A Nonparametric Pooling Operator Capable of Texture Extraction

Vincent Vigneron, Hichem Maaref

A nonparametric pooling operator capable of texture extraction*

V. Vigneron[1 [0000−0001−5917−6041]] and H. Maaref[1 [0000−0002−1192−7333]]

Univ Evry, Université Paris-Saclay, IBISC EA 4526, Evry, France
{vincent.vigneron,hichem.maaref}@ univ-evry.fr

Abstract. Much of Convolutional neural networks (CNNs)'s profound success lies in translation invariance. The other part lies in the almost infinite ways of arranging the layers of the neural network to make decisions in particular in computer vision problems, taking into account the whole image. This work proposes an alternative way to extend the pooling function, we named rank-order pooling, capable of extracting texture descriptors from images. Efforts to improve pooling layers or replace-add their functionality to other CNN layers is still an active area of research despite already a quite long history of architecture. Rank-order clustering is non-parametric, independent of geometric layout or image region sizes, and can therefore better tolerate rotations. Many related metrics are available for rank aggregation. In this article we present the properties of some of these metrics, their concordance indices and how they contribute to the efficiency of this new pooling operator.

Keywords: Deep CNN, pooling function, rank aggregation, LBP, optimization, linear programming, rank-order, contour extraction, segmentation.

1 Introduction

A deep CNN stacks four different processing layers: convolution, pooling, ReLU and fully-connected [5]. CNNs architecture is augmented by multi-resolution (*pyramidal*) structures which come from the idea that the network needs to see different levels of resolutions to produce good results.

Placed between two convolutional layers, the pooling layer receives several input feature maps. Pooling [4] (a) reduces the number of parameters in the model (subsampling) and computations in the network while preserving their important characteristics (b) improves the efficiency of the network (c) prevents overtraining. Even though pooling layers do not have parameters, they affect the backpropagation (derivatives) calculation. Back-propagating through the max-pooling layer simply selects the maximum neuron from the previous

* This research was supported by the program *Cátedras Franco-Brasileiras no Estado de São Paulo*, an initiative of the French consulate and the state of São Paulo (Brazil). We thank our colleagues Prof. João M. T. Romano, Dr. Kenji Nose and Dr. Michele Costa, who provided insights that greatly assisted this work.

layer (on which the max-pooling was performed) and continues backpropagation only through it. The max function is locally linear for the activation that obtained the max, so its derivative is 1, and 0 for the activation which did not succeed. This is conceptually very similar to the differentiatiin of the activation function $\mathrm{ReLU}(x) = \max(0, x)$.

Suppose a layer H_ℓ comes on top of a layer $H_{\ell-1}$. Then the activation of the ith neuron of the layer H_ℓ is

$$H_{\ell i} = f(\sum_j w_{ij} H_{(\ell-1)i}), \tag{1}$$

where f is the activation function and $W = \{w_{ij}\}$ are the weights. The derivation of Eq. (1) by the chain-rule gives the gradient flows as follows

$$\mathrm{grad}(H_{(\ell-1)i}) = \sum_i \mathrm{grad}(H_{\ell i}) f' w_{ij}. \tag{2}$$

In the case of max-pooling, f =id for the max neuron and $f = 0$ for all other neurons, so $f' = 1$ for the max neuron of the previous layer and $f' = 0$ for all other neurons. Back-propagating through the max-pooling layer simply selects the max neuron from the previous layer (on which the max-pooling was done) and continues back-propagation only through that.

The max-pooling function downsamples the input representation (image, hidden layer output matrix, etc.) making it less sensitive to re-cropping, rotation, shifting, and other minor changes.

Weaknesses of pooling functions are well identified [21]: (a) they don't preserve all spatial information (b) the maximum chosen by the max-pooling in the pixel grid is not the true maximum (c) average pooling assumes a single mode with a single centroïd. How optimally take into account the characteristics of the input image grouped in the pooling operation ? Part of the answer lies in the work of Lazebnik's who demonstrated the importance of the spatial structure of close neighborhoods [8]: indeed, local spatial variations of image pixel intensities (also called textures) characterize an "organized area phenomenon" [11] which cannot be captured in pooling layers.

This paper proposes a new pooling operation, independent of the geometric arrangement or sizes of image regions, and can therefore better tolerate rotations [1].

Notations Throughout this paper small Latin letters $a, b, \ldots$ represent integers. Small bold letters $\mathbf{a}, \mathbf{b}$ are put for vectors and capital letters A, B for matrices or tensor depending of the context. The dot product between two vectors is denoted $< \mathbf{a}, \mathbf{b} >$. We denote by $\|\mathbf{a}\| = \sqrt{< \mathbf{a}, \mathbf{a} >}$, the ℓ_2 norm of a vector. $X_1, \ldots, X_n$ are non ordered variates, $x_1, \ldots, x_n$ non ordered observations. "Ordered statistics" means either $p_{(1)} \leq \ldots \leq p_{(n)}$ (ordered variates) and $p_{(1)} \leq \ldots \leq p_{(n)}$ (ordered observations). The extreme order statistics are $p_{(1)} = \min\{x_1, x_2 \ldots, x_n\}$, $p_{(n)} = \max\{x_1, x_2, \ldots, x_n\}$. The sample range is $p_{(n)} - p_{(1)}$. The $p_{(i)}$ are necessarily *dependent* because of the inequality relations among them.

Definition 1 (Savage [12]). *The rank order corresponding to the n distinct numbers $x_1, \ldots, x_n$ is the vector $\boldsymbol{r} = (r_1, \ldots, r_n)^T$ where r_i is the number of x_j's$\leq x_i$ and $i \neq j$.*

The rank order $\boldsymbol{r}$ is always unambiguously defined as a *permutation* of the first n integers.

