A Population-Level Approach to Temperature Robustness in Neuromorphic Systems

Eric Kauderer-Abrams*, Andrew Gilbert*, Aaron Voelker†, Ben Benjamin*, Terrence C. Stewart†, Kwabena Boahen*
{ekabrams@stanford.edu, boahen@stanford.edu}
*Electrical Engineering and Bioengineering, Stanford University, Stanford, CA, U.S.A
†Centre for Theoretical Neuroscience, University of Waterloo, Waterloo, ON, Canada.

***Abstract*—We present a novel approach to achieving temperature-robust behavior in neuromorphic systems that operates at the population level, trading an increase in silicon-neuron count for robustness across temperature. Our silicon neurons' tuning curves were highly sensitive to temperature, which could be decoded from a 400-neuron population with a precision of 0.07°C. We overcame this temperature-sensitivity by combining methods from robust optimization theory with the Neural Engineering Framework. We developed two algorithms and compared their temperature-robustness across a range of 2°C by decoding one period of a sinusoid-like function from populations with 25 to 800 neurons. We find that 560 neurons are required to achieve the same precision across this temperature range as 35 neurons achieved at a single temperature.**

I. Temperature's Effect On Biological and Neuromorphic Systems

Temperature variation presents a major challenge for both biological and silicon systems. The computational primitives in both classes of systems—ion channels and transistors, respectively—are inherently temperature sensitive, so achieving robustness to temperature variation is nontrivial. Nevertheless, neurons in cold-blooded animals, such as crustaceans, manage to produce behavior that is robust over 20°C. They do so by regulating the concentrations of various species of ion channels, each with a different temperature-sensitivity. Temperature-robustness is achieved at the cost of adding more ion-channel species than would otherwise be necessary to produce the same behavior at a single temperature [1].

Taking inspiration from the crustacean's solution, we present a novel approach to temperature-robustness in neuromorphic systems that trades an increase in neuron count for a decrease in temperature-variation. The Neural Engineering Framework (NEF) [2], a common method for mapping computations onto networks of spiking neurons, does not produce temperature-robust computations (Fig. 1), because it assumes that the neurons' tuning curves do not vary with temperature (Fig. 2). In this paper, we review the NEF (Section II), extend it using robust optimization techniques (Section III), describe the silicon-neuron circuits whose tuning curves we measured (Section IV), and demonstrate that our extensions produce computations that are robust to these tuning curves' temperature variation (Section V).

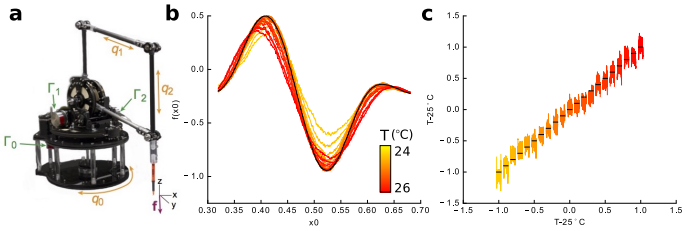

Fig. 1: Computing with spiking silicon-neurons. **a**: To control a robotic arm, for example, complicated functions of the joint angles must be computed. **b**: When mapped onto silicon-neurons using the NEF, this computation is not robust to temperature because the neurons' tuning curves are sensitive to temperature. **c**: The chip's temperature (black horizontal line segments) can be decoded (colored vertical line segments) with 0.07°C precision from a population of 400 neurons.

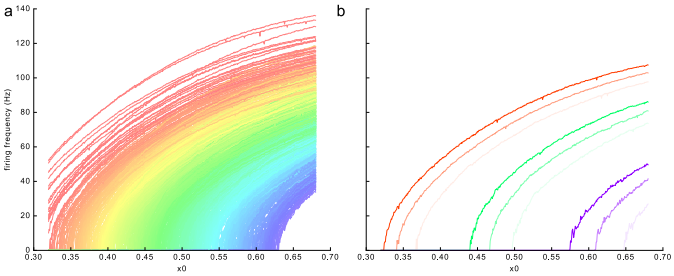

Fig. 2: Tuning curves' temperature-variation **a**: Tuning curves of 1000 neurons at 25°C. Differences arise from transistor mismatch, which shifts and scales each curve differently. **b**: Tuning curves of the 5th, 50th and 95th-percentile neuron—ranked by the input level at which spiking begins—at 24, 25, and 26°C. As temperature increases, the curves shift leftward.

