information

MDPI

Article

Local Multi-Head Channel Self-Attention for Facial Expression Recognition

Roberto Pecoraro *, Valerio Basile and Viviana Bono

Department of Computer Science, University of Turin, C.so Svizzera 185, 10147 Turin, Italy; valerio.basile@unito.it (V.B.); viviana.bono@unito.it (V.B.)
* Correspondence: roberto.pecoraro@unito.it

Abstract: Since the Transformer architecture was introduced in 2017, there has been many attempts to bring the self-attention paradigm in the field of computer vision. In this paper, we propose *LHC*: Local multi-Head Channel self-attention, a novel self-attention module that can be easily integrated into virtually every convolutional neural network, and that is specifically designed for computer vision, with a specific focus on facial expression recognition. *LHC* is based on two main ideas: first, we think that in computer vision, the best way to leverage the self-attention paradigm is the channel-wise application instead of the more well explored spatial attention. Secondly, a local approach has the potential to better overcome the limitations of convolution than global attention, at least in those scenarios where images have a constant general structure, as in facial expression recognition. *LHC-Net* achieves a new state-of-the-art in the FER2013 dataset, with a significantly lower complexity and impact on the "host" architecture in terms of computational cost when compared with the previous state-of-the-art.

Keywords: self-attention; facial expression recognition; convolutional neural networks; computer vision

Citation: Pecoraro, R.; Basile, V.; Bono, V. Local Multi-Head Channel Self-Attention for Facial Expression Recognition. *Information* **2022**, *13*, 419. https://doi.org/10.3390/info13090419

Academic Editor: Gholamreza Anbarjafari (Shahab)

Received: 11 August 2022
Accepted: 30 August 2022
Published: 6 September 2022

1. Introduction

Facial Emotion Recognition (FER) is the task of identifying the emotion expressed by the facial expression of a person in a photo or video frame. FER is deeply connected to facial expression recognition [1], and it can be considered a specialized version of the latter task.

The aim of this work is to explore the capabilities of the self-attention paradigm in the context of computer vision, and in particular, in facial expression recognition. We introduce a new channel self-attention module, the *Local (multi-)Head Channel (LHC)*, which is thought of as being a processing block to be integrated into a pre-existing convolutional architecture.

LHC inherits the basic skeleton of the self-attention module from the well-known *Transformer* architecture by Vaswani et al. [2], with a new design aimed at improving it and adapting it to a computer vision pipeline. We call the final architecture *LHC-Net*: Local multi-Head Channel self-attention Network.

LHC-Net should be generally considered as a family of neural network architectures having a backbone represented by a convolutional neural network in which one or more *LHC* modules are integrated. More specifically, in this paper, we will refer to *LHC-Net* as a *ResNet34* (short for *Residual Network*) [3], having five *LHC* modules integrated, as shown in Figure 1.

Although the local approach could be hypothetically applied to several computer vision tasks, we found it most suitable for facial expression recognition, probably due to its ability to better exploit the constant structure of the images. In facial expression recognition, it is likely that, for example, the eyes will always appear in the upper portion of the image.

The local approach, especially in our implementation with horizontal stripes, seems capable for exploiting this aspect of the problem.

In the experiments presented in this paper, *LHC-Net* achieves the current single deep learning model state-of-the-art classification accuracy, both with and without test time augmentation, and with a computational cost which is only a fraction of the previous state-of-the-art (SOTA) architecture.

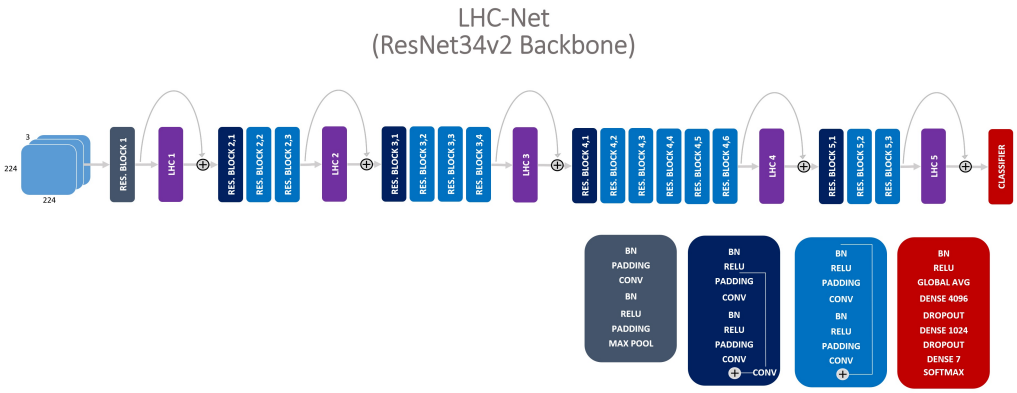

Figure 1. Five *LHC* modules integrated into a *ResNet34v2* architecture. Every module features a residual connection to obtain an easier integration, especially when pre-training is used for the backbone architecture.

2. Related Work

A task related to Facial Expression Recognition is Face Recognition, where the identity of the person is to be determined, rather than the expression. Several techniques have been proposed in the recent Machine Learning (ML) literature for Face Recognition, including Probabilistic Face Embeddings [4,5], Loss Function Search [6], and Lie Algebra Residual Networks [7]. Attention-based mechanisms have also been proposed, to enhance the learning towards Face Recognition, including based on depth [8] and the simulation of the human visual cortex and perception [9]. To our knowledge, this is the first work proposing *self*-attention as a mechanism to improve learning face representations.

2.1. Attention

The attention paradigm became very popular in the last few years with a large variety of mechanics and implementations in both Natural Language Processing (NLP) and computer vision scenarios. There are two main attention paradigms: either we pay attention to the data, with the idea of enhancing the most meaningful aspects, or we can try to exploit the inner relationships within these aspects, in order to produce a more meaningful representation of the data. The latter approach is usually called *self-attention*, because in some way, the data pay attention to themselves.

The first approach was introduced in 2014 by Bahdanau et al. [10] and was updated by Luong et al. in 2015 [11]. The proposed solutions were used in neural machine translation and integrated in a classic "seq to seq" encoder/decoder architecture, in which the decoder learns what the outputs of the encoder are, and where to pay more attention dinamically. Self-attention was introduced in 2017 by Vaswani et al. [2] (again, for neural machine translation), and it is considered by many as the greatest breakthrough technology in Artificial Intelligence (AI) since backpropagation was introduced in 1986 [12]. It fully replaced the previous state-of-the-art technologies, and the recurrent and convolutional paradigms, in NLP.

Since then, there has been many attempts to bring self-attention in computer vision, but as of now, with only partial success. Opposite to the NLP case, in computer vision, self-attention struggles to outperform the SOTA architectures like the classical *Inception* [13], *ResNet* [3], *VGG* [14], etc.

In computer vision, there are several type of attention paradigms. For clarity, from now on, we will use the following nomenclature:

- **Global Attention**: is usually only a module used before another model with the idea to enhance the important parts of an image.
- **Spatial Attention**: the attention modules focus on single pixels or areas of the feature maps.
- **Channel Attention**: the attention modules focus on entire feature maps.
- **Self-Attention**: the attention tries to find relationships between different aspects of the data.
- **Stand-Alone Attention**: the architecture is aimed at fully replacing the convolutional blocks and at defining a new processing block for computer vision based on some attention mechanism (mostly self-attention).