2 Texture encoding

Local image descriptors are compact and rich descriptions capable of encoding local patterns *e.g.* local binary patterns (LBP) and its variants [2]. Since imagery data is noisy, locally correlated, and usually with too many data points per "unit" of useful information, these intermediate representations lead to an understanding of the scene in an image without the noisy and devoid influence sense of very many pixels that carry little inferable information. These representations typically encode either structural (texture) information in the form of a set of repeated primitive *textons*, or statistical information indicating how different pixel intensities are arranged in a local neighborhood.

The amount of information extracted from different regions of an image depends on (a) the size of the neighborhood (b) the reading order of the neighbors (c) and the mathematical function that is used to extract the relationship between two neighboring pixels. Texture gives information on the spatial arrangement of the grey levels in an image.

For instance, given a monochromatic image I, LBP generates 8-bit string for a 3×3 neighborhood by computing the Heaviside function $t(x)$ of the difference of neighboring pixel $\{g_i | i = 0, \ldots, p-1\}$ and the central pixel g_c *i.e.* $(g_i - g_c)$ (see Fig. 1), but $(g_c - g_i)$ in case of Census transform. The only difference between these two descriptors is the reading order of neighboring pixels and the sign of the difference which results in 2 different bit patterns. Given the 8-bit string, the LBP code is calculated in the range of 0 to 255 as:

$$L_{p,r} = \sum_{i=0}^{p-1} 2^p \cdot t(g_i - g_c) \text{with } t(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{otherwise} \end{cases}, \tag{3}$$

where p counts the number of pixels in the neighborhood of g_c, considering the distance R between central pixel g_c and the neighboring pixel g_i.

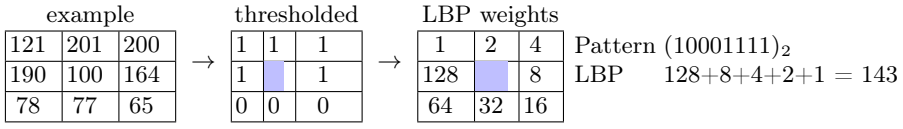


Fig. 1: Example of 3×3 image neighborhood ($p = 8$ and $R = 1$).

Invariance w.r.t. any monotone transformation of the gray scale is obtained by considering in (3) the signs of the differences $t(g_i - g_c), i = 0, \ldots, P - 1$.

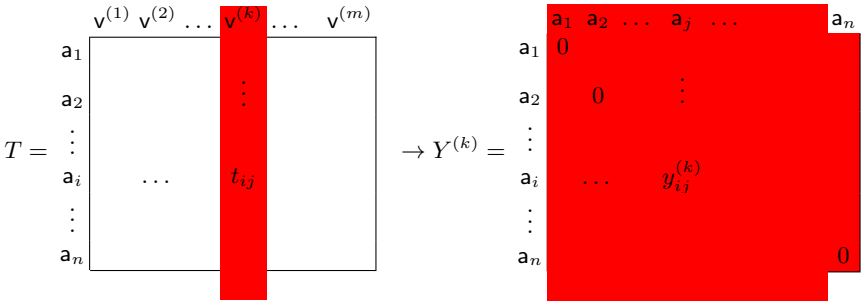
But the *independence* of g_c and $\{|g_0 - g_c|, \ldots, |g_{P-1} - g_c|\}$ is not warranted in practice. Moreover, under certain circumstances, LBP misses the local structure as it does not consider the central pixel. The binary data produced by these descriptors are sensitive to noise mainly in uniform regions of an image.To reduce the noise sensitivity [15] have proposed a 3-level operator which describes a pixel relationship with its neighbor by a ternary code *i.e.* -1,0,1 rather than a binary code *i.e.* 0,1. These methods ignore the relationship between neighboring pixels themselves and fail to get robust results [10]. It is inspired by the traditional image feature extraction method HOG [19] which extracts the relative orders on small local areas.

In this paper, a new encoding function is used to decide the relative order between the pixels in each local area of a given pixel. This cost function is minimized by linear programming and generates rank orders (including ex-aequos positions) which could be used in contour detection, segmentation or adaptive image quantization.

3 Total rank order for image texture encoding

Let $A = \{\mathsf{a}_1, \mathsf{a}_2, \ldots, \mathsf{a}_n\}$ be a set of alternatives, candidates, individuals, etc. with cardinality $|A| = n$ and let V be a set of voters, judges, criteria, etc. with $|V| = m$. The data is collected in a $(n \times m)$ table T of general term $\{t_{ij}\}$ crossing the sets A and V (Tab. 1a). t_{ij} can be marks ($t_{ij} \in \mathbb{N}$), value scales ($t_{ij} \in \mathbb{R}$), ranks of notes or binary ($t_{ij} \in \{0, 1\}$ such as opinion yes/no).

In the following, m is the number of alternatives and m the number of voters.



(a) Data matrix T ($t_{ij} \geq 0$).

(b) kth pairwise comparison matrix between alternatives a_i and a_j.

Table 1: The data are collected in a $(n \times m)$ table T.

The objective is to find a distribution of values $\boldsymbol{x}^*$ attributed by a virtual judge to the n individuals of the studied population by minimizing the disagree-

ments of opinions of the m judges, *i.e.*

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{t}} \sum_{k=1}^{m} \mathrm{d}(\boldsymbol{t}, \boldsymbol{t}^{(k)}), \quad \text{s.t.} \quad \boldsymbol{x} \geq 0, \tag{4}$$

$\mathrm{d}(\boldsymbol{t}, \boldsymbol{t}^{(k)})$ being a metric measuring the proximity between $\boldsymbol{t}$ and $\boldsymbol{t}^{(k)}$, chosen a priori, and $\boldsymbol{t}^{(k)}$ is the kth column of the table T. Depending on the nature of d, we will see that we will be dealing with a nonlinear optimization program with an explicit or implicit solution.

