Data Analysis for Single Photon Calcium Imaging with Deep Learning

Wai Po Kevin Teng, Praveen Putti and Lisa Schneider
Artificial Intelligence Lab, Otto von Guericke University Magdeburg
Magdeburg, Germany
Email: wai.teng@st.ovgu.de, praveen.putti@st.ovgu.de, lisa.schneider@st.ovgu.de

***Abstract*—Calcium Imaging is a powerful method to simultaneously capture the activity of cells in the brain of rodents. Especially one-photon calcium imaging with head-mounted miniature microscopes became increasingly popular since it allows the observation of freely behaving rodents. However, single-photon imaging data processing remains a challenging task due to missing depth information, background fluctuation, low image contrast, and movement artifacts. Distinct cells are not clearly delineated and might overlap with others. Within this work, a novel deep learning architecture is proposed to detect cells within one-photon calcium imaging sessions. The approach introduces learnable coordinates, that learn to locate themselves in the center of cell activations. Our deep learning framework is presented in an end-to-end unsupervised learning manner, given the lack of annotated data for each frame. The performance of the proposed method is benchmarked against manually annotated data, generated by two neuroscientists, that aggregates the occurrence of neurons over the whole session. At the current state, the implemented deep learning framework reaches a mean precision of 0.46 and leaves room for improvement.**

***Index Terms*—Single-photon Calcium Imaging, Rodents, Learnable Coordinates, Deep Learning, Analysis of Synthesis, Attention mechanism, CoordConv**

I. Introduction

One of the vital questions in neuroscience revolves around the correlation between neural activity and behavioral choices. Resulting knowledge gives insights into the brain and contributes to the development of real-time brain-computer interfaces. For instance, neuron activation patterns may be analyzed to predict or enable a particular behavior. Calcium imaging is frequently applied within research to capture neural activity. It utilizes the small number of calcium ions that cross the neuron membrane when neurons propagate impulses [1], [2]. Accordingly, the activation of a neuron correlates directly with the dynamics of the calcium concentration. This correlation is leveraged to visualize neuron activities. Thereby, calcium indicators are utilized due to their ability to respond to the binding of calcium ions with their fluorescence properties. The activity of hundreds to thousands of cells is then captured via fluorescence microscopes. Often, calcium imaging is carried out with two-photon microscopes. Therefore, the processing for two-photon imaging data is relatively sophisticated. However, two-photon calcium imaging often requires a head restraint, which naturally affects the rodent's behavior and limits the scope of research. In order to capture action potential activity on freely behaving rodents, recently developed head-mounted miniature microscopes became increasingly popular [3]–[5], [5]. This method allows studying animal behavior in an uncontrolled way without any behavioral effects caused by restraints. Though, the processing of one-photon imaging data is still at the very beginning. The main challenges revolve around missing depth information and low image contrast. This involves that individual cells are not clearly outlined and might overlap. Movement artifacts are common, and computer memory requirements are high. As a consequence, cells are not always correctly identified by the existing detection algorithms, and manual validation of the outcome becomes crucial [6]. Several tools have been proposed to combine automatic cell detection algorithms with manual corrections of a human expert [6], [7]. Nevertheless, the manual correction and annotation of neuron masks is a tedious task. It is based on an analysis of spatial and temporal properties of potential neuron candidates over the whole sequence, which has to be performed by domain experts. A further issue is that different domain experts, often yield different manual annotations on the same dataset. As reported by [7], different labelers may disagree with 20% of the same dataset. In order to overcome these limitations, a deep learning framework is created within this work to analyze one-photon imaging data of freely moving rodents. The main objective is the identification of neurons and their positions based on the whole imaging sequences. A deep learning approach is selected over traditional methods [6]–[11] to reduce the need of a feature extraction algorithm, enabling the network to learn the feature representation by itself. However, a deep learning approach requires a large amount of data and entails a fine-tuning process of hyperparameters. Several deep learning frameworks have been proposed [7], [12], [13]. Multiple approaches incorporate a feature extraction pipeline as enhancement before processing the data in the deep learning framework [7], [13]. An end to end deep learning framework is implemented by [12]. However, this approach requires neuron annotations for each frame. In this work, an end-to-end unsupervised deep learning framework is proposed aiming to learn an extraction of spatiotemporal behavior of neurons over the whole sequence. This deep learning model is inspired by an autoencoder model, which widely used in the regime of unsupervised learning. Moreover, a soft attention mechanism [14]–[16] is implemented as a building block of

the latent space. Furthermore, a novel variant of the standard convolution layer, namely, CoordConv, is utilized to introduce learnable coordinates, which capture the spatiotemporal behavior of neuron candidates over the imaging sequence. Due to its unsupervised manner, this approach does not require manual annotations. However, the model prediction is compared to a manually annotated neuron mask to benchmark the model's performance.

The content of this work is divided into seven sections. The first section includes introductions to the topic of calcium imaging data analysis and states the objective of this work. Section II outlines existing traditional and deep learning approaches applied as cell detection methods. Afterwards, section III combines methods that are included in the proposed deep learning framework. This section also wraps up how the performance of the model is benchmarked against manually annotated data. Section IV reports training as well as testing results and discusses their output. Subsequently, section V reveals the challenges of the existing model and states attempts that were conducted to solve them. Furthermore, this section includes an introspection of the latent space. Finally, section VI gives a conclusive statement, before section VII outlines possible approaches that may improve the performance of the proposed model.

II. Related work

This section outlines traditional existing cell detection algorithms that have been applied to one-photon calcium imaging sequences. In addition, two deep learning approaches are introduced.

A. PCA/ICA

Mukamel et al. [8] proposed an automated cell detection algorithm based on principal and independent component analysis (PCA/ICA). Thereby, PCA is utilized as a dimensionality reduction method that extracts linear transformations of the data. Each linear combination (Principle Component) yields to extract a maximum variance of the data. In order to reduce dimensionality, only linear combinations with the most variance are kept to represent the data. Others are identified as noise and are therefore eliminated [9]. Naturally, PCA alone is not suited to detect cells. This is based on the fact that fluorescence signals, as well as the time variation within the fluorescence signals of individual cells, tend to be of a similar extent. Since principal components rely on variance, they are rather representing signals from multiple cells instead of a collection of signals of an individual cell. Motivated by this, Mukamel et al. [8] include an Independent Component Analysis (ICA) subsequent to PCA for cell detection. ICA separates linearly combined signals into independent sources. This naturally fits the cell detection task since it refers to independent sets of spatial locations and temporal activation traces. The PCA/ICA algorithm highly depends on the number of principal components as well as the weighting of the two objective functions, spatial skewness, and temporal skewness. Tegtmeier et al. [6] implemented the PCA/ICA approach for automatic cell detection in their Calcium ActiVity Explorer (CAVE) tool for the analysis of one-photon calcium imaging data. The PCA/ICA algorithm detected 80% of manually extracted cells, whereas 20% of ROIs were missed. The false-positive rate is reported as approximately 8%.

B. Constraint non-negative matrix factorization

Maruyama et al. [10] proposed a cell detection algorithm based on non-negative matrix factorization (NMF). NMF decomposes a matrix into two-component matrices with non-negative values. In order to detect cells, Maruyama et al. added the bleaching line of the background fluorescence intensity as a constraint to the NMF. The background constraint dissociates the background components from the cells. The CNMF algorithm yields more accurate results than ICA at a low signal-to-noise ratio [10]. However, since CNMF is originally optimized for two-photon imaging data, it performs poorly on one-photon imaging data. This derives from background fluctuations that corrupt local neuron activations. Motivated by this, Zhou et al. [11] developed the extended CNMF-E framework for microendoscope data. CNMF-E relies on a more flexible background model that is able to handle background fluctuation. CNMF-E detected 200 out of 200 neurons, whereas PCA/ICA detected 195 [11]. Giovannucci et al. [7] provides a calcium imaging analysis toolbox, which includes the CNMF-E algorithm as an automatic cell detection for one-photon imaging data.

C. Deep learning-based approaches

Lu et al. . [13] proposed the Miniscope 1-Photon-Based Calcium Imaging Signal Extraction Pipeline (MIN1PIPE). MIN1PIPE aims to tackle the false-positive and false-negative issues occurring in PCA/ICA [8] [9] and CNMF [10] [11] [7] related methods. For this, MIN1PIPE incorporates recurrent neural networks (RNNs) with long-short term memory(LSTM) [17]. The MIN1PIPE pipeline was formed with the three core modules, neural enhancing module, movement correction module, and seeds-cleansed neural signal extraction module. Within the seeds-cleansed neural signal extraction module, the two-component Gaussian Mixture Model (GMM) algorithm is responsible for removing false positives in the background. Afterwards, the LSTM is implemented as an enhancement to remove further false-positive ROIs caused by abnormal background fluctuations. Soltanian-Zadeh et. al [12] proposed the Spatiotemporal NeuroNet (STNeuroNet) for neuron segmentation in two-photon calcium imaging. STNeuroNet was designed to alleviate the need for manually annotated ground truth via a neuronal segmentation task. A 3D convolutional neural network(CNN) is adapted in STNeuroNet due to the advantages of capturing temporal information into an end-to-end learning process. Despite the high computational cost of 3D CNNs, they have outperformed 2D CNNs in spatiotemporal segmentation tasks. The backbone network for STNeuroNet comprised of Dense V-Net, which is a combination of V-Net

[18] and DenseNet [19]. However, STNeuroNet is originally implemented for two-photon calcium imaging.

III. Methods

This section starts with a short description of the provided data before explaining the performed preprocessing steps. Afterwards, the model's architecture is introduced, followed by a detailed description of the input data and the basic building blocks: DeCoordConv, Encoder Network, Latent space, Decoder Network. Furthermore, the evaluation metrics is outlined to benchmark the proposed model to the manual annotation before an introspection method to get insights into the reasoning of the latent space is described.

A. Data

The utilized calcium imaging sequences were originally obtained by the Leibniz-Institut für Neurobiologie, Magdeburg. Information about the animals, surgeries, and tools used for the head-mounted calcium imaging is reported within [6] and is not further discussed within this work. Each imaging session is stored as a sequence of frames with a resolution of 630x630 and a frame rate of 5 frames/second. There are in total seven sessions with 756 (DG-13-3-08-17), 605 (DG-13-7-12-17), 643 (DG-13-8-11-17), 1886 (DG-13-7-16-17), 664 (DG-13-8-16-17), 658 (DG-13-8-18-17) and 629 (DG-13-8-08-17) frames. An exemplary frame extracted from one of the raw imaging sequences is illustrated in Figure 1.

This frame demonstrates the challenging low image contrast and resolution of one-photon calcium imaging data. In order to yield better results in the processing, the raw data is previously prepared. All preprocessing steps are discussed in the next section.

Along with the raw calcium imaging data, the Leibniz-Institut für Neurobiologie provided manually annotated data created by two neuro-scientist as a benchmark for the model performance. The manually annotated data includes a mask of all regions of interest (ROIs) over the whole sequence, as displayed in Figure 1. Furthermore, each ROI's activity pattern over the whole temporal sequence is provided, as well as an overview of the neuron activation per second.