Xu et al. proposed a global attention module for medical image classification [15], this module pre-processes images, enhancing important areas pixel-by-pixel, before feeding them into a standard convolutional neural network. This kind of pre-processing is thought to make the following convolution processing more robust . It could be associated with the one proposed by Jaderberg et al. [16], which attempts to compensate for the lack of rotation/scaling invariance of the convolution paradigm. The proposed module learns a sample-dependant affine transformation to be applied to images, in order to make them centered and properly scaled/rotated.

The channel approach that we propose in this paper, despite being relatively unexplored in our self-attention mode, is instead very popular when associated with vanilla attention. Hu et al. proposed the *SE-Net* (Squeeze and Excitation) [17], a simple and effective module that enhances the more important features of a convolutional block. Squeeze and excitation has lately become a key module in the very popular *Efficient-Net* by Tan et al. [18], which set a new SOTA on several benchamrk datasets. Similarly, Woo et al. proposed the *CBAM* (Convolutional Block Attention Module), a sequential module composed of a spatial and a channel attention sub-module [19]. There are other examples of channel and spatial vanilla attention: *ECA-Net* (Efficient Channel Attention) by Wang et al. [20] is a new version of Squeeze and Excitation; *SCA-CNN* (Spatial and Channel-wise attention Convolutional Neural Network) proposed by Chen et al. [21] combines both spatial and channel vanilla attention for image captioning. *URCA-GAN* (UpSample Residual Channel-wise Attention) by Nie et al. [22] is a GAN (Generative Adversarial Network) featuring a residual channel attention mechanism thought for image-to-image translation.

Channel attention was not only used in vanilla approaches; similar to our architecture, Fu et al., Liu et al., and Tian et al. proposed self-attention architectures [23–25], respectively, for scene segmentation, feature matching between pairwise images, and video segmentation. The main differences between these modules and ours are the following:

- In all of them, channel attention always has a secondary role and there is always a spatial attention sub-module with a primary role;
- In all of them, the crucial multi-head structure is lacking;
- All of them implement channel attention as a "passive" non-learning module;
- None of them integrates our local spatial behavior for channel attention;
- None of them integrates our dynamic scaling, which is very specific of our architecture.

Opposite to channel self-attention, spatial self-attention is widely explored, and in most cases with the ambitious goal of totally replacing the convolution in computer vision, just like Vaswani's Transformer made Long Short-Term Memory (LSTM) networks obsolete. Bello et al. proposed an attention-augmented convolutional network [26] in which Vaswani's self-attention is straightforwardly applied to pixels representations and integrated in a convolutional neural network.

Similarly, Wu et al. proposed the Visual Transformer [27], an architecture in which many "tokens" (i.e., image sections derived from a spatial attention module) are fed into a transformer. The entire block is integrated in a convolutional network. The Visual

Transformer is inspired by *ViT*, the Vision Transformer by Dosovitskiy et al. [28], *ViT* is a stand-alone spatial self-attention architecture in which the transformer's inputs are patches extracted from the tensor image. Previous attempts to implement stand-alone spatial attention were made by Ramachandran et al. [29] and Parmar et al. [30]. Spatial self-attention was also used in GANs by Zhang et al., with their *SAGAN* (Self-Attention Generative Adversarial Network) [31].

More recently, Liu et al. and Dai et al. proposed other two spatial stand-alone self-attention architectures, respectively, the Swin Transformer [32] and the *CoAtNet* [33] (COnvolutional and self-ATention Network). We can think of stand-alone architectures as attempts of rethinking convolution and replacing it in a manner that is able to address its limitations. Many improvements of convolution were proposed, mainly to make them invariant for more general transformations than translations, such as the Deep Symmetry Network proposed by Gens et al. [34], or the Deformable Convolutional Network by Dai et al. [35].

Both *ViT* and *CoAtNet* can be considered the current state-of-the-art on Imagenet, but they outperform Efficient Net by only a very small margin [36], and at the price of a complexity of up to 30×, and of a pre-training on the prohibitive JFT-3B dataset containing 3 billion images.

These are good reasons for considering convolution as not yet fully replaceable by self-attention in computer vision. However, the main reason for why we did not pursue the goal of a stand-alone architecture is that we do not believe in the main assumption that spatial self-attention is based on. Self-attention had a great success in NLP because it eventually exploited the inner relationships between the words in a phrase, which sequential approaches were not able to model effectively. Every word in a sentence has a strong and well defined relationship with any other word in that phrase, and they finally form a complex structure composed of these relationships. However, for instance, if we take a picture of a landscape we see no reason to believe that such a relationship could exist between a rock on the ground and a cloud in the sky or, even more extremely, between two random pixels, at least not in the same way in which the subject of a phrase is related to its verb. On the other hand, this observation does not hold for the features extracted from a picture, and so far, the best way we know of how to extract features from a picture is convolution. These are the main reasons for why we decided to further explore channel self-attention in synergy with convolution, and not as a stand-alone solution.

2.2. FER2013

In the context of a wider research focused on the recognition of human emotions, we tested *LHC-Net* on the FER2013 dataset, a dataset for facial emotion recognition [37]. FER2013 was the subject of a 2013 Kaggle competition (https://www.kaggle.com/datasets/nicolejyt/facialexpressionrecognition, accessed on 5 September 2022). It is a dataset composed of 35,587 grey-scale 48 × 48 images of faces classified in seven categories: anger, disgust, fear, happiness, sadness, surprise, and neutral. The dataset is divided in a training set (28,709 images), a public test set (3589 images), which we used as validation set, and a private test set (3589 images), usually considered as the test set for final evaluations. FER2013 is known as a challenging dataset because of its noisy data, with a relatively large number of non-face images and misclassifications. It is also strongly unbalanced, with only 436 samples in the least populated category, "Disgust", and 7215 samples in the more populated category, "Happiness", as shown in Figure 2.

Table 1 summarizes the results reported by Goodfellow et al. In the paper, linear Support Vector Machine (SVM) is successfully used, reaching 71.16% accuracy, while the human accuracy on FER2013 is limited to 65 ± 5% [37]. The highest reported performance on FER2013 is by Pham et al., who designed the *ResMaskingNet*, which is a *ResNet* backbone augmented with a spatial attention module based on the popular *U-Net*, a segmentation network mostly used in medical image processing. *ResMaskingNet* achieves a remarkable

accuracy of 74.14%. Pham et al. also reported that an ensemble of six convolutional neural networks, including *ResMaskingNet*, reaches 76.82% accuracy [38].

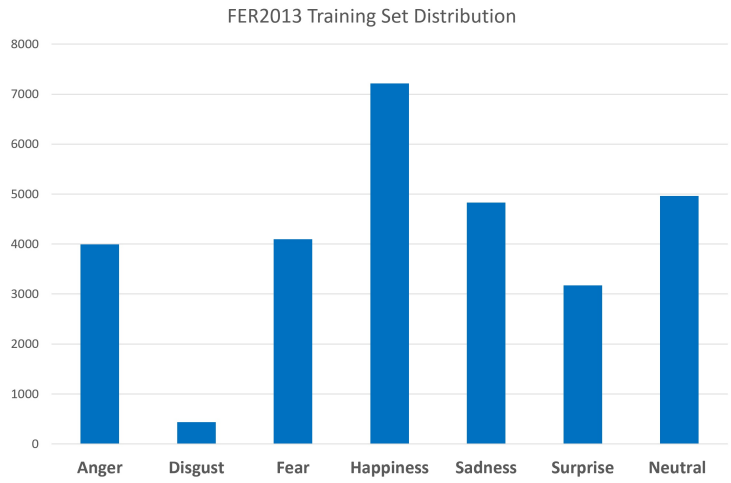

Figure 2. The category distribution on FER2013.

