Spike Distance Function as a Learning Objective for Spike Prediction

Kevin Doran [1 2] **Marvin Seifert** [1] **Carola A. M. Yovanovich** [1] **Tom Baden** [1 3]

Abstract

Approaches to predicting neuronal spike responses commonly use a Poisson learning objective. This objective quantizes responses into spike counts within a fixed summation interval, typically on the order of 10 to 100 milliseconds in duration; however, neuronal responses are often time accurate down to a few milliseconds, and Poisson models struggle to precisely model them at these timescales. We propose the concept of a spike distance function that maps points in time to the temporal distance to the nearest spike. We show that neural networks can be trained to approximate spike distance functions, and we present an efficient algorithm for inferring spike trains from the outputs of these models. Using recordings of chicken and frog retinal ganglion cells responding to visual stimuli, we compare the performance of our approach to that of Poisson models trained with various summation intervals. We show that our approach outperforms the use of Poisson models at spike train inference.

1. Introduction

This paper proposes a new learning objective for the problem of spike prediction. Spike prediction is the task of estimating the timing of future action potentials (spikes) of a neuron; for example, given 1000 ms of stimulus and spike activity, predict the next 80 ms of spike activity. The task is illustrated in Figure 1.

The design of retinal prosthetics is an example where predicting spike sequences is important: patients suffering from diseases such as retinitis pigmentosa can have their vision partially restored through electrical or optogenetic stimulation that attempts to reproduce the spike sequences of retinal ganglion cells (RGCs) in a healthy retina (Borda and Ghezzi, 2022; Sahel et al., 2021).

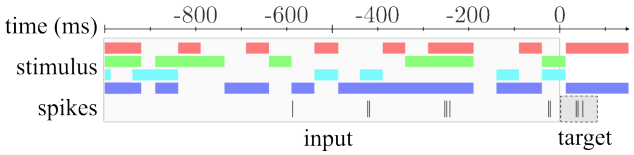


Figure 1. **Spike prediction task:** predict future spikes given stimulus and spike history. For this work, stimulus history is a 1-second snippet of a widefield stimulus at four wavebands (420 nm, 480 nm, 505 nm and 630 nm). The prediction duration is not fixed—several durations are explored. Information about the stimulus after $t = 0$ is not available to the model, matching the task faced by retinal prosthetic devices.

A widely used approach to spike prediction is to train a model to predict the number of spikes that will occur within a fixed time interval. A probabilistic motivation is often given by describing the neuron's output as a Poisson process. In this setting, the model's output is interpreted as a firing rate and corresponds to the single parameter of the Poisson process at a certain point in time. Training the model amounts to maximizing the likelihood of this time-varying parameter with respect to the data.

For the Poisson approach, the interval length over which spikes are summed is a hyperparameter that impacts the training and inference of a model. As will be described in Section 3, the desire to reach millisecond resolution encourages shorter summation intervals; however, as the summation interval is reduced, training becomes more difficult, and both the effects of binning and the shape of the Poisson distribution can limit performance.

In this work, we propose a summation-free approach. Rather than reduce a chunk of a spike train to a single number, we choose a representation that preserves its details. Inspired by the idea of a signed distance function used in geometric modelling, we propose the idea of a *spike distance function* that maps points in time to the temporal distance to the nearest spike. We train neural networks to output this representation and use it to predict spike trains. A demonstration of the spike distance approach (still imprecisely defined for now) compared to the Poisson approach is shown in Figure 2. The process of predicting spikes is shown in Figure 3.

[1]School of Life Sciences, University of Sussex, UK [2]School of Engineering and Informatics, University of Sussex, UK [3]Institute of Ophthalmic Research, University of Tübingen, Germany. Correspondence to: Kevin Doran <k.a.doran@sussex.ac.uk>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

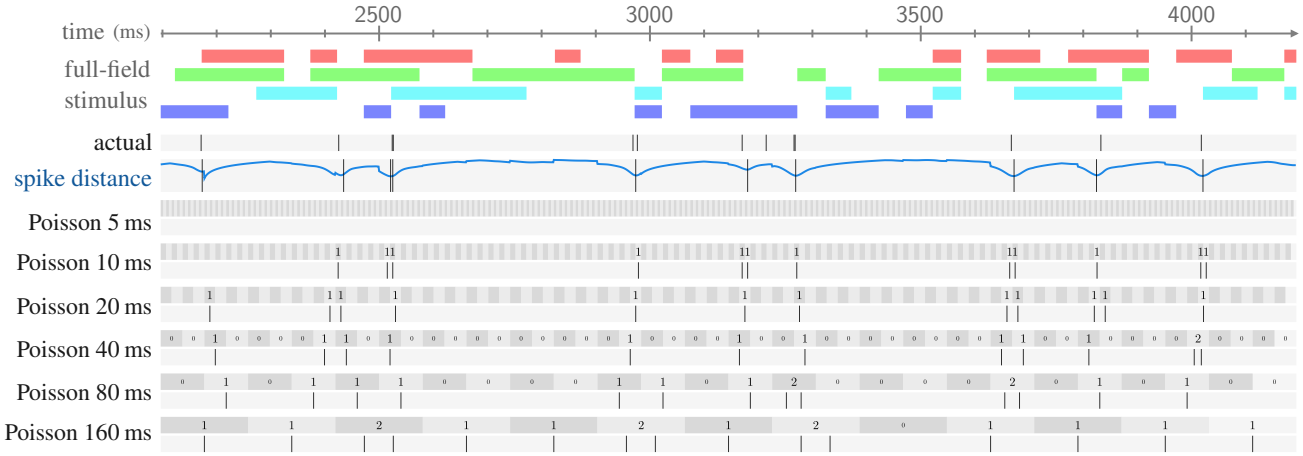


Figure 2. **Spike prediction using spike distance.** The output of a spike distance model (depicted by the blue line) can be used to precisely predict individual spikes. Section 5.1 describes the algorithm used for inferring spikes from a spike distance array. The Poisson models have their outputs rounded and that number of spikes are tiled over the interval. With this inference strategy, a fixed summation interval introduces a trade-off: longer intervals prevent precise spike localization, while shorter intervals cause predictions to tend towards 0 for all time steps. Section 5.2 and 5.3 describe other approaches for inferring spikes from outputs of Poisson models.

We inspect the effectiveness of this new approach by comparing a neural network trained using the spike distance objective with neural networks trained using a range of Poisson objectives, each using a different summation interval. Comparison is based on models' ability to infer spike trains: we predict spike trains from model outputs and compare these against ground truth spike trains using spike train similarity measures.

Our spike prediction dataset is formed from a multi-electrode array (MEA) recording of 60 chicken RGCs responding to visual stimuli. We repeat the experiments for 113 frog RGCs from a separate recording.

In the next section, we present related work. We come back to set the scene more concretely by elucidating the drawbacks of the Poisson approach in Section 3 and describing the spike distance function in Section 4. How to infer spike trains from model outputs is covered in Section 5. The experiments are described in Section 6 with results appearing in Section 7, and a discussion follows in Section 8.

2. Related Work

For the task of spike prediction, the effectiveness despite simplicity of generalized linear models (GLMs) makes them important models to study. Using GLMs to model neurons is covered by Weber and Pillow (2017). GLMs are further relevant because their success has contributed to the popularity of the Poisson learning objective.

GLMs are effectively single-layer neural networks, and so the neural network approaches that follow can be thought of as trading the simplicity of GLMs with the increased power of deeper neural networks. McIntosh et al. (2016) used convolutional neural networks (CNNs) and recurrent neural networks (RNNs) to predict spiking behaviour of tiger salamander RGCs. In parallel, Batty et al. (2016) carried out similar work using RNNs to predict spiking behaviour of macaque RGCs. Goglettino et al. (2024) used CNNs to predict responses of human RGCs. These three studies recorded the cells using MEAs. Cadena et al. (2019) compared CNNs trained from scratch with a fine-tuned vision model at predicting spiking behaviour of macaque V1 neurons. This study recorded cells using penetrative probes. In all of these studies, models have a single output which is interpreted as a parameter of a probability distribution modelling the spike count in an interval.

At a finer scale, spike rates are no longer a suitable abstraction, and spike prediction becomes an example of the general problem of locating points in 1 or more dimensions or locating surfaces in 2 or more dimensions and so on, of which there are other applications. Earthquake prediction faces such a problem where the task is framed as predicting the "time to failure". Computer vision and graphics have an analogous concept—a signed distance function—which is effectively a "distance to surface" and is used for implicitly modelling 2D surfaces in 3D space. It is from these contexts that we draw inspiration to propose "distance to spike" as an effective tool for spike prediction.

For earthquake prediction, Rouet-Leduc et al. (2017) and (Wang et al., 2022) applied machine learning to predict time-to-failure for a laboratory fault system. See the review by (Ren et al., 2020) for broad coverage. Interestingly, these approaches stop at predicting time-to-failure signals and do not take the next step of inferring precise rupture times from

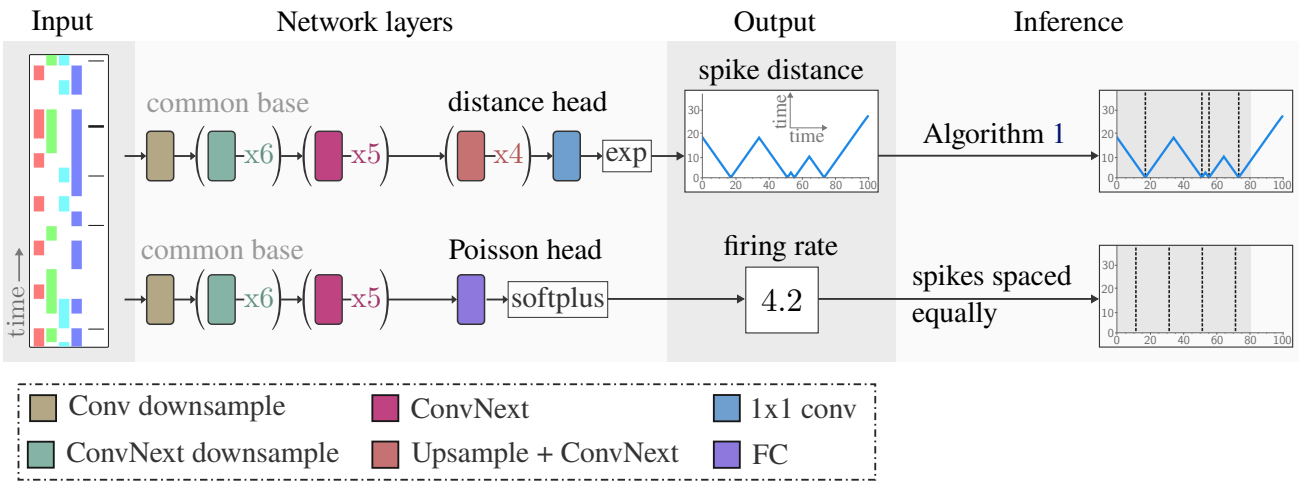


Figure 3. **Overview** of spike prediction for both the spike distance approach and the Poisson approach. To make the two comparable, the networks used by each approach were designed with the preponderance of weights and computation carried out using the same twelve-layer base architecture. ConvNext refers to the layers introduced by Liu et al. (2022). Algorithm 1 is described in Section 5.1.

these signals. The inference procedure described in Section 5 will work with these signals also.

In computer vision, the notion of a signed distance function is used to implicitly define surfaces. Our work to predict spike distance is analogous to the work by Zeng et al. (2017) and Dai et al. (2017) who used neural networks to represent 3D shapes using a grid of truncated signed distance values. Later work in this area has tackled the scaling issues with 3D data, such as the work by Park et al. (2019), where neural networks were given the additional job of querying the 3D representation. The directional queries of the 3D representation are conceptually similar to spike-stimulus history input that we will use to request output behaviour from our neural network model. Although not considered in this work, the creative forms of querying 3D representations, such as Sitzmann et al.'s (2021) use of Plücker coordinates, raise the question: are there other useful ways to query neural network representations for neuronal behaviour?

Instead of predicting spikes in an interval, an alternative is to predict the time until the next spike. The area of neural temporal point processes tackles this problem for arbitrary event sequences by outputting a probability distribution over the next event time, given an event history. Two reviews of this area are Shchur et al. (2021) and Bosser and Taieb (2023). A drawback of this approach is that inference requires a forward pass through the neural network after each spike, which becomes impractical for applications such as retinal prosthetics, as spikes of a single cell can be separated by a few milliseconds, and across cells, spikes can be recorded as coincident. Additionally, the stimulus stream strongly affects responses and quickly invalidates any prediction made beyond a short future interval, making next-spike prediction less suitable in this setting.

Outside the context of neural networks, other works have investigated non-Poisson approaches to modelling spike emission. For example, both the work of Teh and Rao (2011) and Liu and Lengyel (2023) model spike emission with renewal processes modulated by Gaussian processes. These works support the idea that in the context of neural networks for spike prediction, alternatives to the incumbent Poisson approach are worth exploring.

3. Issues With the Poisson Approach

The summation interval used in the Poisson approach to spike prediction leads to a trade-off between temporal resolution and effective training and inference.

The goal of having predictions with high temporal resolution encourages the choice of a shorter summation interval. Recordings of chicken RGCs show bursts of spikes with spike intervals as short as 1–2 ms (Seifert et al., 2023); and primate, salamander, cat and rabbit RGCs have been shown to have stimulus-driven variability across trials as low as 1 ms (Uzzell and Chichilnisky, 2004; Berry and Meister, 1998; Keat et al., 2001; Berry et al., 1997). Neurons from other brain regions also demonstrate this precision, such as neurons in the lateral geniculate nucleus of cats (Butts et al., 2011; Keat et al., 2001). Furthermore, there is evidence that this precision is linked to function: retinal ganglion cells have been shown to vary their response time to stimuli on the order of milliseconds depending on the spatial structure of stimuli (Gollisch and Meister, 2008) and the wavelength of stimuli (Seifert et al., 2023).

