ARTICLE TITLE

FABIA SERGATZ* & ISABEL RITTER* & NADINE KWAPIL*

CONTENTS

LIST OF FIGURES

LIST OF TABLES

* *Cognitive Science, University of Osnabrück, Germany*

1 INTRODUCTION AND MOTIVATION

The human brain is undoubtedly the most complex and intelligent thing we know of. For centuries humans try now to understand how it works. And interestingly, more and more the trend in research goes towards using artificial intelligence to better understand the human one. Artificial neural structures are being compared to the human brain on the neuronal level in a simplified way. In recent years research has made significant strides closing the gap between of biological and artificial intelligence. Since, improving the understanding of human and artificial intelligence is a fundamental goal in cognitive science, we decided to participate in the Algonauts Project 2023 Challenge. The motivation of the project is to gain new and deep insights in terms of understanding the brain through AI models and also the other way around. As an overarching goal the intersection of the two key areas biology and artificial intelligence are further analysed which may lead to groundbreaking findings. The challenge's task is the optimization of an innovative encoding model that receives visual stimuli in form of natural scene images (NSD Dataset) and predicts the resulting neural responses. Our approach is to train two different model architectures (ResNet50 and a simple CNN image recognition model) on the Cifar100 and Eco dataset. Afterwards we let the NSD Data run through the models. From certain layers we extract the features and therefore get new representations of the NSD images which are used for the challenge. The new representations serve as input for a linear regression model that trains on the NSD features and their labels (the activated brain regions (fMRI) and learns to predicts brain activity to visual inputs. The following section explains the exact methods that were used and why we expect new representations to perform better.

2 CURRENT STATE

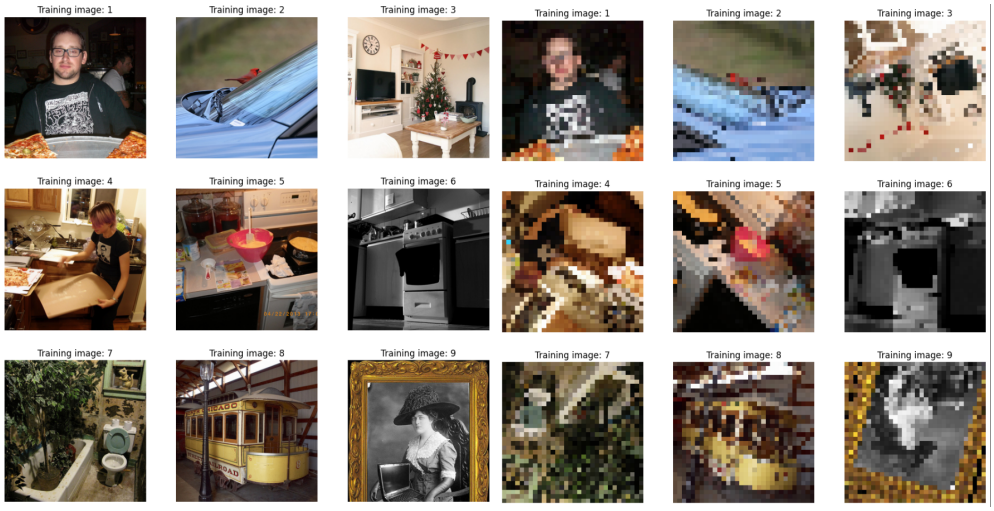
The Algonauts Challenge 2023 is based on the challenges from the previous years, where the goal was to explain brain responses to still images (Challenge 2019) and to short video clips (Challenge 2021). This year's challenge goes one step further with the task being the prediction of the activation of certain visual brain regions when subjects see an image of a natural scene (NSD Dataset). The main idea is to learn more about the human brain and how it works but also improve the performance of artificial networks. Investigating this intersection hopefully brings great new findings in both areas. When exploring the intersection between artificial and human intelligence researchers have already found similarities between the human brain and CNN models: for example that earlier brain regions like the V1 (here concentrating on the visual information processing) encode shape information just like the earlier layer in a CNN and later brain regions like the anterior visual temporal cortex encode classification just like in the later layer in a CNN. So there seems to be a representation of the key information in the different layers of the models. An interesting approach now is to see whether these representations are sufficient to predict the activity of different brain regions. Furthermore, there has also been research on Deep Neural Networks to predict brain activity in the V1 to V4, also on the NSD Dataset, and how to optimize these so that the performance of the network increases. Deep Neural Networks show a good performance in tasks like this. (BrainOptNNReference?).