Table 1. Summary of the results reported by Goodfellow et al. [37] on the FER2013 benchmark.

Authors	Method	Accuracy
Minaee et al. [39]	CNN with global spatial attention	70.02%
Pramerdorfer et al. [40]	*Inception*	71.6%
Pramerdorfer et al. [40]	*ResNet*	72.4%
Pramerdorfer et al. [40]	*Visual Geometry Group (VGG)*	72.7%
Khanzada et al. [41]	*SE-ResNet50*	72.7%
Khanzada et al. [41]	*ResNet50*	73.2%
Khaireddin et al. [42]	*VGG* with hyper-parameters fine tuning	73.28%
Pham et al. [38]	*ResMaskingNet* (*ResNet* with spatial attention)	74.14%
Pham et al. [38]	ensemble of 6 convolutional neural networks	76.82%

3. *LHC-Net*

As already mentioned and shown in Figure 1 our *LHC* module can be integrated in virtually any existing convolutional architecture, including, of course, *AlexNet* [43], *VGG* [14], *Inception* [13], and *ResNet* [3].

In this section, we will give a detailed mathematical definition of *LHC* as shown in Figure 3, starting from a generic tensor and forward propagating it through the entire architecture.

3.1. Architecture

We first need to define the model hyper-parameters: Let $n \in \mathbb{N}^+$ be the number of local heads, $s \in \mathbb{N}^+$ the kernel size of the convolution we will use to process the value tensor, $p \in \mathbb{N}^+$ the pool size used in the average pooling and max pooling blocks, $d \in \mathbb{N}^+$ the embedding dimension of every head, and $g \in \mathbb{R}^{\geq 0}$ a constant that we will need in the dynamic scaling module.

Let $\mathbf{x} \in \mathbb{R}^{H,W,C}$ be a generic input tensor, where H, W, and C are respectively, the height, width, and number of channels, with the constraint that $H \times W$ must be divisible by n.

We define **Q**, **K**, and **V** as follows:

$$\mathbf{Q} = \text{AvgPool}_{p,1}(\mathbf{x}) \in \mathbb{R}^{H,W,C} \tag{1}$$

$$\mathbf{K} = \text{MaxPool}_{p,1}(\mathbf{x}) \in \mathbb{R}^{H,W,C} \tag{2}$$

$$\mathbf{V} = \text{AvgPool}_{3,1}(\text{2D-Conv}_{s,1}(\mathbf{x})) \in \mathbb{R}^{H,W,C} \tag{3}$$

where the pooling operators' subscripts are, respectively, the pool size and the stride, and the convolution operator subscripts are, respectively, the kernel size and the stride.

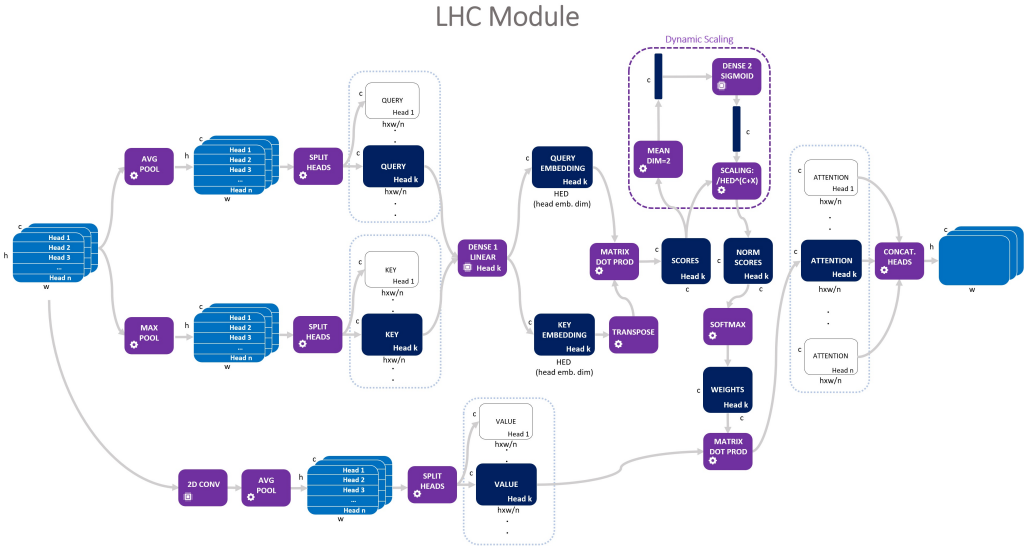

Figure 3. The *LHC* module in its more general multi-head form. Image tensors of shape $H \times W \times C$ are in pale blue; when reshaped/processed, they are in dark blue. The processing units are in violet.

Now, we want to split the tensors **Q**, **K**, and **V** into n horizontal slices and reshape the resulting tensors as follows: $\forall h = 1, \ldots, n$

$$\mathbf{q}_h = [\text{SplitHeads}(\mathbf{Q})]_h \in \mathbb{R}^{C,(H\times W)/n} \tag{4}$$

$$\mathbf{k}_h = [\text{SplitHeads}(\mathbf{K})]_h \in \mathbb{R}^{C,(H\times W)/n} \tag{5}$$

$$\mathbf{v}_h = [\text{SplitHeads}(\mathbf{V})]_h \in \mathbb{R}^{C,(H\times W)/n} \tag{6}$$

Every head is deputed to process a triplet $(\mathbf{q}_h, \mathbf{k}_h, \mathbf{v}_h)$, then we have n separate fully connected layers with linear output and weights/biases: $\mathbf{w}_{1,h} \in \mathbb{R}^{(H\times W)/n,d}$, $\mathbf{b}_{1,h} \in \mathbb{R}^d$.

Queries and keys will share the same dense blocks, resulting in n embeddings, as follows:

$$\tilde{q}_h^{i,j} = \sum_{t=1}^{(H\times W)/n} q_h^{i,t} w_{1,h}^{t,j} + b_{1,h}^j \in \mathbb{R} \tag{7}$$

$$\tilde{k}_h^{i,j} = \sum_{t=1}^{(H\times W)/n} k_h^{i,t} w_{1,h}^{t,j} + b_{1,h}^j \in \mathbb{R} \tag{8}$$

$$\forall h = 1, \ldots, n$$
$$\forall i = 1, \ldots, C$$
$$\forall j = 1, \ldots, d$$

Or, more shortly (from now on, we will omit the head logic quantifier):

$$\tilde{\mathbf{q}}_h = \mathbf{q}_h \cdot \mathbf{w}_{1,h} + \mathbf{b}_{1,h} \in \mathbb{R}^{C,d} \tag{9}$$

$$\tilde{\mathbf{k}}_h = \mathbf{k}_h \cdot \mathbf{w}_{1,h} + \mathbf{b}_{1,h} \in \mathbb{R}^{C,d} \tag{10}$$