This suggests that summation intervals as short as 1 ms in duration would be useful. However, decreasing the summation interval can cause several issues. Training slows

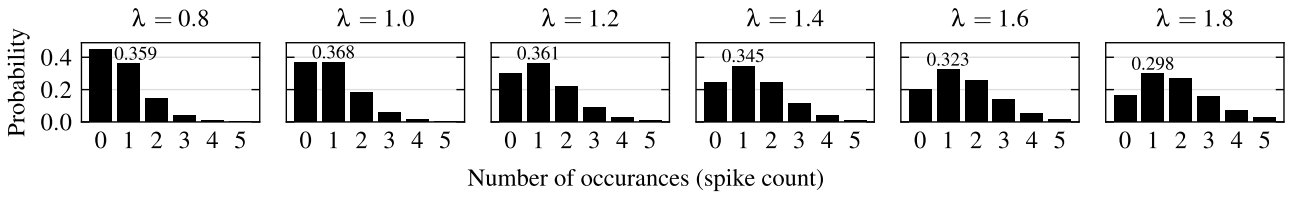


Figure 4. The probability mass function of the Poisson distribution for six values of the distribution's parameter, λ. The probability assigned to 1 is labelled. For a single non-zero value such as 1, no λ allows a concentration of probability at that value. The distribution is unsuitable as part of a learning objective where a model is expected to express strong confidence in any single non-zero value.

and becomes more difficult as data becomes spread over an increased number of samples, most of which are empty of spikes. If carrying out inference by maximizing probability, zero spikes can become the most compatible prediction for all time steps, as short summation intervals can lead to model outputs being consistently close to zero. These issues are ameliorated with longer intervals, but at the cost of temporal resolution.

With a long interval of say 80 ms, 80 ms of neural activity will be compressed into a single scalar value—a quick burst of spikes a few milliseconds apart will be indistinguishable from the same number of spikes spread over the entire interval. At training time, this results in a loss signal that has no sensitivity to how spikes fall within the interval but is overly sensitive to subtle movements of spikes crossing the interval boundary. At inference time, a model's scalar output does not indicate the positions where spikes might occur, despite it being plausible that a capable model would estimate this information in some form before arriving at a spike rate.

The above issues will affect any setup where training and inference works with isolated bins of spike counts. Using a Poisson distribution has an additional issue in that its shape is restrictive. The distribution's single parameter is simultaneously the mean and variance of the distribution, which does not allow a model to express strong confidence in any specific non-zero spike count. For example, the largest probability assignable to the occurrence of 1 spike is 36.8%, which occurs when the Poisson's λ parameter is set to 1.0; however, at this λ, 36.8% is also assigned to the occurrence of 0 spikes. Figure 4 demonstrates this situation by displaying the Poisson distribution for a range of λ values between 0.8 and 1.8. This becomes an issue at low summation intervals as the preponderance of spike counts will be 0 or 1.

The exponential distribution is the inter-event interval equivalent of the Poisson distribution. When modelling inter-event intervals, it is common to deviate from the exponential distribution, for example by using self-exciting Hawkes processes (Hawkes, 1971). This supports the idea that it is reasonable to seek flexibility beyond what the Poisson distribution offers.

With spike prediction in mind, it would be desirable to have a loss function that changes proportionately to changes in spike position and to have an output representation that facilitates spike position inference. Furthermore, a model should not be hindered from expressing strong confidence in specific spike patterns. The spike distance function described next can be used to create a learning objective with these properties.

4. Spike Distance

A *Spike distance function* maps each point in time to a scalar representing the temporal distance to the nearest spike. This is either the time elapsed since the last spike or the time remaining until the next spike, whichever is less. This is an implicit representation of a set of spikes using contours. A comprehensive treatment of using implicit functions to represent points and surfaces is Osher and Fedkiw (2003).

Formally, let $S \in \mathbb{R}^N$ be a set of N spike times. Let $I = [a, b]$ be an interval of $\mathbb{R}$. The spike distance function of S on I is the function $f_S : I \to \mathbb{R}$ defined as:

$$f_S(t) = \min_{s \in S} |t - s|$$

Any interval can support a spike distance function, and this spike distance function can depend on spikes both inside and outside the interval. An example spike distance function evaluated over the interval $[0\,\text{ms}, 128\,\text{ms}]$ for the spikes $[20\,\text{ms}, 60\,\text{ms}, 65\,\text{ms}, 86\,\text{ms}]$ is shown in Figure 5.

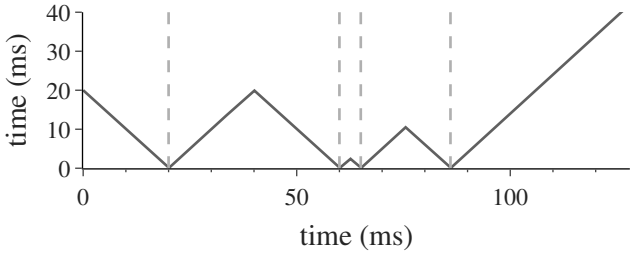


Figure 5. The spike distance function in milliseconds over the interval $[0, 128]$ for the spikes $[20, 60, 65, 86]$.

From the example, it can be seen that subtle or large changes to a spike train will be reflected proportionately in the spike distance function, which was the characteristic lacking in the Poisson loss signal.

In practice, we will work with discrete arrays and use a *discrete spike distance function*, $\hat{f}_S : \mathbb{N} \to \mathbb{R}$, where S is now an array of spike counts. We assume that the sample rate is fast enough that no two spikes are recorded in the same bin, and so S can be treated as a binary array. Appendix A.14 covers the discrete spike distance in more detail, including generalizing it to allow multiple spikes to be recorded in a single sample. Two other considerations—the choice of a maximum distance and the choice of the evaluation interval—are discussed in Appendix A.13.

5. Spike Train Inference

For the spike prediction task, we will infer a spike train from a model's outputs. For both the spike distance model and the Poisson models this will be done autoregressively: concatenating model outputs and using previously predicted spikes as input to subsequent forward passes. The two approaches differ in how they carry out a single step—each is covered below.

5.1. Spikes From a Spike Distance Array

We describe spike inference from a spike distance array from the perspective of energy minimization. An array of spike distance values outputted by a model can be seen as parameterizing some function that assigns a scalar energy value to any set of spikes. Various functions are possible. We use the L2 norm between the model output and the discrete spike distance function of a candidate spike sequence. Let the model output $M = (m_i)_{i=0}^{L-1}$ be a sequence of reals of length L, and let the binary sequence $S = (s_i)_{i=0}^{L-1}$ be a candidate spike count sequence of the same length. For example, with $L = 5$, the model output could be $M = [1.1, 0.5, 0.9, 1.1, 0.6]$ and a candidate spike sequence count be $S = [0, 1, 0, 0, 1]$. The energy assigned to S is given by:

$$E_M(S) = \sum_{i=0}^{L-1} (\hat{f}_S(i) - m_i)^2 \tag{1}$$

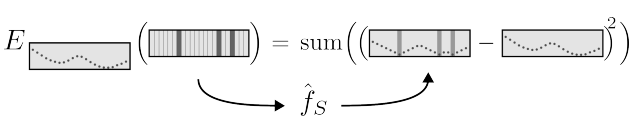


Figure 6. Visual form of Equation 1. The discrete spike distance, $\hat{f}_S$, is described further in Appendix A.14.

The energy is a measure of incompatibility between the candidate spikes and the model output (here we are following the convention that low energy values correspond to greater compatibility). Spike train inference amounts to finding the spike train with the lowest energy.

This optimization problem can be solved exactly by brute-force search. Dynamic programming can also be used, as the problem can be decomposed into left and right subproblems. Both of these approaches were found to be too slow in practice. We propose an inexact solution, Algorithm 1, that begins by predicting a spike in every time step then iteratively refines the prediction by removing spikes.

Algorithm 1 Spike inference algorithm: predict a spike in every time step then iteratively remove spikes—remove if the L2 norm between the candidate and target distance arrays is lower when the spike is not present. Iterate spikes in order of the estimated effect on the error. The function SPIKEDIST converts a set of spikes into an array of spike distances (see Appendix A.14 for details).

Require: w is the target spike distance (model output) of length L and s_0 is the most recent "past" spike.

```
function SPIKEINFERENCE(w, s_0)
    S ← indices of w  (i.e. [0, 1, ..., L − 1])
    scores ← w
    repeat
        S_old ← S
        for all s ∈ S sorted descending by scores[s] do
            err ← L2NORM(w, SPIKEDIST(S, s_0))
            S' ← S with s removed
            err' ← L2NORM(w, SPIKEDIST(S', s_0))
            δ ← err − err'
            scores[s] ← δ
            if δ > 0 then
                S ← S'
            end if
        end for
    until S = S_old
    return S
end function
```

In terms of time complexity, applying this algorithm autoregressively to infer a spike train L bins long, broken into N steps of l bins uses $\sim Nl^2 \log l$ compares in the worst case—this corresponds to an $l \log l$ cost of sorting that may be repeated l times each inference step, of which there are N. Except in pathological cases, the outer loop of SPIKEINFERENCE requires very few iterations—for the combined 90 minutes of spike train inference carried out on the chicken test set, no inference step required more than two iterations. Thus, with comparisons as a cost model, the algorithm's average cost is $\sim Nl \log l$: growing linearly with the number of inference steps (N) and linearithmic with the resolution

(l) of a single inference step. Figure 18 in Appendix A.12 shows inference running times on the authors' hardware. The algorithm uses additional memory that grows linearly with l.

5.2. Spike Counts From a Poisson Distribution

A Poisson distribution has a single parameter λ. A model trained with a Poisson learning objective outputs a single scalar value, y, which is used to parameterize a Poisson distribution by setting $\lambda = y$. From such a parameterized Poisson distribution, we consider three ways to infer spike counts: maximizing probability, sampling, and approximating the mean.

Maximizing probability involves selecting the spike count with maximum probability under the Poisson distribution with parameter $\lambda = y$. By nature of the Poisson distribution, this value—the mode of the distribution—is $\lfloor y \rfloor$.

Sampling a spike count from the Poisson distribution with parameter $\lambda = y$ is another approach to inferring a spike count. Instead of selecting the most likely spike count, the distribution is sampled to produce a random spike count.

Approximating the mean is an alternative whereby the model's output is rounded to produce a spike count prediction. This approach is a less principled approach; however, empirically, it outperforms sampling or using the mode. Indeed, the overall best-scoring spike trains using Poisson models were produced with this approach at a summation interval of 80 ms. It is hypothesized but not tested that the reason for the rounded mean being more effective than the mode is that the mean is always larger and so will take the value zero less often in the case where the Poisson distribution skews towards zero.

The main results are reported using both sampling and the rounded mean strategy. Inference by maximizing probability (using the mode) performs significantly worse than using the rounded mean and is reported in Appendix A.2.

5.3. Spikes From a Spike Count

The Poisson distribution gives no information other than the spike rate, so a reasonable strategy for spike placement must be chosen. In this work, given an inferred spike count n, we tile n spikes uniformly over the interval. In experiments not reported here, stochastic spike placement was also tested—by sampling n times from a uniform distribution; however, this approach performed notably worse than uniform tiling.

6. Experiment Settings

A spike distance model was compared to a family of 6 Poisson-based models, each differing by the length of their summation interval: 5 ms, 10 ms, 20 ms, 40 ms, 80 ms and 160 ms. Below we describe the dataset, models, training procedure and evaluation procedure.

6.1. Dataset

Model input was a (5, 992)–shaped array corresponding to 1 second of history: a 4-channel stimulus and a single spike train, both sampled at 992 Hz. The data came from a 15-minute recording of chicken RGCs exposed to full-field colour noise, recorded by Seifert et al. (2023). The recording contains 154 cells. Models were trained separately for each cell, so each cell can be considered a separate 15-minute dataset. The 15 minutes was split according to the ratio (7,2,1) into training, validation and test sets. Appendix A.7 describes the recording in more detail.

Our computational budget restricted how many cells we could work with—training time was considered too long for more than 60 cells (60 cells requires 500 hours of training, as described in Appendix A.12). A principled way to choose cells is to focus on cells with more spikes. The evaluation metrics that will be introduced in Section 6.4 become less informative at low spike counts—the spike train metrics need spikes to operate on, and at low spike counts, it is difficult to distinguish models from a naive model that outputs an empty spike train. We consider only cells that contain an average spike rate greater than 0.75 spikes per second in the training set. This threshold resulted in 60 of 154 cells being used. Cells that meet this criterion will typically have more than 65 spikes in the 90-second test set (the exact number depends on their spike rate in the test set). Appendix A.8 describes the construction of the dataset in more detail.

6.2. Models

For each of the 60 cells, 7 models were trained: 1 spike distance model and 6 Poisson models. Each Poisson model used a different summation interval length: 5, 10, 20, 40, 80 and 160 bins. As the duration of 1 bin (1.0008 ms) is approximately 1 ms, these models are referred to as Poisson-5 ms, Poisson-10 ms, and so on.

