Foveated Neural Computation

Matteo Tiezzi, Simone Marullo, Alessandro Betti, Enrico Meloni, Lapo Faggi, Marco Gori, Stefano Melacci

Foveated Neural Computation

Matteo Tiezzi[1] ✉, Simone Marullo[1,2], Alessandro Betti[3], Enrico Meloni[1,2], Lapo Faggi[1,2], Marco Gori[1,3], and Stefano Melacci[1]

[1] DIISM, University of Siena, Italy
[2] DINFO, University of Florence, Italy
[3] Inria, Lab I3S, MAASAI, Universitè Côte d'Azur, France
mtiezzi@diism.unisi.it, simone.marullo@unifi.it, alessandro.betti@inria.fr, {enrico.meloni,lapo.faggi}@unifi.it, {marco,mela}@diism.unisi.it

Abstract. The classic computational scheme of convolutional layers leverages filter banks that are shared over all the spatial coordinates of the input, independently on external information on what is specifically under observation and without any distinctions between what is closer to the observed area and what is peripheral. In this paper we propose to go beyond such a scheme, introducing the notion of Foveated Convolutional Layer (FCL), that formalizes the idea of location-dependent convolutions with foveated processing, i.e., fine-grained processing in a given-focused area and coarser processing in the peripheral regions. We show how the idea of foveated computations can be exploited not only as a filtering mechanism, but also as a mean to speed-up inference with respect to classic convolutional layers, allowing the user to select the appropriate trade-off between level of detail and computational burden. FCLs can be stacked into neural architectures and we evaluate them in several tasks, showing how they efficiently handle the information in the peripheral regions, eventually avoiding the development of misleading biases. When integrated with a model of human attention, FCL-based networks naturally implement a foveated visual system that guides the attention toward the locations of interest, as we experimentally analyze on a stream of visual stimuli.

Keywords: Foveated Convolutional Layers · Convolutional Neural Networks · Visual Attention.

1 Introduction

In several visual tasks the salient information is distributed in regions of limited size. Objects of interest do not typically occupy the whole visual field, while peripheral areas could contain both relevant or redundant (if not misleading) information. Processing the whole visual scene in a uniform manner can lead to the development of learning machines which inherit spurious correlations from the training data [18,23], that might behave in an unexpected manner at inference time, when exposed to out-of-distribution inputs. A *foveated* artificial vision

system is characterized by a high-acuity fovea, at the center of gaze and a lower resolution in the periphery. Several recent approaches implement this principle by transforming the input [4,22], for example blurring the image periphery [16] or sub-selecting a portion of it. Many other works introduced foveation patterns into a variety of tasks [8], architectures [24] and rendering operations [11]. Foveation in machine vision systems has been investigated both for its computational advantages [15] and for its representational and perceptual consequences, it can play a relevant role in terms of reducing undesirable correlations, noise dependence and weakness to adversarial attacks [10,17].

The importance of foveated vision systems clashes with how classic 2D convolutional layers are designed, where all the input locations are treated the same way, exploiting and sharing the same bank of filters over all the image plane [1]. This entails an architectural prior, implicitly assuming that all the input locations equally contribute to the learning process of the layer filters. From the perspective of the computational costs, extracting convolutional features requires the same computational budget over all the spatial locations. Transformer architectures and related models [5,20,21] marked a paradigm shift towards the removal of the inductive bias induced by convolutional layers, thanks to the self-attention mechanism which basically gives different importance to sub-portions of the vision field. However, this actually takes place due to further operations that are applied to predict the importance of the convolutional features extracted on image patches, and not due to an inherently foveated computational scheme, with low computational efficiency. Similar considerations hold for the efficiency of Locally Connected Layers (LCL) [6,13], that implements different filters for each local receptive field. Moreover, LCLs hinder the generalization capability of the network, losing interesting properties (invariances) and not capturing some correlations due to their strong locality [14].

Recent activity in the context of modeling human attention [26] has shown that it is possible to predict human-like scanpaths to tell deep networks what are the important locations to "observe/focus", thus filtering out non-relevant information [19]. When paired with the aforementioned properties and benefits of foveated visual systems, this calls to the need of developing neural models that can naturally and efficiently exploit the information on what is focused. Inspired by this intuition, we introduce a novel kind of foveated neural layer for computer vision, named Foveated Convolutional Layer (FCL), that rethinks the role of classic 2D convolutions. FCLs go beyond the idea of exploiting the same filters over all the spatial coordinates of the input stimuli, formalizing the idea of location-dependent foveated convolutions. Given the coordinates of a point of interest, also referred to as focus of attention (FOA), either coming from external knowledge on the task at hand or generated by a scanpath predictor [26], an FCL extracts feature maps that depend on the FOA coordinates and on where convolution is evaluated, giving a different emphasis to what is closer and farther from the FOA. In particular, FCLs perform a finer-grained processing in the focused areas (*foveal* region), and a progressively coarser processing when moving far away (*peripheral* regions). We propose several variants of FCL, which

differ in the way this principle is implemented. One of the proposed instances of FCLs easily leads to faster processing with respect to classic convolutional layers, allowing the user to select the appropriate trade-off between the processing capability and computation streamlining, where the reduction of per-pixel floating point operations (faster processing) is due to the coarser feature extraction in non-focused areas. We show how some instances of FCLs can loosen the weight sharing constraint of classic 2D convolutions [13], limiting it to portions of the image at similar distances from the FOA, thus introducing a FOA-based form of locality in the connections [6]. When integrated with the dynamic model of human attention of [26], FCL-based networks naturally implement an efficient foveated visual system that guides the attention toward plausible locations of interest, leveraging peripheral low-budget computations, as we experiment in the context of a continual stream of visual stimuli.

