3D CNNs as the solution for inverse EEG problem (Machine Learning 2023 Course)

Andrei Kalinichenko [1] **Victor Kozhevnikov** [1] **Pavel Tikhonov** [1]

Abstract

We trained two autoencoders on two separate datasets (original and dilated) for two differently formulated tasks (regression and classification). We were able to achieve any significant learning only with VNet on classification model for both datasets, in all other scenarios the predicition was the empty image. We were slightly more successful with original dataset getting the results better than random. It is not enough to compete with any modern prediction technique, however we show that there is a potential to apply convolutional 3d autoencoders for inverse EEG task.

Github repo: ml_eeg_project
Dataset link: inverse_EEG_dataset

1. Introduction

Electroencephalography (EEG) is a powerful, non-invasive imaging method that is commonly used to measure brain electrophysiological activity. EEG has superior temporal resolution (milliseconds) when compared to metabolism-based approaches such as fMRI (seconds), which outperforms EEG in spatial resolution performance (cms for EEG vs mms for MRI/CT) (Hecker et al., 2021). The reason for such poor spatial vision is that the signals are collected from the surface of the head. Brain currents are muddled by skull and scalp tissues with varying conductivity. In those circumstances, forward and inverse problems were formulated (Razorenova et al., 2020). The forward problem assumes that we can predict an EEG signal (or any other method) if we are aware of brain activity. Conversely, the solution to the inverse problem is an algorithm that can reconstruct inner currents in brains from EEG signals collected from them. Solving the inverse problem would allow one to get insight into the spatial dynamics of the EEG, which includes identifying the particular source of the EEG signal up to a distinguishable group of neurons. Solutions can be extremely useful both in clinical neuroscience, for instance, to find and manifest tentative localization of epileptogenic foci, and in neuroscience research to push our understanding about circuits and fields responsible for multiple tasks. This requires constructing a source model applied to a model of the head.

1.1. Classical Approaches to the EEG Inverse Problem

The first method to describe the distribution of brain activity sources from EEG/MEG data is equivalent current dipole (ECD) (Sarvas, 1987). It assumes that a signal source measured with EEG can be represented by a single or a few dipoles (Kavanagk et al., 1978). However, this approach is often inadequate for determining currents across the cortex. The distributed dipole model is a more physiologically meaningful approach that hypothesizes that the neural activity responsible for EEG measurement is distributed across a larger region of the brain instead of being limited to small dipoles. The objective of distributed dipole models is to locate the 3D distribution of neural activity that accounts for the EEG measurement.

1.2. Artificial Neural Networks and Inverse Solutions

Regarding all approaches described above, the inverse problem is a target for ANN based solutions. With different problem statements, many models are capable of predicting activity sources with some accuracy. Cui showed that a neural network can be trained to reconstruct the position of a single source short time span with a long-short-term memory (LSTM) recurrent neural network architecture (Cui et al., 2019). In situations with one or multiple dipoles, the ConvDip model (Hecker et al., 2021) has been demonstrated to surpass the performance of the traditional methods for localizing 63 % of sources. In 2019 it was shown by Tankelevich (Tankelevich, 2019) that deep feed-forward NN can predict a set of neuron clusters that produced a given signal. In a recent work, Razorenova (Razorenova et al., 2020) showed that the inverse problem for cortical potential imaging could be solved using a deep U-Net architecture that could reconstruct source distribution

[1]Skolkovo Institute of Science and Technology, Moscow, Russia. Correspondence to: Andrei Kalinichenko <sk>.

Final Projects of the Machine Learning 2023 Course, Skoltech, Moscow, Russian Federation, 2023.

with 1–45 dipoles with a relative error of 2–10 %. In this approach, the prediction is done on a 2D projection of the brain.