3 METHODS

4 THE IMPLEMENTATION

The project code is divided into four different libraries. The library 'DataProcessing_Library' provides essential function for pre-processing and visualization for the data. The use of this library is facilitating the preprocessing of the different data sets and visualizing samples. We created two different preprocessing function since we treat the cifar100 and the eco as a tf dataset whereas for the NSD data we only preprocessed the images, not the targets. In the "data_prep" function for the NSD data, we cast, normalize the images. The data augmentation is created using tensorflow's Sequential API. It applies random horizontal and vertical flipping with a rate of 0.1 to the images. The normalization function converts the data to type float32 and scales the pixel to a range between zero and one. The resizing of the images to a 32x32 resolution is done by the usage of a single tf.keras.data.Resizing() layer. The preprocessing for the cifar100 and the eco dataset just like the preprocess function for the NSD data, includes normalization, resizing and data augmentation as well. Except that it takes the tf dataset as an input and maps an image to a target. Additionally, we create on hot vectors for the targets, with a depth of ten, according to the 100 different classes of cifar and eco datatset. In the end of the function, we cache the dataset to our memory. After that, just like usual, we shuffle the data, create batches and prefetch it. Finally, we implemented a visualization function to show samples of the different datasets. The images which are shown are the raw data. The function takes the images as input and plots that using Matplotlib. It allows a quick insight into the loaded data. However, we also want to show the preprocessed data to see, how the images look like after being resized or augmented.

Figure 1: raw and preprocessed image data



In the second library "models" we created three different model. The first one is a CNN. We are creating a functional API and subclassing. The model class ConvModel consists of an initialization of the parameters which mostly are the layers. Further, it owns the metrics property, to keep track of the accuracies and losses. The class has four different methods. A call methods giving the input through all layers by calling them with self.layer(), a step and a train method and The CNN model itself has the following layers.

The second model is stored in the 'untrainedResNetModel' class.

It simply consist of the attributes, which are the layers, optimizer and the metrix. Further, it also has a call method, where again, the input is passed through every

Figure 2: CNN architecture

```
self.Conv1 = tf.keras.layers.Conv2D(input_shape=(32, 32, 3), kernel_size=(2, 2), padding='same', strides=(2, 2), filters=32)
self.BatchNorm1 = tf.keras.layers.BatchNormalization()
self.MaxPool1 = tf.keras.layers.MaxPooling2D(pool_size=(2, 2), strides=(1, 1), padding='same')
self.BatchNorm2 = tf.keras.layers.BatchNormalization()

self.Conv2 = tf.keras.layers.Conv2D(kernel_size=(2, 2), padding='same', strides=(2, 2), filters=64)
self.BatchNorm3 = tf.keras.layers.BatchNormalization()
self.MaxPool2 = tf.keras.layers.MaxPooling2D(pool_size=(2, 2), strides=(1, 1), padding='same')
self.BatchNorm4 = tf.keras.layers.BatchNormalization()

self.Conv3 = tf.keras.layers.Conv2D(kernel_size=(2, 2), padding='same', strides=(2, 2), filters=128)
self.BatchNorm5 = tf.keras.layers.BatchNormalization()
self.MaxPool3 = tf.keras.layers.MaxPooling2D(pool_size=(2, 2), strides=(1, 1), padding='same')
self.BatchNorm6 = tf.keras.layers.BatchNormalization()

self.flatten = tf.keras.layers.Flatten()
self.BatchNorm7 = tf.keras.layers.BatchNormalization()
self.dense1 = tf.keras.layers.Dense(256)
self.dense2 = tf.keras.layers.Dense(128)
self.dense3 = tf.keras.layers.Dense(100)
```

layer and the output is returned. The model consists of the resnet50 architecture, followed by a flatten layer and a single output layer. The third model has the same architecture as the resnet50 model class, with the only difference, that the resnet50 is pretrained on the imagenet. In the 'main_file' we are downloading the datatsets Since cifar100 is a tfds.dataset, we simply loaded it from tensorflow datasets, which we then split into train and test data using the "as_supervised" arguemnet of the tfds.load function. We downloaded the eco dataset via google drive from the hp5py file by reading it and accessing the different keys. Because the eco dataset consists of 250 000 train images and the cifar100 of 50 000 We cropped the amount of the eco data we used, to 50 000. The NSD data came in in a separate folder for each subject, consisting of 'training_split', 'test_split' and 'roi_masks' data. Because the train images where all stored as separate png.files in a folder we created a function to load, open and convert each image-file and then stored it as a Tensor in a npy.file. We chose this format, since the fmri images are stored in the same one. After we had once created a single file for each of the 8 subject, we used NumPy to load the NSD train_images data from the npy.file. The structure we were following in our main.file is downloading, preprocessing. Next up, creating the models, compiling them and training them by using the model.fit() function.In each epoch we printed the train and test accuracies and losses. After training, we stored all the "METRICS INFORMATIN" in the variable "history" which we later used for visualization. We added 'cnn_model.save_weights('./logs/path/')' so that the weights can be saved in google drive and later on reloaded for using them in the main_NSD file. The weights are being saved for the cnn trained on cifar and eco separately, and also for the resnet50 trained on cifar100 and the eco dataset. In the main_NSD file we first load the NSD data. Then create models and we reload the saved weights to reconstruct the trained models. The saving of the weights is only necessary because we create and use the trained models in separate files. With the functions 'output_to_input' (separate for resnet and cnn due to their different structures) we can extract the output of a certain layer (determined by a parameter given into the funtion) from the models when the NSD data runs on them. These features are the fed into the Linear Regression models which trains to connect the features to the fmri responses of the subjects.

