Search for Gender Difference in Functional Connectivity of Resting State fMRI

© Dmitry Kovalev[1] © Sergey Priimenko[2] © Natalya Ponomareva[3]

[1] Federal Research Center "Computer Science and Control" of Russian Academy of Sciences, Moscow, Russia

[2] Lomonosov Moscow State University, Moscow, Russia

[3] Research Center of Neurology, Moscow, Russia

dkovalev@ipiran.ru mior12@mail.ru ponomare@yandex.ru

Abstract. During past several year huge sets of fMRI data were obtained within Human Connectome Project. Despite this, technologies for scalable analysis of large amounts of data are rarely used to analyze whole data set. Authors conducted virtual experiment on a large sample of data taken from the HCP to find the gender differences in functional connectivity. A review of methods for search for the functional connectivity is fulfilled. Further analysis of distributed use and scalability on large datasets of rfMRI data is provided with the discussion of existing libraries and suggestions of how to integrate them with a distributed system. As a result, the distributed architecture of the software based on the Apache Spark framework is developed. Being fairly complex, it includes ontology, conceptual schema and workflow. The results of this experiment may be of interest to neurophysiologists for further analysis.

Keywords: data intensive research, distributed infrastructure, problem solving in neurophysiology.

1 Introduction

Today in many branches of science it is necessary to solve problems associated with increasing scale of data [1–3]. This led to the development of specialized tools, which primarily focus on structured data, but are increasingly being adapted for more general forms [4, 5]. Yet this tools and software are not widely used in data intensive research and methodology to correctly apply them has still to be developed. Different use-cases from multidisciplinary fields can greatly impact the evolution of this methodology and tools.

One of the most prominent examples of data intensive domains is the field of neurophysiology, where the amount of data has reached petabyte scale. Neurophysiology allows to visualize the structure, functions and biochemical characteristics of the brain. In particular, approaches to find the functional connections of the brain departments are being explored [6]. One way to do that is to measure the functional connectivity between brain regions as the level of co-activation of spontaneous functional time series of resting-state fMRI [7–9].

During past several years, major projects such as the Human Connectome Project (HCP) and the 1000 connectome have started with more than a thousand people participating. Datasets are open to the scientific community. Such large-scale data warehouses could serve as the beginning for the use of technologies for analyzing large amounts of data in the neuroimaging of the human brain, yet there are some limitations. One of the reasons why the community of neurobiologists do not use tools to work with large amounts of data is that standard file formats, such as NIFTI[10], are binary and possess additional costs to deliver to distributed file systems. Another problem is that many distributed systems do not effectively perform iterative algorithms, such as principal component analysis (PCA) and the independent component analysis (ICA), which are actively used in the field of neuroimaging.

One of significant are of research in neurophysiology is the study of gender difference in functional connectivity [11]. For example, there is a study of army veterans that experience physical and psychiatric complications, including craniocerebral trauma, post-traumatic stress and depression. The integration of a large number of women into military operations attracted attention to the potential sexual differences in the frequency and recovery from craniocerebral trauma, as well as from other concomitant disorders. Understanding the role of gender-related effects can provide information on the needs for evaluating treatment for women, which can demonstrate both similarities and differences from men.

This article aims at developing approach for a distributed analysis of data intensive neurophysiology domain. The article is structured as follows. Section 2 surveys existing distributed methods ant tools to process and analyze neurophysiological datasets. Section 3

Proceedings of the XIX International Conference "Data Analytics and Management in Data Intensive Domains" (DAMDID/RCDL'2017), Moscow, Russia, October 10–13, 2017

presents domain ontology that was created to better interact with domain experts. Section 4 describes distributed programming implementation on the existing computational infrastructure, as well as output results. Section 5 concludes the article.

