First Connectomics Challenge: From Imaging to Connectivity

Javier G. Orlandi* ORLANDI@ECM.UB.EDU
Estructura i Constituents de la Materia, Univ. de Barcelona, Barcelona, Spain

Bisakha Ray* BISAKHA.RAY@NYUMC.ORG
Center for Health Informatics and Bioinformatics
New York University Langone Medical Center, New York, NY, USA

Demian Battaglia DEMIAN.BATTAGLIA@UNIV-AMU.FR
Inst. for Systems Neuroscience, Univ. Aix-Marseille, Marseille, France
Bernstein Ctr. for Computat. Neurosci., Göttingen, Germany

Isabelle Guyon GUYON@CHALEARN.ORG
ChaLearn, Berkeley, CA

Vincent Lemaire VINCENT.LEMAIRE@ORANGE.COM
Orange Labs, Lannion, France

Mehreen Saeed MEHREEN.SAEED@NU.EDU.PK
Natl. Univ. of Computer Emerging Sci., Lahore, Pakistan

Jordi Soriano JORDI.SORIANO@UB.EDU
Estructura i Constituents de la Materia, Univ. de Barcelona, Barcelona, Spain

Alexander Statnikov ALEXANDER.STATNIKOV@NYUMC.ORG
Center for Health Informatics and Bioinformatics
New York University Langone Medical Center, New York, NY, USA

Olav Stetter OLAV@NLD.DS.MPG.DE
Max Planck Inst. for Dynamics and Self-Organization, Göttingen, Germany
Bernstein Ctr. for Computat. Neurosci., Göttingen, Germany

Abstract

We organized a Challenge to unravel the connectivity of a simulated neuronal networks. The provided data were solely based on fluorescence time series of spontaneous activity in a network constituted by 1000 neurons. The task of the participants was to compute the effective connectivity between neurons, with the goal to approach as accurately as possible the ground truth topology of the network. The procured data are similar to the one measured in *in vivo* and *in vitro* recordings of calcium fluorescence imaging, and therefore the algorithms developed by the participants may largely contribute in the future to unravel major topological features of living neuronal networks from just the analysis of recorded data, and without the need of slow, painstaking connectivity labeling methods. Among 143 entrants, 16 teams participated in the final round of the challenge to compete for prizes. The winners significantly outperformed the baseline method provided by the organizers. To measure influences between neurons the participants used an array of diverse

* The two first authors contributed equally. The remaining authors are in alphabetical order.

methods, including transfer entropy, regression algorithms, correlation, deep learning, and network deconvolution. The development of "connectivity reconstruction" techniques is a major step in brain science, with many ramifications in the comprehension of neuronal computation, as well as the understanding of network dysfunctions in neuropathologies.

Keywords: neuronal networks, effective connectivity, fluorescence calcium imaging, reconstruction, graph-theoretic measures, causality.

1. Introduction

All living neuronal tissues, from the smallest *in vitro* culture up to the entire brain, exhibit activity patterns that shape the modus operandi of the network. Activity may take the form of spontaneous discharges, as occurs in the absence of stimuli, or in the form of precise patterns of activity during information processing, memory, or response to stimuli. A major paradigm in modern neuroscience is the relation between the observed neuronal activity (function) and the underlying circuitry (structure). Indeed, activity in a living neuronal network is shaped by an intricate interplay between the intrinsic dynamics of the neurons and their interconnectivity throughout the network.

In the quest for understanding the structure-function relationship, the neuroscience community has launched a number of endeavors which, in an international and cooperative effort, aim at deciphering with unprecedented detail the structure of the brain's circuitry (connectome) and its dynamics (Kandel et al., 2013; Yuste and Church, 2014). In Europe, the Human Brain project aspires at developing a large-scale computer simulation of the brain, taking advantage of the plethora of data that is continuously being gathered. In the United States, the BRAIN Initiative aims at developing technologies to record neuronal activity in large areas of the brain, ultimately linking single-cell dynamics, connectivity, and collective behavior to comprehend brain's functionality. The difficulty and high cost of these quests (Grillner, 2014) have called for parallel, more accessible strategies that can complement these large-scale projects.

With the hope to delineate parallel strategies in the understanding of neuronal circuits, we launched in April 2014 a 'Connectomics Challenge' aimed at developing computational tools to answer a simple yet defying question: how accurately can one reconstruct the connectivity of a neuronal network from activity data? To shape the challenge, we built a numerical simulation in which we first designed a neuronal circuit, therefore establishing its ground–truth topology, and later simulated its dynamics considering neurons as leaky integrate-and-fire units.

The network that we simulated mimics the spontaneous activity observed in neuronal networks *in vitro*. Indeed, neuronal cultures, i.e. neurons extracted from brain tissue and grown in a controlled environment (Fig. 1A), constitute one of the simplest yet powerful experimental platforms to explore the principles of neuronal dynamics, network connectivity, and the emergence of collective behavior (Eckmann et al., 2007; Wheeler and Brewer, 2010). The relative small size of these networks, which typically contain a few thousand neurons, allow for the monitoring of a large number of neurons or the entire population (Spira and Hai, 2013; Orlandi et al., 2013; Tibau et al., 2013). The subsequent data analysis —often in the context of theoretical models— provides the basis to understand the interrelation between the individual neuronal traces, neuronal connectivity, and the emergence of collective behavior. Activity in cultures can be recorded by a number of techniques, from direct

electrical measurements (Spira and Hai, 2013) to indirect measurement such as fluorescence calcium imaging (Grienberger and Konnerth, 2012; Orlandi et al., 2013), which uses the influx of Calcium upon firing to reveal neuronal activation (Fig. 1B). Although Calcium imaging has a typical temporal resolution on the order of ms, its non-invasive nature and the possibility to simultaneously access a large number of neurons with accurate spatial resolution (only limited by the optical system for measurements) have made it a very attractive experimental platform both *in vitro* and *in vivo* (Bonifazi et al., 2009; Grewe et al., 2010).

2. Challenge Design

The goal of the Challenge was to identify directed connections of a neuronal network from observational data. Using this kind of data constitutes a paradigm shift from traditional approaches based on interventional data and causal inference, where a planned experiment is required to perturb the network and record its responses. Although interventional approaches are required to unambiguously unravel causal relationships, they are often costly and many times technically impossible or unethical. On the other hand, observational data, which means recordings an unperturbed system, can be used to study much larger systems and for longer periods.

The data for the challenge were generated using a simulator previously studied and validated (Stetter et al., 2012; Orlandi et al., 2014) for neuronal cultures. As shown in Fig. 1, mature neuronal cultures usually develop into a bursting regime, characterized by long periods of very low neuronal activity and short periods of very high (bursting) activity (Orlandi et al., 2013; Tibau et al., 2013). This is a very interesting regime to check connectivity inference algorithms, since the system switches from a scenario where connections play almost no role to another one where the system appears to be highly coherent with effective all-to-all connectivity profiles (Stetter et al., 2012). Although these two dynamic states shape different effective connectivities, the actual structural connectivity layout remains unchanged.

Connectivity inference techniques have usually focused on analyzing spiking data, with binary signals identifying the presence (1) or absence (0) of neuronal firing. However, real spiking data are only available for a narrow set of experimental systems, and usually involve invasive electrode arrays or single-cell (path clamp) techniques. Recent advances in imaging allow the simultaneous recording of thousands of neurons (Ohki et al., 2005; Panier et al., 2013). However, the identification of single spikes in imaging data cannot always be accomplished and one has to directly analyze the fluorescence signals. Our data also take that into account and the signal given to participants models the fluorescence signal of a calcium marker activated when a neuron fires. It also takes into account most of the experimental limitations, such as low acquisition speed, noisy data, and and light scattering artifacts (Stetter et al., 2012). The latter is important, since the fluorescence of a neuron influences the neighboring ones, giving rise to correlations between signals that are spurious.

