Neuroinformatics
https://doi.org/10.1007/s12021-018-9361-5

ORIGINAL ARTICLE

Large-scale Exploration of Neuronal Morphologies Using Deep Learning and Augmented Reality

Zhongyu Li[1] · Erik Butler[1] · Kang Li[2] · Aidong Lu[1] · Shuiwang Ji[3] · Shaoting Zhang[1]

Abstract

Recently released large-scale neuron morphological data has greatly facilitated the research in neuroinformatics. However, the sheer volume and complexity of these data pose significant challenges for efficient and accurate neuron exploration. In this paper, we propose an effective retrieval framework to address these problems, based on frontier techniques of deep learning and binary coding. For the first time, we develop a deep learning based feature representation method for the neuron morphological data, where the 3D neurons are first projected into binary images and then learned features using an unsupervised deep neural network, i.e., stacked convolutional autoencoders (SCAEs). The deep features are subsequently fused with the hand-crafted features for more accurate representation. Considering the exhaustive search is usually very time-consuming in large-scale databases, we employ a novel binary coding method to compress feature vectors into short binary codes. Our framework is validated on a public data set including 58,000 neurons, showing promising retrieval precision and efficiency compared with state-of-the-art methods. In addition, we develop a novel neuron visualization program based on the techniques of augmented reality (AR), which can help users take a deep exploration of neuron morphologies in an interactive and immersive manner.

Keywords Neuron morphology · Large-scale retrieval · Deep learning · Binary coding · Augmented reality

Introduction

Investigating neuron morphology is an important topic in neuroscience since morphology plays a major role in determining neurons' connectivity and functional property. Recent advances in neuroscience such as NeuroMorpho (http://neuromorpho.org) and BigNeuron (https://github.com/BigNeuron) have developed multiple reconstruction/tracing techniques for the research of neuron morphology, resulting in an increasing number of reconstructed neurons that are added to these public repositories. However, the large-scale data size and the complex neuron morphologies prevent the realization of the full potential of these data, e.g., efficiently finding neurons sharing similar morphologies, identifying neuron types, correlating neuron morphologies with properties, all of which require a deep and exhaustive exploration of neuron morphologies.

In recent years, multiple neuron morphological retrieval methods have been developed to address the above problems. For example, Wan et al. (2015) designed *BlastNeuron* for automated comparison, retrieval and clustering of 3D neuron morphologies. In the retrieval stage, the *BlastNeuron* first employed several quantitative measurements (Costa et al. 2010) as features to represent each neuron, then searching similar neurons via the normalization of the ranked scores in terms of the similarity of feature vectors. Subsequently, Conjeti et al. (2016a, b) and Mesbah et al. (2015) proposed a hashing based neuron morphological retrieval framework, i.e., Hashing Forest, to search among large neuron databases by generating binary codes from the quantitative measurements. Li et al. (2017b) developed an efficient neuron retrieval framework based on the maximum inner product search (MIPS) and feature hierarchy, where features can be grouped with different similarity levels and compressed into short binary codes

✉ Shaoting Zhang
rutgers.shaoting@gmail.com

[1] Department of Computer Science, University of North Carolina at Charlotte, Charlotte, NC 28223, USA

[2] Department of Industrial and Systems Engineering, The State University of New Jersey, Piscataway, NJ 08854, USA

[3] School of Electrical Engineering and Computer Science, Washington State University, Pullman, WA 99164, USA

Published online: 12 February 2018

for more accurate retrieval. More recently, they designed an interactive retrieval strategy to explore the continuously expanding neuron databases (Li et al. 2017c). The online binary coding can efficiently tackle the continuously expanding databases, and the users' feedback can improve the initial retrieval results with higher accuracy.

Despite the above methods have achieved good performance in many neuron retrieval and analytical tasks, there are still challenges that need to be addressed, particularly at present time that neuroscience is transiting into the era of Big Data. Firstly, current feature representation methods are based on the quantitative measurements of neurons, which are designed for the previous small-sized data sets. These hand-crafted features cannot fully represent neurons and are not suitable for the large-scale and continuously expanding neuron databases. Secondly, large-scale databases require more efficient and accurate retrieval algorithms. Differentiating the fine-grained difference among massive neurons, and searching similar samples on-the-fly, both are tough problems in large-scale. Thirdly, there are still no specific tools developed, which can immersively visualize the retrieved neurons and help users implement a comprehensive analysis of neuron morphologies.

To alleviate these challenges, we develop a novel framework for accurate and efficient neuron analysis in large-scale databases. As shown in Fig. 1, for the original neuron data, we first design a neuronal projection method to transform 3D neurons into 2D binary images with three angles of view. Afterwards, a stacked convolutional autoencoders (SCAEs) (Masci et al. 2011) is introduced to automatically learn deep features from these 2D images. Subsequently, we fuse the learned deep features and the traditional hand-crafted features together as the representation of each neuron. To further improve the retrieval efficiency, binary coding is employed which can compress the fused feature vectors into short binary codes, and also preserve the similarity among original features. According to Fig. 1, our framework can be separated into four steps, i.e., neuronal projection, SCAEs based deep feature learning, feature fusion, and binary coding. For a query neuron that needs to search similar neurons in a database, its original neuron data can be sequentially processed and represented by tens of bits of binary codes. Then the similarity searching can be treated as the Hamming distance ranking between the binary codes of query neuron and each neuron in the database. Additionally, to help users take a deep understanding of neuron morphologies, we develop a novel neuron visualization program through the Microsoft HoloLens augmented reality (AR) headset. To the best of our knowledge, this is the first work that applied deep learning and augmented reality techniques for the analytics of 3D neuron data.

The remaining paper is organized as follows: "Related Work" briefly reviews relevant frontiers of neuron morphology and large-scale image retrieval. Section "Methodology" provides the algorithm details of deep feature representation and binary coding. Followed by experimental results and discussions in the Section of "Experiment". Section "Augmented Reality for Neuron Visualization" introduces our developed AR program for neuron visualization. Finally, "Conclusions" concludes the paper and presented future works.

Related Work

This paper mainly focuses on the neuron morphological retrieval in large databases. In this section, we briefly introduce related works on neuron morphology and large-scale image retrieval.

