Time Series under Temporal Label Noise

Sujay Nagaraj
University of Toronto

Walter Gerych
Massachusetts Institute of Technology

Sana Tonekaboni
University of Toronto

Anna Goldenberg
University of Toronto

Berk Ustun
University of California, San Diego

Thomas Hartvigsen
University of Virginia

Abstract

Many time series classification tasks where labels vary over time are affected by *label noise* that also varies over time. Such noise can cause label quality to improve, worsen, or periodically change over time. We first propose and formalize *temporal label noise*, an unstudied problem for sequential classification of time series. In this setting, multiple labels are recorded in sequence while being corrupted by a time-dependent noise function. We demonstrate the importance of modelling the temporal nature of the label noise function and how existing methods consistently underperform. We then demonstrate the surprising noise tolerance of time series foundation models and how this collapses under temporal label noise.

1 Introduction

Most methods addressing label noise are designed for static, time-invariant data [2, 22, 24, 34], but many real-world tasks, such as healthcare, involve time series data where labels evolve over time. We introduce *temporal label noise*, a concept that remains largely unstudied. Examples include seasonal biases in self-reporting for mental health studies [4, 27], mislabeled activities in human activity recognition [15, 31], and noisy clinical annotations in electronic health records during busy or chaotic periods [40].

Existing static noise-robust methods [25, 26] underperform with temporal noise. We propose novel loss-correction techniques to train time series classifiers that are provably robust to label noise. We then explore the effects of temporal label noise on time series foundation models, and demonstrate the utility of our approaches even with pre-trained models.

2 Related Work

Time Series Machine learning for time series, especially in healthcare, often combines autoregressive modeling with deep neural networks [7, 19, 33]. RNNs and state space models are popular for sequence-to-sequence tasks due to their ability to represent data as latent states evolving over time. Attention mechanisms enhance performance by focusing on relevant data segments [37], improving tasks like outcome prediction and treatment optimization [9, 16, 32, 35, 38, 45]. These models mainly use supervised learning to capture changes in latent states, such as clinical labels (e.g., sick vs. healthy). A less explored issue is label noise in time series — how time series model performance is affected when training labels are corrupted.

Label Noise Our work addresses gaps in the literature on noisy labels [2, 22, 24, 34]. Most research addresses static label noise, which does not vary over time. We examine label noise that evolves

NeurIPS 2024 Workshop on Time Series in the Age of Large Models.

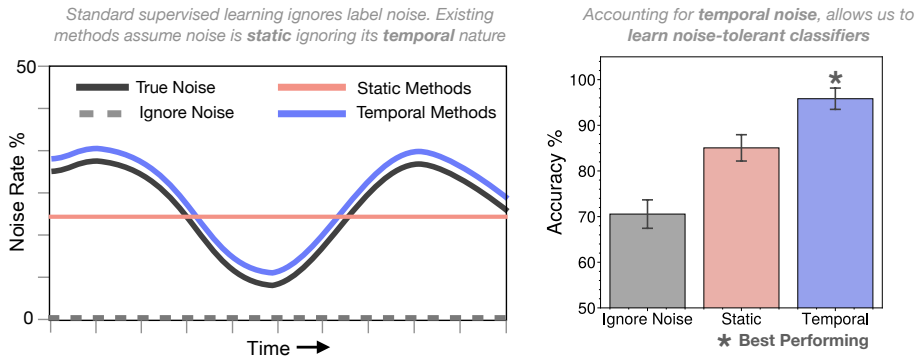

Figure 1: Label quality can vary over time due to *temporal* label noise. Existing methods assume noise is time-invariant (*static*) leading to loss in performance. Accurate modeling of temporal label noise, improves performance.

dynamically. Existing methods for noise in time series [1, 8] focus on identifying noisy samples rather than developing robust methods for temporal label noise.

We propose learning through empirical risk minimization with *noise-robust loss functions* [10, 18, 22, 25, 39]. Static approaches identify correct labels [13, 44] or use regularization to mitigate incorrect labels [14, 20]. Our results highlight learning from noisy labels when the noise process is known [25, 26]. We also provide additional methods to model noise directly from noisy data [17, 18, 26, 41, 42, 46] in Appendix A.

