Electronic Research Archive

https://www.aimspress.com/journal/era

ERA, 33(8): 4785–4798.
DOI: 10.3934/era.2025215
Received: 01 March 2025
Revised: 25 April 2025
Accepted: 09 May 2025
Published: 19 August 2025

Research article

Introducing attention shortcuts in convolutional neural networks

David Erroz* **and Mikel Galar**

Departamento de Estadística, Informática y Matemáticas, Institute of Smart Cities (ISC), Universidad Pública de Navarra (UPNA), Campus Arrosadia, Pamplona 31006, Spain

* **Correspondence:** Email: david.erroz@unavarra.es.

Abstract: It is a proven fact that nowadays, thanks to convolutional neural networks that implement skip connection mechanisms, we can train increasingly deeper, more accurate, and efficient networks. These networks successfully address the *degradation problem* previously experienced in very deep networks. Among the most popular are ResNets and DenseNets. ResNets introduce skip connections using summation, while DenseNets employ concatenation. The summation mechanism in ResNets can limit the adaptation of prior information to the specific needs of each layer. In contrast, the DenseNet concatenation mechanism can become computationally expensive as convolutional blocks attempt to process all accumulated prior information. Therefore, in this paper, we proposed a new attention-based skip connection mechanism: Attention Shortcuts. This mechanism allows convolutional blocks to process the most relevant prior information, reducing computational burden. We conducted a preliminary experimental study adapting the proposed mechanism to the ResNet-50 backbone and compared its performance to the original ResNet.

Keywords: skip connections; attention; shortcuts; convolutional neural networks; downsampling

1. Introduction

Recent advances in supervised machine learning (ML) have been largely driven by the development of very deep neural networks. Within the field of computer vision, these networks have achieved state-of-the-art performance [1–4]. Despite the promising emergence of Vision Transformers [5], convolutional neural networks (CNNs) remain the predominant models for pattern recognition tasks in computer vision. Although CNNs were introduced over 20 years ago [6], their success is a more recent phenomenon, enabled by our ability to train very deep CNN architectures [7, 8]. Since then, the trend has been to increase the size of these networks, surpassing the barrier of 100 and 200 layers [9, 10]. Evidence suggests that the levels of abstraction of features detected by networks can be enriched as their depth increases, making the depth of the network of

vital importance [8, 11].

However, simply adding more layers does not guarantee improved network performance, as optimization becomes increasingly complex. One key challenge is the problem of *vanishing/exploding* gradients [12, 13]. Various approaches have been developed to address this issue, including specialized initialization schemes [12] and intermediate normalization layers [14]. Additionally, the *degradation problem* [9] occurs due to the difficulty of approximating simple functions using complex networks. To address this, various architectures have been developed that use skip connections to shorten the path between distant network layers [9, 10, 15–17], enabling more effective training. Among them, the most popular is ResNet [9], which reuses previous features through identity shortcuts that are added to the outputs of each convolutional block with a simple summation operation.

The key to skip connection models is to provide each block with new and decisive information, while explicitly preserving the potentially useful information computed by previous layers. This approach enables the construction of states that consistently contain the most relevant information for the final prediction. Without skip connections, the network must implicitly preserve past information within its learnable transformations, whereas explicitly including this information through skip connections significantly facilitates learning in deep networks.

While ResNets improve the training of deep networks with their identity shortcuts, the direct addition approach limits control over the information flow, since each block must preserve all potentially useful information, regardless of its immediate relevance within that block. This characteristic makes ResNets somewhat similar to unrolled recurrent neural networks, although with different parameters per block, since both architectures follow the paradigm of storing all previous computations within a single state vector [18]. This limitation motivates the approach used in DenseNets [10], where concatenation replaces direct summation for adding shortcuts. This approach allows each block to select the most relevant information while still allowing subsequent blocks to have access to all previous information for future use.

However, the DenseNet approach comes at the cost of increased graphics processing unit (GPU) memory consumption. As the network grows deeper, convolutions operate on tensors with an ever-increasing number of elements and parameters. This makes training very deep networks with multiple kernels per layer impractical. To address this, the authors of DenseNets employ a limited number of kernels (only 12) and introduce transition blocks between dense blocks [10]. In this case, following the simile with sequence models, DenseNets would resemble feed-forward networks, where there is no state that collects all the previous information, but rather each of the elements serves directly as input to the network, that is, the different elements have different weights. In addition, appealing again to this simile, in view of the close conceptual relationship that exists between sequential networks and skip connection networks, in this work we want to raise the possibility of replacing the summation or concatenation mechanisms by one of the most recognized and used today in sequential networks, the attention mechanism [19, 20]. We hypothesize that incorporating the self-attention mechanism [20] into skip connections can achieve more precise information extraction for each block, without increasing the size of the inputs as one moves through the network and with a negligible increase in parameters, similar to what Transformers [20] achieve for sequence modeling. Therefore, this work aims to investigate the practical implications of the integration of a self-attention mechanism into skip connections, exploring whether extrapolating this mechanism for managing

shortcuts can provide advantages.

