entropy

MDPI

Article

Why Dilated Convolutional Neural Networks: A Proof of Their Optimality

Jonatan Contreras, Martine Ceberio and Vladik Kreinovich *

Department of Computer Science, University of Texas at El Paso, El Paso, TX 79968, USA; jmcontreras2@utep.edu (J.C.); mceberio@utep.edu (M.C.)
* Correspondence: vladik@utep.edu

Abstract: One of the most effective image processing techniques is the use of convolutional neural networks that use convolutional layers. In each such layer, the value of the layer's output signal at each point is a combination of the layer's input signals corresponding to several neighboring points. To improve the accuracy, researchers have developed a version of this technique, in which only data from some of the neighboring points is processed. It turns out that the most efficient case—called dilated convolution—is when we select the neighboring points whose differences in both coordinates are divisible by some constant ℓ. In this paper, we explain this empirical efficiency by proving that for all reasonable optimality criteria, dilated convolution is indeed better than possible alternatives.

Keywords: convolutional neural networks; dilated neural networks; optimality

check for updates

Citation: Contreras, J.; Ceberio, M.; Kreinovich, V. Why Dilated Convolutional Neural Networks: A Proof of Their Optimality. *Entropy* **2021**, *23*, 767. https://doi.org/10.3390/e23060767

Academic Editor: Rafał Rak

Received: 18 April 2021
Accepted: 14 June 2021
Published: 18 June 2021

1. Introduction

1.1. Convolutional Layers: Input and Outpu

At present, one of the most efficient techniques in image processing and in other areas is a convolutional neural network; see, e.g., [1]. Convolutional neural networks include special types of layers that perform linear transformations.

Each such layer is characterized by integer-values parameters $\underline{X} \le \overline{X}$, $\underline{Y} \le \overline{Y}$, $d_{\text{in}} \ge 1$, and $d_{\text{out}} \ge 1$; then:

- the input to this layer consists of the values $F_{d'}(x', y')$, where d', x', and y' are integers for which $\underline{X} \le x' \le \overline{X}$, $\underline{Y} \le y' \le \overline{Y}$, and $1 \le d' \le d_{\text{in}}$; and
- the output of this layer consists of the values $G_d(x, y)$, where d, x, and y are integers for which $\underline{X} \le x \le \overline{X}$, $\underline{Y} \le y \le \overline{Y}$, and $1 \le d \le d_{\text{out}}$.

1.2. Convolutional Layer: Transformation

A general linear transformation has the form

$$G_d(x,y) = \sum_{d'=1}^{d_{in}} \left(\sum_{x'=\underline{X}}^{\overline{X}} \sum_{y'=\underline{Y}}^{\overline{Y}} K_d(x,x',y,y',d') \cdot F_{d'}(x',y') \right), \tag{1}$$

for some coefficients $K_d(x, x', y, y', d')$.

Transformations performed by a convolutional layer are a specific case of such generic linear transformations, where the following two restrictions are imposed:

- first, each value $G_d(x, y)$ depends only on the values $F_{d'}(x', y')$, for which both differences $|x - x'|$ and $|y - y'|$ do not exceed some fixed integer L, and
- the coefficients $K_d(x, x', y, y', d')$ depend only on the differences $x - x'$ and $y - y'$:

$$K_d(x,x',y,y',d') = k_d(x-x',y-y',d') \tag{2}$$

for some coefficients $k_d(i, j, d')$ defined for all pairs (i, j) for which $|i|, |j| \le L$.

The values $k_d(i,j,d')$ are known as a *filter*.

The resulting linear transformation takes the form

$$G_d(x,y) = \sum_{d'=1}^{d_{\text{in}}} \left(\sum_{-L \le i,j \le L} k_d(i,j,d') \cdot F_{d'}(x-i, y-j) \right). \tag{3}$$

Thus, the output $G_d(x,y)$ of a convolutional layer corresponding to the point (x,y) is determined by the values $F_{d'}(x-i,y-j)$ of the input to this layer at points $(x-i,y-j)$ corresponding to $|i| \le L$ and $|j| \le L$. This is illustrated by Figure 1, where, for $L = 1$ and for a point (x,y) marked by an asterisk, we show all the points $(x',y') = (x_0 - i, y_0 - j)$ that determine the values $G_d(x,y)$. For convenience, points (x',y') that do not affect the values $G_d(x,y)$, are marked by zeros.

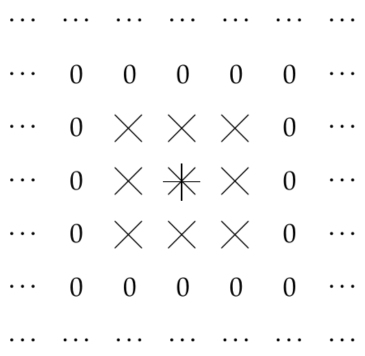

Figure 1. Convolution filter: case of $L = 1$.

For $L = 2$, a similar picture has the following form. This is illustrated by Figure 2.

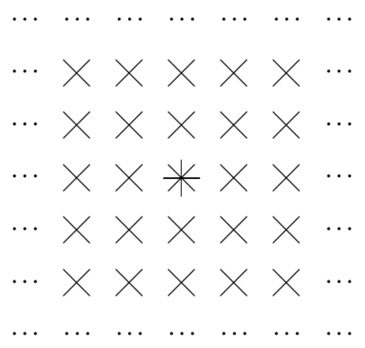

Figure 2. Convolution filter: case of $L = 2$.

1.3. Sparse Filters and Dilated Convolution

Originally, convolutional neural networks used filters in which all the values $k_d(i,j,d')$ for $|i|, |j| \le L$ can be non-zero. It turned out, however, that we can achieve a better accuracy if we consider sparse filters, i.e., filters in which, for some pairs (i,j) with $|i|, |j| \le L$, all the values $k_d(i,j,d')$ are fixed at 0; see, e.g., [2–4].

In Figure 3, we show an example of such a situation, when $L = 2$ and only values $k_d(i,j,d')$, for which both i and j are even allowed to be non-zero.

$$\begin{array}{ccccccccc}
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
\cdots & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\
\cdots & 0 & \times & 0 & \times & 0 & \times & 0 & \cdots \\
\cdots & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\
\cdots & 0 & \times & 0 & \ast & 0 & \times & 0 & \cdots \\
\cdots & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\
\cdots & 0 & \times & 0 & \times & 0 & \times & 0 & \cdots \\
\cdots & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots
\end{array}$$

Figure 3. Case when $L = 2$ and only values $k_d(i, j, d')$ with even i and j can be no-zero.