3 Methods

Preliminaries Consider a sequential classification task over C classes and T time steps. Each instance is characterized by a triplet of *sequences* over T time steps $(\mathbf{x}_{1:T}, \mathbf{y}_{1:T}, \tilde{\mathbf{y}}_{1:T}) \in \mathcal{X} \times \mathcal{Y} \times \mathcal{Y}$. Where $\mathcal{X} \subseteq \mathbb{R}^{d \times T}$ and $\mathcal{Y} = \{1, \ldots, C\}^T$. Here, $\mathbf{x}_{1:T}$, $\mathbf{y}_{1:T}$, and $\tilde{\mathbf{y}}_{1:T}$ are sequences of instances, *clean labels* and *noisy labels*, respectively.

Under temporal label noise, the *true* label sequence $\mathbf{y}_{1:T}$ is unobserved, and we only have access to a set of n noisy instances $D = \{(\boldsymbol{x}_{1:T}, \tilde{\boldsymbol{y}}_{1:T})_i\}_{i=1}^n$. We assume that each sequence in D is generated i.i.d. from a joint distribution $P_{\mathbf{x}_{1:T}, \mathbf{y}_{1:T}, \tilde{\mathbf{y}}_{1:T}}$, where the label noise process can vary over time. This distribution obeys two standard assumptions in sequential modeling and label noise [3, 6, 36]:

Assumption 1 (Future Independence)**.** *A label at time t depends only on the* past *sequence of feature vectors up to t:* $p(\mathbf{y}_{1:T} \mid \mathbf{x}_{1:T}) = \prod_{t=1}^{T} p(\mathrm{y}_t | \mathbf{x}_{1:t})$

Assumption 2 (Feature Independence)**.** *The sequence of noisy labels is conditionally independent of the features given the true labels:* $\tilde{\mathbf{y}}_{1:t} \perp\!\!\!\perp \mathbf{x}_{1:t} \mid \mathbf{y}_{1:t}$ *for* $t = 1, \ldots, T$

Assumption 1 allows the joint sequence distribution to factorize as: $p(\tilde{\mathbf{y}}_{1:T} | \mathbf{x}_{1:T}) = \prod_{t=1}^{T} q_t(\tilde{\mathrm{y}}_t | \mathbf{x}_{1:t})$. Assumption 2 allows for the noisy label distribution at t to be further decomposed as:

$$q_t(\tilde{\mathrm{y}}_t | \mathbf{x}_{1:t}) = \sum_{y \in \mathcal{Y}} q_t(\tilde{\mathrm{y}}_t \mid \mathrm{y}_t = y) p(\mathrm{y}_t = y \mid \mathbf{x}_{1:t}) \tag{1}$$

3.1 Learning from Temporal Label Noise

Our goal is to learn a sequential classification model $\boldsymbol{h}_\theta : \mathcal{X} \to \mathbb{R}^C$ with model parameters $\theta \in \Theta$. Here, $\boldsymbol{h}_\theta(\mathbf{x}_{1:t})$ returns an estimate of $p(\mathrm{y}_t | \mathbf{x}_{1:t})$. To infer the label at time step t, $\boldsymbol{h}_\theta$ takes as input a sequence of feature vectors up to t, and outputs a sequence of labels by taking the $\arg\max$ of the predicted distribution for each time step (see e.g., [3, 36]). We estimate parameters $\hat{\theta}$ for a model robust to noise, by maximizing the expected accuracy as measured in terms of the *clean* labels:

$$\hat{\theta} = \operatorname*{argmax}_{\theta \in \Theta} \mathbb{E}_{\mathbf{y}_{1:T} | \mathbf{x}_{1:T}} \prod_{t=1}^{T} p(\mathrm{y}_t = \boldsymbol{h}_\theta(\mathbf{x}_{1:t}) \mid \mathbf{x}_{1:t})$$

However, during training time we only have access to sequences of *noisy* labels. To demonstrate how we can sidestep this limitation, we first need to introduce a flexible way to model sequential noisy labels. Existing methods assume that noise is time-invariant (Fig. 1). To relax this assumption, we capture the temporal nature of noisy labels using a *temporal label noise function*, in Def. 1.