Prior work has already explored the integration of attention mechanisms within CNNs. Some approaches develop procedures that resemble attention mechanisms by extracting only the most relevant features from the activation maps computed by a CNN, using various modules or branches of feature map transformations [21, 22]. Others apply the self-attention mechanism to the spatial locations of a single feature map, enabling information sharing between distant regions [23]. However, to the best of our knowledge, none of these methods propose either self-attention or any form of attention mechanism applied across the different feature maps of a CNN to recover relevant information from earlier stages of computation. Thus, although the connection between attention mechanisms and CNNs has been examined, to our knowledge, their application within skip connections remains uninvestigated.

The rest of this paper is organized as follows. First, Section 2 examines related works, establishing the context for our research. Next, in Section 3, we introduce our proposed attention shortcuts (AS) mechanism and its underlying principles. Section 4 covers the experiments, including the models and baselines, the experimental framework, and the results. Finally, Section 5 presents the conclusions and future work.

2. Related works

2.1. Skip connection CNNs

Neural network architecture design remains a major topic of interest in deep learning research. The goal is to find network structures that consistently improve performance across different tasks. Within computer vision, architectures are usually evaluated in standard classification benchmarks based on popular datasets such as ImageNet [24] or CIFAR [25]. Pioneering deep learning architectures in vision include AlexNet [7], VGGNet [8], and GoogleNet along with its variants [15, 26]. The latter architectures already started to introduce shortcut branches along with few deeper branches. Previously, skip connections had already been studied [27, 28], although they gained popularity with the introduction of highway networks [17]. These were among the first to demonstrate their potential for efficient end-to-end training of very deep neural networks, with more than 100 layers.

Inspired by long short-term memory (LSTM) recurrent networks [29], highway networks employ a learned gating mechanism to regulate the flow of information between newly generated data at each layer and that coming from skip connections, enabling efficient optimization. ResNets [9] buid upon this concept, eliminating logic gates and ensuring persistent skip connections. This allows them to learn residual functions within each convolutional block, relative to that block's input. This approach achieved groundbreaking results in image recognition, localization, and detection tasks. Its impact is such that ResNets remain a popular choice for many computer vision problems. Many methods have been inspired by ResNets. For example, networks with stochastic depth [30] improved their learning by randomly dropping layers, highlighting the inherent redundancy within these networks. DenseNets [10] emerged as an alternative approach, maximizing feature reuse by introducing skip connections that concatenate the outputs from all previous blocks. With a different approach, FractalNet [31] also trained very deep networks, avoiding both the usage of dense concatenation and identity shortcut summation.

Overall, DenseNets and particularly ResNets have become the most popular CNN architectures

within the research community due to their competitive performance and design simplicity. Analyses comparing their theoretical foundations highlight a key difference: ResNets share convolutional parameters across outputs of preceding blocks, while DenseNets do not [32]. Building upon this analysis, DSNets [32] introduce a hybrid approach. They densely connect outputs from all convolutional blocks but reuse the same block parameters. DSNets achieve this by applying channel-wise weights to each shortcut, subsequently summing them with the current output.

2.2. Attention mechanism

The attention mechanism was designed to improve the performance of *encoder-decoder* models for machine translation. It aimed to address the bottleneck issue in RNNs, where the *decoder* receives only a fixed-length vector representing all information encoded by the *encoder*, regardless of the input sequence's length [19]. The mechanism involves accessing each of the *encoder*'s hidden states and calculating their relative alignment with the current *decoder* state. This allows the construction of a contextualized state containing the most relevant information at each decoding step.

However, years later a more general attention mechanism was proposed that allowed application not only between sequences (from *decoder* to *encoder*), but also within the same sequence [20]. This is termed *self-attention*. In self-attention, we start with a set of *queries* Q, *keys* K, and *values* V with dimensions d_k, d_k, and d_v, respectively. These are derived from each individual element of the sequence through the application of linear projections W^Q, W^K, W^V. The contextualized states Z are calculated by a linear combination of the values V, whose weights are determined by the similarity between each query (Q) and the different keys (K). Mathematically, this is expressed as:

$$Z = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V\,. \tag{2.1}$$

Furthermore, in practice, it is beneficial to apply the attention mechanism multiple times with different linear projections each time. This is known as *multi-head self-attention*. In this case, to maintain computational efficiency, the dimensions of Q, K, and V are often reduced for each attention head. The results of these multiple heads are concatenated and projected back to the original space.