One could also stand the dual problem of the previous one, *i.e.* : is there a distribution of ratings/values that could have been attributed by the m voters to a virtual alternative 'a' summarizing the behaviour of the set of individuals A [20]? The first problem is linked to the idea of aggregating of points of view, the second to the idea of summarizing behaviors.

3.1 Explicit resolution

The distance $\mathrm{d}(\boldsymbol{t}^{(k)}, \boldsymbol{t}^{(k')})$ between the voter k and the voter k' can be chosen for instance as the *disagreement* distance $\sum_{i=1}^{n} \mathrm{sgn}\,|t_{ik} - t_{ik'}|$ where $\mathrm{sgn}(a) = a/|a|$, the holder distance $(\sum_{i=1}^{n} |t_{ik} - t_{ik'}|^m)^{1/m}, m > 1$, the d_∞ distance defined as $\max_i |t_{ik} - t_{ik'}|$ which is also the limit of the Holder distance when $m \to \infty$, la distance du χ^2, etc.

Surprisingly, the explicit resolution with linear programming optimization explicitly leads to central tendency statistics, *e.g.* the mode for the disagreement distance, the median for the absolute deviations, the mean for the Holder distance ($m = 2$), etc. [3].

Let's take an example. Assume the t_{ik} represent the scores obtained by a neurologist i in m services: emergency, surgery, palliative care, rehabilitation, etc. and suppose that n neurologists are candidates for a position in a neurovascular unit (NVU). What can be the hiring technique of the chief of the NVU? If he is careful and wants his staff to be interchangeable throughout the year, he can take either of these "medium" measures. If he wants to be efficient, and accepts that the neurologist is not interchangeable, he will take the disagreement metric, and choose the neurologist whose mode is the strongest. If the chief of the NVU wants to satisfy neurologists with the breadth of their capabilities, he will use the d_∞ metric.

In this example it is asked *in fine* to compare different individuals using a common scale.

On the opposite, ranked variables are (a) easily collated (b) easily categorized (c) easy to analyze (d) they have an intuitive and plausible interpretation (e) they provide the best possible description of the process of ranking items as performed by a human (f) provide very good concordance indicator.

We still have n voters, and m alternatives but in this case, the m voters give a ranking of the candidates with or without ex-aequo in the form of the table T or $t_{ik} = r_{ik}$ (rank given by voter k to individual i). Each ranking V^k is a permutation P of the first n integers identifying the candidates in case of strict preference.

3.2 Order disagreement distance

When looking for the optimal consensus $\boldsymbol{r}^*$ of m voters who attributed the votes $\boldsymbol{r}^{(1)}, \boldsymbol{r}^{(2)}, \ldots, \boldsymbol{r}^{(m)}$ to the n candidates $\{\mathsf{a}_1, \mathsf{a}_2, \ldots, \mathsf{a}_n\}$, we minimize the absolute deviation distance

$$\sum_{k=1}^{m}\sum_{i=1}^{n} |r_i^* - r_{ik}|. \tag{5}$$

where, for ease of writing, $r_{ik} = r_i^{(k)}$. A voter can give *ex-aequo* positions. Note that $\boldsymbol{r}^{(k)} \notin \mathcal{S}_n$, with $\mathcal{S}_n$ the symmetric group of the $n!$ permutations [6] because of the ex-aequo. Hence $\boldsymbol{r}^* \notin S_n$.

To define this distance, we define a new set of tables $\{Y^{(1)}, \ldots, Y^{(m)}\}$, where $Y_{ij}^{(k)} = \mathbb{1}_{i<j}$ denotes the indicator matrix for which $y_{ij}^{(k)} = 1$ if the rank of the alternative a_i is *less* than the alternative a_j and 0 otherwise (see Tab. 1b).

Using the tables $Y^{(k)}$

$$\sum_{k=1}^{m}\sum_{i=1}^{n} |r_i^* - r_{ik}| = \frac{1}{2}\sum_i\sum_j |y_{ij}^{(k)} - y_{ij}^{(k')}| \tag{6}$$

or $\frac{1}{2}\sum_i\sum_j (y_{ik}^{(k)} - y_{ik}^{(k')})^2$. which can be simplified in the case of total order as

$$\sum_i\sum_j y_{ij}^{(k)} y_{ji}^{(k')}. \tag{7}$$

The $y_{ij}^{(k)}$ verify *(a)* the transitivity relationship : if $y_{ij}^{(k)} = 1$ and $y_{j\ell}^{(k)} = 1$ then $y_{i\ell}^{(k)} = 1$, equivalent to $y_{ij} + y_{ji} - y_{ik} \leq 1, i \neq j \neq k,\ y_{ij} \in \{0,1\}$ *(b)* $y_{ij} + y_{ji} \leq 1$, with equality only when $\boldsymbol{r}^{(k)}$ is a total order.