Definition 1. Given a sequential classification task with C classes and noisy labels, the *temporal label noise function* is a matrix-valued function $\boldsymbol{Q} : \mathbb{R}_+ \to [0,1]^{C\times C}$ that specifies the label noise distribution at any time $t > 0$.

We denote the output of the temporal noise function at time t as $\boldsymbol{Q}_t := \boldsymbol{Q}(t)$. This is a $C \times C$ matrix whose i, j^{th} entry encodes the flipping probability of observing a noisy label j given clean label i at time t: $q_t(\tilde{\mathrm{y}}_t = j \mid \mathrm{y}_t = i)$. We observe that $\boldsymbol{Q}_t$ is positive, row-stochastic, and diagonally dominant — ensuring that $\boldsymbol{Q}_t$ encodes a valid probability distribution [18, 26].

3.2 Loss Correction with Temporal label Noise

Modeling temporal label noise is crucial for training robust time series classifiers, but leveraging these models in empirical risk minimization is still a challenge. It is unclear how existing loss correction methods apply to time series. We provide theoretical results showing that learning is feasible when the true temporal noise function $\boldsymbol{Q}(t)$ is known. Proofs are included in Appendix B.

We begin by treating the noisy posterior as the matrix-vector product of a noise transition matrix and a clean class posterior (Eq. (1)). To this effect, we define the *forward sequence loss*:

Definition 2. Given a sequential classification task over T time steps, a noise function $\boldsymbol{Q}(t)$, and a proper composite loss function[1] ℓ_t, the *forward sequence loss* of a model $\boldsymbol{h}_\theta$ on an instance $(\tilde{\boldsymbol{y}}_{1:T}, \boldsymbol{x}_{1:T})$ is:

$$\overrightarrow{\ell}_{seq}(\tilde{\boldsymbol{y}}_{1:T}, \boldsymbol{x}_{1:T}, \boldsymbol{h}_\theta) := \sum_{t=1}^{T} \ell_t(\tilde{y}_t, \boldsymbol{Q}_t^\top \boldsymbol{h}_\theta(\boldsymbol{x}_{1:t}))$$

An intriguing property of the *forward sequence loss* is that the minimizer of the *forward sequence loss* over the noisy labels maximizes the likelihood of the data over the clean labels. This suggests that the *forward sequence loss* is robust to label noise:

Theorem 1. A classifier that minimizes the empirical *forward sequence loss* over the noisy labels maximizes the empirical likelihood of the data over the clean labels.

$$\operatorname*{argmin}_{\theta\in\Theta} \mathbb{E}_{\tilde{\mathbf{y}}_{1:T}, \mathbf{x}_{1:T}} \overrightarrow{\ell}_{seq}(\tilde{\mathbf{y}}_{1:T}, \mathbf{x}_{1:T}, \boldsymbol{h}_\theta) = \operatorname*{argmin}_{\theta\in\Theta} \sum_{t=1}^{T} \mathbb{E}_{\mathbf{y}_{1:t}, \mathbf{x}_{1:t}} \ell_t(\mathrm{y}_t, \boldsymbol{h}_\theta(\mathbf{x}_{1:t}))$$

4 Experiments

We benchmark our methods on real-world time series classification tasks to evaluate robustness to temporal label noise and identify when accounting for such noise is critical. Our datasets include accelerometer data for human activity recognition (`har` and `har70`) and EEG signals for sleep and blink detection (`eeg_sleep` and `eeg_eye`). We consider ground truth labels as clean and corrupt them using various temporal noise functions for training, as is the gold-standard evaluation in noisy labels research [34, 25, 22]. Accuracy of models is evaluated on a test set with clean labels.

We compare our proposed *forward sequence loss* under three settings where we either: Ignore Noise, assume the noise is Static (i.e., best case performance of existing approaches) or model the Temporal noise. In our first set of experiments, we fit GRU models on each dataset. In the second set, we leverage a popular time series foundation model MOMENT [12] to study the effects of temporal label noise on pre-trained models with and without loss correction.