3. Proposal

Although ResNets and DenseNets are considered the two most representative CNNs using skip connections, a common misconception exists regarding their exact operation. Initially, ResNets were believed to only incorporate the prior activation map, while DenseNets used all the preceding blocks. However, research has demonstrated that both architectures share the philosophy of dense connectivity to all previous activation maps [32–34]. The critical distinction lies solely in the method used to combine these connections: summation (ResNets) or concatenation (DenseNets). While summation can limit information preservation and on-demand access for each block, concatenation offers a more comprehensive approach at the cost of increased memory usage and parameters.

Therefore, based on the strengths and weaknesses of current skip connection mechanisms (summation and concatenation), we draw inspiration from the Transformer architecture [20] to propose the AS mechanism, which leverages self-attention to regulate how skip connections are integrated within the network. This approach aims to ensure that each block selectively extracts the

most relevant information from previous activations, while critically preserving discarded information for potential use by subsequent blocks. This selectivity potentially offers more flexible feature integration than summation or concatenation mechanisms and is achieved without incurring a significant increase in network parameters. Importantly, our proposed AS mechanism is applicable to a wide range of network backbones with minimal modifications.

We will start by defining terms. Let us consider an image X_0 as input to a convolutional network. The network comprises L convolutional layers or blocks, each represented by a nonlinear transformation $H_l(\cdot)$ (where l is the block index). Note that $H_l(\cdot)$ may consist of various sublayers, including convolutional layers, ReLU activations [35], batch normalization (BN) [14], and pooling layers [36]. The output of block l (defined by H_l) is denoted as X_l.

In traditional CNNs, each block's input is derived solely from the immediately preceding block's output: $X_l = H_l(X_{l-1})$. Skip connections introduce a modification, allowing blocks to incorporate information from multiple prior outputs. Let us represent this with a function g that operates on the set of preceding outputs, $X_{1:l-1}$. The modified relationship is then expressed as:

$$X_l = H_l(g(X_{1:l-1}))\,. \tag{3.1}$$

In ResNets, g represents the summation of all inputs. In DenseNets, g is the concatenation function. Alternatively, g could also implement a channel-weighted summation [32]. With our proposed method, g applies the *multi-head self-attention* mechanism. Here, the *query* (Q) is derived from the most recent output X_{l-1}, while the *keys* (K) and *values* (V) are obtained from all previous elements $X_{1:l-1}$. The attention is calculated spatially across all pixels. The function g is defined as follows:

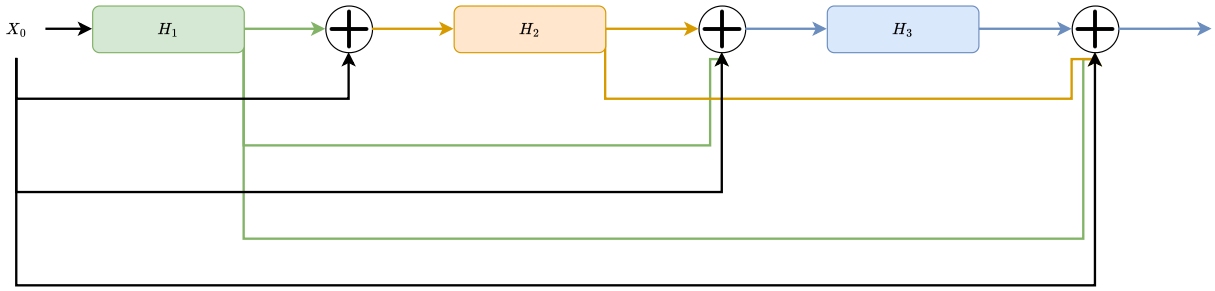
$$\begin{aligned} g(X_{1:l-1}) &= \text{Concat}(\text{head}_1, ..., \text{head}_h)W^O \\ \text{head}_k &= \text{softmax}\left(\frac{(X_{l-1}W_k^Q)(X_{1:l-1}W_k^K)^T}{\sqrt{c/h}}\right)(X_{1:l-1}W_k^V) \end{aligned} \tag{3.2}$$

where c represents the number of channels in the activation maps (which depends on the adopted CNN structure), and $W^O \in \mathbb{R}^{c\times c}$, $W_k^Q \in \mathbb{R}^{c\times(c/h)}$, $W_k^K \in \mathbb{R}^{c\times(c/h)}$, and $W_k^V \in \mathbb{R}^{c\times(c/h)}$ are matrices of trainable parameters. The number of heads h is a tunable hyperparameter, set by default to 4 based on experimental tests. Figure 1 provides a schematic comparison of this method with ResNet and DenseNet skip connection architectures.