The scope of our work is different from the one of Recurrent Attention Models [12] (and related work), that iteratively process the input, focussing on different portions of it, and learning to identify what is more relevant for the task at hand [9]. In these models, the way attention behaves is intrinsically interleaved with the task-related predictor, either by means of non-differentiable components [12] or differentiable approximations, while what we study is indeed completely agnostic to the source of the attention coordinates. Moreover, this paper is not oriented toward designing systems that make predictions as the outcome of a dynamic exploration of the input, being potentially complementary to the aforementioned approaches and other dynamic models [7]. The idea of re-structuring the kernel function is also present in Locally Smoothed Neural Networks (LSNNs) [14], that, however, are based on the idea of factorizing the weight matrix to determine the importance of different local receptive fields.

In detail, the contributions of this paper are the following: (i) We propose Foveated Convolutional Layers (FCLs) to implement location-dependent foveated processing, investigating several out-of-the-box FCLs. (ii) We study how FCLs can be stacked or injected into neural architectures, reducing the overall number of floating point operations and running times, as experimentally investigated in multiple tasks. (iii) Thanks to faster processing on the peripheral areas, we implement an all-in-one foveated visual system that can be used to drive the gaze patterns of a focus-of-attention trajectory predictor, extending a state-of-the-art scanpath model [26]. (iv) We evaluate the foveated visual system in continual learning, manipulating attention at a symbolic level, coherently with the skills that are progressively gained by the network.

2 Foveated Convolutional Layers

Let us consider an input image/tensor $I\colon \Omega \to \mathbb{R}^c$ with c channels, where $I(x)$ is the c-element vector at coordinates $x = (x_1, x_2) \in \Omega$, being Ω the domain to which the spatial coordinates belong. Let us also introduce a 2D convolutional layer composed of a bank of F kernels/filters. Without any loss of generality, and for the sake of simplicity, we restrict the following analysis to the case of

$c = 1$. For each kernel $k^j : \mathbb{R}^2 \to \mathbb{R}$, $j = 1, \ldots, F$, the convolution between I and k_j is defined as follows,

$$o^j(x) = \left(k^j * I\right)(x) = \int_\Omega k^j(\tau) I(x - \tau) d\tau = \int_\Omega I(\tau) k^j(x - \tau) d\tau. \quad (1)$$

where $o^j(x)$ denotes the j-th output feature map computed at coordinates x. Notice that, as usual, the same filter is shared over all the spatial locations. This implies that all $o^j(x)$'s, $\forall x$, are the outcome of having applied the exact same filter function, after having centered it in x. However, from a very qualitative standpoint, this clashes with the fact that humans do not process the visual scene in such a uniform manner. The extraction of visual information depends on what the gaze is specifically observing, $a \in \Omega$. What is closer to a, the *foveal* region, is not processed the same way as what is far from it. Usually, a finer-grain processing is applied when close to a, while a coarser visual representation is modeled as long as we depart from a. It is convenient to think that the former is related to a larger usage of the computational resources, while the latter is associated to less expensive processing.

We propose a novel class of convolutional layers, named Foveated Convolutional Layers (FCLs), that make convolution dependent on a given location of interest a, and that do not exploit the exact same filter over all the x's. The information on a might come from additional knowledge (e.g., knowing the location of an object or simply focussing on the center of the image in Image Classification) or from a model of human attention that predicts where to focus, both in static images and videos [26,2]. In FCLs, the kernel exploited for the convolution operation in Eq. 1 becomes a function of a, in order to model the dependency on the location focused by the gaze, and also function of x, to differentiate the way convolution is performed in different locations of the image plane. For example, the notion of foveated processing implies that a coarser computation is performed when x is far from a. We propose to implement this behaviour by transforming the original kernel k^j through a spatial convolution with a newly introduced function μ, that depends both on a and x, and, in particular, on the relative location of x with respect to a,

$$\tilde{k}^j_{x,a}(z) = \left(\mu_{\theta, x-a} * k^j\right)(z) := \int_\Omega \mu(\theta, x - a, z - \xi) k^j(\xi) d\xi. \quad (2)$$

We refer to μ as the modulating function, while θ are its structural parameters. Notice that when $c > 1$, k^j includes c 2D spatial components, and the spatial convolution of Eq. 2 is intended to be applied to each of them. Features $o(x)$ are obtained as in Eq. 1, replacing k^j with $\tilde{k}^j_{x,a}$ from Eq. 2.

This definition paves the road to a broad range of instances of FCLs that differ in the way in which the modulating function $\mu(\theta, \cdot, \cdot)$ is defined, and in how we make operations less costly when departing from the FOA, that will be the subject of the rest of this section, and that are briefly summarized in Fig. 1 (top-left). In 2D convolutional layers, k^j is assumed to be defined on a limited region that, in the discrete case, is ($c\times$) $K \times K$. The portion of I (resolution $w \times h$)

that is covered by k^j when computing such a discrete convolution at a certain location x is what is usually referred to as receptive input. The major assumption that we follow in designing out-of-the-box FCLs is that features extracted in the peripheral regions (far from a) or in the focused regions (close to a) should be about input portions of the same size, to avoid introducing strong biases in the nature of the features extracted when varying x or a. In other words, all the filters $\tilde{k}^j_{x,a}$, regardless of x and a, must cover a receptive input of the same size. We term this condition the *uniform spatial coverage assumption*.