Neuron Morphology

Neuron morphology is an important topic in the field of neuroscience. It has a strong relevance with neuron's particular properties, such as brain regions, cell types, developmental stages, etc. In recent years, the well-developed

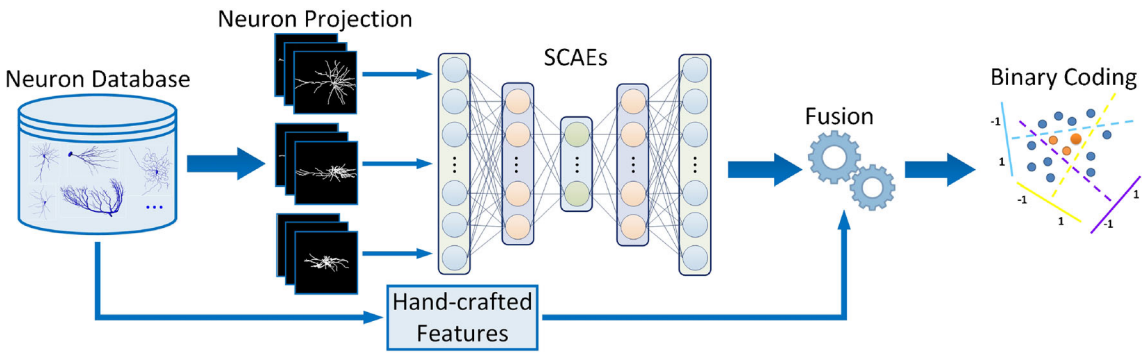

Fig. 1 Overview of our proposed framework, including four steps, i.e., neuronal projection, SCAEs based deep feature learning, feature fusion, and binary coding

microscopy and tracing techniques have greatly improved the study of neuron morphology (Zhou et al. 2015, 2016; Mukherjee et al. 2013; Ji 2013; Wu et al. 2011; Li et al. 2017a), where 3D neurons are reconstructed with high precision. For the 3D neuron data, the tree-like structure makes them possible to be described by some pre-defined quantitative measurements. For example, Costa et al. (2010) proposed the concept of neuromorphological space, where a set of measurements were designed to represent the neuron morphology. Scorcioni et al. (2008) developed L-Measure tool for the quantitative characterization of neuroanatomical parameters.

The above measurements have been adopted as morphological features and achieved good performance in many neuron analytical tasks (Wan et al. 2015; Conjeti et al. 2016b; Le et al. 2016, 2017b, c; Costa et al. 2016). However, these hand-crafted features cannot fully represent the morphology of neurons, particularly when tackling large-scale databases, as there may be outliers and cases not covered by standardized rules. Moreover, the complicated neuronal structure makes hand-crafted features even harder to differentiate and identify the difference among neurons. Accordingly, more effective feature representation methods are required for the large-scale neuron morphological retrieval.

Large-scale Image Retrieval

In recent years, image and video analytics in large-scale have become a hot topic in the field of computer vision, machine learning and high-performance computing. Bunch algorithms and techniques have been developed to improve the performance of large-scale analytics (Yan et al. 2014a, b, c; Li et al. 2018).

Generally, image retrieval can be divided into two stages, i.e., feature representation and feature indexing. In the feature representation part, deep neural networks are the most popular methods that particularly suitable for the large-scale data sets (LeCun et al. 2015), since massive data can boost the retrieval performance by training deep and complex neural networks. Convolutional neural network (CNN) (LeCun et al. 1998) is a type of network that has been widely employed for large-scale retrieval in recent years. Representative methods include AlexNet (Krizhevsky et al. 2012), GoogLeNet (Szegedy et al. 2015), VGGNet (Simonyan and Zisserman 2014) and ResNets (He et al. 2016). The above deep neural networks require supervised information during training. When the image labels are unavailable, unsupervised neural networks can be employed for feature representation, e.g., Stacked Autoencoder (SAE) (Bengio et al. 2007), Deep Belief Network (DBN) (Hinton and Salakhutdinov 2006) and Deep Boltzmann Machine (DBM) (Salakhutdinov 2015). Despite deep neural networks becoming the benchmark in many large-scale image analytical tasks (e.g., natural images), there are still no relevant works that focused on the deep feature representation of neuron morphological data.

For the feature indexing in large-scale, the key problem is computational complexity, i.e., similarity searching in ten thousands of images with thousands dimensional of features. In recent years, binary coding/hashing has become a kind of most popular methods for indexing similar samples in massive data sets (Wang et al. 2016). By compressing long image features into short binary codes and keeping their original similarities, the computational complexity will be significantly reduced. In the field of biomedical informatics, multiple binary coding methods have been applied for large-scale retrieval, including Iterative Quantization (ITQ) (Gong et al. 2013; Liu et al. 2017), Kernel-Based Supervised Hashing (KSH) (Liu et al. 2012; Zhang et al. 2015b, c), Hashing Forest (Yu and Yuan 2014; Conjeti et al. 2016a), MIPS (Shen et al. 2015; Li et al. 2017b), etc. In this work, we need to explore suitable binary coding method that can tackle the label unavailable and linearly inseparable neuron data.

Methodology

In this section, we present the theoretical and technical details of our neuron analytical framework, including neuronal projection, deep feature representation, feature fusion, and binary coding.

Feature Representation for Neuron Morphology

Transforming 3D Neuron into Images As the traditional hand-crafted features are insufficient to represent each neuron in large-scale databases, we seek for a new avenue for the neuron feature representation, based on recent advances in deep learning. Considering the original neuron morphological datasets (i.e., the *SWC format files* Cannon et al. 1998) provide the spatial coordinate for each point, an intuitive solution is to employ 3D deep neural networks that can directly learn deep features from 3D point sets. Unfortunately, this approach is hard to implement due to three reasons: (1) training 3D neural networks are usually very time-consuming, particularly when tackling large neuron databases; (2) the 3D point sets in each neuron are composed based on the tree-topological structure, which are extremely sparse in 3D space; (3) Neurons have dramatically different scales and different numbers of point sets (from hundreds to tens of thousands) that cannot be processed in a generalized framework. How to adapt the 3D neuron data with a suitable modality for deep learning is a critical step in feature representation.

Taking the above problems into account, we first propose a method to transform 3D neurons into 2D binary images and also preserve their primary morphologies. Currently, 3D neuromorphological data are stored in the *SWC* format file with hundreds to tens of thousands of points (Cannon et al. 1998). For each point, its spatial information is determined by the location (i.e., x, y, z coordinates) and connection (i.e., the indexing of its parent point). Here, we denote the 3D neuronal data as $\mathbf{x}_i \in \mathbb{R}^{n_i \times 4}$, which include n_i points. Each point includes the above spatial information with 4 dimensions. As shown in Fig. 2, given a neuron data $\mathbf{x}_i$, we first plot the location of its 3D points. Considering the initial neurons may not oriented properly, we employ the principal component analysis (PCA) algorithm to shift and rotate neurons to a normalized axis. For the point set of each neuron (i.e., a $n_i \times 3$ matrix), the PCA algorithm first compute their three-dimensional mean value (i.e., a 1×3 vector) and coefficients (i.e., a 3×3 matrix). Then each point can be shifted and rotated by the mean value and coefficients respectively. This transformation can make sure that similar 3D neurons can be transformed into similar 2D images after the following neuronal projection, regardless of their initial orientation.