The current study aims to upscale the problem to 3D space and recreate the location of diverse dipoles among the cortex. Nowadays, there are several 3D CNN architectures based on the Unet model. Here we utilize the 3DUnet and VNet architectures that were explicitly designed for the reconstruction of medical data (Çiçek et al., 2016; Milletari et al., 2016). We anticipate that models are capable of detecting multiple sources using training data with biologically plausible constraints in 3D volumetric space. The main contributions of this report are:

- we demonstrated the results of parameters search suitable for training with 3D-UNet and VNet. In particular to show that we could find optimal parameters only for VNet for prediction EEG sources, while 3D U-Net suffer from gradient vanishing and fail to learn any type of pattern.
- we show and evaluate the pipeline with VNet model that after 170 epochs was able to roughly predict the location of current points with accuracy above random

2. Methods

All the code is stored at ml_eeg_project.

2.1. Models architectures

We used two 3D CNN architectures: VNet and 3D U-Net. The realizations on PyTorch were taken from the open-source GitHub repositories wolny and mattmacy.

2.1.1. 3D U-Net

3D U-Net is a convolutional autoencoder with additional connections between the encoder and the decoder parts (Çiçek et al., 2016). In the encoding path, each layer has a double convolution of 3×3×3 each, followed by a rectified linear unit (ReLu), and then a 2×2×2 max pooling with strides of two in each dimension. In the synthesis path, each layer consists of an upconvolution of 2×2×2 by strides of two in each dimension, followed by doubled 3×3×3 convolutions, each with a ReLu. As in Unet to get fine features from the original, there are shortcut connections between all levels of the equal resolution. In the last layer a 1×1×1 convolution reduces the number of output channels to one for regressin task and two for classification (first conrain background probabilities, the second – dipoles probabilities).

2.1.2. VNet

The architecture of the VNet model was the same as described in the original article (Milletari et al., 2016). In comparison to 3D-Unet there are several significant differences. The Vnet architecture does not rely on pooling layers during downsampling but contains an alternative approach with 2x2x2 convolution with stride 2. In that case, downsampling layers also contain weights, so the downsampling process can be trained conversely to pooling layers. The activation function in Vnet is PReLU instead of ReLU in all places required. The last upgrade includes adding a skip connection at the end of each level in which the entry is summed up with the level product with an element-wise sum.

2.2. Dataset

As there are not enough annotated EEG signals to correspond to specific source locations, simulated data is generated with a forward model to train the models. The dataset was synthetically generated and generously provided by A. Razorenova. The dataset consists of vector pairs: the first contains all EEG electrode parameters; the second contains the representation of brain activity with dipoles and their parameters corresponding to the state of electrodes.

2.2.1. Distribute dipole representation of brain activity (predicted vector)

The set of 4000 dipole matrices of size $nx4$, where n is the number of dipoles generated (around 1000). One raw contains three coordinates relative to the center of the head model and the current value. Based on four different head shapes (modeled from four MRI scans; 1000 matrices for each head), sets of evenly distributed points were generated for the surfaces of both hemispheres. Each point belongs to one of the 64 functional zones of the brain cortex. A random non-zero Gaussian distribution was chosen for each image and for each functional zone to generate intensity points that belong to the zone. Thus, the resulting image contains a set of points, where each one is equal to a dipole with a particular current value. The values were normalized from 0 to 1.

2.2.2. EEG electrode vectors (input for a model)

Each 128x4 matrix represents one momentary stage of 128 electrodes in a standard EEG montage. One raw contains three coordinates (from the same space as for dipoles) relative to the center of the head model and the voltage. The electrode voltage values were simulated as the forward solution implemented by the MNE library from the paired images that contain a unique brain activity state (Gramfort et al., 2013).

2.3. Preprocessing

We transformed each vector into a 3D array with non-zero voxels placed according to the coordinates in the space with a resolution of 64x64x64 and the resting points as a background with zero value. The given resolution was chosen to fit the allotted time and the limits of GPU powers granted by the Zhores cluster (Zacharov et al., 2019) In all pairs, the coordinates were rescaled to most effectively occupy the 64^3 volume, therefore the scales of the dipole and electrodes spaces are no longer equal.

