Sparse Autoencoders for Word Decoding from Magnetoencephalography

Michelle Shu[1] and Alona Fyshe[2,3]

[1] Department of Computer Science, Dartmouth College, Hanover, NH 03755
[2] Machine Learning Department, Carnegie Mellon University, Pittsburgh, PA 15213
[3] Center for Neural Basis of Cognition, Carnegie Mellon University, Pittsburgh, PA 15213

Abstract. An open problem in the processing and analysis of neuroimaging data is finding the optimal set of features from which to decode or study underlying neural processes. Handcrafted features sets have been examined in the past with varying degrees of success. Here, we explore sparse autoencoders, a method to extract high-level feature representations of neural data in an unsupervised way, without any manual feature engineering. We pair this unsupervised feature creation step with a L_2 regularized regression, which is robust to overfitting. We apply this method to the task of single word decoding from Magnetoencephalography (MEG) data which is known for its high degree of noise and complexity. We show that, for some subjects, sparse autoencoding is advantageous, though it does not produce an overall increase in decoding accuracy. Our results represent the first step towards a neural-network system that could join feature extraction and decoding into one powerful joint system.

1 Introduction

Magnetoencephalography (MEG) is a brain imaging method that records the magnetic field created as neurons fire in a coordinated fashion. MEG has high temporal resolution (on the order of milliseconds), and can therefore be used to characterize neuronal processes that occur on a rapid time scale. However, it can be difficult to analyze MEG data, due to its characteristically low signal-to-noise ratio (SNR) and the complexity of the signal itself. Identifying the features to be extracted from MEG time series data for subsequent analysis is still an open question. Common feature choices include the slope of fitted lines, variance, kurtosis and spatiofrequency values such as the coefficients of a Fourier or Gabor transform [2, 13].

Unfortunately, there is no guarantee that these hand-picked features are useful or that they complement one another. Thus, it may be wise to consider an alternative paradigm which sidesteps the problem of feature engineering by utilizing an unsupervised algorithm to *learn* features [1]. One such an algorithm is an artificial neural network variant called a Sparse Autoencoder (SAE). Autoencoders have been successfully used for unsupervised feature extraction from photographs of objects such as faces [6], and also for detecting markers of Alzheimer's disease in fMRI brain images [4]. We use an SAE to learn features from MEG time series data and decode semantic content from the MEG signal recorded as a subject reads a noun.

1.1 Sparse Autoencoders

An autoencoder is a neural network trained with backpropagation by gradient descent. Let the input to the autoencoder be of size m, and the hidden layer be of size k. The autoencoder learns a nonlinear approximation of the identity function – the output of the autoencoder, $\hat{\mathbf{X}} = \{\hat{x}_1, \ldots \hat{x}_m\}$,

reconstructs the input, $\mathbf{X} = \{x_1, \ldots x_m\}$, via a (possibly lower dimensional) hidden layer Z (See Fig. 1). Hidden layer nodes take on values $\mathbf{Z} = \{z_1, \ldots z_k\}$, which are weighted sums of the input values and bias. The set of weights between the input and hidden layers is denoted as $\mathbf{W^{(1)}} = \{w_{11}^{(1)}, w_{12}^{(1)}, \ldots, w_{km}^{(1)}\}$, where $w_{ij}^{(1)}$ represents the weight between input node j and hidden node i. The bias terms are $b^{(1)} = \{b_1^{(1)}, b_2^{(1)}, \ldots, b_k^{(1)}\}$, where $b_i^{(1)}$ is the bias of hidden node i. Thus, the value of hidden layer node z_i is defined as

$$z_i = b_i^{(1)} + \sum_{j=1}^{m} w_{ij}^{(1)} x_j \tag{1}$$

Each hidden node outputs activation $a_i = f(z_i)$, where f is an activation function applied to the value z_i. We use a logistic activation function to capture nonlinear relationships:

$$f(z) = \frac{1}{1 + e^{-z}} \tag{2}$$

The output layer $\hat{\mathbf{X}}$ is constructed in a similar way, using the activations of the hidden layer as input, bias $b^{(2)}$ and learned weights $\mathbf{W}^{(2)} = \{w_{11}^{(2)}, w_{12}^{(2)}, ..., w_{km}^{(2)}\}$:

$$\hat{x}_i = f\left(\sum_{j=1}^{k} w_{ij}^{(2)} a_j\right) \tag{3}$$

We use backpropagation via gradient descent to adjust the weights and biases of the connections in the network so as to minimize the difference between the output vectors $\hat{\mathbf{X}}$ and corresponding target vectors [7]. In the general neural network, the targets can be anything, but for an autoencoder, the targets equal the input values.