A single architecture is not used for both the spike distance objective and the Poisson objective, as the former requires output in the form of an array and the latter a scalar; instead, two neural networks are used. These were designed towards maximizing our ability to compare the two learning objectives. An overview of the architectures is shown in Figure 3. The two architectures were designed around a shared base architecture that accounts for the preponderance of parameters and compute. A CNN using ConvNext blocks described by Liu et al. (2022) forms the shared base. The Poisson architecture then ends in a fully-connected layer, whereas the spike distance architecture's head consists of 4 upsampling ConvNext blocks. There is no weight sharing

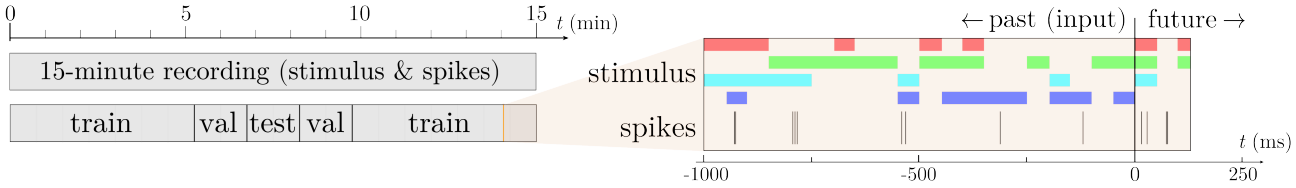


Figure 7. **Left:** A 15-minute retina recording, split into training, validation and test subsets. **Right:** A single ~1.1-second snippet taken from the training subset. $t = 0$ separates the snippet into the "past" data, available to the model, and the "future" data, for which models will be expected to make predictions. In our experiments, the length of the past data is always 1 second (992 samples); the length of the future data depends on the model; for example, the Poisson-5 ms model with a 5-sample summation interval will use 5 samples (~5 ms).

employed for training on multiple cells—the models are trained end-to-end for each cell individually. The alternative of training on all cells at once by adding per-cell modulation to the network was avoided, as the outputs of the two architectures are very different in nature, and any single modulation scheme may favour one over the other. Further details on the models are described in Appendix A.9.

One concern addressed is layer count. The spike distance architecture has additional layers in its head, and we wished to rule out this difference leading to a performance advantage. This concern was ameliorated by optimizing the number of layers in the shared base architecture to maximize Poisson performance. This architecture search over layer count is described in Appendix A.10.

6.3. Training

All models were trained for 80 epochs using the AdamW optimizer with the 1-cycle learning rate policy (Smith, 2017). Loss for the spike distance model was mean squared error between the model output and the log of the target spike distance. Loss for Poisson models was negative log-likelihood with respect to the target Poisson distribution. The λ parameter of the target distribution is the count of spikes within the summation interval. Final models were chosen based on the lowest validation loss across the 80 epochs. Training hyperparameters are described further in Appendix A.11. Total training times are summarized in Appendix A.12.

6.4. Evaluation

Once trained on a cell, a model was used to predict a spike train using the 90-second test data. Each inferred spike train was compared to the ground truth spike train using three metrics for spike train similarity/dissimilarity: Van Rossum distance (Van Rossum, 2001), Schreiber similarity (Schreiber et al., 2003) and Pearson correlation. Appendix A.5 covers these metrics in more detail. Schreiber similarity and Van Rossum distance were chosen because they are frequently used and easy to explain. Pearson correlation is reported so as to have a common metric with the work of McIntosh et al. (2016) and Gogliettino et al. (2024). We refer to Paiva et al. (2010) and Sihn and Kim (2019) for a comparison of different spike train measures.

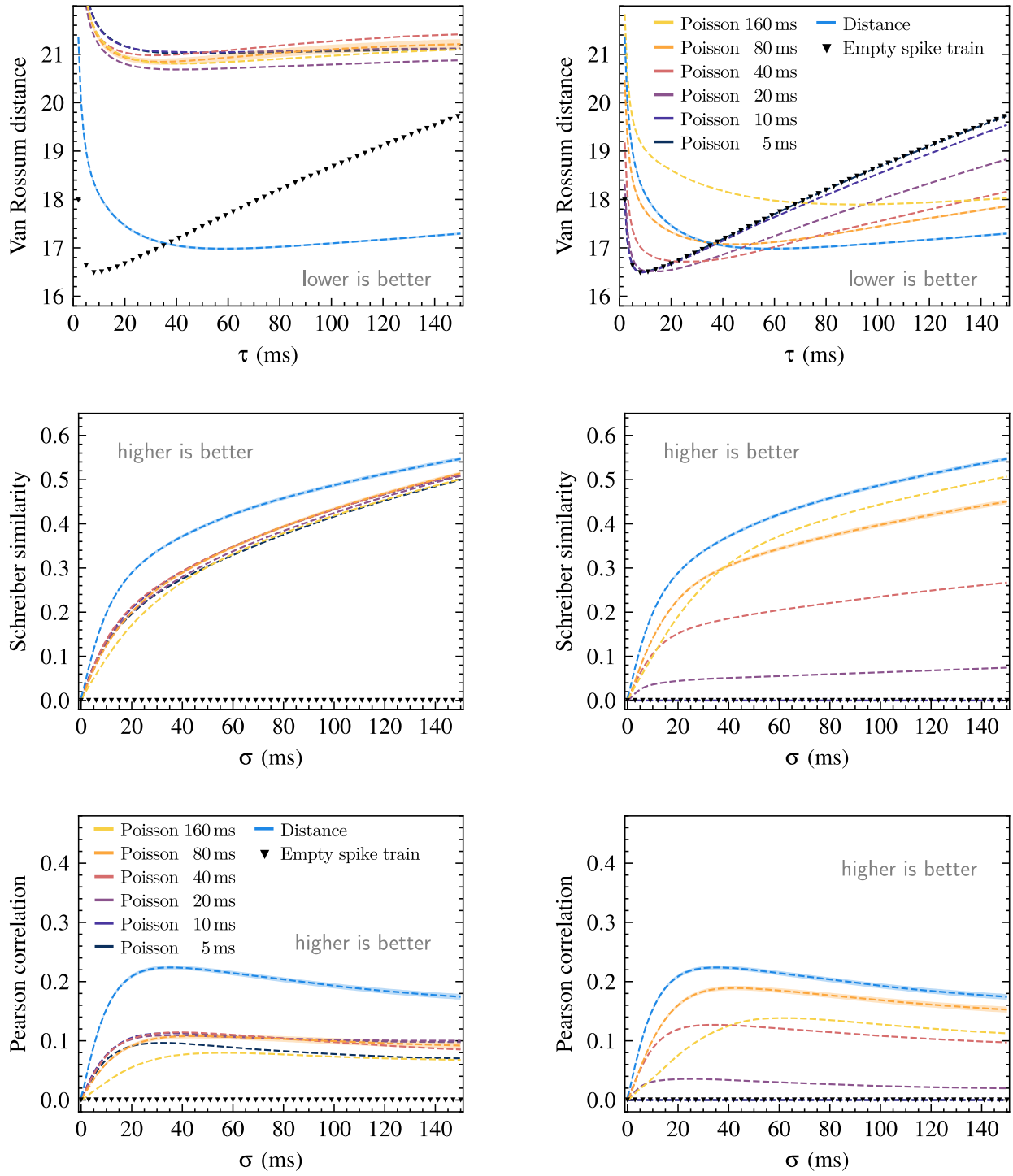
The spike train metrics are each parameterized by a smoothing parameter. For Van Rossum distance and Schreiber similarity, their parameters—τ and σ respectively—are integral to their definition. We augment the Pearson correlation by prepending a smoothing step copied from the Schreiber similarity. The comparison of spike trains is sensitive to the degree of smoothing, and different models perform best at different smoothing "sweet spots". There is a scale from instantaneous spikes to average spike rate. No single point on this scale can capture all that matters about a neuron's response, so it is important to investigate spiking behaviour over a range of timescales. We evaluate the three metrics at a range of smoothing values, from 0 to 150.

7. Results

Training and evaluating a model 60 times, once for each cell, is considered a run. 11 runs were performed for both the spike distance model and the Poisson-80 ms model (the most competitive model). All other models were evaluated for a single run. We follow the recommendation of Agarwal et al. (2021) and report aggregate performance for a model using interquartile mean (IQM) across cells and runs; and for the two models with multiple runs, uncertainty is estimated with stratified bootstrap confidence intervals.

Figure 8 records the three metrics for all models. Both sampling (Figure 8a) and rounded mean (Figure 8b) are used for inference with the Poisson models. An additional perspective on the variability across cells is provided by Figure 9 (Appendix A.1) which zooms in on a single smoothing parameter. The results when the Poisson models use inference via the mode are presented in Figure 10 (Appendix A.2). For a qualitative comparison of inference strategies for the Poisson models, the snippet from Figure 2 is repeated in Appendix A.3 for inference using sampling (Figure 11) and the distribution mode (Figure 12).

The experiment was repeated for a recording of frog RGCs, and the results for this data are presented in Appendix A.4.



(a) Poisson models use inference via sampling

(b) Poisson models use inference via the rounded mean

Figure 8. Performance comparison between the spike distance model and the Poisson models at spike prediction. For the Poisson models, two spike count inference strategies are considered: **(left)** sampling and **(right)** using the rounded mean. Three metrics are considered: **(top)** Van Rossum distance, **(middle)** Schreiber similarity and **(bottom)** Pearson correlation between the ground truth and output spike trains, reported as interquartile mean over the 60 chicken cells. The metrics are evaluated for a range of their smoothing parameters, (τ and σ), from 0 to 150. 95% confidence intervals are included for the Poisson-80 ms and spike distance models. Scores for the zero spike spike train are included for comparison, highlighting that the metrics become less effective at comparing models at low smoothing levels—in particular, the Van Rossum distance eventually prefers the zero spike train for sufficiently low τ values. See Figure 9 in Appendix A.1 for a visualization of the variability across cells at a specific smoothing value, $\tau = \sigma = 60$.

8. Discussion

An effective model should perform well over a wide range of the metrics' smoothing parameters, and in this regard, the spike distance model outperforms the Poisson models in all three metrics.

For the Poisson models, the spike inference strategy has a major effect on the quality of the spike trains produced. When using inference via sampling (Figure 8a), all Poisson models perform similarly but are outperformed by the spike distance model, and considerably so in terms of Van Rossum distance. Inference via the rounded mean allows a more deterministic placement of spikes; however, as the summation interval is reduced, models begin to approach the performance of the zero spike train model due to model outputs being rounded to zero. Eventually, the model with the shortest summation interval, Poisson 5 ms, has performance indistinguishable from that of the zero spike train. When inference uses the distribution's mode, (Figure 10 in Appendix A.2) this effect is exacerbated, and both Poisson 20 ms and Poisson 10 ms also have performance indistinguishable from that of the zero spike train.

The Poisson 80 ms model using inference via rounded mean is the overall best-performing Poisson model. It can be understood to sit in a sweet spot where the summation interval is long enough to be less affected by the rounding to zero effect but short enough to have good temporal resolution. This result suggests that the less principled approach of inferring spike counts by rounding the model output should be considered when choosing a spike count inference strategy. If one is unable to experiment with various summation intervals, using sampling can give a higher lower-bound performance across summation intervals at the cost of incurring a lower peak performance.

The evaluation settings that least favour the distance model are the Van Rossum distances at low smoothing levels, where the Van Rossum distance prefers Poisson models with short summation intervals using the mode or rounded mean for inference. This is explained by these models predicting very few spikes and having a performance similar to the empty spike train model. For Van Rossum distance, there will always be a τ value below which predicting zero spikes performs best. At low smoothing values, inexact spikes are penalized twice—once for missing the correct spike and once for a spike at an incorrect time. Paiva et al. (2010, p. 408) describes this as Van Rossum distance acting as a co-incidence detector at low τ values. When comparing models via the Van Rossum distance, the τ values where the empty spike train performs best are less informative, and above these τ values, it is the spike distance model that is most competitive.

9. Limitations and Improvements

The results supporting the use of the spike distance objective are limited by the scope of the experiment. The two learning objectives were compared in the context of a neural network model with mostly convolutional layers, and how the properties of this architecture, such as the translation invariance of the convolutional layers, support or undermine each learning objective is an interesting question that is not investigated in this work. How the differences between the model outputs interact with hyperparameters or affect regularization are other considerations that are not investigated.

A strength of the Poisson objective that remains undisputed is interpretability—GLMs trained with a Poisson objective can be easily probed to obtain interpretable stimulus filters (Weber and Pillow, 2017). It is unclear whether models with a spike distance output would be as useful in this regard. It is only in the context of predicting precise spike times that this work argues for the use of the spike distance objective.

There are opportunities to increase the competitiveness of the spike distance approach. The model architecture can be optimized with spike distances in mind (the architecture used in this work was optimized for the performance of the Poisson models). As some aspects of generating spike distance arrays do not need real spike data to be learned, pretraining the network using synthetic data may be effective. As for the real data—it can be better utilized by training a single model on multiple cells, which was avoided in this work in order to improve our ability to compare models. Finally, the space of algorithms for inferring spikes from spike distance arrays has yet to be explored deeply. Algorithm 1 demonstrates effectiveness in practice; however, it is an approximate solution, and to what extent an optimal solution improves spike prediction is an interesting question.

It is also interesting to ask: what other representations can create effective learning objectives? We experimented with other representations such as a Van Rossum-like representation and a representation that tracked separately the time until and time since a spike. With these representations, we observed model outputs that were more difficult to infer spikes with, and we did not pursue these past early experiments. Such representations are worth further investigation.