B. Data preprocessing

As visible in Figure 2, the raw calcium imaging data is of low contrast and includes various artifacts. Static irregularities may be caused by dust, whereas the mouse's movement induces motion artifacts. In order to remove this, the raw data is preprocessed. All preprocessing is carried out within CAVE: An Open-Source MATLAB Tool for Combined Analysis of Head-Mounted Calcium Imaging and Behavior developed by Tegtmeier et al. [6]. Within the first step, each image sequence's contrast and brightness sequence is altered to reach a maximum contrast between ROI and background. Afterwards, static irregularities are removed manually. Next, an accumulated preprocessing step is applied that includes downsampling the calcium imaging sequence to 40% of the original resolution. Furthermore, completely black frames are substituted by previous frames, and a simplified flat field correction is applied to the whole sequence. Thereby, each image is divided by an average blurred image generated from the whole sequence. The result is multiplied with an average value of the whole sequence. This computation corrects the brightness gradient, which is darker at the edges and lighter in the center of an image. In order to entirely discard dark borders, only 80% of the center of each image is kept. Within the next step, the images are aligned to their previous frame for motion correction. For this, the subpixel registration algorithm [20] is applied as recommended by [6] for images without visible local features, such as blood vessels. The subpixel registration computes a discrete Fourier transformation matrix to cross-correlate two images in order to align them. The last preprocessing step includes a ΔF/F calculation to receive the fluorescence change over the baseline. The ΔF/F representation is computed as follows.

$$\Delta F/F = \frac{F_i - \overline{F}}{\overline{F}} \tag{1}$$

with F_i denoting an individual frame i and $\overline{F}$ the mean image over the whole sequence. The output of an exemplary preprocessed frame is represented in Figure 2.

After preprocessing, it is observed that the frames of each session vary in their brightness. Figure 3 gives an overview of the median pixel values per frame for each session. DG-13-8-08-17 is the session with the most extensive interquartile range and the broadest range between the whiskers. Therefore, this session holds a large range of values. Moreover, the mean value of all median pixel values per frame is larger than the median value. Hence, the session is likely to have many dark frames with multiple very bright frames that lift the mean value. DG-13-3-08-17, on the contrary, has a smaller interquartile range and range between whiskers. Furthermore, its mean value is similar to the median value. This session is likely to contain images with similar brightness.

C. Model Architecture

From a high-level perspective, the model architecture intends to move randomly initialized coordinates towards the position of potential neurons. Bright areas in the input data represent these candidate neurons. Based on this, the first objective revolves around enhancing the resolution of potential neuron activations within the input images. Afterwards, the coordinates are randomly initialized, and the model learns to move them towards the high-intensity areas of the input. For this, the model utilizes a mapping of the coordinates from vector space to a pixel representation as visualized in Figure 4. The model also learns how to adjust the intensity of these learnable coordinates in the pixel space corresponding to the brightness of the potential region of interest in the input data. *Similarity Scores* are introduced to capture the intensity of the coordinates over the whole sequence. The learnable coordinates, combined with their similarity scores, represent the *latent space*. Generally, the latent space is

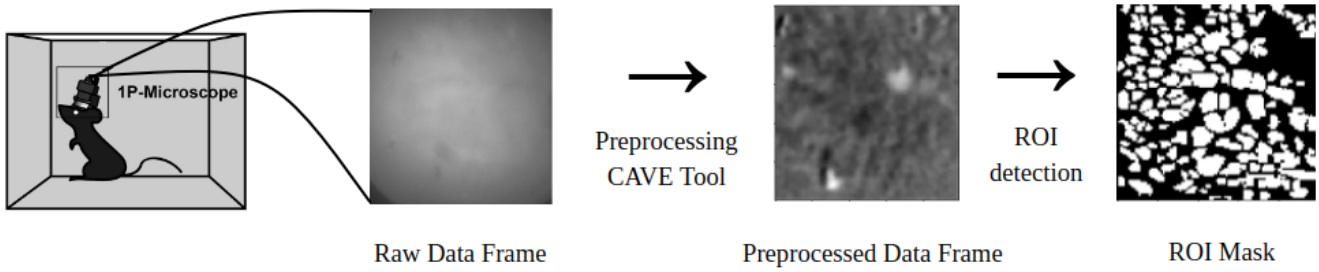


Figure 1: Calcium imaging analysis pipeline. From left to right: Simplified calcium imaging set up, raw data frame captured during calcium imaging, output after preprocessing in CAVE tool, ROI mask as output from CAVE ROI detection.

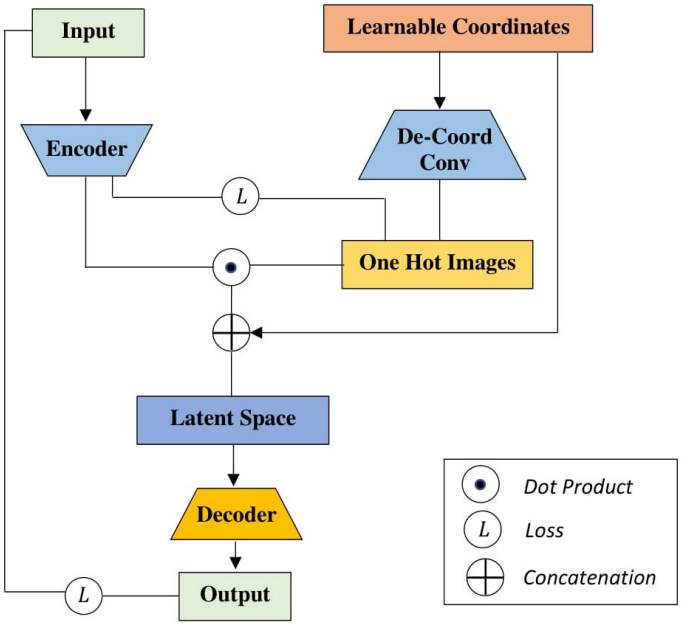


Figure 2: Complete pipeline

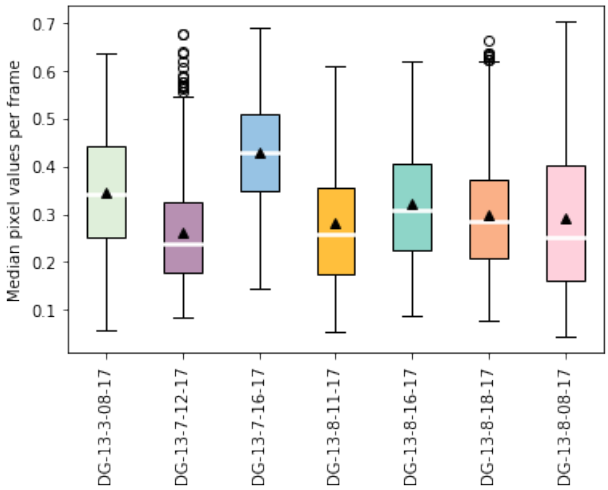


Figure 3: Boxplot of median pixel values per frame for each session. Median over the whole session marked by the white lines. Mean over the whole session marked by the black arrow.

Figure 4: Sparse visualization of learnable coordinates generated by pre-trained De-CoordConv network

composed of essential features from the input image. In order to validate the latent space, the input data is reconstructed only by the content of the latent. The model's architecture is represented in Figure 2. On the left-hand side, the model consists of an encoder network, aiming to enhance the input images' signal-to-noise ratio. On the right-hand side, there is pre-trained De-CoordConv network, which generates a sparse visualization of all learnable coordinates. The similarity scores are obtained at the bottleneck layer of this network by computing the dot product between the encoder output and the learnable coordinates' sparse visualization. The latent space is constructed by concatenating the learnable coordinates with their similarity scores. The decoder reconstructs the input image from the latent space.

After outlining the approach from a high-level perspective, the following sections are dedicated to describing the individual building blocks in detail.

D. Model Input

The pre-processed calcium imaging videos represent the input of the deep learning pipeline. Before that, all video files are converted to NumPy files and due to computational and memory constraints, the resolution of the frames is lowered to 100x100. Hence the dimension of each individual input image, including their binary color channel, is [100,100,1].

Correspondingly, a batch of frames with batch size B follows the dimension [B,100,100,1]. Furthermore, the deep learning pipeline utilizes a stack of N random coordinate pairs with the dimensions [N,1,1,2] as learnable coordinates. The last dimension holds one x-coordinate and one y-coordinate with x,y $\in$ [5..95]. The range of the random coordinates is intentionally chosen to be smaller than the image resolution to prevent the learnable coordinates from leaving the coordinate range of the image.

E. De-CoordConv

The De-CoordConv network is located on the right-hand side of the deep learning pipeline. Initially, it takes randomly created coordinates, as described in section III-D and creates a sparse visualization of these coordinates. Mapping coordinates from a cartesian space to a sparse, pixel-based representation is surprisingly not a trivial task. More precisely, regular CNN's fail within this simple coordinate transformation problem. An additional CoordConv layer, as proposed by [21] solves this problem by supplying the convolution operation with its input coordinates. For this, two extra coordinate channels are concatenated channel-wise to the input data. Coordinate channel i is an *height* $\times$ *width* matrix, where elements carry the number of their row. For coordinate channel j, each element carries its column count. Finally, coordinate channels are scaled so that i,j $\in [-1, 1]$. The CoordConv layer is an essential part of the De-CoordConv network as represented in Figure 5. Together with further convolutional layers and a final softmax layer, the De-CoordConv model is able to learn the classification task to draw a pixel representation of the given coordinates. Within this work, the De-CoordConv is pre-trained, and weights are frozen during the learning process. The output of the De-CoordConv model follows the dimension [N, IMAGE WIDTH, IMAGE HEIGHT,1]. Here, *IMAGE WIDTH* and *IMAGE HEIGHT* correlate with the input image's dimension. Such pixel representation of the coordinates is necessary to enable the coordinates to move to the bright regions of the encoder output. Since the coordinates are learning where to locate themselves, they are referred to as *Learnable Coordinates*. The learnable coordinates are the only learnable parameter of the De-CoordConv building block.

