Benchmarking of Neuromorphic Hardware Systems

Christoph Ostrau
Christian Klarhorst
Michael Thies
Ulrich Rückert
costrau@techfak.uni-bielefeld.de
Bielefeld University
Bielefeld, Germany

ABSTRACT

With more and more neuromorphic hardware systems for the acceleration of spiking neural networks available in science and industry, there is a demand for platform comparison and performance estimation of such systems. This work describes selected benchmarks implemented in a framework with exactly this target: independent black-box benchmarking and comparison of platforms suitable for the simulation/emulation of spiking neural networks.

CCS CONCEPTS

• **Hardware** → **Power and energy**; • **Computing methodologies** → **Neural networks**.

KEYWORDS

neuromorphic computing, spiking neural networks, benchmark

ACM Reference Format:
Christoph Ostrau, Christian Klarhorst, Michael Thies, and Ulrich Rückert. 2020. Benchmarking of Neuromorphic Hardware Systems. In *Neuro-inspired Computational Elements Workshop (NICE '20), March 17–20, 2020, Heidelberg, Germany*. ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3381755.3381772

There is large interest in neuromorphic computing: a growing number of proposed architectures have been fabricated and are ready to use. To name just the largest projects, there are for example the BrainScaleS [21], DYNAPs [18], Loihi [8], SpiNNaker [11] and TrueNorth [5] systems, with some successors already planned or under construction. Numerous publications have shown the applicability of these systems in different areas of computation, e.g. [2, 3, 24, 26, 28]. Furthermore, there are attempts to make many of these systems accessible via a unified API [9]. Still, a lack of spiking neural networks descriptions deployable to several of these platforms complicates direct comparisons and benchmarking. Only a few attempts have been made (see e.g. [15, 25]), and those are insufficient to compare these systems fairly, as every application benchmark reveals only certain properties of a hardware system. At the same time, the need for comparative studies grows [6, 7, 27] to measure the state of the current hardware systems, but also to quantify the progress of following hardware generations. However, it is not yet clear what "the" application for spiking neural networks will be. Thus, state-of-the-art applications of these networks are changing swiftly, and in any possible direction. For preexisting applications it is not immediately clear, whether these map well to all systems, or whether a certain architecture type is in general not suited. Finding a network or algorithm that is working on a variety of hardware accelerators and not just solutions confined to a single platform complicates this endeavour.

In this work we elaborate on our benchmarking approach for neuromorphic platforms. Previous work presented a sample application [25] and our software framework *SNABSuite* (**S**piking **N**eural **A**rchitecture **B**enchmark **Suite**) for benchmarking neuromorphic hardware [19][1]. In this approach, we take a user's perspective and use the same network description on every hardware platform (compare Figure 1). We presume that the preparation and mapping of spiking neural networks is optimized by the respective front-end software of the target platform. Due to obvious differences in the hardware, we nevertheless allow adjusted sizes of neuron populations (e.g. dependent on chip size) and neuron parameters (due to different neuron models, device mismatch or calibration issues). This is only one part of the modularity of the suite, as the

[1]https://github.com/hbp-unibi

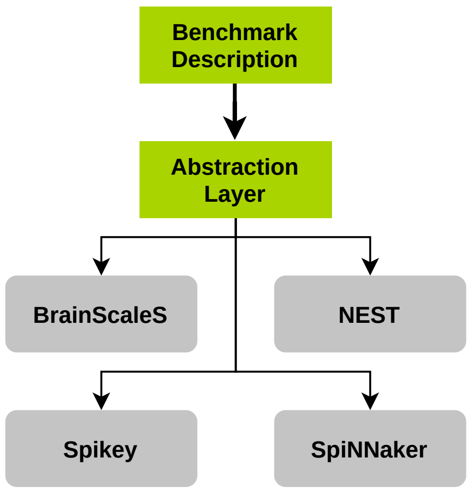

Figure 1: Benchmark work-flow presented in this paper.