The major features of the simulated networks for the Challenge are the following:

- **Network structure**. Our simulated data are based on experimental recordings at different cell densities in an area of roughly 1 mm^2 (Stetter et al., 2012; Tibau et al.,

2013). In that region all neurons are able to physically reach any other neuron and the network can be considered as a random graph. For the small training datasets we used $N = 100$ neurons with an average connectivity of $\langle k \rangle = 12$ and varying levels of clustering (Guyon et al.), from 0.1 to 0.6, and the neurons were placed randomly in a 1×1 mm square area. For the larger datasets however, we used a different network structure that was never revealed to the participants. This network is shown in Figure 2A, and its *reconstruction* by the participants shaped the overall goal of the challenge.Those datasets (including the ones used for the final scores) consisted of $N = 1000$ neurons. The neurons were distributed in 10 subgroups of different sizes, and each neuron connected with other neurons in the same subgroup with the same probability, yielding an internal average connectivity of $\langle k_i \rangle = 12$. Each subgroup had a different internal clustering coefficient, ranging from 0.1 to 0.6. Additionally, each neuron was randomly connected with $\langle k_o \rangle = 2$ other neurons of a different subgroup (Figure 2B). All the neurons were then randomly placed on a 1×1 mm square area and their indices randomized, so the network structure was not obvious in the adjacency matrix. In fact, none of the participants reported any knowledge of the real network structure.

- **Neuron dynamics**. We used leaky integrate and fire neurons with short term synaptic depression, implemented in the NEST simulator (Gewaltig and Diesmann, 2007). For the small networks, $N = 100$, the synaptic weights were the same for any neuron and were obtained through an optimization mechanism to reproduce the observed experimental dynamics with a bursting rate of 0.1 Hz. For the big networks, $N = 1000$, we ran the optimization mechanism independently for each subnetwork and then for the whole network to also achieve the target of a 0.1 Hz bursting rate. In this way, the whole network was bursting as a single unit, but each subnetwork had a different set of synaptic weights.

- **Fluorescence model**. The fluorescence model that we used mimics the fluorescence response of calcium markers inside neurons (Stetter et al., 2012). When a neuron fires, calcium enters the cell and binds to the marker, which becomes fluorescent. This fluorescence signal has a slow response and an even slower decay time. It also saturates if the neuron fires multiple times in a short interval. Illustrative fluorescence traces of the simulated networks are shown in Figure 2C.

The network architectures used to generate the simulated data are summarized in Figure 3.

3. Results

The challenge lasted three months (from February 5 to May 5, 2014) and attracted 143 participants. The participants received immediate feed-back on validation data on a public leaderboard. Their ranking on the final test data remained hidden until the end of the challenge. The scores from the private leaderboard (calculated on test data) for the top ten ranking participants are shown in Table 1. The calculated metric is the 'area under the curve' (AUC) of a *Receiver-Operator Characteristic* analysis (Bradley, 1997; Stetter

et al., 2012), a metric commonly used in classification problems. Here, we brought back the problem of network reconstruction to a two-class classification problem: edge present or absent. The motivation for using this metric is its availability on the Kaggle platform used for the challenge and its familiarity to challenge participants. In Section 3.3, we compare this metric with the area under the *Precision Recall* (PR) curve, a metric often used in information retrieval, which could be used as an alternative scoring method.

The results of the top ranking participants who submitted their code were verified by the organizers, who successfully reproduced their results. These results and pointers to code are shown in Appendix A. The second ranked participants chose not to submit their code and renounced to their prize.

Table 1: Private leaderboard rankings of the top 10 participants (test AUC scores).

#	Team Name	Score
1	AAAGV	0.94161
2	Matthias Ossadnik	0.94102
3	Ildefons Magrans	0.94063
4	Lukasz 8000	0.93956
5	Lejlot and Rafal	0.93826
6	Sium	0.93711
7	Alexander N and Vopern	0.93666
8	gaucho 81	0.93385
9	killertom	0.93011
10	dhanson	0.92885

We also surveyed the participants to compile statistics about algorithm, software and hardware usage, as well as human and computer time spent [1].

Below we provide some general analyses of the results of the competition. Extensive details will be given in an upcoming croudsourced paper co-authored by the organizers and the participants.

3.1. Challenge duration

The graph in Figure 4 shows performance progression as a function of time, obtained from the Kaggle website. The performances increased slowly throughout the challenge, but most notably in the first two months. However, the survey indicates that only one third of the participants estimated that they had sufficient time to complete the tasks of the challenge. One third also expressed their interest to continue refining the methods.

3.2. Overfitting

The graph in Figure 5 plots the results on test data *vs.* validation data for the final submissions, limited to scores exceeding the results obtained with plain correlation (i.e. Pearson correlation coefficient with no lag and no preprocessing). We see a strong correlation

1. https://docs.google.com/a/chalearn.org/viewer?a=v&pid=sites&srcid=Y2hhbGVhcm4ub3JnfGNvbm5lY3RvbWljc3xneDo1OTkwMWM5MWJkN2NjYzZj

between the validation and test results. At low scores, the final test data seem "easier" (larger scores are obtained by most participants on test data than on validation data). Few participants overfitted by obtaining better results on validation data than on test data.

3.3. PR curves

First, we compared ROC curves and precision-recall (PR) curves, as depicted in Figure 6. We show in blue the curves of the top ranking participants, in green those of the winner (team AAAGV) and in red those of the baseline method based on Transfer Entropy. We remind that TPR is the true positive rate (fraction of correct connections found among all true connections), FPR is the false positive rate (fraction of connections erroneously guessed among truly absent links), "recall" is a **synonym** of TPR and "precision" is the fraction of correct connections found among all connections called significant.

In many ways the PR curve is more useful for experimentalists to assess the accuracy of the networks. For instance, using the green curve, we can see that, if we are willing to accept that 50% of the connections are wrong (precision of 0.5), we can retrieve 40% of the connections of the network (recall or TPR of 0.4). In contrast, the readings of the ROC curve may be deceivingly good: for a TPR of 0.4 ($\log 10(0.4) \simeq -0.4$), we obtain an FPR in 0.01, but, we care much less about correctly identifying absent connections than missing true connections.

3.4. Edge orientation

Another important aspect of the problem we posed is the capability of network reconstruction algorithms to identify the direction of the connection, not only the presence or absence of a connection. Our metric of success did not emphasize connection orientation, making it possible to obtain good results even with a symmetric matrix. To separate the algorithms with respect to edge orientation, we computed the score of the challenge (AUC) limited to the pairs of neurons having at least one connection in either direction ("connected neurons"). The results are shown in Table 2. It illustrates that edge orientation is very difficult compared to merely detectng the presence of a connection: the best score drops from 0.94 for the undirected network to 0.64 for the directed one. Team ranked number 4 (Lukasz 8000) performed best with respect to this metric. This team used a deep learning method based on convolutional neural networks. Feature learning may have played an important role in detecting details of the time series that are useful to determine edge orientation.