Autoencoders can implement any of several variations of dimensionality reduction. A linear autoencoder will learn the principal variance directions (Eigenvectors) of the data, equivalent to applying PCA to the inputs [3]. A *nonlinear* autoencoder is capable of discovering more complex, multi-modal structure in the data. Hinton and Salakhutdinov [5] have shown that nonlinear autoencoder networks can produce low-dimensional codes that outperform PCA for handwriting and face recognition. To reveal nonlinear structure in MEG data, we apply a logistic activation function to the nodes that comprise the autoencoder network. Another useful modification to the autoencoder is the sparse coding requirement, i.e. limiting the number of hidden units that can be activated by each input pattern. By imposing constraints on the number of hidden layer nodes and/or the sparsity of the hidden layer representations, we can reduce the input to a feature space that is optimally compressed and enhance the linear separability of the data. A *sparse* autoencoder includes a sparseness constraint on hidden layer activation that is enforced by the addition of the Kullback-Leibler (KL) divergence term to the objective function.

Overall, the objective function $J(\mathbf{W}, \mathbf{B})$ minimized by the SAE is:

$$J(\mathbf{W}, \mathbf{B}) = \frac{1}{2m} \sum_{i=1}^{m} \|\hat{\mathbf{X}}^{(i)} - \mathbf{X}^{(i)}\|^2 + \lambda \|\mathbf{W}\|^2 + \beta \sum_{j=1}^{k} KL(\rho \| \hat{\rho}_j) \tag{4}$$

where

$$KL(\rho \| \hat{\rho}_j) = \rho \log \frac{\rho}{\hat{\rho}_j} + (1 - \rho) \log \frac{1 - \rho}{1 - \hat{\rho}_j} \tag{5}$$

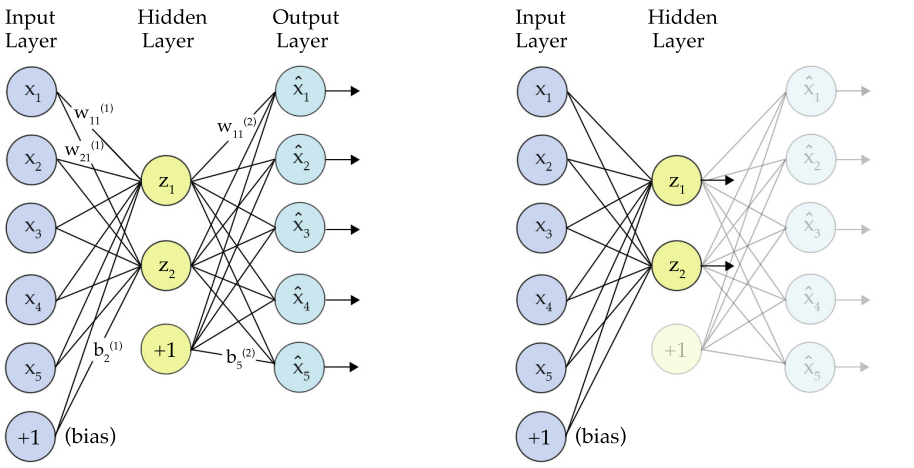

Fig. 1. a. The Sparse Autoencoder during training. The weights of the network are adjusted so as to minimize the difference between the output $\hat{\mathbf{X}}$ and input $\mathbf{X}$ over all training examples. **b. Using a Sparse Autoencoder.** After training, an input example is mapped into feature space by being propagated through the first set of network weights to the hidden layer.

$\hat{\mathbf{X}}$ is the output obtained by propagating input $\mathbf{X}$ through the logistic activation functions in the hidden and output layers (Fig.1). The value ρ is a sparsity parameter we specify; it indicates the desired frequency of activation of the hidden nodes. In the KL divergence term, $\hat{\rho}_j$ is the average thresholded activation of hidden node j over all training examples. For computing the cost function only (not for computing the network output values), we threshold the hidden node activations so that their values become 1 if greater than 0.5 and 0 otherwise. We penalize $\hat{\rho}_j$ terms deviating too far from ρ. λ and β are hyperparameters that determine the relative importance of the weight decay regularization term and the sparseness term in the cost function.