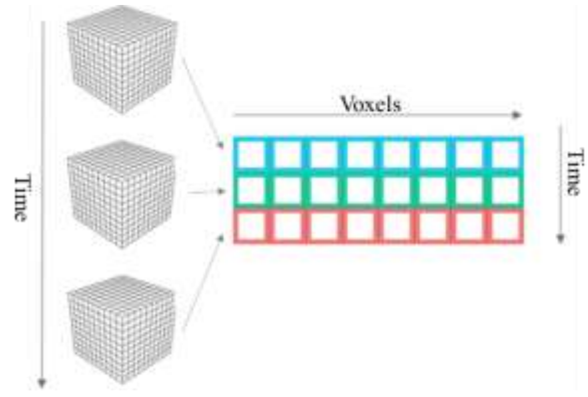

Figure 1 Transformation of 4-D array into 2-D array

2 Data analysis methods

2.1 Data processing

Resting-state fMRI dataset from the HCP project is used. The HCP consortium has developed an information platform for storing raw and processed data, systematic processing and analysis of data, obtaining and researching data. One of the main components of the project is ConnectomeDB. ConnectomeDB provides database services for the storage and dissemination of datasets that are open to the scientific community. The data is already preprocessed. Preprocessing consists of removing spatial artifacts, distortion, surface formation and alignment to a single standard space.

Data processing is divided into two parts: data cleaning inside the brain (FMRIVolume) and on the brain surface (FMRISurface) [3].

At the FMRIVolume stage, spatial distortion removal, volume redistribution due to subject movement during the session, normalization of 4D images to the standard value and creation of the final brain mask are done.

The main purpose of FMRISurface is to display time series in the standard CIFTI space. This is achieved by comparing the voxels in the cortical region of the gray matter to the native surface of the cortex and transforming each subcortical region for each individual to a standard set of voxels for each data set.

After processing the data, resting-state fMRI time series are stored in a special format – NIFTI. As a result, the data obtained with the resting-state fMRI yields more than 10 TB obtained for more than 1000 people. During the experiment, each patient was placed in a dark room and asked to relax, but not to fall asleep. The experiment was conducted in 4 sessions for 15 minutes. Two sessions of the fMRI device took pictures from the left side of the brain to the right side of the brain, and the other two sessions from the right side of the brain to the left.

2.2 Data analysis methods

The data of each subject is represented as a matrix $X_{(txv)}$ (see Fig. 1), where each row represents a set of voxels of the brain at a particular time, and each column is a time series for the corresponding voxel [12]. It is assumed that the data has already been pre-processed to remove artifacts and scaled to a standard space (coordinate system) so that the voxels are anatomically compatible for all subjects. It is also assumed that the time series of each voxel is shifted by its mean (and, possibly, normalized to the variance) [13].

If the data set consists of one object, in order to reduce the dimensionality of the data, the PCA is applied:

$$X_{(txv)} = U_{(txn)} \times S_{(nxn)} \times V^T_{(nxv)},$$

where n is the number of main components (usually much smaller than t), U is the set of temporal eigenvectors, V is the set of spatial eigenvectors, and the corresponding eigenvalues on the main diagonal of the matrix S (n largest eigenvalues). Then, ICA is applied to the matrix V, estimating a new set of spatial components that are linear combinations of the vectors of the matrix V and are maximally independent of each other. If the data set consists of several subjects, then initially all the data is combined into one large set consisting of s subjects, and then PCA and ICA are applied. The resulting approximation will be the same as above, but now with dimensions $n * s \times t$ (see Fig. 2).

With large data sets, or with a large number of subjects, it becomes unreasonable to form a complete set of data, and then apply PCA and ICA due to memory and time limitations. To solve this problem, several algorithms were invented.

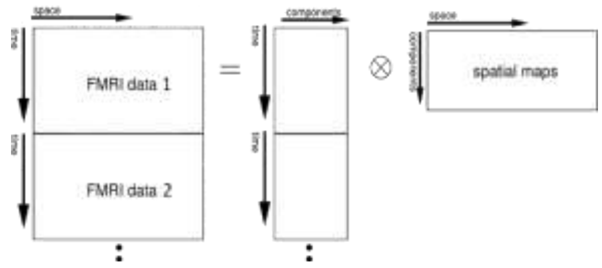

Figure 2 PCA for concatenated data

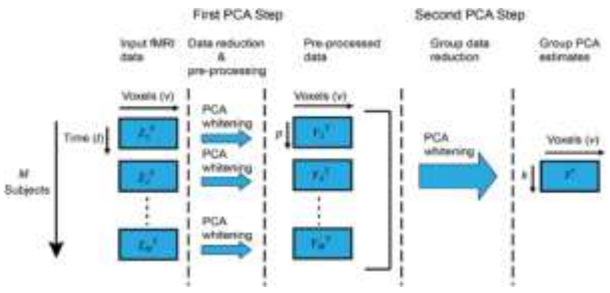

Figure 3 Parallel execution of PCA

In 2001, it was suggested to approximate the concatenation of all data sets by first reducing each set of data to m main spatial vectors using PCA and then concatenating them and applying the final PCA to reduce the final dataset to n components and then apply ICA [14]. Although using a small value of m limits the memory requirements for these operations, the data size is scaled linearly with the number of objects, which can eventually become impractically large. In addition, an important piece of information can be lost if m is not relatively large (usually it should not be large). Information can be difficult to assess at the level of an individual subject, but it can be important at the group level (see Fig. 3).