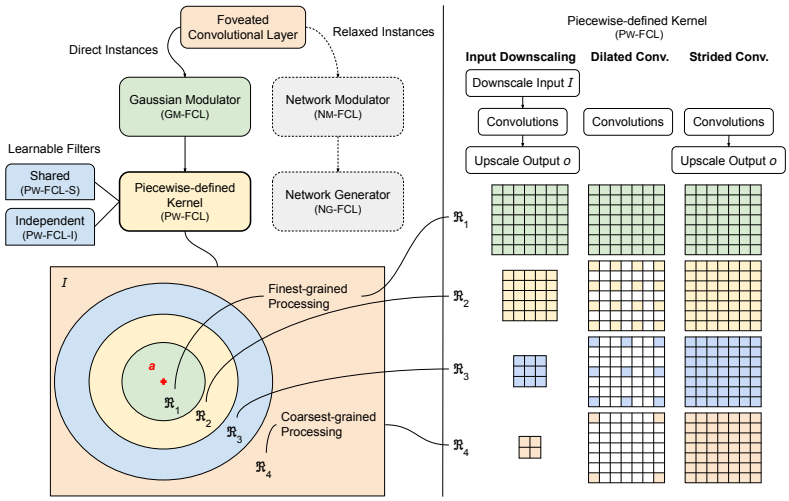

Fig. 1: Top-left: out-of-the-box FCLs. Bottom-left: example of $R = 4$ regions in the piecewise-defined kernel case, when the attention a is given. Right: three strategies (one-per-column) to implement a piecewise-defined kernel, with examples of spatial coverage of the 4 region-wise kernels (coordinates not covered due to dilation are blank). We report right after the strategy name further operations needed to fulfil the uniform spatial coverage assumption.

The most basic instance of FCLs that directly applies the idea of a finer-grade feature extraction around a and a coarser processing in the periphery, can be obtained by blurring kernel k^j with increasing intensity of the blurring operation as long as we move farther from a. We can achieve this behaviour by selecting $\mu(\theta, x-a, \cdot)$ to be a Gaussian function $g(\sigma, x-a, \cdot)$, or, more compactly, $g_{\sigma,x-a}$, characterized by a standard deviation $\sigma(x-a)$ that depends on the distance between x and a. FCLs that exploit such Gaussian modulator are referred to as GM-FCLs, and are based on σ defined as

$$\sigma(x-a) = \hat{\sigma}_a \cdot \ell\left(\|x-a\|^2\right) + \hat{\sigma}_A \cdot \left(1 - \ell\left(\|x-a\|^2\right)\right), \tag{3}$$

being $\cdot$ the classic multiplication, ℓ a function that is 1 when $x = a$ and it is 0 when $\|x-a\|$ reaches the maximum possible distance considering the image

resolution, while σ_a and σ_A are the standard deviations on a and on the farthest location from it.[4] Due to the commutative property of the convolution (when Ω is $\mathbb{R}^2$), putting together Eq. 2 and Eq. 1 and replacing $\mu_{\theta,x-a}$ with the $g_{\sigma,x-a}$, we have $o^j(x) = (\tilde{k}^j_{x,a} * I)(x) = (g_{\sigma,x-a} * k^j_{x,a} * I)(x) = (k^j_{x,a} * (g_{\sigma,x-a} * I))(x)$. This shows that we can implement this type of FCLs by blurring I with a location-dependent Gaussian, filling the gap between FCLs and the common idea of blurring the visual scene with a progressively increasing levels of intensity [15,16]. However, the computational burden is larger than classic 2D convolutional layers, due to the additional convolution with $g_{\sigma,x-a}$.

If σ of Eq. 3 is modeled with a piecewise-constant function defined in non-overlapping ranges involving its argument, such as $\rho_i \leq \|x - a\| < \rho_{i+1}$, $i = 1, \ldots, R-1$, $\rho_1 = 0$, $\rho_R = \infty$, then $\tilde{k}^j_{x,a}$ in Eq. 2 becomes a piecewise-defined kernel. In other words, once we are given a, the exact same kernel is used to compute features $o^j(x)$ of Eq. 1, for all x's which fall inside the same range. Moreover, as long as σ returns larger standard deviations, $\tilde{k}^j_{x,a}$ becomes a blurrier copy of k^j, leaving room to approximated representations that reduce its $K \times K$ spatial resolution. These considerations open to the definition of another instance of FCLs that is specifically aimed at exploiting foveated processing to speedup the computations, giving the user full control on the trade-off between computational cost and the level of detail of the features extracted when moving away from a. In piecewise-defined FCLs (Pw-FCLs), the input image is divided into $R \geq 2$ regions $\mathcal{R}_1, \ldots, \mathcal{R}_R$, centered in a and with no overlap, e.g., in function of $\|x - a\|$ as previously described, naturally introducing a dependency on the focused location, as shown in Fig. 1 (bottom-left).[5] The cost of the convolution operation is controlled by a user-customizable *reduction factor* $0 < r_i \leq 1$, defined for each $\mathcal{R}_i$, where $r_1 = 1$ and $r_{i+1} < r_i, \forall i$. In particular, the cost of computing a convolution in $x \in \mathcal{R}_i$ is forced to be $r_i\mathcal{C}_1$, where $\mathcal{C}_1$ is the cost of a convolution for $x \in \mathcal{R}_1$. This means that convolutions in peripheral regions are performed in a faster way than those in regions closer to a. As we will describe in the following, there are multiple ways of fulfilling this computational constraints by introducing a coarser processing. In turn, coarser processing makes Pw-FCLs exposed to less details and less data variability when far away from a, that can result in a more data-efficient learning in non-focused areas.

We propose three different strategies for enforcing the cost requirement imposed by the reduction factor, summarized in Fig. 1 (right). The first two ones are based on the fact that the cost of convolution is directly proportional to the number of spatial components of the kernel, thus the computational burden can be controlled by reducing the size of the kernel defined in each region in function of r_i. However, a smaller kernel size implies covering smaller receptive inputs, thus violating the previously introduced uniform spatial coverage assumption.

[4] In our experiments we used an exponential law, with $\hat{\sigma}_a$ almost zero and $\hat{\sigma}_A = 10$. Function g is computed on a discrete grid of fixed-size 7×7.

[5] The innermost region $\mathcal{R}_1$ is then a circle, and the other regions are circular crowns with increasing radii. The outermost region $\mathcal{R}_R$ is simply the complementary area. We also tested the case of a squared $\mathcal{R}_1$ and frame-like $\mathcal{R}_i$, $i > 1$.

For this reasons, further actions are needed in order to ensure such a condition is fulfilled, leading to two variants of Pw-FCLs, based on *input downscaling* and *dilated convolutions* [25], respectively. The former appropriately downscales the input to compensate the kernel reduction, and it requires then to upscale the feature maps to match their expected resolution (Fig. 1, top-right). The third strategy consists in not reducing the kernel and relying on *strided convolutions*, thus skipping some x's, that still requires to upscale the resulting feature maps (more details in Appendix A.1).