As $y_{ij}^2 = y_{ij} = {y_{ij}^{(k)}}^2 = y_{ij}^{(k)} = 0$ or 1, the distance function associated to Eq. (6) is given by

$$\frac{1}{2}\left[\sum_{i=1}^{n}\sum_{j=1}^{n} m y_{ij} + \sum_{i=1}^{n}\sum_{j=1}^{n}\left(\sum_{k=1}^{m} y_{ij}\right) - 2\sum_{i=1}^{n}\sum_{j=1}^{n} y_{ij}\sum_{k=1}^{m} y_{ij}^{(k)}\right]. \tag{8}$$

Let $\alpha_{ij} = \sum_{k=1}^{p} y_{ij}^{(k)}$ the total number of voters preferring alternative a_i to a_j and define a matrix $\mathcal{A} = \{\alpha_{ij}\}$, summing the m matrices $Y^{(k)}$ associated to the rankings $\boldsymbol{r}^{(k)}$ of the voters $V^{(k)}$. Eq. (8) can be rewritten using the α_{ij}:

$$\frac{1}{2}\left[\sum_{i=1}^{n}\sum_{j=1}^{n} m y_{ij} + \sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_{ij} - 2\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_{ij} y_{ij}\right] \tag{9}$$

As $\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_{ij} < \dfrac{n(n-1)}{2}$, and let $K = \sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_{ij}$ a constant. Then (9) is:

$$K - \sum_{i=1}^{n}\sum_{j=1}^{n}(\alpha_{ij} - m/2) y_{ij}. \tag{10}$$

Finally the search of a order given by a matrix Y is the optimal solution of the following *linear program*

$$\begin{aligned} \max_{Y} \sum_{i=1}^{n}\sum_{j=1}^{n}(\alpha_{ij} - m/2)y_{ij} \quad \text{s.t.} \quad \alpha_{ij} = \sum_{k=1}^{m} y_{ij}^{(k)}, \\ y_{ij} + y_{ji} = 1, i < j, y_{ii} = 0 \\ y_{ij} + y_{ji} - y_{ik} \leq 1, i \neq j \neq k,\ y_{ij} \in \{0,1\}. \end{aligned} \tag{11}$$

From a machine learning perspective, Eq. (11) is remarkably simple and provide an exact solution using a linear programming solver [7].

4 Rank-Order Principal Components

4.1 Importance ranking

Several strategies have been proposed in the literature to extract important variables or develop parsimonious models and deal with the dimensionality. The dimension of observed data being generally higher than their intrinsic dimension, it is theoretically possible to reduce the dimension without loosing information.

Among the unsupervised tools, principal component analysis (PCA) or factor analysis (FA) are certainly the most used techniques to optimize the understanding insight into of a data set. They aim to project the data onto a lower dimensional subspace in which axes are constructed either by maximizing the variance of the projected data or by explaining the overall covariance structure.

PCA and FA are both linear tools. This means that nonlinear dependencies are not taken into account.

The question is simply: can we extract a set of the most decorrelated rank-order variables to each other capable of capturing distinct information? The overall framework for this objective suggests a rank-order decomposition. The principle remains remarkably simple: it consists into a re-distributive effect of the rank variables – similar to PCA – on a Hilbert space.

Lemma 1 (Vigneron and Duarte [16]). *Consider a collection of rank-orders (with ex-aequo or not) $R = \{\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_m\}$ (data). It is always possible to extract a total rank-order component $\boldsymbol{g}_\ell$ minimizing its proximity to the data $\{\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_m\}$ and simultaneously maximizing the distance to the collection of previously calculated ranks $\{\boldsymbol{g}_1, \ldots, \boldsymbol{g}_{\ell-1}\}$.*

The algorithm is as follows:

Algorithm 1 Rank-order decomposition Algorithm.

Require: $Y^{(1)}, \ldots, Y^{(m)} \leftarrow \{\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_m\}$ {order disagreement matrices $Y^{(k)} = \{y_{ij}^{(k)}\}\}\vee$ stack $A = \emptyset$ {contain the reranked components}
Ensure: $\{\boldsymbol{g}_1, \ldots, \boldsymbol{g}_m\}$ {Postcondition}
1: **for** $\ell = 1$ **to** m **do**
2: Compute $\alpha_{ij} = \sum_{k=1}^{m} y_{ij}^{(k)}$, $\beta_{ij} = \sum_{k=1}^{\ell-1} z_{ij}^{(k)}$
3: $\mathcal{A} = \{\alpha_{ij}\}, \mathcal{B} = \{\beta_{ij}\}$
4: **LP**$(\mathcal{A}, \mathcal{B}, Z^{(\ell)})$ under constraints (12) {solve linear program}
5: $\boldsymbol{g}_\ell \leftarrow Z^{(\ell)}$
6: **end for**
7: **return** $\{\boldsymbol{g}_1, \boldsymbol{g}_2, \ldots, \boldsymbol{g}_m\}$

At stage ℓ, the search of the ℓth total order $\boldsymbol{g}_\ell$ represented by the matrix $Z^{(\ell)}$ in the case of the order disagreement distance (see section 3.2) reduced to

$$\begin{aligned} &\max_{Z^{(\ell)}} \sum_{i=1}^{n}\sum_{j=1}^{n}(\alpha_{ij} - m/2)z_{ij}^{(\ell)} - \max_{Z^{(\ell)}} \sum_{i=1}^{n}\sum_{j=1}^{n}\beta_{ij}z_{ij}^{(\ell)} \quad \text{s.t.} \\ &\alpha_{ij} = \sum_{k=1}^{m} y_{ij}^{(k)},\ \beta_{ij} = \sum_{k=1}^{\ell-1} z_{ij}^{(k)},\ z_{ij}^{(\ell)} + z_{ji}^{(\ell)} = 1, i < j, \\ &z_{ii} = 0\ \ z_{ij}^{(\ell)} + z_{ji}^{(\ell)} - z_{ik}^{(\ell)} \leq 1, i \neq j \neq k,\ z_{ij}^{(\ell)} \in \{0, 1\}. \end{aligned} \tag{12}$$

Algorithm 1 stops when $\ell = m$ and provide $\{\boldsymbol{g}_1, \ldots, \boldsymbol{g}_m\}$ such that $\boldsymbol{g}_\ell$ is the most decorrelated to the previous ranks $\{\boldsymbol{g}_1, \ldots, \boldsymbol{g}_{\ell-1}\}$. Until now, on the contrary to PCA, there is no index capable of indicating the quantity of information captured by each vector $\boldsymbol{g}_\ell$.