In general, it turned out that such a restriction works best if we only allow $k_d(i, j, d') \neq 0$ for pairs (i, j) which are divisible by some integer ℓ, i.e., if we take

$$G_d(x, y) = \sum_{d'=1}^{d=d_{\text{in}}} \left(\sum_{-L \le i,j \le L:\ i/\ell \in \mathbb{Z},\ j/\ell \in \mathbb{Z}} k_d(i, j, d') \cdot F_{d'}(x - i, y - j) \right). \tag{4}$$

In this case, the layer's output signal $G_d(x, y)$ can be written in the following equivalent form:

$$G_d(x, y) = \sum_{d'=1}^{d_{\text{in}}} \left(\sum_{-\widetilde{L} \le \widetilde{i}, \widetilde{j} \le \widetilde{L}} \widetilde{k}_d\left(\widetilde{i}, \widetilde{j}, d'\right) \cdot F_{d'}\left(x - \ell \cdot \widetilde{i}, y - \ell \cdot \widetilde{j}\right) \right), \tag{5}$$

where we denoted $\widetilde{L} \overset{\text{def}}{=} L/\ell$, $\widetilde{i} \overset{\text{def}}{=} i/\ell$, $\widetilde{j} \overset{\text{def}}{=} j/\ell$, and $\widetilde{k}_d\left(\widetilde{i}, \widetilde{j}, d'\right) \overset{\text{def}}{=} k\left(\ell \cdot \widetilde{i}, \ell \cdot \widetilde{j}, d'\right)$.

The resulting networks are known as dilated convolutional neural networks, since skipping some points (i, j) in the description of the filter is kind of equivalent to extending (dilating) the distance between the remaining points; see, e.g., [2–4].

1.4. Empirical Fact That Needs Explanation

In principle, we could select other points (i, j) at which the filter can be non-zero. For example, we could select points for which j is even, but i can be any integer. This is illustrated by Figure 4.

$$\begin{array}{ccccccccc}
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
\cdots & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\
\cdots & 0 & \times & \times & \times & \times & \times & 0 & \cdots \\
\cdots & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\
\cdots & 0 & \times & \times & \ast & \times & \times & 0 & \cdots \\
\cdots & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\
\cdots & 0 & \times & \times & \times & \times & \times & 0 & \cdots \\
\cdots & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots
\end{array}$$

Figure 4. Case when $L = 2$ and only values $k_d(i, j, d')$ with even j can be non-zero.

Alternatively, for $L = 2$, as points (i, j) at which $k_d(i, j, d')$ can be non-zero, we could select the points $(0, 0)$, $(0, \pm 1)$, $(\pm 1, 0)$, and $(\pm 2, \pm 2)$, see Figure 5.

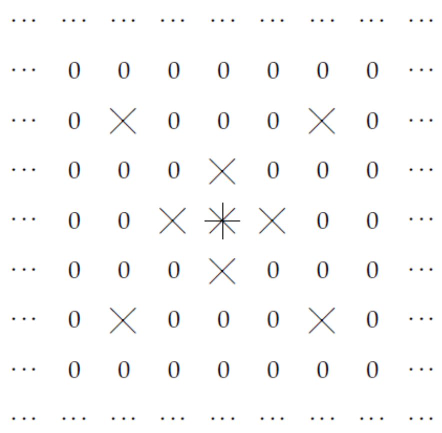

Figure 5. A possible selection of points (i,j) for which $k_d(i,j,d')$ can be no-zero.

However, empirical evidence shows that the selection corresponding to dilated convolution—when we select points for which i and j are both divisible by some integer ℓ—works the best [2–4].

To the best of our knowledge, there is no theoretical explanation for this empirical result—that dilated convolution leads to better results that select other sets of non-zero-valued points (i,j). The main objective of this paper is to provide such an explanation.

Comment. Let us emphasize that the only objective of this paper is to explain this empirical fact; we are not yet at a stage where we can propose a new method or even any improvements to the known methods.

2. Analysis of The Problem

2.1. Let Us Reformulate This Situation in Geometric Terms: Case of Traditional Convolution

In the original convolution Formula (1), to find the values $G_d(x,y)$, the layer's output signal at a point (x,y), we need to know the values $F_{d'}(x',y')$, the layer's input signal at all the points (x',y') of the type $(x-i,y-j)$ for $|i|,|j|\le L$. We can reformulate it by saying that we need to know the values $F_{d'}(x',y')$ at all the points (x',y') at which the ℓ_∞ distance

$$d_\infty((x,y),(x',y'))\stackrel{\text{def}}{=}\max(|x-x'|,|y-y'|), \tag{6}$$

does not exceed L:

$$G_d(x,y)=\sum_{d'=1}^{d_{\text{in}}}\left(\sum_{(x',y')\in D:\ d_\infty((x,y),(x',y'))\le L}k_d(x-x',y-y',d')\cdot F_{d'}(x',y')\right), \tag{7}$$

where we denoted

$$D\stackrel{\text{def}}{=}(\mathbb{Z}\cap[\underline{X},\overline{X}])\times(\mathbb{Z}\cap[\underline{Y},\overline{Y}]). \tag{8}$$

We use, in this formula, the bounded subset D of the "grid" $\mathbb{Z}\times\mathbb{Z}$ and not the whole set $\widetilde{S}\stackrel{\text{def}}{=}\mathbb{Z}\times\mathbb{Z}$ only matters at the border of the domain D. So, to simplify our formulas, we can follow the usual tradition (see, e.g., [3]) and simply use the whole set $\widetilde{S}=\mathbb{Z}\times\mathbb{Z}$ instead of the bounded set D:

$$G_d(x,y)=\sum_{d'=1}^{d_{\text{in}}}\left(\sum_{(x',y')\in\widetilde{S}:\ d_\infty((x,y),(x',y'))\le L}k_d(x-x',y-y',d')\cdot F_{d'}(x',y')\right). \tag{9}$$

Comment. Note that the set $\widetilde{S}$ is potentially infinite. What makes the set of all the points (x',y')—which affects the values $G_d(x,y)$—finite is the restriction $d_\infty((x,y),(x',y'))\le L$, whose meaning is that such points (x',y') should belong to the corresponding neighborhood of the point (x,y).

2.2. Case of Dilated Convolution

The dilated convolution can be described in a similar way. Namely, we can describe the Formula (4) as

$$G_d(x,y) = \sum_{d'=1}^{d_{\text{in}}} \left(\sum_{(x',y') \in S_\ell(x,y):\ d_\infty((x,y),(x',y')) \le L} k_d(x-x', y-y', d') \cdot F_{d'}(x',y') \right), \qquad (10)$$

where $S_\ell(x,y)$ denotes the set of all the points (x',y') for which both differences $x-x'$ and $y-y'$ are divisible by ℓ:

$$S_\ell(x,y) \stackrel{\text{def}}{=} \{(x',y') : x' \equiv x \bmod \ell,\ y' \equiv y \bmod \ell\}. \qquad (11)$$

Note that, in this representation of dilated convolution, while we have several different sets $S_\ell(x,y)$ for different points (x,y), there is only one filter $k_d(x-x', y-y', d')$, namely the same filter that was used in the original representation (4). So, in this new representation, we have exactly as many parameters as before.