The piecewise defined kernel is the outcome of adapting the same base kernel k^j over the different regions, thus we denote what we described so far as Pw-FCL with *shared* weights, or, more compactly, Pw-FCL-S (Fig. 1, top-left). This implies that all the region-defined filters share related semantics across the image plane, due to the shared nature of the learnable k^j. A natural alternative to this model consists in using *independent* learnable filters in each region. In this case, referred to as Pw-FCL-I, features extracted in different areas could be fully different (or not) and associated to different (or same) meanings. Of course, if the information stored in the foveal and peripheral areas share some properties, then a Pw-FCL-S might be more appropriate.

It is interesting to show how the framework of FCLs can be further extended along directions that depart a little bit from the idea of foveated processing, but that are still oriented toward location-dependent processing in function of the FOA coordinates a. Of course, a detailed analysis of them goes beyond the scope of this paper. In particular, the degrees of freedom of FCLs can be extended when $\mu(\theta, x-a, z)$ is implemented with a neural network modulator, naming these layers Nm-FCLs. It is convenient to think about such modulator as a multi-layer feed-forward network with input $x-a$, and that yields $N \times N$ real numbers as output, which is the filter that modulates k^j in the discrete counterpart of Eq. 1. Output values are normalized to ensure the filter sums to 1 in the $N \times N$ grid, and we set $N = 7$ in our experiments, in analogy with the way $g_{\sigma,x-a}$ was discretized. Another step in relaxing the formulation of FCLs consists in fully re-defining $\tilde{k}_{x,a}$ of Eq. 2 as a discrete filter whose values are generated by a neural network. The net acts as a neural generator, thus Ng-FCLs, that learns to produce a bank of F distinct ($c\times$) $K \times K$ filters given $x-a$, to be used when computing convolution in coordinates x. It is easy to see that the computation/memory burden of generating a different kernel for (almost) every location x in the image plane, by means of a multi-layer network, makes it less practical than the other described types of FCLs, even if it opens to further investigations into this direction (details in Appendix A.2/A.3 suppl. material).

2.1 Learning with Attention in Foveated Neural Networks

From the point of view of the input-output, FCLs are equivalent to classic 2D convolutional layers, with the exception of the additional input signal a. As a result, they can be straightforwardly stacked into deep architectures, learning the kernel components by Backpropagation. It is pretty straightforward to exploit (single or stacked) FCLs to extract features for each pixel of the $w \times h$ network

input. Whenever FCLs are used after pooling layers or, in any case, after having reduced the resolution of the latent representations, a must be rescaled accordingly. Depending on the properties of the considered task, it might be convenient to use FCLs in the last portions of a deep architecture, at the beginning of it, or in other configurations, for example stacking FCLs in a way that the lower layers are mostly specialized in fine-grained processing over large areas around a, that progressively get smaller in the upper layers.

In this paper, we study the case in which a is given, either due to specific external knowledge or when it is estimated by a human-like model of visual attention. A very promising model of human attention was recently proposed in [26], well suited for generic free-viewing conditions too. Such a model estimates attention a at time t, i.e., $a(t)$, as a dynamic process driven by the following law

$$\ddot{a}(t) + \gamma a(t) - E(t, a(t), \{m_j(x,t),\ j = 1, \ldots, M\}) = 0, \tag{4}$$

where E is a gravitational field that depends on a distribution of masses, each of them indicated with m_j, and γ is a customizable weight that controls dissipation. Each mass attracts the attention in a way that is proportional to the value of $m_j(x,t)$, eventually tuned by a customizable scalar. The authors of [26] considered the case of $M = 2$, with m_1 and m_2 that yield high values when x includes strong variations of brightness and motion, respectively. However, other masses could be added over time, as briefly mentioned in [26] but never investigated. Let us introduce a stream of visual information, being I_t the frame at time t, and a neural network $f(I_t, \omega_t)$ with weights ω_t. In a C-class semantic labeling problem, f returns a vector of C class membership scores for each image coordinate x, i.e., $f(I_t, \omega_t)(x)$. The notation $f_{i,j,z,\ldots}(I_t, \omega_t)(x)$ is used to consider only the scores of the classes listed in the subscript. Let us assume that the user is interested in forcing the model to focus on specific object classes h and z. For example, in a stadium-like scene during a soccer match, the model should be attracted by the players, by the ball, and not by all the people in the bleachers or by the sky. We can pool the class membership scores to simulate a novel mass function, such as $m_q(x,t) = (\max f_{h,z}(I_t, \omega_t)(x))$, so that the attention model is automatically attracted by those pixels that are predicted as belonging to classes h or z (or both—it holds for any number of classes). If f is based on PW-FCLs, then peripheral areas will be characterized by faster processing and slightly lower prediction quality (due to the coarser feature extraction), as it happens in the human visual system, and attention will be also influenced by such predictions.

3 Experiments

We implemented FCLs using PyTorch,[6] performing experiments in a Linux environment, using a NVIDIA GeForce RTX 3090 GPU (24 GB). We performed four types of experiments on four different datasets, aimed at comparing FCLs with classic convolutional layer in a variety of settings. Before

[6] `https://github.com/sailab-code/foveated_neural_computation`

going into further details, we showcase the speedup obtained when using PW-FCLs ($R = 4$ regions, $r_2 = 0.7$, $r_3 = 0.4$, $r_4 = 0.1$), comparing the 3 different strategies of Section 2 and Fig. 1 (right), also including a baseline with a classic convolutional layer, and a vanilla configuration of PW-FCLs without any cost reduction (i.e., $r_i = 1$, $\forall i$). The bounding box of each region $\mathcal{R}_i$ (up to $i = R - 1$) linearly increases from 10% of w up to 70% of it. Fig. 2 (left) shows the average time required to process an input I with $c = 64$ channels, whose resolution varies in $\{256 \times 256, 512 \times 512, 1024 \times 1024, 1920 \times 1080\}$, using $K = 11$, while Fig. 2 (right) is about 512×512 and variable base kernel size K. Overall, the proposed reduction strategies achieve big improvements with respect to classic convolutional layers, with better scaling capabilities for larger resolutions and kernels. The region-management overhead, that is evident in the vanilla case, is completely compensated by all the reduction mechanisms. The downscaling strategy is the faster solution (of course, reducing the number of regions or r_i's yields even better times–see Appendix C.1, supplementary material), that is what we will use in the following experiments.