To overcome these limitations, the MELODIC's Incremental Group-PCA (MIGP) algorithm was proposed[15]. MIGP is an incremental approach, the goal of which is to provide a very close approximation to the complete concatenation of the data set followed by the PCA, but without large memory requirements. High accuracy is achieved due to the fact that individual sets of subjects' data do not decrease to a small number of components of PCA. The incremental approach preserves the inner space of PCA from m weighted spatial eigenvectors, where m is usually larger than the number of time points in each individual data set. By "weighted" is meant that the eigenvalues are included in the matrix of spatial eigenvectors. The final set of m components representing the temporarily concatenated output of the PCA can then be reduced to the required dimension n simply by storing the upper n components and, if necessary, discarding the weighting coefficients (eigenvalues).

Usually, 2–3 sets of data are first concatenated. This data set is then fed into an m-dimensional PCA and following matrix is obtained: $W_{(m\times v)} = S_{(m\times m)} \times V^T_{(m\times v)}$. Each vector is multiplied by its own value. The eigenvalues characterize the importance of the component here, so statistical information is not lost. W becomes the current evaluation of the group set and can be considered as a matrix of pseudo-series consisting of m time points and v voxels. For each data set of each subject, we gradually update W by combining W with each data set X_i and applying the ICA to get the updated W, saving only m main components. Thus, the variance of each batch of data is preserved (see Fig. 4).

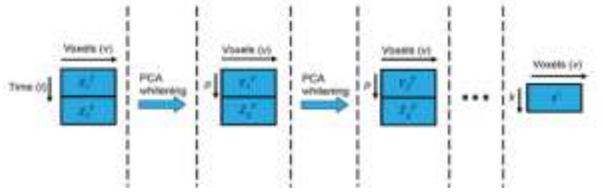

Figure 4 MELODIC Incremental Group PCA

MIGP does not increase the memory requirement with an increase in the number of subjects, large matrices are never formed, and the computation time varies linearly with the number of objects. This is easily parallelized by applying the approach in parallel to subsets of entities, and then combining them using the same approach of "concatenation and reduction" described above.

2.3 Libraries

Nibabel [16] is a library that provides an API for reading and writing some common file formats for neuroimaging. These formats include: ANALYZE (plain, SPM99, SPM2 and higher), GIFTI, NIfTI1, NIfTI2, MINC1, MINC2, MGH and ECAT, as well as Philips PAR/REC. Different image format classes provide full or selective access to header information (meta), and access to image data is made available through the arrays of the numpy library.

Objects of the image of nibabel consist of three elements:

1. The *n*-dimensional array containing the image
2. Matrix of affine transformations of size 4x4, which correlates the image coordinates with the standard world coordinate space.
3. Image metadata, stored in the header.

When an image is loaded, an object of type Nifti1mage is created. The file name can have an extension of both .nii and .nii.gz.

It is worth noting that when the load function is called directly, image data is not loaded into memory, since images can be stored as a numpy array or stored on a disk. To load data from a disk, you need to call the *get_data()* function of an object of type Nifti1Image. This function returns an n-dimensional numpy array.

In addition, an object of type Nifti1Image is created from numpy arrays. To do this, one should pass an n-dimensional data array and an affine transformation matrix to the Nifti1Image constructor must.

Nitime[17] is a library for the analysis of time series in the field of neuroimaging. Nitime can be used to represent, process and analyze time series data from experimental data. The main purpose of the library is to serve as a platform for analyzing data collected in neurophysical experiments. The basic principle of nitime implementation is the division of time series representation and time series analysis.

An important feature of the nitime library is lazy initialization. Most attributes of both time series and analysis objects are used only when necessary. That is, the initialization of a time series object or an analysis object does not cause any intensive calculations. In addition, after the calculation starts, the object is saved and ensures that access to the results of the analysis will cause the calculation to be performed only when the analysis is performed for the first time. After that, the result of the analysis is saved for further use.

