KalWeNN: Kalman-Filtered Memory for Weightless Neural Network Classifiers

Helena A. B. Moreno[1], Natasha C. da Fonseca[1], Gustavo R. Villela[1], Priscila M. V. Lima[1], Pablo Rangel[1], Leandro S. de Araújo[2], Claudio M. de Farias[1]

[1] PESC / COPPE / UFRJ - RJ, Brazil

[2]IC / UFF - RJ, Brazil

{helena,natasha,grvillela,priscilamvl}@cos.ufrj.br

{pablorangel,cmicelifarias}@cos.ufrj.br, leandro@ic.uff.br

Abstract. *Weightless Neural Networks (WNNs) such as WiSARD learn by storing information directly in RAM-based memory structures, enabling fast training and efficient hardware implementation. However, conventional WiSARD memories store only binary activations or occurrence counts, providing no measure of confidence in the stored information. This paper introduces KalWeNN (Kalman Weightless Neural Network), a weightless classifier that replaces the traditional counting memory with Kalman filter-based memory cells. Each memory address maintains both an estimate of its association with a class and the uncertainty of that estimate. During training, address activations are treated as noisy observations and incorporated through recursive Kalman update. We evaluate KalWeNN on twelve classification datasets under two architectural configurations and compare it with a standard WiSARD implementation. Results show that KalWeNN consistently matches or improves classification accuracy, achieving approximately 93% accuracy on MNIST. More importantly, it substantially improves run-to-run stability, often reducing performance variability by a factor of four to five. These gains are obtained with only modest increases in memory usage and execution time, preserving the lightweight nature of weightless neural networks.*

1. Introduction

Weightless Neural Networks (WNNs) are classifiers that store learned knowledge directly in lookup tables built on top of Random Access Memories (RAMs) rather than in trainable connection weights, as ordinary neural networks. Among them, the WiSARD model is the most prominent representative, being the first artificial neural network architecture to be patented and commercially marketed [Aleksander et al. 1984]. Unlike conventional neural networks, which rely on iterative optimization procedures such as gradient descent [LeCun et al. 2015], WiSARD performs learning through direct memory updates in a single training pass. This characteristic enables extremely fast training and efficient hardware implementation, making WNNs attractive for resource-constrained environments, including embedded systems, edge computing, and real-time stream processing [Susskind et al. 2022].

Despite its computational efficiency, the memory mechanism employed by WiSARD remains fundamentally simplistic. Each address stores either a binary indicator or

a frequency count, treating all activations as equally reliable regardless of their statistical significance. From a probabilistic perspective, the information associated with each address can be viewed as an estimate of the strength of association between a given input pattern and a target class, inferred from a sequence of noisy observations. On the other hand, state estimation under uncertainty is the central problem addressed by the Kalman filter, originally introduced by Kalman in 1960 [Khodarahmi and Maihami 2023]. The Kalman filter recursively maintains both an estimate of a latent variable and a measure of its associated uncertainty. This framework has been successfully applied to neural network training, most notably through the Extended Kalman Filter, which formulates weight adaptation as a state estimation problem [Haykin 2001, Hammond et al. 2025]. To the best of our knowledge, however, uncertainty-aware state estimation has not yet been incorporated into the memory structures of weightless neural networks.

To address this gap, we propose the KalWeNN (Kalman Weightless Neural Network), a novel weightless classifier that replaces the conventional counting memory of WiSARD with Kalman filter-based memory cells. Instead of storing binary values or occurrence counts, each memory address maintains a state estimate representing the degree of association between the address and a class, together with its uncertainty. We evaluate KalWeNN on multiple classification datasets against a standard WiSARD under identical architectural configurations, analyzing predictive performance, run-to-run stability, memory consumption, and the influence of the tuple size, encoding dimensionality, and Kalman noise parameters. Overall, these gains come at a modest cost in memory and training time, preserving the lightweight nature of weightless networks while opening new possibilities for uncertainty-aware weightless models.