4.2 Experiment: Application of rank-order pooling (RO) principal components to textured image

Now consider the "neighborhood" of a pixel 'p' in an image I, *i.e.* the set of pixels touching it (a maximum of 8 pixels) as shown in Fig. 1.

The 4×4 image I in Figure 2a can be transformed using the 8-connectivity into the 4×8 matrix C (Fig. 2b) where column 0 refers to the 1st pixel (clockwise), column 1 refers to the 2nd pixel, etc. The 8 neighboring pixels around 'p' can be seen as "voters" from which a pseudo-rank is expected. The matrix C is then transformed into the rank-matrix $R = \{\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_8\}$ (Fig. 2c) by simple ordering on which algorithm 1 can be applied. Note that usually the peripheral zone is filled with zero, *i.e.* added with borders of zeros, to preserve the size of the image.

As an illustration LBP is applied to original picture 3a of Lena which provides the texture representation given in Figure 3b. From the figure 3a,algorithm 1 provides a new set of images (Figures 3c-3f). Usually, the 1st plot is more informative the 2nd one, which itself is more informative than the 3rd, etc. Vigneron

	1	2	3	4
1	4	8	20	18
2	17	12	17	17
3	6	1	17	17
4	4	6	5	14

(a) 4×4 image I.

		neighbors							
		0	1	2	3	4	5	6	7
center pix.	12	4	8	20	17	17	1	6	17
	17	8	20	18	17	17	17	1	6
	1	17	12	17	17	5	6	4	6
	17	12	17	17	17	14	5	6	1

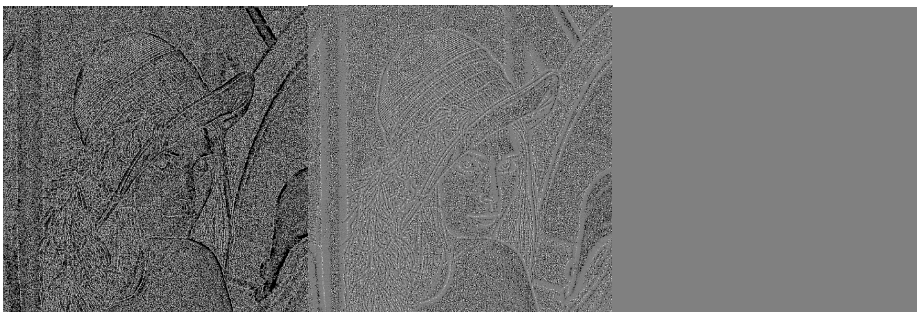
(b) 8-connectivity matrix C.

		ranks							
		$\boldsymbol{r}_0$	$\boldsymbol{r}_1$	$\boldsymbol{r}_2$	$\boldsymbol{r}_3$	$\boldsymbol{r}_4$	$\boldsymbol{r}_5$	$\boldsymbol{r}_6$	$\boldsymbol{r}_7$
center pix.	12	1	1	3	1	3	1	3	3
	17	2	4	2	1	3	4	1	2
	1	4	2	1	1	1	3	2	2
	17	3	3	1	1	2	2	3	1

(c) Rank matrix R.

Fig. 2: The colored pixels are 8-connected. The 8 neighbor pixels around the central pixel 'p' can be seen as "voters" from which we expect a pseudo-rank.

(a) Original **(b)** Classic LBP **(c)** RO comp. 1



(d) RO comp. 2 **(e)** RO comp. 3 **(f)** RO comp. 8

Fig. 3: Lena' original (a) is compared to classic LBP representation (b) and with the 1st, 2nd, 3rd and eighth rank-order components obtained from Algorithm 1. The eighth component is apparently the less informative component.

and Duarte have shown [17] that this decomposition is sensitive to the chosen metric.

5 Rank-order pooling operator

5.1 Pooling layer

Pooling perform a derivative operation of the entries in the window (*e.g.* max, mean, median, etc.) to "collapse" over values. Max-pooling down-samples the convolutional output in a discrete way. The method of calculating the 2d- pooling layer is the same as for the convolutional layer and produces a image of size :

$$\left(\left\lfloor \frac{I_x - P}{S} \right\rfloor + 1, \left\lfloor \frac{I_y - P}{S} \right\rfloor + 1 \right) \tag{13}$$

with two hyperparameters: the stride S and the spatial extent P. For instance, for a 13×13 image, 2×2 pooling window and a stride of 2, the x-dimension of the resulting image is floor((13-2)/2)+1 = 6. It is not common to pad the input using zero-padding for the pooling.

The rules presented in section 1 apply to all types of pooling layers with some adjustments. Forward propagation for max pooling means creating a mask that stores the position of the retained maximum values through which the gradients are transferred.

Instead of taking the maxima in each filter, the rank-order pooling approach takes the consensus rank in a neighborhood. Interestingly, the most common tested configurations of RO in practice are: $P = 3, S = 2$ and $P = 3, S = 3$. Pooling sizes with larger receptive fields result in too much loss and are destructive. It should also be noted that as max-pooling, the RO pooling introduces zero parameters since it computes a fixed function of the input.

Different pooling operations were performed in a categorization context to compare the behavior of different pooling operations in a categorization context.

