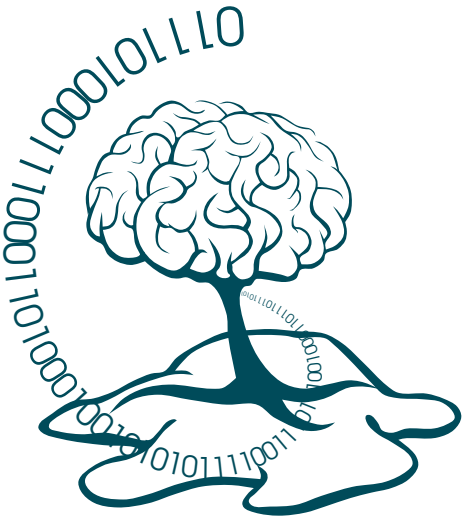

Ischemic Stroke Lesion Segmentation

www.isles-challenge.org

Proceedings

5th October 2015

Munich, Germany

UNIVERSITÄT ZU LÜBECK

Technische Universität München

UNIVERSITÄT BERN

MICCAI'15 MUNICH

Preface

Stroke is the second most frequent cause of death and a major cause of disability in industrial countries. In patients who survive, stroke is generally associated with high socioeconomic costs due to persistent disability. Its most frequent manifestation is the ischemic stroke, whose diagnosis often involves the acquisition of brain magnetic resonance (MR) scans to assess the stroke lesion's presence, location, extent, evolution and other factors. An automated method to locate, segment and quantify the lesion area would support clinicians and researchers alike, rendering their findings more robust and reproducible.

New methods for stroke segmentation are regularly proposed. But, more often than desirable, it is difficult to compare their fitness, as the reported results are obtained on private datasets. Challenges aim to overcome these shortcomings by providing (1) a public dataset that reflects the diversity of the problem and (2) a platform for a fair and direct comparison of methods with suitable evaluation measures. Thus, the scientific progress is promoted.

With ISLES, we provide such a challenge covering ischemic stroke lesion segmentation in multi-spectral MRI data. The task is backed by a well established clinical and research motivation and a large number of already existing methods. Each team may participate in either one or both of two sub-tasks:

SISS Automatic segmentation of ischemic stroke lesion volumes from multi-spectral MRI sequences acquired in the sub-acute stroke development stage.

SPES Automatic segmentation of acute ischemic stroke lesion volumes from multi-spectral MRI sequences for stroke outcome prediction.

The participants downloaded a set of training cases with associated expert segmentations of the stroke lesions to train and evaluate their approach, then submitted a short paper describing their method. After reviewing by the organizers, a total of 17 articles were accepted and compiled into this volume. At the day of the challenge, each teams' results as obtained on an independent test set of cases will be revealed and a ranking of methods established.

For the final ranking and more information, visit WWW.ISLES-CHALLENGE.ORG.

Oskar Maier, Universität zu Lübeck

Mauricio Reyes, University of Bern

Björn Menze, TU Munich

August 2015

Version: 2015-09-30

Organizers

Oskar Maier, Universität zu Lübeck, Germany

Mauricio Reyes, University of Bern, Switzerland

Björn Menze, TU Munich, Germany

Sponsoring Institutions

Institute of Medical Informatics, Universität zu Lübeck, Germany

Institute for Surgical Technology & Biomechanics, University of Bern, Switzerland

Computer Science, TU Munich, Germany

Version: 2015-09-30

Distributed Deep Learning Framework for Large-Scale 3D Medical Image Segmentation

T. Klein[1,2], N.K. Batmanghelich[2], and William M. Wells III[1,2]

[1] Brigham and Women's Hospital, Harvard Medical School, Boston, USA
[2] Massachusetts Institute of Technology, Cambridge, USA

Abstract. In this paper, we propose a novel distributed framework that makes the excellent performance of the Deep Learning (DL) available for large-scale medical applications. More specifically, we propose a distributed optimization approach based on Alternating Direction Method of Multipliers (ADMM) enabling us to optimize the cost function of the Convolutional Neural Network (CNN) across multiple machines. This framework makes the computational power and memory of multiple machines available to our algorithm, which facilitates computationally and memory intensive tasks such as medical image segmentation. We apply our method on the Ischemic Stroke Lesion Segmentation (SPES) task.