One of the algorithms of the nitime library is the correlation analysis of brain regions. It calculates the correlation between one time series that represents a given area of the brain, with other areas that are also represented by a time series. To calculate the correlation between regions in the nitime library, there is a SeedCoherencAnalyzer function that takes two time series inputs and returns a correlation matrix that can be used for further analysis.

Nilearn [18] is a Python module for statistical processing of neuroimaging data.

It uses scikit-learn module for multidimensional statistics with applications in intelligent modeling, classification, decoding, and connectivity analysis. Nilearn can work NiftiImage objects from the nibabel library.

Nilearn library has great functionality for working with nii-images. It allows visualizing, decoding, exploring the functional connectivity, and performing various manipulations, such as smoothing, marking and advanced statistical analysis.

Nilearn provide CanICA method that is the ICA method for analyzing fMRI data at the group level. Compared to other strategies, it brings a well-controlled group model, as well as a threshold algorithm that controls specificity and sensitivity with an explicit signal model.

In order to get a time series and build a correlation matrix for it, nilearn provides the NiftiMapsMasker object. To create an object, one needs to specify an atlas of the brain regions.Nilearn provides the ability to create a correlation matrix for independent components that iscomputed by CanICA.

3 Ontology

The study of neuroimaging with large amounts of data represents the intersection of different areas of science. In order to use the same terms and concepts, simple ontology was developed that describes the main entities used in this work and a conceptual schema that defines the types of data, constraints on these data types and the means of interaction between them. Ontology is a formal specification of shared conceptualization [19].

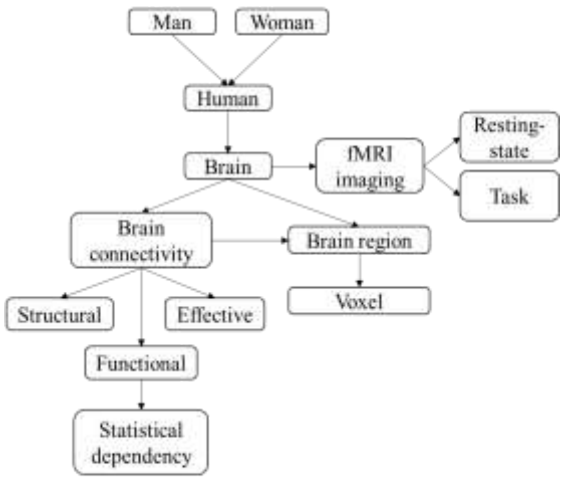

Figure 5 Main concepts of the domain ontology

The ontological specification of the subject area of neuroimaging consists of the following components (see Fig. 5):

- Neuro-image – a 3-dimensional or 4-dimensional image (a series of 3-dimensional images), reflecting the distribution of metabolic activity in different regions of the brain in different time intervals [20].
- The area of the brain is a set of voxels, sorted by a certain feature. Most often presented in the form of time series [20].
- Voxel is an element of a three-dimensional image containing some value.
- Independent models - a model for investigating thefunctional connectivity of the entire brain. They are designed to search for general patterns of functional connectivity between brain regions.Dependent models are a model for analyzing the correlation of a given region of the brain.
- Brain connectivity – the structure of anatomical connections, statistical dependencies or cause-effect interactions between individual units within the brain's nervous system [21].
- Structural connectivity refers to a network of physical or structural links linking sets of neurons or neural elements to structural biophysical features [22].
- Functional connectivity is a statistical type of connection between anatomically unconnected areas of the brain that have common functional properties [7].
- Effective connectivity – the combination of structural and functional connectivity. It describes the networks of directions of one neural element over another.
- The resting-state fMRI is a neural image obtained as a result of an experiment when the subject was at rest and did not engage in active tasks.
- The task fMRI is the neuro-images obtained as a result of the experiment, when the subject performed active actions, e. g., listened to music.

4 Implementation

4.1 Laboratory cluster specifications

Virtual experiment was executed on the laboratory cluster (see Fig. 6). It consists of 2 master nodes and 6 slave nodes. Each master node has 32Gb of RAM, 24 threads and 2 Tb of disk space in RAID1. Slave nodes have 64Gb of RAM, 24 threads and 4 Tb of disk space attaches as JBOD. All the machines are connected to 10Gbs switch.