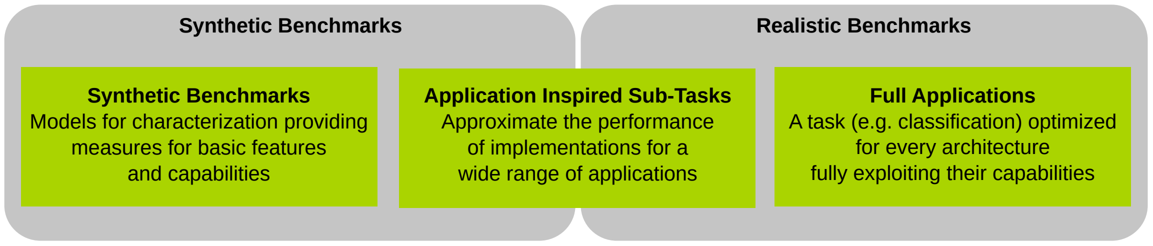
Synthetic Benchmarks
Realistic Benchmarks
Synthetic Benchmarks
Models for characterization providing measures for basic features and capabilities
Application Inspired Sub-Tasks
Approximate the performance of implementations for a wide range of applications
Full Applications
A task (e.g. classification) optimized for every architecture fully exploiting their capabilities

Figure 2: Benchmarks are separated into three categories.

support for target platforms is independent of benchmark implementations, and new benchmarks can be added without touching the platform-specific backends. This results in flexibility in several directions. First, new simulation backends can be added that adopt any existing benchmarks with minimal effort. Second, the number of implemented benchmarks is constantly growing and third, existing benchmarks can be used beyond their elementary benchmark procedure, e.g. for parameter sweeps or tests. To tackle the problem of varying applications in the field, we found it useful to have a rather broad approach to benchmarking. The scope of benchmarks ranges from low-level benchmarks, measuring application independent raw performance of hardware properties, up to application benchmarks, which have a specific task to solve. In between, there are application inspired sub-tasks, that are self-contained core algorithms of applications (compare Figure 2). Application benchmarks give very specific performance measures that are tightly coupled to the application itself, while only benchmarks from the other two categories allow to extrapolate results for multiple application domains or different spiking networks.

At the time of writing, the presented benchmark framework targets four neuromorphic simulators via the Cypress abstraction layer (compare [19, 25]). First, it supports the NEST (NEural Simulation Tool) CPU simulator [12, 14] as a baseline in regard to computational accuracy. The NEST-specific backend is implemented either using NEST via PyNN [9], or NEST's native SLI interface. NEST allows scalable simulations supporting multi-threading on local nodes, as well as MPI for simulations on distributed high-performance computing clusters [13]. Second target system is the SpiNNaker platform [11], which simulates spiking neural networks digitally on a distributed system. Its architecture is comprised of 4 to 48 SpiNNaker chips, with each of them containing 18 ARM968 general purpose processing cores. In general, the SpiNNaker system allows networks to run in real-time, with the option to slowdown the simulation for increased accuracy. The third platform is the mixed-signal Spikey [22] system. Spikey uses analog cores and emulates a parametrizable integrate-and-fire neuron model with conductance based synapses 10,000 times faster compared to biological realtime and employs a digital interconnect for the transmission of spikes. The full system consists of two chips, where each chip supports the emulation of 192 neurons with 256 synapses. As Spikey's neuron model is fixed, our benchmarks default to this model even on the more flexible target platforms. Finally, the mixed signal BrainScaleS system [21] is coupled via its high-level C++ interface. In contrast to Spikey, BrainScaleSs emphasis is on scaling the architecture to larger systems. A single HICANN chip supports 512 neuron circuits with 220 synapses each, and up to 64 circuits can be combined to form a virtual neuron, if more synapses are needed. BrainScaleS employs wafer-scale integration [23] to group 352 HICANN chips to be used by a single network.

Platform	Average freq. in ms^{-1}	Std. dev. in ms^{-1}	Max in ms^{-1}	Min in ms^{-1}
BrainScaleS	2.15	0.40	4.17	0.89
Spikey	2.80	0.13	2.86	2.50
NEST	5.00	0.00	5.00	5.00
SpiNNaker	1.00	0.00	1.00	1.00

Table 1: Results of a benchmark measuring the maximal frequency of a single neuron.