We also utilized an additional stage of preprocessing by adding simple augmentation for both input images and label images: enclosing each dipole with the same value by the length of 2 in each direction and, thus, obtaining 5x5x5 voxel cubes as the representation of one electrode or dipole in its center. Doing so, we expected that models would perform better in differentiating the sources from the background as we reduced the sparsity of the image. Thus, we created two types of inputs for training: the augmented one and the original one.

2.4. Optimization and Loss Function

The training of the neural netykh was performed on the ZHORES supercomputer using Tesla V100-SXM2 GPUs (Zacharov et al., 2019). During the parameter search, 800 pairs were used for training and 200 for validation. For final training, the full dataset was randomly split into 3600 train pairs (that contain all pairs used for parameter tuning) and 800 test pairs.

The primary challenge encountered in this task is the sparsity of the data. To tackle this issue, we experimented with several approaches, such as increasing the value of true labels (multiplying all values by a constant value), enlarging the volume of all non-zero voxels (the additional stage of preprocessing), and reformulating the task as a classification problem. In the case of the classification task, all intensities of non-zero voxels on label images were set to one.

In the training phase, we utilized the Adam optimizer with a batch size of 10 for 3D Unet and 8 for VNet during parameter search and 48 or 40 for 3D Unet and VNet, respectively, during training on the full dataset. The learning rate was inspected from 1E-5 to 0.1, and 1E-3 was used for final training. The learning rate decreased by a factor of 0.995 after every epoch (and inspected in range from 0.99 to 0.999). For the regression task, we employed the mean squared error (MSE) loss and L1 loss functions, while for the classification task, we utilized the weighted cross-entropy loss function with reduced weight for background zero values of 1E-7. Additionally, for VNet, weight decay was 1E-4. During parameter search, all experiments were performed until the stable loss values, or until 100^{th} epoch; the final model was trained for 270 epochs that was limited by the time available.

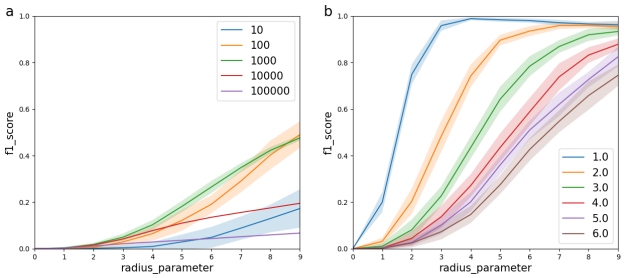

Figure 1. The metric values obtained when "predictions" are **a.** completely random: m points uniformly distributed in the cube, mean±SD for 20 samples each, legend show m value; **b.** almost perfect: each sample generated by drawing points from a Normal distribution centered at each of the n true points, mean±SD for 20 samples of n points each.

2.5. Metrics

To evaluate the prediction, we calculated a performance metric. The proposed metric is computed through the following process: Let n be the total number of true points, and m be the total number of predicted points. Let r be the radius of the ball selected around each true point. First, we perform Agglomerative clustering with a distance threshold of r, to exclude points that are close enough to each other. The true positive (TP) value is obtained by summing the indicator functions for each true point i that has at least one predicted point j lying within its ball:

$$TP = \sum_{i=1}^{n} [\exists j \in [1, m] : |y_i - \hat{y}_j|_1 \leq r]$$

The false positive (FP) value is then calculated as the difference between the total number of predicted points m and the true positive value TP:

$$FP = m - TP$$

Similarly, the false negative (FN) value is the difference between the total number of true points n and the true positive value TP:

$$FN = n - TP$$

The final metric of interest is the F1 score, which can be computed using the following expression:

$$F_1 = \frac{2TP + FP + FN}{2TP}$$

It is worth noticing that the radius r is a hyperparameter and the selection of it is some task. We calculated the

metric for different values of the radius parameter from 0 to 9 and plotted the curve for evaluation. The average and standard deviation were computed for each radius value while evaluation the prediction results. To draw conclusions from obtained results with this metric we provide an instance of metric with random results and almost perfect results that could be observed in case of reasonable prediction (Fig.1). We use estimated metric for random and fine results when evaluating resulting metric.