F. Encoder Network

The encoder building block follows the objective to increase the signal to noise ratio. More precisely, it raises the intensity of bright regions, while the background, often represented by black or less bright pixels, is darkened. Ideally, the encoder output is a sparse and binary representation of the input data. The architecture of the encoder is inspired by Denoising Autoencoders [22], as shown in Figure 6. Accordingly, there are 128 filters applied within the first layer of the network. For the following layers, fewer filters are applied than in the first layer. This approach prevents the model from learning the identity function. Furthermore, a CoordConv layer is included in the encoder network to keep a translation dependency and encourage translation invariance. Since the coordinate channels of the CoordConv layer are concatenated to the input channels, the model is able to neglect the coordinate channels, if they do not contribute to the learning tasks [21]. The Encoder network utilizes several methods to avoid border artifacts, as illustrated in Figure 7. First of all, external padding layers are introduced instead of valid or same padding. For the external padding layers, symmetric or reflective padding may be applied. Symmetric padding is a non-zero padding method, where the padded values are symmetric to pixels in the image. Reflective padding is also non-zero padding, where the padded values are reflections of other pixels in the image. Both methods achieve a similar encoder output within this work. The proposed model architecture utilizes symmetric padding. Additionally, the encoder network uses reducing blocks instead of max-pooling layers, as proposed by [23]. The architecture of the reducing block is illustrated in Figure 6. Accordingly, the input of the reducing block is padded and passed through two strided convolution layers with a stride of 2$\times$2, kernel size 3$\times$3, and 32 output filters. Afterwards, both strided convolution layers are concatenated to receive the double amount of output filters. This halves the image dimension. There are two reducing blocks utilized in the second and third layer of the encoder network. After applying two reducing layers, the image has to be upsampled to reach the original input image size again. Upsampling is performed by resizing layers as proposed by [24]. Resize layers avoid artifacts created by the standard convolutional upsampling layer. The resize layer utilizes bi-linear interpolation and a scaling factor of 2$\times$2. Since L1 regularization encourages sparsity [25], it is applied in all convolutional layers with a regularizing coefficient of 0.1. Subsequently, dropout layers [26] with a dropout rate of 0.4 are added at specific layers to regularize the Encoder network. The activation function applied in the encoder is the Rectified Linear Unit(ReLU) for all layers, except the output layer, which requires a Sigmoid activation function. The Sigmoid activation function leads to pixel values between 0 and 1.

G. Latent Space

The idea of the project is based on the Analysis by Synthesis approach [27]. The *Synthesis* element of this method carries condensed information of the input data, whereas the *Analysis* part gives insights into how well the compressed information of the Synthesis part, reflects the actual input data. As applied by [28]–[30], the Analysis of Synthesis approach may be used for 3D image reconstruction. Thereby, features of the input image such as depth of an observed object or coordinates of an object make up the Synthesis, while the reconstruction of the input image acts as the Analysis. The Analysis by Synthesis approach is an essential part of the proposed model architecture. Accordingly, the input image features are received by computing the dot product of the encoder output and the sparse visualization of the coordinates generated by De-CoordConv. The obtained features are concatenated with the coordinates to build the latent space and enable a reconstruction of the image. Within this work, the latent space includes learnable co-

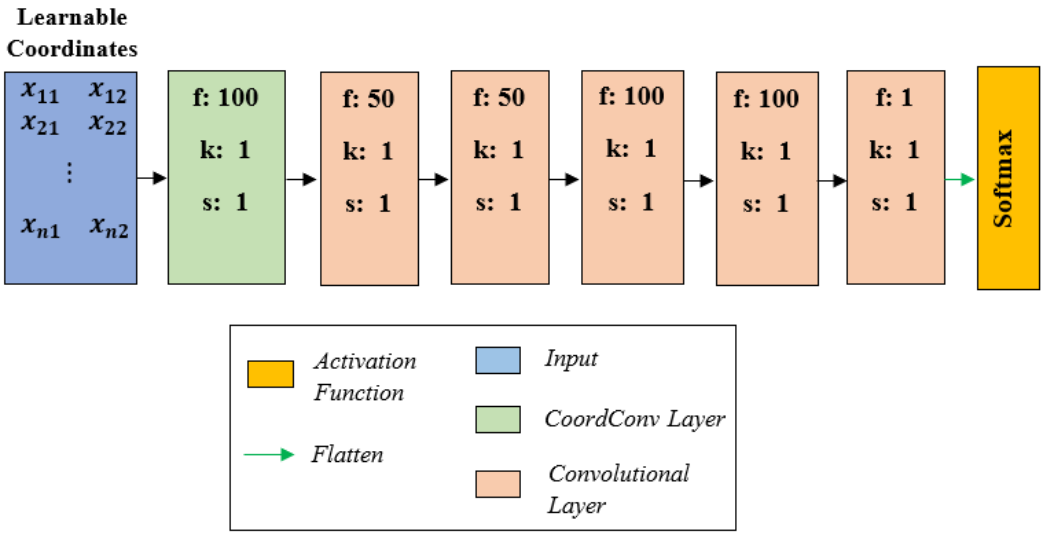


Figure 5: Model architecture for De-CoordConv with learnable coordinates as input, where *n* is the number of neurons defined by the user. Learnable coordinates are tiled to accustom the shape of a desired output before connecting to the next layer. The first layer consisted of CoordConv [21] layer, follow by multiple convolutional layers. The output from the last convolutional layer is flattened and transformed with softmax operation. All convolutional layers adopted ReLU as an activation function, except the last convolutional layer which yields a linear output.

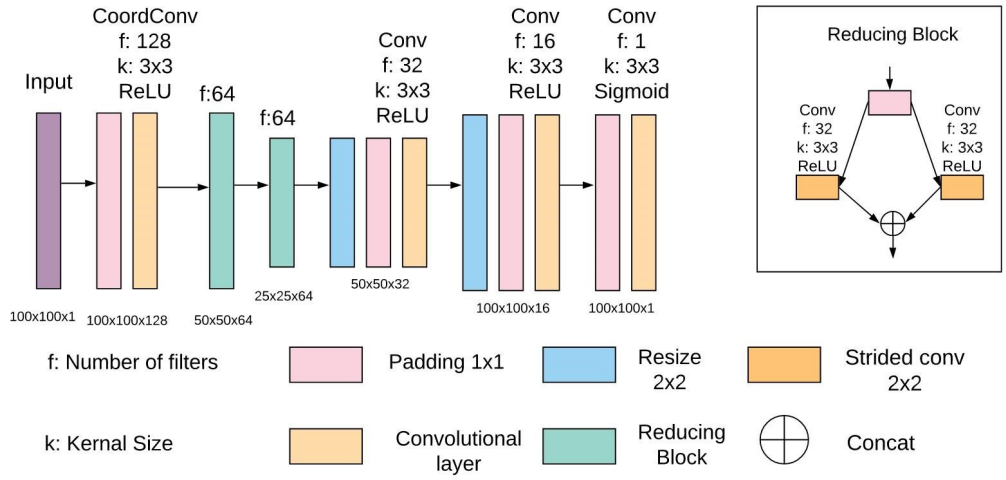


Figure 6: Encoder network to detect Region of Interest

ordinates. Inspired by the soft attention mechanism [14]–[16], learnable coordinates generate a query, indicating where they want to focus in the pixel space. The query is conducted via the transformation of De-CoordConv, where coordinates are mapped from a vector space to a pixel space forming a sparse visualization of the coordinates. The dot product of each bright pixel in the visualization of the coordinates with the encoder output produces a similarity score. The number of similarity scores $\in \mathbb{R}^N$, correlates with the hypothesized number of neurons, N. Each similarity score describes how well its query matches. From a high-level perspective, a similarity score acts as a metric that measures how well the intensity of a pixel position matches with the targeted ROI in the encoder output. Therefore, similarity scores act as a driving force to encourage the learnable coordinates to converge towards the bright ROIs. In the context of [14]–[16], similarity scores $\in [0, 1]$ can be referred to as attention coefficient, where high scores indicate a good match, whereas low scores stand for a lack of fit. The latent space is constructed by concatenating the learnable coordinates with similarity scores, where $LatentSpace = \{[x_1, y_1, sim_1], [x_2, y_2, sim_2]...[x_n, y_n, sim_n]\}$. Here, x and y correlates to the coordinates in the vector space, whereas, n stands for the number of neurons specified by the user.

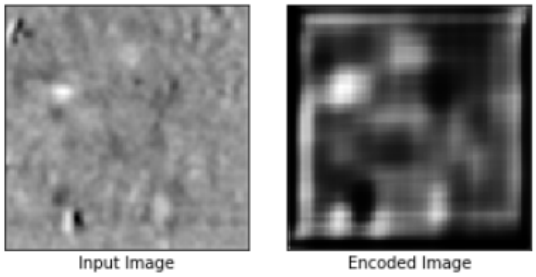


Figure 7: Exemplary input image and associated encoder output with border artifacts

H. Decoder Network

The decoder network reconstructs the input image from the latent space. Consequently, the quality of the reconstructed image represents how well the latent space encodes the input data. Figure 8 illustrates a schematic plot of the decoder model architecture with *f: number of filters, k: kernel size, s: strides*.

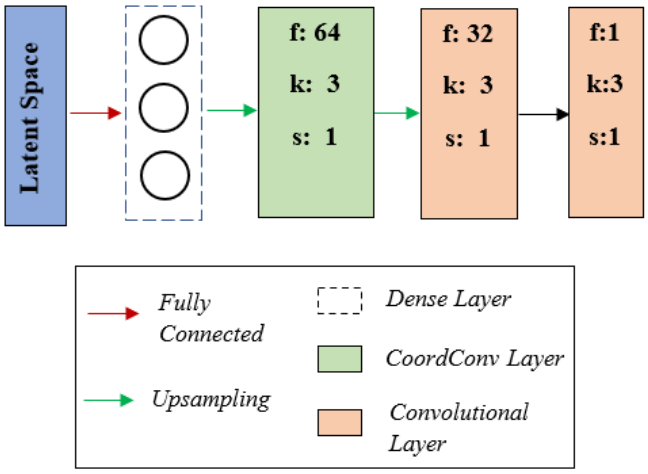


Figure 8: Model architecture of decoder

In this work, the convolution after upsampling method is applied as a deconvolution method. Layers are upsampled with a scale factor of 2 and bilinear interpolation. The decoder network connects the latent space with a fully connected dense layer. This layer is afterwards reshaped into a higher dimension to match the dimensions of the next deconvolution layer. The first deconvolutional layer is a Coordconv layer to keep the translation dependency in the feature maps. The feature maps are then passed to the next deconvolutional layer with standard convolution. The last layer consists of a convolutional layer with a filter number of 1 without upsampling to match the input images' dimensions. The output of the decoder network is the reconstruction of the input image. The activation function for all the layers is ReLU, other than the output layer, where a linear output is desired.

I. Loss

Two losses are implemented in the network pipeline, namely *reconstruction loss* and *sparsity loss*. As depicted in Figure 2, the reconstruction loss is the Mean Squared Error (MSE) between the decoder output and the input image. Minimizing the MSE loss encourages the decoder network to learn a reasonable reconstruction of input from the latent space. The MSE loss is also known as global loss since the backpropagation of MSE loss flows throughout the complete pipeline to update learnable parameters. However, the reconstruction loss is not sufficient for the encoder network to learn a sparse output. In order to encourage the sparsity of the encoder network output, a further binary cross-entropy loss is computed. The binary cross-entropy loss is calculated between the encoder output and the sparse visualization of the learnable coordinates, multiplied with the similarity scores. The sparsity of the visualization of the learnable coordinates (Figure 4), multiplied with the similarity scores, naturally encourages the encoder to create a sparser output. The sparsity loss is only backpropagated through the encoder network and the learnable coordinates.

J. Optimizer

The Adam optimizer [31] is adapted for both losses. The learning rate for the global loss is 0.00001. Recall that the global loss flows through the whole pipeline. The sparsity loss is implemented with a learning rate of 0.000001 and is only backpropagated through the Encoder network and the learnable coordinates. In order to get the coordinates to move, it is necessary to pass the global loss again through the learnable coordinates with a higher learning rate of 0.1. Table I summarizes the losses with their associated learning rates.