10. Conclusion

We propose using a "distance to spike" representation of a spike train which we call a *spike distance function*. We show that neural networks can be trained to approximate spike distance functions and that these outputs can be used to predict spikes. For the task of spike train prediction, we show that models trained to approximate a spike distance function outperform models trained using the standard Poisson objective.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

Acknowledgements

KD would like to acknowledge scholarship support from the Leverhulme Trust (DS-2020-065). CY acknowledges funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement no: 101026409. TP acknowledges funding from the Wellcome Trust (Investigator Award in Science 220277/Z20/Z), the European Research Council (ERC-StG "NeuroVisEco" 677687), URKI (BBSRC, BB/R014817/1, BB/W013509/1 and BB/X020053/1), the Leverhulme Trust (PLP-2017-005, RPG-2021-026 and RPG-2-23-042) and the Lister Institute for Preventive Medicine. This research was funded in whole, or in part, by the Wellcome Trust [220277/Z20/Z]. For the purpose of open access, the author has applied a CC BY public copyright licence to any Author Accepted Manuscript version arising from this submission.

References

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C Courville, and Marc Bellemare. Deep Reinforcement Learning at the Edge of the Statistical Precipice. In *Advances in Neural Information Processing Systems*, volume 34, pages 29304–29320. Curran Associates, Inc., 2021.

Eleanor Batty, Josh Merel, Nora Brackbill, Alexander Heitman, Alexander Sher, Alan Litke, E. J. Chichilnisky, and Liam Paninski. Multilayer Recurrent Network Models of Primate Retinal Ganglion Cell Responses. In *International Conference on Learning Representations*, November 2016.

Michael J. Berry and Markus Meister. Refractoriness and Neural Precision. *Journal of Neuroscience*, 18(6):2200–2211, March 1998. ISSN 0270-6474, 1529-2401. doi: 10.1523/JNEUROSCI.18-06-02200.1998.

Michael J. Berry, David K. Warland, and Markus Meister. The structure and precision of retinal spike trains. *Proceedings of the National Academy of Sciences*, 94(10): 5411–5416, May 1997. doi: 10.1073/pnas.94.10.5411.

Eleonora Borda and Diego Ghezzi. Advances in visual prostheses: Engineering and biological challenges. *Progress in Biomedical Engineering*, 4(3):032003, August 2022. ISSN 2516-1091. doi: 10.1088/2516-1091/ac812c.

Tanguy Bosser and Souhaib Ben Taieb. On the Predictive Accuracy of Neural Temporal Point Process Models for Continuous-time Event Data. *Transactions on Machine Learning Research*, March 2023. ISSN 2835-8856.

Romain Brette. Philosophy of the Spike: Rate-Based vs. Spike-Based Theories of the Brain. *Frontiers in Systems Neuroscience*, 9, 2015. ISSN 1662-5137.

Daniel A. Butts, Chong Weng, Jianzhong Jin, Jose-Manuel Alonso, and Liam Paninski. Temporal precision in the visual pathway through the interplay of excitation and stimulus-driven suppression. *The Journal of Neuroscience: The Official Journal of the Society for Neuroscience*, 31(31):11313–11327, August 2011. ISSN 1529-2401. doi: 10.1523/JNEUROSCI.0434-11.2011.

Santiago A. Cadena, George H. Denfield, Edgar Y. Walker, Leon A. Gatys, Andreas S. Tolias, Matthias Bethge, and Alexander S. Ecker. Deep convolutional models improve predictions of macaque V1 responses to natural images. *PLOS Computational Biology*, 15(4):1–27, April 2019. doi: 10.1371/journal.pcbi.1006897.

Stuart Coles. *An Introduction to Statistical Modeling of Extreme Values*. Springer Series in Statistics. Springer, London, 2001. ISBN 978-1-84996-874-4 978-1-4471-3675-0. doi: 10.1007/978-1-4471-3675-0.

Angela Dai, Charles Ruizhongtai Qi, and Matthias NieBner. Shape Completion Using 3D-Encoder-Predictor CNNs and Shape Synthesis. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6545–6554, Honolulu, HI, July 2017. IEEE. ISBN 978-1-5386-0457-1. doi: 10.1109/CVPR.2017.693.

Alex R. Gogliettino, Sam Cooler, Ramandeep S. Vilkhu, Nora J. Brackbill, Colleen Rhoades, Eric G. Wu, Alexandra Kling, Alexander Sher, Alan M. Litke, and E.J. Chichilnisky. Modeling responses of macaque and human retinal ganglion cells to natural images using a convolutional neural network, March 2024.

Tim Gollisch and Markus Meister. Rapid Neural Coding in the Retina with Relative Spike Latencies. *Science*, 319 (5866):1108–1111, February 2008. doi: 10.1126/science.1149639.

Alan G. Hawkes. Spectra of Some Self-Exciting and Mutually Exciting Point Processes. *Biometrika*, 58(1):83–90, 1971. ISSN 0006-3444. doi: 10.2307/2334319.

Jeremy Howard, Sylvain Gugger, and Soumith Chintala. *Deep Learning for Coders with Fastai and PyTorch: AI Applications without a PhD*. O'Reilly Media, Inc, Sebastopol, California, first edition edition, 2020. ISBN 978-1-4920-4552-6.

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, May 2001. ISSN 0896-6273. doi: 10.1016/S0896-6273(01)00322-1.

David Liu and Mate Lengyel. Bayesian nonparametric (non-)renewal processes for analyzing neural spike train variability. *Advances in Neural Information Processing Systems*, 36:68013–68027, December 2023.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A ConvNet for the 2020s. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11966–11976, June 2022. doi: 10.1109/CVPR52688.2022.01167.

Lane McIntosh, Niru Maheswaranathan, Aran Nayebi, Surya Ganguli, and Stephen Baccus. Deep Learning Models of the Retinal Response to Natural Scenes. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

Stanley Osher and Ronald Fedkiw. *Level Set Methods and Dynamic Implicit Surfaces*, volume 153 of *Applied Mathematical Sciences*. Springer, New York, NY, 2003. ISBN 978-1-4684-9251-4 978-0-387-22746-7. doi: 10.1007/b98879.

Antonio R. C. Paiva, Il Memming Park, and Jose C. Principe. A comparison of binless spike train measures. *Neural Computing and Applications*, 19(3):405–419, April 2010. doi: 10.1007/s00521-009-0307-6.

Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 165–174, June 2019. doi: 10.1109/CVPR.2019.00025.

Christopher X. Ren, Claudia Hulbert, Paul A. Johnson, and Bertrand Rouet-Leduc. Chapter Two - Machine learning and fault rupture: A review. In Ben Moseley and Lion Krischer, editors, *Advances in Geophysics*, volume 61 of *Machine Learning in Geosciences*, pages 57–107. Elsevier, January 2020. doi: 10.1016/bs.agph.2020.08.003.

Bertrand Rouet-Leduc, Claudia Hulbert, Nicholas Lubbers, Kipton Barros, Colin J. Humphreys, and Paul A. Johnson. Machine Learning Predicts Laboratory Earthquakes. *Geophysical Research Letters*, 44(18):9276–9282, 2017. ISSN 1944-8007. doi: 10.1002/2017GL074677.

José-Alain Sahel, Elise Boulanger-Scemama, Chloé Pagot, Angelo Arleo, Francesco Galluppi, Joseph N. Martel, Simona Degli Esposti, Alexandre Delaux, Jean-Baptiste de Saint Aubert, Caroline de Montleau, Emmanuel Gutman, Isabelle Audo, Jens Duebel, Serge Picaud, Deniz Dalkara, Laure Blouin, Magali Taiel, and Botond Roska. Partial recovery of visual function in a blind patient after optogenetic therapy. *Nature Medicine*, 27(7):1223–1229, July 2021. ISSN 1546-170X. doi: 10.1038/s41591-021-01351-4.

Susanne Schreiber, Susanne Schreiber, Jean Marc Fellous, Diane Whitmer, Paul H. E. Tiesinga, and Terrence J. Sejnowski. A new correlation-based measure of spike timing reliability. *Neurocomputing*, 52:925–931, June 2003. doi: 10.1016/s0925-2312(02)00838-x.

Marvin Seifert, Paul A. Roberts, George Kafetzis, Daniel Osorio, and Tom Baden. Birds multiplex spectral and temporal visual information via retinal On- and Off-channels. *Nature Communications*, 14(1):5308, August 2023. ISSN 2041-1723. doi: 10.1038/s41467-023-41032-z.

Oleksandr Shchur, Ali Caner Türkmen, Tim Januschowski, and Stephan Günnemann. Neural Temporal Point Processes: A Review. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pages 4585–4593, Montreal, Canada, August 2021. International Joint Conferences on Artificial Intelligence Organization. ISBN 978-0-9992411-9-6. doi: 10.24963/ijcai.2021/623.

Duho Sihn and Sung-Phil Kim. A Spike Train Distance Robust to Firing Rate Changes Based on the Earth Mover's Distance. *Frontiers in Computational Neuroscience*, 13:82–82, January 2019. doi: 10.3389/fncom.2019.00082.

Vincent Sitzmann, Semon Rezchikov, Bill Freeman, Josh Tenenbaum, and Fredo Durand. Light Field Networks: Neural Scene Representations with Single-Evaluation Rendering. In *Advances in Neural Information Processing Systems*, volume 34, pages 19313–19325. Curran Associates, Inc., 2021.

Leslie N. Smith. Cyclical Learning Rates for Training Neural Networks. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 464–472, March 2017. doi: 10.1109/WACV.2017.58.

Leslie N. Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates. In Tien Pham, editor, *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications*, page 36, Baltimore, United States, May 2019. SPIE. ISBN 978-1-5106-2677-5 978-1-5106-2678-2. doi: 10.1117/12.2520589.

Yee Teh and Vinayak Rao. Gaussian process modulated renewal processes. In *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011.

V. J. Uzzell and E. J. Chichilnisky. Precision of Spike Trains in Primate Retinal Ganglion Cells. *Journal of Neurophysiology*, 92(2):780–789, August 2004. ISSN 0022-3077. doi: 10.1152/jn.01171.2003.

M. C. W. Van Rossum. A Novel Spike Distance. *Neural Computation*, 13(4):751–763, April 2001. doi: 10.1162/089976601300014321.

Kun Wang, Christopher W. Johnson, Kane C. Bennett, and Paul A. Johnson. Predicting Future Laboratory Fault Friction Through Deep Learning Transformer Models. *Geophysical Research Letters*, 49(19):e2022GL098233, October 2022. ISSN 0094-8276, 1944-8007. doi: 10.1029/2022GL098233.

Alison I. Weber and Jonathan W. Pillow. Capturing the Dynamical Repertoire of Single Neurons with Generalized Linear Models. *Neural Computation*, 29(12): 3260–3289, December 2017. ISSN 0899-7667. doi: 10.1162/neco_a_01021.

Sanghyun Woo, Shoubhik Debnath, Ronghang Hu, Xinlei Chen, Zhuang Liu, In So Kweon, and Saining Xie. ConvNeXt V2: Co-Designing and Scaling ConvNets With Masked Autoencoders. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16133–16142, 2023.

Andy Zeng, Shuran Song, Matthias Nießner, Matthew Fisher, Jianxiong Xiao, and Thomas Funkhouser. 3DMatch: Learning Local Geometric Descriptors from RGB-D Reconstructions. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 199–208, July 2017. doi: 10.1109/CVPR.2017.29.

A. Appendix

Appendices A.1-A.4 present supplementary results. Appendix A.5 describes the evaluation metrics in more detail. Appendix A.6 argues for the suitability of our evaluation methodology. Details of the MEA recording and how it is processed to form a dataset are presented in Appendices A.7 and A.8. Appendices A.9, A.10 and A.11 cover the models used. Computational resources used are listed in Appendix A.12. Appendices A.13 and A.14 go into more detail on working with spike distances.

A.1. Variability Across Cells

Figure 9 explores the variability of the data by zooming in on a single smoothing parameter value (the x-axis value $\tau = \sigma = 60$) of the results in Figure 8. The purpose of this figure is to give readers a better sense of the spread of the data that is being condensed in Figure 8.

