SYNTHESIZING REALISTIC NEURAL POPULATION ACTIVITY PATTERNS USING SEMI-CONVOLUTIONAL GANS

Anonymous authors
Paper under double-blind review

ABSTRACT

The ability to synthesize realistic patterns of neural activity is crucial for studying neural information processing. Here we used the Generative Adversarial Networks (GANs) framework to simulate the concerted activity of a population of neurons. We embedded a semi-convolutional network architecture in a Wasserstein-GAN to facilitate the generation of unconstrained neural population activity patterns while still benefiting from shift invariance in the temporal domain. We demonstrate that our proposed GAN, which we termed Spike-GAN, generates spike trains that match accurately the first- and second-order statistics of datasets of tens of neurons and also approximates well their higher-order statistics. We show that, when applied to a real dataset recorded from salamander retina, Spike-GAN produces population spike trains that match the second-order statistics as well as state-of-the-art maximum entropy models. In addition, Spike-GAN faithfully reproduces the temporal dynamics of the retinal population's activity, an aspect that is typically ignored by maximum entropy models and most existing methods. Finally, we show how to exploit a trained Spike-GAN to construct 'importance maps' to detect the most relevant statistical structures present in a spike train. Spike-GAN provides a powerful, easy-to-use technique for generating realistic spiking neural activity and for describing the most relevant features of the large-scale neural population recordings studied in modern systems neuroscience.

1 INTRODUCTION

Understanding how to generate synthetic spike trains simulating the activity of a population of neurons is crucial for systems neuroscience. In computational neuroscience, important uses of faithfully generated spike trains include creating biologically consistent inputs needed for the simulation of realistic neural networks, generating large and faithful data samples to be used for the development and validation of new advanced spike train analysis techniques, and estimating the response probabilities of large scale neural responses to stimuli needed to extrapolate the information coding capacity of neurons beyond what can be computed from the neural data obtained experimentally (Ince et al., 2013; Moreno-Bote et al., 2014). In experimental systems neuroscience, the ability to develop models that produce realistic neural population patterns and that identify the key sets of features in these patterns is fundamental to disentangling the encoding strategies used by neurons for sensation or behavior (Panzeri et al., 2017) and to design closed-loop experiments (Kim et al., 2017) in which synthetic patterns, representing salient features of neural information, are fed to systems of electrical micro-stimulation (Tehovnik et al., 2006) or patterned light optogenetics (Panzeri et al., 2017; Bovetti & Fellin, 2015) for naturalistic intervention on neural circuits.

One successful way to generate realistic spike trains is that of using a bottom-up approach, focusing explicitly on replicating selected low-level aspects of spike trains statistics. Popular methods include renewal processes (Stein (1965); Gerstner & Kistler (2002)), latent variable models (Macke et al., 2009) and maximum entropy approaches (Tang et al., 2008; Schneidman et al., 2006; Savin & Tkačik, 2017), which typically model the spiking activity under the assumption that only first and second-order correlations play a relevant role in neural coding (but see Cayco-Gajic et al. (2015); Köster et al. (2014); Ohiorhenuan et al. (2010)). Other methods model spike train responses assuming linear stimulus selectivity and generating single trial spike trains using simple models of

input-output neural nonlinearities and neural noise (Keat et al., 2001; Pillow et al., 2008; Lawhern et al., 2010). These methods have had a considerable success in modeling the activity of populations of neurons in response to sensory stimuli (Pillow et al., 2008). Nevertheless, these models are not completely general and may fail to faithfully represent spike trains in many situations. This is because neural variability changes wildly across different cortical areas (Maimon & Assad, 2006) due to the fact that responses, especially in higher-order areas and in behaving animals, have complex non-linear tuning to many parameters and are affected by many behavioral variables (e.g. the level of attention (Fries et al., 2001)).

An alternative approach is to apply deep-learning methods to model neural activity in response to a given set of stimuli using supervised learning techniques (McIntosh et al., 2016). The potential advantage of this type of approach is that it does not require to explicitly specify any aspect of the spike train statistics. However, applications of deep networks to generate faithful spike patterns have been rare. Here, we explore the applicability of the Generative Adversarial Networks (GANs) framework (Goodfellow et al., 2014) and extend it to synthesize realistic neural activity. In particular, we adapt the recently proposed Wasserstein-GAN (WGAN) (Arjovsky et al., 2017), which has been proven to stabilize training, and use a semi-convolutional network architecture to model invariance in the temporal dimension while keeping dense connectivity across the modeled neurons.

Three aspects of GANs (and in particular WGANs) make this technique a good candidate to model neural activity. First, GANs are an unsupervised learning technique and therefore do not need labeled data (although they can make use of labels (Odena et al., 2016; Chen et al., 2016)). This largely increases the amount of neural data available to train them. Second, WGANs are good at fitting distributions presenting multiple modes (Arjovsky et al., 2017; Gulrajani et al., 2017). This is an aspect that is crucial for neural data because the presentation of even a single stimulus can elicit very different spatio-temporal patterns of population activity (Churchland et al., 2007; Morcos & Harvey, 2016). We thus need a method that generates sharp realistic samples instead of producing samples that are a compromise between two modes (which is typical, for instance, of methods seeking to minimize the mean squared error between the desired output and the model's prediction (Goodfellow, 2016; Lotter et al.)). Finally, using as their main building block deep neural networks, GANs inherit the capacity of scaling up to large amounts of data and therefore constitute a good candidate to model the ever growing datasets provided by experimental methods like chronic multi-electrode and optical recording techniques.

We show that the proposed GAN, which we called Spike-GAN, is able to produce highly realistic spike trains matching the first and second-order statistics of a population of neurons. We further demonstrate the applicability of Spike-GAN by applying it to a real dataset recorded from salamander retina (Marre et al., 2014) and comparing the activity patterns the model generates to those obtained with a maximum entropy model (Tkačik et al., 2014). Finally, we describe a new procedure to detect in a given activity pattern those spikes participating in a specific feature characteristic of the probability distribution underlying the training dataset.

2 METHODS

2.1 NETWORK ARCHITECTURE

We adapted the Generative Adversarial Networks described by Goodfellow et al. (2014); Goodfellow (2016) to produce samples that simulate the spiking activity of a population of N neurons as binary vectors of length T (spike trains, Supp. Fig. Fig. S2). In their original form, GANs proved to be difficult to train, prompting several subsequent studies that focused on making them more stable (Radford et al., 2015; Chintala et al., 2016). In the present work we used the Wasserstein-GAN variant described by Arjovsky et al. (2017). Wasserstein-GANs (WGAN) minimize the Earth-Mover (or Wasserstein-1) distance (EM) between the original distribution $\mathrm{P}_{\mathrm{data}}$ and the distribution defined by the generator, P_G. Arjovsky et al. (2017) showed that the EM distance has desirable properties in terms of continuity and differentiability that ensure that the loss function provides a meaningful gradient at all stages of training, which boosts considerably the stability of the training. A further improvement was later introduced by Gulrajani et al. (2017), who provided an alternative procedure to ensure that the critic is Lipschitz (via gradient penalization), which is required in the WGAN framework.