Subsequently, all 3D points are orthogonally projected into three angles of view, i.e., the x-y, x-z, and y-z plane, respectively. These projected 2D point sets haven't reflected the connection of branches and bifurcations in neuron morphology. Thus, we link each point with their parent point and then map these points and lines into binary images. The generated binary images can replace the 3D neuron data for deep feature representation. In general, transforming 3D point sets into binary images can significantly improve the computational efficiency when training deep neural networks. More importantly, our method can preserve the structure of neuron morphologies by the three angles' orthogonal projection and child-parent points linking.

Feature Learning using SCAEs After receiving binary images from the original 3D neuron data, we set them as input for the deep feature representation. In current neuron morphological databases, due to the fact that there are no sufficient annotations to identify and classify each neuron, only unsupervised deep neural networks can be utilized for feature learning. In addition, not only the image pixels, the structure of neurons also need to be considered in the network. Here, we employ the stacked convolutional autoencoders (SCAEs) (Masci et al. 2011), which can explore the intrinsic structure of neurons in an unsupervised manner.

Figure 3 present the architecture of our SCAEs, including convolutional encoder and decoder two parts. In the encoder part, the input neuron images are sequentially processed through convolution (in blue color) and max-pooling (in red color). The size of input images were 128×128, with the filter size of 3×3 and the max-pooling size of 2×2. In the decoder part, the learned features are also sequentially processed with two operations, i.e., upsampling (in green color) and convolution, using the same size of filter and pooling with the encoder part. Specifically, in our SCAEs, we employ activation and batch normalization (Ioffe and Szegedy 2015) right after convolution. The activation function is ReLU (Nair and Hinton 2010) except the last decoder layer, where we utilize the sigmoid function to reconstruct binary images. The number of feature maps is ranging from 4 to 64 in the encoder part, and then ranging from 64 to 1 in the decoder part. The upsampling operation follows the method used in Zeiler et al. (2011), i.e., restoring the location of the previously stored maximum value, and assigning the other locations as zero. In the training stage, the SCAEs' output images try to reconstruct the original input images using the loss function of binary cross-entropy.

After training the SCAEs, we can remove the decoder part. Given a neuron data, its three binary images are sequentially set as the input in the trained SCAEs. Then

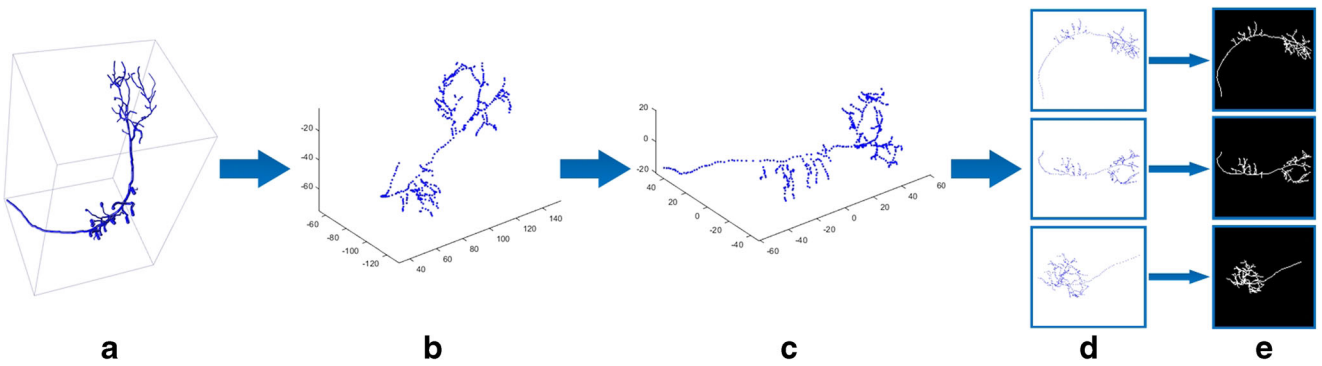

Fig. 2 The pipeline of transforming 3D neurons into images: **a** the original neuron data visualized by Vaa3D (Peng et al. 2010); **b** the 3D points of neuron data in initial orientation; **c** the 3D points after principal component analysis (PCA); **d** the projection of each point in three angles of view; **e** the generated binary images by linking each point with their parent points

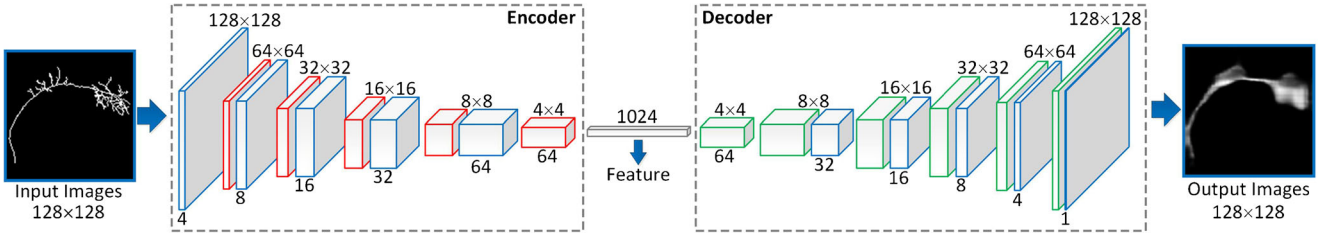

Fig. 3 The architecture of our stacked convolutional auto-encoders, including convolutional encoder and decoder two parts

their last encoder layer's output can be combined together as the deep feature of that input neuron.

Feature Fusion Considering the information loss from 3D neuron data to binary images, the deep features also cannot fully represent the original neuron morphology. Therefore, we propose to fuse the deep features with the hand-crafted features to pursue more accurate retrieval results. For the deep features, as they are usually noisy and redundant with thousands of dimensions, we first employ the PCA algorithm to reduce the dimension and preserve the main components in deep features. For the hand-crafted features, we compute the quantitative measurements in each 3D neuron based on the L-Measure toolbox (Scorcioni et al. 2008), including global, branch, and bifurcation three levels of measurements (Li et al. 2017b).

With regards to fusing two features, there are mainly two levels of fusion in the field of image retrieval, i.e., feature-level and decision-level. For the feature-level fusion, the goal is to combine two or more feature vectors into a single one with more discriminative power than any of the input feature vectors. This fusion can be implemented before similarity searching. For the decision-level fusion, the goal is to weight the retrieval results from different features and fuse these results via techniques such as majority voting (Jain et al. 2005). Despite the fact that the decision-level fusion has demonstrated excellent performance in many image retrieval tasks (Zhang et al. 2015a, 2016), it may not be suitable in our case that the method is developed towards large neuron databases. Particularly, the decision-level fusion in our case needs to learn binary coding functions and search the whole database with different kind of features respectively, which are inefficient for large-scale retrieval. Therefore, we directly combine the deep features with the hand-crafted features together before binary coding. In practice, we find that such feature-level fusion can achieve excellent performance in large-scale neuron retrieval.