The main difference between this formula and the Formula (9) is that, in contrast to the usual convolution (9), where the same set $\widetilde{S} = \mathbb{Z} \times \mathbb{Z}$ could be used for all the points (x,y), here, in general, we may need different sets $S_\ell(x,y)$ for different points (x,y).

For example, if $\ell = 2$, then we need four such sets:

- for points (x,y) for which both x and y are even, the Formula (10) holds for

$$S_2(0,0) = S_2(0,2) = \ldots = S_{0,0}^{(\ell=2)} \stackrel{\text{def}}{=} \{(x,y) \in \mathbb{Z} \times \mathbb{Z} : x \text{ and } y \text{ are even}\}; \qquad (12)$$

- for points (x,y) for which x is even but y is odd, the Formula (10) holds for

$$S_2(0,1) = S_2(0,3) = \ldots = S_{0,1}^{(\ell=2)} \stackrel{\text{def}}{=} \{(x,y) \in \mathbb{Z} \times \mathbb{Z} : x \text{ is even and } y \text{ is odd}\}; \qquad (13)$$

- for points (x,y), for which x is odd but y is even, the Formula (10) holds for

$$S_2(1,0) = S_2(1,2) = \ldots = S_{1,0}^{(\ell=2)} \stackrel{\text{def}}{=} \{(x,y) \in \mathbb{Z} \times \mathbb{Z} : x \text{ is odd and } y \text{ is even}\}; \qquad (14)$$

- finally, for points (x,y) for which x and y are both odd, the Formula (10) holds for

$$S_2(0,1) = S_2(0,3) = \ldots = S_{1,1}^{(\ell=2)} \stackrel{\text{def}}{=} \{(x,y) \in \mathbb{Z} \times \mathbb{Z} : x \text{ and } y \text{ are odd}\}. \qquad (15)$$

In this case, instead of the single set $S_1(x,y) = \widetilde{S}$ (as in the case of the traditional convolution), we have a set of such sets

$$\mathcal{F} = \left\{ S_{0,0}^{(\ell=2)}, S_{0,1}^{(\ell=2)}, S_{1,0}^{(\ell=2)}, S_{1,1}^{(\ell=2)} \right\}. \qquad (16)$$

To avoid confusion, we will call subsets of the original "grid" $\mathbb{Z} \times \mathbb{Z}$ *sets*, while the set of such sets will be called a *family*. In these terms, the Formula (10) can be described as follows:

$$G_d(x,y) = \sum_{d'=1}^{d_{\text{in}}} \left(\sum_{(x',y') \in S_{\mathcal{F}}(x,y):\ d_\infty((x,y),(x',y')) \le L} k_d(x-x', y-y', d') \cdot F_{d'}(x',y') \right), \qquad (17)$$

where $S_{\mathcal{F}}(x,y)$ denotes the set $S \in \mathcal{F}$ from the family $\mathcal{F}$ that contains the point (x,y):

$$(x,y) \in S_{\mathcal{F}}(x,y) \text{ and } S_{\mathcal{F}}(x,y) \in \mathcal{F}. \qquad (18)$$

In this representation, all four sets S from the family $\mathcal{F}$ are infinite—just like the set $\widetilde{S}$ corresponding to the traditional convolution is infinite. Similarly to the traditional convolution, what makes the set of all the points (x', y')—which affects the values $G_r(x, y)$—finite is the restriction $d_\infty((x, y), (x', y')) \le L$, whose meaning is that such points (x', y') should belong to the corresponding neighborhood of the point (x, y).

Figure 6 describes which of the four sets $S \in \mathcal{F}$ corresponds to each point (x, y) from the "grid" $\mathbb{Z} \times \mathbb{Z}$:

$$\begin{array}{ccccc}
\ldots & \ldots & \ldots & \ldots & \ldots \\
\ldots & S^{(\ell=2)}_{1,1} & S^{(\ell=2)}_{0,1} & S^{(\ell=2)}_{1,1} & \ldots \\
\ldots & S^{(\ell=2)}_{1,0} & S^{(\ell=2)}_{0,0} & S^{(\ell=2)}_{1,0} & \ldots \\
\ldots & S^{(\ell=2)}_{1,1} & S^{(\ell=2)}_{0,1} & S^{(\ell=2)}_{1,1} & \ldots \\
\ldots & \ldots & \ldots & \ldots & \ldots
\end{array}$$

Figure 6. Sets $S_{\mathcal{F}}(x, y)$ corresponding to different points (x, y)—for filters presented in Figure 3.

For $\ell = 3$, we can get a similar reformulation, with the family

$$\mathcal{F} = \left\{ S^{(\ell=3)}_{0,0}, S^{(\ell=3)}_{0,1}, S^{(\ell=3)}_{0,2}, S^{(\ell=3)}_{1,0}, S^{(\ell=3)}_{1,1}, S^{(\ell=3)}_{1,2}, S^{(\ell=3)}_{2,0}, S^{(\ell=3)}_{2,1}, S^{(\ell=3)}_{2,2} \right\}, \quad (19)$$

where $S^{(\ell=3)}_{i,j}$ is the set of all the pairs $(x, y) \in \mathbb{Z} \times \mathbb{Z}$, in which both differences $x - i$ and $y - j$ are divisible by 3.

In general, for an arbitrary point (x, y), we should use the set $S_{\mathcal{F}} = S^{(\ell=2)}_{x \bmod \ell,\, y \bmod \ell}$.

2.3. Other Cases

Such a representation is possible not only for dilated convolution. For example, the above case when we allow arbitrary value i and require the value j to be even can be described in a similar way, with

$$\mathcal{F} = \{S_0, S_1\}, \quad (20)$$

where:

- for points (x, y) for which y is even, we take

$$S_{\mathcal{F}}(0, 0) = S_{\mathcal{F}}(1, 0) = \ldots = S_0 \overset{\text{def}}{=} \{(x, y) \in \mathbb{Z} \times \mathbb{Z} : y \text{ is even}\}, \quad (21)$$

- and for points (x, y) for which y is odd, we take

$$S_{\mathcal{F}}(0, 1) = S_{\mathcal{F}}(1, 1) = \ldots = S_1 \overset{\text{def}}{=} \{(x, y) \in \mathbb{Z} \times \mathbb{Z} : y \text{ is odd}\}. \quad (22)$$

We can also have families which have an infinite number of sets; an example of such a family will be given below.

