Improving the Identification of Layers in 3D Images of Ancient Papyrus using Artificial Neural Networks

Supplementary Material

Nicolas Klenert[1] klenert@zib.de
Finn Schwoerer[1]
Noushin Hajarolasvadi[1]
Siloé Bournez[1]
Tobias Arlt[2]
Heinz-Eberhard Mahnke[2]
Verena Lepper[3]
Daniel Baum[1] baum@zib.de

[1]Zuse Institute Berlin, Department of Visual and Data-Centric Computing, Berlin, Germany
[2]Helmholtz-Zentrum Berlin für Materialien und Energie, Berlin, Germany
[3]Ägyptisches Museum und Papyrussammlung, Staatliche Museen zu Berlin, Berlin, Germany

Abstract

Here we present additional information for the manuscript entitled "Improving the Identification of Layers in 3D Images of Ancient Papyrus using Artificial Neural Networks". Due to page limitations, this information did not fit into the main manuscript.

S1. Details to the used U-Net Architecture

Fig. S1 shows the architecture of our U-Net, which only consists of Stacked Convolutional Blocks (SCB) and transposed convolution operations (CT). An SCB with a CT afterwards is also abbreviated with SCT. All neural network operations are implemented with PyTorch and we refer to their layer names. The only exception to this is the *CAT* operation, which takes two tensors and concatenates them along the feature map axis. Fig. S1 shows the concatenation operation as blue balls with two inputs and one output. A CT is called a *ConvTranspose3d* module in PyTorch. An SCB consists of two blocks containing three modules each: *Conv3d*, *InstanceNorm3d*, and *LeakyReLU*. All *Conv3d* modules have a padding of 1 for each axis and a bias. The dimension reduction of the images are done by a stride length of 2 at the first *Conv3d* module in an SCB and are represented by the red boxes in Fig. S1. The stride length, image dimension and number of feature maps are shown in Tab. S1. In total our network has $30,785,994$ parameters.

The input is a normalized image of size $64 \times 64 \times 64$. The output has as many feature maps as the number of classes the network was trained on. In our case, as we have a binary label field, there are exactly 2 feature maps. The output feature maps can be interpreted as the probability maps of the image patch, that is, each voxel has a value representing the probability that the voxel is of the specified class. The probability maps shown in the main paper are directly taken from the network.

S2. Details to the Tomograms

Tab. S2 lists the properties of the papyrus image data used for inference. The original data was cropped. Note that the voxels are isotropic which means that the images have uniform resolution in all directions and such a single length specification is enough to define the voxel size. The grayscale image data consists of 16 bit unsigned values.

