DeepBrain: Functional Representation of Neural In-Situ Hybridization Images for Gene Ontology Classification Using Deep Convolutional Autoencoders

Ido Cohen[1], Eli (Omid) David[1(✉)], Nathan S. Netanyahu[1,2], Noa Liscovitch[3], and Gal Chechik[3]

[1] Department of Computer Science, Bar-Ilan University, Ramat-Gan, Israel
cido15@gmail.com, mail@elidavid.com, nathan@cs.biu.ac.il,
nathan@cfar.umd.edu
[2] Center for Automation Research, University of Maryland, College Park, MD, USA
[3] Gonda Multidisiplinary Brain Research Center, Bar-Ilan University,
Ramat-Gan, Israel
noalis@gmail.com, gal.chechik@mail.biu.ac.il

Abstract. This paper presents a novel deep learning-based method for learning a functional representation of mammalian neural images. The method uses a *deep convolutional denoising autoencoder* (CDAE) for generating an invariant, compact representation of *in situ hybridization* (ISH) images. While most existing methods for bio-imaging analysis were not developed to handle images with highly complex anatomical structures, the results presented in this paper show that functional representation extracted by CDAE can help learn features of functional *gene ontology categories* for their classification in a highly accurate manner. Using this CDAE representation, our method outperforms the previous state-of-the-art classification rate, by improving the average AUC from 0.92 to 0.98, i.e., achieving 75% reduction in error. The method operates on input images that were downsampled significantly with respect to the original ones to make it computationally feasible.

1 Introduction

A very large volume of high-spatial resolution imaging datasets is available these days in various domains, calling for a wide range of exploration methods based on image processing. One such dataset has become recently available in the field of Neuroscience, thanks to the Allen Institute for Brain Science. This dataset contains *in situ hybridization* (ISH) images of mammalian brains, in unprecedented amounts, which has motivated new research efforts [3,11,12]. ISH is a powerful technique for localizing specific nucleic acid targets within fixed tissues and cells; it provides an effective approach for obtaining temporal and spatial information about gene expression [16]. Images now reveal highly complex patterns of gene expression varying on multiple scales.

However, analytical tools for discovering gene interactions from such data remain an open challenge due to various reasons, including difficulties in extracting canonical representations of gene activities from images, and inferring statistically meaningful networks from such representations. The challenge in analyzing these images is both in extracting the patterns that are most relevant functionally, and in providing a meaningful representation that allows neuroscientists to interpret the extracted patterns.

One of the aims at finding a meaningful representation for such images, is to carry out classification to *gene ontology* (GO) categories. GO is a major Bioinformatics initiative to unify the representation of gene and gene product attributes across all species [5]. More specifically, it aims at maintaining and developing a controlled vocabulary of gene and gene product attributes and at annotating them. This task is far from done; in fact, several gene and gene product functions of many organisms have yet to be discovered and annotated [14]. Gene function annotations, which are associations between a gene and a term of controlled vocabulary describing gene functional features, are of paramount importance in modern biology. They are used to design novel biological experiments and interpret their results. Since gene validation through in vitro biomolecular experiments is costly and lengthy, deriving new computational methods and software for predicting and prioritizing new biomolecular annotations, would make an important contribution to the field [20]. In other words, deriving an effective computational procedure that predicts reliably likely annotations, and thus speed up the discovery of new gene annotations, would be very useful [9].

Past methods for analyzing brain images had to reference a brain atlas, and based on smooth non-linear transformations [10,13]. These types of analyses may be insensitive to fine local patterns, like those found in the layered structure of the cerebellum[1], or to spatial distribution. In addition, most machine vision approaches address the challenge of providing human interpretable analysis. Conversely, in bioimaging usually the goal is to reveal features and structures that are hardly seen even by human experts. For example, one of the new functions that follow this approach is presented in [17], using a histogram of local *scale-invariant feature transform* (SIFT) [8] descriptors on several scales.

Recently, many machine learning algorithms have been designed and implemented to predict GO annotations [1,7,15,18,22]. In our research, we examine an *artificial neural network* (ANN) with many layers (also known as *deep learning*) in order to achieve functional representations of neural ISH images.