II. The Neural Engineering Framework

The Neural Engineering Framework (NEF) [2] provides a framework for encoding continuous signals in populations of spiking neurons and decoding transformations of those signals from these populations. This framework enables arbitrary static or dynamic transformations to be computed by networks of spiking neurons. For this work, we focus on computing static transformations.

An input signal $\mathbf{x} \in \mathbb{R}^k$ is encoded into a collection of spike-rates $\{a_j(\mathbf{x})\}_{j=1}^N$ of a population of N neurons. Each

neuron's spike-rate is given by its tuning curve:

$$a_j(\mathbf{x}) = G(\alpha_j(\mathbf{e}_j \cdot \mathbf{x}) + \beta_j) \tag{1}$$

where G is the neuronal transfer function, which maps input currents to steady-state spike-rates, α_j is a gain factor, $\mathbf{e}_j$ is a unit-length *encoding vector* and β_j is a bias. All three are randomly distributed across the population. The transformation $\hat{f}(\mathbf{x})$ computed by the population is obtained by taking a linear combination of its tuning curves

$$\hat{f}(\mathbf{x}) = \sum_{j=1}^{N} d_j a_j(\mathbf{x}) \tag{2}$$

where d_j is the *decoding weight* assigned to neuron j.

To solve for the optimal decoding weights that implement a transformation $f : \mathbb{R}^k \to \mathbb{R}$ of the input signal $\mathbf{x}$ encoded by the population of N neurons, we sample $\mathbf{x}$ at equally spaced points $\mathbf{x}_1, ..., \mathbf{x}_Q$ and define the vector $\mathbf{f}(\mathbf{x}) \in \mathbb{R}^{\mathrm{Q}}$ with elements $\mathbf{f}(\mathbf{x})_i = f(\mathbf{x}_i)$. Similarly, we define each neuron's tuning curve over the discretized domain as the vector $\mathbf{a}_j(\mathbf{x}) \in \mathbb{R}^{\mathrm{Q}}$ with elements $(\mathbf{a}_j(\mathbf{x}))_i = \mathbf{a}_j(\mathbf{x}_i)$—the spike-rate of neuron j in response to the input $\mathbf{x}_i$. We define the vector of decoding weights $\mathbf{d} \in \mathbb{R}^{\mathrm{N}}$ and the matrix $\mathbf{A} \in \mathbb{R}^{\mathrm{Q}\times\mathrm{N}}$ whose columns are the tuning curves $\mathbf{a}_j(\mathbf{x})$. The optimal decoding weights $\mathbf{d}_*$ are computed by minimizing $\|\hat{\mathbf{f}}(\mathbf{x}) - \mathbf{f}(\mathbf{x})\|^2 = \|\mathbf{A}\mathbf{d} - \mathbf{f}(\mathbf{x})\|^2$. This is the traditional least-squares problem, with solution given by

$$\mathbf{d}_* = \sum_{i=1}^{N} \frac{\mathbf{u}_i^\intercal \mathbf{f}(\mathbf{x})}{s_i} \mathbf{v}_i \tag{3}$$

where $\mathbf{u}_i$ and $\mathbf{v}_i$ are the i^{th} left and right singular vectors of $\mathbf{A}$ and s_i is the corresponding singular value.

III. Temperature-Robust Decoding

Performing a singular-value decomposition (SVD) on $\tilde{\mathbf{A}} \in \mathbb{R}^{\mathrm{RQ}\times\mathrm{N}}$—a stack of tuning-curve matrices $\{\mathbf{A}_{T_k}\}_{k=1}^{R}$ measured at R different temperatures—yields insight into the nature of the neuronal population's temperature variation. Each of $\tilde{\mathbf{A}}$'s left singular vectors, $\tilde{\mathbf{u}}_i$, is naturally segmented into R vectors, $\mathbf{u}_{i,T_k}$, each associated with a particular temperature (Fig. 3). Evidently, some modes vary with temperature more than others. To quantify each mode's temperature robustness, we define the measure

$$r_i = \sum_{k=1}^{R} \mathbf{u}_{i,T_{k+1}}^\intercal \mathbf{u}_{i,T_k} \tag{4}$$

where $\mathbf{u}_{i,T_{R+1}} := \mathbf{u}_{i,T_1}$. Note that $-1 \le r_i \le 1$ because the $\tilde{\mathbf{u}}_i$ are unit-vectors. The decrease in r_i with increasing i indicates that there is a trade-off between accuracy and temperature-robustness (Fig. 4). Decoding becomes more accurate as higher-order modes are included, however, these higher-order modes are the least-temperature robust.