Detailed setup and results are provided in Appendix E.

4.1 Results and Discussion

We now study the effects of temporal noise when learning models that are more robust to label noise.

[1] A loss that is well-calibrated for probability estimation (e.g., NLL loss) and incorporates a link function mapping model outputs to probability estimates (see [28] for more detail)

		Loss Correction Strategy		
Dataset	Attributes	Ignore Noise (Acc %)	Static (ΔAcc %)	Temporal (ΔAcc %)
HAR[29]	$n = 192, d = 14, T = 50$	83.0±4.8	+8.6±4.8	**+15.1±0.8**
HAR70[23]	$n = 444, d = 6, T = 100$	81.5±1.4	+1.0±1.8	**+11.0±0.3**
EEG_SLEEP[11]	$n = 964, d = 7, T = 100$	76.4±1.7	-3.2±0.9	**+10.7±0.5**
EEG_EYE[30]	$n = 299, d = 14, T = 50$	67.0±3.0	+0.2±3.3	**+5.9±3.3**

Table 1: Acc % when we Ignore Noise and Δ Acc % assuming noise is either Static or Temporal. We report the mean value ± st.dev for each metric over 10 runs. We show results for *Mixed* noise function, which denotes class-conditional noise where one class has increasing noise and one class has decreasing noise over time. In all settings, accounting for temporal noise leads to the most gains in performance.

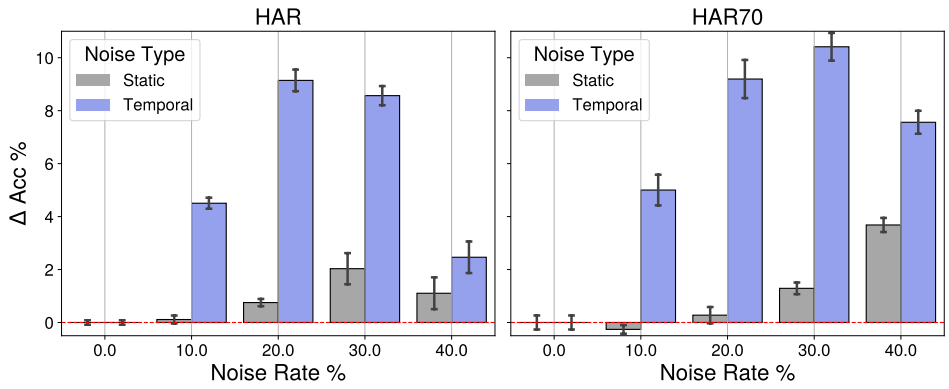

Figure 2: Performance of MOMENT time series foundation model after 25 epochs of fine-tuning using *forward sequence loss*. Δ Acc % measures the improvement in accuracy with loss correction when noise is Static or Temporal. The baseline performance of MOMENT is not dramatically improved when the label noise is Static but it is substantially improved when noise is Temporal. We show results for on `har` and `har70`. We report the mean value ± st.dev for each metric over 10 runs. We show results for *Mixed* noise function, which denotes class-conditional noise where one class has increasing noise and one class has decreasing noise over time.

On the Importance of Modeling Temporal Noise First, we show clear value in accounting for temporal label noise in Table 1. We find that the modelling the Temporal label noise leads to better performing models than assuming the noise is Static (e.g., existing approaches), highlighting the importance of modelling temporal noise.Overall, these findings suggest that we can improve performance by explicitly modeling how noise varies across time instead of assuming it is distributed uniformly in time.

On the Noise Robustness of Time Series Foundation Models In Fig. 2, we find that when label noise is Static, pre-trained foundation models do not substantially benefit from loss correction - indicating that they remain quite noise tolerant without modification. However, when the underlying label noise process is Temporal, loss correction dramatically improves performance.

5 Concluding Remarks

Many classification tasks, especially in healthcare, involve sequential label classification under label noise. While it is known that noisy labels impact static classification, time series labels can experience varying noise rates over time, complicating classification. Existing static methods are not equipped to handle temporal label noise, and our work demonstrates their significant underperformance in such settings. We demonstrate the importance of modelling temporal label noise when training from scratch as well as with time series foundation models.