2 Data

MEG data were collected from 9 subjects, who each viewed 60 concrete nouns from 12 categories (animals, tools, buildings, food, furniture, insects, transportation, clothing, body parts, building parts, utensils and objects) along with corresponding line drawings. The MEG signal is collected from groups of 3 sensors at 102 positions on the head for a total of 306 sensors. Each sensor records a time series at 1000 Hz, which is downsampled to 200 Hz.

The MEG data were preprocessed using the Signal Space Separation method (SSS) [11, 9] and temporal extension of SSS (tSSS) [10] to remove artifacts and noise unrelated to brain activity. In addition, we used tSSS to realign the head position measured at the beginning of each block to a common location. The Signal Space Projection method (SSP) [12] was then used to remove signal contamination by eye blinks or movements, as well as MEG sensor malfunctions or other artifacts.

For our analysis, we used the signal recorded for the 750 ms after stimulus onset, the typical amount of time required for semantic processing [8]. Each of the 50 words is presented 20 times (randomly interleaved). All 20 repetitions are averaged together to create a sensor-space feature matrix with 150 time samples for each of the 306 sensors.

In addition, we have 218 semantic features for each of the 60 words. These features are ratings from 1-5 collected from Amazon Mechanical Turk workers in response to questions characterizing the nouns, such as *"Was it ever alive?"* or *"Can you pick it up?"*. Using a set of semantic features allows us to characterize any word in terms of a 218-dimensional semantic vector, and allows us to attribute neural activity to the semantic features of words.

3 Methods

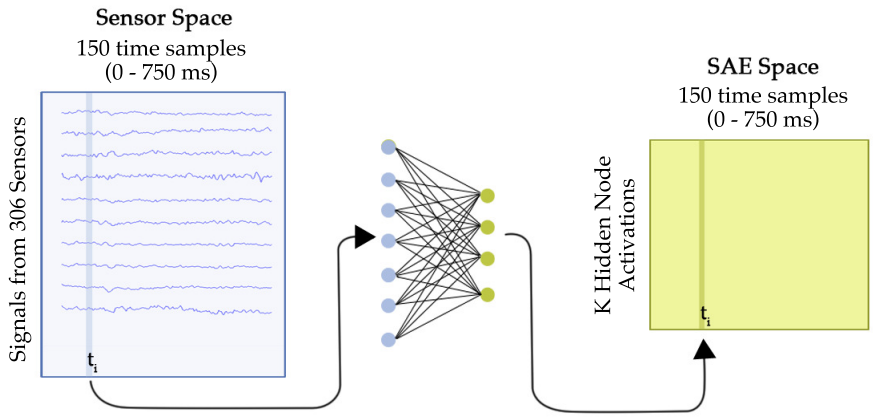

Fig. 2. Creating a SAE space for one word Features are extracted by sequentially propagating the 306 sensor activations at each time point through the hidden layer of the sparse autoencoder.

We used a sparse autoencoder to learn a set of features that describe the activations across all 306 MEG sensors at a single moment in time. That is, we train the SAE to transform a "snapshot" of brain activation ($\mathbf{X} \in \mathbb{R}^{306}$) to a feature space defined by the activations of its hidden nodes ($\mathbf{Z} \in \mathbb{R}^{k}$). We will refer to the input space to which $\mathbf{X}$ belongs as "sensor space" and the feature space to which $\mathbf{Z}$ belongs as "SAE space". We used the trained network to represent an entire MEG time series (the pattern of brain activity as a subject reads and comprehends a single word) by forward propagating the data (from all sensors) at each individual time sample through the hidden layer and then concatenating the hidden layer representations in the original temporal sequence (Fig.2). Thus, we preserve temporal ordering and resolution, while creating a more spatially compact representation of brain activity.

Our objective is to predict from the SAE encoding of the signal which of the 60 nouns the subject was reading. We compare the original sensor space representation to the newly calculated SAE space and ask if SAE space can be used to better decode the word a subject is viewing.

To decode the word a person is viewing, we train 218 independent linear regression models to predict the answers to each of the 218 semantic features (described above) from either the SAE or sensor space representation. We use L_2 regularized regression, or Ridge Regression, to build our linear regression models. L_2 regularization has the property of down-weighting less important features and avoids over-fitting our data [2]. We evaluate word prediction using a Leave-Two-Out test: we train regressors on 58 of the 60 words, and test on the remaining two words. We perform

this test for 150 pairs of words drawn from the 60 words such that each word is chosen an equal number of times. This results in 150 pairs of 218-dimensional *predicted* semantic vectors and 150 pairs of 218-dimensional *true* semantic vectors.