Here we adapted the WGAN-GP to simulate realistic neural population activity patterns. Our samples are matrices of size $N \times T$, where N is the number of neurons and T the number of time bins, each bin usually corresponding to a few milliseconds (Fig. S2). Importantly, while samples present a high degree of invariance along the time dimension, they are usually not spatially structured (i.e. across neurons) and thus we cannot expect any invariance along the dimension spanning the different neurons. For this reason, in order to take advantage of the temporal invariance while being maximally agnostic about the neural correlation structure underlying the population activity, we modified a standard 1D-DCGAN architecture (Radford et al., 2015) by transposing the samples so as to make the spatial dimension correspond to the channel dimension (Fig. 1). Therefore, we are *de facto* performing a semi-convolutional GAN, in which the spatial dimension is densely connected while weights are shared across the temporal dimension thus improving training, efficiency and the interpretability of the trained networks.

The main modifications we have introduced to the WGAN-GP (the convolutional variant) are:

1. The responses of different neurons are fed into different channels.
2. Following Chintala et al. (2016) we made all units LeakyReLU (the slope of the leak was set to 0.2) except for the last layer of the generator where we used sigmoid units.
3. The critic consists of two 1D convolutional layers with 256 and 512 features, respectively, followed by a linear layer. The generator samples from a 128-dimension uniform distribution and its architecture is the mirror image of that of the critic.
4. To avoid the checkerboard issue described by Odena et al. (2016), we divided all generator's fractional-strided convolutions (i.e. deconvolutions) into two separate steps: upsampling and convolving. The upsampling step is done using a nearest neighbor procedure, as suggested by Odena et al. (2016).

We called the network described above Spike-GAN. As in Arjovsky et al. (2017), Spike-GAN was trained with mini-batch stochastic gradient descent (we used a mini-batch size of 64). All weights were initialized from a zero-centered normal distribution with standard deviation 0.02. We used the Adam optimizer (Kingma & Ba, 2014) with learning rate = 0.0001 and hyperparameters $\beta_1 = 0$ and $\beta_2 = 0.9$. The parameter λ, used for gradient penalization, was set to 10. The critic was updated 5 times for each generator update.

2.2 Spike train analysis

To compare the statistics of the generated samples to the ones contained in the ground truth dataset, we first discretized the continuously-valued samples produced by the generator and then, for each bin with activation h, we drew the final value from a Bernoulli distribution with probability h. Note that the last layer of the generator contains a sigmoid function and thus the h values can be interpreted as probabilities.

We assessed the performance of the model by measuring several spike train statistics commonly used in neuroscience: 1) Average number of spikes (spike-count) per neuron. 2) Average time course across activity patterns. 3) Covariance between pairs of neurons. 4) Lag-covariance between pairs of neurons: for each pair of neurons, we shift the activity of one of the neurons by one bin and compute the covariance between the resulting activities. This quantity thus indicates how strongly the activity of one of the neurons is related to the future activity of the other neuron. 5) Distribution of synchrony (or k-statistic), which corresponds to the probability $\mathrm{P}_N(k)$ that k out of the N neurons spike at the same time. 6) Spike autocorrelogram, computed by counting, for each spike, the number of spikes preceding and following the given spike in a predefined time window. The obtained trace is normalized to the peak (which is by construction at 0 ms) and the peak is then zeroed in order to help comparisons.

3 Results

3.1 Fitting the statistics of simulated spike trains

We first tested Spike-GAN with samples coming from the simulated activity of a population of 16 neurons whose firing probability followed a uniform distribution across the whole duration

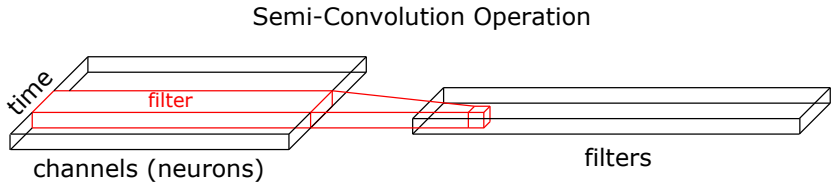

Figure 1: Semi-convolution operation: samples are transposed so as to input the neurons' activities into different channels. The convolutional filters (red box) then span all neurons but share weights across the time dimension.

(T=128 ms) of the samples (average firing rate around 100 Hz, Fig. 2D). In order to test whether Spike-GAN can approximate second-order statistics, the neurons' activities present two extra features that are commonly found in neural recordings. First, using the method described in Mikula & Niebur (2003), we introduced correlations between randomly selected pairs of neurons (correlation coefficient values around 0.3). Second, we imposed a common form of temporal correlations arising from neuronal biophysics (*refractory period*): following an action potential, a neuron typically remains silent for a few milliseconds before it is able to spike again. This phenomenon has a clear effect on the spike autocorrelogram that shows a pronounced drop in the number of spikes present at less than 2 ms (see Fig. 2E). We trained Spike-GAN on 8192 samples for 500000 iterations (Fig. S3 shows the critic's loss function across training).

A representative sample produced by a trained Spike-GAN together with the resulting patterns (after binarizing the samples, see Section 2.2) is shown in Fig. 2 (panel A). Note that the sample (black traces) is mostly binary, with only a small fraction of bins having intermediate values between 0 and 1. We evaluated the performance of Spike-GAN by measuring several spike train statistics commonly used in neuroscience (see Section 2.2). For comparison, we also trained a generative adversarial network in which both the generator and the critic are a 4-layer multi-layer perceptron (MLP) and the number of units per layer is adjusted so both models present comparable numbers of trainable variables (490 units per layer which results in ≈ 3.5M trainable variables). As Fig. 2 shows, while both models fit fairly well the first three statistics (mean spike-count, covariances and k-statistics), the Spike-GAN's approximation of the features involving time (average time course, autocorrelogram and lag-covariance) is considerably better than that of the MLP GAN. This is most likely due to the weight sharing performed by Spike-GAN along the temporal dimension, that allows it to easily learn temporally invariant features.