2. Kalman Filter

The Kalman filter is a recursive estimator for the latent state $x_k \in \mathbb{R}^n$ of a linear dynamical system observed through noisy measurements $z_k \in \mathbb{R}^m$, where k indexes discrete time steps. It assumes a transition model $F \in \mathbb{R}^{n\times n}$ for the state evolution and an observation model $H \in \mathbb{R}^{m\times n}$ relating the state to the measurement, each subject to zero-mean Gaussian noise, giving the linear-Gaussian state-space model:

$$\begin{cases} x_k = F\,x_{k-1} + w_k, & w_k \sim \mathcal{N}(0, Q), \\ z_k = H\,x_k + v_k, & v_k \sim \mathcal{N}(0, R), \end{cases} \tag{1}$$

where w_k, v_k are the process and observation noise with covariances Q and R. Alongside the state estimate $\hat{x}_k$, the filter propagates an estimation covariance $\hat{P}_k$ that quantifies its uncertainty. From initial conditions $\hat{x}_0, \hat{P}_0$, each iteration has two stages.

Prediction projects the estimate and covariance forward, producing the *a priori* quantities (subscript pred):

$$\hat{x}_{k,\mathrm{pred}} = F\,\hat{x}_{k-1}, \quad \hat{P}_{k,\mathrm{pred}} = F\,\hat{P}_{k-1}F^{\mathsf{T}} + Q, \quad \hat{z}_k = H\,\hat{x}_{k,\mathrm{pred}}. \tag{2}$$

Correction then folds in the observation z_k through the innovation $z_k - \hat{z}_k$, yielding the *a posteriori* estimates:

$$S_k = H\,\hat{P}_{k,\mathrm{pred}}H^{\mathsf{T}} + R, \quad K_k = \hat{P}_{k,\mathrm{pred}}H^{\mathsf{T}}S_k^{-1}, \tag{3}$$

$$\hat{x}_k = \hat{x}_{k,\mathrm{pred}} + K_k\,(z_k - \hat{z}_k)\,, \quad \hat{P}_k = (I - K_kH)\,\hat{P}_{k,\mathrm{pred}}. \tag{4}$$

The Kalman gain K_k in Equation (3) balances prediction and observation: when the predicted covariance is large relative to R the observation dominates, while a confident estimate yields only small corrections.

3. Kalman Weightless Neural Network

In the classical WiSARD, each class is represented by a discriminator of RAM nodes, and each n-tuple extracted from the binarized input addresses one position of its RAM, whose integer counter is incremented during training. KalWeNN is a multi-discriminator classifier that replaces this counting mechanism with a recursive Bayesian estimation process in which each RAM address is associated with an independent Kalman filter: rather than storing occurrence counts, each address maintains an estimate of its association with a class together with the uncertainty of that estimate, updating it from the binary activations treated as noisy observations.

Input binarization. WiSARD operates on binary inputs, each numeric feature is first converted into bits through a thermometer encoding. For every feature, its range over the training set is split into a fixed number of levels, the thermometer size, and a value sets the leading bits up to its level to one and leaves the rest at zero, so larger values activate more bits. The per-feature codes are then concatenated into a single binary vector, which the model partitions into tuples and reads as RAM addresses. The thermometer size controls how finely each value is quantized.

State-Space Model. KalWeNN instantiates a separate state-space model for each RAM address by adapting Equation (1) as scalar form ($n = m = 1$). Let x_k^i denote the (unobserved) content of a given address i and $\hat{x}_k^i$ its estimate, where the index k now counts the training presentations of patterns belonging to the discriminator's class. At each such presentation, the tuple t of input sample updates every address i of the accessed RAM according to one binary observation:

$$z_k^i = \begin{cases} 1, & \text{if the tuple } t \text{ addresses position } i. \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

so that x_k^i can be interpreted as the underlying activation frequency of the address i for the corresponding class.