Keywords: Distributed Computing, Deep Learning, Convolutional Neural Network

1 Method

The general idea of the paper is as follows, we divide the image into set of volumetric patches denoted by x_i. For each patch, we train a DL discriminative model to predict the labels of central voxels, denoted by y_i. We use Convolutional Neural Networks (CNN) [3] as the discriminative model. CNN consists of stacked up layers, combining a convolution followed by a downsampling step referred to as pooling. As the convolution is applied by shifting a filter element over the input domain, translational invariance is achieved. This also restricts the complexity of the model by reducing the number of free parameters. Finally, we collect the class labels of all central patches and use a voting scheme to get the final segmentation results.

1.1 Softmax Regression

In the following, we first revisit the preliminaries of multi-class classification, *i.e.* softmax regression, in the context of the deep learning. The input to the prediction model is the patch data of the D image channels. The output is the class labels (segmentation labels) of L voxels in the center of the patch. Therefore, it is multi-task classification problem. We denote the input patch data with $x \in \mathbb{R}^{W \times W \times W \times D}$, where W is the width of the patch and D is number

of image channels and the corresponding labels with $y \in [1,..,K]^L$ where $[1..K]$ is set of all integers between 1 to K, each denoting one tissue type. We use y_i^ℓ to denote element ℓ of the patch i. For notational brevity and future reference, we use $\mathcal{X} = \{x_i\}_{i=1}^N$, $\mathcal{Y} = \{y_i\}_{i=1}^N$. We use $(\mathcal{X}_m, \mathcal{Y}_m)$ to represent a batch of training data.

Given input patch image x, the proposed method estimates the probability of the label map of the central voxels, $P_\theta(y_i^\ell|x_i)$ where $\theta \in \mathbb{R}^n$ are the parameters of our model. We use a CNN to model $P_\theta(y_i^\ell|x_i)$. Given y_i^ℓ takes discrete values, with abuse of notation, we can represent $P_\theta(y_i^\ell|x_i)$ as a vector:

$$P_\theta(y_i^\ell|x_i) = \begin{bmatrix} P_\theta(y_i^\ell = 1|x_i) \\ P_\theta(y_i^\ell = 2|x_i) \\ \vdots \\ P_\theta(y_i^\ell = K|x_i) \end{bmatrix} = \frac{1}{\sum_{j=1}^K \exp\left[f_k(x_i;\theta)\right]} \begin{bmatrix} \exp\left[f_1(x_i;\theta)\right] \\ \exp\left[f_2(x_i;\theta)\right] \\ \vdots \\ \exp\left[f_K(x_i;\theta)\right] \end{bmatrix} \tag{1}$$

For $P_\theta(y_i^\ell|x_i)$, we propose the use of a 3D deep convolutional neural network [3]. Given the CNN, the parameters θ consist of filter maps as well as weights (and biases) for fully connected layers, representing the CNN. Finally, we learn the optimal θ by solving an optimization problem, where the objective function is the cross entropy between $P_\theta(y_i|x_i)$ and given class label y_i, defined as

$$J(\theta; \mathcal{X}, \mathcal{Y}) = -\left[\sum_{i=1}^N \sum_{k=1}^K \mathbf{1}\left\{y_i = e_k\right\} \log \frac{\exp\left[f_k(\theta, x_i)\right]}{\sum_{j=1}^K \exp\left[f_j(\theta, x_i)\right]}\right]. \tag{2}$$

Here $\mathbf{1}\{\cdot\}$ denotes the indicator function and e_k is the base vector where the k'th element is 1.

For solving θ, one has to resort to an iterative optimization algorithm such as gradient descent. Specifically, one has to use back-propagation of the neural network. We opted for a modified version of gradient descent, AdaGrad [2], which allows to automatically determine optimal learning rates for the gradient descent scheme.