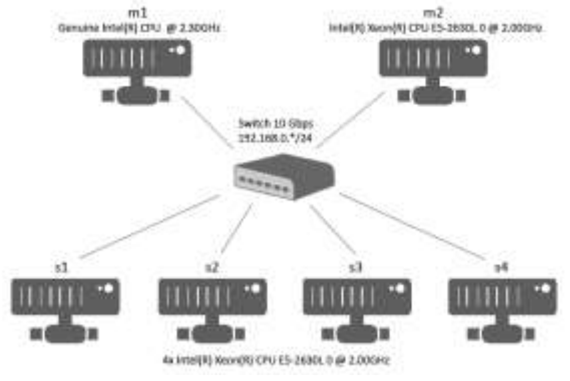
m1
Genuine Intel(R) CPU @ 2.30GHz
m2
Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz
Switch 10 Gbps
192.168.0.*/24
s1
s2
s3
s4
4x Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz

Figure 6 Cluster Architecture

On the cluster, the Hortonworks Data Platform (HDP) distribution package is installed. This distribution represents a set of tools from the Hadoop infrastructure running Apache Ambari.

A distributed file system (HDFS) (Hadoop Distributed File System) is installed file system. HDFS consists of a NameNode server and DataNode servers. The NameNode server manages the namespace of the file system and manages the clients' access to the data. The main NameNode server is installed on the *m1*node and records all transactions associated with changing the file system metadata to a special file called *EditLog*. When rt the main NameNode server is started, it reads the HDFS image and applies all the changes to it. This is done once at startup. A similar operation is performed by the Secondary NameNode, which is installed on the *m2* machine. On machines *s1*-*s4* DataNode servers are installed, which are responsible for storing the data itself and keeping its integrity.

For the sharing, scalability and reliability of the Hadoop cluster, a resource manager YARN [5] is used. YARN offers a hierarchical approach to the cluster infrastructure. The root of the YARN hierarchy is the ResourceManager. This daemon manages the entire cluster and assigns applications to the underlying computing resources. It allocates resources (computing resources, memory, and bandwidth) for the basic NodeManager. ResourceManager interacts with ApplicationMaster when allocating resources and with NodeManager when starting and monitoring basic applications. ResoureManager is located on *m2*, and NodeManager on nodes *s1*–*s4*.

Another important module for the Hadoop cluster is the Zookeper. ZooKeeper is a server that coordinates distributed processing. It provides a distributed configuration service, a synchronization service, and a registry of names for distributed systems. Distributed applications use ZooKeeper to store and notify updates of important configuration information. The Zookeper server is running on the *m1* node.

Since most of the calculations are iterative algorithms, Apache Spark was chosen as the computational backbone. Apache Spark provides a fast and versatile platform for data processing. In comparison with Hadoop, Spark accelerates the work of programs by minimizing disk input-output operations.

In Spark, the concept of RDD (stable distributed data set) is introduced – an unchangeable fault-tolerant distributed collection of objects that can be processed in parallel. RDD can contain objects of any type. RDD is created by loading an external data set or distributing a collection from the main program (driver program). In RDD, two types of operations are supported:

- Transformations are operations (for example, mapping, filtering, merging, etc.) performed over RDD. The result of the transformation is a new RDD containing its result.
- Actions are operations (eg, reduction, count, etc.) that return a value that results from some calculations in RDD.

The cluster has Spark History Server installed on *m1*, Spark Thrift Server on *m2*, Livy Server on *m2* and Spark Clients on all nodes.

For more convenient programming on a cluster, we use Apache Zeppelin – a web-based notebook that allows to conduct interactive data analytics. It supports many interpreters, including the Spark interpreter and the Python interpreter.

Scalability. As of algorithm used, each slave machine handles several independent fMRI images, so scalability increases almost linearly with using more slave nodes. It is bounded by the network speed when transmitting initial image data into slave memory, however the transmission time is several seconds and is negligible compared to processing time.

4.2 Workflow

Workflow is depicted on Fig. 7. The program reads all files from the directory, checks the validity of the format (all data are compressed zip folders). After that, the subject number is extracted from the file name and its gender is checked using an additional metadata file. When the gender is known, the file is unzipped to the corresponding folder. Inside the unzipped folder is a 4-D image in the .nii.gz format. Using the *nibabel* library, the image is loaded into memory as an array of type *numpy.array*. From this array, a new array is created with information about the spatial coordinates before the value of the voxel. The new array is compressed by the gzip algorithm and stored in HDFS.