Finally, in Supp. Section A.1 we show that Spike-GAN is able to produce new, realistic activity patterns, hence demonstrating that it is not only memorizing the samples present in the training dataset but it is able to effectively mimic their underlying distribution.

3.2 Comparing to a state-of-the-art method

We next tested the Spike-GAN model on real recordings coming from the retinal ganglion cells (RGCs) of the salamander retina (Marre et al., 2014; Tkačik et al., 2014). The dataset contains the response of 160 RGCs to natural stimuli (297 repetitions of a 19-second movie clip of swimming fish and water plants in a fish tank) discretized into bins of 20 ms. We randomly selected 50 neurons out of the total 160 and partitioned their activity into non-overlapping samples of 640 ms (32 time bins) which yielded a total of 8817 training samples. We obtained almost identical results for a different set of 50 randomly selected neurons (data not shown).

In order to provide a comparison between Spike-GAN and an existing state-of-the-art method, we fit the same dataset with a maximum entropy approach developed by Tkačik et al. (2014), the so-called k-pairwise model. Briefly, maximum entropy (MaxEnt) models provide a way of fitting a predefined set of statistics characterizing a probability distribution while being maximally agnostic about any other aspect of such distribution, i.e. maximizing the entropy of the probability distribution given the constraints in the statistics (Pressé et al., 2013). In neuroscience, the most common approach has been to design MaxEnt models fitting the first and second-order statistics, i.e. the average firing rate and pairwise correlations between neurons (Tang et al., 2008; Schneidman et al., 2006; Shlens

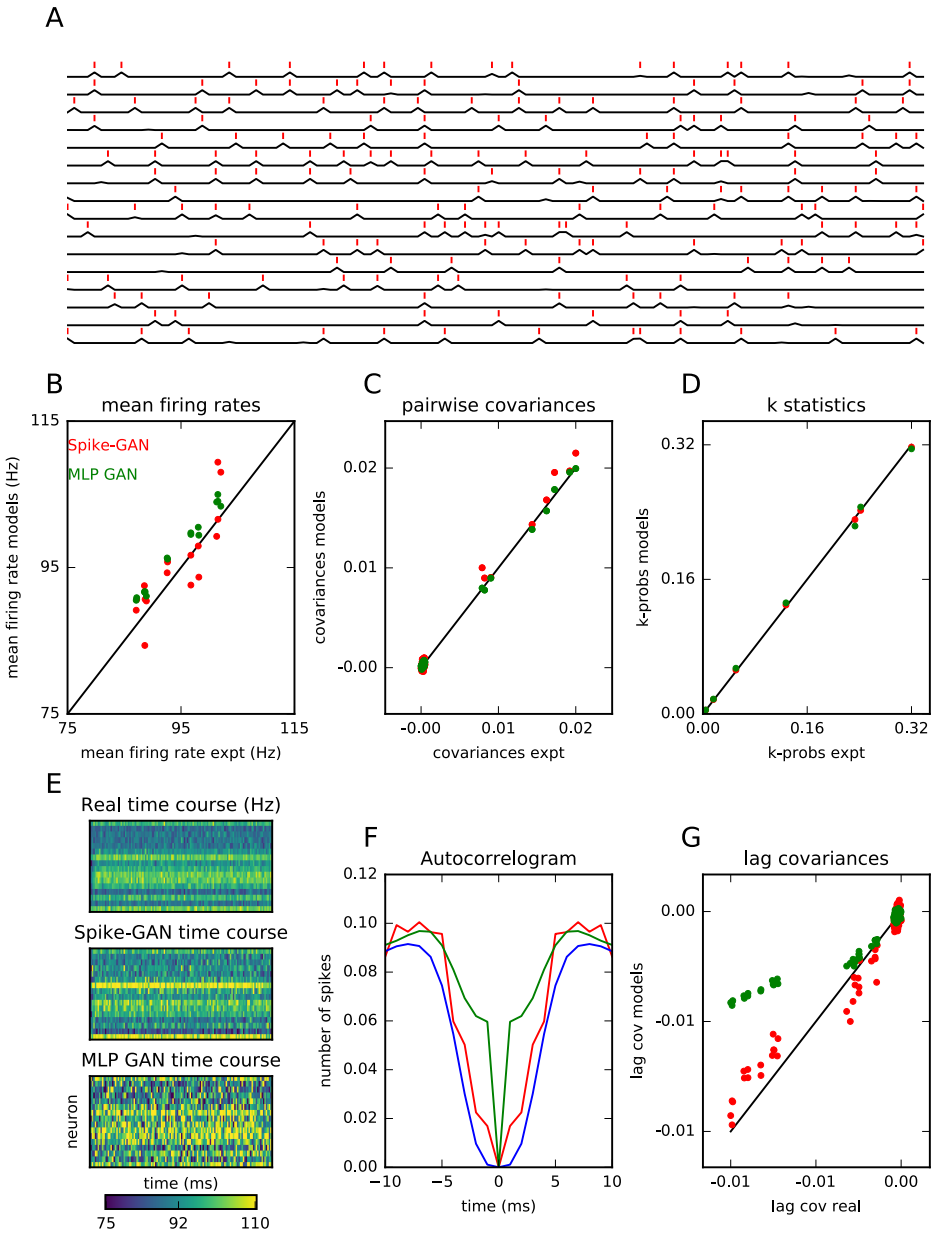

Figure 2: Fitting the statistics of simulated population activity patterns. A) Representative sample generated by Spike-GAN (black lines) and the resulting spike trains after binarizing (red lines). B-D) Fitting of the average spike-count, pairwise covariances and k-statistics done by Spike-GAN (red dots) and by a MLP GAN (green dots). Line indicates identity. E) Average time courses corresponding to the ground truth dataset and to the data obtained with Spike-GAN and the MLP GAN. F-G) Fitting of the autocorrelogram and the lag-covariances done by Spike-GAN (red line/dots) and a MLP GAN (green line/dots). Blue line corresponds to the autocorrelogram resulting from the ground truth distribution.