In order to find a compact representation of these ISH images, we explored *autoencoders* (AE) and *convolution neural networks* (CNN), and found the *convolutional autoencoder* (CAE) to be the most appropriate technique. Subsequently, we use this representation to learn features of functional GO categories for every image, using a simple *support vector machine* (SVM) classifier [4], as in [17]. As a result, each image is represented as a point in a lower-dimensional space whose axes correspond to meaningful functional annotations. A similar example

---

[1] The cerebellum is a region of the brain. It plays an important role in motor control, and has some effect on cognitive functions [23].

to ours is the work of Krizhevsky and Hinton [2], who used deep autoencoders to create short binary codes for content-based images. The resulting representations define similarities between ISH images which can be easily explained, hopefully, by such functional categories.

Our experimental results demonstrate that a so-called *convolutional denoising autoencoder* (CDAE) representation (see Subsect. 3.2) outperforms the previous state-of-the-art classification rate, by improving the average AUC from 0.92 to 0.98, i.e., achieving 75% reduction in error. The method operates on input images that were downsampled significantly with respect to the original ones to make it computationally feasible.

2   Background

2.1   FuncISH - Learning Functional Representations

ISH images of mammalian brains reveal highly complex patterns of gene expression varying on multiple scales. Our study follows [17], which we pursue using deep learning. In [17] the authors present *FuncISH*, a learning method of functional representations of ISH images, using a histogram of local descriptors on several scales.

They first represent each image as a collection of local descriptors using SIFT features. Next, they construct a standard bag-of-words description of each image, giving a 2004-dimension representation vector for each gene. Finally, given a set of predefined GO annotations of each gene, they train a separate classifier for each known biological category, using the SIFT bag-of-words representation as an input vector. Specifically, they used a set of 2081 L_2-regularized logistic regression classifiers for this training. A scheme representing the work flow is presented in Fig. 2 (see Sect. 4).

Applying their method to the genomic set of mouse neural ISH images available from the Allen Brain Atlas, they found that most neural biological processes could be inferred from spatial expression patterns with high accuracy. Despite ignoring important global location information, they successfully inferred ∼700 functional annotations, and used them to detect gene-gene similarities which were not captured by previous, global correlation-based methods. According to [17], combining local and global patterns of expression is an important topic for further research, e.g., the use of more sophisticated non-linear classifiers.

2.2   Deep Learning Techniques

Pursuing further the above classification problem poses a number of challenges. First, we cannot define a certain set of rules that an ISH image has to conform to in order to classify it to the correct GO category. Therefore, conventional computer vision techniques, capable of identifying shapes and objects in an image, are not likely to provide effective solutions to the problem. Thus, we use deep learning to achieve better results, as far as functional representations of the

ISH images. This yields an interpretable measure of similarity between complex images that are difficult to analyze and interpret.

Deep learning techniques that support this kind of problems use AE and CNN, as well as CAE, which are successful in preforming feature extraction and finding compact representations for the kind of large ISH images we have been dealing with. While traditional machine learning is useful for algorithms that learn iteratively from the data, our second issue concerns the type of data we possess. Our data consist of 16K images, representing about 15K different genes, i.e., an average of one image per gene. This prevents us from extracting features from each gene independently, but rather consider the data in their entirety. Moreover, not only is there only one image per gene, there are merely a few genes in every examined GO category, and the genes are not unique to one category, i.e., each gene may belong to more than one category. Despite these difficulties, machine learning is capable of capturing underlying "insights" without resorting to manual feature selection. This makes it possible to automatically produce models that can analyze larger and more complex data, achieving thereby more accurate results.

In the next section we present our convolutional autoencoder approach, which operates solely on raw pixel data. This supports our main goal, i.e., learning representations of given brain images to extract useful information, more easily, when building classifiers or other predictors. The representations obtained are vectors which can be used to solve a variety of problems, e.g., the problem of GO classification. For this reason, a good representation is also one that is useful as input to a supervised predictor, as it allows us to build classifiers for the biological categories known.

3 Feature Extraction Using Convolutional Autoencoders

3.1 Auto-Encoders (AE)

While convolutional neural networks (CNN) are effective in a supervised framework, provided a large training set is available, this is incompatible to our case. If only a small number of training samples is available, unsupervised pre-training methods, such as *restricted Boltzmann machines* (RBM) [21] or autoencoders [24], have proven highly effective.