Due to Apache Spark limitations files larger than 2.5 GB in binary format can not be loaded. In the uncompressed form, the sizeis 4.3 GB, so file needs to be compression. After compression, the file occupies just 700 MB.

Spark task is started with the following parameters:

- *num-executor=4* – number of executable entities;
- *executor-memory*=25 GB – the amount of memory used for one execution process;
- *executor-cores*=2 – the number of cores used for each executive entity.
- *driver-memory*=8 GB – the amount of memory used for the driver process, that is, where SparkContext is initialized.

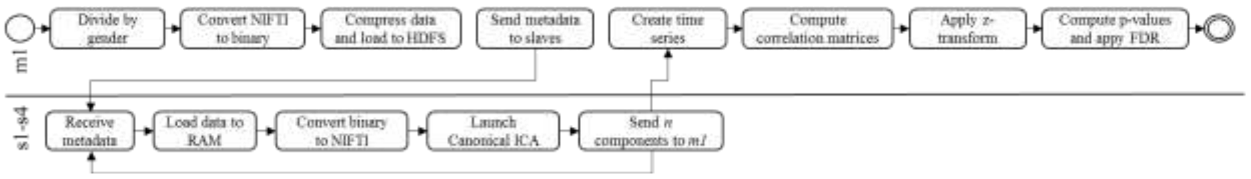

Figure 7 Workflow

YARN creates on each node a container that receives information from the driver. All calculations occur in two streams. When metadata is received, a file with a compressed binary array is loaded into memory. The program decompresses it and converts it to a normal array without information about the indexes. Then, using the resulting array and affinity transformation matrix, Nifti1Image and the CanICA object are created with the following parameters: *n_components*=20*;* *Smoothing_fwhm*=6*;* *N_init*=10*;* *Threshold*=3*;* *Verbose*=10.

The CanICA object is passed to the Nifti1Image object and an image consisting of 20 components is output. This image is returned to the m1 driver. Thus, each node receives a portion of the paths to the compressed images, processes them, and returns the result to the driver. The task is executed until all the files specified for analysis on the m1 driver are processed. When the nodes complete the tasks, the driver comes with a list of Nibabel1Image objects that contain independent components. The data for all objects is averaged and a time series is created using the NiftiLabelsMasker object.

A map of regions of the brain is transferred to the constructor of the NiftiLabelsMasker object. Using the ConnectivityMeasure object, which is created with the correlation parameter, the correlation matrix for the brain regions is considered.The correlation matrix for men and women is calculated separately. After this, the Fisher transform (z-transform) is applied to each matrix.

After a new sample is calculated, which is obtained as the difference between the male z_m obtained and the female sample z_w. This sample will have a normal

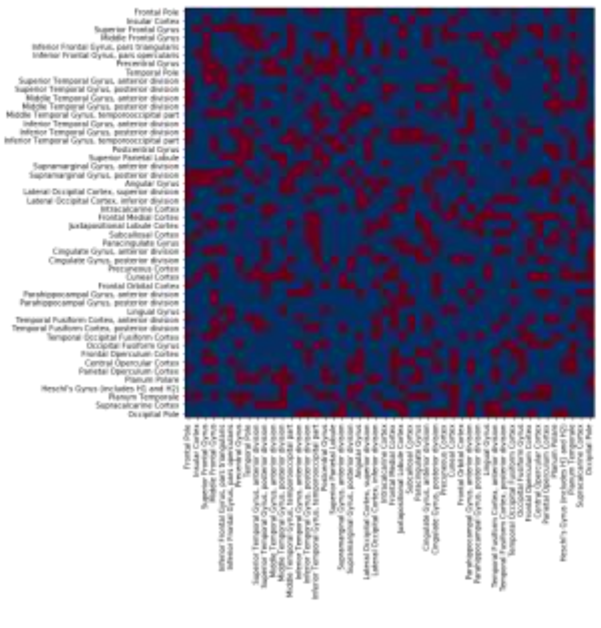

Figure 8 Binary matrix of functional connectivity difference

distribution with a mathematical expectation of 0 and a variance of 2/(n-3). For this sample, calculates a critical area with a significance level of 0.05 and c is corrected for multiple testing of the Benjamin–Hochberg hypotheses. As a result, a binary matrix is obtained that shows the deviation or acceptance of hypotheses for each area of the brain.