We use two tests to measure predictive performance: the 1 vs. 2 and 2 vs. 2 tests. For these tests, we compute the cosine distance between *predicted* and *true* semantic vectors of the held-out words. In the 1 vs. 2 test, for each held out word, we ask if the word's *predicted* vector is closer to the correct *true* vector than incorrect *true* vector (corresponding to the other held out word). If the correct assignment is the smaller distance, the 1 vs. 2 test is marked correct. In the 2 vs. 2 test, both held out words are matched simultaneously; if the sum of the distances for the correct assignment of true to predicted semantic vectors is less than the sum for the incorrect assignment, the 2 vs. 2 test is marked as correct. Accuracies are the percentage of 1 vs. 2 or 2 vs. 2 tests marked correct.

We used one subject's MEG data exclusively for tuning the hyperparameters λ, β, ρ and k of the model. The data for the other 8 subjects were reserved for the main experiment. We determined by a grid search through candidate values $\lambda = [0.00005, 0.0001, 0.0005, 0.001]$, $\beta = [1, 3, 5]$, $\rho = [0.001, 0.01, 0.05, 0.1, 0.2]$ and $k = [100, 200, 306]$ that 2 vs. 2 and 1 vs. 2 performance was ideal for the values $\lambda = 0.001$, $\beta = 3$, $\rho = 0.05$ and $k = 306$. We observed good performance on some trials with smaller values of k as well, but results were noisy for these trials it was difficult to identify a consistent peak in performance as a function of ρ.

4 Results and Discussion

The results for the 1 vs. 2 and 2 vs. 2 accuracy for the 8 subjects whose data was not used for parameter tuning appear in Table 1. The hyperparameters used for training were $\lambda = 0.001$, $\beta = 3$, $\rho = 0.05$ and $k = 306$ (such that the hidden layer had the same dimensionality as the input layer). The accuracy results for the regressors built on the SAE space do not significantly differ from those for regressors built on the sensor space, according to the Wilcoxon rank sum test. Applying the SAE neither improves nor worsens the decodability of the raw data.

1 vs 2 Accuracy									
	S1	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**	**S8**	**Mean**
Sensor Space	84.33	82.00	85.00	80.33	78.33	78.00	74.67	82.33	80.63
SAE Space	83.33	79.67	83.00	80.67	78.00	78.67	74.00	82.33	79.96

2 vs 2 Accuracy									
	S1	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**	**S8**	**Mean**
Sensor Space	94.67	87.33	95.33	89.33	87.33	83.33	84.00	92.67	89.25
SAE Space	92.00	88.00	95.33	91.33	86.67	83.33	81.33	93.33	88.92

Table 1. Word prediction 1 vs 2 and 2 vs 2 accuracy for sensor and SAE space. Table values indicate the percentage of correct matchings of held-out words.

Transforming MEG data to SAE space did not yield significant improvements in our prediction accuracy tasks; however, it did not significantly worsen the results either. SAE performance could

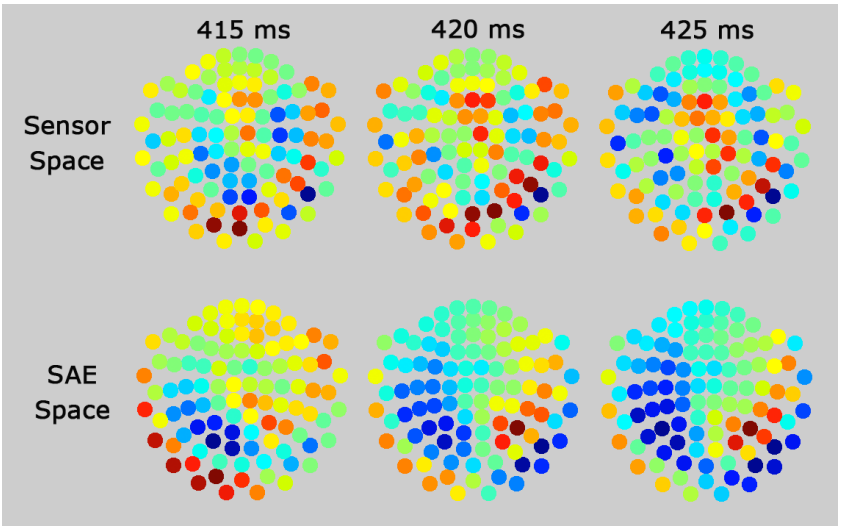

Fig. 3. MEG gradiometer 1 activation patterns in sensor space and SAE space that produce a large value (i.e. strongly yes) for feature "Is it alive?" for subject S1. The sensor layout is as would be seen from above the helmet, with the subject's nose pointing to the top of the diagram. Note that SAE patterns are smoother and more robust to noise.