3. Results

3.1. Parameters search

Throughout the described set of parameters the results were obtained for 3D U-Net model. Regardles any parameters in regression tas the results were the same and unimpressive. The prediction was always empty (all values set to zero or close to it). During the training for several epochs the model rapidly starts predicting empty volume that results in a very small loss. Compared to the weights of the model, which are in $10^{-2} - 10^{-3}$ range the loss was smaller in range of 10^{-4}.

Due to poor results for regression solution, we changed the approach to our problem as a classification. Since the quantity and relative position of dipoles is more decisive factor, than their values.

3.2. VNet training results with full dataset

In VNet model training, regardless of whether the data was dilated or not, we observed an unstable loss curve decline with a sharp rise in values from tenths to 10 to the 12th degree (Fig.2. a,b). To analyze the model predictions, we examined 3 different checkpoints, at which the loss was the least. - (20, 270) and 120 epochs for models trained on dilated and original data. In the case with dilation, after 20 epochs, the model predicts zero background. The problem was solved at epoch 30 (loss explosion), after which the model began to predict the position of the dipoles, improving the prediction with each subsequent epoch. By the 270 epoch, we see VNet's preference to collocate dipoles only in areas of the strongest activities recorded by electrodes both on training and test datasets as well as for non-dilated data (epoch 120) (Fig.3 a,b).

The reason for sudden loss explosions might be that some weights or gradients might become too small, to handle that it is recommended to use clipping in the loss calculation. Another possible problem might be that we do not reduce learning rate enough via exponent scheduler while training (gamma=0.995). Updates become too drastic, and therefore divergence from the optimum.

We estimated the $F1\text{-}score$ based on the radius of the model

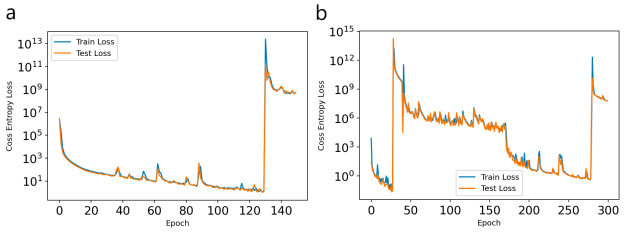

Figure 2. Train and test Cross Entropy Loss for VNet on two generated datasets; **a** - original data, **b** - 5x5x5 dilated data

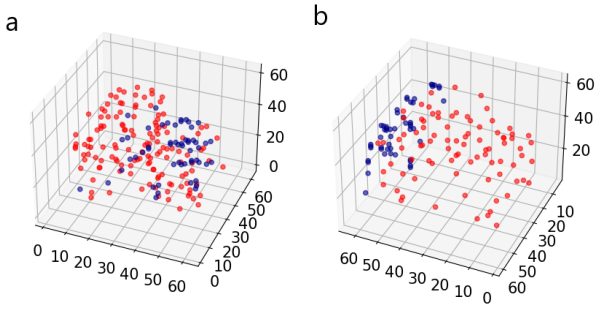

Figure 3. Test predictions of VNet; **a** - original data, **b** - 5x5x5 dilated data; red dots - true dipoles, blue dots - predicted dipole coordinates

to evaluate the predictions. The distance from true dipoles, at which the predicted ones will be considered valid, determines the credibility of the predicted ones. The minimum prediction probability threshold, which is accepted by the model as true in each preset and training setup, always differs and is selected based on the training data for which the area under the $F1\text{-}score$ - $radius$ curve is maximum. Values that are less then a chosen threshold set as the 0 background. For now, thresholds were selected for dilated and non-dilated as data 0.63 and 0.72, respectively.