3.5. Subnetworks

Unknown to the participants, the large networks that we used for validation and test data had a substructure: they were organized in 10 subnetworks with varying clustering coefficients. We define the clustering coefficient as the average over the sub-network of local clustering coefficients. Local clustering coefficients introduced by (Watts and Strogatz, 1998) compute the ratio of connected neighbors of a node over the total number of possible connections. In Figure 7 we show how that the AUC scores of subnetworks (averaged over the top ten ranking participants) vary linearly with the log of the average clustering coefficients of the subnetworks.

Table 2: Analysis of edge orientation (AUC scores).

#	Team Name	Undirected Network	Directed Network
1	AAAGV	0.94	0.61
2	Matthias Ossadnik	0.94	0.63
3	Ildefons Magrans	0.94	0.60
4	Lukasz 8000	0.94	0.64
5	Lejlot and Rafal	0.94	0.63
6	Sium	0.94	0.63
7	Alexander N and Vopern	0.94	0.61
8	gaucho 81	0.93	0.61
9	killertom	0.93	0.61
10	dhanson	0.93	0.61
Mean		0.94	0.62

We also computed the "long range" AUC score, *i.e.* the AUC score restricted to connections between subnetworks. On average over all top 10 ranking participants we obtained 0.8 (compared to 0.94 for the overall network).

4. Methods

For each category of methods (pre-processing, feature selection, dimensionality reduction, classification etc.) we report the fraction of participants having used each method. Note that the sum of these numbers do not necessarily add up to 100%, because the methods are not mutually exclusive and some participants did not use any of the methods.

The algorithmic steps for network reconstruction could be very broadly divided into the following steps:

1. **Preprocessing of fluorescence signals**: Figure 8 summarizes the different preprocessing techniques used by the participants. Some of the methods of the participants were spike timing extraction using either filtering and thresholding techniques, or through deconvolution methods such as (Vogelstein, 2009; Vogelstein et al., 2009).

2. **Feature extraction**: Figure 9 shows the different feature extraction techniques used by the participants. Inverse correlation was used to filter out indirect interactions via fast partial correlations (Ryali et al., 2012).

3. **Dimensionality reduction**: The statistics in terms of number and percentage of participants for the different techniques used for dimensionality reduction is shown in Figure 10.

4. **Classification techniques**: Some recurrent techniques used by the participants were deep learning (Weston et al., 2012), generalizations of transfer entropy (Barnett et al., 2009; Orlandi et al., 2014) and information theoretical features, ad hoc topological features (e.g. geometric closure) and growing "a lot of trees" (random forests (Breiman,

2001), boosting methods). The statistics in terms of number and percentage of participants for the different techniques used for classification are shown in Figure 11.

We also analyzed the factsheets with respect to the hardware and software implementations:

- **Hardware:** Many participants made use of parallel implementations (80% used multiple processor computers and 13% ran analyses in parallel). Memory usage was substantial (50% used less than 32 GB and 27% less than 8 GB).
- **Software:** Most participants used Linux (50%), followed by Windows (40%) and MAC OS (30%). Python was the top choice (67%) for coding, followed by MATLAB (37%).

The amount of human effort involved in adapting the code to the problems of the challenge varied but was rather significant because about 37% of the participants reported spending more than two weeks of programming. The total machine effort varied, with 43% reporting more than a few hours while another 27% reported more than two weeks.

A brief description of the methods of the top four ranking participants is given in the Appendix. Common to all method was the importance of preprocessing, including signal discretization or inference of spike trains. But the network inference step was rather different in the various methods. The winners (AAAGV) inferred an undirected network obtained through partial correlations, estimated with inverse covariance matrix, then post-processed the network in an attempt to recover edge directions (see Sutera et al. (2014) for details). Hence this method is multivariate: it takes into account all neurons in the network, it is not solely based on pairs of neurons like the baseline method used in Generalized Transfer Entropy. Matthias Ossadnik (ranked second) used a different multivariate approach: he used multivariate logistic regression of inferred spike trains, followed by an AdaBoost classifier integrating other information, including neuronal firing rates. Ildefons Magrans (ranked third) used multiple pairwise connectivity indicators varying the preprocessing parameters, integrated by an overall classifier based on ensembles of trees (see de Abril and Nowe (2014) for details). Multivariate interactions were taken into account in that method by post-processing the connectivity matrix with network deconvolution. Lukasz 8000 (ranked fourth) used deep convolutional neuronal networks (see Romaszko (2014) for details). Although the method is sophisticated in the sense that it is based on learned features of the temporal signal, it is not multivariate in the sense that it treats pairs of neurons independently. The proceedings of the challenge also include descriptions of the method of team Lejlot and Rafal (Czarnecki and Jozefowicz, 2014), ranked 5, using several based predictors integrated with a Random Forest classifier and the method of killertom (Tao et al., 2014), ranked 9, using an improved version of Generalized Transfer Entropy (which was given as baseline method).

It is promising to see that several of the top ranking participants obtained good performance based only on statistics of pairs of neurons. Although clearly multivariate methods should provide superior performance, pairwise methods promise to scale much better to larger networks.

5. Conclusions

This first connectomics challenge allowed us to identify state-of-the-art methods to solve a difficult network reconstruction problem. The methods of the top ranking participants were very diverse and will pave the way to further research, integrating key ideas and analysis tools to increase performance. The participants performed better on the problem of edge detection than on edge orientation. More emphasis should be put on orientation in upcoming challenges. In an upcoming croudsourced paper we intend to involve both the challenge organizers and the participants in a deeper analysis of the devised strategies and analysis tools. We are also in the process of applying the methods of the top ranking participant to real biological data to assess their ability to reveal or predict key connectivity features of living neuronal networks. In collaboration with biologists, we are also preparing new data for upcoming connectomics challenges on real data.

Acknowledgments

This challenge is the result of the collaboration of many people. We are particularly grateful to our advisors and beta-testers who contributed to the challenge website: Gavin Cawley, Gideon Dror, Hugo-Jair Escalante, Alice Guyon, Vincent Lemaire, Sisi Ma, Eric Peskin, Florin Popescu, and Joshua Vogelstein. The challenge was part of the WCCI 2014 and ECML 2014 competition programs. Prizes were donated by Microsoft. The challenge was implemented on the Kaggle platform, with funds provided by the EU FP7 research program "Marie Curie Actions". This work used computing resources at the High Performance Computing Facility of the Center for Health Informatics and Bioinformatics at the NYU Langone Medical Center.

References

Lionel Barnett, Adam B Barrett, and Anil K Seth. Granger causality and transfer entropy are equivalent for gaussian variables. *Physical review letters*, 103(23):238701, 2009.

P Bonifazi, M Goldin, M A Picardo, I Jorquera, A Cattani, G Bianconi, a Represa, Y Ben-Ari, and R Cossart. GABAergic hub neurons orchestrate synchrony in developing hippocampal networks. *Science (New York, N.Y.)*, 326(5958):1419–24, December 2009. ISSN 1095-9203. doi: 10.1126/science.1175509.

Y-Lan Boureau, Jean Ponce, and Yann LeCun. A theoretical analysis of feature pooling in visual recognition. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 111–118, 2010.

Andrew P Bradley. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern recognition*, 30(7):1145–1159, 1997.

Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

Wojciech M. Czarnecki and Rafal Jozefowicz. Neural connectivity reconstruction from calcium imaging signal using random forest with topological features. *JMLR, proceedings track*, This volume, 2014.

Ildefons Magrans de Abril and Ann Nowe. Supervised neural network structure recovery. *JMLR, proceedings track*, This volume, 2014.