possibly be improved by establishing more training examples or choosing a more intelligent method of initializing the network weights so that we do not get trapped in poor local minima. Still, even our neutral result may indicate that the SAE could be a useful component of a predictive neural network model. Indeed we see that the learned SAE encodings lead to a smoother weighting of activation patterns for predicting the semantic feature "Is it alive?", as seen in Figure 3.

As our next step, we would like to incorporate the task of semantic feature prediction into the training of the neural network. Then, instead of output that reconstructs the input (as our SAE does) we would produce the semantic features as output. This would allow us to perform unsupervised feature learning geared specifically towards the task of interest. We believe sparse autoencoders and neural networks in general represent a promising method for processing the high SNR data recorded with MEG.

Furthermore, we plan to explore the use of autoencoders with multiple layers ("deep" autoencoders). The addition of multiple layers may improve the SAE representation of MEG data by creating a hierarchy of *higher order* feature detectors. Each layer captures high-order correlations between the activity of the units in the preceding layer, representing progressively more complex nonlinear structure in the inputs [5].

5 Acknowledgements

This research was supported by the Mellon Mays Undergraduate Fellowship and the NIH-sponsored MNTP Fellowship through the Center for the Neural Basis of Cognition.

References

1. Adam Coates, Honglak Lee, and Andrew Y Ng. An Analysis of Single-Layer Networks in Unsupervised Feature Learning. *Engineering*, 1001:1–9, 2011.
2. Alona Fyshe, Gustavo Sudre, Leila Wehbe, Brian Murphy, and Tom Mitchell. Decoding Word Semantics from Magnetoencephalography Time Series Transformations. *2nd NIPS Workshop on Machine Learning and Interpretation in NeuroImaging (MLINI)*, 2012.
3. Rostislav Goroshin and Yann LeCun. Saturating Auto-Encoders. *International Conference on Learning Representations*, 2013.
4. Ashish Gupta, Murat Seckin Ayhan, and Anthony S Maida. Natural Image Bases to Represent Neuroimaging Data. *Journal of Machine Learning Research: Workshop and Conference Proceedings*, 28(3):977–984, 2013.
5. G.E. Hinton and R.R. Salakhutdinov. Reducing the Dimensionality of Data with Neural Networks. *Science (New York, N.Y.)*, 313:504–507, 2006.
6. Quoc V Le, Marc Aurelio Ranzato, Matthieu Devin, Greg S Corrado, and Andrew Y Ng. Building High-level Features Using Large Scale Unsupervised Learning. *Proceedings of the 29th International Conference on Machine Learning*, 2012.
7. David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.
8. Gustavo Sudre, Dean Pomerleau, Mark Palatucci, Leila Wehbe, Alona Fyshe, Riitta Salmelin, and Tom Mitchell. Tracking neural coding of perceptual and semantic features of concrete nouns. *NeuroImage*, 62(1):451–63, August 2012.
9. S Taulu and J Simola. Spatiotemporal signal space separation method for rejecting nearby interference in meg measurements. *Physics in Medicine and Biology*, 51:1–10, 2006.
10. Sami Taulu and Riitta Hari. Removal of magnetoencephalographic artifacts with temporal signal-space separation: demonstration with single- trial auditory-evoked responses. *Human brain mapping*, 30(5):1524–34, May 2009.
11. Samu Taulu, Matti Kajola, and Juha Simola. The signal space separation method. *ArXiv Physics*, 2004.
12. M.A. Uusitalo and R.J. Ilmoniemi. Signal-space projection method for separating meg or eeg into components. *Medical and biological engineering and computing*, 35(2):135–40, March 1997.
13. Wojciech Zaremba. *Modeling the variability of EEG/MEG data through statistical machine learning.* Master's thesis, École Polytechnique, University of Warsaw, 2012.