Now, we can compute the attention scores through transposition and the matrix product:

$$\mathbf{S}_h = \tilde{\mathbf{q}}_h \cdot \tilde{\mathbf{k}}_h^T \in \mathbb{R}^{C,C} \tag{11}$$

Dynamic scaling produces a channel-wise learned scaling (not dependent from heads) through averaging of the scores and passing them through another fully connected layer with sigmoid activation and weights/biases $\mathbf{w}_2 \in \mathbb{R}^{C,C}$, $\mathbf{b}_2 \in \mathbb{R}^C$:

$$\tilde{\mathbf{S}}_h = \text{Mean}_{dim=2}(\mathbf{S}_h) \in \mathbb{R}^C \tag{12}$$

$$T_h^i = \text{Sig}\left(\sum_{t=1}^{C} \tilde{S}_h^t w_2^{t,i} + b_2^i\right) \in \mathbb{R} \quad \forall i = 1, ..., C \tag{13}$$

$$N_h^{i,j} = \frac{S_h^{i,j}}{d^{(g+T_h^i)}} \in \mathbb{R} \qquad \forall i, j = 1, ..., C \tag{14}$$

$$\mathbf{W}_h = \text{Softmax}_{dim=2}(\mathbf{N}_h) \in \mathbb{R}^{C,C} \tag{15}$$

where $\mathbf{T}_h$ is the tensor of the scaling factors, $\mathbf{N}_h$ the tensor of the normalized attention scores, and $\mathbf{W}_h$ are the final attention weights associated with the head h.

Now, we can compute the final attention tensor for head h very straightforwardly:

$$\mathbf{A}_h = \mathbf{W}_h \cdot \mathbf{v}_h \in \mathbb{R}^{C,(H \times W)/n} \tag{16}$$

and using simple transpose, reshape, and concatenation operators, we can compose the output $\mathbf{y}$ by assembling the n heads:

$$\mathbf{y} = \text{SplitHeads}^{-1}([\mathbf{A}_1, \mathbf{A}_2, ..., \mathbf{A}_n]) \in \mathbb{R}^{H,W,C} \tag{17}$$

3.2. Motivation and Analysis

3.2.1. Channel Self-Attention

We already explained the main reasons behind our choice of channel-wise self-attention. We can summarize them as follows:

- Spatial attention in computer vision strongly relies on the main assumption that a relationship between the single pixels or areas of an image exists. This assumption is not self-evident, or at least, not as evident as the relationship between words in a phrase, which spatial attention is inspired by.
- All attempts to pursue spatial self-attention in computer vision (especially in stand-alone mode) have gained only minor improvements over the previous state-of-the-art architectures, and most of the times, at the price of an unreasonably higher computational cost and a prohibitive pre-training on enormous datasets.
- Much more simple and computationally cheaper approaches, like Squeeze and Excitation in *Efficient Net*, have already been proven to be very effective without the need to replace convolution.

3.2.2. Dynamic Scaling

In Vaswani's *Transformer*, the scaling is static and constant among the sequence. Equation (14) becomes:

$$\mathbf{N} = \frac{\mathbf{S}}{\sqrt{d}}$$

The idea behind our dynamic scaling is to exploit the following behavior of the Softmax function. Given a non-constant vector $\mathbf{x} \in \mathbb{R}^n$ and a positive constant $\alpha > 0$, it results:

$$\lim_{\alpha \to +\infty} \frac{e^{\alpha x_i}}{\sum_{j=1}^{n} e^{\alpha x_j}} = \begin{cases} 1, & \text{if } x_i = \max(\mathbf{x}) \\ 0, & \text{otherwise} \end{cases} \tag{18}$$

$$\lim_{\alpha \to 0^+} \frac{e^{\alpha x_i}}{\sum_{j=1}^{n} e^{\alpha x_j}} = \frac{1}{\sum_{j=1}^{n} 1} = \frac{1}{n} \tag{19}$$

$$\frac{e^{x_{i_1}}}{\sum_{j=1}^{n} e^{x_j}} < \frac{e^{x_{i_2}}}{\sum_{j=1}^{n} e^{x_j}} \Leftrightarrow \frac{e^{\alpha x_{i_1}}}{\sum_{j=1}^{n} e^{\alpha x_j}} < \frac{e^{\alpha x_{i_2}}}{\sum_{j=1}^{n} e^{\alpha x_j}} \tag{20}$$

These equations imply that we can multiply a logit vector $\mathbf{x}$ by a positive constant α without altering its softmax ranking (Equation (20)), and if α is small, the resulting softmax vector approximates an arithmetic average (Equation (19)); if it is large, it will be close to a one-hot vector valued 1 on the max of $\mathbf{x}$, and 0, otherwise (Equation (18)). In other words, the dynamic scaling module learns how complex the new feature maps must be. If the α associated with a given new feature map is large, this feature map will be a strict selection of the old feature maps; if it is small, the new feature map will be a more complex composition of old feature maps involving a greater number of them.

3.2.3. Shared Linear Embedding and Convolution

A shared linear layer was already explored by Woo et al., with their *CBAM* vanilla attention architecture [19]. Our idea is exploiting the "self" nature of our attention mechanism. Using Vaswani's terminology, self-attention means that the query, key, and value originate from the same tensor. We decided to leverage this aspect and save some complexity by first differentiating query and key, respectively, with average and max pooling in order to enhance different scale aspects of the input, and then feed them into a single shared linear embedding layer. Dense mapping is also helped by the big dimensionality reduction due to head splitting.

On the other hand, we used global convolution for the entire value tensor in order to preserve the bi-dimensional structure of the data.

3.2.4. Local Multi-Head

In the original *Transformer*, the multi-head structure is a concatenation of blocks all processing the same input. Voita et al. [44] analyzed the *Transformer* and found a surprising redundancy in the representation offered by different heads: pruning 44 out of 48 heads from the *Transformer*'s encoder block results only in a residual performance drop. Only four heads (8%) were necessary to maintain a performance level very close to the one of the entire architecture. We tried to perform a similar evaluation for the *LHC-Net* by simply "switching off" *LHC* blocks and by "de-training" them (i.e., set the weights and biases of *LHC* blocks at the initialization status, before training). In our case, it was feasible to just switch off or de-train the new blocks without any further training, because the entire *ResNet* backbone of the network was pre-trained and already able to deliver a very high level of

performance. With this approach we found that at least 16 heads out of 31 (52%) were necessary; more precisely, the first two*LHC* blocks.

We further analyzed this behavior, and in order to make another comparison, we trained a standard *Transformer* encoder block as a simple classifier for an NLP classification problem reaching a very high accuracy, then we evaluated the model by simple correlation between the output of the heads and found a correlation between heads of up to 93%. As a comparison, our architecture had a correlation between heads of 63%.

There are many attempts to improve the attention mechanism of the *Transformer*. Cordonnier et al. tried to address the redundancy issue with a modified multi-head mechanism [45]. Sukhbaatar et al. proposed a modified *Transformer* with an adaptive attention span [46].

More similarly to our local approach, India et al. proposed a multi-head attention mechanism for speaker recognition, in which every head processes a different section of the input [47]. There are two main differences with our approach (other than the application field and implementation details):

- Their approach is not designed for self-attention.
- Their local processing units are used at a later stage. They directly calculate local attention weights from embeddings (scalar output with softmax activation). Our local processing units calculate the initial embeddings (high dimension output with linear activation).