Binary Coding Based Large-scale Retrieval

The above feature learning and fusion stages can help to improve the accuracy of neuron retrieval. To further improve the retrieval efficiency of large-scale datasets, we employ binary coding methods which can learn coding functions to compress the fused features into short binary codes and also preserve similarities among original features. However, two problems need to be considered when applying binary coding in large-scale neuron data: 1) current neuron databases haven't provide sufficient annotations for each neuron that can support the supervised training of coding functions; 2) some neuron morphologies (also their features) are hard to differentiate under extremely subtle and linearly inseparable differences. Here, we adopt a recently proposed method, i.e., the MIPS based binary coding (Shen et al. 2015, 2017), which can iteratively learn two asymmetric and non-linear coding functions from training samples without any supervised information.

Given two training sets which are randomly selected from the neuron database, the fused features of these two sets can be denoted as $\mathbf{A} \in \mathbb{R}^{n \times d}$ and $\mathbf{X} \in \mathbb{R}^{m \times d}$, including n and m neurons with d dimensional features respectively. Assuming $h(\cdot)$ and $z(\cdot)$ are the coding functions for $\mathbf{A}$ and $\mathbf{X}$, the MIPS based binary coding aims to optimize the above objective function:

$$\min_{h,z} \left\| h(\mathbf{A}) z(\mathbf{X})^{\mathrm{T}} - \hat{\mathbf{S}} \right\|^2 \tag{1}$$

where $\hat{\mathbf{S}}$ is the binarization of similarity matrix between $\mathbf{A}$ and $\mathbf{X}$, i.e., $\hat{\mathbf{S}} = \mathrm{sgn}(\mathbf{A}\mathbf{X}^{\mathrm{T}})$. In Shen et al. (2015), the authors first transform the above objective as a maximum inner product search (MIPS) problem. Then they provide an asymmetric optimization algorithm to alternately learn two coding functions in several iterations. The asymmetric and inner product strategies can map coding functions into high dimensional and non-linear space in an unsupervised manner, which are particularly suitable for neuron morphological retrieval. Subsequently, the learned two coding functions $h(\cdot)$ and $z(\cdot)$ can be applied to the fused features of the database neurons and the query neuron in respective. Storing their binary codes is generally more space-saving than that of long feature vectors. More importantly, we will demonstrate in the experiment that the binary codes are capable of real-time similarity searching in large-scale databases without losing too much accuracy.

Experiment

In this section, we first evaluate the effectiveness of our proposed neuron retrieval framework. Then we discuss its advantages and limitations.

Evaluation of Neuron Retrieval

Experimental Setting During the experiment, we carry out the evaluation on the NeuroMorpho database (http://neuromorpho.org), including 40 species and in total 62,304 reconstructed neurons from 278 labs. For the hand-crafted features, we employ the L-measure toolbox (Scorcioni et al. 2008) to extract 38 quantitative measurements, following the setting with previous articles (Conjeti et al. 2016a, b; Mesbah et al. 2015; Li et al. 2016, 2017b, c). In the process of transforming 3D neuron into 2D images, we project each neuron into three images with the size of 128×128. For our SCAEs, we set a weight decay of 10^{-4} and momentum of 0.9. The whole neural networks are trained end-to-end with 100 epochs and an initial learning rate of 0.01. We finally learn 3072 dimensional deep features for each neuron data, with 1024 dimensions in each projected image. In the MIPS based binary coding, according to Shen et al. (2015, 2017), we set the parameter λ to 100 and the maximum iteration number $t = 5$. In the model training phase (i.e., SCAEs training and binary coding training), we randomly select 30,000 neurons as training data, excluding the test neurons we later used. All experiments are conducted on a desktop with a 1.6GHz processor of twelve cores and 128G RAM.

Validation of Feature Representation We first evaluate the retrieval precision of our proposed method in the whole NeuroMorpho database (http://neuromorpho.org), which include 58,414 valid neurons (we exclude neurons that cannot be read and measured by L-measure toolbox (Scorcioni et al. 2008)). As the current database hasn't provide definite classes of each neuron, we evaluate the retrieval precision based on their properties, e.g., brain regions, cell types, development stages, etc, which have strong relevance with neuron morphologies. Particularly, we first select the Drosophila's projection neurons as queries. The characteristics of these neurons are summarized in Table 1, which can be easily located and identified in the NeuroMorpho database (http://neuromorpho.org) (233 such projection neurons in total). We denote these query neuron as uPNs. This selection of query neurons also consistent with the previous articles (Wan et al. 2015; Costa et al. 2016; Li et al. 2017b, c), since uPNs are the one kind of most well-classified neurons in the database. We provide the comparison of the retrieval precision with three feature representation methods, i.e., our deep feature representation, the hand-crafted feature measurements used in previous articles (Conjeti et al. 2016a, b; Mesbah et al. 2015; Li et al. 2016, 2017b, c), and our feature fusion, which are abbreviated as Deep-fea, Hand-fea, and Fused-fea respectively.

Table 2 records the average retrieval precision of the three methods under different numbers of retrieved neurons. For each uPNs, it matches similar neurons with the entire dataset (58,414 neurons in total), i.e., computing the Euclidean distance one by one through feature vectors and ranking them in ascending order. Then its retrieval precision is computed by the percentage of uPNs (except itself) appeared in all top similar neurons, e.g., top-10 retrieved neurons. In Table 2, the average precision is recorded by evaluating all 233 uPNs. For the Fused-fea, we first employ PCA to compress the deep features into 40 dimensions, then fusing them with the hand-crafted features. In Table 2, the Fused-fea method can achieve the highest retrieval precision compared with other two methods. The Deep-fea and Hand-fea methods also achieve reasonable retrieval precision. According to these results, we validate that both our SCAEs based deep features and the traditional hand-crafted features have the ability to represent 3D neuron morphologies. More importantly, these two kinds of features are complementary with each other, since their fusion results have a significant improvement in precision. Based on the reconstructed images from SCAEs, we find that our learned deep features are likely to explore and represent holistic structures in neuron morphologies. As a contrast, the hand-crafted method is not good at exploring neuron's holistic structure, where it can only compute some global measurements (e.g., neuron's total height, length, etc) as features that are indiscriminative for large-scale neuron databases.