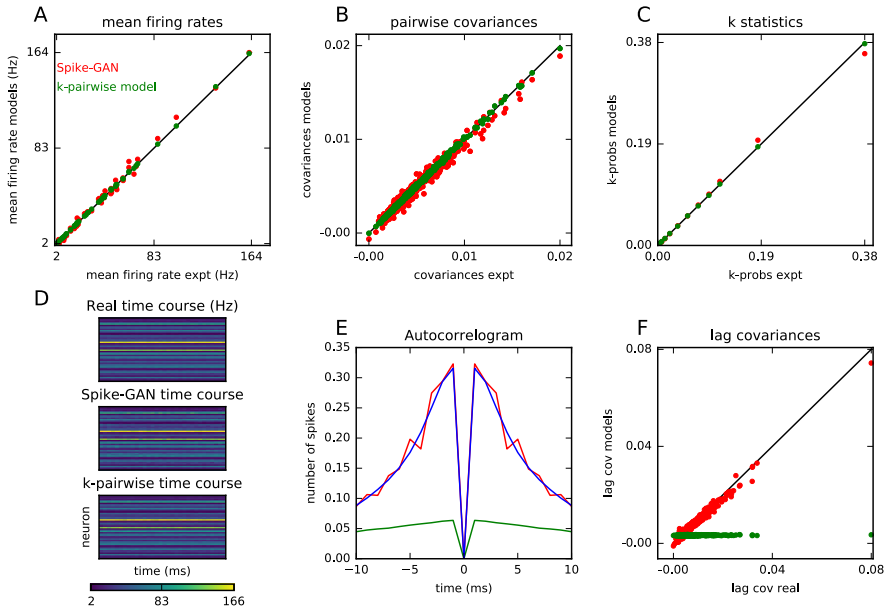

Figure 3: Fitting the statistics of real population activity patterns obtained in the retinal salamander. A-C) Fitting of the average spike-count, pairwise covariances and k-statistics done by Spike-GAN (red dots) and by the k-pairwise model (green dots). Line indicates identity. D) Average time courses corresponding to the ground truth data and to the data obtained with Spike-GAN and the k-pairwise model. E-F) Fitting of the autocorrelogram and the lag-covariances done by Spike-GAN (red line/dots) and the k-pairwise model (green line/dots). Blue line corresponds to the autocorrelogram resulting from the ground truth distribution.

et al., 2006). The k-pairwise model extends this classical approach to further constrain the activity of the neural population by fitting the k-statistics of the dataset of interest, which provides a measure of the neural population synchrony (see Section 2.2).

As shown in Fig. 3, both methods provide a good approximation of the average firing rate, the covariance and the k-statistics, but the fit performed by the MaxEnt model (green dots) is somewhat tighter than that produced by Spike-GAN (red dots). This is not surprising, as these are the aspects of the population activity distribution the k-pairwise model is specifically designed to fit. By contrast, Spike-GAN does remarkably well without any need for these statistical structures to be manually specified as features of the model.

Importantly, like most MaxEnt methods that have been proposed to model the activity of a population of neurons (Savin & Tkačik, 2017), the k-pairwise model does not take into account the temporal dynamics of the population and therefore ignores well-known neural features that are very likely to play a relevant role in the processing of incoming information (e.g. refractory period, burst or cross-correlation between pairs of neurons). Fig. 3 shows that Spike-GAN also approximates well the ground truth autocorrelogram and lag-covariances while the k-pairwise model, as expected, entirely fails to do so.

The above results demonstrate that Spike-GAN generates samples comparable to those produced by state-of-the-art methods and at the same time provides a good approximation of the neural activity dynamics, which can be of capital importance in many neuroscience studies.

3.3 Using the trained critic to infer relevant neural features

We then investigated what a trained critic can tell us about the population activity patterns that compose the original dataset. In order to do so, we designed an alternative dataset in which neural samples contain stereotyped activation patterns each involving a small set of neurons (Fig. 4A). This type of activation patterns, also called packets, have been found in different brain areas and have been

suggested to be fundamental for cortical coding, forming the basic symbols used by populations of neurons to process and communicate information about incoming stimuli (Luczak et al., 2015). Thus, besides being a good test for the capability of Spike-GAN to approximate more intricate statistical structures, analyzing simulated samples presenting packets constitutes an excellent way of demonstrating the applicability of the model to a highly relevant topic in neuroscience. We trained Spike-GAN on a dataset composed of neural patterns of 32 neurons by 64 ms that present four different packets involving non-overlapping sets of 8 neurons each (Fig. 4A). The probability of each type of packet to occur was set to 0.1. Packets could not overlap. We trained Spike-GAN on 8192 samples for 50000 iterations. Importantly, only a few neurons out of all the recorded ones typically participate in a given packet and, moreover, neurons are usually not sorted by the packet to which they belong. Therefore, real neural population activity is extremely difficult to interpret and packets are cluttered by many other 'noisy' spikes (Fig. 4B). In order to assess the applicability of Spike-GAN to real neuroscience experiments, we trained it on these type of realistic patterns of activity.

The most straightforward approach to investigate which aspects of a given input are most relevant for a neural network is to visualize the filters learned by the first layer of the network. Fig. S4 shows 64 randomly selected filters learned by the first layer of the Spike-GAN critic. Importantly, many of them display spatial distributions ideal to detect the packets (note that filters have been sorted in the neurons' dimension to help visualization).

As an alternative, Zeiler & Fergus (2014) proposed to systematically alter different parts of the input and compute the change each alteration produces in the output of different layers of the network. Here we have adapted this idea to investigate which are the most relevant features of a given neural activity pattern. We first compute the output produced by the critic for a real sample. Then, for a given neuron and a given temporal window of several milliseconds, we shuffle across time the spikes emitted by the neuron during that period of time and compute the output of the critic when using as input the altered sample. The absolute difference between the two outputs gives us an idea of how important is the structure of the spike train we have disrupted. We can then proceed in the same fashion for all neurons and for several time windows and obtain a map of the importance of each particular spike train emitted by each neuron (importance maps, see Fig. 4C, heatmaps).

To highlight the usefulness of the procedure explained above, we produced a dataset composed by neural activity patterns obtained from a population of neurons, some of which (8 out of 32in our example) respond to a particular stimulus with a latency of 16 ms and they do so for 16 ms by firing in packets as the ones described above. Fig. 4C (gray scale panels) shows 5 representative example patterns (see also Fig. S5). The packets emitted by the 8 participating neurons are highlighted for visualization, but it is clear that patterns containing packets are almost indistinguishable from those without them. Noticeably, the importance maps (heatmaps) are able to pinpoint the spikes belonging to a packet. Note that this does not require re-training of Spike-GAN. Further, by averaging the importance maps across time and space, we can obtain unambiguous results regarding the relevance of each neuron and time period (Fig. 4 D and E). The importance-map analysis thus constitutes a very useful procedure to detect the aspects of a given neural population activity pattern that characterize the probability distribution underlying that specific pattern.