With the trade-off between decoding accuracy and temperature-robustness in mind, we design two algorithms that balance these competing objectives. The first, a simple

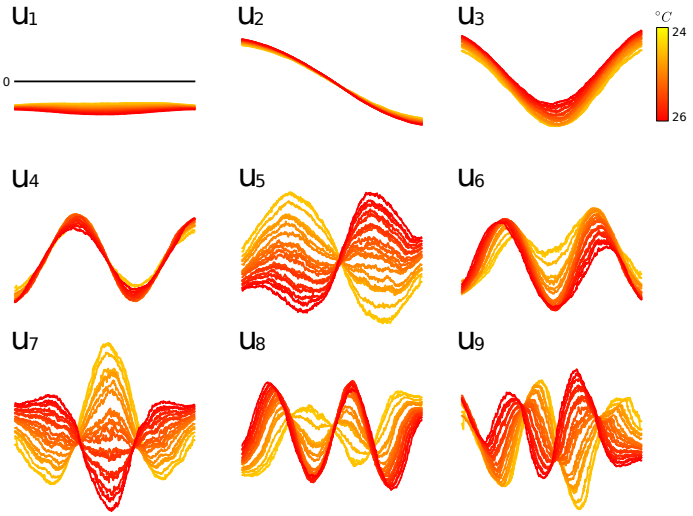

Fig. 3: The first nine left singular vectors $\mathbf{u}_{1-9}$ of 21 stacked matrices with 400 tuning-curves each. These matrices were measured at temperatures equally spaced between 24°C and 26°C. Each panel shows the corresponding 21 segments of a singular vector, color-coded by temperature.

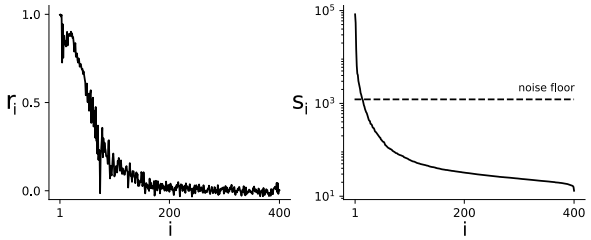

Fig. 4: Temperature-robustness (r_i) and singular values (s_i) for all 400 left singular vectors. The noise floor was computed by measuring the largest singular value of the difference between two different samples of the tuning curve matrix.

extension of least-squares, minimizes a weighted average of decoding error and temperature variation:

$$\underset{\mathbf{d}}{\text{minimize}} \quad \sum_{k=1}^{R} \|\mathbf{A}_{T_k}\mathbf{d} - \mathbf{f}(\mathbf{x})\|^2 + \frac{\kappa}{2}\|\mathbf{A}_{T_{k+1}}\mathbf{d} - \mathbf{A}_{T_k}\mathbf{d}\|^2 \tag{5}$$

where $\mathbf{A}_{T_{R+1}} = \mathbf{A}_{T_1}$ and κ is the relative weighting. The solution is given by

$$\mathbf{d}_* = \sum_{i=1}^{N} \frac{1}{1 + \kappa\,(1 - r_i)} \frac{\sum_{k=1}^{R} \mathbf{u}_{i,T_k}^\intercal \mathbf{f}(\mathbf{x})}{s_i} \mathbf{v}_i \tag{6}$$

which is equivalent to replacing $\mathbf{u}_i$ in (3) with the sum of $\mathbf{u}_{i,T_k}$ across temperature, except that each mode's contribution increases with its temperature-robustness, r_i. We call this algorithm MinChange.