The ideas behind local heads are mainly three:

- Local heads have the advantage of working at a much lower dimension. Detecting a pattern of a few pixels is harder if the input includes the entire feature map.
- Splitting the images in smaller parts gives to local heads the ability to build new feature maps, considering only the important parts of the old maps. There is no reason to compose feature maps in their entirety when only a small part is detecting an interesting feature. Local heads are able to add a feature map to a new feature map only if the original map is activated by a pattern and only around that pattern, avoiding the addition of not useful information.
- Local heads seem to be more efficient in terms of parameters allocation (see Appendix B).

4. Experiments

We mainly focused on using *LHC* in conjunction with a pre-trained backbone, the *ResNet34v2*. The training process consisted of training a *ResNet34v2* (with Imagenet pre-training initialization) on FER2013, then adding 5 *LHC* modules, as shown in Figure 1 and then further training the entire architecture. The idea was to design modules with a small impact on the "host" network, similar to the approach of the Squeeze and Excitation modules [17]. In other words, our main goal was to test the ability of *LHC* to give an extra performance boost to an existing good performing model. Secondarily, we also tested *LHC-Net* as a stand-alone model trained (only the Imagenet pre-training of the *ResNet* part) from scratch, and we obtained limited but very good results. In this section, we will discuss the experimental results.

4.1. Experimental Setup

We rescaled the FER2013 images to 224×224 and converted them to RGB, in order to match the resolution of Imagenet and to make them compatible with the pre-trained *ResNet*. For rescaling, we used bilinear interpolation. Then, in order to save RAM, we stored the entire training set as jpeg images, accepting some neglectable quality loss, and we used the TensorFlow Image Data Generator to feed the model during training. Saving images in jpeg format implies two different quality losses: the jpeg compression itself, and the need to approximate the tensor images to be uint8 (bilinear interpolation in rescaling generates non-integer values). To achieve that, the tensors could be rounded or truncated. Considering that truncation is only a rounding with the input is shifted by 0.5, and that this

shifting makes the training set in FER2013 better match the validation and test set average pixel value, we proceeded with raw truncation.

The implementation details of *ResNet* are reported in Figure 1, and the model parameters of the 5 *LHC* blocks are in Table 2.

We trained the *ResNet* backbone in a sequential mode with 3 training stages, using standard crossentropy loss, varying the data augmentation protocol, the batch size, and the optimizer at every stage. Early stopping was performed on the validation set. The training hyper-parameters are detailed in Tables 3–5.

Table 2. Model hyper-parameters.

Block	Heads	Dim	Pool	Scale	Ker
LHC1	8	196	3	1	3
LHC2	8	196	3	1	3
LHC3	7	56	3	1	3
LHC4	7	14	3	1	3
LHC5	1	25	3	1	3

Table 3. Training hyper-parameters in Stage 1.

Optimizer	Adam, learning rate = 0.0001
Batch Size	48
Patience	30 epochs
Augmentation	30 degree rotation

Table 4. Training hyper-parameters in Stage 2.

Optimizer	Stochastic Gradient Descent, learning rate = 0.01
Batch Size	64
Patience	10 epochs
Augmentation	10 degree rotation 0.1 horizontal/vertical shift 0.1 zoom

Table 5. Training hyper-parameters in Stage 3.

Optimizer	Stochastic Gradient Descent, learning rate = 0.01
Batch Size	64
Patience	5 epochs
Augmentation	-

At this point, we have our *ResNet* ready to be augmented with *LHC* and further trained. We used the very simple training protocol in Table 6.

Table 6. Training hyper-parameters in Stage 4.

Optimizer	Stochastic Gradient Descent, learning rate = 0.01
Batch Size	64
Patience	3 epochs
Augmentation	-

We observed in some cases, depending on the *LHC* initialization, that the added modules are somehow "rejected" by the host network and the training struggles to converge; in one case, it totally diverged. This happened in a minority of the total attempts, but to

perform the following evaluations, we kept only the models whose training loss was less than the starting *ResNet* training loss, plus an extra 10% to take into account the augmented complexity of the model.

To evaluate *LHC*, we first applied stage 4 to the single best *ResNet34* model that we managed to achieve (with stages 1, 2 and 3), varying the data generator seed without *LHC* modules (set A). Then, starting from the same base network, we augmented it with *LHC* modules and trained it using the same protocol. We tried a small number of trainings with a variety of model parameters (keeping the data generator seed fixed) and clearly detected a neighborhood of settings appearing to work well (set B). At this point, we trained several other models with the best promising parameters setting, varying the generator seed (set C). We then compared set A with set $B \cup C$.

We also considered a minor variation of *LHC-Net*. We tried to exploit the analysis on the 5 modules we discussed in the previous section, showing the last modules playing a minor role, and trained 5 weights, limited by hyperbolic tangent, for every residual sum shown in Figure 1. We manually initialized this 5 weights by setting them as follows: $a_1 = \text{tansig}(0)$, $a_2 = \text{tansig}(0)$, $a_3 = \text{tansig}(0)$, $a_4 = \text{tansig}(-1)$, $a_5 = \text{tansig}(-0.5)$, with the idea of limiting the impact of the last 2 modules. We call it *LHC-NetC*. Accordingly, with the original Kaggle rules and with almost all evaluation protocols in literature, only the private test set was used for the final evaluations (the public test set performance also appeared to be not well correlated, with neither training nor private test performances). For comparison with *ResNet*, we did not use test time augmentation (TTA). We used TTA only for the final evaluation and comparison with other models in the literature. Our TTA protocol is totally deterministic; we first used a sequence of transformations involving horizontal flipping, ± 10 pixels horizontal/vertical shifts, and finally ± 0.4 radians rotations, in this order. We use rotation after shifting to combine their effect. Rotating first puts the images in only 9 spots, which become 25 if we shift first. At this point, we used a second batch of transformations involving horizontal flipping, 10% zoom, and again, ± 0.4 radian rotations. Finally, we weighted the no-transformation inference at 3 times the weight of the other inferences.

4.2. Results

LHC-Net was able to consistently outperform our best performing *ResNet34v2*, both on average and on the peak result (see Table 7). Note that the average is not dramatically affected by the peak result. Removing the peak results does not alter the average qualitative evaluation. See the Appendix C for a detailed benchmark with several other architectures, with and without TTA.

There are some key points emerging from this analysis:

- *ResNet34* is confirmed to be the most effective architecture on FER2013, especially its v2 version. In our experiments, raw *ResNet34* trained with the multi-stage protocol and inferenced with TTA reaches an accuracy that is not distant from the previous SOTA (*ResMaskingNet*).
- Heavy architectures seem not to be able to outperform more simple models on FER2013.
- *LHC-Net* has the top accuracy, both with and without TTA.
- *LHC-NetC* outperforms *LHC-Net*, but is outperformed when TTA is used.
- More importantly, *LHC-Net* outperforms the previous SOTA with less than one-fourth of its free parameters, and the impact of the LHC modules on the base architecture is much lower (less than 15% vs. over 80%), and it is closer to other attention modules such as *CBAM*/*BAM*/*SE*.

As mentioned, we experimented limitedly with stand-alone training as well, with very good results. We trained in parallel, using the same data generator seeds and the same multi-stage protocol, both *LHC-Net* and *ResNet34v2*. In both models, *ResNet34v2* was initialized with Imagenet pre-trained weights. It resulted in *LHC-Net* consistently

outperforming *ResNet34v2* at the end of every training stage. This is a limited but very encouraging result.