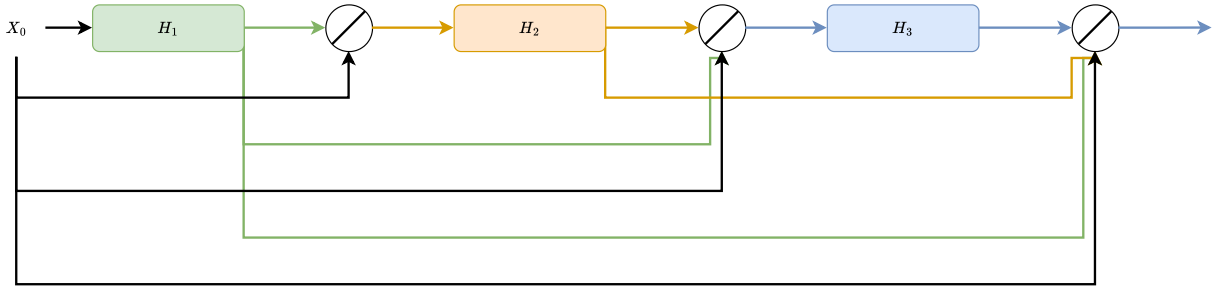
Our mechanism allows each block to selectively incorporate information from previous outputs. For each spatial position, attention is applied to all prior outputs $X_{1:l-1}$, with the current output acting as the query. To maintain consistency, all outputs $X_{1:l-1}$ must share the same spatial dimensions $H \times W$. Deep CNNs, however, progressively reduce spatial resolution while increasing feature channels. This mismatch is typically addressed in architectures like ResNets via 1×1 convolutions with a stride of 2. Applying our attention-based mechanism is less straightforward, as it would require including additional projection matrices (W^Q, W^K, W^V, W^O) to handle different numbers of channels. However, our aim is to intelligently manage information flow while minimizing the number of additional parameters required. Therefore, we explore two strategies to handle downsampling within this framework:

1) *Downsampling of outputs*: All convolutional block outputs are projected into a common channel dimension c. Spatial dimensions are adapted using 1×1 pooling (*stride* of 2) when necessary.

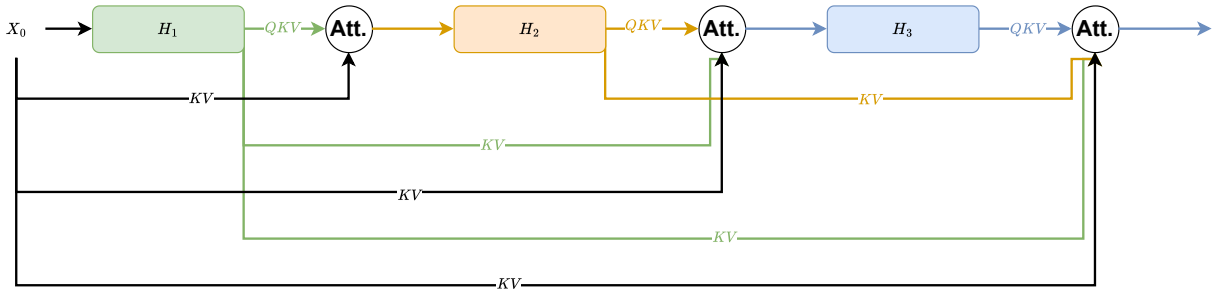
2) *Downsampling of keys and values*: *Keys* and *values* are derived directly from outputs X_l, inheriting their dimensions. Previous *keys* and *values* are adapted via 1×1 convolutions (*stride* of 2) to match dimensions when downsampling occurs.



(a) Summation (+)-ResNets.



(b) Concatenation (/)-DenseNets.



(c) Attention shortcut (AS)-Proposal.

Figure 1. Schematic diagram comparing the AS mechanism with the summation (+) and concatenation (/) mechanisms of ResNets and DenseNets, respectively.

From an implementation perspective, storing the projected outputs (as $K_{1:l-1}, V_{1:l-1}$) offers significant advantages over storing $X_{1:l-1}$. This strategy eliminates the need to redundantly recalculate previous *keys* and *values*, reducing computational overhead. Therefore, within each block, we only apply the projection matrices (W^Q, W^K, W^V) to the most recent output X_{l-1}, substantially reducing the computational cost.

An important consideration before applying attention is the normalization of outputs (X_l). Without normalization, the mechanism might prioritize information based solely on vector magnitudes, leading to undesired behavior. The core function of attention relies on the alignment of vectors, meaning the relative direction rather than absolute length is crucial. Therefore, normalization techniques that preserve vector orientation are preferred. Consistent with Transformers [20], we adopt layer normalization (LN) [37] for this purpose.

3.1. Computational burden

To better understand the efficiency of the proposed AS mechanism, we present a formal analysis of its computational complexity, expressed in floating-point operations (FLOPs). We compare this cost to that of summation-based skip connections (ResNets) and concatenation-based skip connections (DenseNets). For simplicity, we ignore downsampling operations throughout the networks, as their contribution in terms of complexity is neglible in comparison. Additionally, we treat each block of the CNN, $H_l(\cdot)$, as a simple convolutional layer with kernel size $k \times k$. We continue to assume that the activation maps $X_{1:l-1}$ have shape $(H \times W \times c)$.