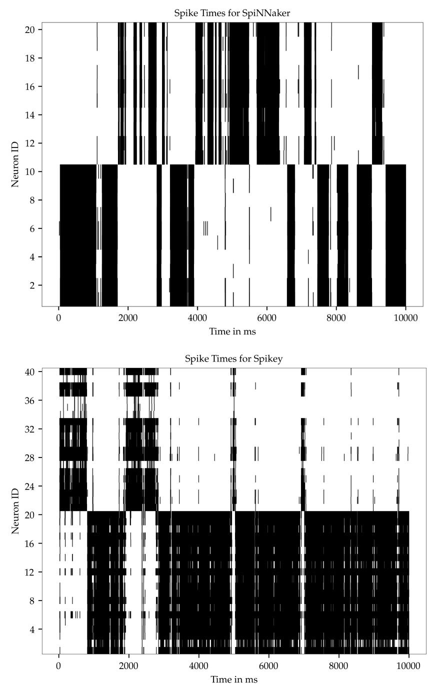

Figure 3: Spike raster plot for winner-takes-all networks on SpiNNaker and Spikey. The SNABSuite framework produces such plots on demand as part of its debug output. The WTA dynamics between two populations is clearly visible.

Platform	#Solved Sudokus	Bio-time to sol. in ms	Standard dev. in ms	Real-time to sol. in s	Power in W	Energy to sol. in J
BrainScaleS	86	3,241.9	4,573.1	$3.24 \cdot 10^{-4}$	NA	†0.0062
NEST	100	214.6	263.1	0.03	45	1.4
SpiNN-3	99	241.2	250.0	2.41	2.7	6.5
Spikey	75	3,745.8	6,041.1	$3.75 \cdot 10^{-4}$	5.6	0.0021

Table 2: Excerpt of a bunch of benchmarks solving Sudokus. Note that the actual network differs on target platforms due to hardware limitations. For more details see [20]. † calculated from values given in [24]

In the remainder of this paper we present an example of such a hierarchy of related benchmarks. Solving constraint satisfactions problems applying a winner-takes-all (WTA) architecture is a known concept [10, 16], and due to the robustness of this kind of networks it is suited to be deployed on neuromorphic hardware. As such, the WTA networks are hardware limited by e.g. read-out bandwidth and communication bandwidth, which are low-level entities and can be measured beforehand. The mere core-algorithm, the WTA network, can also be tested alone.

As a representative of the low-level benchmarks, we take a look at output rates of a single neuron. On hardware, these can be limited by buffers, bandwidth restrictions and more. Looking at the results in Table 1 it becomes clear, that on digital simulators the maximal output rate is determined by the time-step and thus by the accuracy of the simulation. On analog emulators however, the main limitation is reading out events from the analog circuitry. This maximal output rate is key to parameter tuning of the following networks.

In our specific case, the representative medium-level benchmark for the sub-task is evident. The system has to show its capability to run simple WTA networks. Here, we connect two populations of neurons with independent random poisson spike sources, and make use of cross inhibition and self-excitation. Depending on the target architecture, the cross inhibition is realized with the help of direct inhibition between populations, mirror populations or through inhibition by a single separate population. As motivated above, not all network topologies are suitable for all target platforms, which is why every variant is treated as a separate benchmark task. For all setups, both populations are activated with roughly the same probability in the optimal case. Still, one might observe that a population spikes systematically less than then the other, or even that major parts of the populations do not spike at all. This might be a statistical effect of the random input noise, but it might also occur due to bandwidth limitations, especially if full parts of a population do not spike. On both analog platforms we had the experience that, if a single neuron spikes at a rate somewhere close to the values measured in Table 1, the results cannot be trusted anymore. Careful tuning of parameters can reduce the average spike frequency and thus reduce the load on these hardware systems. Investigating the results from Figure 3 one can see that in the simulated case both competing populations are activated roughly with equal probabilities on the SpiNNaker system. However, on the analog Spikey system neuron mismatch impairs the results. There is a difference between how likely neurons get activated, and this leads to favouring of one of the populations. This implies that results with winner-takes-all based architectures on this specific analog hardware may be slightly worse than those from a simulation, at least without neuron specific parameter tuning.