Loss	Encoder	Learnable Coordinates	Decoder
Global loss	0.00001	0.00001	0.00001
Global loss		0.1	
Sparsity loss	0.000001	0.000001	

Table I: Overview of losses and associated learning rates. The global loss is backpropagated twice through the learnable coordinates, once with a higher learning rate to enable the coordinates to move.

The learnable coordinates are updated via gradient descent as stated in equation 2, with *coords: learnable coordinates, α: learning rate, Loss: Reconstruction Loss*.

$$coords \leftarrow coords + \alpha \cdot \frac{\partial(Loss)}{\partial(coords)} \tag{2}$$

The driving force to move coordinates in the pixel space is to update the learnable coordinates in the cartesian space within the range of two digits. Updates of the coordinates in a smaller extent are not sufficient for the model since it does not represent such small updates in the image generation. Therefore, a low learning rate may lead to dead pixels that are, from a visual perspective, not updated. Consequently, a high learning rate for the global loss with respect to the learnable coordinates is crucial to encourage the model to learn the

ROIs' positions. Figure 9 depicts how the sparse visualization of the coordinates converge towards the ROIs from the output of the encoder network after applying a high learning rate.

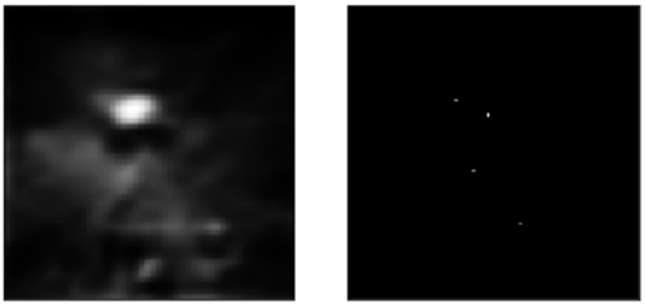


Figure 9: Encoder output *(Left)* and associated sparse visualization of learnable coordinates with an applied threshold of 0.4 *(Right)* to show most intense candidates.

K. Evaluation metrics

In order to evaluate the performance of the model, its prediction is benchmarked against a manually annotated ROI mask. To quantify the performance, it is necessary to provide a suitable metric. This metric has the objective to capture all coordinates that are located within an ROI marked in the manually annotated neuron mask. Naturally, this value depicts the number of True Positives. Furthermore, the number of coordinates that are incorrectly located has to be captured. Incorrect locations refer to areas not marked as ROIs in the manually annotated neuron mask. These displaced coordinates are the False Positives. Finally, the precision metric captures the ratio of both by computing the following.

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

with TP and FP as the number of True Positives and False Positives. Furthermore, a visualization of the coordinate predictions plotted on top of the annotated neuron mask is provided to get better insights into the coordinates' behavior.

L. Introspection of Latent space

A common problem with the application of deep learning methods is their missing transparency and traceability. In order to gain insights into the latent space's reasoning, an introspection is performed on the latent space. Thereby, the latent space is analyzed by generating image reconstruction from different variants of the latent space. This may give insights, whether all coordinates in the latent space are of the same importance. For the introspection, the latent space (dimension: [n, 3]) is reduced by half (dimension: $[\frac{n}{2}, 3]$), where *n:number of neurons*. The number of neurons is defined as 100 within this work. The reduced latent space is composed of different coordinates and similarity scores from the unaltered latent space. For instance, 25 coordinates are removed from the latent space's head and tail before reconstructing the image. In further test cases, 50 coordinates are removed from the head, middle, or tail of the latent space. A final reduced version of the latent space is created by removing 50 coordinates randomly. The second set of test cases revolves around the ordering of the latent space. Note that the coordinates in the latent space are in random order due to the random initialization. Thereby, it is observed whether the coordinates' order plays a significant role within the latent space. For this, all coordinates in the latent space are sorted by their Cartesian coordinates. More precisely, the x-coordinate defines the order of the coordinates in ascending order. Note that the affiliation of the coordinate pairs and their associated similarity score is kept during ranking. The same set of reduction methods is then applied to the ranked version of the latent space to examine the order's relevance in the latent space.

IV. Results

This section refers to the results of this work. Firstly, the final training setup of the proposed network is described before an exemplary output for each pipeline step for the training sessions is shown. Afterwards, IV-A2 reports the final prediction outcome of the model for each training session. Finally, the pipeline output and predictions for an unseen test session is included in this section.

A. Training

The complete network is trained on six calcium imaging sessions. Due to insufficient computational power, one of the provided sessions is excluded from training. It consists of 1886 frames, which is approximately three times the number of frames compared to the other sessions. The excluded session in the training set is later used as a test set. The network is trained for 1000 epochs, where one epoch goes through all six sessions. The user-defined batch size is 8, and the number of neurons, which is the number of learnable coordinates, is defined as 100. The image frames are not shuffled, due to temporal dependencies to previous frames.

1) Pipeline output: Figure 20 depicts the output of the model at each pipeline step. Each row represents exemplary outputs for different sessions with different input image intensities. The first column is the input image from different sessions. The second column is the output from the encoder output, which aims to enhance the intensity of the regions of interest. The third column represents the sparse visualization of the learnable coordinates generated by the De-CoordConv model. Each coordinate pixel is multiplied with the correlated similarity scores and summed along the channel axis. The visualization of the learnable coordinates is thresholded with a value of 0.3 to only keep the brightest neuron positions. The last column shows the reconstructed output by the decoder network. It is expected to represent the input image in case of a reasonable latent space. Figure 10 shows the loss curve for the sparsity loss and the reconstruction loss. The sparsity loss shows a decreasing trend throughout the epochs. The slight increment of loss between 200 and 400 epochs might be caused by the encoder output that fails to learn a sparse

representation. In general, the sparsity loss is able to promote the sparsity of an encoder output throughout the session, as shown in Figure 20. This applies especially for a dark image background, as represented in 20b. The Reconstruction loss shows a steep decrease in the loss value from the beginning and level off with small decrements of loss throughout the sessions. Such a trend of reconstruction loss can be validated by observing the reconstructed image in the last column of Figure 20. It indicates a reasonable reconstruction of an input image by the decoder from the latent space.

2) Predictions: Figure 11 shows the predicted neuron positions of session DG-13-8-16-17 marked in red, plotted onto the manually annotated neuron mask in white. For session DG-13-8-16-17, 146 coordinates were placed in correct ROI positions, while 126 were placed in areas that were not labeled in the manual annotation. Results of all sessions are collected in Figure 21. It is visible that most coordinates gather in small clusters, often positioned around the outer circle of the manually annotated ROI. This indicates that the model successfully learned how to move the coordinates. A few coordinates tend to locate themselves at the edges of the image. This is caused by coordinate updates that pass the boundaries of the image. In order to keep these coordinates in the image, they are clipped to retain an acceptable range. This may cause the appearance of bright coordinate pixels on the edges of the image. In every session, the model predicts multiple ROIs that are not labeled in the manual annotation. To quantify this observation, Table II gives an overview of the true positives, false positives as well as the precision for all sessions. According to the metrics, the model performs poorly on all six sessions. The overall best performance is reached on session DG-13-8-16-17 with a precision of 0.54. However, the precision of the initial coordinate initialization is reported as 0.53. Hence, the model only increased precision by 0.01. For session DG-13-7-12-17, the model only achieved a precision of 0.3 but raised the initial precision by 0.1. Moreover, the masked annotation of this session captures significantly smaller areas than the other masks. This is demonstrated in 21b. In session DG-13-8-08-17, the model yields slightly worse performance than the precision achieved by the random coordinates. Note that in Figure 3, the same session shows a significant difference between the median and mean of the median pixel values per frame. This indicates that most frames of this session have a median pixel in the darker range, but some outlier-frames have a much higher median pixel value. The model with the best precision improvement, on the contrary, has a similar mean and median value. Therefore, the poor performance may be caused by overexposed frames indicated by a larger difference between mean and median value. However, partially, the model's poor precision is caused by coordinates located close around the manually annotated neuron positions. These coordinates are counted as false-positive, but may actually be accepted as a neuron position by an expert. Nevertheless, the coordinates have the objective to gather in the center of the neuron instead of the outer edges. Finally, it is important to note that the neuron mask is not guaranteed to be correct since the manual annotation is not a trivial task. Therefore, the model might predict correct neuron positions that were not recognized by the labeler.

Session	TP	FP	Precision	Improvement
DG-13-3-08-17	192	185	0.51	0.14
DG-13-7-12-17	82	194	0.30	0.10
DG-13-8-11-17	147	149	0.50	0.01
DG-13-8-16-17	146	126	0.54	0.01
DG-13-8-08-17	110	151	0.42	-0.01
DG-13-8-18-17	122	119	0.51	0.01

Table II: True Positives (TP), False Positives (FP), Precision and improvement towards initial precision of random coordinates for each training session. Highest values marked in green.

B. Testing

As observed in section IV-A2, the model does not perform well for the training data. Nevertheless, it is important to test the model performance on unseen data in order to recognize, for instance, overfitting. The largest dataset excluded during training and is used as a test set.

1) Pipeline output: Figure 12 shows exemplary pipeline outputs for dark and bright images of the test set. It is observed that the encoder output is comparable to the training data output, and the learnable coordinates are moving towards the bright areas in the encoder output. However, it is clearly visible that the reconstruction of the input image does not keep up with the reconstructed images of the training sequences. Especially, the reconstruction of the dark image shows little to no resemblance to the input image. One possible reasoning for such behavior may be that the decoder is overfitting. Overfitting is indicated by a good training performance and much worse test performance. In addition, as visualized in Figure 3, the test session DG-13-7-16-17 has a significantly varying pixel range compared to all other training sessions. This may have an influence on the testing performance of the decoder and indicates that the decoder is not generalizing well.

Predictions The predictions for the test session DG-13-7-16-17 are illustrated in Figure 13. The predicted ROI positions are plotted in red on top of the manually annotated ROI mask, represented in white. As in the training set, it is observed that the coordinates are moving to create clusters. However, its coordinates tend to cluster up in line-shapes instead of circled shapes. The number of True Positives is 404, and the number of False Positives is 962. The significant higher numbers derive from the higher amount of frames in this session. The test precision is 0.3, which increases the initial precision by 0.03. Considering the precision, the test session's predictions are significantly worse than for the majority of the training sessions, as reported in Table II.