Starting with AS, we can unfold Eq (3.2) and divide the mechanism into four components:

1) Linear projections for extracting Q, K, V: $\text{FLOPs}_{\text{proj}} = O(3HWc^2) = O(HWc^2)$
2) Attention score computation: $\text{FLOPs}_{\text{attnScores}} = O(HW(l-1)c) = O(HWlc)$
3) Attention-weighted sum with V: $\text{FLOPs}_{\text{attn}} = O(HW(l-1)c) = O(HWlc)$
4) Final output projection W^O: $\text{FLOPs}_{\text{out}} = O(HWc^2)$

Summing all components and adding the complexity due to the convolution operation, we obtain the total FLOPs per layer:

$$\text{FLOPs}_{\text{l}} = O(\underbrace{2HWc^2 + 2HWlc}_{\text{Attn FLOPs}} + \underbrace{HWk^2c^2}_{\text{Conv FLOPs}}) = O(HWc(c + l + k^2c)) = O(HWc(l + k^2c))$$

Aggregating over all L layers of the network, and assuming that the few downsampling operations scale dimensions by constant factors, we obtain the total complexity of an AS-Net:

$$\begin{aligned}\text{FLOPs}_{\text{AS-Net}} &= O\left(\sum_{l=1}^{L}\left(HWc(l + k^2c)\right)\right) = O\left(HWc\left(\sum_{l=1}^{L} l + Lk^2c\right)\right) \\ &= O\left(HWc\left(\frac{L(L+1)}{2} + Lk^2c\right)\right) = O\left(HWc(L^2 + Lk^2c)\right) = O\left(HWLc(L + k^2c)\right).\end{aligned} \tag{3.3}$$

A similar analysis for concatenation-based skip connection CNNs (DenseNets) yields:

$$\text{FLOPs}_{\text{DenseNet}} = O\left(\sum_{l=1}^{L}(HWk^2(l-1)c^2)\right) = O\left(HWk^2c^2L^2\right) \tag{3.4}$$

and for summation-based skip connection CNNs (ResNets):

$$\text{FLOPs}_{\text{ResNet}} = O\left(HWk^2c^2L\right). \tag{3.5}$$

Clearly, summation-based skip connections are the most computationally efficient. Between concatenation-based and attention-based mechanisms, since $(L + k^2c) < Lk^2c$, AS is more efficient. This result supports our objective of developing a method that combines the advantages of both approaches—the selectivity and feature enrichment of concatenation, along with an improved scalability in FLOPs, characteristic of the summation-based mechanism.

4. Experiments

In this section, we present the experiments designed to fairly evaluate the AS mechanism. Different configurations of AS are explored within a defined evaluation framework, enabling a proper comparison against existing skip connection mechanisms.

4.1. Models and baselines

To evaluate the effectiveness of our AS mechanism, we will compare its performance against ResNet, which is the most widespread skip connection mechanism. Since our approach is flexible and can be integrated with almost any CNN backbone, we opted for the ResNet-50 architecture for this study motivated by its extensive use in image classification tasks [32, 38]. Here, we aim to incorporate the AS mechanism within the fundamental building block structure of ResNet-50, minimizing modifications and parameter growth. This strategy ensures that the primary differentiating factor between the compared models is the skip connection mechanism itself.

To implement the two downsampling alternatives proposed earlier (Section 3), we modify the structure of ResNet-50 as follows:

1) *Downsampling of outputs* (AS_{out}-Net-50): To ensure a fair comparison with the ResNet-50 baseline, we avoid increasing network depth. Instead, we replace the existing 1×1 convolutions in ResNet-50's bottleneck blocks with those operating on a fixed number of output channels. To manage the computational cost, we have chosen a moderate channel dimension of $c = 128$.

2) *Downsampling of keys and values* (AS_{KV}-Net-50): This approach requires a distinct set of projection matrices (W^O, W^K, W^Q, W^V) for each *stage*, where dimensions change. To compensate for the increased parameter and computational overhead inherent to this strategy, the number of output channels (dimension of X_l) is reduced to one-quarter of the original ResNet-50 configuration.

In both variants, we replace the original BN layer following the last convolution of each block with LN. Table 1 provides a detailed comparison between the original ResNet-50 backbone and its modifications incorporating AS with the two downsampling strategies. This comparison includes the specific layer configurations and highlights the total number of parameters for each model.

In addition to the two variants explored based on different downsampling strategies, we consider a third variant: $\text{AS}_{\text{stages}}$-ResNet-50. This approach preserves the original ResNet-50 backbone entirely, including residual connections, while incorporating the AS mechanism between stages. Therefore, each stage of the network is treated as a single block for the purposes of attention. The motivation for this approach is to increase the distinctiveness of the inputs fed to the attention module, potentially boosting its effectiveness. Additionally, this strategy offers a reduced computational overhead compared to finer-grained attention within each block.