In addition to the above quantitative evaluation, we random select four query neurons and present their top-5 retrieved similar neurons using the Vaa3D software (Peng et al. 2010). As shown in Fig. 4, the morphologies of most retrieved neurons are quite similar to their query neurons, which validate that our proposed method can

Table 1 Summarizing the characteristics of our query neurons

Brain Regions	Olfactory antennal lobe; Glomerulus
Cell Classes	Principal cell; Uniglomerular projection
Development	Adult

Table 2 Comparing the average retrieval precision of three methods under different numbers of retrieved neurons

	top10	top20	top30	top50
Deep-fea	0.7113	0.6127	0.5538	0.4654
Hand-fea	0.8586	0.7776	0.7239	0.6407
Fused-fea	**0.9083**	**0.8353**	**0.7948**	**0.7354**

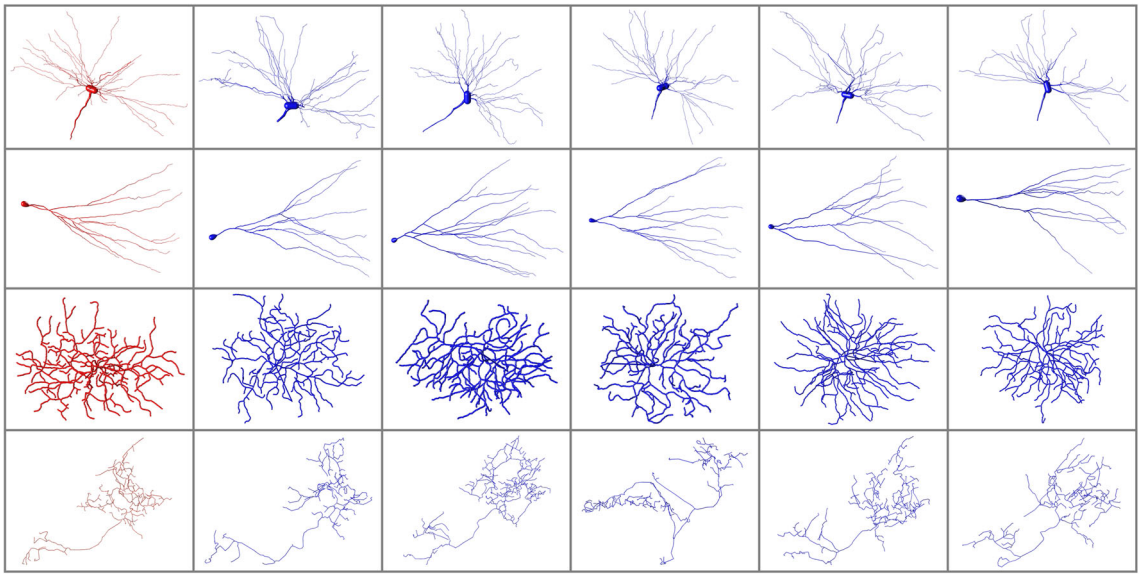

Fig. 4 Four randomly selected query neurons (in red) and their top-5 retrieved similar neurons (in blue) through our proposed method

effectively retrieve similar neurons in large-scale databases. Moreover, we find that the retrieved neurons also share common properties with the query neuron. For example, the second query neuron is selected from a rat's brain, where the neuron is located in the brain regions of the hippocampus, dentate gyrus, and granule layer with the cell classes of principal cell and granule. By checking properties in the NeuroMorpgo database (http://neuromorpho.org), the retrieved top-5 similar neurons also reflect same species, brain regions and cell classes with the query neuron.

Validation of Binary Coding In this part, we first validate the effectiveness of binary coding, which can dramatically improve the retrieval efficiency without losing too much precision. Table 3 compares the retrieval precision and efficiency before and after the MIPS based binary coding, using the feature fusion results. For the 78 dimensional fused features, the learned coding functions compress them into 32 bits of binary codes. In this experiment, we also employ the 233 uPNs as queries and record their top-10 average precision. According to Table 3, the retrieval precision after binary coding hasn't lose too much (i.e., only 2% lower compared with the feature fusion results). However, after compressing the fused features into binary codes, the 233 uPNs can achieve real-time retrieval in the whole NeuroMorpho database, i.e., sequentially indexing 58414 neurons with only 0.15 seconds in total. Compared with the exhaustive searching using the fused features, the efficiency of neuron retrieval has a significant improvement through binary coding (i.e., around 500 times faster). This superiority will be especially benefited in the future neuron retrieval tasks since an increasing number of neurons are reconstructed and added to large-scale databases (https://github.com/BigNeuron).

Table 3 Comparison of neuron retrieval precision and efficiency (in second) before and after MIPS based binary coding, using the feature fusion results

	Precision	Time(s)
Fused-fea	**0.9083**	73.76
MIPS	0.8876	**0.15**

In addition, we evaluate that the MIPS based binary coding is suitable for the retrieval of neuron morphologies. We compare it with three popular binary coding/hashing methods that have been widely used in recent years, i.e., ITQ (Gong et al. 2013), AGH (Liu et al. 2011), and SH (Weiss et al. 2009). Figure 5a presents the average retrieval precision from top-1 to top-100 samples of uPNs, using the above four binary coding methods. All methods compress the fused features into 32 bits of binary codes. According to Fig. 5a, the MIPS based binary coding achieves the best performance among the four methods. This is mainly because of its learned non-linear coding functions that can effectively discriminate the linearly inseparable neuron data.

Discussions

Parameter Settings In our framework, the SCAEs model is trained based on the projected 2D images with the size of 128 × 128. This setting is a trade-off between retrieval precision and computational efficiency. Figure 5b records the average retrieval precision from top-1 to

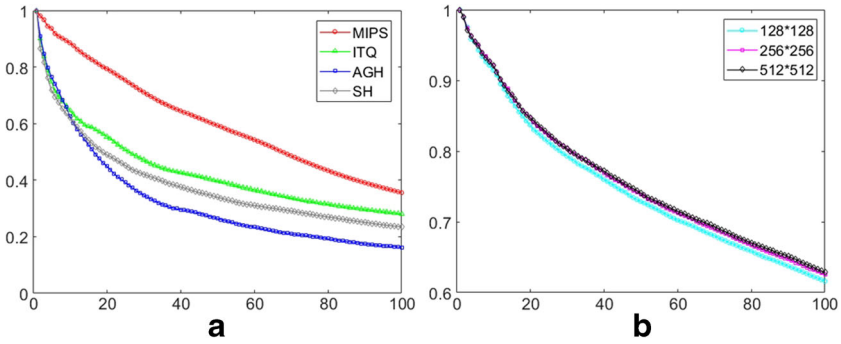

Fig. 5 Average retrieval precision from top-1 to top-100 samples of uPNs: **a** comparison of four binary coding methods; **b** comparison of three sized images for SCAEs model training