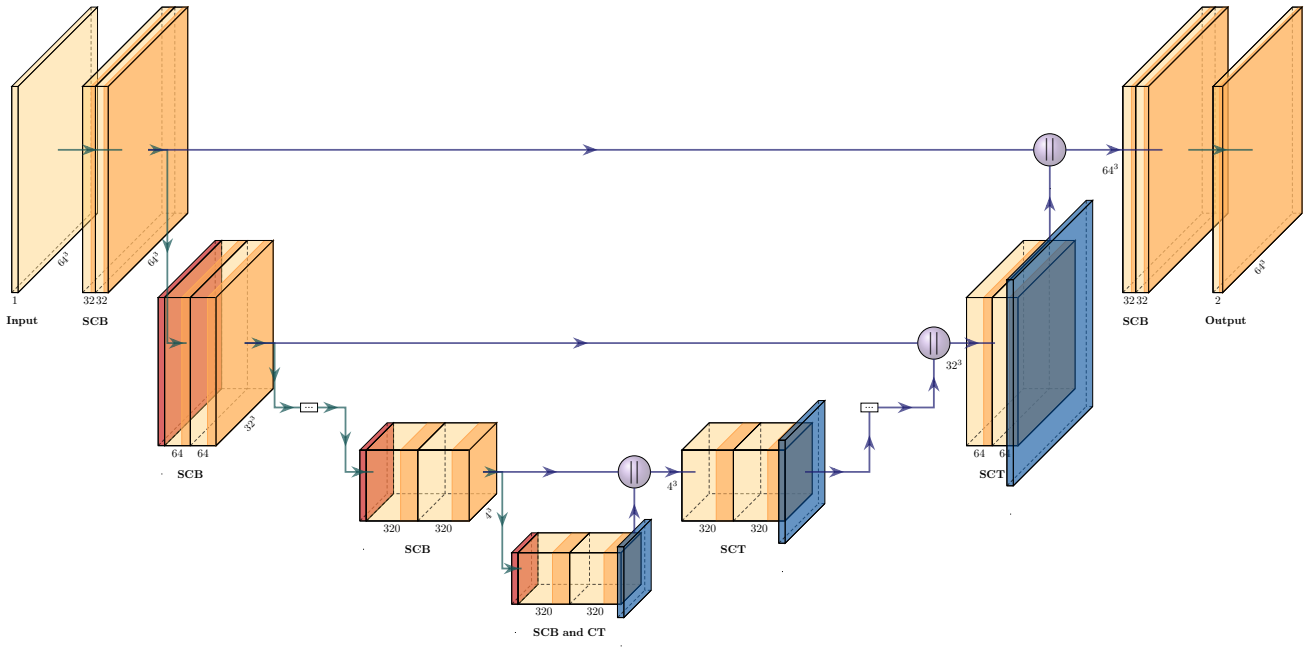

Figure S1. The uppermost and lowermost two rows of the U-Net architecture are shown. In total the network has 6 rows. SCB stands for Stacked Convolution Block with instance normalization and leaky ReLu. CT stands for Convolutional Transpose and SCT for Stacked Convolution Transpose. The exact makeup of these blocks is described in Sec. S1. The input is a normalized image of size $64 \times 64 \times 64$. The output are the probability maps for each class.

Table S1. All Modules of the U-Net. N is the number of feature maps, while X, Y and Z are the image dimensions. One CT, CAT and SCB application can be seen as one step for the decoder of the U-Net, which mirrors one SCB step of the encoder. The Output is generated by one last *Conv3d* module, reducing the number of feature maps. Note that the listed stride only applies to the first convolution in an SCB. The second one always uses a stride of $1 \times 1 \times 1$.

operation	N	$X \times Y \times Z$	stride
Input	1	$64 \times 64 \times 64$	
SCB	32	$64 \times 64 \times 64$	$1 \times 1 \times 1$
SCB	64	$32 \times 32 \times 32$	$2 \times 2 \times 2$
SCB	128	$16 \times 16 \times 16$	$2 \times 2 \times 2$
SCB	256	$8 \times 8 \times 8$	$2 \times 2 \times 2$
SCB	320	$4 \times 4 \times 4$	$2 \times 2 \times 2$
SCB	320	$4 \times 2 \times 2$	$1 \times 2 \times 2$
CT	320	$4 \times 4 \times 4$	$1 \times 2 \times 2$
CAT	640	$4 \times 4 \times 4$	
SCB	320	$4 \times 4 \times 4$	$1 \times 1 \times 1$
CT	256	$8 \times 8 \times 8$	$2 \times 2 \times 2$
CAT	512	$8 \times 8 \times 8$	
SCB	256	$8 \times 8 \times 8$	$1 \times 1 \times 1$
CT	128	$16 \times 16 \times 16$	$2 \times 2 \times 2$
CAT	256	$16 \times 16 \times 16$	
SCB	128	$16 \times 16 \times 16$	$1 \times 1 \times 1$
CT	64	$32 \times 32 \times 32$	$2 \times 2 \times 2$
CAT	128	$32 \times 32 \times 32$	
SCB	64	$32 \times 32 \times 32$	$1 \times 1 \times 1$
CT	32	$64 \times 64 \times 64$	$2 \times 2 \times 2$
CAT	64	$64 \times 64 \times 64$	
SCB	32	$64 \times 64 \times 64$	$1 \times 1 \times 1$
Output	2	$64 \times 64 \times 64$	$1 \times 1 \times 1$

Table S2. Properties of tomogram data (used as input for inference).

name	number of voxels	$X \times Y \times Z$	voxel size	memory footprint
L/El227b/1-pC	$1,442,568,998$	$487 \times 1534 \times 1931$	$12.0\,\mu\text{m}$	$2885.1\,\text{MB}$
L/El227b/2-pU	$3,719,830,662$	$1737 \times 1481 \times 1446$	$10.6\,\mu\text{m}$	$7439.7\,\text{MB}$
L/El227b/3-pU	$1,793,260,959$	$707 \times 1319 \times 1923$	$10.6\,\mu\text{m}$	$3586.5\,\text{MB}$
L/El227b/4-pG	$1,829,304,783$	$1873 \times 549 \times 1779$	$9.5\,\mu\text{m}$	$3658.6\,\text{MB}$