Table 7. Test accuracy statistics without TTA.

Model	Top 40%	Top 40% w/o Best	Top 25%	Top 25% w/o Best	Best
ResNet34v2	72.69%	72.65%	72.75%	72.69%	72.81%
LHC-Net	**72.89**%	72.77%	**73.02**%	72.83%	**73.39**%
LHC-NetC	**73.04**%	72.79%	**73.21**%	72.89%	**73.53**%

5. Conclusions and Future Developments

Attention is a powerful idea, and despite that its impact on computer vision might be not as revolutionary as on NLP, it has still been proven to be an important and sometimes decisive tool.

We designed a novel local multi-head channel self-attention module, the *LHC*, and it contributed in proving that channel self-attention, in synergy with convolution, could be a functioning paradigm, by setting a new best single-model performance on the well known FER2013 dataset. We also proved that self-attention works well as a small attention module intended as a booster for pre-existing architectures, like other famous vanilla attention modules as *CBAM* or Squeeze and Excitation.

The future research on this architectures will include several aspects:

- Testing *LHC* on other, more computational intensive scenarios such as the Imagenet dataset.
- Testing *LHC* with other backbone architectures and with a larger range of starting performances (not only peak performances).
- We did not optimize the general topology of *LHC-Net*, and the model hyper-parameters of the attention blocks are hand-selected with only a few attempts. There is evidence that both the 5 blocks topology and hyper-parameters might be sub-optimal.
- Further research on the stand-alone training mode will be necessary.
- Normalization blocks before and after the *LHC* blocks should be better evaluated, in order to mitigate the divergence issue mentioned in the previous section.
- A second convolution before the residual connection should be considered, to mimic the general structure of the original *Transformer*.
- A better head splitting technique could be key in future research. The horizontal splitting we used was only the most obvious way to achieve it, but not necessarily the most effective. Other approaches should be evaluated, e.g., learning the optimal areas through spatial attention.

The main results of this paper are replicable by cloning the repository and by following the instructions available at: https://github.com/Bodhis4ttva/LHC_Net (last accessed on 5 September 2022).

Author Contributions: Conceptualization, methodology, software, R.P.; writing—original draft preparation, R.P.; writing—review and editing, V.B. (Valerio Basile) and V.B. (Viviana Bono); supervision, V.B. (Valerio Basile) and V.B. (Viviana Bono). All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The FER2013 dataset is publicly available at https://paperswithcode.com/dataset/fer2013 (last accessed on 5 September 2022).

Acknowledgments: We would like to express our deepest appreciation to Carmen Frasca for her crucial support to our research. We would also like to extend our sincere thanks to Luan Pham and Valerio Coderoni for their helpfulness and kindness.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Acronyms

Table A1 summarizes the acronyms used throughout the paper and their respective meanings.

Table A1. Meaning of the acronyms used in the paper.

Acronym	Meaning
AI	Artificial Intelligence
CBAM	Convolutional Block Attention Module
CNN	Convolutional Neural Network
CoAtNet	COnvolutional and self-ATention Network
ECA	Efficient Channel Attention Network
FER	Facial Expression Recognition
GAN	Generative Adversarial Network
LHC	Local multi-Head Channel self-attention
LSTM	Long Short-Term Memory
ML	Machine Learning
NLP	Natural Language Processing
ResNet	Residual Network
RM-Net	Residual Masking Network
SAGAN	Self-Attention Generative Adversarial Network
SCA	Spatial and Channel wise Attention
SE	Squeeze and Excitation Network
SGD	Stochastic Gradient Descent
SOTA	State of the art
SVM	Support Vector Machine
TTA	Test Time Augmentation
URCA	UpSample Residual Channel-wise Attention
VGG	Visual Geometry Group
VIT	Vision Transformer

Appendix B. A Qualitative Analysis of Multi-Head Efficiency over Global Head

We experimentally found the performance positively correlated with the number of heads, but we also tried to give a qualitative explanation of the third observation we mentioned in the motivation of the multi-head approach by designing a concrete example. Let's say that we have n feature maps as the output of the previous convolution block, and that the optimal composition of those maps includes a combination of two of them, the ith and the jth, in the kth target feature map. In order to learn this pattern, using Equations (9) and (10) (omitting the biases), a single global head must map:

$$\tilde{\mathbf{q}} = \mathbf{q} \cdot \mathbf{w}_1 \in \mathbb{R}^{C,d}$$
$$\tilde{\mathbf{k}} = \mathbf{k} \cdot \mathbf{w}_1 \in \mathbb{R}^{C,d}$$

in such a way that $\tilde{\mathbf{q}}_k$ and $\tilde{\mathbf{k}}_i$ must be collinear in order to produce a high attention score in the kth target feature for the ith old feature map by dot product. The same goes for $\tilde{\mathbf{q}}_k$ and $\tilde{\mathbf{k}}_j$. To summarize, we have three vectors that need to be mapped in other three vectors linked by two constraint rules. In total, we have $3(HW + d)$ dimensions or $3(HWd)$ relationships subject to two constraints to be modeled. To achieve this with the embedding

linear layer, we have a matrix $\mathbf{w} \in \mathbb{R}^{H \times W, d}$, equivalent to HWd free parameters. So, we have:

$$G1 = \frac{HWd}{3(HW + d)2} \tag{A1}$$

$$G2 = \frac{HWd}{3(HWd)2} = \frac{1}{6} \tag{A2}$$

where $G1$ is the number of free parameters for the dimension for every constraint, and $G2$ is the number of free parameters for relationship for every constraint. We see them as the qualitative measures of the efficiency of the global single head. Now, we want to calculate them in the case of n local heads. The difference is that the local heads work only on fractions of the entire input tensor, so we have to take into account where the ith and the jth filters are eventually activated. For a given section of the input tensor, there are three cases: only one of them could be activated in that area, both of them or none of them. We call A the number of sections with one possible activation, B the number of sections with two possible activations, and C the number of sections with no possible activations. It results in:

$$A + B + C = n$$

but this time, $\mathbf{w}_{1,h} \in \mathbb{R}^{(H \times W)/n, d}$; hence, we have:

$$L1 = \left(A \frac{\frac{HW}{n} d}{2(\frac{HW}{n} + d)} + B \frac{\frac{HW}{n} d}{3(\frac{HW}{n} + d)2} \right) / (A + B) \tag{A3}$$

$$\begin{aligned} L2 &= \left(A \frac{\frac{HW}{n} d}{2(\frac{HW}{n} d)} + B \frac{\frac{HW}{n} d}{3(\frac{HW}{n} d)2} \right) / (A + B) \\ &= \left(\frac{A}{2} + \frac{B}{6} \right) / (A + B) \end{aligned} \tag{A4}$$

We have immediately:

$$\begin{aligned} &L2 > G2 \Leftrightarrow \\ &\left(\frac{A}{2} + \frac{B}{6} \right) / (A + B) > \frac{1}{6} \Leftrightarrow A > 0 \end{aligned}$$

Or more shortly:

$$L2 \geq G2 \tag{A5}$$

$$L2 = G2 \Leftrightarrow A = 0 \tag{A6}$$

if the ith and the jth filters are possibly activated in every section of the input tensor, local multi-head is equivalent to global single head in terms of efficiency and effectiveness, but a single section of the input tensor with only one possible activation is enough to make local multi-head more effective.