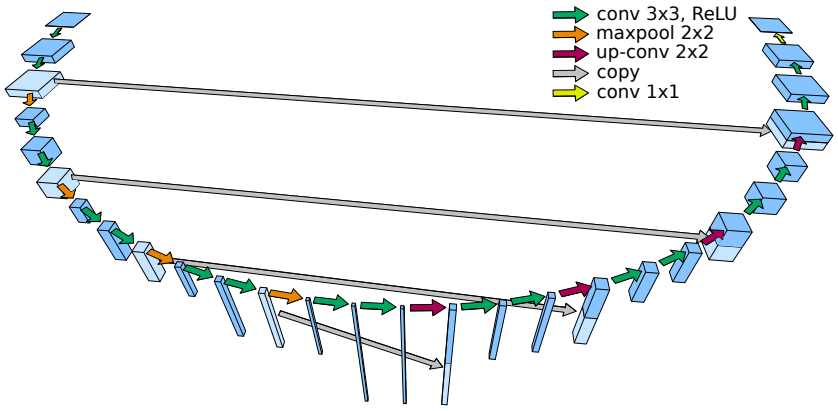
5.2 Experiment: RO pooling for automatic tumor segmentation

For the second experiment we chose a real data set which is renowned for its difficulty. The brain tumor segmentation (BraTS) [9] challenge is a recurring challenge attached to the MICCAI Conference. Each year the segmentation results become better, but the problem is an ongoing research. For this experiment we use the high grade glioma part of the BraTS 2017 data-set[1]. It contains multi-modal MRI of 210 patients which were manually segmented my experts, *i.e.* a ground truth is available. On these image three different classes have to be segmented from the background. The enhancing tumor, the necrotic and non-enhancing tumor and as a third class the peritumoral edema. This makes it an ideal real data-set for supervised learning of a multi-class segmentation task.

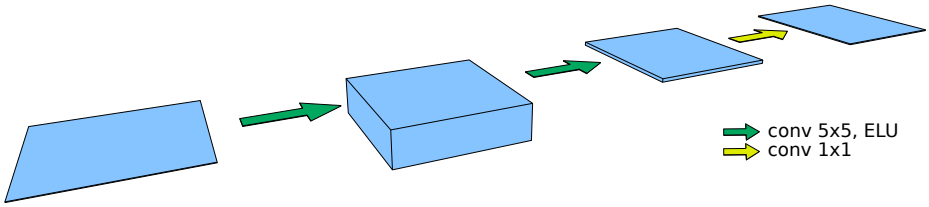
Architectures We compare architectures containing the RO pooling to model without [18]. The U-Net was chosen because it is the archetype of modern convolutional networks used for bio-medical image segmentation tasks and achieved

[1] www.med.upenn.edu/sbia/brats2017.html

good performance in many applications. To prove that the tasks are not too simple and that the RO layer is responsible for the improved results, a simple reference network with max-pooling layers is included. For evaluating the results we not only use the loss but also the Dice coefficient [14] which is the standard measure for segmentation quality.



(a) U-Net. Arrows represent operations and cubes represent feature maps where the height of the cube stands for the number of feature maps and the width and depth of the cubes for the size of the feature maps. It is clearly visible that the U-Net uses a pyramidal structure for feature extraction.



(b) Reference-Net.The most basic CNN. It suffers greatly from a small receptive field. As expected our first experiment showed that it has trouble with objects that are larger than the convolution kernels. This weakness can be overcome either by adding a single transfer layer to the architecture (Transfer-Net), or by adding so many convolutional layers that the chained convolution kernels extend the receptive field to the whole input image (U-Net).

Fig. 4: DNN architectures.

All of the following network architectures are implemented in Pytorch [2] and the implementations will be freely accessible. Convolution layers are chosen to keep the output size the same as the input size by padding the input image with zeros. Also all architectures are followed by a softmax layer. and cross entropy is

[2] pytorch.org

used as loss function. The loss is minimised using Adam [13]. The best learning rate for each architecture has been determined experimentally.

U-Net: [18] (figure 4a) consists mainly of a feature extraction pyramid followed by an expanding path which up-samples the features to the space of the original image. A special feature of the U-Net are its skip connections which allow it to preserve fine grained details. The U-Nets work with RO pooling layers with hyperparameters 3,3 and 3,2 (F, S), see section 5.1.

Reference Net (R-Net):, see Fig. 4b which is used to demonstrate that it is our pooling layer which is responsible for our results.

Training Data Sampling: We divided the data-set into 100 patients for learning , 4 for validation and left 106 aside as test set. As we just wanted to demonstrate a concept we never used the test set in the end. Four different MRI modalities (compare figure 5 a)-d)) are available, which means that the input image has four channels. The images were normalized to the mean value of healthy tissue, *i.e.* the intensity values were divided by the intensity value of the highest peak in the histogram of brain tissues.

As the whole MRI is too large to process in one step it was broken down into patches of size $64 \times 64 \times 10$. Patches are generated until the whole brain is sampled or a number of 100 patches is reached. This procedure is repeated for each patient. The batches for the training are generated as follows: Each odd sample of the batch is the guaranteed tumor patch of a random patient and the following even sample is a random patch of the same patient. This sampling scheme guarantees that each batch shows tumor as well as non tumor regions and effectively combats class imbalance. We use a batch size of four patches per batch, *i.e.* after 50 patches, the guaranteed tumor patch of all 100 training patients have been seen. This may be considered an epoch. All patches are not guaranteed to be seen during training.

We forego any advanced data augmentation. Only a small random constant is added to the entire patch.