We can also, in principle, consider the situations when we do not require that the coefficients $k_d(x, x', y, y', d')$ depend only on the differences $x - x'$ and $y - y'$. Thus, we arrive at the following general description.

2.4. General Case

In the general case, we get the following situation:

- we have a family $\mathcal{F}$ of subsets of the "grid" $\mathbb{Z} \times \mathbb{Z}$;
- the values $G_d(x, y)$ of the layer's output signal at a point (x, y) are determined by the formula

$$G_d(x, y) = \sum_{d'=1}^{d_{\text{in}}} \left(\sum_{(x', y') \in S_{\mathcal{F}}(x, y):\ d_\infty((x, y), (x', y')) \le L} K_d(x, x', y, y', d') \cdot F_{d'}(x', y') \right), \quad (23)$$

for some values $K_d(x, x', y, y', d')$, where $S_{\mathcal{F}}(x, y)$ denotes the set $S \in \mathcal{F}$ from the family $\mathcal{F}$ that contains the point (x, y).

For the Formula (23) to uniquely determine the values $G_d(x, y)$, we need to make sure that the set $S_{\mathcal{F}}(x, y)$ is uniquely determined by the point (x, y), i.e., that for each point (x, y), the family $\mathcal{F}$ contains one, and only one, set S that contains this point. In other words:

- different sets from the family $\mathcal{F}$ must be disjoint, and
- the union of all the sets $S \in \mathcal{F}$ must coincide with the whole "grid" $\mathbb{Z} \times \mathbb{Z}$.

In mathematical terms, the family $\mathcal{F}$ must form a partition of the "grid" $\mathbb{Z} \times \mathbb{Z}$.

Comment. To avoid possible confusion, it is worth mentioning that while *different sets S* from the family $\mathcal{F}$ are disjoint, this does not preclude the possibility that sets $S_{\mathcal{F}}(x, y)$ and $S_{\mathcal{F}}(x', y')$ corresponding to *different points* $(x, y) \neq (x', y')$ can be identical. For example, in the description of the traditional convolution, the family $\mathcal{F}$ consists of only one set $\mathcal{F} = \left\{\widetilde{S}\right\}$. In this case, for all points (x, y) and (x', y'), we have $S_{\mathcal{F}}(x, y) = S_{\mathcal{F}}(x', y') = \widetilde{S}$.

In terms of sets corresponding to different points, disjointness means that *if* the sets $S_{\mathcal{F}}(x, y)$ and $S_{\mathcal{F}}(x', y')$ are different, *then* these sets must be disjoint: $S_{\mathcal{F}}(x, y) \cap S_{\mathcal{F}}(x', y') = \emptyset$.

2.5. We Do Not a Priori Require Shift-Invariance

Please note that we do not a priori require that the sets $S_{\mathcal{F}}(x, y)$ and $S_{\mathcal{F}}(x_0, y_0)$ corresponding to two different points (x, y) and (x_0, y_0) should be obtained from each other by shift—this property is known as *shift invariance* and is satisfied both for the usual convolution and for the dilated convolution.

It should be emphasized, however, that we will show that this shift-invariance property holds for the optimal arrangement.

2.6. Let Us Avoid the Degenerate Case

From a purely mathematical viewpoint, we can have a partition of the "grid" $\mathbb{Z} \times \mathbb{Z}$ into one-point sets $\{(x, y)\}$. This is an example when the family $\mathcal{F}$ has infinitely many subsets.

In this case, no matter what value L we choose, the Formula (23) implies that the values $G_d(x, y)$ of the layer's output signal at a point (x, y) are determined only by the values $F_{d'}(x, y)$ of the layer's input at this same point. This is equivalent to using a convolution with $L = 0$; such a convolution is known as the 1-by-1 convolution.

While such a convolution is often useful, in this case, for each point (x, y), there is only one point $(x', y') = (x, y)$, so it is not possible to select only some of the points (x', y')—which is the whole idea of dilation. Since, in this paper, we study dilation, we will therefore avoid this 1-by-1 situation and additionally require that at least one set from the family $\mathcal{F}$ should contain more than one element.

2.7. What We Plan to Do

We will consider all possible families $\mathcal{F}$ that form a partition of the "grid" $\mathbb{Z} \times \mathbb{Z}$, and we will show that for all optimality criteria that satisfy some reasonable conditions, the optimal family is either the family of sets corresponding to the dilated convolution—or a natural modification of this family.

Let us describe what we mean by optimality criteria.

2.8. What Does "Optimal" Mean?

In our case, we select between different families of sets $\mathcal{F}, \mathcal{F}', \ldots$. In general, we select between alternatives a, b, etc. Out of all possible alternatives, we want to select an optimal one. What does "optimal" mean?

In many cases, "optimal" is easy to describe:

- we have an objective function $f(a)$ that assigns a numerical value to each alternative a—e.g., the average approximation error of the numerical method a for solving a system of differential equations, and

- optimal means that we select an alternative for which the value of this objective function is the smallest possible (or, for some objective functions, the largest possible).

However, this is not the only possible way to describe optimality.

For example, if we are minimizing the average approximation error, and there are several different numerical methods with the exact same smallest value of average approximation error, then we can use this non-uniqueness to select, e.g., the method with the shortest average computation time. In this case, we have, in effect, a more complex preference relation between alternatives than in the case when decision is made based solely on the value of the objective function. Specifically, in this case, an alternative b is better than the alternative a—we will denote it by $a < b$—if:

- either we have $f(b) < f(a)$,
- or we have $f(a) = f(b)$ and $g(b) < g(a)$.

If this still leaves several alternatives which are equally good, then we can optimize something else, and thus, have an even more complex optimality criterion.

In general, having an optimality criterion means that we are able to compare pairs of alternatives—at least some such pairs—and conclude that:

- for some of these pairs, we have $a < b$,
- for some of these pairs, we have $b < a$, and
- for some others pairs, we conclude that alternatives a and b are, from our viewpoint, of equal value; we will denote this by $a \sim b$.

Of course, these relations must satisfy some reasonable properties. For example, if b is better than a, and c is better than b, then c should be better than a; in mathematical terms, the relation $<$ must be transitive.

We must have some alternative which is better than or equivalent to all others—otherwise, the optimization problem has no solutions. It also makes sense to require that there is only one such optimal alternative—indeed, as we have mentioned, if there are several equally good optimal alternatives, this means that the original optimality criterion is not final, that we can use this non-uniqueness to optimize something else, i.e., in effect, to modify the original criterion into a final (or at least "more final") one.

2.9. Invariance

There is an additional natural requirement for possible optimality criteria, which is related to the fact that the original "grid" $\mathbb{Z} \times \mathbb{Z}$ has lots of *symmetries*, i.e., transformations that transform this "grid" into itself.