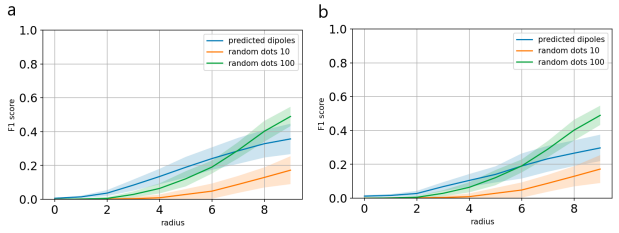

Figure 4. Metric estimations results; **a** - original data, **b** - 5x5x5 dilated data

We applied our metric to indicate $F1\text{-}score$ for model test predictions, comparing it with 10 and 100 randomly distributed points, since we have cases with number of points less than 100. VNet trained on data with no dilation preformed better than on 100 randomly distributed points in a selected radius range of 1 to 6,5 (Fig. 4.a). We could state almost the same for dilated data, but it is still slightly worse compared to original data (radius range of 1 to 6) (Fig. 4.b). We still suppose that dilation could produce better results or at least reduce the time spent, however for now we cant prove our assumption as we have to achieve the accuracy level at which it could be relevant. To conclude, model performance is better than a random guess in the radius less than 6, where it is the most crucial to get the improvement in prediction, as it indicates the match of prediction dipole with true one with high precision.

4. Conclusion

Only with VNet on the classification model for both datasets were we able to achieve any appreciable learning; in all other cases, the prediction was an empty image. With the original dataset, we had a little bit more success in producing findings that were superior to random. Though it is insufficient to compete now with any current prediction method, we have discovered the optimal parameters to create the VNet model and train some pattern. Therefore, we continue training to achieve the limitations of the accuracy for given parameters.

References

Çiçek, Ö., Abdulkadir, A., Lienkamp, S. S., Brox, T., and Ronneberger, O. 3d u-net: learning dense volumetric segmentation from sparse annotation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2016: 19th International Conference, Athens, Greece, October 17-21, 2016, Proceedings, Part II 19*, pp. 424–432. Springer, 2016.

Cui, S., Duan, L., Gong, B., Qiao, Y., Xu, F., Chen, J., and Wang, C. Eeg source localization using spatio-temporal neural network. *China Communications*, 16(7):131–143, 2019.

Gramfort, A., Luessi, M., Larson, E., Engemann, D., Strohmeier, D., Brodbeck, C., Goj, R., Jas, M., Brooks, T., Parkkonen, L., et al. Meg and eeg data analysis with mne-python. front. neurosci. 7 (267), 1–13 (2013), 2013.

Hecker, L., Rupprecht, R., Tebartz van Elst, L., and Kornmeier, J. Convdip: A convolutional neural network for better eeg source imaging. *Frontiers in Neuroscience*, 15: 569918, 2021.

Kavanagk, R. N., Darcey, T. M., Lehmann, D., and Fender, D. H. Evaluation of methods for three-dimensional localization of electrical sources in the human brain. *IEEE Transactions on Biomedical Engineering*, (5):421–429, 1978.

Milletari, F., Navab, N., and Ahmadi, S.-A. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 fourth international conference on 3D vision (3DV)*, pp. 565–571. Ieee, 2016.

Razorenova, A., Yavich, N., Malovichko, M., Fedorov, M., Koshev, N., and Dylov, D. V. Deep learning for non-invasive cortical potential imaging. In *Machine Learning in Clinical Neuroimaging and Radiogenomics in Neuro-oncology: Third International Workshop, MLCN 2020, and Second International Workshop, RNO-AI 2020, Held in Conjunction with MICCAI 2020, Lima, Peru, October 4–8, 2020, Proceedings 3*, pp. 45–55. Springer, 2020.