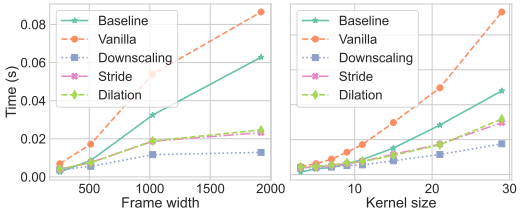

Fig. 2: Inference time of the 3 PW-FCLs strategies, of a convolutional layer (baseline), of a fixed-kernel-size PW-FCLs (vanilla). Left: changing input resolution. Right: varying K.

The first task we consider is referred to as DUAL-INTENTION, and it consists in classifying 200×200 images in one out of 10 classes. In each image two different digits are present (each of them covering $\approx 28 \times 28$ pixel, MNIST dataset), one in the middle of it and one closer to the image border. A special *intention sign* is also present, far from the middle (one of {□, ●})). If the sign is □, then the image class is the one of the digit placed close to the border. When ● is used the target class is the one of the digit in the center of the image—Fig. 3 (left). The challenging nature of the task comes from fact that the training data only include intention signs randomly located in locations close to the bottom border of the image, while in the test data the sign is randomly located closer to the top border (the peripheral digit stands on the opposite side with respect to the intention sign). In order to gain generalization skills, the network must be able to learn representations that are pretty much location-independent in the peripheral area, and to differentiate them from the ones developed in the central area. We compared multiple convolutional feature extractors, each of them followed by pooling operation and a classification head (128 hidden neurons, ReLU). The first extractor is a CNN with 4 layers (denoted with CNN*), with final *global-max* pooling. Then, we focused on a single classic convolutional layer (CL) followed by *max* pooling (stride 10) or *global-max* pooling. We also introduced another type of pooling that is aware of the existence of two main image regions, that we simulated with a 33×33 box surrounding the image center and

the rest of the image, performing max pooling in each of them and concatenating the results. We refer to it as region-wise max pooling (*reg-max*). Finally, we extracted features with FCLs, leveraging PW-FCL-S and PW-FCL-I with FOA positioned at the image center and $R = 2$ regions (a 33×33 box region and the complementary one—in this case, region-wise max pooling is always exploited). We report the average test accuracy over three runs (and standard deviations) in

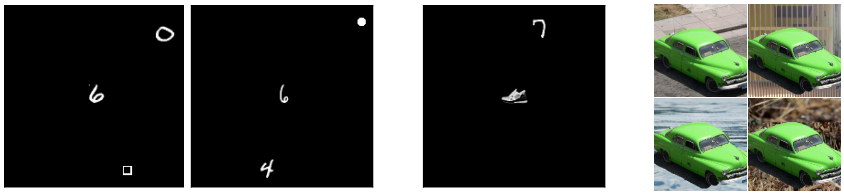

Fig. 3: Left: train and test samples from DUAL-INTENTION dataset (class 0, and class 6, respectively). Middle: training sample from STOCK-FASHION (class is "in stock shoes"). Right: sample from the data by Xiao et al. [23] in its ORIGINAL, MIXED-SAME (1st row), MIXED-RAND and MIXED-NEXT (2nd row) versions.

Table 1, considering a 1K samples dataset and a 10K samples one (see Appendix C.2, supplementary materials, for the model validation procedure; r_2 is always < 1). Models based on *global-max* pooling, being them deeper (CNN*) or shallower (CL+*global-max*), loose all the location-related information, thus they do not distinguish among what is in the middle of the image and what is in the peripheral area. CL-MAX only aggregates a few spatial coordinates, thus yielding (relaxed) location-related features that let the classifier learn that the intention sign that is expected to be always at the bottom. Thanks to region-wise max pooling, the CL+*reg-max* network achieves good results, even if at the same computational cost of CL. Interestingly, when a PW-FCL is used, strongly reducing the computational burden, we still achieve similar performances to CL+*reg-max*, and in the low-data regime both PW-FCLs outperform it. The foveated computations in the peripheral region implicitly filters the image, reducing noise and smoothing samples, that turns out in making FCLs more data efficient (1K dataset). In Table 2 we restrict our analysis in the case of $K = 15$ and $F = 64$, comparing the two best models equipped with *reg-max* pooling, and showing the performance relative to the different values of the reduction factor r_2 (the one of the peripheral region), along with the number of performed floating point operations (GFLOPs).[7] Interestingly enough, the proposed PW-FCL-S can achieve a very similar performance with respect to CL+*reg-max*, at a fraction of its cost.

Our next experimental activity is about a task that is based on what we refer to as STOCK-FASHION dataset, that is somewhat related to the previous one, since we still have an entity in the middle of the image and another one

[7] We measured the number of floating point operations using the PyTorch profiling utilities `https://pytorch.org/docs/stable/profiler.html`.

Table 1: DUAL-INTENTION. Average test accuracy (and std) in low (1K) and large (10K) data regimes.

Model	1K	10K
CNN*	54.3 ± 3.3	57.3 ± 1.0
CL+*global-max*	53.0 ± 1.4	53.1 ± 0.5
CL+*max*	24.7 ± 1.9	47.6 ± 1.1
CL+*reg-max*	71.0 ± 2.4	**96.5** ± 0.0
PW-FCL-S	**73.7** ± 0.5	95.4 ± 0.3
PW-FCL-I	72.3 ± 0.5	95.7 ± 0.1

Table 2: DUAL-INTENTION. The case of $K = 15$ and $F = 64$, varying the reduction factor r_2 (peripheral region).