For example, if we change the starting point of the coordinate system to a new point (x_0, y_0), then a point that originally had coordinates (x, y) now has coordinates $(x - x_0, y - y_0)$. It makes sense to require that the relative quality of two different families $\mathcal{F}$ and $\mathcal{F}'$ will not change if we simply change the starting point.

Similarly, we can change the direction of the x-axis, then a point (x, y) becomes $(-x, y)$. If we change the direction of the y-axis, we obtain a transformation $(x, y) \to (x, -y)$. Finally, we can rename the coordinates: what was x will become y and vice versa; this corresponds to the transformation $(x, y) \to (y, x)$. Such transformations should also not affect the relative quality of different families.

Please note that we are not requiring that the *family* $\mathcal{F}$ of sets be shift-invariant, what we require is that the *optimality criterion* is shift-invariant.

Let us explain why, in our opinion, it makes sense to require that the optimality criterion is shift-invariance—as well as having other invariance properties. Indeed, let us consider any usual optimality criterion such as accuracy of classification, robustness to noise, etc. What each criterion means is, e.g., the overall classification accuracy over the set $\mathcal{S}$ of all possible cat and not-a-cat images $I \in \mathcal{S}$. We want this method to correctly classify images into cats and not-cats, whether these images are centered or somewhat shifted. Thus, to adequately compare different methods, we should test these methods on a set $\mathcal{S}$ of images that includes both original and shifted images.

Here:

- if we shift each image I from the set $\mathcal{S}$ by the same shift (x_0, y_0), i.e., replace each image $I \in S$ by a shifted image $I' = T_{x_0,y_0}(I)$ for which $I'(x,y) = I(x - x_0, y - y_0)$,
- then, we should get, in effect, the exact same set of images:

$$T_{x_0,y_0}(\mathcal{S}) \stackrel{\text{def}}{=} \{T(x_0,y_0)(I) : I \in \mathcal{S}\} \approx \mathcal{S}. \tag{24}$$

The only difference between these two sets of images may be the few images where the cat is right at the image's boundary; in this paper, we will ignore this difference—just like we ignored the bounded-ness in the previous text. In this ignoring-bounds approximation, we conclude that

$$T_{x_0,y_0}(\mathcal{S}) = \{T(x_0,y_0)(I) : I \in \mathcal{S}\} = \mathcal{S}. \tag{25}$$

How does a shift of the original image affect the input signals to the following convolution layers? In between the very first input layer and the following convolution layers, we may have (and usually do have) layers that perform the "compression" of the (x, y) part—i.e., that transform:

- values corresponding to several points (x, y)
- into values corresponding to a single new point (x', y').

In general, the (x, y)-shift of the original data corresponds to a shift of the transformed data—but by smaller shift values. For example, if data corresponding to each new (x, y)-point come from data from four different "pre-compression" points, then the shift by (x_0, y_0) in the pre-(x, y)-compression layer corresponds to a shift of the convolution layer input by $(x_0/2, y_0/2)$.

Since the set of input images should not change if we apply a shift, we can conclude that for each convolution layer, the set of the corresponding inputs to this layer should also not change if we shift all these inputs, i.e., if we replace each input $F_d(x, y)$ with a shifted input

$$F'_d(x,y) \stackrel{\text{def}}{=} F_d(x - x_0, y - y_0) \tag{26}$$

for some shift (x_0, y_0).

The set of inputs on which we compare different methods does not change when we apply a shift. So, if one method was better when we processed original inputs, it should still be better if we process shifted inputs—since the resulting set of inputs is the same. In other words, the quality (e.g., accuracy) $Q_{\mathcal{F}}(\mathcal{S})$ of a method corresponding to the family $\mathcal{F}$, when gauged by the set of inputs corresponding to original images, should be the same as this method's quality $Q_{\mathcal{F}}(T_{x_0,y_0}(\mathcal{S}))$ on the set

$$T_{x_0,y_0}(\mathcal{S}) = \{T_{x_0,y_0}(F_d) : F_d \in \mathcal{S}\} \tag{27}$$

of all the inputs obtained from the original set $\mathcal{S}$ by this shift—since these two sets of inputs are, in effect, the same set: $T_{x_0,y_0}(\mathcal{S}) = \mathcal{S}$. Thus, $Q_{\mathcal{F}}(T_{x_0,y_0}(\mathcal{S})) = Q_{\mathcal{F}}(\mathcal{S})$.

However, as one can see, shifting all the inputs is equivalent to shifting all the sets from the family $\mathcal{F}$. Indeed, if we apply the Formula (23) to the shifted layer's input $F'_d(x,y) \stackrel{\text{def}}{=} F_d(x - x_0, y - y_0)$, we get

$$G_d(x,y) =$$

$$\sum_{d'=1}^{d_{\text{in}}} \left(\sum_{(x',y') \in S_{\mathcal{F}}(x,y):\ d_\infty((x,y),(x',y')) \le L} K_d(x,x',y,y',d') \cdot F_{d'}(x' - x_0, y' - y_0) \right), \tag{28}$$

i.e., in terms of the shifted coordinates $X \stackrel{\text{def}}{=} x - x_0$ and $Y \stackrel{\text{def}}{=} y - x_0$ for which $x = X + x_0$ and $y = Y + y_0$, we get—taking into account that the distance d_∞ does not change with shift—that:

$$G_d(X,Y) = \sum_{d'=1}^{d_{\text{in}}} \left(\sum_{C':\, d_\infty((X,Y),(X',Y'))\le L} K'_d(X,X',Y,Y',d') \cdot F_{d'}(x'-x_0, y'-y_0) \right), \tag{29}$$

where we denoted

$$K'_d(X,X',Y,Y',d') \stackrel{\text{def}}{=} K_d(X+x_0, X'+x_0, Y+y_0, Y'+y_0), \tag{30}$$

and where C' denotes the condition $(X'+x_0, Y'+y_0) \in S_{\mathcal{F}}(X+x_0, Y+y_0)$.

In terms of the family $\mathcal{F}$, the main difference between the Formulas (23) and (29) is that instead of the condition $(x',y') \in S_{\mathcal{F}}(x,y)$, we now have a new condition

$$C' \Leftrightarrow (X'+x_0, Y'+y_0) \in S_{\mathcal{F}}(X+x_0, Y+y_0), \tag{31}$$

i.e., equivalently, $(X',Y') \in S_{\mathcal{F}}(X+x_0, Y+y_0) - (x_0,y_0)$. It is easy to check that this new condition is equivalent to $(Y',Y') \in S_{\mathcal{F}'}(X,Y)$, where the new family $\mathcal{F}'$ is obtained by shifting sets from the original family $\mathcal{F}$.