top-100 samples of uPNs, with the projected image size of 128 × 128, 256 × 256, 512 × 512 respectively. All three sized images are transformed into 1024 dimensional feature vectors in SCAEs models. According to Fig. 5b, the retrieval performance haven't changed too much, i.e., no more than 2% difference. This is mainly because of the properties of the SCAEs, which can only explore and represent holistic structures in neuron morphologies (discussed in "Evaluation of Neuron Retrieval"). While the holistic structures in neurons do not have remarkably difference under different image resolutions, e.g., from 128 × 128 to 512 × 512. On the other side, their computational efficiency has a tremendous difference. For example, in our experiment, it only costs 3 hours to train the SCAEs model using 90,000 images with the size of 128 × 128. While the training time is 2 days using the 512 × 512 sized images. Therefore, we project 2D images with the size of 128 × 128 which can achieve state-of-the-art precision under sustainable time complexity.

Advantages The proposed framework achieves large-scale neuron morphological retrieval with superior accuracy and efficiency, demonstrating excellent performance in assisting neuron exploration and analysis. These results mainly benefited from our proposed neuron feature representation and the employed MIPS based binary coding. When neuron databases are large, the traditional hand-crafted features cannot fully represent and differentiate each neuron. Thus we designed a novel feature representation method using deep neural networks, which can transform 3D neuron data into 2D images and automatically learn features end-to-end. Our learned deep features have different aspects of representation compared with the hand-crafted features (e.g., holistic structures compared with branch/bifurcation topologies). Therefore, their combination will accordingly more representative for the neuron morphology. For the binary coding part, we directly adopt the MIPS method which has been used for the large-scale neuron retrieval (Li et al. 2017b).

Limitations There are also some limitations in our neuron retrieval framework. The first limitation is the information loss when transforming 3D neuron data into 2D images. This limitation mainly reflects in two situations: 1) the 2D images cannot preserve the fine-grained spatial structures in neurons, especially for the neuron with complicated morphologies, e.g., the right dendrites in Fig. 6a; 2) some neurons provided by the NeuroMorpho database (http://neuromorpho.org) are only two dimensions, i.e., the third dimension is a fixed constant. As shown in Fig. 6b, in such case, our method can only get straight lines in the second and third projected image. These two situations may influence the retrieval results for some specific neurons. The second limitation is the separation of feature representation and binary coding. In our framework, we process these two stages in sequence, which indicates two times of information loss, i.e., from 3D neuron data to features to binary codes. This may influence the retrieval precision of the whole framework.

Augmented Reality for Neuron Visualization

The above experiments validate that our framework can effectively retrieve similar neurons in large-scale databases. Based on the retrieval results, we develop a novel neuron visualization program in assisting neuron exploration through the Microsoft HoloLens augmented reality (AR) headset. Here, we introduce the implementation details and discuss its applications in neuron exploration and analysis.

Implementation Our visualization program is developed using the Vuforia AR platform (https://www.vuforia.com/) and the Unity game engine, which can create a powerful tool for the application development on Microsoft's Windows Holographic Platform. Particularly, when visualizing the neurons, we start by mapping the data points for each neuron to the real world. Based on the spatial anchors (i.e., data structures containing a world coordinate and information about the surrounding environment), anchor points are first created in the virtual environment that the HoloLens has generated in the correspondence with the real world. Each anchor point has their own coordinate systems in which the data points are rendered. To compete for the

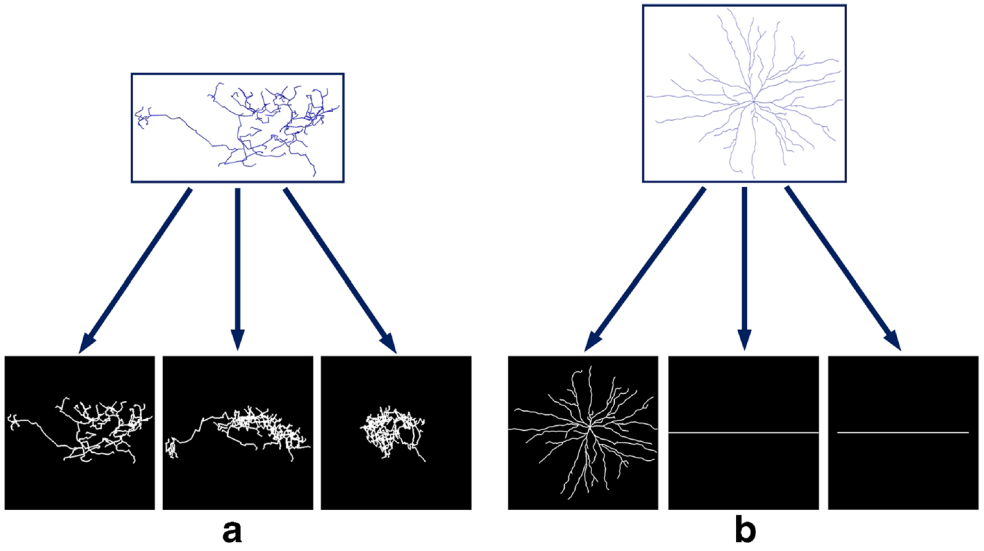

Fig. 6 Limitations in our neuron projection strategy, **a** the 2D image cannot preserve the fine-grained spatial structures in neurons; **b** some neurons in the NeuroMorpho (http://neuromorpho.org) are only two dimensions

visualization, the Unity's built-in line renderer is employed to create lines connecting each data point to their parent data point, according to the *SWC* data format provided in neuron databases. The package uses the Bézier curve in connecting these lines, which is a type of parametric curve that can be scaled indefinitely. Once the neurons are mapped in the virtual environment, the Hololens uses stereo images to visualize the neurons on the transparent screens of the headset, so that they appear as if they existed in the real world in front of the user. As the anchor points correspond to the real world, the visualizations will appear to be fixed to a given location before the user unless moved. In addition, the neuron visualizations are rendered with depth-based coloring, which adjusts based on the user's proximity to the neuron.

We also made the visualized neurons interactable. We place a transparent transform at the center of each neuron, which also acts as the parent for all the data points of an individual neuron. Upon placing the visually directed cursor of the HoloLens on the center of the neuron, the cursor gives a visual cue, which indicates that an object is interactable. As the transform is transparent, this gives the impression that the neuron itself is the object you are interacting with.