Finally, Table 2 shows some selected results for solving Sudokus, as an example for constraint satisfaction problems, on all target platforms. Target benchmark criterion is the fraction of solved Sudokus for 100 of these puzzles and the average solving time. The latter is separated into two measures, the bio-time required to find a solution, referring to the neuron model internal elapsed time, and the elapsed wall-clock time to solution. This wall-clock time takes into account the backend-specific acceleration or slow-down factor, for example BrainScaleS requires only 0.1 µs for simulating 1.0 ms of the model time. The applied winner-takes-all architecture resembles the one described before. All populations have independent random noise input that represents the stochastic portion of this algorithm. Sudoku rules are then implemented through inhibitory connections, realized again either via direct inhibition or via mirror populations. All in all, we implemented three different flavours of the solver:

- Using a virtual population for every possible number in the Sudoku and direct inhibition between competing neuron populations. This leads to large overheads on e.g. the SpiNNaker system, where every population is mapped to a single core
- Using a single virtual population for the whole network and direct inhibition mitigating the aforementioned drawbacks of the SpiNNaker network mapping
- Using a single virtual population for the whole network and inhibition via mirror populations to satisfy network constraints on the Spikey system

Although the benchmark framework distinguishes between all three implementation styles, we will only present one of the results for every benchmark for the sake of simplicity. The most striking results are that the analog systems are much more time- and energy-efficient with the drawback of reduced solving capabilities (as predicted by the WTA benchmark). This is caused by the huge speed-ups of these analog systems, as they run 10,000 times faster than biological real-time [22]. Second, the SpiNNaker system is less efficient than a general purpose CPU. This is mainly due to two facts: the benchmark did not utilize the full SpiNNaker system, leaving some cores idle. Furthermore, the manufacturing process of the SpiNNaker CPU is 130 nm, while the employed CPU uses a modern and more efficient 22 nm technology. For larger Sudokus and networks this difference shrinks (not shown in Table 2), which supports the focus of the SpiNNaker architecture on large scale networks. For a comprehensive analysis of all results and implementation details see [20].

At the time of writing, the presented benchmark framework supports the four discussed simulator backends (compare Figure 1). Furthermore, support for the CPU/GPU code-generation framework GeNN [29] has been added and is currently under test. For the future we plan to extend the suite in two directions: First, we will consider the extension of the suite to support more hardware platforms, like Loihi [8] and DYNAPs [18], as well as the second generation BrainScaleS [1] and SpiNNaker [17] platforms as soon as they are available. Second, we plan to extend the range of benchmarked applications. We identified pre-trained artificial neural networks that can be converted to spiking neural networks [4], combined with an in-the-loop training for the analog target systems [24] as promising candidates for benchmarking neuromorphic hardware and are currently working on fully integrating these into our suite.

FUNDING/ACKNOWLEDGMENTS

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7) under grant agreement no 604102 and the EU's Horizon 2020 research and innovation programme under grant agreements No 720270 and 785907 (Human Brian Project, HBP). It has been further supported by the Cluster of Excellence Cognitive Interaction Technology "CITEC" (EXC 277) at Bielefeld University, which is funded by the German Research Foundation (DFG). Furthermore, we thank the Electronic Vision(s) group from Heidelberg University and Advanced Processor Technologies Research Group from Manchester University for access to their hardware systems and continuous support.

REFERENCES

[1] Sebastian Billaudelle, Yannik Stradmann, Korbinian Schreiber, Benjamin Cramer, Andreas Baumbach, Dominik Dold, Julian Göltz, Akos F Kungl, Timo C Wunderlich, Andreas Hartel, et al. 2019. Versatile emulation of spiking neural networks on an accelerated neuromorphic substrate. *arXiv preprint arXiv:1912.12980* (2019).

[2] Peter Blouw, Xuan Choo, Eric Hunsberger, and Chris Eliasmith. 2019. Benchmarking keyword spotting efficiency on neuromorphic hardware. In *Proceedings of the 7th Annual Neuro-inspired Computational Elements Workshop*. ACM, 1.

[3] Petruț Antoniu Bogdan, Andrew Graham David Rowley, Oliver Rhodes, and Steve B Furber. 2018. Structural plasticity on the SpiNNaker many-core neuromorphic system. *Frontiers in Neuroscience* 12 (2018), 434.

[4] Yongqiang Cao, Yang Chen, and Deepak Khosla. 2015. Spiking deep convolutional neural networks for energy-efficient object recognition. *International Journal of Computer Vision* 113, 1 (2015), 54–66.