So:

- the relative quality of two families does not change if we shift all the layer's inputs;
- however, shifting all the layer's inputs is equivalent to shifting all the sets from the family $\mathcal{F}$.

Thus, the relative quality of two families does not change if we shift both families. In other words, a reasonable optimality criterion—which describes which family is better—should be invariant with respect to shifts.

Similarly, we can argue that a reasonable optimality criterion should not change if we rename x- and y-axes, etc.

Now, we are ready for the precise formulation of the problem.

3. Definitions and the Main Result

Definition 1. *By a* family, *we mean a family of non-empty subsets of the "grid" $\mathbb{Z} \times \mathbb{Z}$, a family in which:*

- *all sets from this family are disjoint, and*
- *at least one set from this family has more than one element.*

Terminological comment. To avoid possible misunderstandings, let us emphasize that here, we consider several levels of sets, and to avoid confusion, we use different terms for sets from different levels:

- first, we consider *points* $(x,y) \in \mathbb{Z} \times \mathbb{Z}$;
- second, we consider *sets* of points $S \subseteq \mathbb{Z} \times \mathbb{Z}$; we call them simply sets;
- third, we consider sets of sets of points $\mathcal{F} = \{S, S', \ldots\}$; we call them *families;*
- finally, we consider the set of all possible families $\{\mathcal{F}, \mathcal{F}', \ldots\}$; we call this a *class.*

Comment about the requirements. In the previous text, we argued that for each family $\mathcal{F}$, the union of all its sets $\cup\{S : S \in \mathcal{F}\}$ should coincide with the whole "grid" $\mathbb{Z} \times \mathbb{Z}$. However, in our definition of an alternative, we did not impose this requirement. We omitted this requirement to make our result stronger—since, as we see from the following Proposition, this requirement actually follows from all the other requirements.

Definition 2. *By an* optimality criterion, *we mean a pair of relations $(<, \sim)$ on the class of all possible families that satisfy the following conditions:*

- *if $\mathcal{F} < \mathcal{F}'$ and $\mathcal{F}' < \mathcal{F}''$, then $\mathcal{F} < \mathcal{F}''$;*

- *if $\mathcal{F} < \mathcal{F}'$ and $\mathcal{F}' \sim \mathcal{F}''$, then $\mathcal{F} < \mathcal{F}''$;*
- *if $\mathcal{F} \sim \mathcal{F}'$ and $\mathcal{F}' < \mathcal{F}''$, then $\mathcal{F} < \mathcal{F}''$;*
- *if $\mathcal{F} \sim \mathcal{F}'$ and $\mathcal{F}' \sim \mathcal{F}''$, then $\mathcal{F}' \sim \mathcal{F}''$;*
- *we have $\mathcal{F} \sim \mathcal{F}$ for all $\mathcal{F}$; and*
- *if $\mathcal{F} < \mathcal{F}'$, then we cannot have $\mathcal{F} \sim \mathcal{F}'$.*

Comment. The pair of relations $(<, \sim)$ between families of subsets forms what is called a *pre-order* or *quasi-order*. This notion is more general than partial order, since, in contrast to the definition of the partial order, we do not require that if $a \le b$ and $b \le a$, then $a = b$: in principle, we can have $a \sim b$ for some $a \neq b$.

Definition 3. *We say that a family $\mathcal{F}$ is* optimal *with respect to the optimality criterion $(<, \sim)$, if for every other family $\mathcal{F}'$, we have either $\mathcal{F}' < \mathcal{F}$ or $\mathcal{F}' \sim \mathcal{F}$.*

Definition 4. *We say that the optimality criterion is* final *if there exists exactly one family which is optimal with respect to this criterion.*

Definition 5. *By a* transformation $T : \mathbb{Z} \times \mathbb{Z}$, *we mean one of the following transformations: $T_{x_0,y_0}(x, y) = (x - x_0, y - y_0)$, $T_{-+}(x, y) = (-x, y)$, $T_{+-}(x, y) = (x, -y)$, and $T_{\leftrightarrow}(x, y) = (y, x)$.*

Definition 6. *For each family $\mathcal{F}$ and for each transformation T, by the* result $T(\mathcal{F})$ *of applying the transformation T to the family $\mathcal{F}$, we mean the family $T(\mathcal{F}) = \{T(S) : S \in \mathcal{F}\}$, where, for any set S, $T(S) \stackrel{\text{def}}{=} \{T(x, y) : (x, y) \in S\}$.*

Definition 7. *We say that the optimality criterion is* invariant *if for all transformations T, $\mathcal{F} < \mathcal{F}'$ implies that $T(\mathcal{F}) < T(\mathcal{F}')$, and $\mathcal{F} \sim \mathcal{F}'$ implies that $T(\mathcal{F}) \sim T(\mathcal{F}')$.*

Proposition 1. *For every final invariant optimality criterion, the optimal family is equal, for some integer $\ell \ge 1$, to one of the following two families:*

- *the family of all the sets $S_{\ell,x_0,y_0} \stackrel{\text{def}}{=} \{(x_0 + \ell \cdot n_x, y_0 + \ell \cdot n_y) : n_x, n_y \in \mathbb{Z}\}$ corresponding to all possible pairs of integers (x_0, y_0) for which $0 \le x_0, y_0 < \ell$;*
- *the family of all the sets*

$$S'_{\ell,x_0,y_0} \stackrel{\text{def}}{=} \{(x_0 + \ell \cdot n_x, y_0 + \ell \cdot n_y) : n_x, n_y \in \mathbb{Z} \text{ and } n_x + n_y \text{ is even}\}$$

  corresponding to all possible pairs of integers (x_0, y_0) for which $0 \le x_0, y_0 < \ell$.

  Comments.

- This proposition takes care of all invariant (and final) optimality criteria. Thus, it should work for all usual criteria based on misclassification rate, time of calculation, used memory, or any others used in neural networks: indeed, if one method is better than another for images in general, it should remain to be better if we simply shift all the images or turn all the images upside down. Images can come as they are, they can come upside down, they can come shifted, etc. If, for some averaging criterion, one method works better for all possible images, but another method works better for all upside-down versions of these images—which is, in effect, the same class of possible images—then, from the common sense viewpoint, this would mean that something is not right with this criterion.
- The first possibly optimal case corresponds to dilated convolution. In the second possibly optimal case, the optimal family contains similar but somewhat different sets; an example of such a set is given in Figure 7.