An AE is a neural network which sets the target values (of the output layer) to be equal to those of the input, using hidden layers of smaller and smaller size, which comprise a bottleneck. Thus, an AE can be trained in an unsupervised manner, forcing the network to learn a higher-level representation of the input. An improved approach, which outperforms basic autoencoders in many tasks is due to *denoising autoencoders* (DAEs) [24,25]. These are built as regular AEs, where each input is corrupted by added noise, or by setting to zero some portion of the values. Although the input sample is corrupted, the network's objective is to produce the original (uncorrupted) values in the output layer. Forcing the network to recreate the uncorrupted values results in reduced network overfitting (also due to the fact that the network rarely receives the same input twice), and

in extraction of more high-level features. For any autoencoder-based approach, once training is complete, the decoder layer(s) are removed, such that a given input passes through the network and yields a high-level representation of the data. In most implementations (such as ours), these representations can then be used for supervised classification.

3.2 Convolutional Autoencoders (CAE)

CNNs and AEs can be combined to produce CAEs. As with CNNs, the CAE weights are shared among all locations in the input, preserving spatial locality and reducing the number of parameters. In practice, to combine CNNs with AEs (or DAEs), it is necessary for each encoder layer to have a corresponding decoder layer. *Deconvolution* layers are essentially the same as convolutional layers, and similarly to standard autoencoders, they can either be learned or set equal to (the transpose of) the original convolution layers, as with tied weights in autoencoders (both work well). For the *unpooling* operation, more than one method exists [6,19]. In the CAE we use, during unpooling all locations are set to the maximum value which is stored in that layer (Fig. 1).
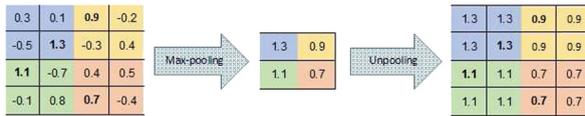

Fig. 1. Pooling and unpooling layers; for each pooling layer, the max value is kept, and then duplicated in the unpooling layer.

Similarly to an AE, after training a CAE, the unpooling and deconvolution layers are removed. At this point, a neural net, composed from convolution and pooling layers, can be used to find a functional representation, as in our case, or initialize a supervised CNN. Similarly to a DAE, a CAE with input corrupted by added noise is called a *convolutional denoising autoencoders* (CDAE).

4 CDAE for GO Classification

Figure 2 depicts a framework for capturing the representation of FuncISH. A SIFT-based module was used in [17] for feature extraction. Alternatively, our scheme learns a CDAE-based representation, before applying a similar classification method as in [17], where two layers of 5-fold cross-validation were used, one for training the classifier and the other for tuning the logistic regression regularization hyperparameter.

For unsupervised training of our CDAE we use the genomic set of mouse neural ISH images available from the Allen Brain Atlas, which includes 16,351 images representing 15,612 genes. These JPEG images have an average resolution
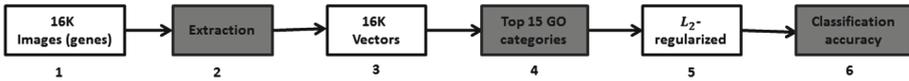
Fig. 2. (1) 16K grayscale images indicating level of gene expression, (2) SIFT- or CDAE-based feature extraction for a compact vector representation of each gene, (3) vector representation due to feature extraction, (4) 16K vectors trained with respect to each of 15 GO categories with best AUC classification in [17], (5) L_2-regularized logistic regression classifiers for top 15 GO categories, and (6) classification accuracy measured.

of $15,000 \times 7,500$ pixels. To get a representation vector of size $\sim 2,000$, the images were downsampled to 300×140 pixels.

The CDAE architecture for finding a compact representation for these downsampled images is as follow: (1) **Input layer**: Consists of the raw image, resampled to 300×140 pixels, and corrupted by setting to zero 20% of the values, (2) three sequential convolutional layers with 32 9×9 filters each, (3) max-pooling layer of size 2×2, (4) three sequential convolutional layers with 32 7×7 filters each, (5) max-pooling layer of size 2×2, (6) two sequential convolutional layers with 64 5×5 filters each, (7) convolutional layer with a single 5×5 filter, (8) unpooling layer of size 2×2, (9) three sequential deconvolution layers with 32 7×7 filters each, (10) unpooling layer of size 2×2, (11) three sequential deconvolution layers with 32 9×9 filters each, (12) deconvolution layer with a single 5×5 filter, and (13) **output layer** with the uncorrupted resampled image.