Application Fig. 7 presents some screen shots of our visualization results. The developed AR program can interactively and immersively visualize neurons in different scales and different view angles. As shown in Fig. 7, the program can first present some retrieved neurons to users using our neuron retrieval method. Then users can utilize the HoloLens built-in "finger tap" gesture to select an interested neuron for further analysis. Once selected, various voice commands, created by us but powered by Microsoft "Cortona" voice assistant technology, can be used to identify the type of interaction to be performed on neurons. We include functionality allowing users to move, scale, and rotate the selected neurons. Users can also use the built-in "pinch" gesture along with hand motions to manipulate neurons. This provides a simple and intuitive

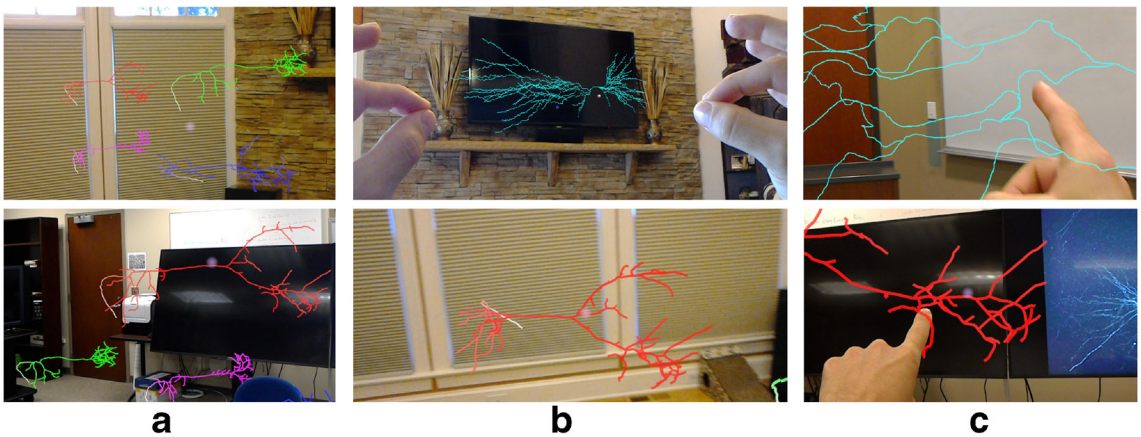

Fig. 7 The screen shots of neuron visualization and analysis using the Microsoft HoloLens AR headset: **a** visualize some of the retrieved neurons; **b** manipulate and enlarge interested neurons ; **c** continue to enlarge neurons, interactively analyze their fine-grained details

way for users to interact with the visualizations, and to do so from any position in the environment. The screen shots in Fig. 7 can only provide a glimpse of how our program works, where it lost the biggest benefits that are gained freeing users from a screen. Compared with other neuron visualization techniques (e.g., Vaa3D software Peng et al. 2010), the main advantage of our AR based program is that it can visualize neurons in an interactive and immersive way, and thus help users taking a deep and comprehensive analysis of neurons.

Conclusions

In this paper, we present an accurate and efficient neuron retrieval framework, which can help users exploring large-scale neuron morphological databases. Particularly, we propose a novel feature representation method for the 3D neuron data, which are the first time that introduces deep learning techniques in the 3D neuromorphological analysis. Then we employ binary coding method to compress neuron features into short binary codes, which are especially suitable for the retrieval in large-scale neuron databases. Experimental results validate the efficacy of our proposed framework. Additionally, we develop a new neuron visualization program based on the AR techniques, where neurons can be explored and analyzed in an interactive and immersive way. Our future work will focus on the performance improvement of deep feature representation and visualization for neuron morphologies, and then develop a more comprehensive tool for neuron exploration and analysis.

Information Sharing Statement

The dataset used in this paper are from the NeuroMorpho.Org (RRID:SCR_002145), which are available at http://neuromorpho.org/. The L-Measure toolbox (RRID:SCR_003487) is available at http://cng.gmu.edu:8080/Lm/. Both the source code and documentation are available on request. Contact: rutgers.shaoting@gmail.com and zhongyu.emerald@gmail.com.

Acknowledgements This work is partially supported by the National Science Foundation under grant ABI-1661280, ABI-1661289, and CNS-1629913.

References

Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H. (2007). Greedy layer-wise training of deep networks. *NIPS*, *19*, 153.

Cannon, R.C., Turner, D.A., Pyapali, G.K., Wheal, H.V. (1998). An on-line archive of reconstructed hippocampal neurons. *Journal of Neuroscience Methods*, *84*(1), 49–54.

Conjeti, S., Katouzian, A., Kazi, A., Mesbah, S., Beymer, D., Syeda-Mahmood, T.F., Navab, N. (2016a). Metric hashing forests. *Medical image analysis*, *34*, 13–29.

Conjeti, S., Mesbah, S., Negahdar, M., Rautenberg, P.L., Zhang, S., Navab, N., Katouzian, A. (2016b). Neuron-miner: an advanced tool for morphological search and retrieval in neuroscientific image databases. *Neuroinformatics*, *14*(4), 369–385.

Costa, L.D.F., Zawadzki, K., Miazaki, M., Viana, M.P., Taraskin, S. (2010). Unveiling the neuromorphological space. *Frontiers in Computational Neuroscience*, *4*, 150–163.

Costa, M., Manton, J.D., Ostrovsky, A.D., Prohaska, S., Jefferis, G.S. (2016). NBLAST: Rapid, sensitive comparison of neuronal structure and construction of neuron family databases. *Neuron*, *91*(2), 293–311.

Gong, Y., Lazebnik, S., Gordo, A., Perronnin, F. (2013). Iterative quantization: a procrustean approach to learning binary codes for large-scale image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(12), 2916–2929.

He, K., Zhang, X., Ren, S., Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 770–778).

Hinton, G.E., & Salakhutdinov, R.R. (2006). Reducing the dimensionality of data with neural networks. *Science*, *313*(5786), 504–507.

Ioffe, S., & Szegedy, C. (2015). Batch normalization: accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)* (pp. 448–456).

Jain, A., Nandakumar, K., Ross, A. (2005). Score normalization in multimodal biometric systems. *Pattern recognition*, *38*(12), 2270–2285.

Ji, S. (2013). Computational genetic neuroanatomy of the developing mouse brain: dimensionality reduction, visualization, and clustering. *BMC bioinformatics*, *14*(1), 222.

Krizhevsky, A., Sutskever, I., Hinton, G.E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)* (pp. 1097–1105).

LeCun, Y., Bengio, Y., Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444.

LeCun, Y., Bottou, L., Bengio, Y., Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324.

Li, R., Zeng, T., Peng, H., Ji, S. (2017a). Deep learning segmentation of optical microscopy images improves 3-D neuron reconstruction. *IEEE Transactions on Medical Imaging*, *36*(7), 1533–1541.