A.2. Inference by Maximizing Probability

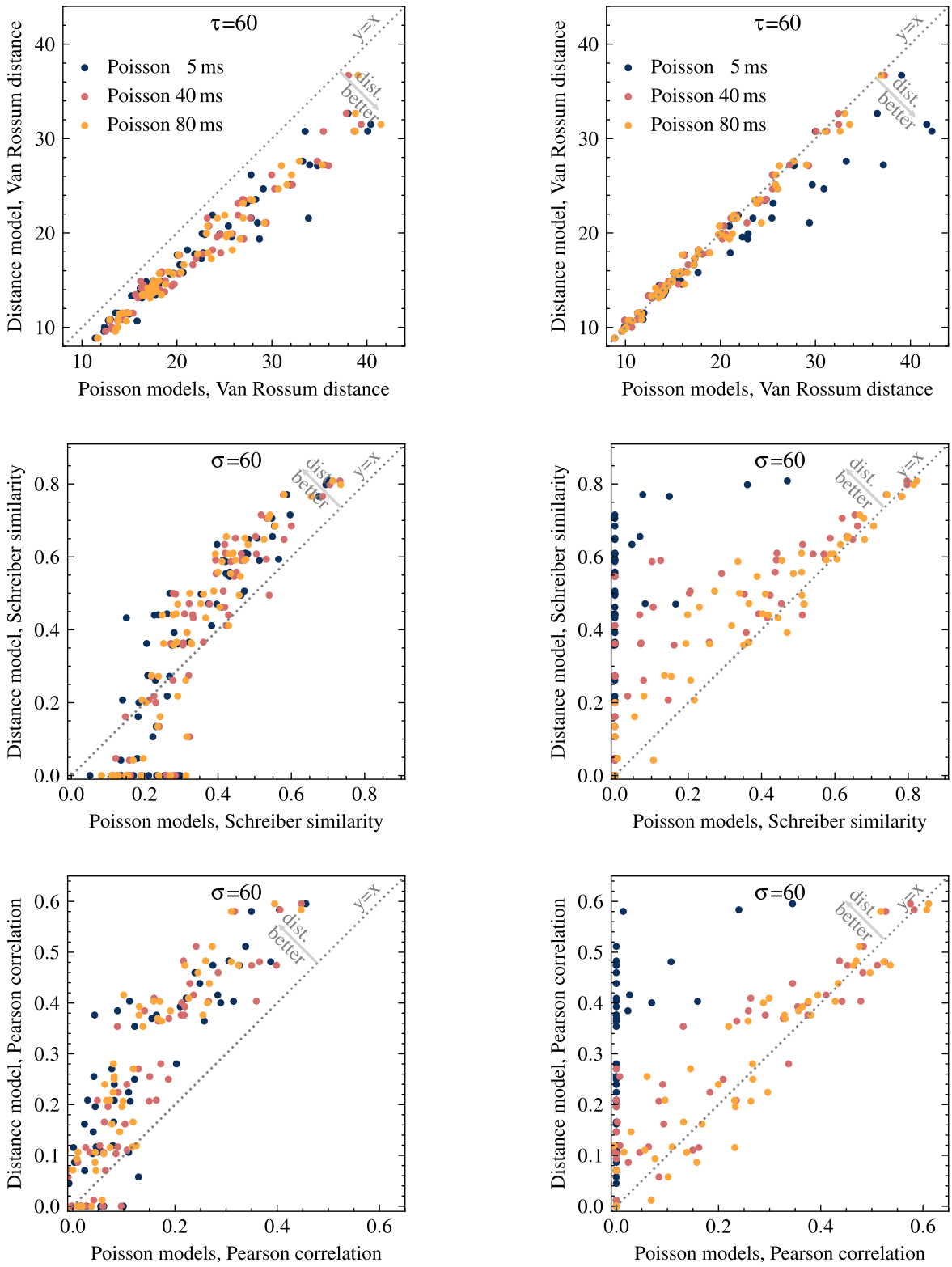
During inference, using the mode of the Poisson distribution (the floor of a Poisson model's output) is a principled approach to selecting the spike count with the highest probability. Using this strategy performs significantly worse than using the rounded mean of the distribution (the rounded model's output). Figure 10 complements Figure 8a and Figure 8b by reporting the performance of the Poisson models when using the mode for inference.

A.3. Input-Output Snippets for Inference Using Sampling and Using the Mode

An input-output snippet comparing the spike distance model to the Poisson models was shown in Figure 2. The Poisson models in this figure used the rounded model outputs for spike count inference. We repeat this figure but with Poisson models using inference via sampling (Figure 11) and inference via the mode (Figure 12).

A.4. Xenopus (Frog) RGCs

Xenopus (frog) RGCs were recorded under similar conditions to those of the chicken RGCs. The same MEA device and recording process was used. One difference is that, unlike the chicken dataset for which Seifert et al. (2023) applied a quality criterion to filter out poorly responding cells, no such filtering was applied to the frog data. The same average spike rate threshold of 0.75 spikes per second in the training set was applied, and this filtering resulted in 113 cells. Figure 13 shows the Van Rossum distance, Schreiber similarity and Pearson correlation for 113 recorded frog RGCs. All models were trained for a single run only (no confidence intervals). The gap in performance between the spike distance model and the Poisson models is narrower for the frog RGCs in comparison to the chicken RGCs (Figure 8); however, the spike distance model is still the most consistently competitive model across the three metrics.



(a) Poisson models use inference via sampling

(b) Poisson models use inference via the rounded mean

Figure 9. Zoomed in to a single smoothing value from Figure 8, the spike distance model is compared to 3 of the Poisson models for all 60 cells. The left and right columns of this figure correspond to the left and right columns of Figure 8: **(left)** Poisson models use sampling for inference, **(right)** Poisson models use the rounded mean for inference. All three metrics are presented: **(top)** Van Rossum distance at $\tau = 60\,\mathrm{ms}$, **(middle)** Schreiber similarity at $\sigma = 60\,\mathrm{ms}$, and **(bottom)** Pearson correlation at $\sigma = 60\,\mathrm{ms}$.

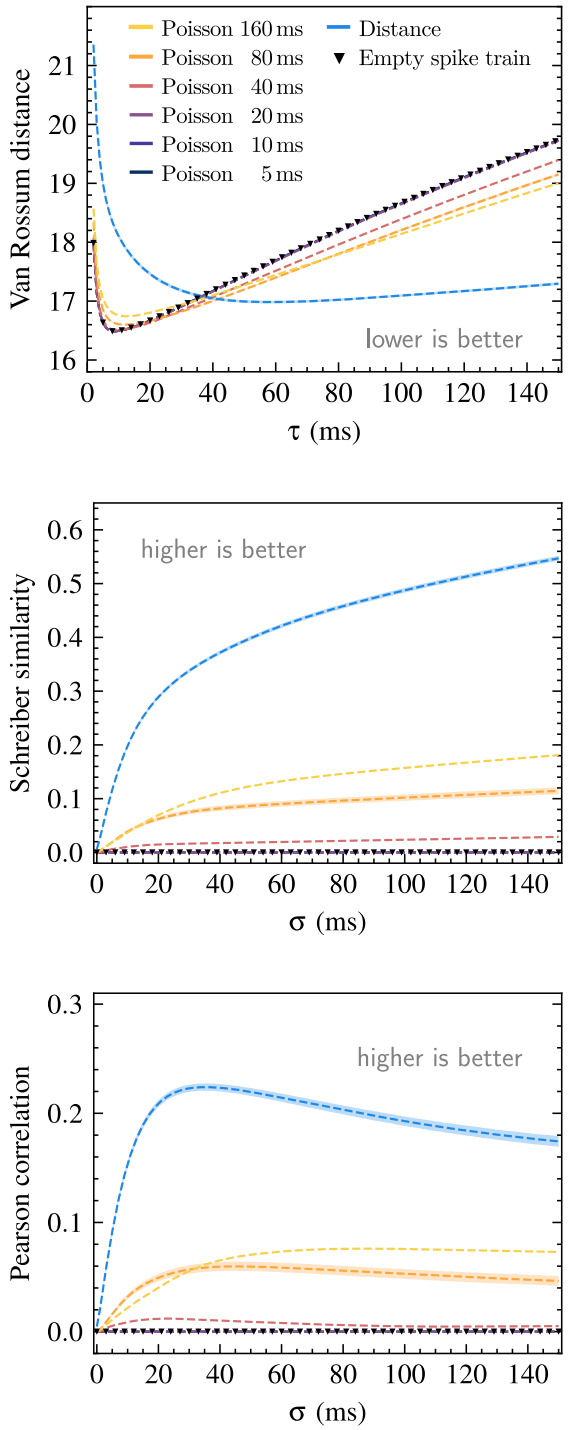


Figure 10. Model comparison when the mode of the Poisson distribution is used for spike prediction with the Poisson models. Van Rossum distance **(top)**, Schreiber similarity **(middle)** and Pearson correlation **(bottom)** between the ground truth and output spike trains, reported as interquartile mean over the 60 chicken cells. All metrics are evaluated for a range of smoothing parameters. 95% confidence intervals are included for the Poisson-80 ms and spike distance models.

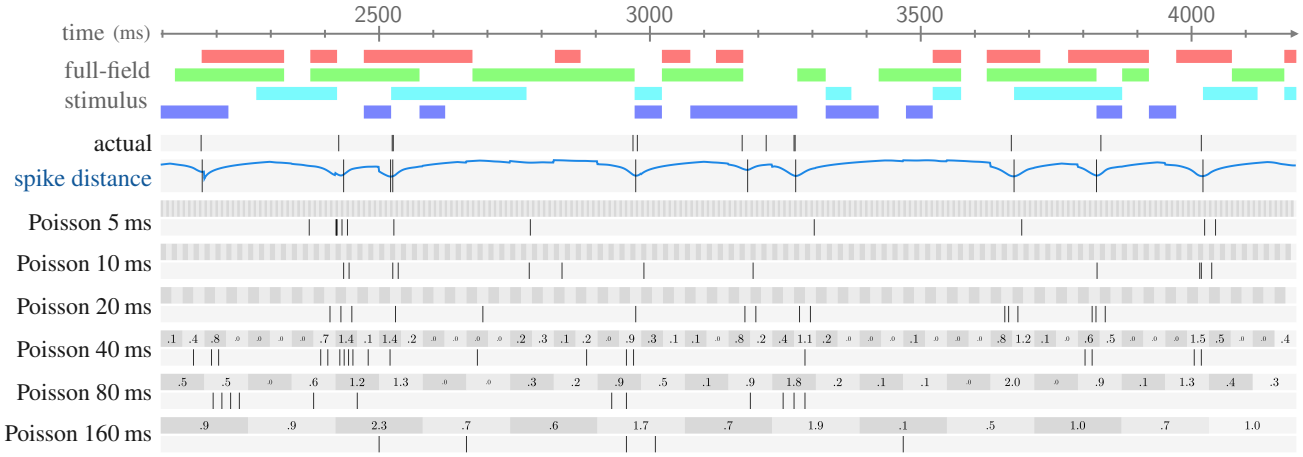


Figure 11. **Inference via sampling, for Poisson models.** The same input-output snippet from Figure 2 is repeated here, but with spike count inference via sampling being used for the Poisson models. The boxed numbers represent the neural network outputs, rounded to two decimal places (rounded to fit in the figure). When using sampling, a spike count quite different from a model's output can become the prediction. This is beneficial for short summation intervals, where outputs might otherwise be rounded or floored to zero, but with longer intervals, the model's output may be an accurate prediction that then gets perturbed by the sampling process. Note: due to limited space in the figure, the models with summation interval below 40 ms have model output values omitted.

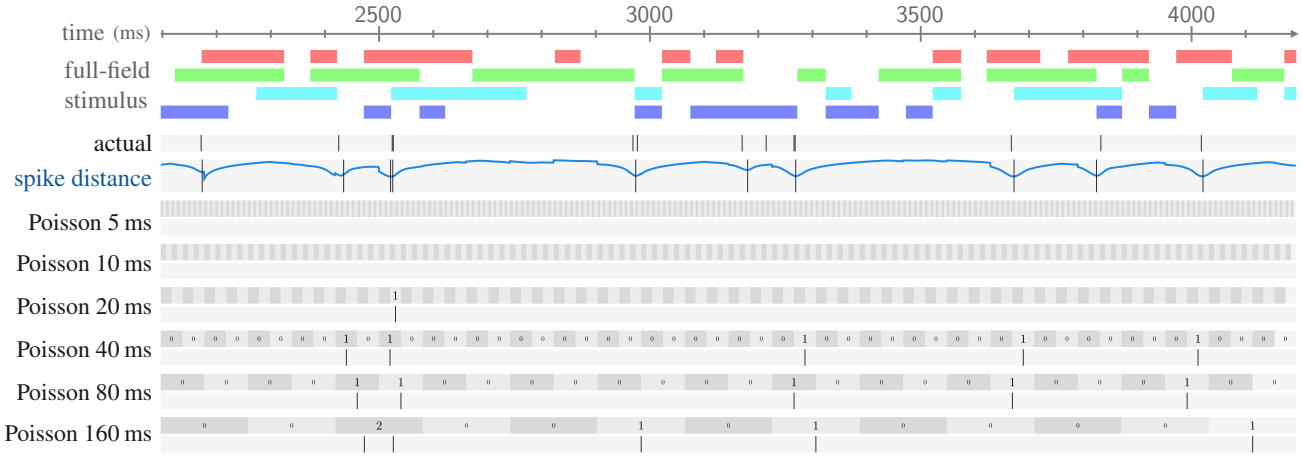
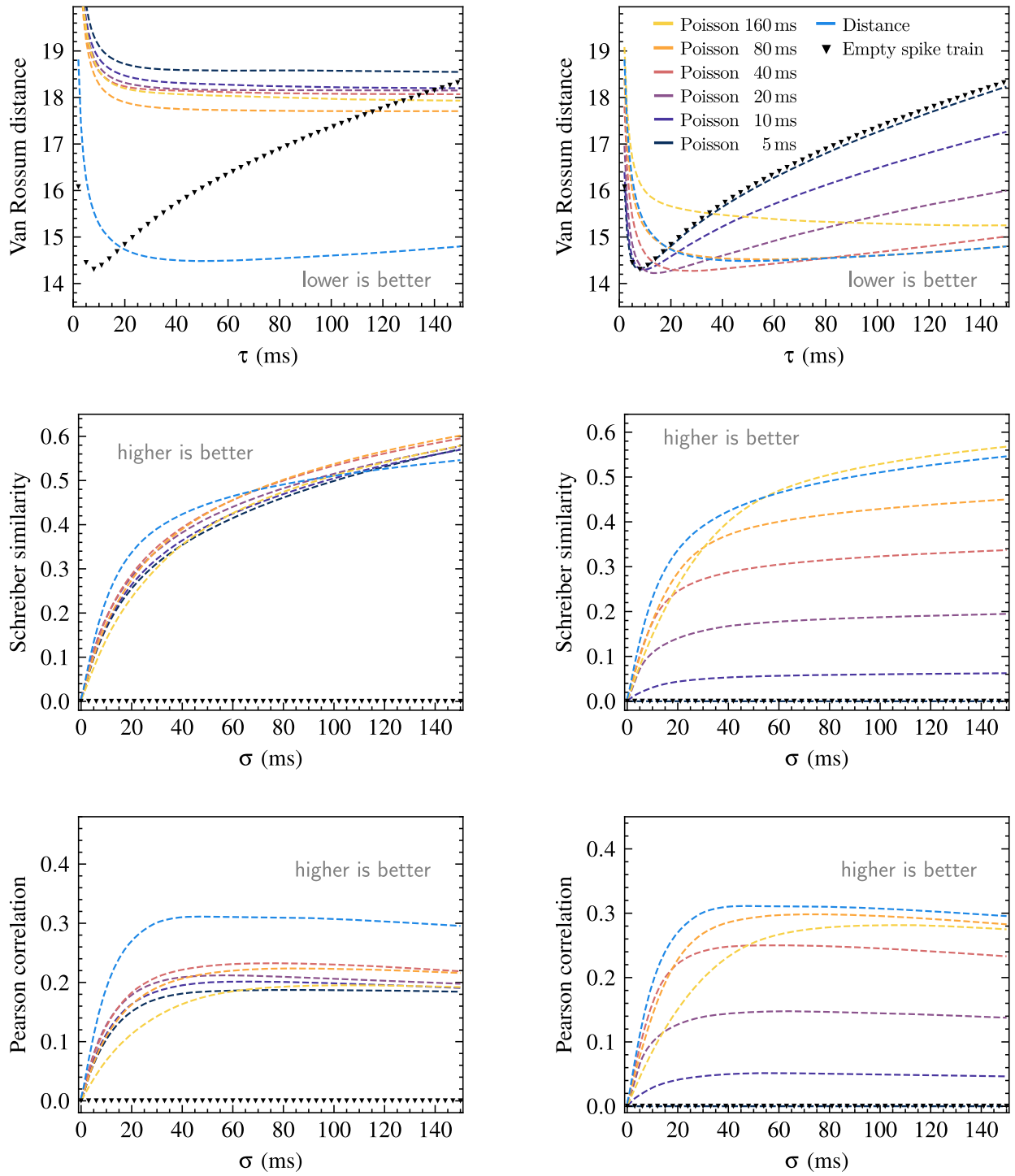