Alberto De La Fuente, Nan Bing, Ina Hoeschele, and Pedro Mendes. Discovery of meaningful associations in genomic data using partial correlation coefficients. *Bioinformatics*, 20(18): 3565–3574, 2004.

J Eckmann, O Feinerman, L Gruendlinger, E Moses, J Soriano, and T Tlusty. The physics of living neural networks. *Physics Reports*, 449(1-3):54–76, September 2007. ISSN 03701573. doi: 10.1016/j.physrep.2007.02.014.

Soheil Feizi, Daniel Marbach, Muriel Médard, and Manolis Kellis. Network deconvolution as a general method to distinguish direct dependencies in networks. *Nature biotechnology*, 2013.

Yoav Freund and Robert E Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *Computational learning theory*, pages 23–37. Springer, 1995.

Marc-Oliver Gewaltig and Markus Diesmann. Nest (neural simulation tool). *Scholarpedia*, 2(4):1430, 2007.

Benjamin F Grewe, Dominik Langer, Hansjörg Kasper, Björn M Kampa, and Fritjof Helmchen. High-speed in vivo calcium imaging reveals neuronal network activity with near-millisecond precision. *Nature methods*, 7(5):399–405, May 2010. ISSN 1548-7105. doi: 10.1038/nmeth.1453.

Christine Grienberger and Arthur Konnerth. Imaging calcium in neurons. *Neuron*, 73(5): 862–885, 2012.

Sten Grillner. Megascience efforts and the brain. *Neuron*, 82(6):1209–11, June 2014. ISSN 1097-4199. doi: 10.1016/j.neuron.2014.05.045.

Isabelle Guyon, Demian Battaglia, Alice Guyon, Vincent Lemaire, Javier G Orlandi, Mehreen Saeed, Jordi Soriano, Alexander Statnikov, Olav Stetter, and Bisakha Ray. Design of the first neuronal connectomics challenge: From imaging to connectivity.

Eric R Kandel, Henry Markram, Paul M Matthews, Rafael Yuste, and Christof Koch. Neuroscience thinks big (and collaboratively). *Nature reviews. Neuroscience*, 14(9):659–64, September 2013. ISSN 1471-0048. doi: 10.1038/nrn3578.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Andy Liaw and Matthew Wiener. Classification and regression by randomforest. *R news*, 2(3):18–22, 2002.

Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 807–814, 2010.

Kenichi Ohki, Sooyoung Chung, Yeang H Ch'ng, Prakash Kara, and R Clay Reid. Functional imaging with cellular resolution reveals precise micro-architecture in visual cortex. *Nature*, 433(7026):597–603, February 2005. ISSN 1476-4687. doi: 10.1038/nature03274.

Javier G. Orlandi, Jordi Soriano, Enrique Alvarez-Lacalle, Sara Teller, and Jaume Casademunt. Noise focusing and the emergence of coherent activity in neuronal cultures. *Nature Physics*, 9(9):582–590, 2013. doi: 10.1038/nphys2686. URL http://dx.doi.org/10.1038/nphys2686.

Javier G Orlandi, Olav Stetter, Jordi Soriano, Theo Geisel, and Demian Battaglia. Transfer entropy reconstruction and labeling of neuronal connections from simulated calcium imaging. *PLoS One*, 9(6):e98842, 2014.

Thomas Panier, Sebastián a Romano, Raphaël Olive, Thomas Pietri, Germán Sumbre, Raphaël Candelier, and Georges Debrégeas. Fast functional imaging of multiple brain regions in intact zebrafish larvae using Selective Plane Illumination Microscopy. *Frontiers in neural circuits*, 7(April):65, January 2013. ISSN 1662-5110. doi: 10.3389/fncir.2013.00065.

Boris Teodorovich Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964.

Greg Ridgeway. Generalized boosted regression models. *Documentation on the R Package gbm, version 1· 5*, 7, 2006.

Lukasz Romaszko. Signal correlation prediction using convolutional neural networks. *JMLR, proceedings track*, This volume, 2014.

Srikanth Ryali, Tianwen Chen, Kaustubh Supekar, and Vinod Menon. Estimation of functional connectivity in fmri data using stability selection-based sparse partial correlation with elastic net penalty. *Neuroimage*, 59(4):3852–3861, 2012.

Juliane Schäfer and Korbinian Strimmer. A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. *Statistical applications in genetics and molecular biology*, 4(1), 2005.

Thomas Schreiber. Measuring information transfer. *Physical review letters*, 85(2):461, 2000.

Micha E Spira and Aviad Hai. Multi-electrode array technologies for neuroscience and cardiology. *Nature nanotechnology*, 8(2):83–94, February 2013. ISSN 1748-3395. doi: 10.1038/nnano.2012.265.

Olav Stetter, Demian Battaglia, Jordi Soriano, and Theo Geisel. Model-free reconstruction of excitatory neuronal connectivity from calcium imaging signals. *PLoS computational biology*, 8(8):e1002653, 2012.

Antonio Sutera, Arnaud Joly, Vincent Francois-Lavet, Zixiao Aaron Qiu, Gilles Louppe, Damien Ernst, and Pierre Geurts. Simple connectome inference from partial correlation statistics in calcium imaging. *JMLR, proceedings track*, This volume, 2014.

Chenyang Tao, Wei Lin, and Jianfeng Feng. Reconstruction of excitatory neuronal connectivity via metric score pooling and regularization. *JMLR, proceedings track*, This volume, 2014.

Elisenda Tibau, Miguel Valencia, and Jordi Soriano. Identification of neuronal network properties from the spectral analysis of calcium imaging signals in neuronal cultures. *Frontiers in neural circuits*, 7(December):199, January 2013. ISSN 1662-5110. doi: 10.3389/fncir.2013.00199.

Joshua T Vogelstein. *OOPSI: A family of optimal optical spike inference algorithms for inferring neural connectivity from population calcium imaging*. THE JOHNS HOPKINS UNIVERSITY, 2009.

Joshua T Vogelstein, Brendon O Watson, Adam M Packer, Rafael Yuste, Bruno Jedynak, and Liam Paninski. Spike inference from calcium imaging using sequential monte carlo methods. *Biophysical journal*, 97(2):636–655, 2009.

Duncan J. Watts and Steven H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, June 1998.

Jason Weston, Frédéric Ratle, Hossein Mobahi, and Ronan Collobert. Deep learning via semi-supervised embedding. In *Neural Networks: Tricks of the Trade*, pages 639–655. Springer, 2012.

BC Wheeler and GJ Brewer. Designing neural networks in culture. *Proceedings of the IEEE*, 98(3), 2010.

Rafael Yuste and George M. Church. The New Century of the Brain. *Scientific American*, 310(3):38–45, February 2014. ISSN 0036-8733. doi: 10.1038/scientificamerican0314-38.

Appendix A. Challenge Verification Results

1. **Winners prize #1 (first place, verified) 500 USD and 1000 USD travel award + Award certificate**

   AAAGV The code from the winning team AAAGV, which is publicly available at https://github. com/asutera/kaggle-connectomics, was run successfully on a desktop PC, it used 7 GB of RAM and it took 30h to run in single core mode on a 3 GHZ i7 CPU for each dataset. The code is built in Python and only uses standard dependencies. There was a issue with a specific library version but this has been resolved. Also we only need to run 1 script for the whole computation (main.py). From the valid dataset we obtained an AUC of 0.9426 and for the valid dataset and 0.9416 for the test dataset, which are the same as the ones reported in Kaggle.