The second algorithm, MinMax, minimizes a weighted sum of the worst-case decoding-error across temperature and the

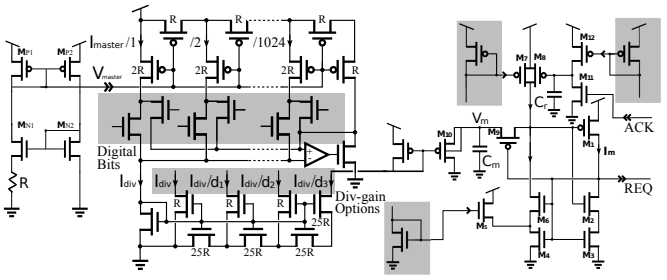

Fig. 5: Bias-generator, DAC, gain-stage, and soma circuit. The bias generator ($M_{\text{P1-2}}, M_{\text{N1-2}}$) produces a current proportional to $T^{1.5}$ (inversely proportional to mobility), where T is absolute temperature. The DAC subdivides this current with 10-bit resolution. The gain-stage mirrors or scales the DAC output by one of three options ($\text{d}_{0\text{-}3}$). The soma circuit ($M_{1\text{-}12}$) is a low-pass filter ($M_{1,7,10}$) with positive feedback ($M_{2\text{-}6}$), which together with the input can overpower the leak to generate a spike, pulling REQ up to V_{dd}. Following a spike, the refractory circuit is activated when ACK pulses from V_{dd} to ground, enforcing a brief quiescent period.

average temperature-variation as follows

$$\underset{\mathbf{d}}{\text{minimize}} \ \max_{T_k}\left(\|\mathbf{A}_{T_k}\mathbf{d}-\mathbf{f}(\mathbf{x})\|^2\right)+\frac{\kappa}{2R}\sum_{k=1}^{R}\|\mathbf{A}_{T_{k+1}}\mathbf{d}-\mathbf{A}_{T_k}\mathbf{d}\|^2 \tag{7}$$

This problem does not admit an analytical solution, but can be solved rapidly after converting it to the Second-Order Cone Program format [3].

IV. Circuit Description

To test our temperature-robust algorithms, we collected data from *Neurogrid* [4], a 16-chip neuromorphic system with one-million spiking soma circuits. The biases for these circuits are generated by a bias-generator, a Digital-to-Analog Converter (DAC), and a programmable gain-stage [5], all of which contribute to the temperature-dependence of the soma circuits' tuning curves (Fig. 5). Note that the bias-generator's primary devices (M_{N1} and M_{N2}) operate above-threshold, because of the large device sizes required for subthreshold operation.

The soma circuit implements the Quadratic Integrate-and-Fire (QIF) neuron model. Its dynamics are given by

$$\tau\frac{dv}{dt}=\frac{1}{2}v^2-v+x_0 \tag{8}$$

where τ is the membrane time-constant and x_0 is the input current. For this work, we use a previous calibration [6] to program the chip directly in terms of these dimensionless model parameters instead of raw current biases.

V. Validation With Neuromorphic-Chip Data

To test the temperature-robust decoding algorithms, we measured tuning curves from 1000 of Neurogrid's spiking soma circuits. We swept x_0 over 500 equally-spaced values from 0.32 to 0.68, a range within which 80% of the neurons go from quiescence to spiking. Neurons that did not spike in this range were not included in decoding. All the neurons' spike-trains were measured simultaneously using the chip's high-throughput spike-routing interface [7]. These measurements

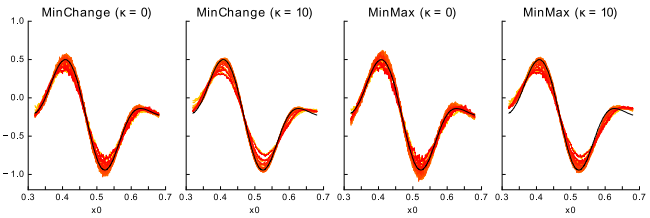

Fig. 6: The temperature-robust algorithms produce fairly accurate computations across a 2°C range. Decodes are computed using 400 neurons at 21 different temperature values.

were repeated across 21 temperatures—equally spaced from 24 to 26°C—by regulating the chip's temperature using a *TE Technology* TC-36-25-RS232 proportional-integral-derivative temperature controller.