Each address i maintains an independent scalar filter with $F = H = 1$, i.e., the address content is modeled as approximately constant between presentations, subject only to a small process noise Q that prevents the filter from becoming insensitive to new evidence. Since $F = H = 1$ implies $\hat{x}_{k,\text{pred}}^i = \hat{z}_k^i = \hat{x}_{k-1}^i$, the recursion of Section 2 reduces to:

$$\hat{P}_{k,\text{pred}}^i = \hat{P}_{k-1}^i + Q, \quad K_k^i = \frac{\hat{P}_{k-1}^i + Q}{\hat{P}_{k-1}^i + Q + R}, \tag{6}$$

$$\hat{x}_k^i = \hat{x}_{k-1}^i + K_k^i \left(z_k^i - \hat{x}_{k-1}^i\right), \quad \hat{P}_k^i = \left(1 - K_k^i\right)\left(\hat{P}_{k-1}^i + Q\right). \tag{7}$$

Equation (7) makes the contrast with the classical scheme explicit: instead of an unbounded integer counter, each address i performs an uncertainty-weighted recursive average of its binary observations. The gain in Equation (6) decreases monotonically as $\hat{P}_k^i$ shrinks, so early observations move the estimate substantially while later ones produce

progressively smaller corrections, and the ratio Q/R controls the steady-state responsiveness of the memory. For $Q \to 0$, the update converges to the empirical activation frequency of the address; for $Q > 0$, it behaves as an adaptive exponential forgetting scheme, retaining sensitivity to distributional drift in the training stream.

4. Experiments and Results

Experimental setup. All experiments were run on a single MacBook Pro (Retina, 13-inch, Early 2015) with a 2.7 GHz dual-core Intel Core i5 and 8 GB of 1867 MHz DDR3 memory. We evaluated the models on twelve classification datasets: wine, iris, breast cancer, digits, vehicle, segment, vowel, letter, pendigits, satellite, texture and MNIST. Each dataset was used in full with a random 80/20 train/test split, except MNIST, for which we kept its official 60,000/10,000 division. Since the bit-to-RAM mapping and, in the Kalman model, the example presentation order depend on a random seed, every reported number is averaged over 30 seeds. We compared four models. *WiSARD* and *KalWeNN* use 8-bit tuples with a 16-level thermometer encoding in dense memory, while *WiSARD (dict)* and *KalWeNN (dict)* use 20-bit tuples with a 20-level thermometer. The Kalman noise parameters were chosen with a small grid search over $q \in \{10^{-4}, 10^{-3}, 10^{-2}\}$ and $r \in \{10^{-2}, 10^{-1}, 5 \times 10^{-1}, 1, 2\}$, with tuple and thermometer sizes held fixed.

Accuracy and Performance Results. For every model and dataset we report the mean accuracy and its standard deviation over the seeds, the training and test times, and the memory of the trained model. Table 1 brings the results together. Looking first at accuracy, the Kalman update matches or improves on the classic WiSARD in every one of the twelve datasets, in both regimes. On the easy ones, such as wine and digits, the gain is small, since every model already classifies almost everything correctly. On harder problems the gap opens up: on letter accuracy rises from 0.54 with the small WiSARD to 0.86 with the KalWeNN (dict), on vowel it goes from 0.78 to 0.91, and similar jumps appear on vehicle and satellite. The best result almost always comes from the KalWeNN (dict), and on MNIST it reaches about 93% under the official split, close to the values usually reported for weightless networks. Since MNIST is the largest dataset we use, this shows the improvement is not limited to small problems.

Sensitivity to the Kalman parameters. Since q and r act only through the gain $K_k = (\hat{P} + q)/(\hat{P} + q + r)$, what matters is their ratio, and accuracy varies smoothly across the grid rather than peaking at an isolated point. The best configurations consistently fall in the low-q, high-r corner: a large r makes each activation a weakly trusted observation, suppressing noise in a continuous, per-cell fashion much like bleaching and driving the lower run-to-run variance, while a small q keeps the estimate close to the empirical activation frequency. Increasing q or lowering r trades stability for adaptivity, but within the tested grid these shifts are marginal, so KalWeNN does not require fine tuning to outperform the classic counter.

Stability and Computational Cost Results. Stability is where the Kalman model makes its strongest case. We use the coefficient of variation (CV), the ratio of the standard deviation to the mean accuracy across seeds, which compares stability fairly across datasets sitting at very different accuracy levels. The CV in Table 1 shows the Kalman models to be steadier than the classic ones in almost every dataset and regime, often by a large factor. In MNIST the CV falls from 0.84% to 0.14% in the small regime and from 1.19%

Table 1. Comparison of architectural regimes (small/large tuples) for WiSARD and KalWeNN.