4.3 Results

In total 50 male and 50 female subjects are used. All data is resting-state fMRI images.

Fig. 8 depicts a binary matrix of gender differences in the functional connectivity of healthy middle-aged people. Red spots mark areas that correlate both in men and women, and blue dots indicate a lack of correlation. For example, this experiment shows that the upper front (Superior Frontal Gyrus) of the brain has a significant correlation with the insular cortex (Insular Cortex), but does not have a significant correlation with the front part (Frontal Pole) of the brain.

The independent components of averaged male subject show a greater functional connectivity compared to women. It can be seen that the main activity of the brain of men and women occurs near its cortex.

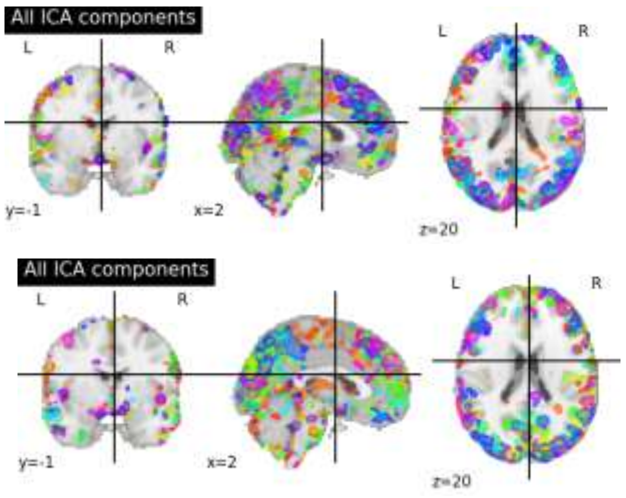

Figure 9 Averaged independent components for men (upper) and women (lower)

5 Conclusion

This paper presents distributed methods and means for searching gender differences in functional connectivity of resting-state fMRI were explored. Several methods for the search for functional connectivity of functionally magnetic resonance tomography of human rest are considered. To work with large amounts of data, machine learning methods were used to identify repetitive patterns and to intelligently reduce data. Their possibilities of parallel and distributed use and scaling are investigated with large amounts of input data. For the sake of better communication with domain experts the domain ontology was specified with main entities that describe this area and the necessary links between them.

The review of existing means of preparation and preprocessing of data on local and distributed systems is carried out. At the moment there are few libraries for working with the NIFTI format on a distributed system, so the input and output procedures for data were implemented in this work. To preprocess the data, we used method compositions from the nibabel and nilearn libraries.To solve the problem, an overview of existing distributed systems was made, among which the Apache Spark framework was most effective. For the experiment, a cluster of 6 machines was taken, where the two machines were the main nodes, and 4 the workers. On the cluster, the minimum set of programs required for the experiment, such as YARN, HDFS, ZooKeeper, Spark and Zeppelin notebook was installed and configured.

A virtual experiment was performed in a distributed system. The time of this experiment was 4 hours for 400 GB of data. As a result of the experiment, matrices of connectivity between the brain regions of men and women were obtained, as well as a binary matrix of gender differences in functional connectivity.

Acknowledgments

This research was partially supported by the Russian Foundation for Basic Research (projects 15-29-06045, 16-07-01028).

References

[1] Council, N.R.: Frontiers in Massive Data Analysis. The National Academies Press, Washington, DC (2013)

[2] Hey, A.J., Tansley, S., Tolle, K.M., others eds: The Fourth Paradigm: Data-Intensive Scientific Discovery. Microsoft Research Redmond, WA (2009)

[3] Van Essen, D.C., Smith, S.M., Barch, D.M., Behrens, T.E., Yacoub, E., Ugurbil, K., Consortium, W.-M.H., others: The WU-Minn Human Connectome Project: An Overview. Neuroimage. 80, 62–79 (2013)

[4] Zaharia, M., Xin, R.S., Wendell, P., Das, T., Armbrust, M., Dave, A., Meng, X., Rosen, J., Venkataraman, S., Franklin, M.J., others: Apache Spark: A Unified Engine for Big Data Processing. Communications of the ACM, 59, pp. 56-65 (2016)