Table 1. Comparison of the ResNet-50 backbone and the proposed AS variants with different downsampling strategies.

	ResNet-50	**AS$_{\text{out}}$-Net-50**	**AS$_{\text{KV}}$-Net-50**
conv 1	$3 \times 3, 64$	$3 \times 3, 128$	$3 \times 3, 64$
stage 1	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 128 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 64 \end{bmatrix} \times 3$
stage 2	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 128 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 128 \end{bmatrix} \times 4$
stage 3	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 128 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 256 \end{bmatrix} \times 6$
stage 4	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 128 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 512 \end{bmatrix} \times 3$
# Params	23.5 M	12.7 M	16.2 M

We will compare the three exposed AS variants with the original ResNet-50. Following [32], we will add a width-reduced version of ResNet-50 (0.5×), aligning its parameter count more closely with the first two AS variants. Finally, we will include a Plain-ResNet-50 without skip connections, enabling a direct assessment of the validity of skip connection mechanisms.

4.2. Experimental framework

To ensure fairness in the comparison, we established a unified experimental framework for all models and baselines, maintaining consistent configurations across them:

Datasets. We used the CIFAR datasets [25] for evaluation. Both datasets contain 60,000 color images of size 32×32 pixels, split into 50,000 training images and 10,000 testing images. CIFAR-10 offers 10 classes, while CIFAR-100 has 100 fine-grained classes.

Model training. For training we followed the configuration in the original ResNets [9]. Following standard practices, all image inputs (both training and testing) were normalized using the mean and standard deviation values calculated from the entire training set. Data augmentation was applied with a 4-pixel padding followed by a random 32×32 crop of the padded image or its horizontal flip [39]. Training was done for $64k$ iterations with a mini-batch size of 128 using SGD with a momentum of 0.9 and initial learning rate lr of 0.1, divided by 10 at $32k$ and $48k$ iterations. We used a weight decay of 1×10^{-4}.

Evaluation. Models were trained on the CIFAR-10 and CIFAR-100 training sets. To assess their generalization performance, we evaluated them on the corresponding test sets. The top-1 error rate

was chosen as the primary evaluation metric. Model evaluation employed 5 independent runs per model, with the final metrics reported as the mean and standard deviation.

4.3. Results

Table 2 summarizes the evaluation results for the methods in the comparison (three AS variants and the baselines). Both the original ResNet-50 and the reduced-parameter ResNet-50 0.5× outperform all models with only AS as the skip connection mechanism. This suggests that the performance difference is not only related to parameter count, but that the AS mechanism is being less effective. Additionally, the significant gap between Plain-ResNet-50 and the rest of the models highlights the importance of skip connections. Even in the case of AS variants, we can observe a significant improvement, showing that the AS mechanism is contributing to the overall performance of the network, although not achieving the performance of ResNet-50.

Table 2. Test classification error (%) on CIFAR 10 and 100 for the different variants (Mean ± SD).

Model	**CIFAR 10**	**CIFAR 100**
AS_{out}-Net-50	5.63 ± 0.13	24.72 ± 0.35
AS_{KV}-Net-50	5.97 ± 0.10	24.92 ± 0.51
AS_{stages}-ResNet-50	5.06 ± 0.09	22.47 ± 0.34
ResNet-50	5.10 ± 0.10	22.12 ± 0.38
ResNet-50 0.5×	5.74 ± 0.14	24.08 ± 0.26
plain-ResNet-50	8.41 ± 0.20	40.02 ± 2.15

Interestingly, AS_{out}-Net-50 and AS_{KV}-Net-50 exhibit similar performance, suggesting that for this specific network backbone, the downsampling strategy within the AS mechanism might not be a critical factor. Furthermore, AS_{stages}-ResNet-50's performance closely mirrors that of ResNet-50, indicating that incorporating attention between stages, while preserving residual connections, has minimal positive or negative effect on the model.

For a more detailed analysis, we tracked the evolution of the test error rate throughout training. Figure 2 presents plots of the test error (%) per epoch for both the CIFAR-10 and CIFAR-100 datasets. The plots reveal a similar evolution across all models except for Plain-ResNet-50, which lags behind from the beginning. This observation reinforces two key points: first, the AS mechanism can be an effective alternative to skip connections, and second, it highlights the importance of incorporating skip connections within CNN architectures, which make a difference.

Overall, within this experimental setup using ResNet architectures, the proposed attention mechanism does not yield significant performance improvements compared to the simpler summation mechanism in ResNets. This might be partially due to our adaptation to the ResNet structure, which may not be optimal for the AS mechanism. Also, although the AS mechanism offers more flexible feature integration, its lower inductive bias compared to fixed-summation in ResNets might lead to suboptimal convergence, especially in low-resolution tasks like CIFAR. Future exploration of architectures specifically designed for attention-based shortcuts, as well as evaluations on higher-resolution datasets, could be a promising direction for further research.