Dataset	Model	Tuple	Thermometer	Accuracy	σ	CV (%)	Train (s)	Test (s)	Mem. (MB)
wine	WiSARD	8	16	0.9750	0.0262	2.688	0.0016	0.0005	0.1597
wine	KalWeNN	8	16	0.9833	0.0136	1.384	0.0030	0.0005	0.3195
wine	WiSARD (dict)	20	20	0.9639	0.0217	2.251	0.0027	0.0014	0.0184
wine	KalWeNN (dict)	20	20	0.9639	0.0279	2.896	0.0028	0.0010	0.0276
iris	WiSARD	8	16	0.9200	0.0452	4.915	0.0008	0.0003	0.0492
iris	KalWeNN	8	16	0.9533	0.0427	4.478	0.0022	0.0002	0.0983
iris	WiSARD (dict)	20	20	0.9300	0.0233	2.509	0.0010	0.0004	0.0030
iris	KalWeNN (dict)	20	20	0.9467	0.0267	2.817	0.0009	0.0003	0.0045
breast_cancer	WiSARD	8	16	0.9149	0.0269	2.941	0.0037	0.0017	0.2458
breast_cancer	KalWeNN	8	16	0.9605	0.0153	1.595	0.0119	0.0015	0.4915
breast_cancer	WiSARD (dict)	20	20	0.9675	0.0171	1.770	0.0058	0.0048	0.0490
breast_cancer	KalWeNN (dict)	20	20	0.9711	0.0124	1.281	0.0225	0.0064	0.0735
digits	WiSARD	8	16	0.9700	0.0067	0.687	0.0305	0.0098	2.6214
digits	KalWeNN	8	16	0.9700	0.0075	0.777	0.0607	0.0099	5.2429
digits	WiSARD (dict)	20	20	0.9775	0.0067	0.690	0.0450	0.0989	0.6173
digits	KalWeNN (dict)	20	20	0.9775	0.0075	0.767	0.1474	0.0858	0.9259
vehicle	WiSARD	8	16	0.6129	0.0307	5.001	0.0035	0.0013	0.2949
vehicle	KalWeNN	8	16	0.6759	0.0187	2.765	0.0152	0.0013	0.5898
vehicle	WiSARD (dict)	20	20	0.7006	0.0246	3.511	0.0056	0.0067	0.0553
vehicle	KalWeNN (dict)	20	20	0.7212	0.0280	3.881	0.0170	0.0067	0.0830
segment	WiSARD	8	16	0.8794	0.0168	1.910	0.0090	0.0029	0.5448
segment	KalWeNN	8	16	0.9234	0.0102	1.100	0.0372	0.0025	1.0895
segment	WiSARD (dict)	20	20	0.9325	0.0116	1.249	0.0151	0.0272	0.0405
segment	KalWeNN (dict)	20	20	0.9535	0.0093	0.980	0.0484	0.0248	0.0608
vowel	WiSARD	8	16	0.7773	0.0376	4.832	0.0027	0.0009	0.4506
vowel	KalWeNN	8	16	0.8566	0.0276	3.221	0.0149	0.0009	0.9011
vowel	WiSARD (dict)	20	20	0.9086	0.0209	2.298	0.0057	0.0099	0.0508
vowel	KalWeNN (dict)	20	20	0.9121	0.0150	1.646	0.0127	0.0094	0.0761
mnist	WiSARD	8	16	0.7036	0.0059	0.843	32.5737	6.4099	32.1126
mnist	KalWeNN	8	16	0.8971	0.0013	0.141	51.4689	7.1759	64.2253
mnist	WiSARD (dict)	28	3	0.8700	0.0104	1.192	4.9867	4.0039	3.7144
mnist	KalWeNN (dict)	28	3	0.9333	0.0026	0.283	13.7407	4.6169	5.5715
letter	WiSARD	8	16	0.5357	0.0200	3.742	0.0771	0.0226	1.7039
letter	KalWeNN	8	16	0.7816	0.0048	0.611	0.1852	0.0166	3.4079
letter	WiSARD (dict)	20	20	0.8002	0.0113	1.415	0.0628	0.3559	0.2839
letter	KalWeNN (dict)	20	20	0.8598	0.0050	0.576	0.2628	0.4019	0.4259
pendigits	WiSARD	8	16	0.9081	0.0103	1.135	0.0506	0.0204	0.6554
pendigits	KalWeNN	8	16	0.9747	0.0033	0.337	0.2983	0.0240	1.3107
pendigits	WiSARD (dict)	20	20	0.9849	0.0022	0.221	0.1101	0.2430	0.3112
pendigits	KalWeNN (dict)	20	20	0.9872	0.0015	0.148	0.5803	0.2451	0.4668
satellite	WiSARD	8	16	0.6946	0.0183	2.634	0.1674	0.0438	0.8847
satellite	KalWeNN	8	16	0.8712	0.0066	0.759	0.2358	0.0213	1.7695
satellite	WiSARD (dict)	20	20	0.8561	0.0083	0.972	0.1719	0.2579	0.2288
satellite	KalWeNN (dict)	20	20	0.8892	0.0046	0.515	0.3392	0.1358	0.3432
texture	WiSARD	8	16	0.8118	0.0129	1.587	0.0718	0.0282	1.8022
texture	KalWeNN	8	16	0.9302	0.0060	0.642	0.1676	0.0229	3.6045
texture	WiSARD (dict)	20	20	0.9342	0.0057	0.605	0.0935	0.2321	0.1356
texture	KalWeNN (dict)	20	20	0.9548	0.0043	0.449	0.5242	0.3030	0.2033