References

[1] Atkinson, Gentry and Vangelis Metsis. Identifying label noise in time-series datasets. In *Adjunct Proceedings of the 2020 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2020 ACM International Symposium on Wearable Computers*, pages 238–243, 2020.

[2] Beigman, Eyal and Beata Beigman Klebanov. Learning with annotation noise. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 280–287, 2009.

[3] Bengio, Yoshua, Réjean Ducharme, and Pascal Vincent. A neural probabilistic language model. *Advances in Neural Information Processing Systems*, 13, 2000.

[4] Bertrand, Marianne and Sendhil Mullainathan. Do people mean what they say? implications for subjective survey data. *American Economic Review*, 91(2):67–72, 2001.

[5] Bertsekas, Dimitri P. *Constrained optimization and Lagrange multiplier methods*. Academic press, 2014.

[6] Choe, Yo Joong, Jaehyeok Shin, and Neil Spencer. Probabilistic interpretations of recurrent neural networks. *Probabilistic Graphical Models*, 2017.

[7] Choi, Edward, Andy Schuetz, Walter F Stewart, and Jimeng Sun. Using recurrent neural network models for early detection of heart failure onset. *Journal of the American Medical Informatics Association*, 24(2): 361–370, 2017.

[8] Cui, Beilei, Minqing Zhang, Mengya Xu, An Wang, Wu Yuan, and Hongliang Ren. Rectifying noisy labels with sequential prior: Multi-scale temporal feature affinity learning for robust video segmentation. *arXiv preprint arXiv:2307.05898*, 2023.

[9] Duan, Huilong, Zhoujian Sun, Wei Dong, Kunlun He, and Zhengxing Huang. On clinical event prediction in patient treatment trajectory using longitudinal electronic health records. *IEEE Journal of Biomedical and Health Informatics*, 24(7):2053–2063, 2019.

[10] Ghosh, Aritra, Himanshu Kumar, and P Shanti Sastry. Robust loss functions under label noise for deep neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

[11] Goldberger, Ary L, Luis AN Amaral, Leon Glass, Jeffrey M Hausdorff, Plamen Ch Ivanov, Roger G Mark, Joseph E Mietus, George B Moody, Chung-Kang Peng, and H Eugene Stanley. Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals. *Circulation*, 101 (23):e215–e220, 2000.

[12] Goswami, Mononito, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. Moment: A family of open time-series foundation models. In *International Conference on Machine Learning*, 2024.

[13] Han, Bo, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *Advances in Neural Information Processing Systems*, 31, 2018.

[14] Harutyunyan, Hrayr, Kyle Reing, Greg Ver Steeg, and Aram Galstyan. Improving generalization by controlling label-noise information in neural network weights. In *International Conference on Machine Learning*, pages 4071–4081. PMLR, 2020.

[15] Hsieh, Gary and Rafał Kocielnik. You get who you pay for: The impact of incentives on participation bias. In *Proceedings of the 19th ACM Conference on Computer-supported Cooperative work & Social Computing*, pages 823–835, 2016.

[16] Lee, Jeong Min and Milos Hauskrecht. Modeling multivariate clinical event time-series with recurrent temporal mechanisms. *Artificial intelligence in medicine*, 112:102021, 2021.

[17] Li, Shikun, Xiaobo Xia, Hansong Zhang, Yibing Zhan, Shiming Ge, and Tongliang Liu. Estimating noise transition matrix with label correlations for noisy multi-label learning. *Advances in Neural Information Processing Systems*, 35:24184–24198, 2022.

[18] Li, Xuefeng, Tongliang Liu, Bo Han, Gang Niu, and Masashi Sugiyama. Provably end-to-end label-noise learning without anchor points. In *International Conference on Machine Learning*, pages 6403–6413. PMLR, 2021.

[19] Lipton, Zachary C, David C Kale, Charles Elkan, and Randall Wetzel. Learning to diagnose with lstm recurrent neural networks. *arXiv preprint arXiv:1511.03677*, 2015.