V. Discussion

This section includes a detailed discussion of the model and its predictions. The architecture is analyzed by determining

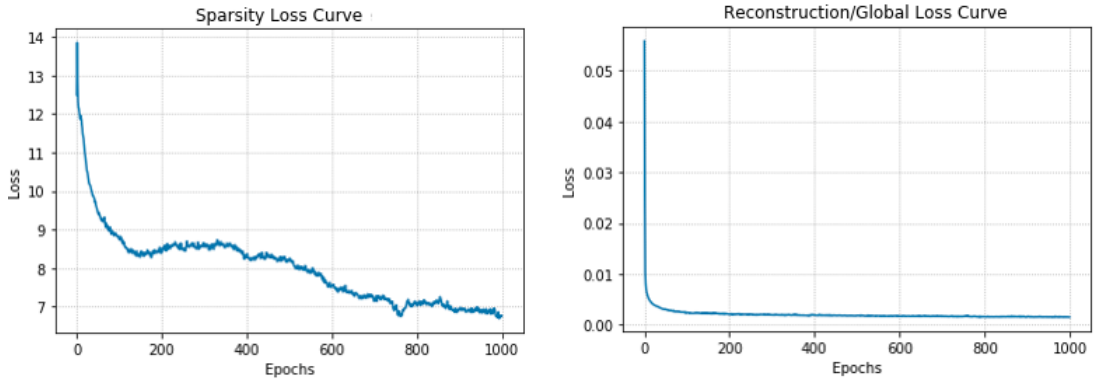


Figure 10: Loss plot for Sparsity Loss *(Left)* and Reconstruction/Global Loss *(Right)*

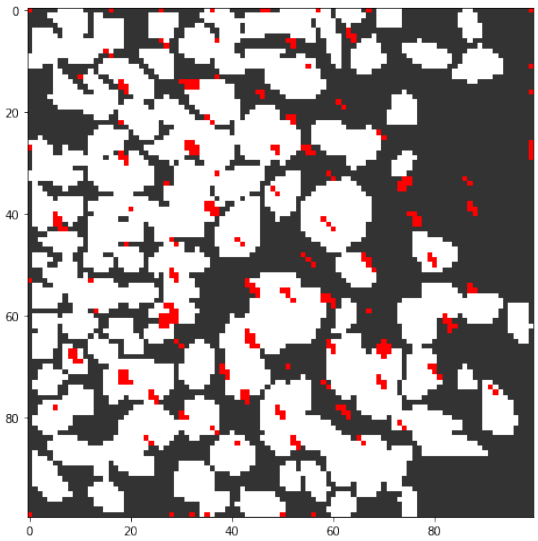
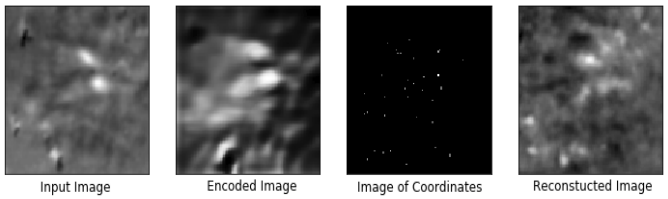
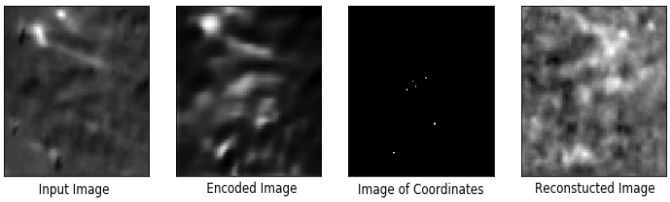


Figure 11: Result of session DG-13-8-16-17. Predicted neuron positions marked in red. Manual annotation illustrated in white.



(a) Exemplary bright frame from validation data



(b) Exemplary dark frame from validation data

Figure 12: Pipeline output for validation data

building blocks that require improvement. For building blocks with the need for improvement, already attempted approaches to improve the performance are summarized and discussed. An introspection of the latent space is the final topic covered within this section.

A. Sparsity and Binarization of the encoder output

Ideally, the output of the encoder is a highly sparse binary representation. Several methods are applied to encourage the sparsity of the encoder output within this work: A sparsity loss function, L1 regularization as well as several dropout layers are added to the encoder network. Nevertheless, the encoder output did not yield the desired level of sparsity. Figure 14 gives an overview of the pixel values in exemplary input images and their associated encoder output. Naturally, there is a large difference in the mean and a varying range of values within the dark, less bright, and bright input images. Ideally, the encoder output should show a median value lower than the mean value. This indicates that most values are in the lower pixel range, with a few very bright outliers. This behavior is only visible for the encoder output of the less bright and the dark image. The larger quartile range of the less bright image indicates that the output is less binary than for the dark image. Generally, the overview shows that the encoder is able to change the pixel values of the input images. The encoder output is sparser for the dark input images than for the brighter ones. In any case, the encoder network needs to be improved to create a sparser output. Within the course of this work, the sparsity loss function was exchanged with a so-called black-and-white-loss (b/w-loss) to improve the encoder's sparsity. The b/w loss function compares the encoder output to one black image and one white image. The squared distance for each pixel to both images is computed, and the lowest distance is kept. Finally, the mean of the distances is calculated and returned as a loss. Naturally, this loss function is zero for binary images, whereas grey images lead to a high loss. Figure 15 shows the pipeline output of a bright and a dark frame for an applied b/w-loss. Accordingly, the b/w-loss enables good

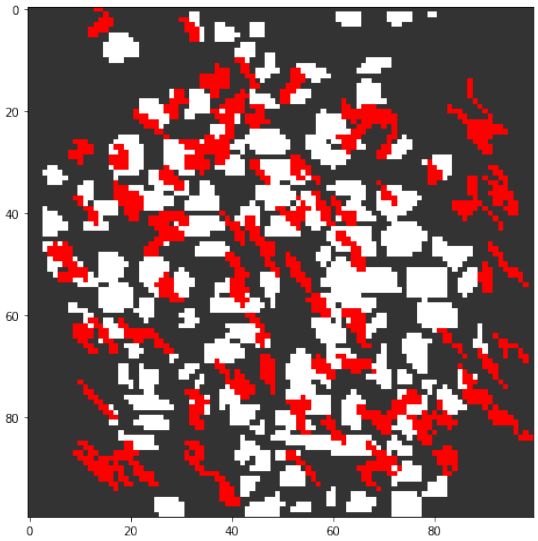


Figure 13: Testing Prediction for test session DG-13-7-16-17. Predicted neuron positions marked in red. Manual annotation illustrated in white.

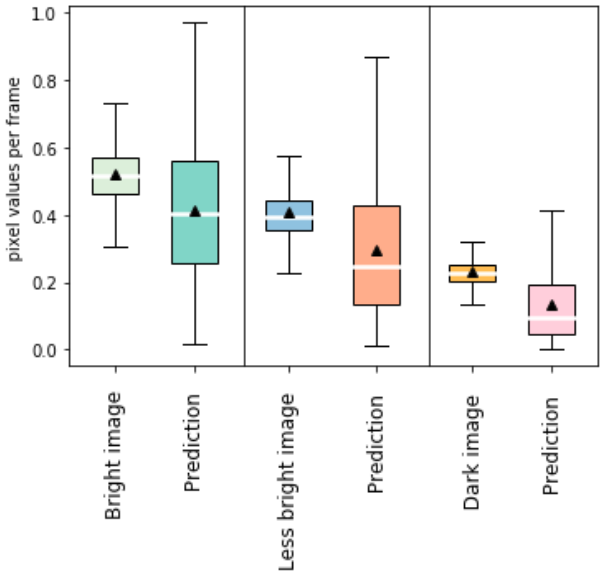
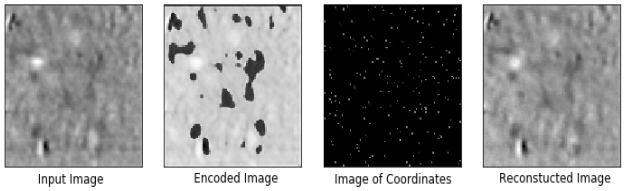
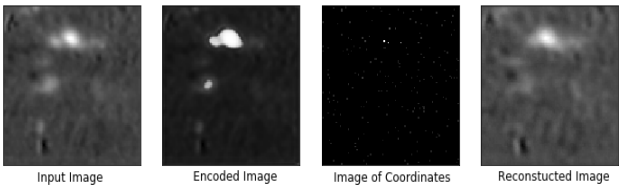


Figure 14: Boxplots of pixel value range for different input images and corresponding predictions.

results on dark images, but poor results for white images. In order to overcome the challenge of the bright images, an extra weight factor was added to the distance towards the white image to steer the model towards a dark representation. However, results on the bright images did not significantly improve. In any case, the encoder output did not yield a sparse output. In another attempt, a Gaussian blur was applied to the encoder output and the visualization of learnable coordinates before passing them through the loss function. An exemplary output for a dark image is represented in Figure 16. This attempt also fails to create a sparse encoder output.



(a) Exemplary dark frame



(b) Exemplary bright frame

Figure 15: Pipeline output of a bright and a dark frame

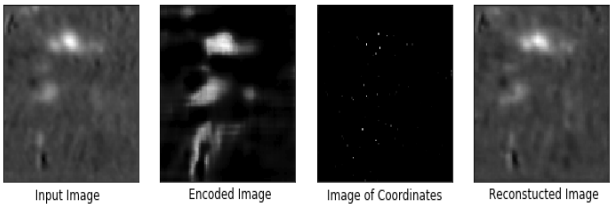
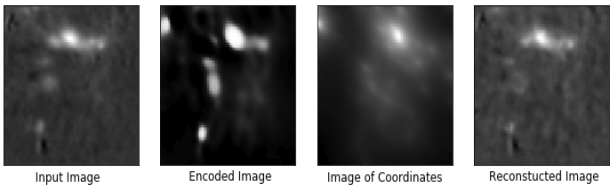


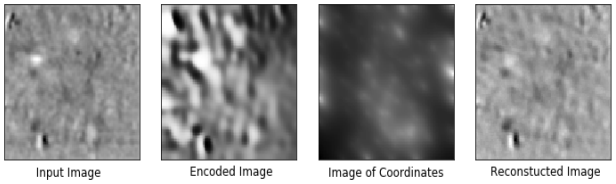
Figure 16: Pipeline output for the attempt to apply a Gaussian blur on the encoder output and the learnable coordinates before computing the loss.

B. Coordinates located at ROI edges

As reported in section IV-A2, the coordinates tend to located at the edges of the ROI positions. In order to overcome this problem, a differently pretrained De-CoordConv model is used. The new model consists of one CoordConv layer, one Conv2D layer, and a final Softmax layer. This network is more shallow than the originally trained DeCoordConv network and generates a smeared representation of the coordinates. The smeared representation has a bright spot in the coordinate location, which may steer the model to place coordinates in the brighter center of the ROI instead of the lighter outer edges. Furthermore, the smeared coordinates may match better with the relatively dense output of the encoder. The pipeline output for utilizing the smearing DeCoordConv is represented in Figure 17. The encoder output for the dark image includes most activations of the input image and displays them with higher intensity. Additional high-intensity areas are added by the encoder that is not apparent in the input image (upper left corner). The coordinates seem to be located around the largest ROI area and located around the slightly greyish areas in the encoder image. For the bright frame, the encoder output includes grey areas that are leveraged by the coordinates. Since the representation of the smeared coordinates does not fit to the utilized plotting method of the coordinates, it was not possible to plot the smeared coordinates on to the manually annotated ROI mask.