Li, Z., Fang, R., Shen, F., Katouzian, A., Zhang, S. (2017b). Indexing and mining large-scale neuron databases using maximum inner product search. *Pattern Recognition*, *63*, 680–688.

Li, Z., Metaxas, D.N., Lu, A., Zhang, S. (2017c). Interactive exploration for continuously expanding neuron databases. *Methods*, *115*, 100–109.

Li, Z., Shen, F., Fang, R., Conjeti, S., Katouzian, A., Zhang, S. (2016). Maximum inner product search for morphological retrieval of large-scale neuron data. In *International Symposium on Biomedical Imaging (ISBI)* (pp. 602–606).

Li, Z., Zhang, X., Mller, H., Zhang, S. (2018). Large-scale retrieval for medical image analytics: A comprehensive review. *Medical Image Analysis*, *43*, 66–84.

Liu, J., Zhang, S., Liu, W., Deng, C., Zheng, Y., Metaxas, D.N. (2017). Scalable mammogram retrieval using composite anchor graph hashing with iterative quantization. *IEEE Transactions on Circuits and Systems for Video Technology*, *27*(11), 2450–2460.

Liu, W., Wang, J., Ji, R., Jiang, Y.G., Chang, S.F. (2012). Super vised hashing with kernels. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 2074–2081).

Liu, W., Wang, J., Kumar, S., Chang, S.F. (2011). Hashing with graphs. In *International Conference on Machine Learning (ICML)* (pp. 1–8).

Masci, J., Meier, U., Ciresan, D., Schmidhuber, J. (2011). Stacked convolutional auto-encoders for hierarchical feature extraction. ICANN, 52–59.

Mesbah, S., Conjeti, S., Kumaraswamy, A., Rautenberg, P., Navab, N., Katouzian, A. (2015). Hashing forests for morphological search and retrieval in neuroscientific image databases. In *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)* (pp. 52–59).

Mukherjee, S., Basu, S., Condron, B., Acton, S.T. (2013). Tree2Tree2: neuron tracing in 3D. In *International Symposium on Biomedical Imaging (ISBI)* (pp. 448–451).

Nair, V., & Hinton, G.E. (2010). Rectified linear units improve restricted boltzmann machines. In *International Conference on Machine Learning (ICML)* (pp. 807–814).

Peng, H., Ruan, Z., Long, F., Simpson, J.H., Myers, E.W. (2010). V3D enables real-time 3D visualization and quantitative analysis of large-scale biological image data sets. *Nature biotechnology*, *28*(4), 348–353.

Salakhutdinov, R. (2015). Learning deep generative models. *Annual Review of Statistics and Its Application*, *2*, 361–385.

Scorcioni, R., Polavaram, S., Ascoli, G.A. (2008). L-Measure: a web-accessible tool for the analysis, comparison and search of digital reconstructions of neuronal morphologies. *Nature protocols*, *3*(5), 866–876.

Shen, F., Liu, W., Zhang, S., Yang, Y., Shen, H.T. (2015). Learning binary codes for maximum inner product search. In *IEEE International Conference on Computer Vision (ICCV)* (pp. 4148–4156).

Shen, F., Yang, Y., Liu, L., Liu, W., Tao, D., Shen, H.T. (2017). Asymmetric binary coding for image search. *IEEE Transactions on Multimedia*, *19*(9), 2022–2032.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition, arXiv:1409.1556.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 1–9).

Wan, Y., Long, F., Qu, L., Xiao, H., Hawrylycz, M., Myers, E.W., Peng, H. (2015). BlastNeuron for automated comparison, retrieval and clustering of 3D neuron morphologies. *Neuroinformatics*, *13*(4), 487–499.

Wang, J., Liu, W., Kumar, S., Chang, S.F. (2016). Learning to hash for indexing big dataa survey. *Proceedings of the IEEE*, *104*(1), 34–57.

Weiss, Y., Torralba, A., Fergus, R. (2009). Spectral hashing. In *Advances in Neural Information Processing Systems (NIPS)* (pp. 1753–1760).

Wu, G., Jia, H., Wang, Q., Shen, D. (2011). SharpMean: group-wise registration guided by sharp mean image and tree-based registration. *NeuroImage*, *56*(4), 1968–1981.

Yan, C., Zhang, Y., Dai, F., Wang, X., Li, L., Dai, Q. (2014a). Parallel deblocking filter for HEVC on many-core processor. *Electronics Letters*, *50*(5), 367–368.

Yan, C., Zhang, Y., Xu, J., Dai, F., Li, L., Dai, Q., Wu, F. (2014b). A highly parallel framework for HEVC coding unit partitioning tree decision on many-core processors. *IEEE Signal Processing Letters*, *21*(5), 573–576.

Yan, C., Zhang, Y., Xu, J., Dai, F., Zhang, J., Dai, Q., Wu, F. (2014c). Efficient parallel framework for HEVC motion estimation on many-core processors. *IEEE Transactions on Circuits and Systems for Video Technology*, *24*(12), 2077–2089.

Yu, G., & Yuan, J. (2014). Scalable forest hashing for fast similarity search. In *IEEE International Conference on Multimedia and Expo (ICME)* (pp. 1–6).

Zeiler, M.D., Taylor, G.W., Fergus, R. (2011). Adaptive deconvolutional networks for mid and high level feature learning. In *IEEE International Conference on Computer Vision (ICCV)* (pp. 2018–2025).

Zhang, S., Yang, M., Cour, T., Yu, K., Metaxas, D.N. (2015a). Query specific rank fusion for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *37*(4), 803–815.

Zhang, X., Dou, H., Ju, T., Xu, J., Zhang, S. (2016). Fusing heterogeneous features from stacked sparse autoencoder for histopathological image analysis. *IEEE journal of biomedical and health informatics*, *20*(5), 1377–1383.

Zhang, X., Liu, W., Dundar, M., Badve, S., Zhang, S. (2015b). Towards large-scale histopathological image analysis: Hashing-based image retrieval. *IEEE Transactions on Medical Imaging*, *34*(2), 496–506.

Zhang, X., Xing, F., Su, H., Yang, L., Zhang, S. (2015c). High-throughput histopathological image analysis via robust cell segmentation and hashing. *Medical image analysis*, *26*(1), 306–315.

Zhou, Z., Liu, X., Long, B., Peng, H. (2016). TReMAP: automatic 3D neuron reconstruction based on tracing, reverse mapping and assembling of 2D projections. *Neuroinformatics*, *14*(1), 41–50.

Zhou, Z., Sorensen, S., Zeng, H., Hawrylycz, M., Peng, H. (2015). Adaptive image enhancement for tracing 3D morphologies of neurons and brain vasculatures. *Neuroinformatics*, *13*(2), 153–166.