[5] Vavilapalli, V.K., Murthy, A.C., Douglas, C., Agarwal, S., Konar, M., Evans, R., Graves, T., Lowe, J., Shah, H., Seth, S., others: Apache Hadoop yarn: Yet Another Resource Negotiator. In: Proc. of the 4th annual Symposium on Cloud Computing. p. 5. ACM (2013)

[6] Huth, A.G., Heer, W.A. de, Griffiths, T.L., Theunissen, F.E., Gallant, J.L.: Natural Speech Reveals the Semantic Maps that Tile Human Cerebral Cortex. Nature, 532, pp. 453-458 (2016)

[7] Friston, K.J.: Functional and Effective Connectivity: A Review. Brain connectivity, 1, pp. 13-36 (2011)

[8] Biswal, B.B., Kylen, J.V., Hyde, J.S.: Simultaneous Assessment of Flow and BOLD Signals in Resting-state Functional Connectivity Maps. NMR in Biomedicine, 10, pp. 165-170 (1997)

[9] Biswal, B.B., Mennes, M., Zuo, X.-N., Gohel, S., Kelly, C., Smith, S.M., Beckmann, C.F., Adelstein, J.S., Buckner, R.L., Colcombe, S., others: Toward Discovery Science of Human Brain Function. Proc. of the National Academy of Sciences, 107, pp. 4734-4739 (2010)

[10] Cox, R.W., Ashburner, J., Breman, H., Fissell, K., Haselgrove, C., Holmes, C.J., Lancaster, J.L., Rex, D.E., Smith, S.M., Woodward, J.B., others: A (sort of) New Image Data Format Standard: Nifti-1. Neuroimage, 22, e1440 (2004)

[11] McGlade, E., Rogowska, J., Yurgelun-Todd, D.: Sex Differences in Orbitofrontal Connectivity in Male and Female Veterans With TBI. Brain imaging and Behavior, 9, pp. 535-549 (2015)

[12] Smith, S.M., Hyvärinen, A., Varoquaux, G., Miller, K.L., Beckmann, C.F.: Group-PCA for Very Large fMRI Datasets. NeuroImage, 101, pp. 738-749 (2014)

[13] Beckmann, C.F., Smith, S.M.: Probabilistic Independent Component Analysis for Functional Magnetic Resonance Imaging. IEEE Transactions on Medical Imaging, 23, pp. 137-152 (2004)

[14] Calhoun, V.D., Adali, T., Pearlson, G.D., Pekar, J.: A Method for Making Group Inferences from Functional MRI Data Using Independent Component Analysis. Human Brain Mapping, 14, pp. 140-151 (2001)

[15] Rachakonda, S., Silva, R.F., Liu, J., Calhoun, V.D.: Memory Efficient PCA Methods for Large Group ICA. Frontiers in Neuroscience, 10 (2016)

[16] Gorgolewski, K., Burns, C.D., Madison, C., Clark, D., Halchenko, Y.O., Waskom, M.L., Ghosh, S.S.: Nipype: A Flexible, Lightweight and Extensible Neuroimaging Data Processing Framework in Python. Frontiers in Neuroinformatics, 5 (2011)

[17] Rokem, A., Trumpis, M., Perez, F.: Nitime: Time-Series Analysis for Neuroimaging Data. In: Proc. of the 8th Python in Science Conf., pp. 68-75 (2009)

[18] Abraham, A., Pedregosa, F., Eickenberg, M., Gervais, P., Mueller, A., Kossaifi, J., Gramfort, A., Thirion, B., Varoquaux, G.: Machine Learning for Neuroimaging With Scikit-learn. Frontiers in Neuroinformatics, 8 (2014)

[19] Sowa, J.F., others: Knowledge Representation: Logical, Philosophical, and Computational Foundations. MIT Press (2000)

[20] Poldrack, R.A.: Region of Interest Analysis for fMRI. Social Cognitive and Affective Neuroscience, 2, pp. 67-70 (2007)

[21] Van Den Heuvel, M.P., Pol, H.E.H.: Exploring the Brain Network: A Review on Resting-State fMRI Functional Connectivity. European Neuropsychopharmacology, 20, pp. 519-534 (2010)

[22] Sporns, O.: Discovering the Human Connectome. MIT Press (2012)