After training the CDAE, all layers past item 8 are removed, so that item 7 (the convolutional layer of size $1 \times 2,625$) becomes the output layer. Therefore, each image is mapped to a vector of 2,625 functional features. Given a set of predefined GO annotations for each gene (where each GO category consists of 15–500 genes), we trained a separate classifier for each biological category. Training requires careful consideration, in this case, due to the vastly imbalanced nature of the training sets. Similarly to [17], we performed a weighted SVM classification using 5-fold cross-validation.

This network yields remarkable AUC results for every category of the top 15 GO categories reported in [17]. Figure 3 illustrates the AUC scores achieved for various representation vectors. While the average AUC score (of the top 15 categories) reported in [17] was **0.92**, the average AUC using our CDAE scheme was **0.98**, i.e., a **75%** reduction in error.

4.1   Reducing Vector Dimensionality

The above improvement was achieved with a vector size of 2,625, which is larger than the 2004-dimensional vector obtained by SIFT. In an attempt to maintain, as much as possible, the scheme's performance for a comparable vector size, we explored the use of smaller vectors, by resampling the images to different scales, and constructing CDAEs with various numbers of convolution and pooling layers.

Figure 3(b) shows the average AUC for the top 15 categories mentioned earlier, with the same CDAE structure and the images resampled to smaller scales, thus obtaining lower-dimensionality representation vectors.

Downsampling to 240×120 images, we obtained a 1800-dimensional representation vector, for which the AUC scores are still superior (relatively to [17]) for each of the top 15 GO categories (as shown in Fig. 3(a)). The **10%**-dimensionality reduction results only in a slightly lower AUC average of **0.97** (see Fig. 3(b)).

The CDAE network for the more compact representation is shown in Fig. 4. The architecture consists of the following layers: (1) **Input layer**: consists of the raw image, resampled to 240×120 pixels, and corrupted by setting to zero 20% of the values, (2) four sequential convolutional layers with 16 3×3 filters each, (3) max-pooling layer of size 2×2, (4) four sequential convolutional layers with 16 3×3 filters each, (5) max-pooling layer of size 2×2, (6) three sequential convolutional layers with 16 3×3 filters each, (7) convolutional layer with a single 3×3 filter, (8) unpooling layer of size 2×2, (9) four sequential deconvolution layers with 16 3×3 filters each, (10) unpooling layer of size 2×2, (11) four sequential deconvolution layers with 16 3×3 filters each, (12) deconvolution layer with a single 3×3 filter, and (13) **output layer** with the uncorrupted resampled image.

We used the ReLU activation function for all convolution and deconvolution layers, except for the last deconvolution layer, which uses tanh.
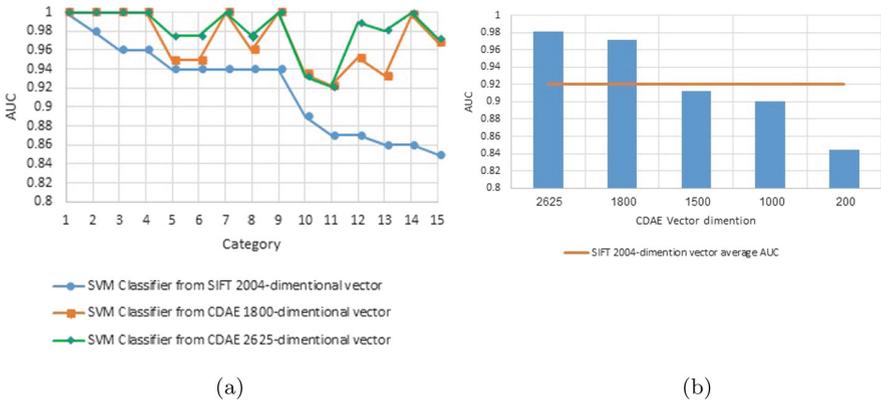