Table 2: Network configurations for the BraTS experiment and their final dice score on the training and validation set. $m \in \mathbb{N}$ is the multiplicity, *i.e.* the number of input channels.

Model	U-Net Maxpool	U-Net 3.3	U-Net 3.2	R-Net 3.2	R-Net maxPool
Kernel Size	3	3	5	5	5
Parameters	31032516	7698116	141929	19359	19359
Learning Rate	0.001	0.002	0.0005	0.002	0.002
m	-	-	10	10	-
Training Speed τ	10737	15032	13322	28761	48767
Training Dice	0.96	0.82	0.87	0.80	0.64
Validation Dice	0.89	0.51	0.77	0.51	0.49

6 Results and conclusion

This real data-set is a way more difficult task than texture characterization. Looking at the dice score (Tab. 2) shows a lower loss value does not necessarily indicate a better segmentation performance. The networks based on the RO pooling perform well. Of course we cannot expect to beat the U-Net 3.2 with a network which has only 0.44% of the parameters. But the R-Net 3.2 comes remarkably close in terms of performance and clearly beats the U-Net 3.3 which still has 56 times more parameters. Even the R-Net 3.2 is close to the U-Net 4 on the validation set while having less than 20 thousand parameters compared to more than 7 million of the U-Net 4.

Segmentations of one patch are given in figure 5. Here the quality of the segmentation of the U-Net 3.2 and R-Net 3.2 is visible. They are very close to the ground truth and the U-Net MaxPool segmentation while the U-Net 3.3 segmentation is clearly worse. This proves that theRO pooling works, and that it delivers performance at a low parameter cost.

We challenged the concept for feature extraction which has been uncontested for three decades, the feature extraction pyramid. In two experiments we showed that our RO pooling can master the same tasks as a feature extraction in emphasizing low/high frequencies, contours, etc. We shows rank-order pooling leads to CNN models which can optimally exploit information from their receptive field.

References

1. Baker, R.: New order-statistics-based ranking models and faster computation of outcome probabilities. IMA Journal of Management Mathematics **31**(1), 33–48 (2019). https://doi.org/10.1093/imaman/dpz001
2. Brahnam, S., Jain, L., Nanni, L., Lumini, A. (eds.): Local Binary Patterns: New Variants and Applications, Studies in Computational Intelligence, vol. 506. Springer (2014)
3. Figueira, J., Greco, S., Roy, B.: Electre-score: A first outranking based method for scoring actions. European Journal of Operational Research **297**(3), 986–1005 (2022). https://doi.org/https://doi.org/10.1016/j.ejor.2021.05.017, https://www.sciencedirect.com/science/article/pii/S0377221721004318
4. Gholamalinezhad, H., Khosravi, H.: Pooling methods in deep neural networks, a review. ArXiv **abs/2009.07485** (2020)
5. Goodfellow, I., Bengio, Y., Courville, A.: Deep learning, vol. 1. MIT press Cambridge (2016)
6. Gurevich, S., Howe, R.E.: Rank and duality in representation theory. Japanese Journal of Mathematics **15**, 223–309 (2020)
7. Iatan, I.: Issues in the Use of Neural Networks in Information Retrieval. Studies in Computational Intelligence 661, Springer International Publishing, 1 edn. (2017)
8. Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In: Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition. vol. 2, pp. 2169 – 2178 (Feb 2006). https://doi.org/10.1109/CVPR.2006.68

9. Menze, H.: The multimodal brain tumor image segmentation benchmark (brats). IEEE Transactions on Medical Imaging **34**(10), 1993–2024 (Oct 2015). https://doi.org/10.1109/TMI.2014.2377694
10. Ouslimani, F., Ouslimani, A., Ameur, Z.: Rotation-invariant features based on directional coding for texture classification. Neural Computing and Applications **31**, 1–8 (10 2019). https://doi.org/10.1007/s00521-018-3462-9
11. Satoh, Y., Hirata, K., Tamada, D., Funayama, S., Onishi, H.: Texture analysis in the diagnosis of primary breast cancer: Comparison of high-resolution dedicated breast positron emission tomography (dbpet) and whole-body pet/ct. Frontiers in Medicine **7** (2020). https://doi.org/10.3389/fmed.2020.603303, https://www.frontiersin.org/article/10.3389/fmed.2020.603303
12. Savage, R.: Contributions to the theory of rank-order statistics – the trend case. The Annals of Mathematical Statistics **27**(3), 590–615 (Sept 1956)
13. Sen, S., Ozkurt, N.: Convolutional neural network hyperparameter tuning with adam optimizer for ecg classification. In: 2020 Innovations in Intelligent Systems and Applications Conference (ASYU). pp. 1–6 (2020). https://doi.org/10.1109/ASYU50717.2020.9259896
14. Shamir, R., Duchin, Y., Kim, J., Sapiro, G., Harel, N.: Continuous dice coefficient: a method for evaluating probabilistic segmentations. bioRxiv (2018). https://doi.org/10.1101/306977, https://www.biorxiv.org/content/early/2018/04/25/306977
15. Tan, X., Triggs, B.: Enhanced local texture feature sets for face recognition under difficult lighting conditions. In: Analysis and Modeling of Faces and Gestures. Lecture Notes in Computer Science, vol. 4778, pp. 235–249. Springer, Berlin (2007)
16. Vigneron, V., Duarte, L.: Toward rank disaggregation: An approach based on linear programming and latent variable analysis. In: Tichavský, P., Babaie-Zadeh, M., Michel, O.J., Thirion-Moreau, N. (eds.) Latent Variable Analysis and Signal Separation. pp. 192–200. Springer International Publishing, Cham (2017)
17. Vigneron, V., Tomazeli Duarte, L.: Rank-order principal components. A separation algorithm for ordinal data exploration. In: 2018 International Joint Conference on Neural Networks, IJCNN 2018, Rio de Janeiro, Brazil, July 8-13, 2018. pp. 1–6 (2018)
18. Wang, F., Jiang, R., Zheng, L., Meng, C., Biswal, B.: 3d u-net based brain tumor segmentation and survival days prediction. In: Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries, pp. 131–141. Springer International Publishing (2020). https://doi.org/10.1007/978-3-030-46640-4_13, https://doi.org/10.1007%2F978-3-030-46640-4_13
19. Wang, L., Li, T.: Research on image feature extraction method fusing hog and canny algorithm. In: 2021 4th International Conference on Data Science and Information Technology. p. 208–211. DSIT 2021, Association for Computing Machinery, New York, NY, USA (2021). https://doi.org/10.1145/3478905.3478947, https://doi.org/10.1145/3478905.3478947
20. Yadav, N.. author. abd Yadav, A., Kumar, M.: An Introduction to Neural Network Methods for Differential Equations. Springer, Gurgaon, Haryana, India (2015)
21. Yu, D., Wang, H., Chen, P., Wei, Z.: Mixed pooling for convolutional neural networks. In: International Conference on Rough Sets and Knowledge Technology. pp. 364–375 (2014)