(a) Exemplary dark frame

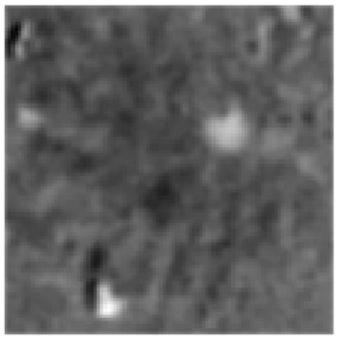


(b) Exemplary bright frame

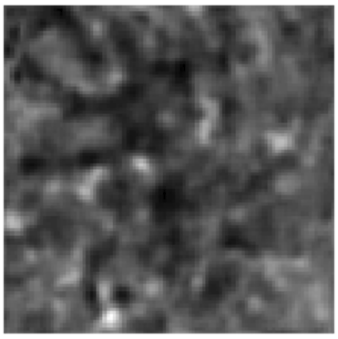
Figure 17: Pipeline output for utilizing a shallow DeCoord-Conv that generates smeared coordinates

C. Introspection of a latent space

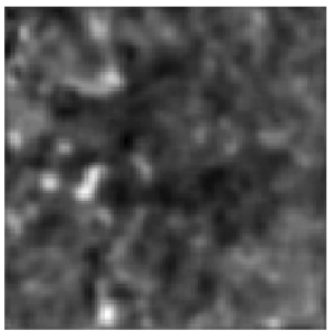
In order to get insights into the reasoning of the latent space, introspection is performed, as described in section III-L. Accordingly, the representation of the latent space (dimension: [n, 3]) is reduced by half (dimension: [$\frac{n}{2}$, 3]) with *n:number of neurons*, in this work defined as 100. Each reconstructed image is then compared to the reconstructed image of the entire latent space. The reduced latent spaces are generated by removing 25 coordinates from the head and tail, removing 50 coordinates from the top, middle, or bottom of the latent space or by removing 50 random coordinates. Figure 18 displays the input image, the reconstructions of the entire latent space, and the five reduced latent spaces. The top row of the image shows that the generated image from the original latent space varies significantly from the input image. However, the decoder manages to capture some important features from the latent space, such as the bright spot at the bottom left of the input image. By reducing the latent space's dimensions, it is observed that the additional latent representation loss yields random noise in the reconstruction. In all five cases, the reconstruction of the images varies between the different cases. However, the reconstruction of figures 19c 19d and 19e for instance looks more similar to the original reconstructed image than in figures 19f and 19g. This indicates that the coordinates at the top or bottom of the latent space are less important than coordinates located in the center of the latent space. Furthermore, it can be observed that the most apparent features of the reconstruction output, such as the high-intensity area at the bottom is captured in all five reconstructions. This indicates that more apparent features are captured within a large number of inputs within the latent space. In the second part of the introspection of the latent space, it is examined whether the order of the latent spaces places a significant role. As described in section III-L, all coordinates are ordered by their x-coordinate in ascending order. Afterwards, the same reduction methods as before are applied to the ranked latent space. Figure 19 displays the reconstructed images in comparison to the original unordered latent space. The first row of the image illustrates the immense importance of the order within the latent space. The same set of coordinates with their similarity scores, ordered differently, yields a completely different reconstruction than the unordered version of the latent space. The output of the test cases reassures this behavior. None of the reduced latent spaces seem to create a similar reconstruction output. Furthermore, this rebuts the assumption that coordinates located on the edges of an image may have less effect on the latent space since the latent space emphasizes the order in the latent space itself instead of the actual location in the Cartesian space. All in all, based on these test cases, it can be concluded that the latent space is composed of entries of varying importance. It seems like entries in the middle of the latent space are more important than entries at the head or tail. Besides that, the latent space carries meaning in the order of the entries. The Cartesian locations of the coordinates do not seem to be meaningful for the latent space.



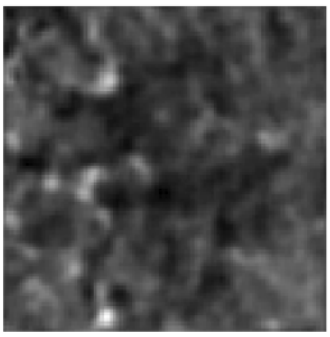
(a) Exemplary input frame of test set



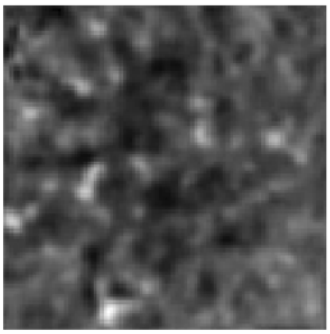
(b) Reconstruction with complete latent space



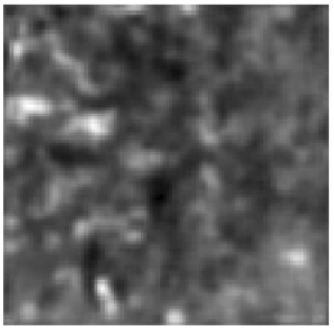
(c) Removing 25 coordinates from head and tail



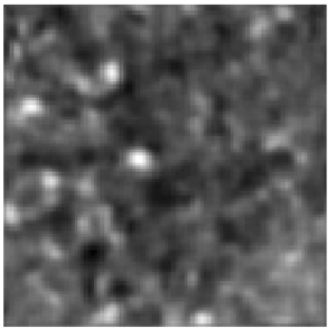
(d) Removing 50 coordinates from the top



(e) Removing 50 coordinates from the bottom

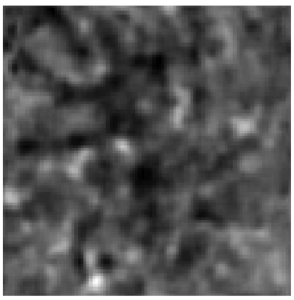


(f) Removing 50 coordinates from the middle

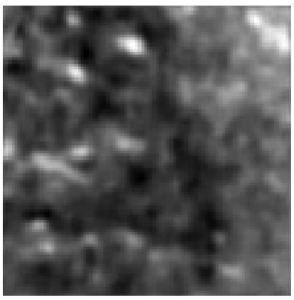


(g) Removing 50 random coordinates

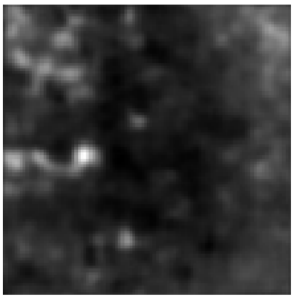
Figure 18: Comparison of input image, reconstructed image with entire latent space and five test cases for the dimensionality reduction of the latent space.



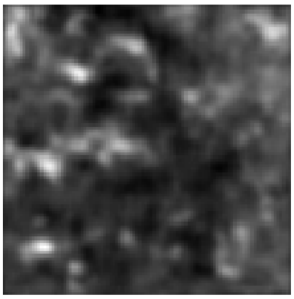
(a) Reconstruction with complete latent space



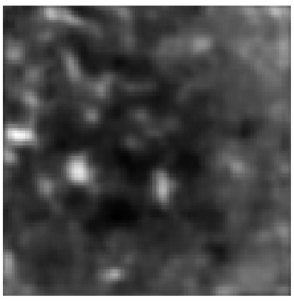
(b) Reconstruction with ranked latent space



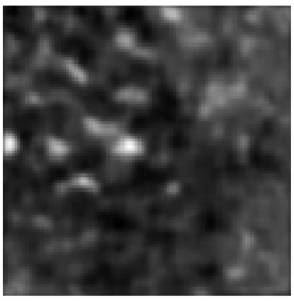
(c) Removing 25 coordinates from head and tail (Ranked latent space)



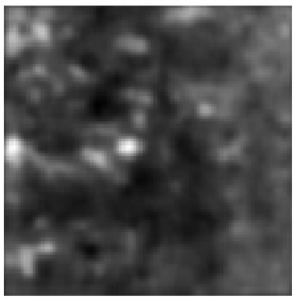
(d) Removing 50 coordinates from the top (Ranked latent space)



(e) Removing 50 coordinates from the bottom (Ranked latent space)



(f) Removing 50 coordinates from the middle (Ranked latent space)



(g) Removing 50 random coordinates (Ranked latent space)

Figure 19: Comparison of reconstructed image with entire latent space, reconstructed image from ranked latent space and five test cases for the dimensionality reduction of the ranked latent space.

VI. Conclusion

This section includes a conclusive statement about the work conducted within this project.

In this paper, a novel deep learning framework for the data analysis of single-photon calcium imaging is proposed. The approach is an unsupervised learning framework including a convolutional layer variant, CoordConv [21], similarity scores inspired by attention mechanism [14]–[16] and analysis-by-synthesis [27]. The approach aims to provide an end-to-end learning method to detect neurons within single-photon calcium imaging sequences. Ideally, the model should outperform conventional automated cell detection methods, such as PCA/ICA and CNMF-E, to lower the necessary manual correction work of these methods. However, unsupervised learning is not a trivial task, and the proposed framework does not yield the desired performance. The encoder network is unable to produce a sufficiently sparse output of the input images, which is necessary to steer the learnable coordinates to the bright areas of the ROIs. This applies especially to bright input images. Despite the encoder's performance, the learnable coordinates learned to move towards the bright areas in the encoder output. Thereby, the coordinates must update their cartesian values to such an extent, that the De-CoordConv model recognizes the difference when mapping the coordinates from a vector space to the pixel space. Applying a high learning rate ensures a sufficient update of the coordinates. Generally, the De-CoordConv network with its CoordConv layer as proposed by [21] performs well and is able to map the vector space to the pixel space. The decoder pipeline can generate a reasonable reconstruction of the input during training, which implies that the latent space, consisting of the learnable coordinates and their similarity scores, represents the features of the training input images. An introspection of the latent space reveals that coordinates carry varying importance within the latent space. Furthermore, the order of coordinates plays an important role. An ordering of the coordinates within the latent space leads to a significantly different reconstruction image. Despite the great performance of the decoder on the training set, it does not perform well for the unseen test dataset. This indicates that the decoder is overfitting. The performance of the model benchmarked against manually annotated ROI masks reaches a mean precision of 0.46 on the training data. Note that the manually annotated ROI mask may include incorrect neuron positions since the annotation is a tedious, non-trivial task. Different human experts often yield varying results on the same dataset. However, especially in the development process of a framework, it serves as a useful benchmark. All in all, the proposed end-to-end unsupervised deep learning framework proves that the general idea to initialize random coordinates and move them towards ROIs by utilizing analysis-by-synthesis [27] and the attention mechanism [14]–[16] shows potential. However, especially the encoder network offers great room for improvement.

VII. Future Work

This section gives an outlook on the possible future of this research area and provides suggestions on how this framework may be enhanced to achieve more competitive results.

A. State of the Art Framework

A novel deep learning framework has been proposed by [32], named RepNet, to count repetitions of essential features along a temporal sequence in videos. This network may be beneficial to count the repetition of particular ROIs. Final ROIs may be retrieved by applying a threshold on the number of counts throughout the sessions. An ROI mask could be created at that particular region if the number of counts at that position surpassed the defined threshold.

B. Pretrained De-CoordConv Network

Initially, De-CoordConv is trained to plot a sparse visualization of the learnable coordinates. During this work, a different DeCoordConv architecture is utilized to steer the coordinates towards the center of the ROIs. This De-CoordConv version plots smeared visualizations of the coordinates. However, this way, coordinates tend to locate also in grey areas of the encoder output. Based on this, it may be beneficial to train the De-CoordConv with the Not-so-Clevr dataset provided by [21]. This dataset enables the model to learn how to draw a square, given the coordinates of the center of a square. Since all squares are of high intensity, the coordinates may not locate at greyish areas of the encoder output. Furthermore, due to the larger size, coordinates may locate in the center of the ROI than the outer edges.