Sarvas, J. Basic mathematical and electromagnetic concepts of the biomagnetic inverse problem. *Physics in Medicine & Biology*, 32(1):11, 1987.

Tankelevich, R. Inverse problem's solution using deep learning: An eeg-based study of brain activity. part 1. *Preprint*, pp. 1–15, 2019.

Zacharov, I., Arslanov, R., Gunin, M., Stefonishin, D., Bykov, A., Pavlov, S., Panarin, O., Maliutin, A., Rykovanov, S., and Fedorov, M. "zhores"—petaflops supercomputer for data-driven modeling, machine learning and artificial intelligence installed in skolkovo institute of science and technology. *Open Engineering*, 9(1):512–520, 2019.

A. Team member's contributions

Explicitly stated contributions of each team member to the final project.

Andrei Kalinichenko (35% of work)

- Reviewing literate on the topic (10 papers)
- Coding the main pipline and preprocessing script
- Building and evaluating of the metric
- Performing experiments on Zhores server
- Preparing the Introduction, Methods and Results sections of this report

Kozhevnikov Victor(35% of work)

- Performing experiments with VNet on Zhores server
- Building dilated dataset and preprocessing optimisation
- Preparing the Results sections of this report

Pavel Tikhonov (30% of work)

- Coding the training script
- Performing experiments with 3D-UNet
- Building Git repository
- Preparing the Methods sections of this report

B. Reproducibility checklist

Answer the questions of following reproducibility checklist. If necessary, you may leave a comment.

1. A ready code was used in this project, e.g. for replication project the code from the corresponding paper was used.
   - ☑ Yes.
   - ☐ No.
   - ☐ Not applicable.

   Students' comment: From the git repository only the scripts buildingblocks.py and models.py were borrowed from outsource (the links are in the Method section)

2. A clear description of the mathematical setting, algorithm, and/or model is included in the report.
   - ☑ Yes.
   - ☐ No.
   - ☐ Not applicable.

   Students' comment: None

3. A link to a downloadable source code, with specification of all dependencies, including external libraries is included in the report.
   - ☑ Yes.
   - ☐ No.
   - ☐ Not applicable.

   Students' comment: None

4. A complete description of the data collection process, including sample size, is included in the report.
   - ☑ Yes.
   - ☐ No.
   - ☐ Not applicable.

   Students' comment: None

5. A link to a downloadable version of the dataset or simulation environment is included in the report.
   - ☑ Yes.
   - ☐ No.
   - ☐ Not applicable.

   Students' comment: None

6. An explanation of any data that were excluded, description of any pre-processing step are included in the report.
   - ☑ Yes.
   - ☐ No.
   - ☐ Not applicable.

   Students' comment: None

7. An explanation of how samples were allocated for training, validation and testing is included in the report.
   - ☑ Yes.
   - ☐ No.
   - ☐ Not applicable.

   Students' comment: None

8. The range of hyper-parameters considered, method to select the best hyper-parameter configuration, and specification of all hyper-parameters used to generate results are included in the report.
   - ☑ Yes.
   - ☐ No.
   - ☐ Not applicable.

   Students' comment: None

9. The exact number of evaluation runs is included.
   - ☑ Yes.
   - ☐ No.
   - ☐ Not applicable.

   Students' comment: None

10. A description of how experiments have been conducted is included.
    - ☑ Yes.
    - ☐ No.
    - ☐ Not applicable.

    Students' comment: None

11. A clear definition of the specific measure or statistics used to report results is included in the report.
    - ☑ Yes.
    - ☐ No.
    - ☐ Not applicable.

    Students' comment: None

12. Clearly defined error bars are included in the report.
    - ☑ Yes.
    - ☐ No.
    - ☐ Not applicable.

    Students' comment: None

13. A description of the computing infrastructure used is included in the report.

- [x] Yes.
- [ ] No.
- [ ] Not applicable.

Students' comment: None