If we decide to consider the dimensions ($L1$ and $G1$ measures instead of $L2$ and $G2$), the calculation is more complicated; to make it easier, we shall make some basic assumptions.

Let's consider the hyper-parameters settings of the actual first two blocks of our *LHC-Net*, where $d = \frac{HW}{2n}$ and $n = 8$. We have:

$$
\begin{aligned}
& L1 > G1 \Leftrightarrow \\
& \frac{\left(A\frac{\frac{HW}{n}d}{2(\frac{HW}{n}+d)} + B\frac{\frac{HW}{n}d}{2(\frac{HW}{n}+d)3}\right)}{(A+B)} > \frac{HWd}{3(HW+d)2} \Leftrightarrow \\
& \left(A\frac{\frac{1}{n}}{2\frac{HW}{n}(1+\frac{1}{2})} + B\frac{\frac{1}{n}}{6\frac{HW}{n}(1+\frac{1}{2})}\right) > \frac{A+B}{6HW(1+\frac{1}{2n})} \Leftrightarrow \\
& \left(A\frac{1}{2(1+\frac{1}{2})} + B\frac{1}{6(1+\frac{1}{2})}\right) > \frac{A+B}{6(1+\frac{1}{16})} \Leftrightarrow \\
& \left(\frac{A}{3} + \frac{B}{9}\right) > \frac{16A+16B}{102} \Leftrightarrow \\
& (3A+B) > \frac{144A+144B}{102} \Leftrightarrow A > 0.26B
\end{aligned}
$$

In this case, the combinations $A = 0$ and $B = 8$, and $A = 1$ and $B = 7$ give an advantage to global single head. Every other possible combination performs the opposite, as shown in this Figure A1.

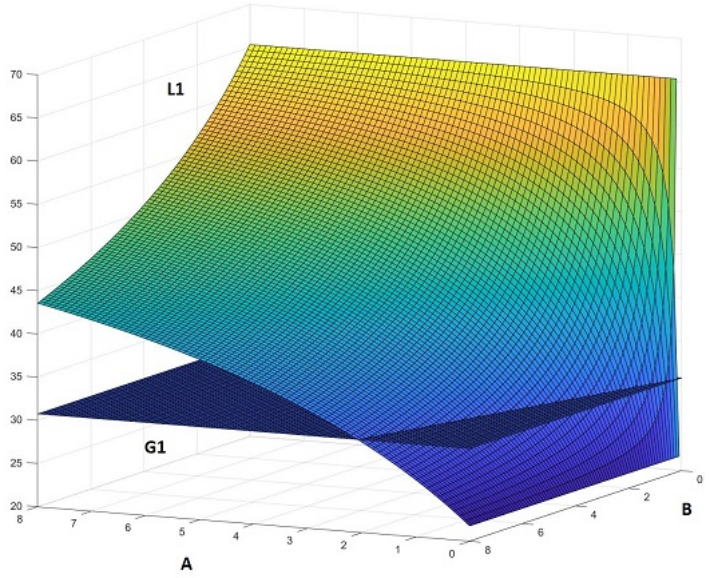

Figure A1. $L1$ and $G1$ as functions of A and B, as described.

This, of course, has not the ambition to be a rigorous proof of the goodness of local heads over global head; it is only a qualitative analysis giving an encouraging view. It shows that local heads might have a mathematical advantage over global heads in those scenarios, like, for example, FER, where images have all a shared general structure.

Appendix C. Benchmark Results

In Table A2, we report the results of our model in comparison with other models from the recent literature on the same benchmark. The models marked by * are reported in the GitHub repository associated with the referenced paper, and not directly into the paper.

Table A2. Performance comparison between *LHC-Net* and several other architectures as reported by the respective authors, with and without TTA. In the last column is reported the weight of the attention modules in the resulting architecture: 5% means that the attention modules are responsible for 5% of the total number of free parameters, while the reamining 95% is due to the backbone architecture. Models marked by * are reported in the GitHub repository and not in the paper.

Model	Accuracy	TTA	Params	Att
BoW Repr. [37]	67.48%	no	-	-
Human [37]	70.00%	no	-	-
CNN [39]	70.02%	no	-	-
VGG19 [38]	70.80%	yes	143.7M	-
EffNet [38] *	70.80%	yes	9.18M	-
SVM [37]	71.16%	no	-	-
Inception [40]	71.60%	yes	23.85M	-
Incep.v1 [38] *	71.97%	yes	5M	-
ResNet34 [40]	72.40%	yes	27.6M	-
ResNet34 [38]	72.42%	yes	27.6M	-
VGG [40]	72.70%	yes	143.7M	-
SE-Net50 [41]	72.70%	yes	27M	5.18%
Incep.v3 [38] *	72.72%	yes	23.85M	-
ResNet34v2	72.81%	no	27.6M	-
BAMRN50 [38] *	73.14%	yes	24.07M	1.62%
Dense121 [38]	73.16%	yes	8.06M	-
ResNet50 [41]	73.20%	yes	25.6M	-
ResNet152 [38]	73.22%	yes	60.38M	-
VGG [42]	73.28%	yes	143.7M	-
CBAMRN50 [38]	73.39%	yes	28.09M	9%
LHC-Net	**73.39**%	no	32.4M	14.8%
LHC-NetC	**73.53**%	no	32.4M	14.8%
ResNet34v2	73.92%	yes	27.6M	-
RM-Net [38]	74.14%	yes	142.9M	80.7%
LHC-NetC	**74.28**%	yes	32.4M	14.8%
LHC-Net	**74.42**%	yes	32.4M	14.8%

References

1. Fasel, B.; Luettin, J. Automatic facial expression analysis: A survey. *Pattern Recognit.* **2003**, *36*, 259–275. https://doi.org/10.1016/S0031-3203(02)00052-3.
2. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 5998–6008.
3. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
4. Shi, Y.; Jain, A.K. Probabilistic Face Embeddings. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea, 27 October–2 November 2019.
5. Huang, Y.; Qiu, S.; Zhang, W.; Luo, X.; Wang, J. More Information Supervised Probabilistic Deep Face Embedding Learning. In Proceedings of the 37th International Conference on Machine Learning Research, Virtual, 13–18 July 2020; Daumé, H., III, Singh, A., Eds.; 2020; Volume 119; pp. 4484–4494.
6. Wang, X.; Wang, S.; Chi, C.; Zhang, S.; Mei, T. Loss Function Search for Face Recognition. In Proceedings of the 37th International Conference on Machine Learning Research, Virtual, 13–18 July 2020; Daumé III, H., Singh, A., Eds.; 2020; Volume 119, pp. 10029–10038.
7. Yang, X.; Jia, X.; Gong, D.; Yan, D.M.; Li, Z.; Liu, W. LARNet: Lie Algebra Residual Network for Face Recognition. In Proceedings of the 38th International Conference on Machine Learning Research, Virtual, 18–24 July 2021; Meila, M., Zhang, T., Eds.; PMLR 139 2021; Volume 139, pp. 11738–11750.
8. Uppal, H.; Sepas-Moghaddam, A.; Greenspan, M.; Etemad, A. Depth as Attention for Face Representation Learning. *IEEE Trans. Inf. Forensics Secur.* **2021**, *16*, 2461–2476. https://doi.org/10.1109/TIFS.2021.3053458.
9. Zhong, S.h.; Liu, Y.; Zhang, Y.; Chung, F.l. Attention modeling for face recognition via deep learning. In Proceedings of the Annual Meeting of the Cognitive Science Society, Sapporo, Japan, 1–4 August 2012; Volume 34.