4 Discussion

We explored the application of the Generative Adversarial Networks framework (Goodfellow et al., 2014) to synthesize neural responses that approximate the statistics of the activity patterns of a population of neurons. For this purpose, we put forward Spike-GAN by adapting the WGAN variant proposed by Arjovsky et al. (2017) and introducing a semi-convolutional operation that allows to share weights across time while maintaining a densely connected structure across neurons. We found that this network generated realistic spike trains without needing to be trained with specific assumptions about the statistics of neural firing or which features of the external world make neurons fire. The method reproduced to an excellent approximation the statistics of neural activity on which it was trained. In particular, and unlike most methods, the Spike-GAN reproduced well the time-dependent variations of firing, as well as the time averaged lower order statistics of the spike train.

Building on the work by Zeiler & Fergus (2014), we also showed how to use Spike-GAN to visualize the particular features that characterize the training dataset. Specifically, we detected the spikes

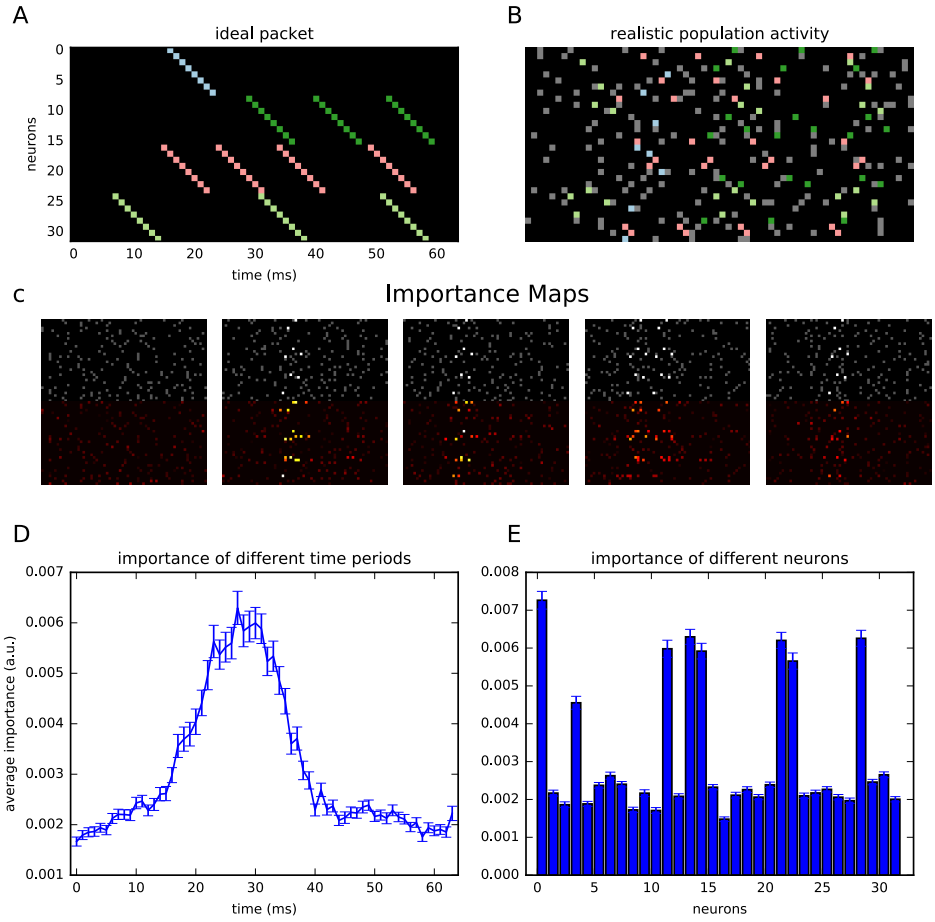

Figure 4: A) An example pattern showing the different packets highlighted with different colors and sorted to help visualization. B) Realistic neural population pattern (gray spikes do not participate in any pattern). C) Examples of activity patterns (grayscale panels) in which only one type of packet is present (one or two times) during a period of time from 16 to 32 ms. Packets are highlighted as white spikes. Heatmaps: importance maps showing the change that disrupting specific spikes has on the critic's output. Note that packet spikes normally show higher values. We have used a sliding window of 8 ms (with a step size of 2 ms) to selectively shuffle the activity of each neuron at different time periods. D) Average of 200 randomly selected importance maps across the neurons dimension, yielding importance as a function of time. E) Average of the same 200 randomly selected importance maps across the time dimension, yielding importance as a function of neurons.

that participate in generating activity motifs that are most salient in the spike trains. We believe that Spike-GAN will be useful for unsupervised identification of highly salient low-dimensional representations of neural activity, which can then be used to describe and interpret experimental results and discover the key units of neural information used for functions such as sensation and behavior.

One promising application of Spike-GAN is that of designing realistic patterns of stimulation that can be used to perturb populations of neurons using electrical or optical neural stimulation (Panzeri et al., 2017; Tehovnik et al., 2006). The ability of Spike-GAN to generate realistic neural activity including its temporal dynamics and to identify its most salient features suggests that it may become a very relevant tool to design perturbations.

Given that Spike-GAN is based on the deep neural networks framework, it should easily scale up to larger populations of neurons and samples of longer duration. Note also that the use of longer samples will only further accentuate the advantages of Spike-GAN over the MLP GAN (which

does not make use of the shift invariance along the temporal dimension) and models not taking into account the dynamics of the neural population.

ACKNOWLEDGMENTS

REFERENCES

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

Serena Bovetti and Tommaso Fellin. Optical dissection of brain circuits with patterned illumination through the phase modulation of light. *Journal of neuroscience methods*, 241:66–77, 2015.

Natasha A. Cayco-Gajic, Joel Zylberberg, and Eric Shea-Brown. Triplet correlations among similarly tuned cells impact population coding. *Frontiers in Computational Neuroscience*, 9: 57, 2015. ISSN 1662-5188. doi: 10.3389/fncom.2015.00057. URL `http://journal.frontiersin.org/article/10.3389/fncom.2015.00057`.

Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. *CoRR*, abs/1606.03657, 2016. URL `http://arxiv.org/abs/1606.03657`.

Soumith Chintala, Emily Denton, Martin Arjovsky, and Michael Mathieu. How to train a GAN? Tips and tricks to make GANs work. *GitHub*, 2016.

Mark M Churchland, M Yu Byron, Maneesh Sahani, and Krishna V Shenoy. Techniques for extracting single-trial activity patterns from large-scale neural recordings. *Current Opinion in Neurobiology*, 17(5):609–618, 2007.

Pascal Fries, John H Reynolds, Alan E Rorie, and Robert Desimone. Modulation of oscillatory neuronal synchronization by selective visual attention. *Science*, 291(5508):1560–1563, 2001.