2. **Winners prize #2 (third place, verified) 250 USD and 750 USD travel award + Award certificate**

   Ildefons

Ildefons code, which is publicly available here https://github.com/ildefons/connectomics consisted of 6 separate scripts. The following are the time and memory requirements for each of the scripts. The main challenges were installing the required R package gbm and his script makeFeatures.R which needed 128 G. This R script started a MATLAB server in the SGE (Sun Grid Engine) background. We had to execute makeFeatures.R separately for normal-1, normal-2, valid, and test. His code was executed on the standard compute nodes on the cluster. The compute nodes have 2 INTEL CPUs, 16 processing cores, and 128 GB RAM. The statistics for the execution of his code can be found in Table 3.

Table 3: Memory Requirements and Time for Ildefons' code

Script	**Time (days:hours: minutes: seconds)**	**Memory**
makeMat.R	09:29	10.937G
makeoo.m	04:22:15	9.617G
makeFeatures.R	normal-1: 2:07:37:25 normal-2: 12:28:46 valid: 12:24:17 test: 12:24:47	normal-1: 30.051G normal-2: 22.287G valid: 23.046G test: 23.055G
normalizeFeatures.R	48:44	44.541G
fitModels.R	02:05:38	12.339G
createSolution.R	10:23	27.082G

The code passed verification successfully. His AUC for the Kaggle submission generated by us is 0.94066. This is better than his leader board score of 0.93900. The difference between the two scores is 0.00166.

3. **Winners prize #3 (fourth place, verified) 100 USD and 400 USD travel award + Award certificate**

   Lukasz Romaszko

   The code of this team is found at: https://github.com/lr292358/connectomics. The details for Lukasz's code can be found in Table 4. His solution involved predicting the outcomes eight different times and averaging. All of his code passed verification successfully. The bottlenecks were installing theano (Python module) on the GPU units and gaining access to the GPU units. We have 5 cluster nodes with GPU accelerators. Each node has 1 accelerator. Each GPU has 2496 cores. The accelerator is NVIDIA Tesla Kepler (K20).

   After merging, his score is 0.93931, which is slightly better than his score of 0.93920 on the leader board. The difference between the two is is 0.00011 or, in other words, negligible.

Appendix B. Description of Sample Methods and Sample Code

Matlab: We provide Matlab sample code to:

Table 4: Memory Requirements and Time for Lukasz's code

Seed	**Max Memory**	**Time (days:hours: minutes: seconds)**	**AUC**
1	31.566 G	2:23:47:32	0.93618
2	31.566 G	2:23:24:37	0.93663
3	31.566 G	3:00:18:40	0.93646
4	31.566 G	3:00:28:06	0.93614
5	31.566 G	2:23:50:08	0.93618
6	31.566 G	2:23:52:20	0.93564
7	31.566 G	2:23:51:33	0.93658
8	31.566 G	2:23:42:50	0.93579

- read the data
- prepare a sample submission
- visualize data
- compute the GTE Stetter et al. (2012) coefficient and a few other causal direction coefficients
- train and test a predictor based on such coefficients.

The Matlab sample code is suitable to get started. We provide a script (challengeFast-Baseline) that computes a solution to the challenge (big "valid" and "test" datasets) in a few minutes, on a regular laptop computer. This uses Pearson's correlation coefficient (Correlation benchmark, AUC = 0.87322 on the public leaderboard). The data are first discretized with a simple method. Using more elaborate discretization methods such as OOPSI may work better. The other network reconstruction methods, including GTE, are not optimized: they are slow and requires a lot of memory.

C++: Network-reconstruction.org provides C++ code which would help participants to:

- read the data
- prepare a sample submission
- compute the GTE coefficient and a few other causal direction coefficients

Note: The fluorescence matrices for small networks have dimension 179498 x 100 and of large networks 179500 x 1000. Even though the GTE code is "optimized" it is still slow and requires 10-12 hours of computation for the big 1000 neuron networks on a compute cluster.

Python: We are providing scripts that:

- read the data
- discretizes
- prepare a sample submission using correlation.

One participant also made Python code available.

The baseline network reconstruction method, which we implemented, is described in details in Stetter et al. (2012). It is based on Generalized Transfer Entropy (GTE), which is an extension of Transfer Entropy first introduced by Schreiber Schreiber (2000), a measure that quantfies predictive information flow between stationary systems evolving in time. It is given by the Kulback-Leibler divergence between two models of a given time series, conditioned on a given dynamical state of the system, which in the case of fluorescence signals corresponds to the population average. Transfer Entropy captures linear and non-linear interactions between any pair of neurons in the network and is model-free, i.e., it does not require any a priori knowledge on the type of interaction between neurons. Apart from GTE, we have also provided the implementation of cross correlation and two information gain (IG) measures based on entropy and gini for network reconstruction. Cross correlation gives best results when there are zero time delays, which reduces it to a simple correlation coefficient measure. Hence, all these methods treat the data as independent instances/points in space instead of time series data. Another module that we have added to our software kit is a supervised learner, which extracts features from a network whose ground truth values are known and builds a simple linear classifier for learning whether a connection is present between two neurons or not. Currently, the features extracted are GTE, correlation, information gain using gini and information gain using entropy.

Appendix C. Description of the Algorithms of the Winners

We provide a high level description of the method of the top ranking participants provided in their fact sheets.

Team: AAAGV

The key point is building an undirected network through partial correlations, estimated through inverse covariance matrix. As preprocessing they use a combination of low and high pass filters to filter the signals and they try to filter out bursts or peak neural activities. They stress that their main contribution is the preprocessing of the data. The calcium fluorescence signal is generally very noisy due to light scattering artifacts. In the first step, a low pass filter is used to smooth the signal and filter out high frequency noise. To only retain high frequency around spikes, the time series is transformed into its backward difference. A hard-threshold filter is next applied to eliminate small variances and negative values. In a final step, another function is applied to magnify spikes that occur in cases of low global activity.

For inference, this team assumed that the fluorescence of the neurons at each point can be modeled as random variables independently drawn from the same time-invariant joint probability distribution. They then used partial correlation to detect direct associations

between neurons and filter out spurious ones. Partial correlation measures contional dependence between variables and has been used for inference in gene regulatory networks De La Fuente et al. (2004); Schäfer and Strimmer (2005).

As the partial correlation matrix is symmetric, this method was not useful in detecting directionality. Some improvement was obtained by choosing an appropriate number of principal components. The method was sensitive to the choice of filter parameters.

Team: Matthias Ossadnik

He uses multivariate logistic regression of inferred spike trains (thresholded derivative signals). Then the scores of the regressive model are fed into a modified AdaBoost Freund and Schapire (1995) classifier together with other information, such as neuronal firing rates.

Team: Ildefons Magrans

Ildefons designed a feature engineering pipeline based on information about connectivity between neurons and optimized for a particular noise level and firing rate between neurons. Instead of using a single connectivity indicator, he optimizes several indicators. As a first step, he used OOPSI, which is based on the sequential Monte-Carlo methods, in his spike inference module. Spikes below a noise-level are treated as background noise and removed. After that, time steps containing spiking activity above the synchronization rate are removed as inter-bursts recordings are more informative for topology reconstruction. As connectivity indicator, he used plain correlation which however did not provide any directionality information. In order to eliminate arbitrary path lengths caused by direct and indirect effects, he used network deconvolution Feizi et al. (2013) which takes into account the entire connectivity matrix. The classifiers he uses with the features generated from correlation are Random Forests Liaw and Wiener (2002) and Gradient Boosting Machines Ridgeway (2006).

This method also could not identify directions of connections and correlation and the singular value decomposition step of network deconvolution had an extremely high computational complexity.