C. Overfitting of the Decoder

In order to solve the overfitting problem of the decoder as observed in IV-B1, it is suggested to add regularization [33] and dropout layers [26] to the decoder network. Furthermore, it can be suggested to include a larger range of varying imaging sessions in the training set to increase the generalization capabilities of the decoder. Visualizations as represented in Figure 3 may assist to find differing sessions.

D. Creating Anti-aliasing

As suggested by [34], it may be beneficial to apply a Gaussian blur before utilizing max-pooling, strided convolution, and average pooling. This enables the model to become shift-invariant and improves the generalization capabilities of the network. This method is referred to as Blurpool. By selecting suitable anti-aliasing kernels to different layers, the performance can be improved.

E. Benchmarking the performance

Once the model reaches a competitive performance, it is suggested to further benchmark it against the performance reached by CNMF-E. Giovannucci et al. [7] proposed the open-source tool CaImAn, which includes CNMF-E as an automated cell detection algorithm. This tool may be helpful to run the algorithm on the provided calcium imaging sequences.

F. Latent Space Introspection

In order to reassure results of the latent space introspection within this work, it is suggested to research more advanced methods that may be applicable for the proposed framework.

VIII. Acknowledgement

Our team would like to express our gratitude to Prof. Dr.-Ing. Sebastian Stober, the head of Artificial Intelligence Lab research group, Department of Computer Science, Otto von Guericke University Magdeburg, for providing insightful guidance in relavance to the field of deep learning in our project. Furthermore, our team is grateful for gaining access to the computational server equipped with two Nvidia 1080ti GPUs. Our team would also like to thank Dr. Michael Lippert and M.Sc.Vivekanandhan Viswanthan from Leibniz Instituite for Neurobiology(LIN) Magdeburg for providing us with the datasets as well as valuable knowledge regarding to neuroscience.

IX. Dependencies

Our work is written in Python3 with Tensorflow verion 2.0 framework. The code can be found in the following GitHub source: *github.com/twpkevin06222/DE_Project*

References

[1] P. Baker, A. Hodgkin, and E. Ridgway, "Depolarization and calcium entry in squid giant axons," *The Journal of physiology*, vol. 218, no. 3, pp. 709–755, 1971.

[2] R. Kerr, V. Lev-Ram, G. Baird, P. Vincent, R. Y. Tsien, and W. R. Schafer, "Optical imaging of calcium transients in neurons and pharyngeal muscle of c. elegans," *Neuron*, vol. 26, no. 3, pp. 583–594, 2000.

[3] T. Francis, R. Chandra, A. Gaynor, P. Konkalmatt, S. Metzbower, B. Evans, M. Engeln, T. Blanpied, and M. Lobo, "Molecular basis of dendritic atrophy and activity in stress susceptibility," *Molecular psychiatry*, vol. 22, no. 11, p. 1512, 2017.

[4] K. K. Ghosh, L. D. Burns, E. D. Cocker, A. Nimmerjahn, Y. Ziv, A. El Gamal, and M. J. Schnitzer, "Miniaturized integration of a fluorescence microscope," *Nature methods*, vol. 8, no. 10, p. 871, 2011.

[5] D. J. Cai, D. Aharoni, T. Shuman, J. Shobe, J. Biane, W. Song, B. Wei, M. Veshkini, M. La-Vu, J. Lou *et al.*, "A shared neural ensemble links distinct contextual memories encoded close in time," *Nature*, vol. 534, no. 7605, p. 115, 2016.

[6] J. Tegtmeier, M. Brosch, K. Janitzky, H.-J. Heinze, F. W. Ohl, and M. T. Lippert, "Cave: An open-source tool for combined analysis of head-mounted calcium imaging and behavior in matlab," *Frontiers in neuroscience*, vol. 12, p. 958, 2018.

[7] A. Giovannucci, J. Friedrich, P. Gunn, J. Kalfon, B. L. Brown, S. A. Koay, J. Taxidis, F. Najafi, J. L. Gauthier, P. Zhou *et al.*, "Caiman an open source tool for scalable calcium imaging data analysis," *Elife*, vol. 8, p. e38173, 2019.

[8] E. A. Mukamel, A. Nimmerjahn, and M. J. Schnitzer, "Automated analysis of cellular signals from large-scale calcium imaging data," *Neuron*, vol. 63, no. 6, pp. 747–760, 2009.

[9] P. P. Mitra and B. Pesaran, "Analysis of dynamic brain imaging data," *Biophysical journal*, vol. 76, no. 2, pp. 691–708, 1999.

[10] R. Maruyama, K. Maeda, H. Moroda, I. Kato, M. Inoue, H. Miyakawa, and T. Aonishi, "Detecting cells using non-negative matrix factorization on calcium imaging data," *Neural Networks*, vol. 55, pp. 11–19, 2014.

[11] P. Zhou, S. L. Resendez, J. Rodriguez-Romaguera, J. C. Jimenez, S. Q. Neufeld, A. Giovannucci, J. Friedrich, E. A. Pnevmatikakis, G. D. Stuber, R. Hen *et al.*, "Efficient and accurate extraction of in vivo calcium signals from microendoscopic video data," *Elife*, vol. 7, p. e28728, 2018.

[12] S. Soltanian-Zadeh, K. Sahingur, S. Blau, Y. Gong, and S. Farsiu, "Fast and robust active neuron segmentation in two-photon calcium imaging using spatiotemporal deep learning," *Proceedings of the National Academy of Sciences*, vol. 116, no. 17, pp. 8554–8563, Apr. 2019. [Online]. Available: https://doi.org/10.1073/pnas.1812995116

[13] J. Lu, C. Li, J. Singh-Alvarado, Z. C. Zhou, F. Fröhlich, R. Mooney, and F. Wang, "MIN1pipe: A miniscope 1-photon-based calcium imaging signal extraction pipeline," *Cell Reports*, vol. 23, no. 12, pp. 3673–3684, Jun. 2018. [Online]. Available: https://doi.org/10.1016/j.celrep.2018.05.062

[14] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014.

[15] A. Graves, G. Wayne, and I. Danihelka, "Neural turing machines," 2014.

[16] C. Olah and S. Carter, "Attention and augmented recurrent neural networks," *Distill*, 2016. [Online]. Available: http://distill.pub/2016/augmented-rnns

[17] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[18] F. Milletari, N. Navab, and S.-A. Ahmadi, "V-net: Fully convolutional neural networks for volumetric medical image segmentation," 2016.

[19] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," 2016.

[20] Q. Tian and M. N. Huhns, "Algorithms for subpixel registration," *Computer Vision, Graphics, and Image Processing*, vol. 35, no. 2, pp. 220–233, 1986.

[21] R. Liu, J. Lehman, P. Molino, F. P. Such, E. Frank, A. Sergeev, and J. Yosinski, "An intriguing failing of convolutional neural networks and the coordconv solution," *CoRR*, vol. abs/1807.03247, 2018. [Online]. Available: http://arxiv.org/abs/1807.03247

[22] L. Gondara, "Medical image denoising using convolutional denoising autoencoders," pp. 241–246, 2016.

[23] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2015, cite arxiv:1511.06434Comment: Under review as a conference paper at ICLR 2016. [Online]. Available: http://arxiv.org/abs/1511.06434

[24] A. Odena, V. Dumoulin, and C. Olah, "Deconvolution and checkerboard artifacts," *Distill*, 2016. [Online]. Available: http://distill.pub/2016/deconv-checkerboard

[25] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks." vol. 15, pp. 315–323, 2011. [Online]. Available: http://dblp.uni-trier.de/db/journals/jmlr/jmlrp15.html#GlorotBB11

[26] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[27] V. Nair, J. M. Susskind, and G. E. Hinton, "Analysis-by-synthesis by learning to invert generative black boxes," in *ICANN (1)*, ser. Lecture Notes in Computer Science, vol. 5163. Springer, 2008, pp. 971–981.

[28] V. Nair, J. Susskind, and G. E. Hinton, "Analysis-by-synthesis by learning to invert generative black boxes," in *Artificial Neural Networks - ICANN 2008*. Springer Berlin Heidelberg, pp. 971–981. [Online]. Available: https://doi.org/10.1007/978-3-540-87536-9_99

[29] I. Yildirim, W. Freiwald, T. Kulkarni, and J. Tenenbaum, "Efficient analysis-by-synthesis in vision: A computational framework, behavioral tests, and comparison with neural representations," 07 2015.

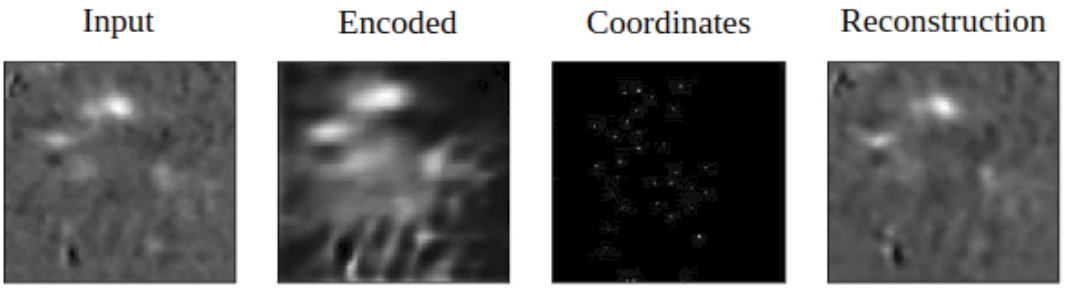
[30] A. Krull, E. Brachmann, F. Michel, M. Y. Yang, S. Gumhold, and C. Rother, "Learning analysis-by-synthesis for 6d pose estimation in rgb-d images," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 954–962.

[31] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014.

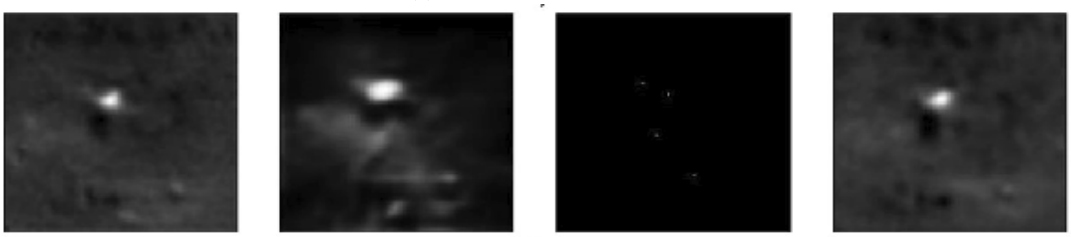
[32] D. Dwibedi, Y. Aytar, J. Tompson, P. Sermanet, and A. Zisserman, "Counting out time: Class agnostic video repetition counting in the wild," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[33] X. Ying, "An overview of overfitting and its solutions," in *Journal of Physics: Conference Series*, vol. 1168, no. 2. IOP Publishing, 2019, p. 022022.