Wulfram Gerstner and Werner M Kistler. *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge University Press, 2002. URL `http://icwww.epfl.ch/~gerstner/BUCH.html`.

Ian Goodfellow. Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*, 2016.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pp. 2672–2680, 2014.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of Wasserstein GANs. *arXiv preprint arXiv:1704.00028*, 2017.

Robin AA Ince, Stefano Panzeri, and Christoph Kayser. Neural codes formed by small and temporally precise populations in auditory cortex. *Journal of Neuroscience*, 33(46):18277–18287, 2013.

Justin Keat, Pamela Reinagel, R Clay Reid, and Markus Meister. Predicting every spike: a model for the responses of visual neurons. *Neuron*, 30(3):803–817, 2001.

Christina K Kim, Avishek Adhikari, and Karl Deisseroth. Integration of optogenetics with complementary methodologies in systems neuroscience. *Nature Reviews Neuroscience*, 18(4):222–235, 2017.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Urs Köster, Jascha Sohl-Dickstein, Charles M Gray, and Bruno A Olshausen. Modeling higher-order correlations within cortical microcolumns. *PLoS Computational Biology*, 10(7):e1003684, 2014.

Vernon Lawhern, Wei Wu, Nicholas Hatsopoulos, and Liam Paninski. Population decoding of motor cortical activity using a generalized linear model with hidden states. *Journal of Neuroscience Methods*, 189(2):267–280, 2010.

William Lotter, Gabriel Kreiman, and David Cox. Unsupervised learning of visual structure using predictive generative networks. nov 2015. *URL http://arxiv. org/abs/1511.06380*.

Artur Luczak, Bruce L McNaughton, and Kenneth D Harris. Packet-based communication in the cortex. *Nature Reviews Neuroscience*, 16(12):745–755, 2015.

Jakob H Macke, Philipp Berens, Alexander S Ecker, Andreas S Tolias, and Matthias Bethge. Generating spike trains with specified correlation coefficients. *Neural Computation*, 21(2):397–423, 2009.

Gaby Maimon and John A Assad. A cognitive signal for the proactive timing of action in macaque lip. *Nature Neuroscience*, 9(7):948–955, 2006.

Olivier Marre, Gasper Tkacik, Dario Amodei, Elad Schneidman, William Bialek, and II Berry, Michael J. Multi-electrode array recording from salamander retinal ganglion cells. *IST Austria*, 2014.

Lane McIntosh, Niru Maheswaranathan, Aran Nayebi, Surya Ganguli, and Stephen Baccus. Deep learning models of the retinal response to natural scenes. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 29*, pp. 1369–1377. Curran Associates, Inc., 2016. URL `http://papers.nips.cc/paper/6388-deep-learning-models-of-the-retinal-response-to-natural-scenes.pdf`.

Shawn Mikula and Ernst Niebur. The effects of input rate and synchrony on a coincidence detector: analytical solution. *Neural Computation*, 15(3):539–547, 2003.

Ari S Morcos and Christopher D Harvey. History-dependent variability in population dynamics during evidence accumulation in cortex. *Nature neuroscience*, 19(12):1672–1681, 2016.

Rubén Moreno-Bote, Jeffrey Beck, Ingmar Kanitscheider, Xaq Pitkow, Peter Latham, and Alexandre Pouget. Information-limiting correlations. *Nature Neuroscience*, 17(10):1410–1417, 2014.

Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier GANs. *arXiv preprint arXiv:1610.09585*, 2016.

Ifije E Ohiorhenuan, Ferenc Mechler, Keith P Purpura, Anita M Schmid, Qin Hu, and Jonathan D Victor. Sparse coding and high-order correlations in fine-scale cortical networks. *Nature*, 466 (7306):617–621, 2010.

Stefano Panzeri, Christopher D Harvey, Eugenio Piasini, Peter E Latham, and Tommaso Fellin. Cracking the neural code for sensory perception by combining statistics, intervention, and behavior. *Neuron*, 93(3):491–507, 2017.

Jonathan W Pillow, Jonathon Shlens, Liam Paninski, Alexander Sher, Alan M Litke, EJ Chichilnisky, and Eero P Simoncelli. Spatio-temporal correlations and visual signalling in a complete neuronal population. *Nature*, 454(7207):995–999, 2008.

Steve Pressé, Kingshuk Ghosh, Julian Lee, and Ken A. Dill. Principles of maximum entropy and maximum caliber in statistical physics. *Rev. Mod. Phys.*, 85:1115–1141, Jul 2013. doi: 10.1103/RevModPhys.85.1115. URL `https://link.aps.org/doi/10.1103/RevModPhys.85.1115`.

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

Cristina Savin and Gašper Tkačik. Maximum entropy models as a tool for building precise neural controls. *Current Opinion in Neurobiology*, 46:120–126, 2017.

Elad Schneidman, Michael J Berry, Ronen Segev, and William Bialek. Weak pairwise correlations imply strongly correlated network states in a neural population. *Nature*, 440(7087):1007–1012, 2006.

Jonathon Shlens, Greg D. Field, Jeffrey L. Gauthier, Matthew I. Grivich, Dumitru Petrusca, Alexander Sher, Alan M. Litke, and E. J. Chichilnisky. The structure of multi-neuron firing patterns in primate retina. *Journal of Neuroscience*, 26(32):8254–8266, 2006.

Richard B. Stein. A theoretical analysis of neuronal variability. *Biophysical Journal*, 5(2):173 – 194, 1965. ISSN 0006-3495. doi: 10.1016/S0006-3495(65)86709-1. URL http://www.sciencedirect.com/science/article/pii/S0006349565867091.

Aonan Tang, David Jackson, Jon Hobbs, Wei Chen, Jodi L Smith, Hema Patel, Anita Prieto, Dumitru Petrusca, Matthew I Grivich, Alexander Sher, et al. A maximum entropy model applied to spatial and temporal correlations from cortical networks in vitro. *Journal of Neuroscience*, 28(2):505–518, 2008.

EJ Tehovnik, AS Tolias, F Sultan, WM Slocum, and NK Logothetis. Direct and indirect activation of cortical neurons by electrical microstimulation. *Journal of Neurophysiology*, 96(2):512–521, 2006.

Gašper Tkačik, Olivier Marre, Dario Amodei, Elad Schneidman, William Bialek, and Michael J Berry II. Searching for collective behavior in a large network of sensory neurons. *PLoS Computational Biology*, 10(1):e1003408, 2014.

Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European Conference on Computer Vision*, pp. 818–833. Springer, 2014.