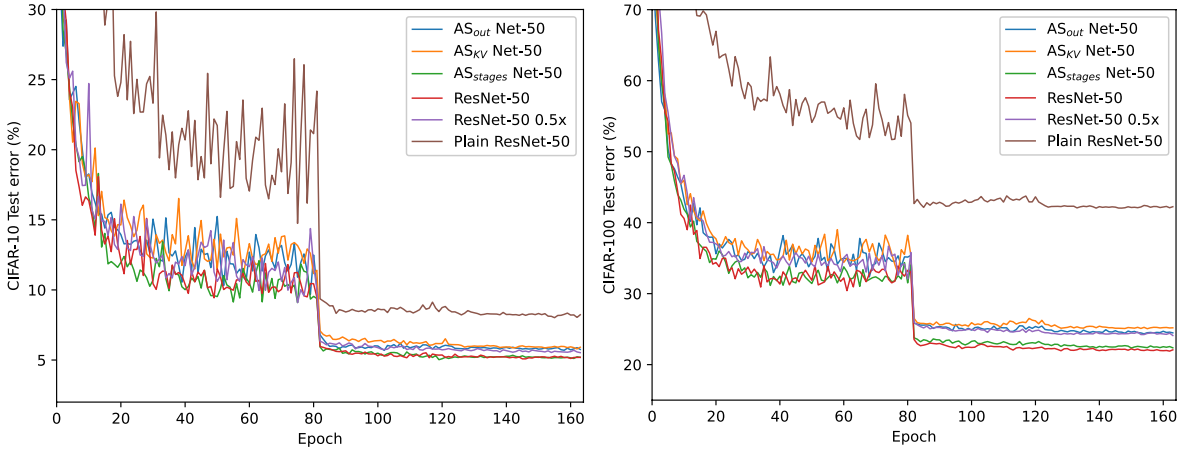


Figure 2. Test error (%) evolution on CIFAR-10 (left) and CIFAR-100 (right) with the six models compared.

5. Conclusions and future work

In this work, we proposed a novel skip connection mechanism, AS, inspired by the self-attention of Transformers. This mechanism aims to address limitations observed in popular skip connection architectures like ResNets and DenseNets. We conducted an initial evaluation by integrating AS into a ResNet backbone. While the results demonstrate the mechanism's viability as a skip connection method, its performance currently falls short of ResNets, and it introduces additional complexity.

However, it is important to consider that our evaluation employed low-resolution datasets and models with a moderate number of layers. Additionally, the ResNet backbone was not specifically designed for AS. Therefore, under different conditions, we might observe more competitive performance.

As future work, it would be valuable to extend the study to larger networks, incorporate diverse datasets, and most importantly, investigate network structures specifically tailored to exploit the strengths of AS.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgements

This work was funded by a predoctoral fellowship from the Research Service of the Universidad Publica de Navarra (predoctoral contract of David Erroz) and the Spanish MCIN (PID2022-136627NB-I00/AEI/10.13039/501100011033 FEDER, UE).

Conflict of interest

The authors declare there are no conflicts of interest.

References

1. C. Zhang, F. Rameau, J. Kim, D. M. Argaw, J. Bazin, I. S. Kweon, DeepPTZ: Deep self-calibration for PTZ cameras, in *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, (2020), 1030–1038. https://doi.org/10.1109/WACV45572.2020.9093629
2. A. Esteva, K. Chou, S. Yeung, N. Naik, A. Madani, A. Mottaghi, et al., Deep learning-enabled medical computer vision, *npj Digital Med.*, **4** (2021), 5. https://doi.org/10.1038/s41746-020-00376-2
3. Y. Feng, B. Chen, T. Dai, S. Xia, Adversarial attack on deep product quantization network for image retrieval, in *Proceedings of the AAAI conference on Artificial Intelligence*, **34** (2020), 10786–10793. https://doi.org/10.1609/aaai.v34i07.6708
4. Z. Zou, K. Chen, Z. Shi, Y. Guo, J. Ye, Object detection in 20 years: A survey, in *Proceedings of the IEEE*, **111** (2023), 257–276. https://doi.org/10.1109/JPROC.2023.3238524
5. A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, et al., An image is worth 16×16 words: Transformers for image recognition at scale, preprint, arXiv:2010.11929.
6. Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, et al., Backpropagation applied to handwritten zip code recognition, *Neural Comput.*, **1** (1989), 541–551. https://doi.org/10.1162/neco.1989.1.4.541
7. A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, *Commun. ACM*, **60** (2017), 84–90. https://doi.org/10.1145/3065386
8. K. Simonya, A. Zisserman, Very deep convolutional networks for large-scale image recognition, preprint, arXiv:1409.1556.
9. K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (2016), 770–778. https://doi.org/10.1109/CVPR.2016.90
10. G. Huang, Z. Liu, L. Van Der Maaten, K. Q. Weinberger, Densely connected convolutional networks, in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (2017), 2261–2269. https://doi.org/10.1109/CVPR.2017.243
11. M. D. Zeiler, R. Fergus, Visualizing and understanding convolutional networks, in *Computer Vision-ECCV 2014*, **8689** (2014), 818–833. https://doi.org/10.1007/978-3-319-10590-1_53
12. X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, (2010), 249–256.
13. Y. Bengio, P. Simard, P. Frasconi, Learning long-term dependencies with gradient descent is difficult, *IEEE Trans. Neural Networks*, **5** (1994), 157–166. https://doi.org/10.1109/72.279181
14. S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, in *Proceedings of the 32nd International Conference on Machine Learning*, (2015), 448–456.
15. C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, et al., Going deeper with convolutions, in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (2015), 1–9. https://doi.org/10.1109/CVPR.2015.7298594

