Neural Information Processing Systems*, 35: 11079–11091.
- Burtsev, M. S.; Kuratov, Y.; Peganov, A.; and Sapunov, G. V. 2020. Memory transformer. *arXiv preprint arXiv:2006.11527*.
- Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; and Zagoruyko, S. 2020. End-to-end object detection with transformers. In *European conference on computer vision*, 213–229. Springer.
- Caron, M.; Misra, I.; Mairal, J.; Goyal, P.; Bojanowski, P.; and Joulin, A. 2020. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in Neural Information Processing Systems*, 33: 9912–9924.
- Cettolo, M.; Niehues, J.; Stüker, S.; Bentivogli, L.; Cattoni, R.; and Federico, M. 2015. The IWSLT 2015 Evaluation Campaign. In *Proceedings of the 12th International Workshop on Spoken Language Translation: Evaluation Campaign*, 2–14. Da Nang, Vietnam.
- Cettolo, M.; Niehues, J.; Stüker, S.; Bentivogli, L.; and Federico, M. 2014. Report on the 11th IWSLT evaluation campaign. In *Proceedings of the 11th International Workshop on Spoken Language Translation: Evaluation Campaign*, 2–17. Lake Tahoe, California.
- Cho, K.; Van Merriënboer, B.; Bahdanau, D.; and Bengio, Y. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*.
- Dai, Z.; Yang, Z.; Yang, Y.; Carbonell, J.; Le, Q. V.; and Salakhutdinov, R. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*.
- Deng, W.; Marsh, J.; Gould, S.; and Zheng, L. 2022. Fine-Grained Classification via Categorical Memory Networks. *IEEE Transactions on Image Processing*, 31: 4186–4196.
- Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; Uszkoreit, J.; and Houlsby, N. 2021. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations*.
- He, K.; Gkioxari, G.; Dollár, P.; and Girshick, R. 2017. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, 2961–2969.
- Kitaev, N.; Kaiser, Ł.; and Levskaya, A. 2020. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*.
- Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25.
- Kuhn, R.; and De Mori, R. 1990. A cache-based natural language model for speech recognition. *IEEE transactions on pattern analysis and machine intelligence*, 12(6): 570–583.
- Kupiec, J. 1989. Probabilistic models of short and long distance word dependencies in running text. In *Speech and Natural Language: Proceedings of a Workshop Held at Philadelphia, Pennsylvania, February 21-23, 1989*.
- Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; and Zitnick, C. L. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, 740–755. Springer.
- Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; and Guo, B. 2021. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 10012–10022.
- Long, A.; Yin, W.; Ajanthan, T.; Nguyen, V.; Purkait, P.; Garg, R.; Blair, A.; Shen, C.; and van den Hengel, A. 2022. Retrieval augmented classification for long-tail visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6959–6969.
- Nangia, N.; and Bowman, S. R. 2018. Listops: A diagnostic dataset for latent tree learning. *arXiv preprint arXiv:1804.06028*.
- Ott, M.; Edunov, S.; Baevski, A.; Fan, A.; Gross, S.; Ng, N.; Grangier, D.; and Auli, M. 2019. fairseq: A Fast, Extensible Toolkit for Sequence Modeling. In *NAACL-HLT (Demonstrations)*.
- Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *ACL*.
- Rae, J. W.; Potapenko, A.; Jayakumar, S. M.; and Lillicrap, T. P. 2019. Compressive transformers for long-range sequence modelling. *arXiv preprint arXiv:1911.05507*.
- Tay, Y.; Dehghani, M.; Abnar, S.; Shen, Y.; Bahri, D.; Pham, P.; Rao, J.; Yang, L.; Ruder, S.; and Metzler, D. 2021a. Long Range Arena : A Benchmark for Efficient Transformers. In *International Conference on Learning Representations*.
- Tay, Y.; Dehghani, M.; Aribandi, V.; Gupta, J.; Pham, P. M.; Qin, Z.; Bahri, D.; Juan, D.-C.; and Metzler, D. 2021b. Omnet: Omnidirectional representations from transformers. In *International Conference on Machine Learning*, 10193–10202. PMLR.
- Tu, Z.; Liu, Y.; Shi, S.; and Zhang, T. 2018. Learning to remember translation history with a continuous cache. *Transactions of the Association for Computational Linguistics*, 6: 407–420.
- Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.
- Wang, Q.; Li, B.; Xiao, T.; Zhu, J.; Li, C.; Wong, D. F.; and Chao, L. S. 2019. Learning deep transformer models for machine translation. *arXiv preprint arXiv:1906.01787*.
- Wang, S.; Li, B. Z.; Khabsa, M.; Fang, H.; and Ma, H. 2020a. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*.

- Wang, W.; Xie, E.; Li, X.; Fan, D.-P.; Song, K.; Liang, D.; Lu, T.; Luo, P.; and Shao, L. 2021. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 568–578.
- Wang, W.; Xie, E.; Li, X.; Fan, D.-P.; Song, K.; Liang, D.; Lu, T.; Luo, P.; and Shao, L. 2022. PVT v2: Improved baselines with Pyramid Vision Transformer. *Computational Visual Media*, 1–10.
- Wang, X.; Zhang, H.; Huang, W.; and Scott, M. R. 2020b. Cross-batch memory for embedding learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6388–6397.
- Wu, Y.; Rabe, M. N.; Hutchins, D.; and Szegedy, C. 2022. Memorizing transformers. *arXiv preprint arXiv:2203.08913*.
- Zaheer, M.; Guruganesh, G.; Dubey, K. A.; Ainslie, J.; Alberti, C.; Ontanon, S.; Pham, P.; Ravula, A.; Wang, Q.; Yang, L.; et al. 2020. Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems*, 33: 17283–17297.
- Zhong, Z.; Zheng, L.; Luo, Z.; Li, S.; and Yang, Y. 2019. Invariance matters: Exemplar memory for domain adaptive person re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 598–607.
- Zhu, C.; Ping, W.; Xiao, C.; Shoeybi, M.; Goldstein, T.; Anandkumar, A.; and Catanzaro, B. 2021. Long-short transformer: Efficient transformers for language and vision. *Advances in Neural Information Processing Systems*, 34.