A APPENDIX

A.1 GENERATING NEW, REALISTIC PATTERNS OF NEURAL ACTIVITY

We investigated how well Spike-GAN fits the whole probability density function from which the population activity patterns present in the ground truth dataset are drawn, following an approach inspired by Macke et al. (2009). We first produced a *ground truth* dataset for a small-size problem (2 neurons x 12 time bins x 8192 samples) so as to reduce the dimensionality of the samples and to be able to approximate the underlying probability density function, i.e. the theoretical probability of each pattern (see below). We trained Spike-GAN on this ground truth dataset, and we used the trained network to generate a synthetic dataset with the same number of samples (8192). We then considered all spike trains appearing in both the ground truth and the synthetic dataset, and we computed their empirical frequencies of occurrence with respect to both datasets. In general, we expect these empirical frequencies to be only an approximation of their true underlying probability, not only for the synthetic dataset, but also for the ground truth, due to the effect of finite sampling. Therefore, to compare how well the two datasets approximate the real underlying distribution, and thus evaluate the performance of Spike-GAN by comparing it with direct sampling of the true distribution, we obtained an approximation of the theoretical probabilities of the spike trains by computing their frequencies in a much larger dataset ($2 \cdot 10^6$ samples). We will call these frequencies 'numerical probabilities'. We then plotted the spike train frequencies with respect to both the ground truth and the generated dataset against these numerical probabilities. By comparing the densities in Fig. S1A, we deduce that the empirical frequency distribution deviates from the theoretical distribution (the identity line) in the same way for the generated data as for the real data. Hence, this deviation can only be attributed to finite sampling effects rather than poor performance of Spike-GAN. This shows that Spike-GAN is approximating very well the higher-order statistics of the spike trains (Macke et al., 2009).

More generally, we reasoned that if Spike-GAN is effectively learning the true distribution from the initial ground truth dataset, any new dataset sampled from Spike-GAN should look like a new dataset sampled from the true distribution, both in the way it overlaps with the initial dataset (as described above) and in the way it doesn't. Therefore, we next focused on those samples that were generated by Spike-GAN which were not contained in the ground truth dataset used for training. To check whether these new spike trains were realistic, we generated 100 synthetic datasets with Spike-GAN and compared the new spike trains (i.e. not present in the ground truth dataset) therein with those produced by sampling 100 new datasets from the true distribution (Fig S1B-C). Unsurprisingly, in both cases (synthetic and newly sampled ground truth data), these new samples had lower numerical probability than those that also appeared in the initial training dataset, as selecting only the "new" samples not present in the training dataset obviously introduces a bias towards less probable patterns. However, the match in the distribution of numerical probability and empirical frequency between synthetic data and newly sampled ground truth data is excellent (Fig S1B). Finally, the distribution of the number of spike trains with zero theoretical probability across generated datasets roughly matched, on average, that obtained by repeated sampling of the true distribution (Fig S1C). Taken together these observations confirmed that Spike-GAN was able to synthesize *new* spike trains that were not observed during its training, which are as realistic as those coming from the true underlying distribution.

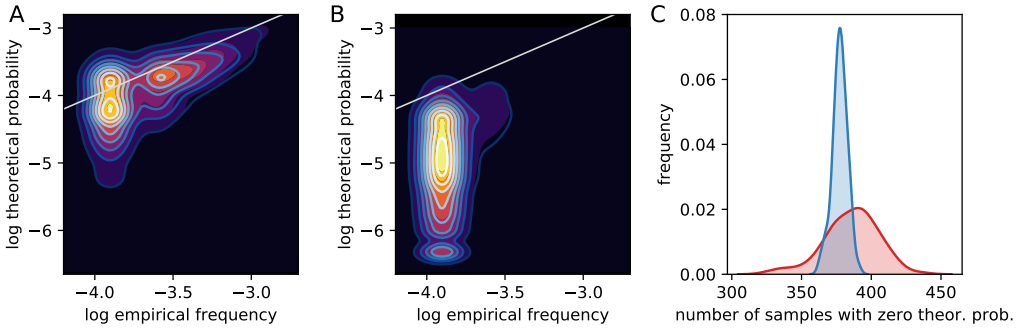

Figure S1: Fitting higher-order statistics and creating new realistic spike trains. A) numerical probability vs empirical frequency of patterns contained both in the ground truth dataset used to train Spike-GAN and in a synthetic dataset of the same size generated by Spike-GAN. Gray line is the identity. Reds: empirical frequencies computed with respect to the ground truth dataset. Blues: empirical frequencies computed with respect to the synthetic data. Both distributions are obtained by kernel density estimation, with a 2D Gaussian kernel with bandwidth=0.1. B) as A, for spike patterns not present in the original training dataset, cumulated over 100 repeated samples of the true distribution (reds) and 100 synthetic datasets generated by Spike-GAN. C) distribution of the number of samples with zero numerical probability, across 100 repeated samples of the true distribution (red) and 100 synthetic datasets generated by Spike-GAN (blue). KDE obtained with a Gaussian kernel with bandwidth chosen by Scott's rule.

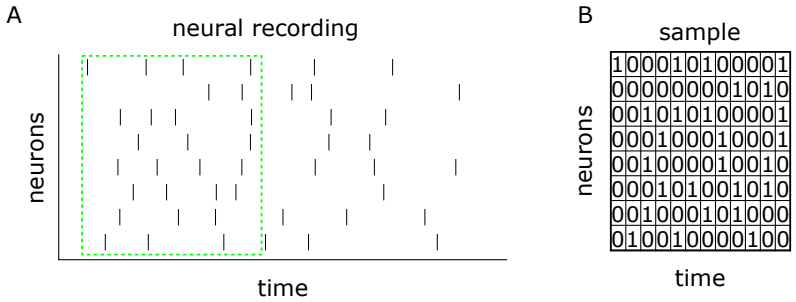

Figure S2: A) Example of neural recording in which the activity of 8 neurons is measured during an arbitrary time period. B) Example of a sample extracted from the activity shown in A (green box).