[20] Liu, Sheng, Jonathan Niles-Weed, Narges Razavian, and Carlos Fernandez-Granda. Early-learning regularization prevents memorization of noisy labels. *Advances in Neural Information Processing Systems*, 33:20331–20342, 2020.

[21] Liu, Tongliang and Dacheng Tao. Classification with noisy labels by importance reweighting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(3):447–461, 2015.

[22] Liu, Yang and Hongyi Guo. Peer loss functions: Learning from noisy labels without knowing noise rates. In *International Conference on Machine Learning*, pages 6226–6236. PMLR, 2020.

[23] Logacjov, Aleksej and Astrid Ustad. HAR70+. UCI Machine Learning Repository, 2023. DOI: https://doi.org/10.24432/C5CW3D.

[24] Long, Philip M and Rocco A Servedio. Random classification noise defeats all convex potential boosters. In *International Conference on Machine Learning*, pages 608–615, 2008.

[25] Natarajan, Nagarajan, Inderjit S Dhillon, Pradeep K Ravikumar, and Ambuj Tewari. Learning with noisy labels. *Advances in Neural Information Processing Systems*, 26, 2013.

[26] Patrini, Giorgio, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1944–1952, 2017.

[27] Pierson, Emma et al. Daily, weekly, seasonal and menstrual cycles in women's mood, behaviour and vital signs. *Nature Human Behaviour*, 2021.

[28] Reid, Mark D. and Robert C. Williamson. Composite binary losses. *Journal of Machine Learning Research*, 11(83):2387–2422, 2010. URL http://jmlr.org/papers/v11/reid10a.html.

[29] Reyes-Ortiz, Jorge, Davide Anguita, Alessandro Ghio, Luca Oneto, and Xavier Parra. Human Activity Recognition Using Smartphones. UCI Machine Learning Repository, 2012. DOI: https://doi.org/10.24432/C54S4K.

[30] Roesler, Oliver. EEG Eye State. UCI Machine Learning Repository, 2013. DOI: https://doi.org/10.24432/C57G7J.

[31] Sano, Akane and Rosalind W Picard. Stress recognition using wearable sensors and mobile phones. In *2013 Humaine Association Conference on Affective Computing and Intelligent Interaction*, pages 671–676. IEEE, 2013.

[32] Sha, Ying and May D Wang. Interpretable predictions of clinical outcomes with an attention-based recurrent neural network. In *Proceedings of the 8th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, pages 233–240, 2017.

[33] Shamshirband, Shahab, Mahdis Fathi, Abdollah Dehzangi, Anthony Theodore Chronopoulos, and Hamid Alinejad-Rokny. A review on deep learning approaches in healthcare systems: Taxonomies, challenges, and open issues. *Journal of Biomedical Informatics*, 113:103627, 2021.

[34] Sugiyama, Masashi, Tongliang Liu, Bo Han, Yang Liu, and Gang Niu. Learning and mining with noisy labels. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 5152–5155, 2022.

[35] Suo, Qiuling, Fenglong Ma, Giovanni Canino, Jing Gao, Aidong Zhang, Pierangelo Veltri, and Gnasso Agostino. A multi-task framework for monitoring health conditions via attention-based recurrent neural networks. In *AMIA annual symposium proceedings*, volume 2017, page 1665. American Medical Informatics Association, 2017.

[36] Sutskever, Ilya, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27, 2014.

[37] Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[38] Wang, Lu, Wei Zhang, Xiaofeng He, and Hongyuan Zha. Supervised reinforcement learning with recurrent neural network for dynamic treatment recommendation. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2447–2456, 2018.

[39] Wang, Yisen, Xingjun Ma, Zaiyi Chen, Yuan Luo, Jinfeng Yi, and James Bailey. Symmetric cross entropy for robust learning with noisy labels. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 322–330, 2019.

[40] Westbrook, Johanna I, Enrico Coiera, William TM Dunsmuir, Bruce M Brown, Norm Kelk, Richard Paoloni, and Cuong Tran. The impact of interruptions on clinical task completion. *BMJ Quality & Safety*, 19(4):284–289, 2010.