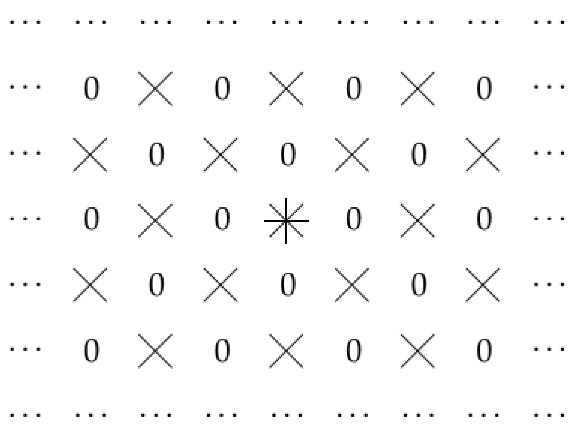

Figure 7. A set from the second possibly optimal family.

Thus, this result explains the effectiveness of dilated convolution—and also provides us with a new alternative worth trying.

- The following proof is similar to several proofs presented in [5,6].

Proof. 1°. Since the optimality criterion is final, there exists exactly one optimal family $\mathcal{F}_{\rm opt}$. Let us first prove that this family is itself invariant, i.e., that $T(\mathcal{F}_{\rm opt}) = \mathcal{F}_{\rm opt}$ for all transformations T.

Indeed, the fact that the family $\mathcal{F}_{\rm opt}$ is optimal means that for every family $\mathcal{F}$, we have $\mathcal{F} < \mathcal{F}_{\rm opt}$ or $\mathcal{F} \sim \mathcal{F}_{\rm opt}$. Since this is true for every family $\mathcal{F}$, it is also true for every family $T^{-1}(\mathcal{F})$, where T^{-1} denotes inverse transformation (i.e., a transformation for which $T(T^{-1}(x,y)) = (x,y)$). Thus, for every family $\mathcal{F}$, we have either $T^{-1}(\mathcal{F}) < \mathcal{F}_{\rm opt}$ or $T^{-1}(\mathcal{F}) \sim \mathcal{F}_{\rm opt}$. Due to invariance, we have $\mathcal{F} = T(T^{-1}(\mathcal{F})) < T(\mathcal{F}_{\rm opt})$ or $\mathcal{F} \sim T(\mathcal{F}_{\rm opt})$. By definition of optimality, this means that the alternative $T(\mathcal{F}_{\rm opt})$ is also optimal. However, since the optimality criterion is final, there exists exactly one optimal family, so $T(\mathcal{F}_{\rm opt}) = \mathcal{F}_{\rm opt}$.

The statement is proven.

2°. Let us now prove that the optimal family contains a set S' that, in its turn, contains the point $(0,0)$ and some point $(x,y) \neq (0,0)$.

Indeed, by definition of a family, every family—including the optimal family—contains at least one set S that has at least two points. Let S be any such set from the optimal family, and let (x_0,y_0) be any of its points. Then, due to Part 1 of this proof, the set $S' \stackrel{\rm def}{=} T_{x_0,y_0}(S)$ also belongs to the optimal family, and this set contains the point

$$T_{x_0,y_0}(x_0,y_0) = (x_0 - x_0, y_0 - y_0) = (0,0).$$

Since the set S had at least two different points, the set $S' = T_{x_0,y_0}(S)$ also contains at least two different points. Thus, the set S' must contain a point (x,y) which is different from $(0,0)$.

The statement is proven.

3°. In the following text, by S', we will mean a set from the optimal family $\mathcal{F}_{\rm opt}$ whose existence is proven in Part 2 of this proof: namely, a set that contains the point $(0,0)$ and a point $(x,y) \neq (0,0)$.

4°. Let us prove that if the set S' contains a point (x,y), then this set also contains the points $(x,-y)$, $(-x,y)$, and (y,x).

Indeed, due to Part 1 of this proof, with the set S' the optimal family $\mathcal{F}_{\rm opt}$ also contains the set $T_{+-}(S')$. This set contains the point $T_{+-}(0,0) = (0,0)$. Thus, the sets S' and $T_{+-}(S')$ have a common element $(0,0)$. Since different sets from the optimal family must be disjoint, it follows that the sets S' and $T_{+-}(S')$ must coincide. The set $T_{+-}(S')$ contains the points $(x,-y)$ for each point $(x,y) \in S$. Since $T_{+-}(S') = S'$, this implies that for each point $(x,y) \in S'$, we have $(x,-y) \in T_{+-}(S') = S'$.

Similarly, we can prove that $(-x,y) \in S'$ and $(y,x) \in S'$. The statement is proven.

5°. By combining the two conclusions of Part 4—that $(x,-y) \in S'$ and that therefore $T_{-+}(x,-y) = (-x,-y) \in S'$, we conclude that for every point $(x,y) \in S'$, the point

$$-(x,y) \overset{\text{def}}{=} (-x,-y)$$

is also contained in the set S'.

6°. Let us prove that if the set S' contains two points (x_1,y_1) and (x_2,y_2), then it also contains the point

$$(x_1,y_1) + (x_2,y_2) \overset{\text{def}}{=} (x_1 + x_2, y_1 + y_2).$$

Indeed, due to Part 1 of this proof, the set $T_{-x_2,-y_2}(S')$ also belongs to the optimal family. This set shares an element

$$T_{-x_2,-y_2}(0,0) = (0 - (-x_2), 0 - (-y_2)) = (x_2, y_2)$$

with the original set S'. Thus, the set $T_{-x_2,-y_2}(S')$ must coincide with the set S'. Due to the fact that $(x_1,y_1) \in S'$, the element

$$T_{-x_2,-y_2}(x_1,y_1) = (x_1 - (-x_2), y_1 - (-y_2)) = (x_1 + x_2, y_1 + y_2)$$

belongs to the set $T_{x_1,y_1}(S') = S'$. The statement is proven.

7°. Let us prove that if the set S' contains a point (x,y), then, for each integer c, this set also contains the point

$$c \cdot (x,y) = (c \cdot x, c \cdot y).$$

Indeed, if c is positive, this follows from the fact that

$$(c \cdot x, c \cdot y) = (x,y) + \ldots + (x,y) \ (c \text{ times}).$$

When c is negative, then we first use Part 5 and conclude that $(-x,-y) \in S'$, and then conclude that the point $(|c| \cdot (-x), |c| \cdot (-y)) = (c \cdot x, c \cdot y)$ is in the set S'.

8°. Let us prove that if the set S' contains points (x_1,y_1), …, (x_n,y_n), then for all integers $c_1,\ldots,c_n$, it also contains their linear combination

$$c_1 \cdot (x_1,y_1) + \ldots + c_n \cdot (x_n,y_n) = (c_1 \cdot x_1 + \ldots + c_n \cdot x_n, c_1 \cdot y_1 + \ldots + c_n \cdot y_n).$$

Indeed, this follows from Parts 6 and 7.

9°. The set S' contains some points which are different from $(0,0)$, i.e., points for which at least one of the integer coordinates is non-zero. According to Parts 4 and 5, we can change the signs of both x and y coordinates and still get points from S'. Thus, we can always consider points with non-negative coordinates.