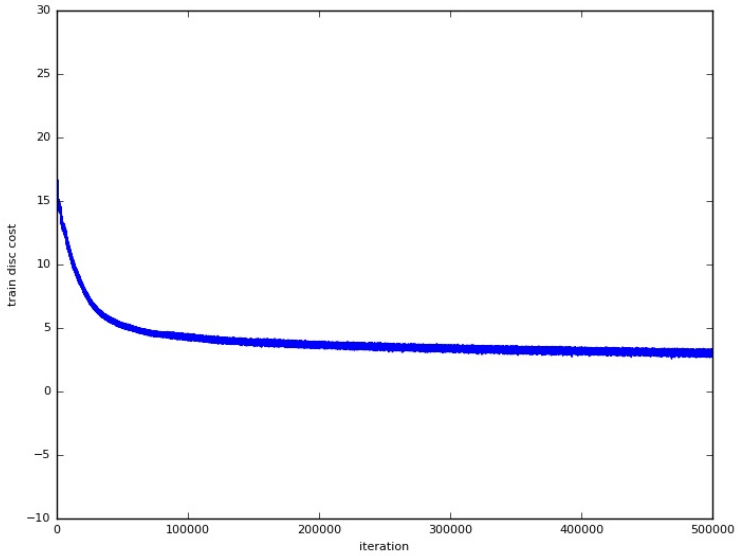

Figure S3: Negative critic loss corresponding to the training of Spike-GAN on samples coming from the simulated activity of a population of 16 neurons whose firing probability follows a uniform distribution across the whole duration (T=128 ms) of the samples (see Section 3).

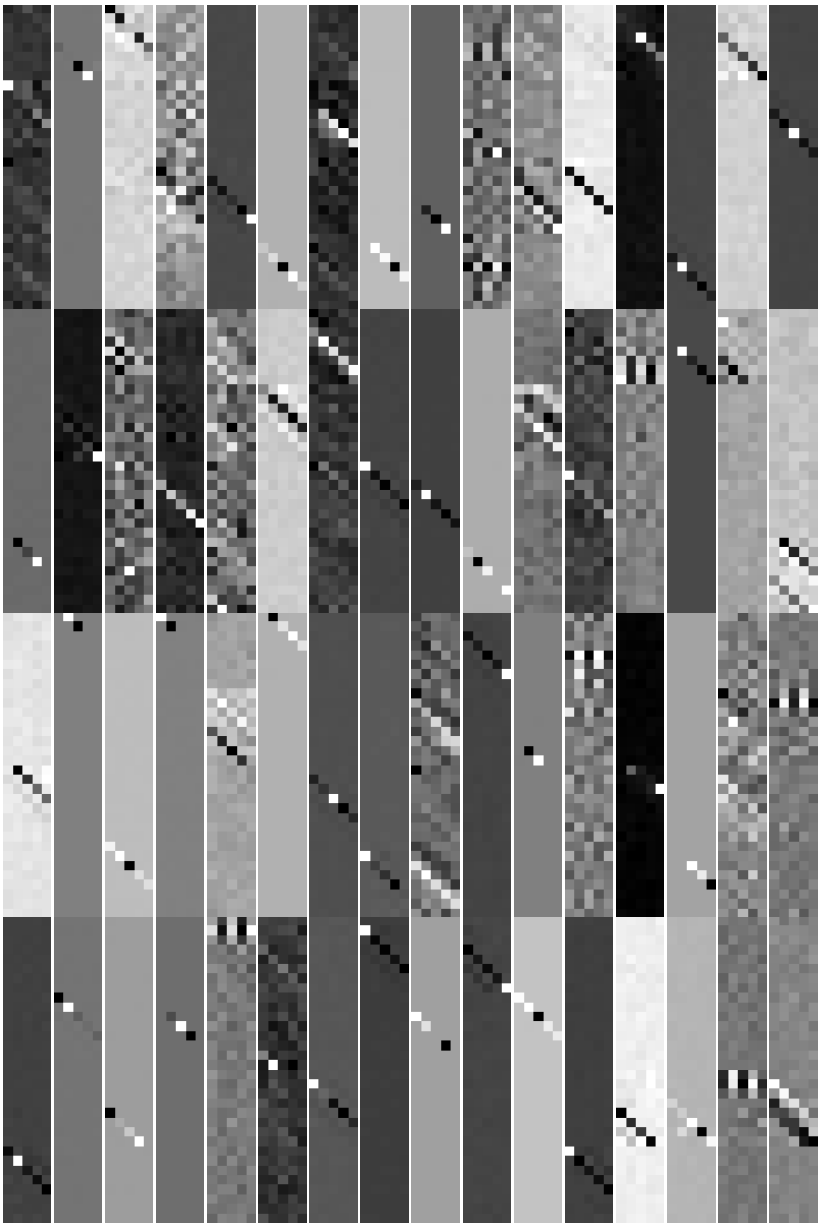

Figure S4: Filters learned by the first layer of Spike-GAN when trained on the dataset described in Section 3.3

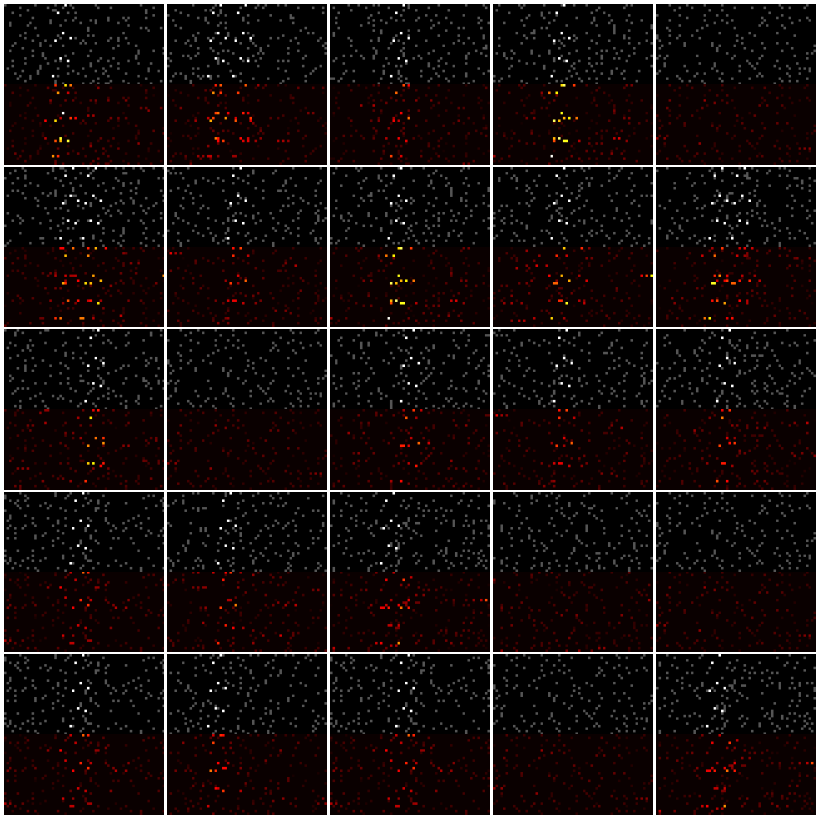

Figure S5: Randomly selected samples and their corresponding importance maps.