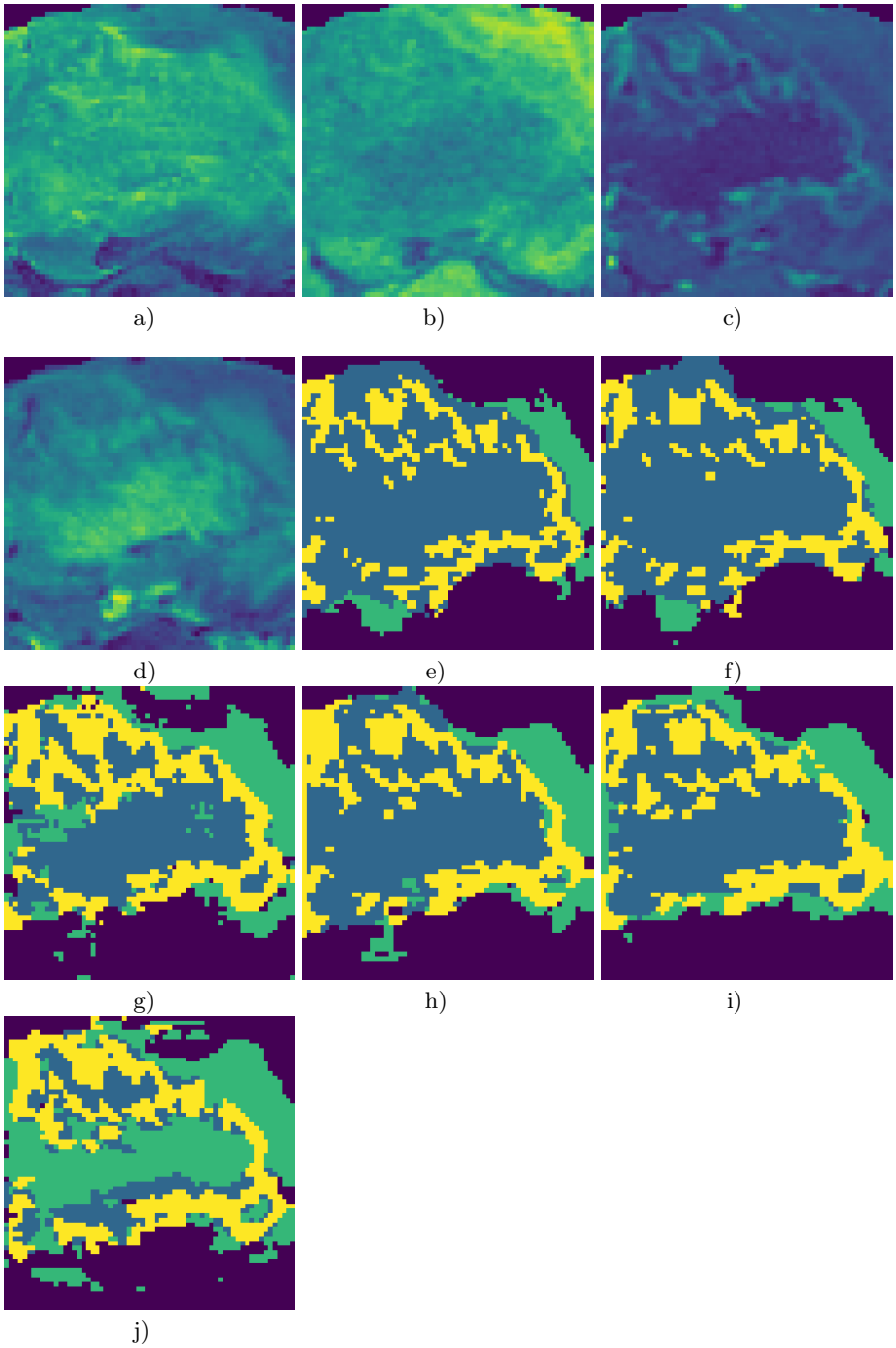


Fig. 5: One patch out of the BraTs validation-set. The input image a)-d) consists of the MRI modalities Flair a), T1 b), contrast enhanced T1 c) and T2 d). The ground truth is seen in e). The following are the predictions of the five networks: U-Net 3.2 f), U-Net 3.3 g), U-Net max-pooling h), R-net 3.2 i) and the R-Net max-pooling).