(a)   (b)

Fig. 3. AUC results using convolutional denoising autoencoder for feature extraction: (a) AUC obtained from training a SVM classifier for each GO category, using a compact representation vector for every gene; representation vector dimensionality depends on method used and image resampling rate; (b) average AUC for top 15 classifiers, trained on different representation vectors due to CDAE (as in Fig. 4), for different resampling of brain images.
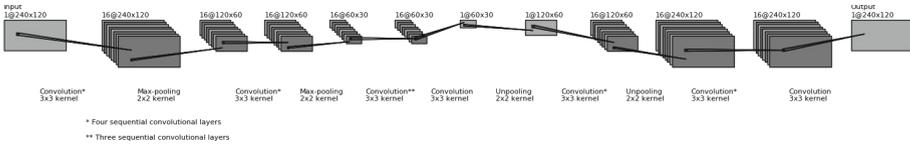
Fig. 4. Illustration of our convolutional denoising autoencoder, achieving a compact representation for each gene.

The learning rate starts from 0.05 and is multiplied by 0.9 after each epoch, and the denoising effect is obtained by randomly removing 20% of the pixels every image in the input layer. We used the AUC as a measure of classification accuracy.

5   Conclusion

Many machine learning algorithms have been designed lately to predict GO annotations. For the task of learning functional representations of mammalian neural images, we used deep learning techniques, and found convolutional denoising autoencoder to be very effective. Specifically, using the presented scheme for feature learning of functional GO categories improved the previous state-of-the-art classification accuracy from an average AUC of 0.92 to 0.98, i.e., a 75% reduction in error. We demonstrated how to reduce the vector dimensionality by 10% compared to the SIFT vectors, with very little degradation of this accuracy. Our results further attest to the advantages of deep convolutional autoencoders, as were applied here to extracting meaningful information from very high resolution images and highly complex anatomical structures. Until gene product functions of all species are discovered, the use of CDAEs may well continue to serve the field of Bioinformatics in designing novel biological experiments.

Appendix: Network Architecture Description

We provide a brief explanation as to the choice of the main parameters of the CDAE architecture. Our objective was to obtain a more compact feature representation than the 2,004-dimensional vector used in FuncISH. Since a CNN is used, the representation along the grid should capture the two-dimensional structure of the input, i.e., the image dimensions should be determined according to the intended representation vector, while maintaining the aspect ratio of the original input image. Thus, we picked an 1,800-dimensional feature vector, corresponding to an (output) image of size 60×30. Taking into account the characteristic of max-pooling (i.e., that at each stage the dimension is reduced by 2), the desire to keep the number of layers as small as possible, and the fact that the encoding and decoding phases each contains the same number of layers (resulting in twice the number of layers in the network), we settled for two max-pooling layers, namely an input image of size 240×120. Between each two

max-pooling layers, which eliminate feature redundancy, there is an "array" of 16 convolution layers, each with the purpose of detecting locally connected features from its previous layer. The number of convolution layers (i.e., different filters used) was determined after experimenting with several different layers, all of which gave similar results. Choosing 16 layers (as shown in Fig. 4) provided the best result. We experimented also with various filter sizes for each layer, ranging from 3×3 to 11×11; while increasing the filter size significantly increased the amount of network parameters learned, it did not contribute much to the feature extraction or the improvement of the results. Using a learning rate decay in the training of large networks (where there is a large number of randomly generated parameters) has proven helpful in the network's convergence. Specifically, the combination of a 0.05 learning rate parameter with a 0.9 learning rate decay resulted in an optimal change of the parameter value. In this case, too, small changes in the parameters did not result in significant changes in the results.