10. Bahdanau, D.; Cho, K.; Bengio, Y. Neural machine translation by jointly learning to align and translate. *arXiv* **2014**, arXiv:1409.0473.
11. Luong, M.T.; Pham, H.; Manning, C.D. Effective approaches to attention-based neural machine translation. *arXiv* **2015**, arXiv:1508.04025.
12. Rumelhart, D.E.; Hinton, G.E.; Williams, R.J. Learning representations by back-propagating errors. *Nature* **1986**, *323*, 533–536.
13. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
14. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
15. Xu, L.; Huang, J.; Nitanda, A.; Asaoka, R.; Yamanishi, K. A Novel Global Spatial Attention Mechanism in Convolutional Neural Network for Medical Image Classification. *arXiv* **2020**, arXiv:2007.15897.
16. Jaderberg, M.; Simonyan, K.; Zisserman, A.; Kavukcuoglu, K. Spatial transformer networks. *Adv. Neural Inf. Process. Syst.* **2015**, *28*, 2017–2025.
17. Hu, J.; Shen, L.; Sun, G. Squeeze-and-excitation networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7132–7141.
18. Tan, M.; Le, Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In Proceedings of the International Conference on Machine Learning, PMLR, Long Beach, CA, USA, 9–15 June 2019; pp. 6105–6114.
19. Woo, S.; Park, J.; Lee, J.Y.; Kweon, I.S. Cbam: Convolutional block attention module. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 3–19.
20. Wang, Q.; Wu, B.; Zhu, P.; Li, P.; Zuo, W.; Hu, Q. ECA-Net: Efficient Channel Attention for Deep Convolutional Neural Networks. *arXiv* **2020**, arXiv:1910.03151.
21. Chen, L.; Zhang, H.; Xiao, J.; Nie, L.; Shao, J.; Liu, W.; Chua, T.S. Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 5659–5667.
22. Nie, X.; Ding, H.; Qi, M.; Wang, Y.; Wong, E.K. URCA-GAN: UpSample Residual Channel-wise Attention Generative Adversarial Network for image-to-image translation. *Neurocomputing* **2021**, *443*, 75–84.
23. Fu, J.; Liu, J.; Tian, H.; Li, Y.; Bao, Y.; Fang, Z.; Lu, H. Dual attention network for scene segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 3146–3154.
24. Liu, X.; Xiao, G.; Dai, L.; Zeng, K.; Yang, C.; Chen, R. SCSA-Net: Presentation of two-view reliable correspondence learning via spatial-channel self-attention. *Neurocomputing* **2021**, *431*, 137–147.
25. Tian, Y.; Zhang, Y.; Zhou, D.; Cheng, G.; Chen, W.G.; Wang, R. Triple attention network for video segmentation. *Neurocomputing* **2020**, *417*, 202–211.
26. Bello, I.; Zoph, B.; Vaswani, A.; Shlens, J.; Le, Q.V. Attention augmented convolutional networks. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019; pp. 3286–3295.
27. Wu, B.; Xu, C.; Dai, X.; Wan, A.; Zhang, P.; Yan, Z.; Tomizuka, M.; Gonzalez, J.; Keutzer, K.; Vajda, P. Visual transformers: Token-based image representation and processing for computer vision. *arXiv* **2020**, arXiv:2006.03677.
28. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An image is worth 16 $\times$ 16 words: Transformers for image recognition at scale. *arXiv* **2020**, arXiv:2010.11929.
29. Ramachandran, P.; Parmar, N.; Vaswani, A.; Bello, I.; Levskaya, A.; Shlens, J. Stand-alone self-attention in vision models. *arXiv* **2019**, arXiv:1906.05909.
30. Parmar, N.; Vaswani, A.; Uszkoreit, J.; Kaiser, L.; Shazeer, N.; Ku, A.; Tran, D. Image transformer. In Proceedings of the International Conference on Machine Learning, PMLR, Stockholm, Sweden, 10–15 July 2018; pp. 4055–4064.
31. Zhang, H.; Goodfellow, I.; Metaxas, D.; Odena, A. Self-attention generative adversarial networks. In Proceedings of the International Conference on Machine Learning, PMLR, Long Beach, CA, USA, 9–15 June 2019; pp. 7354–7363.
32. Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. *arXiv* **2021**, arXiv:2103.14030.
33. Dai, Z.; Liu, H.; Le, Q.V.; Tan, M. CoAtNet: Marrying Convolution and Attention for All Data Sizes. *arXiv* **2021**, arXiv:2106.04803.
34. Gens, R.; Domingos, P.M. Deep symmetry networks. *Adv. Neural Inf. Process. Syst.* **2014**, *27*, 2537–2545.
35. Dai, J.; Qi, H.; Xiong, Y.; Li, Y.; Zhang, G.; Hu, H.; Wei, Y. Deformable convolutional networks. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 764–773.
36. Pham, H.; Dai, Z.; Xie, Q.; Le, Q.V. Meta pseudo labels. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 11557–11568.
37. Goodfellow, I.J.; Erhan, D.; Carrier, P.L.; Courville, A.; Mirza, M.; Hamner, B.; Cukierski, W.; Tang, Y.; Thaler, D.; Lee, D.H.; et al. Challenges in representation learning: A report on three machine learning contests. In Proceedings of the International Conference on Neural Information Processing, Daegu, Korea, 3–7 November 2013; Springer: Berlin/Heidelberg, Germany, 2013; pp. 117–124.
38. Pham, L.; Vu, T.H.; Tran, T.A. Facial Expression Recognition Using Residual Masking Network. In Proceedings of the 2020 25th International Conference on Pattern Recognition (ICPR), Milan, Italy, 10–15 January 2021; pp. 4513–4519.

39. Minaee, S.; Minaei, M.; Abdolrashidi, A. Deep-emotion: Facial expression recognition using attentional convolutional network. *Sensors* **2021**, *21*, 3046.
40. Pramerdorfer, C.; Kampel, M. Facial expression recognition using convolutional neural networks: state of the art. *arXiv* **2016**, arXiv:1612.02903.
41. Khanzada, A.; Bai, C.; Celepcikay, F.T. Facial expression recognition with deep learning. *arXiv* **2020**, arXiv:2004.11823.
42. Khaireddin, Y.; Chen, Z. Facial Emotion Recognition: State of the Art Performance on FER2013. *arXiv* **2021**, arXiv:2105.03588.
43. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **2012**, *25*, 1097–1105.
44. Voita, E.; Talbot, D.; Moiseev, F.; Sennrich, R.; Titov, I. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. *arXiv* **2019**, arXiv:1905.09418.
45. Cordonnier, J.B.; Loukas, A.; Jaggi, M. Multi-head attention: Collaborate instead of concatenate. *arXiv* **2020**, arXiv:2006.16362.
46. Sukhbaatar, S.; Grave, E.; Bojanowski, P.; Joulin, A. Adaptive attention span in transformers. *arXiv* **2019**, arXiv:1905.07799.
47. India, M.; Safari, P.; Hernando, J. Self multi-head attention for speaker recognition. *arXiv* **2019**, arXiv:1906.09890.