To evaluate the algorithms' performance we began by setting half of the encoders e_j to $+1$ and half to -1 (flipping half of the tuning curves). We computed a static transformation $f(x_0)$, synthesized from the $\mathbf{u}_{i,T_k}$ at $T_k = 25°\text{C}$. $f(x_0)$ has 75% of its power in the first 4 modes, which are temperature-robust and above the noise floor, 24% of its power in modes 5 through 10, which are not temperature-robust but are above the noise floor, and the remaining 1% of its power in the higher modes, which are neither temperature-robust nor above the noise floor. We defined the noise floor to be the largest singular value of the difference between two different samples of the tuning curve matrix, a value below which modes can vary substantially across different samples of the data. To solve for the decoding weights, for MinChange we used the pseudo-inverse method and for MinMax we used the CVXPY package [8]. In all cases, we added a regularization term, $\lambda\|\mathbf{d}\|^2$ to the objective function.

With respect to decoding accuracy, both algorithms had roughly similar performance (Fig. 6). Upon closer inspection, MinChange achieved a lower error than MinMax (Fig. 7). However, MinMax achieved a lower error at the temperature extremes. For both algorithms, increasing κ trades decoding error for temperature robustness, as expected.

Next, we examined the cost in number of neurons of decoding accurately across temperature (Fig. 8). The average decoding error across temperature decreased as $N^{-0.22}$ and $N^{-0.39}$ for MinChange with $\kappa = 0, 10$, and as $N^{-0.37}$ and $N^{-0.47}$ for MinMax with $\kappa = 0, 10$. An approximately 16-fold increase in neuron-count—for example from 35 to 560—is required for the robust algorithms to match the decode accuracy of least-squares trained and tested at a single temperature.

To gain insight into how MinChange trades decode-accuracy for temperature-robustness, we solved for decoders at different values of κ. For each decoder, we computed the fraction of cumulative power contained in the first i modes

$$p(i;\mathbf{d}_\kappa)=\frac{\sum_{j=1}^{i}s_j^2(\mathbf{v}_j^\intercal\mathbf{d}_\kappa)^2}{\sum_{j=1}^{N}s_j^2(\mathbf{v}_j^\intercal\mathbf{d}_\kappa)^2} \tag{9}$$

Since the lower modes are more temperature-robust, Min-

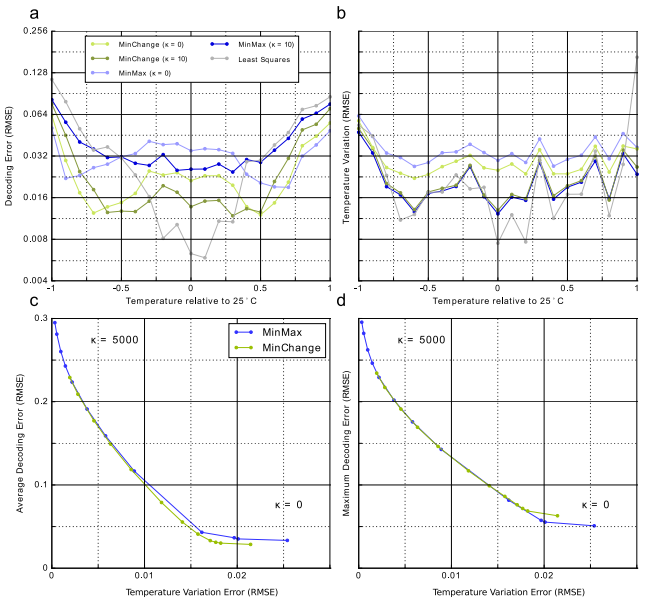

Fig. 7: Tradeoff between decoding accuracy and temperature-robustness. **a**: Decoding error across temperature. **b**: temperature-variation error across temperature. **c**-**d**: The trade-off between average or maximum decoding error and temperature variation error achieved by MinMax and MinChange as κ varies between 0 and 5000.