to 0.28% in the large one, a four to fivefold reduction, and the lowest figures of the whole table belong to the Kalman models, such as 0.15% on pendigits and 0.52% on satellite. This stability comes from the graded memory. The run-to-run randomness, due to the bit-to-RAM mapping and the example order, flips whole recognitions in discrete steps in the classic model, whereas in the Kalman model it only nudges continuous values that average out once summed, so the decision depends far less on the seed.

5. Conclusion

We presented KalWeNN, a weightless classifier that replaces the counting memory of the WiSARD with a scalar Kalman filter inside each address, so that every activation is treated as a noisy observation and each address keeps both an estimate and its uncertainty. Across twelve datasets and two architecture regimes, the Kalman update matched or improved on the classic WiSARD in accuracy, with the largest gains on the harder problems and about 93% on MNIST under the official split. Its clearest advantage, however, was stability: the results changed much less from one run to the next, often by a factor of four or five, even at similar accuracy. This came at a modest cost in memory and training time, which stayed low enough to preserve the lightweight character that makes weightless networks appealing. Some directions remain open. The Kalman noise parameters q and r deserve further exploration, in particular how their values relate to statistical properties of the data. Another promising direction is to extend KalWeNN considering different neuron representation, such as Bloom filter [Santiago et al. 2020].

References

Aleksander, I., Thomas, W. V., and Bowden, P. A. (1984). WISARD: a radical step forward in image recognition. *Sensor Review*, 4(3):120–124.

Hammond, J. E., Soderstrom, T. A., Korgel, B. A., and Baldea, M. (2025). A selective Kalman filtering approach to online neural network updating under system drift. *Scientific Reports*, 15(1):43577.

Haykin, S., editor (2001). *Kalman Filtering and Neural Networks*. John Wiley & Sons, New York, NY, USA.

Khodarahmi, M. and Maihami, V. (2023). A review on kalman filter models. *Archives of Computational Methods in Engineering*, 30(1):727–747.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521(7553):436–444.

Santiago, L., Verona, L., Rangel, F., Firmino, F., Menasché, D. S., Caarls, W., Breternitz Jr, M., Kundu, S., Lima, P. M., and França, F. M. (2020). Weightless neural networks as memory segmented bloom filters. *Neurocomputing*, 416:292–304.

Susskind, Z., Arora, A., Miranda, I. D. S., Villon, L. A. Q., Katopodis, R. F., Araújo, L. S. d., Dutra, D. L. C., Lima, P. M. V., França, F. M. G., Breternitz Jr., M., and John, L. K. (2022). Weightless neural networks for efficient edge inference. In *Proceedings of the International Conference on Parallel Architectures and Compilation Techniques (PACT)*, pages 279–290, Chicago, IL, USA.