[5] Andrew S. Cassidy, Paul Merolla, John V. Arthur, Steven K. Esser, Brian Jackson, Rodrigo Alvarez-Icaza, Piyali Datta, Jun Sawada, Theodore M. Wong, Vitaly Feldman, et al. 2013. Cognitive computing building block: A versatile and efficient digital neuron model for neurosynaptic cores. In *Neural Networks (IJCNN), The 2013 International Joint Conference on*. IEEE, 1–10.

[6] Benjamin Cramer, Yannik Stradmann, Johannes Schemmel, and Friedemann Zenke. 2019. The Heidelberg spiking datasets for the systematic evaluation of spiking neural networks. *arXiv preprint arXiv:1910.07407* (2019).

[7] Mike Davies. 2019. Benchmarks for progress in neuromorphic computing. *Nature Machine Intelligence* 1, 9 (2019), 386–388.

[8] Mike Davies, Narayan Srinivasa, Tsung-Han Lin, Gautham Chinya, Yongqiang Cao, Sri Harsha Choday, Georgios Dimou, Prasad Joshi, Nabil Imam, Shweta Jain, et al. 2018. Loihi: A neuromorphic manycore processor with on-chip learning. *IEEE Micro* 38, 1 (2018), 82–99.

[9] Andrew Davison, Daniel Brüderle, Jens Kremkow, Eilif Muller, Dejan Pecevski, Laurent Perrinet, and Pierre Yger. 2009. PyNN: a common interface for neuronal network simulators. *Frontiers In Neuroinformatics, 2009, 2, 11* (2009).

[10] Gabriel A. Fonseca Guerra and Steve B. Furber. 2017. Using Stochastic Spiking Neural Networks on SpiNNaker to Solve Constraint Satisfaction Problems. *Frontiers in Neuroscience* 11, December (2017). https://doi.org/10.3389/fnins.2017.00714

[11] Steve B. Furber, David R. Lester, Luis Plana, Jim D. Garside, Eustace Painkras, Sally Temple, Andrew D. Brown, et al. 2013. Overview of the SpiNNaker system architecture. *Computers, IEEE Transactions on* 62, 12 (2013), 2454–2467.

[12] Marc-Oliver Gewaltig and Markus Diesmann. 2007. NEST (NEural Simulation Tool). *Scholarpedia* 2, 4 (2007), 1430.

[13] Jakob Jordan, Tammo Ippen, Moritz Helias, Itaru Kitayama, Mitsuhisa Sato, Jun Igarashi, Markus Diesmann, and Susanne Kunkel. 2018. Extremely scalable spiking neuronal network simulation code: from laptops to exascale computers. *Frontiers in neuroinformatics* 12 (2018), 2.

[14] Jakob Jordan, Håkon Mørk, Stine Brekke Vennemo, Dennis Terhorst, Alexander Peyser, Tammo Ippen, Rajalekshmi Deepu, Jochen Martin Eppler, Alexander van Meegen, Susanne Kunkel, Ankur Sinha, Tanguy Fardet, Sandra Diaz, Abigail Morrison, Wolfram Schenck, David Dahmen, Jari Pronold, Jonas Stapmanns, Guido Trensch, Sebastian Spreizer, Jessica Mitchell, Steffen Graber, Johanna Senk, Charl Linssen, Jan Hahne, Alexey Serenko, Daniel Naoumenko, Eric Thomson, Itaru Kitayama, Sebastian Berns, and Hans Ekkehard Plesser. 2019. *NEST 2.18.0*. https://doi.org/10.5281/zenodo.2605422

[15] James C. Knight and Thomas Nowotny. 2018. GPUs Outperform Current HPC and Neuromorphic Solutions in Terms of Speed and Energy When Simulating a Highly-Connected Cortical Model. *Frontiers in Neuroscience* 12, December (2018), 1–19. https://doi.org/10.3389/fnins.2018.00941

[16] Raphaela Kreiser, Panin Pienroj, Alpha Renner, and Yulia Sandamirskaya. 2018. Pose Estimation and Map Formation with Spiking Neural Networks : towards Neuromorphic SLAM. (2018), 2159–2166.

[17] Christian Mayr, Sebastian Hoeppner, and Steve Furber. 2019. SpiNNaker 2: A 10 Million Core Processor System for Brain Simulation and Machine Learning. *arXiv preprint arXiv:1911.02385* (2019).