1.2 Distributed Optimization

Finding optimal θ in Eq. 2 entails solving very large-scale optimization problems that are computational and memory intensive. Due to memory and computational resources limitations, this amounts to a problem that is not suitable for sequential processing rather requiring a high degree of parallelism. One option is to use stochastic gradient descent (SGD). However, it is slow and cannot be operated efficiently in a distributed fashion, e.g. in a cluster environment. In order to scale the optimization for the large-scale problem presented in this paper, we adopted the alternating direction method of multipliers (ADMM) [1]. This frameworks has the advantageous properties of readily providing decomposability of the minimization, while providing comparably favorable convergence properties. We assume to have a cluster of $m \in [1..M]$ computational nodes. Every

node (worker) processes a subset of data, $\mathcal{X}_m$, and updates its local parameters, θ_m. Workers communicate their local parameters with the master node, which holds its own parameter θ. At convergence, the workers and the master should reach *consensus* in term of parameters, which can be written as the following optimization problem:

$$\theta = \arg \max_{\theta,\theta_1,\cdots,\theta_M} \sum_{m=1}^{M} \left(\underbrace{J(\theta_m; \mathcal{X}_m, \mathcal{Y}_m) + (\theta_i - \theta)^T u_m + \frac{\rho}{2}\|\theta_i - \theta\|_2^2}_{L(\theta_m;\theta,u_m,\mathcal{X}_m,\mathcal{Y}_m)} \right),$$

$$\text{subject to:} \quad \theta_1 = \cdots = \theta_M = \theta$$

where we refer to θ and u_m as master and dual variables, respectively. $L(\theta_m; \theta, u_m, \mathcal{X}_m, \mathcal{Y}_m)$ is the objective function processed by node m that optimizes local parameters θ_m given the batch data, $(\mathcal{X}_m, \mathcal{Y}_m)$, and the parameters of the master node θ, and the dual parameter u_m.

At the iteration t, the worker m updates its parameters as,

$$\theta_m^{t+1} = \arg \max_{\theta_m} L(\theta_m; \theta, u_m, \mathcal{X}_m, \mathcal{Y}_m) \tag{3}$$

and the master and the dual variables are updated according to

$$\theta^{t+1} = \frac{1}{M} \sum_{m=1}^{M} \left(\theta_m^{t+1} + \frac{1}{\rho} u_m^t \right) \qquad u_m^{t+1} = u_m^t + \rho \left(\theta_m^{t+1} - \theta^{t+1} \right). \tag{4}$$

The variable $\rho \in \mathbb{R}$ is referred to the ADMM penalty parameter and determines the speed of convergence. Following [1], a dynamic scheme was chosen to adjust ρ.

2 Parameters

For the 3D patch of the following dimensions are set: $41 \times 41 \times 41$ with the central voxels number $L = 3$. In terms of CNN, a 9 layer network was chosen - see Tab. 1 for parameters. In terms of ADMM parameters, the following are set: $\mu = 10, \quad \tau^{inc} = \tau^{dec} = 1.01055$. For learning a batch size of 50 was chosen. The networks was trained for 5000 iterations.

CNN		
1.	Convolution, Max-Pooling	Filter: $12 \times 12 \times 12$, Pooling Size: 2, Stride: 2
2.	Convolution, Max-Pooling	Filter: $8 \times 8 \times 8$, Pooling Size: 2, Stride: 1
3.	Convolution, Max-Pooling	Filter: $5 \times 5 \times 5$, Pooling Size: 2, Stride: 2
4.	Convolution, Max-Pooling	Filter: $3 \times 3 \times 3$, Pooling Size: 2, Stride: 1
5.	Rectified Linear Unit (ReLU)	
6.	Inner Product	Neurons: 500
7.	Rectified Linear Unit (ReLU)	
8.	Inner Product	Neurons: $L \times K$
9.	Softmax Regression Layer	

Table 1. Parameters of the convolutional deep neural network.

References

1. Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J.: Distributed optimization and statistical learning via the alternating direction method of multipliers. Found. Trends Mach. Learn. 3(1), 1–122 (Jan 2011), http://dx.doi.org/10.1561/2200000016
2. Duchi, J., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. J. Mach. Learn. Res. 12, 2121–2159 (Jul 2011), http://dl.acm.org/citation.cfm?id=1953048.2021068
3. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. In: Proceedings of the IEEE. vol. 86, pp. 2278–2324 (1998)