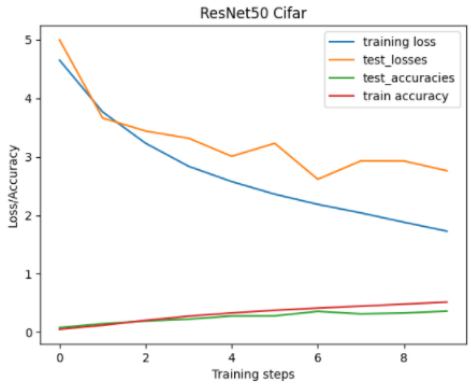
4.1 Data

	CIFAR-100	MiniEcoset-100	NSD
Description	100 classes with 600 images each, grouped in 20 subclasses including nature, people, devices and other objects	100 classes with 2700 images each, subsegt of Ecoset	8-subject 7T fMRI responses to different natural scenes from COCO datanbase
Images	32 x 32 pixels RGB	64 x 64 pixels RGB	425 x 425 pixels RGB
Training and Test set	Training: 50,000 Test: 10,000	Training: 50,000 Test: 5,000	Training (different images per subject): [9841, 9841, 9082, 8779, 9841, 9082, 9841, 8779] Test: (different images per subjects): [159, 159, 293, 395, 159, 293, 159, 395]

5 RESULTS AND DISCUSSION

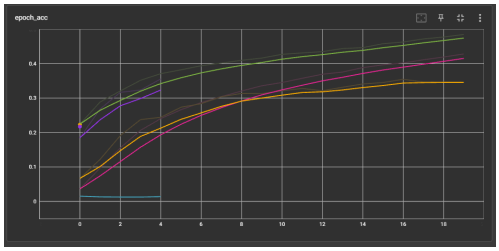
To reflect the outcome of our project and present the results. Starting by first comparing the different models used for training and second, answering the overall question about what the best representation for visual features are. We will also comment the accuracy of the prediction of the responding brain regions to perceived natural visual scenes (from the NSD data set) using our models. The graph below show different results for the two models CNN and UN-pretrained ResNet model, both trained on the Cifar100 data.

Figure 3: CNN vs Resnet model



The train data of both the models have a similar behavior. However, it is significant that the test accuracy of the resnet is exctremley low - with values around 0.03. Equivalent, the loss is pretty high/much higher than for the CNN One reason

Figure 4: Cifar100 on Resnet model



for that could be, that the eco dataset was resized to a lower resolution of 32x32. Because the cifar dataset scenes are less complex, it is easier for the model to learn on them and predict correct results. However, the resnet50 was learning, but the improvement was too small for a low number of epochs, like 10. Further reason for that is that the reason is a very large an complex network which was trained on im-

ages with much higher resolution, with shape 22x224. In addition to being trained on high resoluted images, the complex model might have shown improvement in performance after further epochs Since we had capacity and time limitations for the project, we decided to work with a pretrained resnet50. Due to missing data about the actual outcome, we cannot give a statement on the original question about the optimization of an innovative encoding model such that it receives visual stimuli in form of natural scene images

However, the accuracies shown in the tutorial code of the Algonauts Challlenge showed good results. Therefore, there must be complex models like alexnet that Show a good Performancethe submitted data from other participants of the Algonauts challenge shows, that it is indeed Furthermore, In papers like ... (paper was wir am anfang leojn geschickt haben) it is shown, that "The proposed framework can significantly predict responses of over 20% percent voxels in early visual areas (i.e., V1-lateral occipital region, LO) and achieve unprecedented prediction accuracy.)" They also assume a linear mapping between feature representation to brain activity. However, it remains unknown whether such linear mapping is sufficient for maximizing prediction accuracy

Another interesting question is whether there are any remarkable differences in the accuracy for predicting ROI's, meaning, if any ROI's are easier to predict than others. Further, one could have made research using non-linear model as it is computationally difficult to train a complex nonlinear model separately for each voxel.

6 REFERENCES

Yang X, Yan J, Wang W, Li S, Hu B, Lin J. 2022. Brain inspired models for visual object recognition: an overview. Artificial Intelligence Review, 1-49.

Gifford AT, Lahner B, Saba-Sadiya S, Vilas MG, Lascelles A, Oliva A, Kay K, Roig G, Cichy RM. 2023. The Algonauts Project 2023 Challenge: How the Human Brain Makes Sense of Natural Scenes. arXiv preprint, arXiv:2301.03198. DOI: https://doi.org/10.48550/arXiv.2301.03198.

Zhang, C., Qiao, K., Wang, L., Tong, L., Hu, G., Zhang, R., Yan, B. (n.d.). A visual encoding model based on deep neural networks and transfer learning. National Digital Switching System Engineering and Technological Research Center, Zhengzhou, China, 450000; Center for Magnetic Resonance Imaging, Department of Neuroscience, University of Minnesota at Twin Cities, MN. USA. 55108. [Running title: Visual encoding models and deep neural networks].

.

St-Yves G, Allen EJ, Wu Y, Kay K, Naselaris T. 2022. Brain-optimized neural networks learn non-hierarchical models of representation in human visual cortex. bioRxiv.

.