Team: Lukasz8000

Convolutional Neural Networks (CNN) go beyond feed forward neural networks in their ability to identify spatial dependencies and pattern recognition. CNNs recognize smaller patterns or feature maps in each layer eventually generalizing to more complex patterns in subsequent layers. Each convolutional layer is defined by the number and shapes of filters it has alongwith its ability to learn patterns. In addition, max pooling Boureau et al. (2010) is used to reduce the size of the generated feature maps.

He uses a deep convolutional neuronal network LeCun et al. (1998) to learn features of pairs of time-series hinting at the existence of a connection. In addition he also introduces an additional input layer, the average activity of network. Lukasz used preprocessing to retain regions of higher activity conditioned on a particular threshold. These active regions help to detect interdependencies. The other important choice which influenced results was that of an activation function. He used tanh in the first convolutional layer followed by Rectified Linear Unit Nair and Hinton (2010) in the next two layers. To improve the network structure, he used max pooling. Gradient descent was combined with momentum Polyak

(1964) and this helped to naviagte past local extrema.

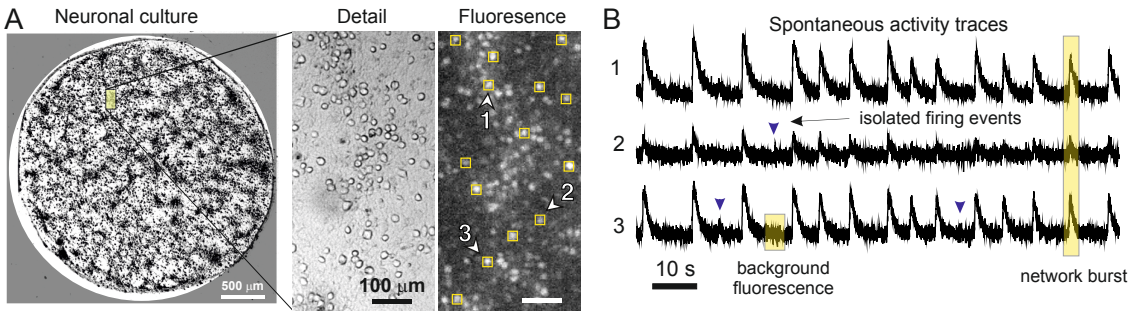


Figure 1: **Experimental motivation**. (A) Example of an *in vitro* neuronal culture, derived from a rat embryonic cortex, containing on the order of 3000 neurons. The detail shows a small area of the network in bright field and fluorescence, depicting individual neurons. In a typical experiment, neurons are identified as regions of interest (yellow boxes), and their analysis provide the final fluorescence times series to be analyzed. (B) Fluorescence spontaneous activity traces for 3 representative neurons. Data are characterized by a background signal interrupted either by episodes of coherent activity termed *network bursts*, or by individual firing events of relative low amplitude and occurrence.

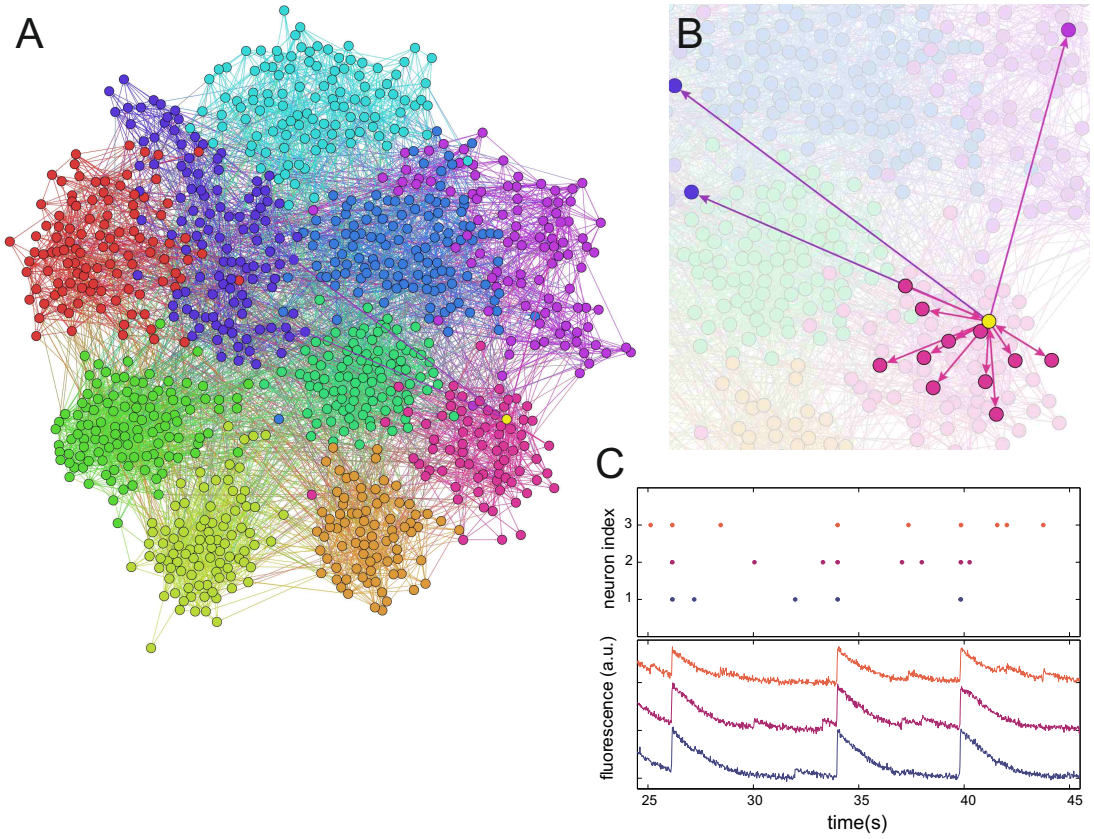


Figure 2: **Simulated neuronal network for the Challenge**. (A) The designed network contained 1000 neurons preferentially connected within 10 communities (marked with different colors in the figure), and with additional connections between communities. Each neuron connected on average with 30 other neurons. (B) A detail of the connections of a single neuron. For clarity, only 50% of the connections are shown. (C) Top: Raster plot showing the spontaneous activity of 3 randomly chosen neurons in the network. Bottom: Corresponding fluorescence traces. Note the existence of both network bursts and isolated firings. Traces are vertically shifted for clarity.

Name of archive	Description	File types
validation	Fluorescence and positional data for the validation phase of the challenge (results on "public leaderboard"). Network of N=1000 neurons.	F, P
test	Fluorescence and positional data for the final test phase of the challenge (results on "private leaderboard"). Network of N=1000 neurons.	F, P
small	Six small networks with N=100 neurons. Each network has the same connectivity degree but different levels of clustering coefficient, intended for fast checks of the algorithms.	F, P, N
normal-1	Network of N=1000 neurons constructed similarly to the "validation" and "test" networks.	F, P, N
normal-2	Network of N=1000 neurons constructed similarly to the "validation" and "test" networks.	F, P, N
normal-3	Network of N=1000 neurons constructed similarly to the "validation" and "test" networks.	F, P, N
normal3-highrate	Same network architecture as normal-3, but with highly active neurons, i.e., higher firing frequency.	F, P, N
normal-4	Network of N=1000 neurons constructed similarly to the "validation" and "test" networks.	F, P, N
normal-4-lownoise	Same network architecture as normal-4 (and same spiking data) but with a fluorescence signal with a much better signal to noise ratio.	F, P, N
highcc	Network of N=1000 neurons constructed similarly to the "validation" and "test" networks, but with a higher clustering coefficient on average.	F, P, N
lowcc	Network of N=1000 neurons constructed similarly to the "validation" and "test" networks, but with a lower clustering coefficient on average.	F, P, N
highcon	Network of N=1000 neurons constructed similarly to the "validation" and "test" networks, but with a higher number of connections per neuron on average.	F, P, N
lowcon	Network of N=1000 neurons constructed similarly to the "validation" and "test" networks, but with a lower number of connections per neuron on average.	F, P, N

Figure 3: **Data table**. The data procured to the participants consisted in the fluorescence time series of simulated neuronal networks. The spatial location of the neurons in the 1×1 mm^2 area was also provided. Small networks, used for validation, contained 100 neurons. Large networks, used for the actual competition, consisted of 1000 neurons.