Let d denote the greatest common divisor of all positive values of the coordinates of points from S'.

If a value x appears as an x-coordinate of some point $(x,y) \in S'$, then, due to Part 4, we have $(x,-y) \in S'$ and thus, due to Part 5,

$$(x,y) + (x,-y) = (2x, 0) \in S'.$$

Similarly, if a value y appears as a y-coordinate of some point $(x,y) \in S'$, then we get $(0,2y) \in S'$ and thus, due to Part 3, $(2y,0) \in S'$.

It is known that a common divisor d of the values $v_1,\ldots,v_n$ can be represented as a linear combination of these values:

$$d = c_1 \cdot v_1 + \ldots + c_n \cdot v_n.$$

For each value v_i, we have $(2v_i, 0) \in S'$, thus, for

$$2d = c_1 \cdot (2v_1) + \ldots + c_n \cdot (2v_n),$$

by Part 8, we get $(2d, 0) \in S'$. Due to Part 4, we thus have $(0, 2d) \in S'$, and due to Parts 6 and 7, all points $(n_x \cdot (2d), n_y \cdot (2d))$ for integers n_x and n_y also belong to the set S'.

If S' has no other points, then for the set containing $(0, 0)$, we indeed conclude that this set is the same as what we described for dilated convolution, with $\ell = 2d$.

10°. What if there are other points in the set S'? Since d is the greatest common divisor of all the coordinate values, each of these points has the form $(c_x \cdot d, c_y \cdot d)$ for some integers c_x and c_y. Since this point is not of the form $(n_x \cdot (2d), n_y \cdot (2d))$, this means that either c_x, or c_y is an odd number—or both.

Let us first consider the case when exactly one of the values c_x and c_y is odd. Without losing generality, let us assume that c_x is odd, so $c_x = 2n_x + 1$ and $c_y = 2n_y$ for some integers n_x and n_y. Due to Part 9, we have $(2n_x \cdot d, 2n_y \cdot d) \in S'$, so the difference

$$((2n_x + 1) \cdot d, 2n_y \cdot d) - (2n_x \cdot d, 2n_y \cdot d) = (d, 0)$$

also belongs to the set S'. Thus, similarly to Part 9, we can conclude that for every two integers c_x and c_y, we have $(c_x \cdot d, c_y \cdot d) \in S'$. So, in this case, S' coincides, for $\ell = d$, with the set corresponding to dilated convolution.

The only remaining case is when not all points $(c_x \cdot d, c_y \cdot d)$ belong to the set S'. This means that for some such point, both values c_x and c_y are odd: $c_x = 2n_x + 1$ and $c_y = 2n_y + 1$ for some integers n_x and n_y. Due to Part 9, we have $(2n_x \cdot d, 2n_y \cdot d) \in S'$, so the difference

$$((2n_x + 1) \cdot d, (2n_y + 1) \cdot d) - (2n_x \cdot d, 2n_y \cdot d) = (d, d)$$

also belongs to the set S'.

Since, due to Part 9, we have $(2n_x \cdot d, 2n_y \cdot d) \in S'$ for all n_x and n_y, we conclude, by using Part 5, that $((2n_x + 1) \cdot d, (2n_y + 1) \cdot d) \in S'$. So, all pairs for which both coordinates are odd multiples of d are in S'. Thus, we get the new case described in the Proposition.

11°. The previous results were about the sets containing the point $(0, 0)$.

For all other sets S containing some other point (x_0, y_0), we get the same result if we take into account that the optimal family is invariant, and thus, with the set S, the optimal family also contains the set $T_{x_0,y_0}(S)$ that contains $(0, 0)$ and is, thus, equal either to the family corresponding to dilated convolution or to the new similar family.

The proposition is proven. □

4. Conclusions and Future Work

4.1. Conclusions

One of the efficient machine learning ideas is the idea of a convolutional neural network. Such networks use convolutional layers, in which the layer's output at each point is a combination of the layer's input corresponding to several neighboring points. A reasonable idea is to restrict ourselves to only some of the neighboring points. It turns out that out of all such restrictions, the best results are obtained when we only use neighboring points, for which the differences in both coordinates are divisible by some constant ℓ. Networks implementing such restrictions are known as dilated convolutional neural networks.

In this paper, we provide a theoretical explanation for this empirical conclusion.

4.2. Future Work

This paper describes a general abstract result: that for any optimality criterion that satisfies some reasonable properties, some dilated convolution is optimal. To be practically useful, it is desirable to analyze which dilated convolutions are optimal for different practical situations and for specific criteria uses in machine learning, such as misclassification rate, time of calculation, used memory, etc. (and the combination of these criteria). It is also desirable to analyze what size neighborhood we should choose for different practical situations and for different criteria.

Author Contributions: All three authors contributed equally to this paper: M.C. formulated the neural-related problem, V.K. provided a general optimization framework for solving such problems, J.C. came up with an idea of how to adjust this general framework for this particular problem, and jointly, all three of us transformed this idea into precise definitions and results. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Science Foundation grants 1623190 (A Model of Change for Preparing a New Generation for Professional Practice in Computer Science), and HRD-1834620 and HRD-2034030 (CAHSI Includes). It was also supported by the program of the development of the Scientific-Educational Mathematical Center of Volga Federal District No. 075-02-2020-1478.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors are greatly thankful to the anonymous referees for valuable suggestions.

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

References

1. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Leaning*; MIT Press: Cambridge, MA, USA, 2016.
2. Li, Y.; Zhang, X.; Chen, D. CSRNet: Dilated convolutional neural networks for understanding the highly congested scenes. In Proceedings of the 2018 Conference on Computer Vision and Pattern Recognition CVPR'2018, Salt Lake City, UT, USA, 18–22 June 2018; pp. 1091–1100.
3. Yu, F.; Koltun, V. Multi-scale context aggregation by dilated convolutions. In Proceedings of the 4th International Conference on Learning Representations ICLR'2016, San Juan, PR, USA, 2–4 May 2016.
4. Zhang, X.; Zou, Y.; Shi, W. Dilated convolution neural network with LeakyReLU for environmental sound classification. In Proceedings of the 2017 22nd International Conference on Digital Signal Processing DSP'2017, London, UK, 23–25 August 2017.
5. Nguyen, H.T.; Kreinovich, V. *Applications of Continuous Mathematics to Computer Science*; Kluwer: Dordrecht, The Netherlands, 1997.
6. Kreinovich, V.; Kosheleva, O. Optimization under uncertainty explains empirical success of deep learning heuristics. In *Black Box Optimization, Machine Learning and No-Free Lunch Theorems*; Pardalos, P., Rasskazova, V., Vrahatis, M.N., Eds.; Springer: Cham, Switzerland, 2021; pp. 195–220.