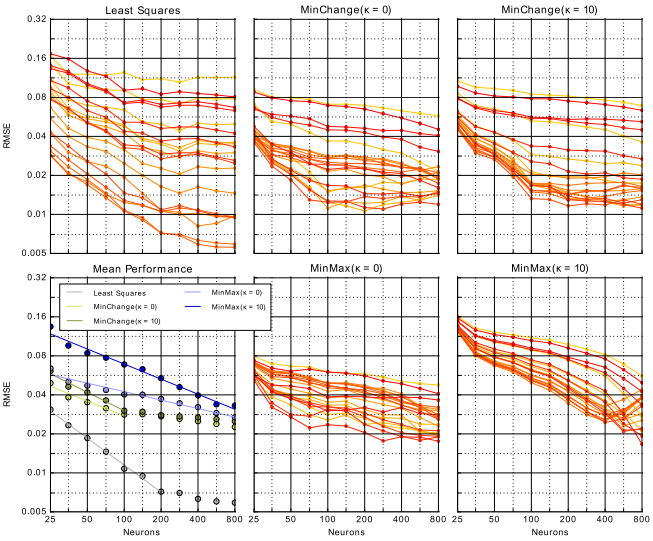

Fig. 8: The temperature-robust algorithms' decoding error as a function of the number of neurons used across 21 different temperatures from $24-26^{\circ}$C (log-log scale). The Least-Squares line shows the performance of least-squares trained and tested at 25°C.

Change assigns them increasingly more power as κ increases (Fig. 9).

VI. Summary

In this paper, we presented a novel solution for computing with neuromorphic chips in a temperature-robust way. Crucially, our solution required neither modifications to the underlying circuits nor energetically-expensive temperature-regulators. Instead, we solved for temperature-robust decoders using algorithms from robust optimization theory.

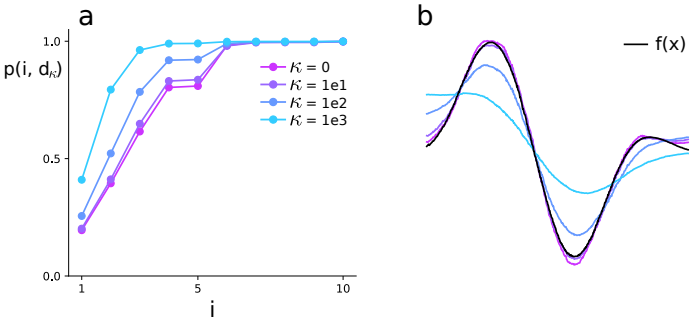

Fig. 9: Temperature-robustness entails shifting the decoder's power to lower modes at the expense of accuracy. **a**: The fraction of the cumulative power contained in the first i singular vectors for decoders obtained with MinChange at different values of κ. **b**: Corresponding functions decoded along with the target function $f(x)$.

We presented two temperature-robust algorithms that are convex and can be solved rapidly with standard software. To validate our solutions, we tested them on tuning curves measured from a neuromorphic chip at many different temperatures. From these results, we demonstrated that the cost of this temperature-robustness is an increase in the number of silicon neurons needed to decode a transformation at a given accuracy level.

VII. Acknowledgements

This work was supported by ONR grants N000141310419 and N000141512827.

References

[1] T. OLeary and E. Marder, "Temperature-robust neural function from activity-dependent ion channel regulation," *Current Biology*, vol. 26, no. 21, pp. 2935–2941, 2016.

[2] C. Eliasmith and C. H. Anderson, *Neural engineering: Computation, representation, and dynamics in neurobiological systems*. MIT press, 2003.

[3] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[4] B. V. Benjamin, P. Gao, E. McQuinn, S. Choudhary, A. R. Chandrasekaran, J.-M. Bussat, R. Alvarez-Icaza, J. V. Arthur, P. A. Merolla, and K. Boahen, "Neurogrid: A mixed-analog-digital multichip system for large-scale neural simulations," *Proceedings of the IEEE*, vol. 102, no. 5, pp. 699–716, 2014.

[5] T. Delbrück and A. Van Schaik, "Bias current generators with wide dynamic range," *Analog Integrated Circuits and Signal Processing*, vol. 43, no. 3, pp. 247–268, 2005.

[6] P. Gao, B. V. Benjamin, and K. Boahen, "Dynamical system guided mapping of quantitative neuronal models onto neuromorphic hardware," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 59, no. 10, pp. 2383–2394, 2012.

[7] K. A. Boahen, "A burst-mode word-serial address-event link-i: Transmitter design," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 51, no. 7, pp. 1269–1280, 2004.

[8] S. Diamond and S. Boyd, "Cvxpy: A python-embedded modeling language for convex optimization," *Journal of Machine Learning Research*, vol. 17, no. 83, pp. 1–5, 2016.