Model	r_2	GFLOPs	ACCURACY	
			1K	10K
CL+*reg-max*	-	0.996	71.0	**96.5**
PW-FCL-S	0.1	0.078	56.7	87.7
	0.25	0.159	69.7	94.7
	0.5	0.398	**73.7**	95.4

closer to the border. The middle area contains patterns that are harder to classify, simulated with samples from Fashion-MNIST dataset, paired with MNIST digits placed in the upper image portion at training time, and in the bottom image portion at test time. The goal is to recognize the class of the fashion item in the middle (10 types) and also it is largely available *in-stock* or with *limited* availability, in function of the value of the peripheral digit (0 to 4: limited; 5 to 9: in-stock)—Fig. 3 (middle). Hence, the total number of classes is 20. The peripheral information is still crucial for the final purpose of classifying the image, but it is of different type with respect to what is in the middle. We follow the same experimental setup of the previous experiment (10K samples) and we report in Table 3 the test accuracy of the compared models. Once again, region-wise pooling-based models are the best performing ones. The independent filters of PW-FCL-I are able to learn dedicated properties for the foveal region and for the peripheral region, that in this task do not share any semantic similarities, leading to better performance (recall that $r_2 < 1$). Recognizing the peripheral digits is relatively simpler with respect to the fashion items, so that the foveated computational scheme perfectly balances the computational resources over the image. In Fig. 4 we report the test accuracy obtained by the foveated models, restricting our analysis to the case of $K = 15$ and $F = 128$ and varying the reduction factor r_2. We remark that even with an evident reduction of their computational capabilities ($r_2 \in \{0.1, 0.25\}$), the models yield a robust representational quality that allows the classifiers to reach large accuracies, with a preference for PW-FCLS-I.

In our next experimental activity, we consider a context in which the information in the background of an image might or might-not help in gaining robust generalization skills, thus ending up in learning *spurious correlations*. An out-of-distribution context change in the background usually leads to poor performances in the classification accuracy of deep neural models, as studied in the benchmark of Xiao et al. [23], based on ImageNet. Multiple test sets are made available, ORIGINAL, MIXED-SAME, MIXED-NEXT, MIXED-RAND, where the background of the images is left untouched, replaced with the one from an-

Table 3: STOCK-FASHION, 10K, average test accuracy (and std).

Model	ACCURACY
CNN*	83.4 ± 1.6
CL+*global-max*	79.8 ± 1.4
CL+*max*	42.4 ± 0.7
CL+*reg-max*	85.9 ± 0.9
PW-FCL-S	83.1 ± 0.1
PW-FCL-I	**86.0** ± 0.3

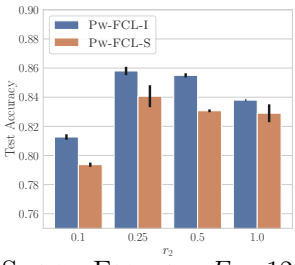

Fig. 4: STOCK-FASHION, $F = 128$, $K = 15$, varying the reduction factor r_2.

other example of the "same" class, of the "next" class, of a "random" class, as shown Fig. 3 (right). We argue that the injection of FCLs into neural architectures, even if very shallow or simple, favors the model robustness to issues related to background correlations. In particular, several advantages come from the fact that FCLs reserve the finest-grained processing solely to the focused area, whilst the peripheral portion of the frame is processed via coarser resources/reduced resolution. Thanks to this architectural prior, the variability of features extracted from peripheral regions is reduced and they are harder to be tightly correlated with a certain class. We considered the already described CNN* as reference, replacing the *first* or *last* convolutional layer with a FCL, evaluating all the types of FCLs of Section 2 (see Appendix C.4 for more details). In this case, the FOA a can be either located at the center of the picture, or on the barycenter of the main object of each picture (the one that yields the class label). Fig. 5 shows our experimental findings in the latter case, analyzing the first three classes of the dataset (dog, bird, vehicle)–similar results with the whole dataset (Appendix C.4). These results support the intuition that a standard CNN* suffers from the background correlation issues, when comparing accuracy in ORIGINAL (90.8%) and MIXED-NEXT (57.1%), where the background is always extrapolated from a different class. As expected, thanks to the introduction of FCLs, the performance drop is remarkably reduced in most of the cases. In PW-FCL-I (*all*) such drop is almost halved (accuracy goes from 86.3% to 68.1%—remind that this model features a reduced number of parameters with respect to CNN*). Using neural modulators or generators (NM-FCL, NG-FCL) was less effective in the *all* and *first* architectures, while in the *last* setting they yielded outstanding performances in terms of generalization both in the ORIGINAL and MIXED-NEXT test sets. This is due to the fact that these models are not explicitly foveated, since they can freely learn how to perform convolutions at different coordinates with no restrictions, thus they can still learn to specialize in background features. However, when used in the last stages of a hierarchy, they actually learn to further refine the learned representations to better focus on the main object.