[18] Saber Moradi, Ning Qiao, Fabio Stefanini, and Giacomo Indiveri. 2018. A scalable multicore architecture with heterogeneous memory structures for Dynamic Neuromorphic Asynchronous Processors (DYNAPs). *IEEE transactions on biomedical circuits and systems* 12, 1 (2018), 106–122.

[19] Christoph Ostrau, Christian Klarhorst, Michael Thies, and Ulrich Rückert. 2019. Benchmarking and Characterization of event-based Neuromorphic Hardware. (2019).

[20] Christoph Ostrau, Christian Klarhorst, Michael Thies, and Ulrich Rückert. 2019. Comparing Neuromorphic Systems by Solving Sudoku Problems. In *Conference Proceedings: 2019 International Conference on High Performance Computing & Simulation*. IEEE.

[21] Mihai A. Petrovici, Bernhard Vogginger, Paul Müller, Oliver Breitwieser, Mikael Lundqvist, Lyle Muller, Matthias Ehrlich, Alain Destexhe, Anders Lansner, René Schüffny, et al. 2014. Characterization and compensation of network-level anomalies in mixed-signal neuromorphic modeling platforms. *PloS one* 9, 10 (2014), e108590.

[22] Thomas Pfeil, Andreas Grübl, Sebastian Jeltsch, Eric Müller, Paul Müller, Mihai A. Petrovici, Michael Schmuker, Daniel Brüderle, Johannes Schemmel, and Karlheinz Meier. 2013. Six networks on a universal neuromorphic computing substrate. *Frontiers in Neuroscience* 7 (2013), 11.

[23] Johannes Schemmel, Daniel Brüderle, Andreas Grübl, Matthias Hock, Karlheinz Meier, and Sebastian Millner. 2010. A wafer-scale neuromorphic hardware system for large-scale neural modeling. In *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*. 1947–1950. https://doi.org/10.1109/ISCAS.2010.5536970

[24] Sebastian Schmitt, Johann Klaehn, Guillaume Bellec, Andreas Gruebl, Maurice Guettler, Andreas Hartel, Stephan Hartmann, Dan Husmann, Kai Husmann, Vitali Karasenko, Mitja Kleider, Christoph Koke, Christian Mauch, Eric Mueller, Paul Mueller, Johannes Partzsch, Mihai A. Petrovici, Stefan Schiefer, Stefan Scholze, Bernhard Vogginger, Robert Legenstein, Wolfgang Maass, Christian Mayr, Johannes Schemmel, and Karlheinz Meier. 2017. Neuromorphic Hardware In The Loop: Training a Deep Spiking Network on the BrainScaleS Wafer-Scale System. *arXiv preprint arXiv:1703.01909* (mar 2017). arXiv:1703.01909 http://arxiv.org/abs/1703.01909

[25] Andreas Stöckel, Christoph Jenzen, Michael Thies, and Ulrich Rückert. 2017. Binary associative memories as a benchmark for spiking neuromorphic hardware. *Frontiers in computational neuroscience* 11 (2017), 71.

[26] Sacha J van Albada, Andrew G Rowley, Johanna Senk, Michael Hopkins, Maximilian Schmidt, Alan B Stokes, David R Lester, Markus Diesmann, and Steve B Furber. 2018. Performance comparison of the digital neuromorphic hardware SpiNNaker and the neural network simulation software NEST for a full-scale cortical microcircuit model. *Frontiers in neuroscience* 12 (2018).

[27] Craig M Vineyard, Sam Green, William M Severa, and Çetin Kaya Koç. 2019. Benchmarking Event-Driven Neuromorphic Architectures. In *Proceedings of the International Conference on Neuromorphic Systems*. 1–5.

[28] Timo Wunderlich, Akos F Kungl, Andreas Hartel, Yannik Stradmann, Syed Ahmed Aamir, Andreas Grübl, Arthur Heimbrecht, Korbinian Schreiber, David Stöckel, Christian Pehle, et al. 2018. Demonstrating Advantages of Neuromorphic Computation: A Pilot Study. *arXiv preprint arXiv:1811.03618* (2018).

[29] Esin Yavuz, James Turner, and Thomas Nowotny. 2016. GeNN: a code generation framework for accelerated brain simulations. *Scientific reports* 6 (2016).