16. T. Raiko, H. Valpola, Y. LeCun, Deep learning made easier by linear transformations in perceptrons, in *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics*, (2012), 924–932.
17. R. K. Srivastava, K. Greff, J. Schmidhuber, Highway networks, preprint, arXiv:1505.00387.
18. Q. Liao, T. Poggio, Bridging the gaps between residual learning, recurrent neural networks and visual cortex, preprint, arXiv:1604.03640.
19. D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, preprint, arXiv:1409.0473.
20. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, et al., Attention is all you need, in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, (2017), 6000–6010.
21. F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, et al., Residual attention network for image classification, in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (2017), 6450–6458. https://doi.org/10.1109/CVPR.2017.683
22. M. Jaderberg, K. Simonyan, A. Zisserman, K. Kavukcuoglu, Spatial transformer networks, in *Proceedings of the 29th International Conference on Neural Information Processing Systems*, **2** (2015), 2017–2025.
23. H. Zhang, I. Goodfellow, D. Metaxas, A. Odena, Self-attention generative adversarial networks, in *Proceedings of the 36th International Conference on Machine Learning*, (2019), 7354–7363.
24. J. Deng, W. Dong, R. Socher, L. Li, K. Li, L. Fei-Fei, ImageNet: A large-scale hierarchical image database, in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, (2009), 248–255. https://doi.org/10.1109/CVPR.2009.5206848
25. A. Krizhevsky, Learning Multiple Layers of Features from Tiny Images, 2009. Available from: https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf
26. C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (2016), 2818–2826. https://doi.org/10.1109/CVPR.2016.308
27. C. M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, 1995. https://doi.org/10.1093/oso/9780198538493.001.0001
28. W. N. Venables, B. D. Ripley, *Modern applied statistics with S* , 4*th* edition, Springer, 2002.
29. S. Hochreiter, J. Schmidhuber, Long short-term memory, *Neural Comput.*, **9** (1997), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735
30. G. Huang, Y. Sun, Z. Liu, D. Sedr, K. Q. Weinberger, Deep networks with stochastic depth, in *Computer Vision-ECCV 2016*, **9908** (2016), 646–661. https://doi.org/10.1007/978-3-319-46493-0_39
31. G. Larsson, M. Maire, G. Shakhnarovich, Fractalnet: Ultra-deep neural networks without residuals, preprint, arXiv:1605.07648.
32. C. Zhang, P. Benz, D. M. Argaw, S. Lee, J. Kim, F. Rameau, et al., Resnet or densenet? Introducing dense shortcuts to resnet, in *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, (2021), 3549–3558. https://doi.org/10.1109/WACV48630.2021.00359

33. Y. Chen, J. Li, H. Xiao, X. Jin, S. Yan, J. Feng, Dual path networks, in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, (2017), 4470–4478.

34. W. Wang, X. Li, T. Lu, J. Yang, Mixed link networks, in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, (2018), 2819–2825. https://doi.org/10.24963/ijcai.2018/391

35. X. Glorot, A. Bordes, Y. Bengio, Deep sparse rectifier neural networks, in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, (2011), 315–323.

36. Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, *Proceedings of the IEEE* , **86** (1998), 2278–2324. https://doi.org/10.1109/5.726791

37. J. L. Ba, J. R. Kiros, G. E. Hinton, Layer normalization, preprint, arXiv:1607.06450.

38. S. Mascarenhas, M. Agarwal, A comparison between vgg16, vgg19 and resnet50 architecture frameworks for image classification, in *2021 International Conference on Disruptive Technologies for Multi-Disciplinary Research and Applications (CENTCON)*, (2021), 96–99. https://doi.org/10.1109/CENTCON52345.2021.9687944

39. C. Lee, S. Xie, P. Gallagher, Z. Zhang, Z. Tu, Deeply-supervised nets, in *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, (2015), 562–570.

AIMS Press