Going beyond experiments on static images, we studied an incremental learning setting in which a video stream is presented, frame-by-frame, to an FCL-based foveated network, with the goal of learning to classify each single pixel as belonging to one of 20 classes or as being part of the background. The attention

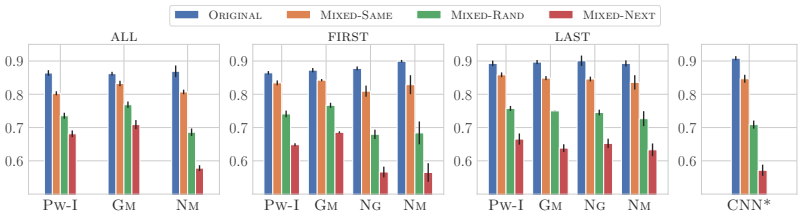

Fig. 5: Background correlations. Accuracy in the first three classes of the dataset in [23]. We dropped the string FCL from all the model names (except CNN*).

model of [26] is exploited to explore the visual scene, yielding $a(t)$ for each frame (Eq. 4). The visual stream is composed of a sequence of what we refer to as *visual stimuli*, each of them involving 20 unique pictures of handwritten digits and letters from the EMNIST dataset, also generically referred to as "entities" (10 digits and 10 letters–A to K, excluding I that is too similar to digit one). A single entity covers $\approx 25 \times 25$ pixels. For each stimulus (2260 frames) an entity enters the scene from top, slowly moving down until it reaches the bottom of the frame, and it stands still. A different entity does the same, until the scene is completely populated by 20 entities with no overlap. Finally, the entities leave the scene moving down (in reverse order, sequentially). The network processes two sequences of training stimuli, learning in a supervised manner, initially receiving supervisions on digits only (first training sequence) and then on letters only (second training sequence). We implemented a simple rehearsal-based continual learning scheme to store a small subset of frames ($\approx 20-40$) for learning purposes [3], that are selected whenever the attention $a(t)$ performs a saccadic movement (details in Appendix C.5, supplementary materials). After each training sequence, a test sequence is presented, generating end evaluating multiple foveated-network-dependent masses that attract the attention toward custom salient elements, as described in Section 2.1. Masses are initially about all the *digits*, *even* digits, *odd* digits. Then, in the second test sequence, they are about *letters*, the first 3 digits and first 3 letters (*mix1*), the last 3 digits and last 3 letters (*mix2*) respectively. Masses are activated in mutually exclusive manner in evenly partitioned portions of each test sequence. All the types of FCLs of this paper are evaluated, excluding the network-generated ones that resulted to be too memory demanding. The foveated network is composed of 2 convolutional layers and a final FCL (*last*), or of 3 FCLs (*all*), $R = 2$, $r_2 = 0.25$, compared with a reference network with classic convolutional layers only (CNN). The $w \times h$ pixel-wise feature vectors of the last layer are processed by a classifying head that marks as "background" those coordinates with too small prediction confidence.

Table 4 reports our results for a stream at the resolution of 200×200, and it shows that foveated models are able to perform similarly to (or even better than) CNN, even if in Pw-FCLs the number of floating point operations is approximately 3 to 5 times smaller. It is interesting to see that Nm-FCLs (*last*) and Gm-FCLs yield significantly better results in the digit/letter pixels, com-

Table 4: Stream of visual stimuli, 200×200. GFLOPs and average accuracy (3 runs) in classifying digits, letters and background.

	Model	GFLOPs	Digits	Letters	Back.
	CNN	6.13	71.9	23.7	84.0
Last	PW-FCLs	2.72	69.2	25.3	80.6
Last	GM-FCLs	6.19	70.4	22.7	79.8
Last	NM-FCLs	6.23	81.0	28.4	67.7
All	PW-FCLs	1.16	70.8	24.7	80.0
All	GM-FCLs	6.25	75.2	24.2	77.6
All	NM-FCLs	6.37	58.0	25.3	52.0

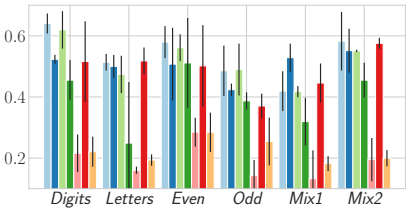

Fig. 6: Stream of visual stimuli, 200×200. Fraction of time spent on the salient elements of the test stimuli (colors are about the models of Table 4).

pared to CNN, but they have more difficulties in classifying background, since the modulated kernel tends to respond in a less precise manner closer to the digit/letter boundaries. In Fig. 6 we report the fraction of time spent in those pixels that are about elements for which an apposite mass was created (colors are about different models, see Table 4). Results show that foveated networks based on PW-FCLs (*last*/blue), GM-FCLs (*last*/light-green, *all*/red) explore the expected elements in a similar manner to what happens in CNN (light-blue). The other models frequently return a small margin between the winning prediction and the other ones, thus the resulting mass is noisy. Fig. 7 shows the saliency map generated by the attention model exploiting foveated networks with PW-FCLs (*last*), from which we can qualitatively observe that the expected salient elements are indeed explored by $a(t)$.

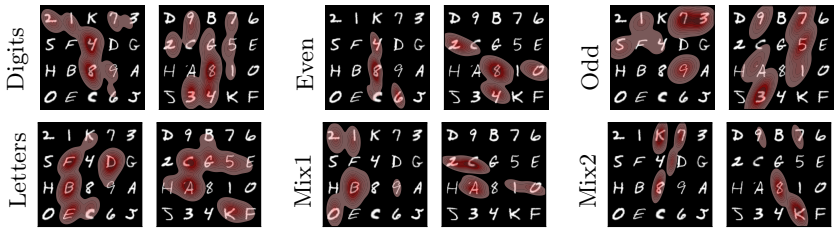

Fig. 7: Attention model of [26] with masses predicted by PW-FCLs (*last*). For each salient element (all *digits*, *even* digits, *odd* digits, *letters*, *mix1*, *mix2*), saliency maps are shown for 2 visual stimuli, when the frame is fully populated (brighter-red: frequently visited by $a(t)$; lighter-red: sporadically visited by $a(t)$).

In order to emphasize the computational gains when using PW-FCLs, we performed the same experiment considering a stream with resolution $1000 \times$

1000. The original size of the digits/letters is left untouched, so that frames are almost composed of background pixels. One might be tempted to simply downscale each frame and provide it to the net, that actually turns out to be a complete failure, since the size of digits/letters become significantly small, as shown in Table 5. Differently, the performance of CNN processing the high-resolution stream is almost matched by Pw-FCLs (*last*, *all*), with a running time that is approximately $70\% - 60\%$ smaller than CNN (the time spent on the salient elements is still comparable to the one of CNN–Appendix C.5). This shows how foveated processing can let the network implement a good trade-off between speed and computational burden, being able to focus and recognize small digits/letters on a large resolution image.