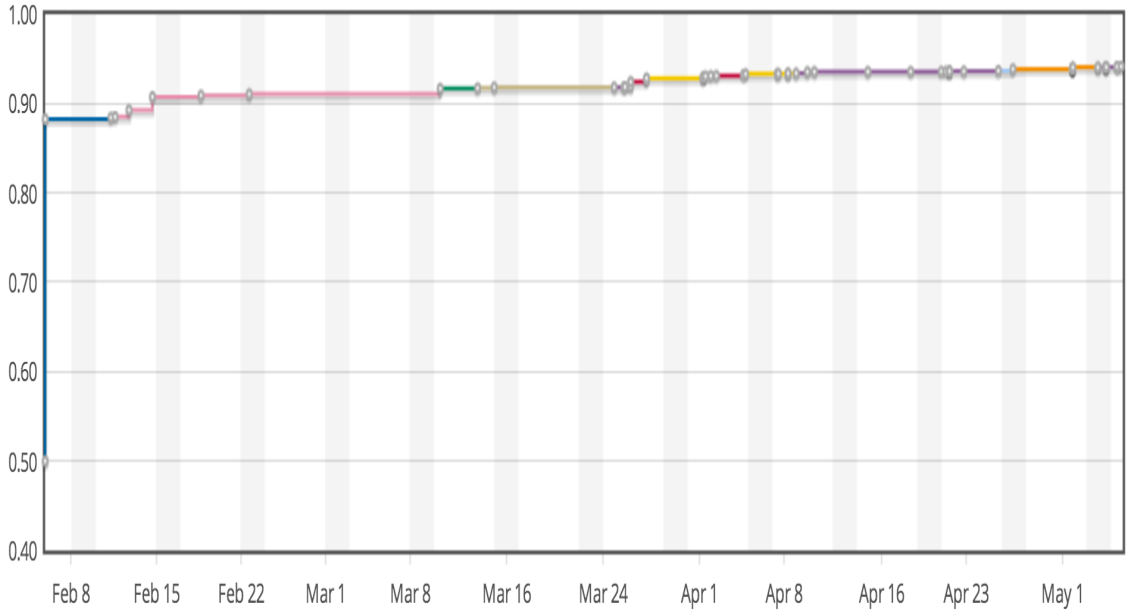


Figure 4: Performance of the top ranking participant as a function of time.

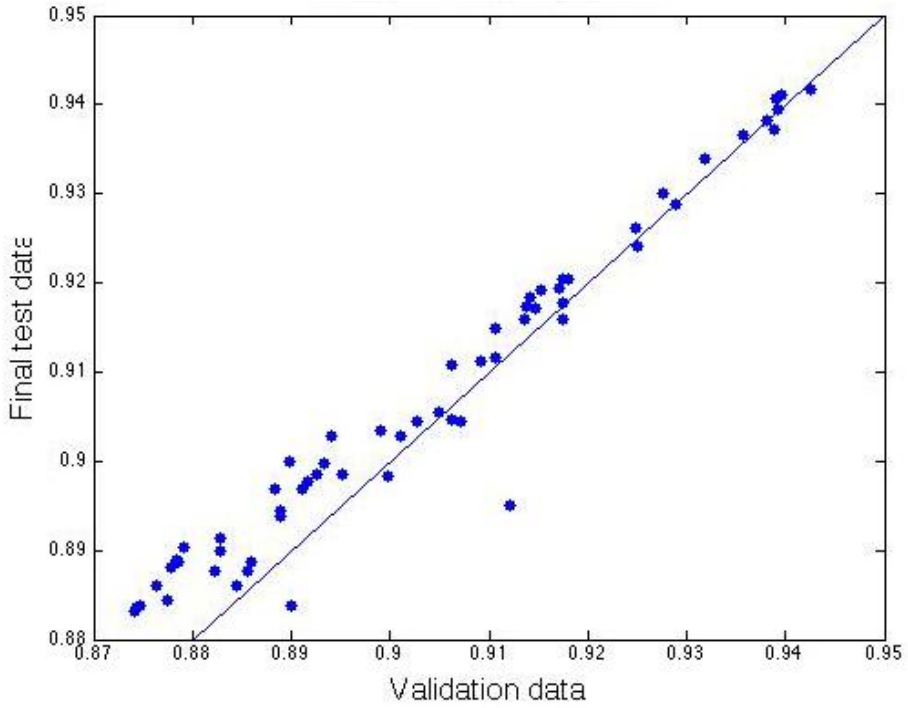


Figure 5: Scatter plot of validation vs test AUC scores.

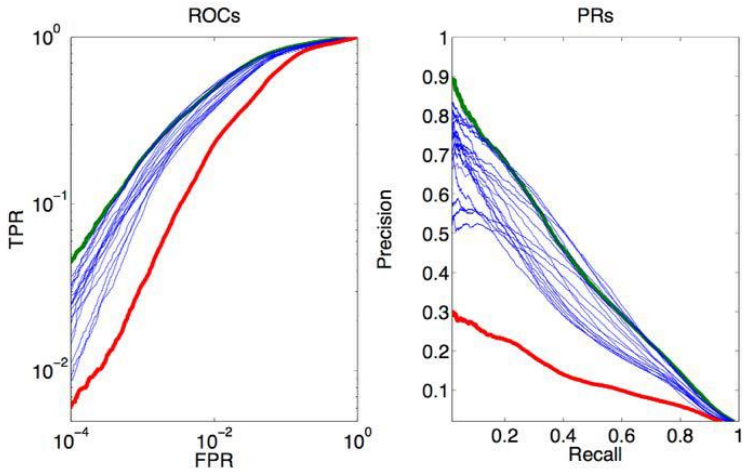


Figure 6: Performance of the Challenge winner (AAAGV team, shown in green), the rest of participants (blue), as well as the performance procured by Transfer Entropy (red).