References

1. Kordmahalleh, M.M., Homaifar, A., Dukka, B.K.C.: Hierarchical multi-label gene function prediction using adaptive mutation in crowding niching. In: Proceedings of IEEE International Conference on Bioinformatics and Bioengineering, pp. 1–6 (2013)
2. Krizhevsky, A., Hinton, G.E.: Using very deep autoencoders for content-based image retrieval. In: Proceedings of European Symposium on Artificial Neural Networks (2011)
3. Henry, A.M., Hohmann, J.G.: High-resolution gene expression atlases for adult and developing mouse brain and spinal cord. Mamm. Genome **23**, 539–549 (2012)
4. Cortes, C., Vapnik, V.: Support vector networks. Mach. Learn. **20**(3), 273–297 (1995)
5. The Gene Ontology Consortium: The gene ontology project in 2008. Nucleic Acids Res. **36**, D440–D444 (2008)
6. Masci, J., Meier, U., Cireşan, D., Schmidhuber, J.: Stacked convolutional autoencoders for hierarchical feature extraction. In: Honkela, T., Duch, W., Girolami, M., Kaski, S. (eds.) ICANN 2011. LNCS, vol. 6791, pp. 52–59. Springer, Heidelberg (2011). doi:10.1007/978-3-642-21735-7_7
7. Pinoli, P., Chicco, D., Masseroli, M.: Computational algorithms to predict gene ontology annotations. BMC Bioinform. **16**(6), S4 (2015)
8. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. **60**(2), 91–110 (2004)
9. Skunca, N., du Plessis, L., Dessimoz, C.: The what, where, how and why of gene ontology-a primer for bioinformaticians. Briefings Bioinform. **12**(6), 723–735 (2011)
10. Hawrylycz, M., Ng, L., Page, D., Morris, J., Lau, C., Faber, S., Faber, V., Sunkin, S., Menon, V., Lein, E., Jones, A.: Multi-scale correlation structure of gene expression in the brain. Neural Netw. **24**, 933–942 (2011)
11. Lein, E.S., et al.: Genome-wide atlas of gene expression in the adult mouse brain. Nature **445**, 168–176 (2007)
12. Ng, L., et al.: An anatomic gene expression atlas of the adult mouse brain. Nat. Neurosci. **12**, 356–362 (2009)

13. Davis, F.P., Eddy, S.R.: A tool for identification of genes expressed in patterns of interest using the allen brain atlas. Bioinformatics **25**, 1647–1654 (2009)
14. Ashburner, M., Ball, C.A., Blake, J.A., Botstein, D., Butler, H., Cherry, J.M.: Gene ontology: tool for the unification of biology. Nat. Genet. **25**(1), 25–29 (2000)
15. King, O.D., Foulger, R.E., Dwight, S.S., White, J.V., Roth, F.P.: Predicting gene function from patterns of annotation. Genome Res. **13**(5), 896–904 (2013)
16. Puniyani, K., Xing, E.P.: GINI: from ISH images to gene interaction networks. PLoS Comput. Biol. **9**, 10 (2013)
17. Shalit, U., Liscovitch, N., Chechik, G.: FuncISH: learning a functional representation of neural ISH images. Bioinformatics **29**(13), i36–i43 (2013)
18. Zitnik, M., Zupan, B.: Matrix factorization-based data fusion for gene function prediction in baker's yeast and slime mold. In: Proceedings of Pacific Symposium on Biocomputing, pp. 400–411 (2014)
19. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8689, pp. 818–833. Springer, Cham (2014). doi:10.1007/978-3-319-10590-1_53
20. Bork, P., Thode, G., Perez, A.J., Perez-Iratxeta, C., Andrade, M.A.: Gene annotation from scientific literature using mappings between keyword systems. Bioinformatics **20**(13), 2084–2091 (2004)
21. Hinton, G.E., Osindero, S., Teh, Y.W.: A fast learning algorithm for deep belief nets. Neural Comput. **18**(7), 1527–1554 (2006)
22. Vembu, S., Morris, Q.: An efficient algorithm to integrate network and attribute data for gene function prediction. In: Proceedings of Pacific Symposium on Biocomputing, pp. 388–399 (2014)
23. Rapoport, M.J., Wolf, U., Schweizer, T.A.: Evaluating the affective component of the cerebellar cognitive affective syndrome. J. Neuropsychol. Clin. Neurosci. **21**(3), 245–253 (2009)
24. Vincent, P., Larochelle, H., Bengio, Y., Manzagol, P.A.: Extracting and composing robust features with denoising autoencoders. In: Proceedings of the 25th International Conference on Machine learning, pp. 1096–1103 (2008)
25. Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.: Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion. J. Mach. Learn. Res. **11**, 3371–3408 (2010)