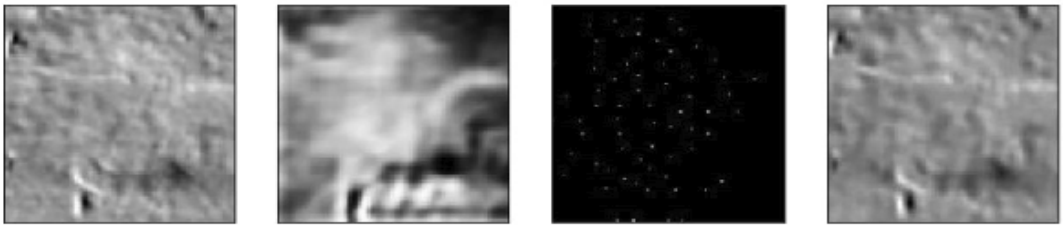
[34] R. Zhang, "Making convolutional networks shift-invariant again," in *ICML*, 2019.



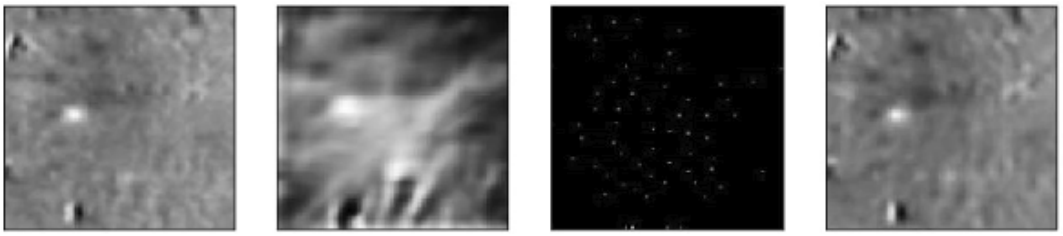
(a) Session DG-13-3-8-17



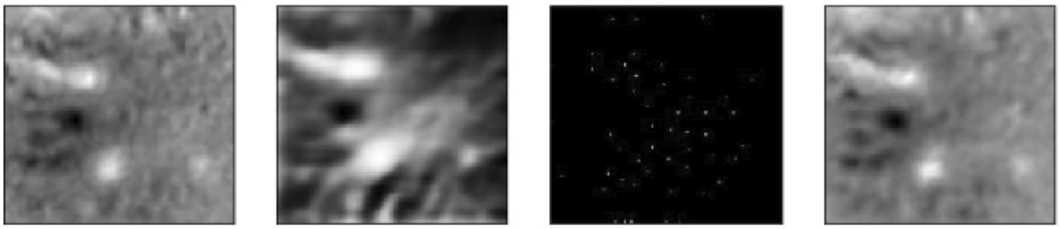
(b) Session DG-13-8-11-17



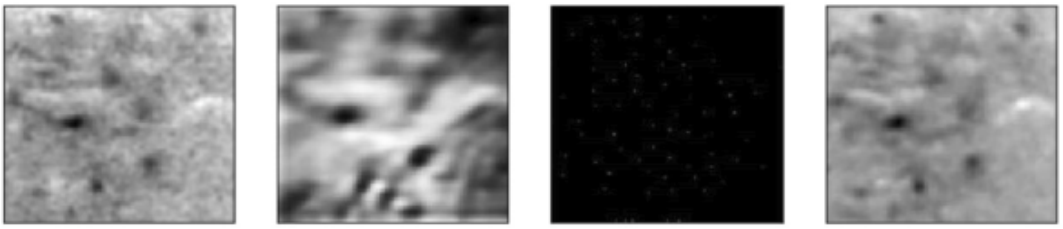
(c) Session DG-13-8-16-17



(d) Session DG-13-8-16-17

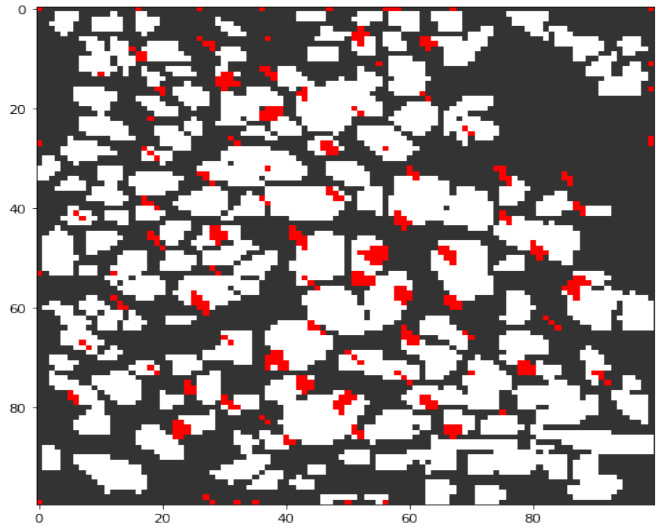


(e) Session DG-13-8-18-17

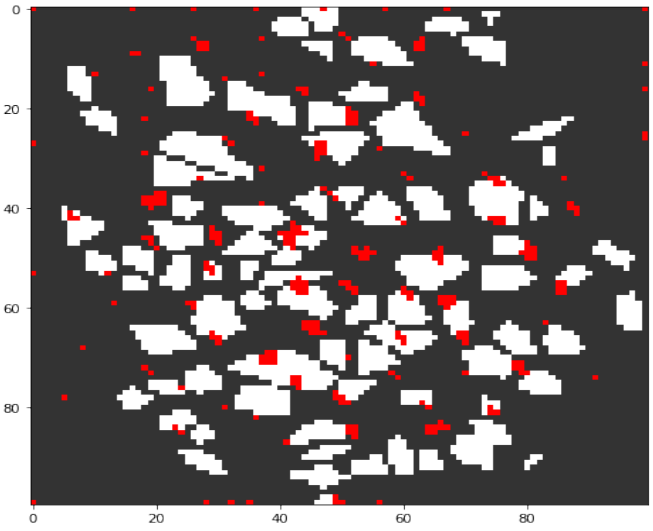


(f) Session DG-13-8-8-17

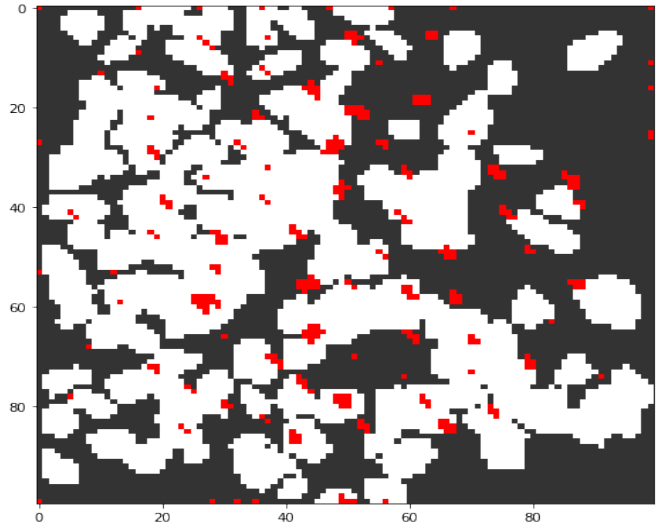
Figure 20: Each row represents the correlated images with different brightness as input and output of the network. First column depicts the input images from various sessions. Second column depicts the output from the encoder network. Third column represents the summed one hot pixel image. The last column represents the reconstructed image from the latent space.



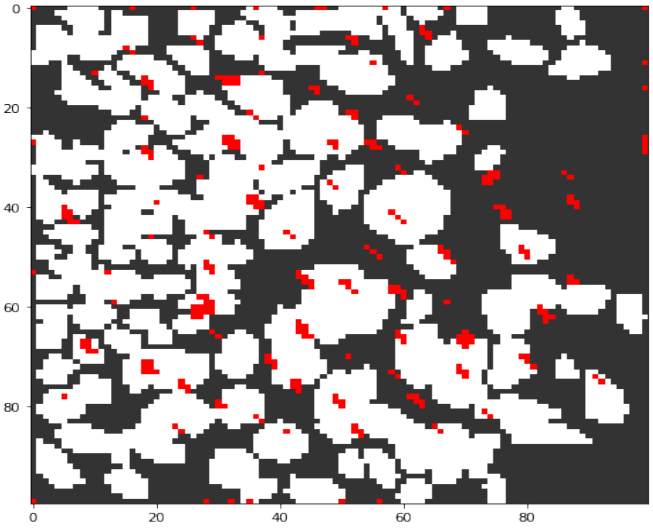
(a) Session DG-13-3-8-17



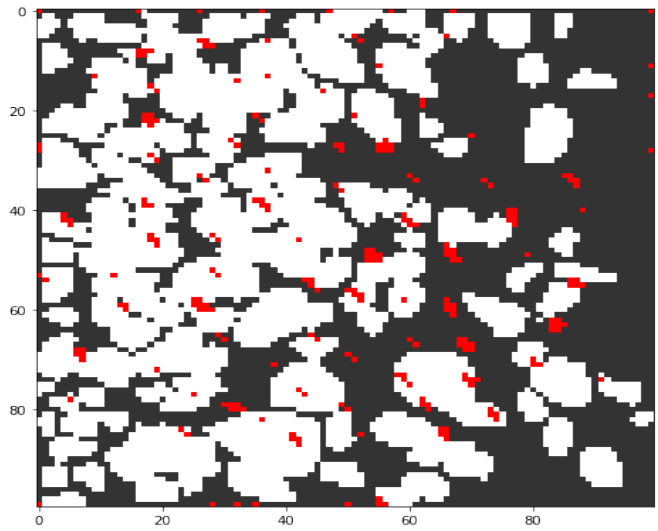
(b) Session DG-13-7-12-17



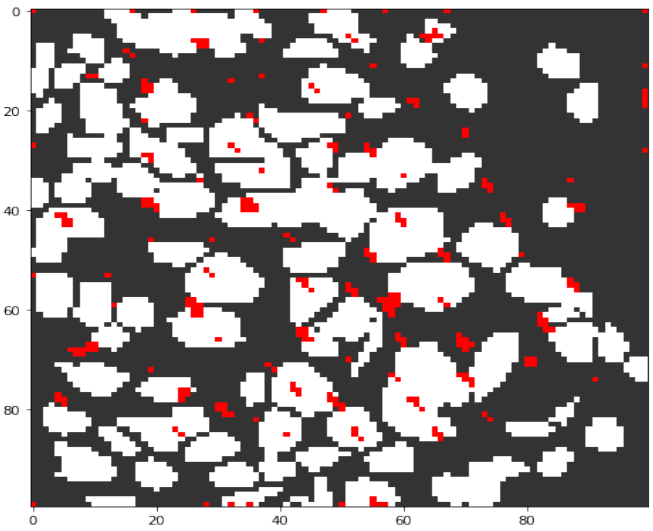
(c) Session DG-13-8-11-17



(d) Session DG-13-8-16-17



(e) Session DG-13-8-18-17



(f) Session DG-13-8-8-17

Figure 21: Visualization of prediction output in red and manually annotated ROI mask in white for each session.