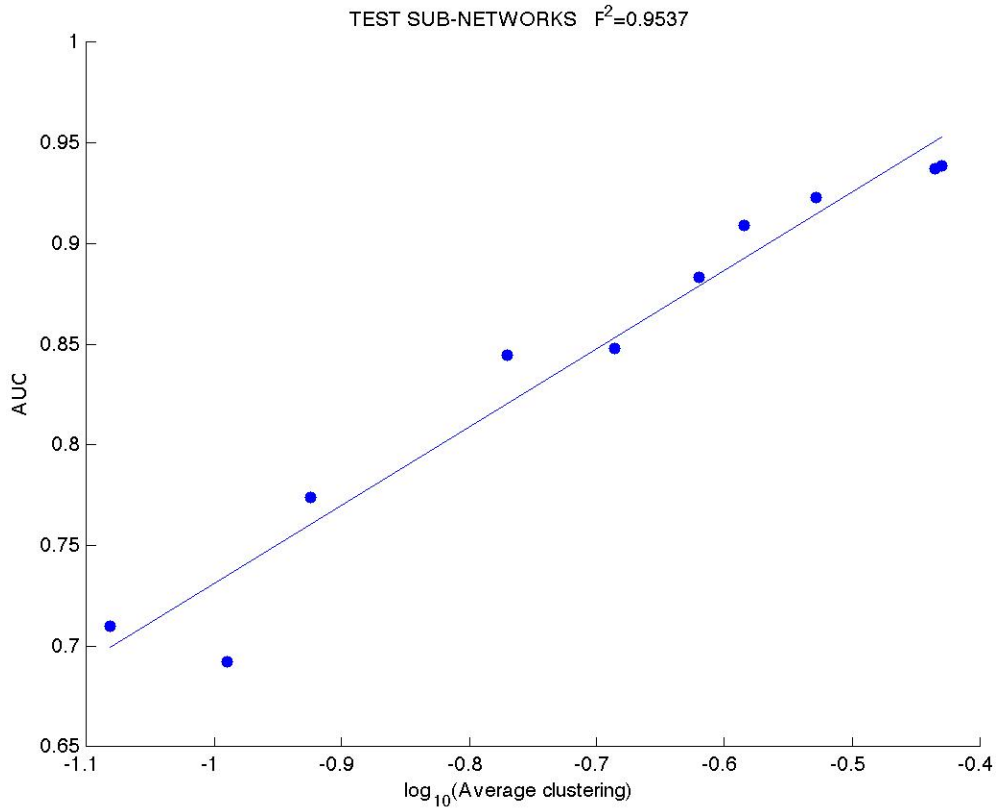


Figure 7: AUC scores of subnetworks (averaged over the top ten ranking participants) as a function of clustering coefficient.

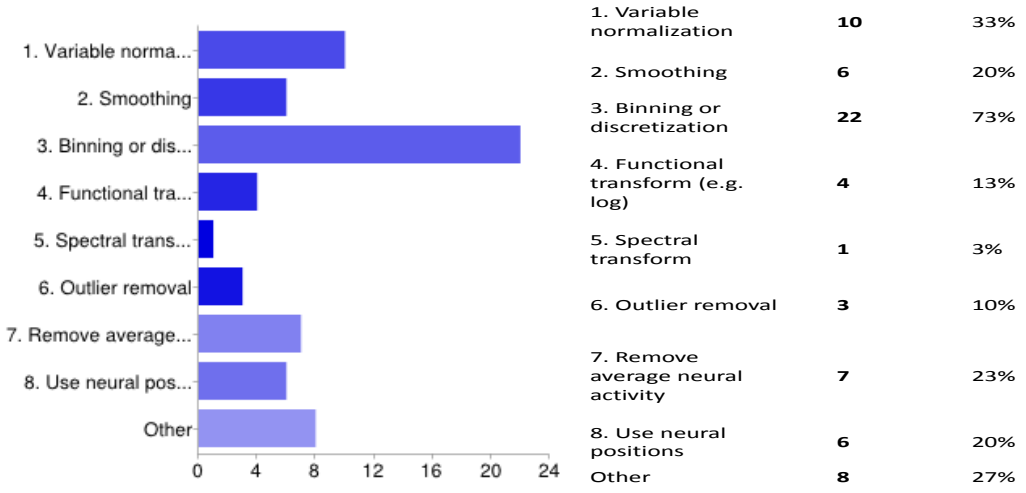
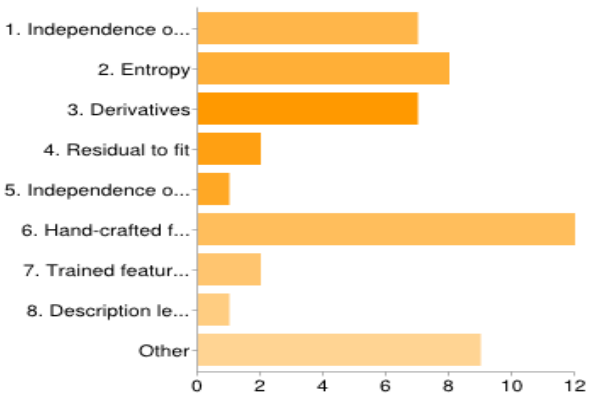
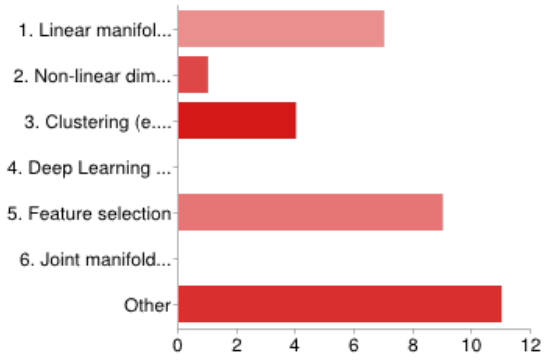


Figure 8: Preprocessing of fluorescence signals



1.Independence of variables	7	23%
2. Entropy	8	27%
3. Derivatives	7	23%
4. Residual to fit	2	7%
5. Independence of input and residual	1	3%
6. Hand-crafted features	12	40%
7. Trained feature extractors	2	7%
8. Description length or complexity of model	1	3%
Other	9	30%

Figure 9: Feature extraction



1. Linear manifold transformations (e.g. factor analysis, PCA, ICA)	7	23%
2. Non-linear dimensionality reduction (e.g. KPCA, MDS, LLE, Laplacian Eigenmaps, Kohonen maps)	1	3%
3. Clustering (e.g. K-means, hierarchical clustering)	4	13%
4. Deep Learning (e.g. stacks of auto-encoders, stacks of RBMs)	0	0%
5. Feature selection	9	30%
6. Joint manifold data fusion	0	0%
Other	11	37%

Figure 10: Dimensionality reduction

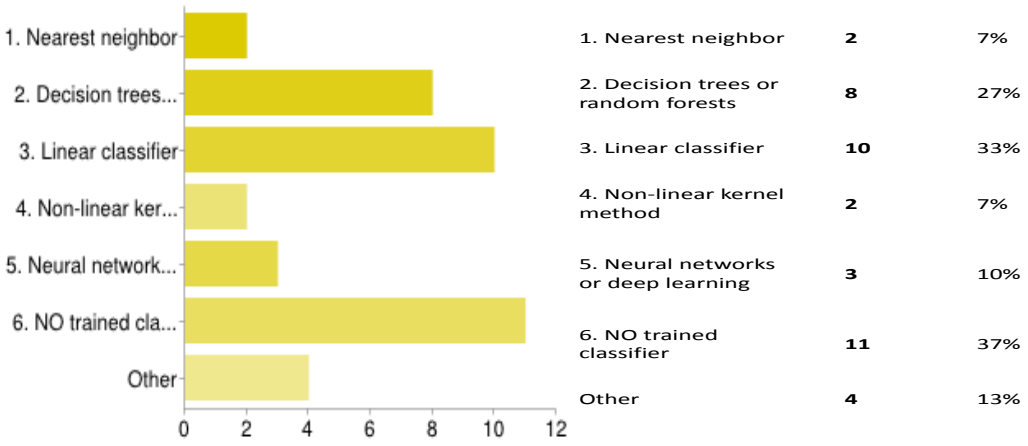


Figure 11: Classification techniques