Figure 12. **Inference via the distribution mode, for Poisson models.** The same input-output snippet from Figure 2 (and Figure 11) is repeated here, but with Poisson models using the distribution's mode for spike count inference. The boxed numbers represent the floored neural network outputs. The mode of the Poisson distribution is the floor of the model's output. For a preponderance of outputs, the model's output is floored to zero. Note: due to limited space in the figure, the models with summation interval below 20 ms have their model output values omitted; predicted spikes are still shown, although there are no predicted spikes to show for the Poisson 10 ms and Poisson 5 ms models.



(a) Poisson models use inference via sampling

(b) Poisson models use inference via the rounded mean

Figure 13. Frog RGCs. Performance comparison between the spike distance model and the Poisson models at spike prediction. For the Poisson models, two spike count inference strategies are considered: **(left)** sampling and **(right)** using the rounded mean. Three metrics are considered: **(top)** Van Rossum distance, **(middle)** Schreiber similarity and **(bottom)** Pearson correlation between the ground truth and output spike trains, reported as interquartile mean over the 113 frog RGCs. All models are trained for a single run (no confidence intervals). Metrics are evaluated for a range of their smoothing parameters, (τ and σ), from 0 to 150. Scores for the zero spike train are included for comparison, highlighting that the metrics become less effective at comparing models at low smoothing levels—in particular, the Van Rossum distance eventually prefers the zero spike train for sufficiently low τ values.

A.5. Evaluation Metrics

This section describes the three spike train metrics used for evaluation. Figure 14 describes Van Rossum distance and Schreiber similarity. Pearson correlation is the well-known normalized covariance.

Choosing a spike train metric remains an unsatisfying activity as there is not yet any strong basis on which the characteristics of different metrics can be definitively ranked. As mentioned in Section 6.4, Schreiber similarity and Van Rossum distance were chosen as they are frequently used and easy to explain. Pearson correlation is reported so as to have a common metric with the work of McIntosh et al. (2016) and Goglettino et al. (2024). We refer to Paiva et al. (2010) and Sihn and Kim (2019) for a comparison of different spike train measures.

Each of the three spike train metrics used in this work is parameterized by a smoothing parameter. For Van Rossum distance and Schreiber similarity, their parameters are an integral component of their definition. We augment the Pearson correlation by prepending a smoothing step copied from the Schreiber similarity. For Schreiber similarity and Pearson correlation, spike trains are first smoothed with a Gaussian kernel with standard deviation parameter σ; for Van Rossum distance, the first step is to smooth spike trains with an exponential kernel with decay parameter τ.

An important distinction between Van Rossum distance and Schreiber similarity is their sensitivity to the overall number of spikes. Van Rossum distance lacks any normalization to spike count—it cares about both where spikes are placed and how many. In comparison, Schreiber similarity includes a normalization to the number of spikes present in both spike trains. This distinction between the metrics explains how some models can perform well in terms of one metric and poorly in terms of the other. To illustrate this, consider an example where a cell's average spike rate is known, but there is no information on when spikes occur. Two prediction strategies are: predict zero spikes or predict a constant rate of spikes equal to the known average rate. Van Rossum distance will score the first strategy more favourably than the second, while it is the reverse for Schreiber similarity. The Van Rossum distance will penalize the many incorrectly timed spikes predicted by the second strategy. Under Schreiber similarity, the first strategy will necessarily score zero while the second strategy scores positively for having some positive projection onto the ground truth spike train. This distinction explains how, if the preponderance of a Poisson model's outputs are close to zero, switching to sampling for spike count inference can lead to better performance in terms of Schreiber similarity and worse performance in terms of Van Rossum distance. This illustrates the benefit of using both Van Rossum distance and Schreiber similarity for evaluation: they give different perspectives on the quality of a predicted spike train.

A.6. Evaluation Methodology

In this section, we justify our departure from other evaluation practices.

This work's approach to evaluation is to consider the quality of generated spike trains. Our evaluation departs from previous work by avoiding binning, avoiding a single fixed smoothing, and avoiding multi-trial averaging. In Batty et al. (2016), McIntosh et al. (2016) and Cadena et al. (2019), ground truth spikes are *binned* then Gaussian *smoothed* and compared to Gaussian *smoothed* model outputs. Furthermore, ground truths are *averaged* over multiple trials in McIntosh et al. (2016) and *normalized* by the average over multiple trials in Batty et al. (2016) and Cadena et al. (2019). In each of these studies, the choice of bin size and smoothing filter width were *fixed* and chosen to match the bin size and smoothing used during training. A downside of our approach is that, by choosing spike trains as the inputs to comparison, we must first convert Poisson model outputs into spike predictions, and this introduces the extra consideration of which spike inference strategy works best.

We also eschew averaging or normalizing across repetitions of the same stimulus. As argued by Brette (2015), neuron state is not stable, and differences across stimulus presentations may arise not solely from noise to be disregarded but from the fact that the state of a neuron or network of neurons can change. With this in mind, we present a single stimulus pattern without repetition and delegate the modelling of uncertainty and identification of latent variables to the neural network.

Finally, we argue against the use of negative log-likelihood (NLL) as an effective evaluation metric. Likelihood-based metrics are common in neural temporal point process literature. In their review of neural temporal point processes, Shchur et al. (2021) argue against the prevalent usage of NLL, saying that "NLL is mostly irrelevant as a measure of error in real-world applications" and claim that the lack of investigations focusing on domain-specific metrics is a major gap in current neural temporal point process literature. In our work, NLL as a metric would not allow the direct comparison of Poisson models with different summation intervals, nor would it provide a way to measure the performance of the spike distance model, which does not output probability distributions. Our results demonstrate the benefits that can be achieved by working outside the probabilistic setting.

A.7. Retina Recording

The main dataset is created from a single 15-minute recording of chicken RGC spike activity carried out by Seifert

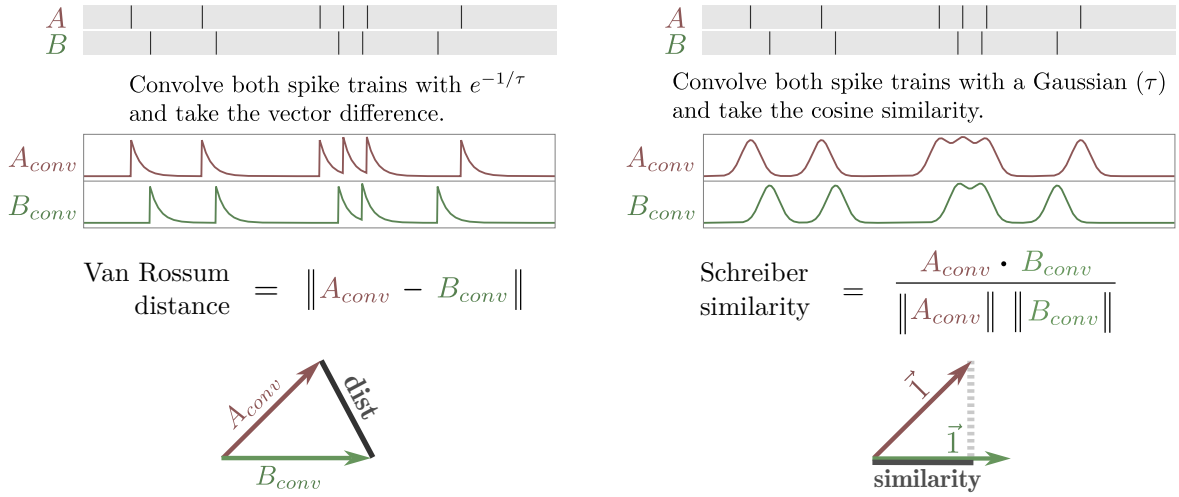


Figure 14. Calculation of Van Rossum distance (**left**) and Schreiber similarity (**right**) from two input spike trains. Both metrics employ distinct smoothing kernels to produce an intermediate vector. Van Rossum distance can be thought of as measuring the Euclidean distance between the vectors, while the Schreiber similarity assesses the angle between them.

et al. (2023). Important aspects of the recording will be summarized here (see Seifert et al. (2023) for full details). A chicken retina was placed on an MEA, and a full-field colour sequence driven by 4-LEDs was projected onto the retina. The colour sequence was a random sequence of 50 ms frames, with each LED having a 50% chance of being on or off in each frame. The electrical activity recorded by the MEA was passed through a spike sorter to produce a sequence of spike events sampled at 17.9 kHz. The stimulus frame changes were also recorded as trigger events at 17.9 kHz. At this point, quality criteria were used to filter out cells which were considered to respond poorly. The stimulus was downsampled by a factor of 18 to 992 Hz, and the spike data was rebinned to 992 Hz. Keeping correspondence between the sample period of the stimulus and spike data allows both to be stacked into a single 2D array. Downsampling by 18 was chosen as the sample period of ~1 ms is short enough that no two spikes are recorded in the same bin, which simplifies the spike distance calculation. Downsampling by 18 is also convenient as it results in a sample period very close to 1 ms (1.0008 ms), enabling number of samples and number of milliseconds to be used interchangeably with little effect on precision.

The same procedure was followed to collect the frog MEA recording, except for one difference: no cell filtering based on quality criteria was carried out.

The spike trains produced by the spike sorter are not a perfect representation of the true spike activity produced by the neurons—false positives, false negatives and the incorrect assignment of spikes to cells are all issues. In this work, we consider the neurons, the multi-electrode array and the spike sorter as a single black box whose behaviour we predict; we do not attempt to make claims about any of the three components individually.

Further processing of the data was carried out in order to create separate training, validation and test sets. This is described next.

A.8. Dataset

The 15-minute recording was split according to the ratio (7, 2, 1) into training, validation and test sets as shown previously in Figure 7. As the health of the retina can degrade over the course of the recording altering its behaviour, the (7, 2, 1) split was formed by first splitting the recording by the ratio (7, 2, 2, 2, 7) and then combining segments: the first and last 5.25-minute segments form the training set, the single middle 90-second segment forms the test set, and the two remaining 90-second segments form the validation set. This approach allows the test set to be exposed to both extremes while keeping the test set as a single contiguous chunk.

For the Poisson models, each dataset sample is a tuple (input, target), where the input is a 5×992 array representing 1 second of stimulus and spike history, and the target is a scalar representing the number of spikes in the summation interval. For the spike distance model, each sample is a tuple (input, target) where the input is the same as that for the Poisson models, and the target is a 128-length array representing ~128 ms of ground-truth spike distance data, with $t = 0$ positioned at index 32. The choice of these values are hyperparameters discussed in Appendix A.11.

As the dataset's sequential nature means that adjacent samples will be the same except for a 1 time step shift, we

introduce a configurable stride parameter for the training dataset. A larger stride decreases the number of samples that constitute an epoch; this allows model evaluation and checkpoints to be carried out frequently while still being synchronized to epoch completions. The intermediate samples are still used in training, but they appear as augmentations by random indexing between stride indices. For all experiments, the stride was set to 13. A value of 13 resulted in the U-shaped learning curve being observed over 80 epochs.

A.9. Models

The convolutional kernels used in the shared base architecture are shown in Table 1. In addition to the convolutional layers, a learnable positional embedding was added to the output of the initial convolutional layer of the shared architecture. The kernels used for the spike distance head and the Poisson head are shown in Tables 2 and 3 respectively. The shared base architecture accounts for 88% of parameters and 93% of multi-adds in the spike distance model, and accounts for practically all the parameters and multi-adds for the Poisson models.