[41] Xia, Xiaobo, Tongliang Liu, Nannan Wang, Bo Han, Chen Gong, Gang Niu, and Masashi Sugiyama. Are anchor points really indispensable in label-noise learning? *Advances in Neural Information Processing Systems*, 32, 2019.

[42] Yao, Yu, Tongliang Liu, Bo Han, Mingming Gong, Jiankang Deng, Gang Niu, and Masashi Sugiyama. Dual t: Reducing estimation error for transition matrix in label-noise learning. *Advances in Neural Information Processing Systems*, 33:7260–7271, 2020.

[43] Yong, LIN, Renjie Pi, Weizhong Zhang, Xiaobo Xia, Jiahui Gao, Xiao Zhou, Tongliang Liu, and Bo Han. A holistic view of label noise transition matrix in deep learning and beyond. In *International Conference on Learning Representations*, 2023.

[44] Yu, Xingrui, Bo Han, Jiangchao Yao, Gang Niu, Ivor Tsang, and Masashi Sugiyama. How does disagreement help generalization against label corruption? In *International Conference on Machine Learning*, pages 7164–7173. PMLR, 2019.

[45] Zhang, Shao, Jianing Yu, Xuhai Xu, Changchang Yin, Yuxuan Lu, Bingsheng Yao, Melanie Tory, Lace M Padilla, Jeffrey Caterino, Ping Zhang, et al. Rethinking human-ai collaboration in complex medical decision making: A case study in sepsis diagnosis. *arXiv preprint arXiv:2309.12368*, 2023.

[46] Zhang, Yivan, Gang Niu, and Masashi Sugiyama. Learning noise transition matrix from only noisy labels via total variation regularization. In *International Conference on Machine Learning*, pages 12501–12512. PMLR, 2021.

A Methods

We have shown how to leverage the temporal label noise function to train models robust to noise. However, the label noise function is generally not available *a priori* and should ideally be estimated from data. In this section we propose a method that simultaneously learns a time series classifier and temporal noise function. Given a noisy dataset, we learn these elements by solving following $\forall t \in [1, T]$:

$$\begin{aligned} \min_{\omega,\theta} \quad & \text{Vol}(\boldsymbol{Q}_\omega(t)) \\ \text{s.t.} \quad & \boldsymbol{Q}_\omega(t)^\top \boldsymbol{h}_\theta(\boldsymbol{x}_{1:t}) = p(\tilde{y}_t \mid \boldsymbol{x}_{1:t}) \end{aligned} \tag{2}$$

This objective — the **Te**mporal **No**ise **R**obust objective (TENOR) —is designed to return a faithful representation of the noise function $\hat{\omega}$ by imposing the *minimum-volume simplex* assumption [18], and a noise-tolerant sequential classifier $\hat{\theta}$ by minimizing the *forward sequence loss* in Def. 2.

Here, the objective minimizes the volume of the noise matrix, denoted as $\text{Vol}(\boldsymbol{Q}_t)$. This returns a matrix $\boldsymbol{Q}_t$, at each time step, that obeys the *minimum-volume simplex* assumption, which is a standard condition used to ensure identifiability in static classification tasks [see e.g., 18, 43]. In practice, the minimum-volume simple assumption ensures that $\boldsymbol{Q}_t$ encloses the noisy conditional data distribution at time t: $p(\tilde{\mathrm{y}}_t \mid \mathbf{x}_{1:t})$. Here containment ensures that the estimated noise matrix could have generated each point in the noisy dataset – i.e., so that the corresponding noisy probabilities $p(\tilde{\mathrm{y}}_t \mid \mathbf{x}_{1:t})$ obey Eq. (1). Our use of this assumption guarantees the identifiability of $\boldsymbol{Q}_t$ when the posterior distribution is sufficiently-scattered over the unit simplex [see also 18, for details].