Table 5: Stream of visual stimuli, 1000×1000. Average test accuracy ($\pm$ std, 3 runs), GFLOPs, inference time in classifying digits, letters and background.

Model	GFLOPs	Time (ms)	Digits	Letters	Background
CNN (resize to 200×200)	6.13	1.3 ± 0.04	27.1 ± 1.1	6.0 ± 0.8	94.4 ± 0.6
CNN (full resolution)	153.15	13.2 ± 0.01	77.6 ± 5.4	30.3 ± 1.4	99.5 ± 0.0
Pw-FCLs (last)	63.72	9.1 ± 0.13	76.0 ± 1.5	30.6 ± 1.8	99.4 ± 0.1
Pw-FCLs (all)	18.84	8.6 ± 0.04	74.4 ± 2.3	28.5 ± 1.1	99.4 ± 0.0

4 Conclusions and Future Work

We presented Foveated Convolutional Layers to extract features in function of a focused location with foveated processing. We proposed several instances of this model, emphasizing the one that yields a significantly faster processing than plain 2D convolutions. When injected into a human-like attention model, FCL-based networks naturally implement a user-customizable visual system with fast inference and coarser processing in the peripheral areas. Future work includes the analysis of foveated networks in problems driven by motion invariance principles.

Acknowledgements This work was partly supported by the PRIN 2017 project RexLearn, funded by the Italian Ministry of Education, University and Research (grant no. 2017TWNMH2), and also by the French government, through the 3IA Côte d'Azur, Investment in the Future, project managed by the National Research Agency (ANR) with the reference number ANR-19-P3IA-0002.

References

1. Almahairi, A., Ballas, N., Cooijmans, T., Zheng, Y., Larochelle, H., Courville, A.: Dynamic capacity networks. In: ICML. pp. 2549–2558. PMLR (2016)
2. Borji, A., Cheng, M.M., Hou, Q., Jiang, H., Li, J.: Salient object detection: A survey. Computational visual media **5**(2), 117–150 (2019)

3. Delange, M., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., Slabaugh, G., Tuytelaars, T.: A continual learning survey: Defying forgetting in classification tasks. IEEE TPAMI pp. 1–1 (2021)
4. Deza, A., Konkle, T.: Emergent properties of foveated perceptual systems. arXiv:2006.07991 (2020)
5. 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 16x16 words: Transformers for image recognition at scale. In: ICLR (2020)
6. Gregor, K., LeCun, Y.: Emergence of complex-like cells in a temporal product network with local receptive fields. arXiv:1006.0448 (2010)
7. Han, Y., Huang, G., Song, S., Yang, L., Wang, H., Wang, Y.: Dynamic neural networks: A survey. IEEE TPAMI pp. 1–1 (2021)
8. Kong, T., Sun, F., Liu, H., Jiang, Y., Li, L., Shi, J.: Foveabox: Beyound anchor-based object detection. IEEE TIP **29**, 7389–7398 (2020)
9. Larochelle, H., Hinton, G.E.: Learning to combine foveal glimpses with a third-order boltzmann machine. Advances in NeurIPS **23** (2010)
10. Luo, Y., Boix, X., Roig, G., Poggio, T., Zhao, Q.: Foveation-based mechanisms alleviate adversarial examples. arXiv:1511.06292 (2015)
11. Malkin, E., Deza, A., et al.: Cuda-optimized real-time rendering of a foveated visual system. In: NeurIPS 2020 Workshop SVRHM (2020)
12. Mnih, V., Heess, N., Graves, A., et al.: Recurrent models of visual attention. Advances in neural information processing systems **27** (2014)
13. Ott, J., Linstead, E., LaHaye, N., Baldi, P.: Learning in the machine: To share or not to share? Neural Networks **126**, 235–249 (2020)
14. Pang, L., Lan, Y., Xu, J., Guo, J., Cheng, X.: Locally smoothed neural networks. In: Asian Conference on Machine Learning. pp. 177–191. PMLR (2017)
15. Poggio, T., Mutch, J., Isik, L.: Computational role of eccentricity dependent cortical magnification. arXiv:1406.1770 (2014)
16. Pramod, R., Katti, H., Arun, S.: Human peripheral blur is optimal for object recognition. arXiv:1807.08476 (2018)
17. Reddy, M.V., Banburski, A., Pant, N., Poggio, T.: Biologically inspired mechanisms for adversarial robustness. Advances in NeurIPS **33** (2020)
18. Sagawa, S., Koh, P.W., Hashimoto, T.B., Liang, P.: Distributionally robust neural networks. In: ICLR (2019)
19. Tiezzi, M., Melacci, S., Betti, A., Maggini, M., Gori, M.: Focus of attention improves information transfer in visual features. Advances in NeurIPS **33**, 22194–22204 (2020)
20. Tolstikhin, I.O., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., Yung, J., Steiner, A., Keysers, D., Uszkoreit, J., et al.: Mlp-mixer: An all-mlp architecture for vision. NeurIPS **34** (2021)
21. Trockman, A., Kolter, J.Z.: Patches are all you need? arXiv:2201.09792 (2022)
22. Wang, P., Cottrell, G.W.: Central and peripheral vision for scene recognition: A neurocomputational modeling exploration. Journal of vision **17**(4), 9–9 (2017)
23. Xiao, K.Y., Engstrom, L., Ilyas, A., Madry, A.: Noise or signal: The role of image backgrounds in object recognition. In: ICLR (2020)
24. Yang, J., Li, C., Gao, J.: Focal modulation networks. arXiv:2203.11926 (2022)
25. Yu, F., Koltun, V.: Multi-scale context aggregation by dilated convolutions (2015). https://doi.org/10.48550/ARXIV.1511.07122
26. Zanca, D., Melacci, S., Gori, M.: Gravitational laws of focus of attention. IEEE TPAMI **42**(12), 2983–2995 (2020)