ConvNext blocks introduced by Liu et al. (2022) were chosen to form the main component of the architecture as they are commonly used and well-tested. Our implementation includes the Global Response Normalization layer (GRN) from ConvNextv2 blocks introduced by Woo et al. (2023). We use dropout at the end of each block, at a rate of 0.2.

A.10. Architecture Search

The number of mid-layers in the shared base architecture was varied in order to increase confidence that performance differences between the spike distance model and the Poisson models are not due to differences in layer and parameter counts. The number of mid-layers was varied between 0 and 7 (inclusive) and the configuration that best served the Poisson-80 ms model was chosen. This selection was based on the interquartile mean of the loss across cells in the validation set. Focus was given to the Poisson-80 ms model as it was observed to be the most competitive Poisson model.

For each of these configurations, the Poisson-80 ms model was trained once for each of the 60 chicken RGCs. The interquartile mean loss is shown in Figure 15. To gauge the magnitude of the effect on metrics, the interquartile mean Pearson correlation is reported in Figure 16. Other metrics show a similar lack of dependency on layer count. The 5-layer configuration was selected based on the loss; however, it seems like the number of mid-layers has very little effect on the performance of the Poisson-80 ms model.

Table 1. Base architecture shared by both Poisson and spike distance models. ~302 k parameters, ~8.28 M multi-adds.

Layer	Input size	Kernels (length, channels, stride)	Output size
Initial conv	5×992	$\begin{bmatrix}15, & 64, & 2\end{bmatrix} \times 1$	64×496
ConvNext blocks (downsampling)	64×496	$\begin{bmatrix}3 & 64 & 2\\ 1 & 128 & 1\\ 5 & 128 & 1\\ 1 & 64 & 1\end{bmatrix} \times 6$	64×8
ConvNext blocks	64×8	$\begin{bmatrix}1, & 128, & 1\\ 3, & 128, & 1\\ 1, & 64, & 1\end{bmatrix} \times 4$	64×8

Table 2. Distance model head. ~29 k parameters, ~587 k multi-adds.

Layer	Input size	Kernels (length, channels, stride)	Output size
ConvNext	64×8	$\begin{bmatrix}1, & 128, & 1\\ 5, & 128, & 1\\ 1, & 16, & 1\end{bmatrix} \times 1$	16×16
ConvNext	16×16	$\begin{bmatrix}1, & 32, & 1\\ 5, & 32, & 1\\ 1, & 16, & 1\end{bmatrix} \times 3$	16×128
Conv	16×128	$\begin{bmatrix}1, & 1, & 1\end{bmatrix} \times 1$	1×128

Table 3. Poisson model head. 513 parameters, 513 multi-adds.

Layer	Input size	Kernels (length, channels, stride)	Output size
Flatten	64×8	None	512
FC layer	512×1	$\begin{bmatrix}1, & 512, & 1\end{bmatrix}$	1

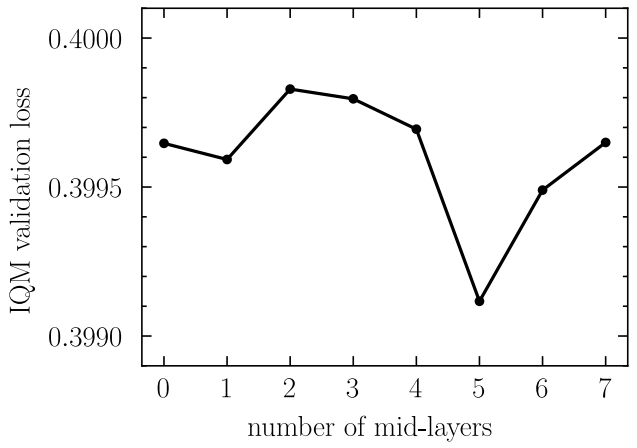


Figure 15. Interquartile mean validation loss for a range of mid-layer counts. The IQM is calculated over the 60 separately trained Poisson-80 models, one for each of the 60 RGCs. The loss is the mean validation set loss.

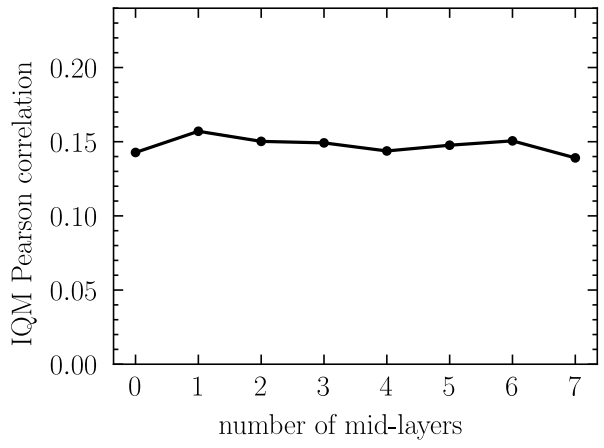


Figure 16. Interquartile mean Pearson correlation for a range of mid-layer counts. The IQM is calculated over the 60 separately trained Poisson-80 models, one for each of the 60 RGCs. Pearson correlation is calculated between the output and validation set ground-truth spike trains, smoothed with a Gaussian kernel with $\sigma = 20$.

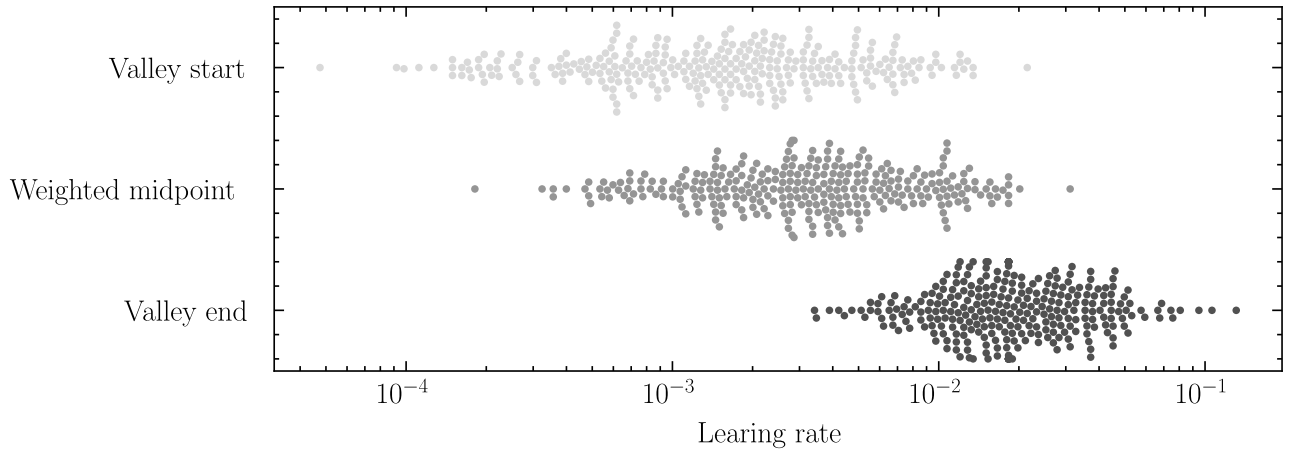


Figure 17. Three learning rates identified by the range test for learning rates described by Smith and Topin (2019). Valley start and valley end describe the edges of the U-shaped valley containing the suggested learning rate, which is a weighting of the edges.

A.11. Hyperparameters

A list of hyperparameters associated with models, training and inference.

Architecture. The following settings are associated with the model architectures.

- Channel count in the shared base: 64. Early experiments not reported here found 32 to have slightly worse performance for all models, and 64 and 128 were observed to have similar performance.
- Channel count in the distance model head: 16. Increasing this setting was observed to improve performance; however, to keep the model's parameter count close to that of the Poisson models, the channel count was kept low.
- Channel expansion factor of the ConvNext blocks: 2. Changes to this value were not tested.
- Dropout rate: 0.2. Early experiments not reported here observed benefits of dropout for all models. Once added, the dropout rate was not experimented with further.

Training. Settings associated with training were chosen as described below.

- Maximum learning rate: 5×10^{-4}. This was chosen by running the LR range test introduced by Smith and Topin (2019) on an earlier architecture for all chicken RGCs. After finalizing the architecture (see Appendix A.10), the range test was run again to ensure that the learning rate was still appropriate. Figure 17 plots the results of this second range test, which supports 5×10^{-4} still being an appropriate learning rate.
- Batch size: 256. Smaller batch sizes resulted in a training speed considered onerously slow.
- Dataset stride: 13. The dataset stride is described in Appendix A.8.
- Epochs: 80. Training is carried out for 80 epochs. Each time step will be part of a model input on average $80 \times \frac{992}{13} = 6105$ times.
- AdamW parameters: $(\beta_1, \beta_2, \text{eps}, \text{weight decay}) = (0.9, 0.99, 1 \times 10^{-5}, 0.3)$. These were chosen in line with heuristics described by Howard et al. (2020).
- Learning rate scheduler: 1-cycle scheduler policy described by Smith and Topin (2019) was used, with three-phase enabled and other options as default values in Pytorch 2.0's implementation.
- Precision: Nvidia's automatic mixed precision was used.

Spike distance. A number of settings were used to control the nature of the output spike distance array and the inference process:

- *stride*: how many time steps to shift forward when carrying out autoregressive inference. This was set to 80 time steps (~80 ms). Shorter strides offer improved inference at the cost of increased compute. 80 was chosen to match the Poisson-80 ms model.
- L: the length of the spike distance array the model outputs. 128 is the next power of two greater than 80, allowing the distance model's head to be a simple sequence of 4 upsample blocks. The additional time steps were split between before $t = 0$ and after $t = 80\,\text{ms}$ according to the next setting, i_0.
- i_o: the index into the array that corresponds to $t = 0$. This was set to 32. A low value negatively affects inference as described in Appendix A.13.2. 32 was chosen as it is the length of 2 activations before passing through the 4 upsample blocks (leaving 1 activation that extends past $t = 80$). A search for an optimal value has not been carried out.
- M: the maximum distance for an element of the spike distance field. This was set to 200 ms. This is not a very sensitive parameter. A few settings were experimented with, as described in Appendix A.13.1.

A.12. Computational Resources

All training and inference was carried out on a single workstation, with CPU, GPU and RAM specifications: AMD Ryzen 9 5900X CPU, Nvidia RTX 3090 GPU and 128 GiB RAM.

Training a single model on all 60 chicken RGCs took ~12 hours. This was repeated 11 times for the Poisson-80 ms model and 11 times for the spike distance model in order to calculate confidence intervals. Only 1 repeat was carried out for the remaining five Poisson models. For architecture variations, the Poisson-80 ms model was trained 8 times for all RGCs to investigate the number of mid-layers. Training on the frog RGCs took ~23 hours, and was done once for each of the 7 models.

The total training time was ~574 hours: 12 hours/model $\times$ $(2 \times 11 + 5 + 8)$ models + 23 hours/model $\times$ 7 models = 581. Inference was considerably quicker, taking a total of ~8 hours summed over all experiments.

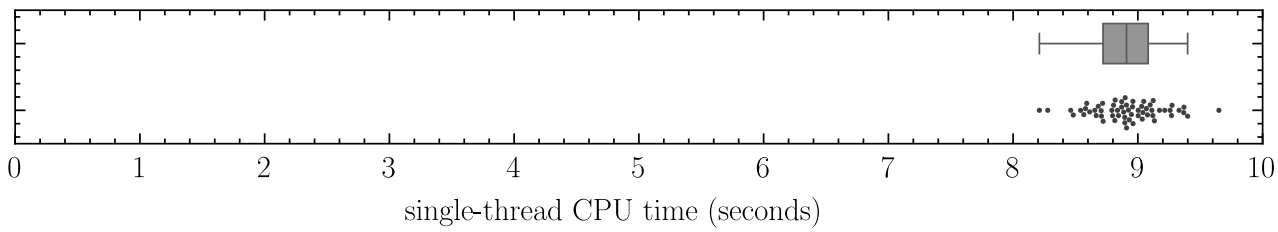


Figure 18. Running time of Algorithm 1 to infer 90 seconds of spike behaviour for 60 cells, in steps of 80 ms. Running time is calculated with Python's `timeit` package. In order to isolate the running time of Algorithm 1, inference is not autoregressive, and instead uses precalculated model outputs as inputs to the algorithm. (Mean, standard deviation) = (8.9, 0.27).

A.13. Spike Distance

This section covers some details on working with spike distance: choosing a maximum spike distance, and the benefits to inference of outputting an extended spike distance.

A.13.1. Maximum Spike Distance

If there are no spikes, then the distance to the nearest spike can be considered infinite. When approximating a distance function, infinite distances can be avoided by fixing a maximum allowed distance. The choice of a maximum is important beyond simply the avoidance of undefined values—it is also useful to limit how far into the past or future we expect a distance function to be reasonably approximated. Consider the spike distance function f_S evaluated at some time point t_1. The single value $f_S(t_1)$ contains information about all time points in the interval $[t_1 - f_S(t_1), t_1 + f_S(t_1)]$. For example, if $f_S(t_1) = 10$, then we know that there are no spikes in the open interval $(t_1 - 10, t_1 + 10)$. When training a model for spike prediction, we do not expect the model to anticipate spiking activity far into the future. For a model that predicts spike distance, one way to limit its exposure to the future is to clamp the target spike distance function. In this paper, we work with a maximum spike distance of 200 ms. Through experimentation not reported here, for the dataset used in this paper, we tested a range of maximum distances and found that there was not a major difference in performance when the maximum spike distance is between 100 ms and 600 ms. Below 100 ms and above 600 ms performance begins to degrade.

A.13.2. An Extended Spike Distance Improves Inference

Spike prediction performance is improved if the spike distance function outputted by the neural network extends both before and after the interval in which spikes are to be predicted. This means that when predicting L time steps starting from t_0, the neural network should output a spike distance function that extends both before t_0 and after $t_0 + L$.

The reason for this lies in the capacity of spike distance to carry information about a spike train both before and after the time point at which it is evaluated. For example, the spike distance at $t_0 - 1$ (past) can be affected by spikes that come after t_0 (future). In other words, the spike distance representation for a spike train does not fall neatly into the same time interval. By extending the spike distance function before t_0 and after $t_0 + L$, there is more information available for the inference algorithm to judge the placement of spikes. The model presented in this work is trained to output a spike distance array of length 128, where 32 elements are assigned to the past. If predicting a spike train for 80 time steps into the future, then there would be 16 elements of the model's output remaining that extend beyond the end of the spike train.

A.14. Discrete Spike Distance

In practice, both spike times and spike distance will be discretized. Instead of the input being exact spike times, we will use spike counts sampled at a certain sample period T, and instead of evaluating the spike distance at any real, we will evaluate the distance at discrete times separated by the same period T and collect the values in a spike distance array.

A naive and computationally simple approach of approximating the continuous spike distance is to count the number of samples until a sample with a non-zero spike count. This is a suitable approach when the sample rate is fast in comparison to the spike rate and spikes are rarely coincident in the same sample. Indeed, the main results in this work involve sample frequencies fast enough that no spikes are coincident in the same sample. When using a slower sampling rate, it may be beneficial to use the approach outlined below, which more faithfully discretizes the minimum distance when there is uncertainty over the precise location of spikes.

To approximate the continuous spike distance function $f_S(t) : \mathbb{R} \to \mathbb{R}$ introduced in Section 4, we are seeking a function $\hat{f}_S(t) : \mathbb{N} \to \mathbb{R}$ that is a faithful approximation when both spike times and the points being evaluated

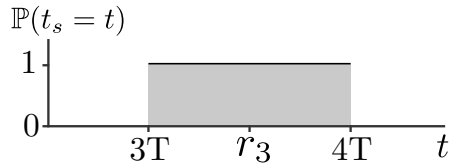


Figure 19. If one spike is detected in sample 3, then the spike distance for sample 3 is given by $\hat{f}(3) = \mathbb{E}[t_3 - r_3] = 0.25T$, where $r_3 = 3.5T$ and t_3 is a random variable uniformly distributed over $[3T, 4T]$.

are discretized. Osher and Fedkiw (2003) presents several methods for discretizing implicit representations; there the authors use known properties of the physical systems being modelled in order to discretize implicit functions. Our spike events are far less analytically constrained, and so we will instead take a probabilistic approach.

A.14.1. Motivating the Discrete Spike Distance

In the continuous case, the two points between which a distance is calculated are obvious: the spike time and the time for which the spike distance is being evaluated. In the discrete case, we must consider candidates for both of these time points.

Consider recording spikes at a sample period $T = 20\,\mathrm{ms}$, where each sample is a sum of the spikes that occurred within a $20\,\mathrm{ms}$ interval. If a single spike is recorded in the 5th sample, and the 5th sample collected spikes from the interval $(80\,\mathrm{ms}, 100\,\mathrm{ms})$, then where in this interval should we consider the spike to have occurred? Another question: if we wish to calculate an array of spike distances at some sample rate, what reference point should each sample use from which to measure distance? For spike times there is uncertainty, and for reference points there is a choice to be made.

A few decisions and assumptions are sufficient to determine a discrete distance. First, we will treat spike times as being random variables distributed uniformly over the sample in which they are recorded. This decision is reasonable when the samples represent a count of spikes within an interval, and there is no further information about the location of spikes within the interval. Indeed, this is the case for this work—the spike events are outputs of a black-box detection mechanism of the MEA and are subsequently rebinned to reduce the sample rate. The second decision is to fix our array of distance calculations to the same frequency as the spike recording, and we measure distances from the midpoint of each interval.

To appreciate the implications of these choices, consider the case where a single spike is recorded in sample i and we wish to evaluate the discrete spike distance for the same index, i. The naive solution is to define $\hat{f}_S(i) = 0$. However, 0 is not a good representation of the spike distance over the whole interval $[iT, (i+1)T]$; 0 is only ever the value of the spike distance at the instantaneous time a spike occurs. A better value for $\hat{f}_S(i)$ should incorporate the fact that there is uncertainty about when exactly a spike occurs within this interval of length T. The approach we take goes as follows. Fix the reference point r_s from which spike distances are measured to be the interval's midpoint, $r_s = (i + 0.5)T$. Consider the spike time to have a uniform distribution over the interval $[iT, (i+1)T]$. The value for $\hat{f}_S(i)$ will be defined as the expected distance to the reference point, where the expectation is calculated over the uniformly distributed spike time. Figure 19 shows this case in more detail. The next section formalizes these ideas and arrives at a simple procedure for calculating the discrete spike distance from a sequence of spike counts.

A.14.2. Definition and Calculation

What follows is a formalization of the discrete spike distance that can be thought of as applying extreme value theory to spike distances. Coles (2001) is a good introduction to extreme value statistics. What prevents the direct use of a plug-and-play theorem from extreme value theory is that in our situation we are not interested in the asymptotic behaviour of an unknown distribution, but the finite behaviour of a known distribution.

As part of the formalization, we will also generalize what was introduced in Section 4. Equation 1 in Section 4 defined a spike count sequence to be a binary array. This was a simplification allowed for by the high sample rate that ensured no two spikes were recorded in the same sample. Below, we relax this assumption and allow for the possibility of multiple spikes in a single sample.

We set the scene with two definitions. Similar to how the continuous spike distance function was parameterized by a set of spike times, the discrete spike distance function is parameterized by a sequence of spike counts.

Definition A.1. Let $T \in \mathbb{R}$ be the sample rate. Let $l \in \mathbb{N}$ be a positive integer representing the number of samples. Then a **spike count sequence** $S = (s_i)_{i=0}^{l-1}$ at sample rate T is a sequence of spike counts, where s_i is the number of spikes recorded in the interval $[iT, (i+1)T]$.

If $S = (s_i)_{i=0}^{l-1}$ is a spike count sequence with a total spike count $n = \sum_{i=0}^{l-1} s_i$, we can associate with it a sequence of independent random variables, $X_0, X_1, \ldots X_{n-1}$, where each X_j represents a spike time distributed uniformly over one of the length T intervals. For example, if $s_0 = 3$ and $s_1 = 1$, then X_0, X_1 and X_2 would be distributed uniformly over the interval $[0, T]$ while X_3 would be uniformly distributed over the interval $[T, 2T]$.

Definition A.2. Let i be a sample index. Let S be a spike

count sequence and let $X_0, X_1, ...X_{n-1}$ be independently identically distributed real random variables representing the spike times of the n spikes from S. Each X_j is distributed uniformly over the sample interval in which it is recorded. Let $r_i = i + 0.5$ represent the midpoint of the ith sample interval. For each X_j let $D_j = |X_j - r_i|$ be the derived random variable representing the distance of the jth spike to the midpoint r_i. The **discrete spike distance function** $\hat{f}_S : \mathbb{N} \to \mathbb{R}$ at i is given by:

$$\hat{f}_S(i) = \mathbb{E}_{D_0, D_1, \ldots, D_{n-1}} \min_{0 \le j < n} D_j \tag{2}$$

and has units of the sampling period, T.

Calculating this value involves considering only the one or two samples containing spikes that are closest to the sample of interest, i. This could be sample i itself, a single sample to the left or right of i or two samples equidistant to i, one on either side. Being able to ignore most of the samples allows for the discrete spike distance to be expressed neatly in terms of the spike counts of the closest non-empty sample(s).

Proposition A.3. *Let $d \in \mathbb{N}$ be the difference in samples between the sample of interest i and the closest spike-containing sample(s). Let $m \in \mathbb{N}$ the number of spikes contained within the closest spike-containing sample(s).*

The value of $\hat{f}_S(i)$ from Equation 2 is given by:

$$\hat{f}_S(i) = \begin{cases} \hat{f}_S(i) = \frac{1}{2(m+1)} & \textit{if one or more spikes in sample } i \\ \hat{f}_S(i) = d - \frac{1}{2} + \frac{1}{m+1} & \textit{otherwise.} \end{cases} \tag{3}$$

Proof. First, the first case: there are $m > 0$ spikes in sample i. The presence of the $\min$ function in Equation 2 means that we can ignore all spikes that are not in sample i, as they will not affect the expectation. Let $D_0, D_1, \ldots, D_{m-1}$ be the random variables representing the distance of the mth spike recorded within sample i. Single out one of these, D_1, and we will return later to include the others. D_1 will contribute to the expectation only when it is smaller than all others: if $D_1 = t$, then all $m - 1$ other distances must be greater than t. The probability $\mathbb{P}(D_1 = t) = 2$ (uniform over interval of length $\frac{1}{2}$) and the probability of other D_j having a value larger than t is $\mathbb{P}(D_j > t) = 2(\frac{1}{2} - t)$.

The contribution to the expectation for D_1 then amounts to:

$$\begin{aligned} &\int_0^{\frac{1}{2}} t \cdot \mathbb{P}(D_1 = t) \prod_{j=1}^{m-1} \mathbb{P}(D_j > t) \, \mathrm{d}t \\ &= \int_0^{\frac{1}{2}} t \cdot (2) \prod_{j=1}^{m-1} (2)(\frac{1}{2} - t) \, \mathrm{d}t \\ &= 2 \int_0^{\frac{1}{2}} t(1-2t)^{m-1} \, \mathrm{d}t \\ &= 2 \left[-\frac{t}{m}(1-2t)^m \right]_0^{\frac{1}{2}} + \int_0^{\frac{1}{2}} \frac{1}{m}(1-2t)^m \, \mathrm{d}t \\ &= 0 + \left[-\frac{1}{2m(m+1)}(1-2t)^{m+1} \right]_0^{\frac{1}{2}} \\ &= \frac{1}{2m(m+1)} \, . \end{aligned}$$

We singled out D_1; however, any of the m spikes are equally likely to be the closest spike. Summing the contribution over m spikes gives us:

$$\frac{1}{2(m+1)}$$

as required by the first case.

What distinguished the second case is that the reference point r_i is not in the same sample as the closest spike(s). Let $m > 0$ be the number of spikes that are in the closest one or two samples, situated d samples from the sample of interest i. We ignore spikes from other samples. Let $D_0, D_1, ...D_{m-1}$ be the random variables representing the distance of the mth spike to the reference point r_i. Single out one of these, D_1. D_1 will contribute to the expectation only when it is smaller than all others: if $D_1 = t$, then all $m - 1$ other distances must be greater than t. This time, D_1 has a uniform distribution over the interval $[d - \frac{1}{2}, d + \frac{1}{2}]$ of length 1 with $\mathbb{P}(D_1 = t) = 1$ in this interval. The probability of other D_j having a value larger than t is $\mathbb{P}(D_j > t) = d + \frac{1}{2} - t$.

The contribution to the expectation for D_1 then amounts to:

Table 4. The approximate spike distance compared to the discrete spike distance. Discrete spike distances are evaluated for the samples 0 to 8 when there are three spikes recorded: one in sample 2 and two in sample 8.

sample index, i	0	1	2	3	4	5	6	7	8
spikes			|						||
m	1	1	1	1	1	3	2	2	2
d	2	1	0	1	2	3	2	1	0
spike distance, $\hat{f}_S(i)$, in units of T	2	1	$\frac{1}{4}$	1	2	$2\frac{3}{4}$	$1\frac{5}{6}$	$\frac{5}{6}$	$\frac{1}{6}$
approx. spike distance, in units of T	2	1	0	1	2	3	2	1	0

$$\int_{d-\frac{1}{2}}^{d+\frac{1}{2}} t \cdot \mathbb{P}(D_1 = t) \prod_{j=1}^{m-1} \mathbb{P}(D_j > t)\,\mathrm{d}t$$

$$= \int_{d-\frac{1}{2}}^{d+\frac{1}{2}} t \cdot (1) \prod_{j=1}^{m-1} (d + \frac{1}{2} - t)\,\mathrm{d}t$$

and by change of variable,

$$= \int_0^1 (d - \frac{1}{2} + t) \cdot (1) \prod_{j=1}^{m-1} (1 - t)\,\mathrm{d}t$$

$$= \int_0^1 (d - \frac{1}{2} + t)(1 - t)^{m-1}\,\mathrm{d}t$$

$$= \left[-\frac{d - \frac{1}{2} + t}{m}(1 - t)^m\right]_0^1 + \int_0^1 \frac{1}{m}(1 - t)^m\,\mathrm{d}t$$

$$= \frac{d - \frac{1}{2}}{m} + \left[-\frac{1}{m(m+1)}(1 - t)^{m+1}\right]_0^1$$

$$= \frac{d - \frac{1}{2}}{m} + \frac{1}{m(m+1)}\,.$$

We singled out D_1; however, any of the m spikes are equally likely to be the closest spike. Summing the contribution over m spikes gives us:

$$d - \frac{1}{2} + \frac{1}{m(m+1)}$$

as required by the second case. □

Table 4 compares the discrete spike distance described above to the approximate spike distance, including the values for m and d from Equation 3.