Implementation The formulation above applies to any generic matrix-valued function according to Def. 1 with parameters ω. Because time series classification tasks can admit many types of temporal label noise functions, we must ensure ω has sufficient representational capacity to handle many noise functions. Therefore in practice, we instantiate our solution as a fully connected neural network with parameters ω, $\boldsymbol{Q}_\omega(\cdot) : \mathbb{R} \to [0, 1]^{c \times c}$, adjusted to meet Def. 1 (see Appendix E.2).

We can now model any temporal label noise function owing to the universal approximation properties of neural networks. Provided the function space Θ defines autoregressive models of the form $p(y_t|\boldsymbol{x}_{1:t})$ (e.g., RNNs, Transformers, etc.), we can solve Eq. (2) using an augmented Lagrangian method for equality-constrained optimization problems [5]:

$$\mathcal{L}(\theta, \omega) = \frac{1}{T} \sum_{t=1}^{T} \left[\|\boldsymbol{Q}_\omega(t)\|_F + \lambda R_t(\theta, \omega) + \frac{c}{2} |R_t(\theta, \omega)|^2 \right] \tag{3}$$

Here: $\|\boldsymbol{Q}_\omega(t)\|_F$ denotes the Frobenius norm of $\boldsymbol{Q}_\omega(t)$, which acts as a convex surrogate for $\text{Vol}(\boldsymbol{Q}_\omega(t))$. Likewise, $R_t(\theta, \omega) = \frac{1}{n} \sum_{i=1}^{n} \ell_t(y_{t,i}, \boldsymbol{Q}_\omega(t)^\top \boldsymbol{h}_\theta(\boldsymbol{x}_{1:t,i}))$ denotes the violation of the equality constraint for each $t = 1 \dots T$. $\lambda \in \mathbb{R}_+$ is the Lagrange multiplier and $c > 0$ is a penalty parameter. Both are initially set to a default value of 1, we gradually increase the penalty parameter until the constraint holds and λ converges to the Lagrangian multiplier of Eq. (2) [5]. This approach recovers the best-fit parameters to the optimization problem in Eq. (2). Additional details on our implementation can be found in Appendix C.

An alternative strategy to learning the noise function, is to assume that there is no temporal relationship between each $\boldsymbol{Q}_t$ across time and treat each time step independently. This is realistic in situations where time steps are unevenly spaced or there are long periods of time between subsequent labels (e.g., some clinical data involves labels collected over years or decades). We can carry out this *discontinuous* estimation by adopting a similar objective as above. Denoting $R_t(\theta) = \frac{1}{n} \sum_{i=1}^{n} \ell_t(y_{t,i}, \hat{\boldsymbol{Q}}_t^\top \boldsymbol{h}_\theta(\boldsymbol{x}_{1:t,i}))$:

$$\mathcal{L}(\theta, [\hat{\boldsymbol{Q}}_t]_{t=1}^T) = \frac{1}{T} \sum_{t=1}^{T} \left[R_t(\theta) + \lambda \cdot \log\det(\hat{\boldsymbol{Q}}_t) \right] \tag{4}$$

The objective in Eq. (4), denoted as VolMinTime, minimizes the volume of $\boldsymbol{Q}$ using the $\log\det$ of a square matrix [18]. In contrast to Eq. (3), each $\hat{\boldsymbol{Q}}_t$ is parameterized with a *separate* set of trainable real-valued weights, which are learned independently with the data from time t using a standard convex optimization algorithm (e.g., gradient descent). This provides a direct time-series modification of a state-of-the-art technique for noise transition matrix estimation in the static setting [18].

A.1 Plug-In Estimation

It is also advantageous to have a simple, plug-in estimator of the temporal noise function. Plug-in estimators are model-agnostic, can flexibly be deployed to other models, and can be efficient to estimate. We can construct a plug-in estimator of temporal label noise using *anchor points*, instances whose labels are known to be correct. Empirical estimates of the class probabilities of anchor points can be used to estimate the noise function. This estimate can then be plugged into any of the aforementioned loss correction methods in **??**. Formally, in a sequential setting, anchor points [21, 26, 41] are instances that maximize the probability of belonging to class i at time step t:

$$\bar{\boldsymbol{x}}_t^i = \arg\max_{\boldsymbol{x}_t} p(\tilde{\mathrm{y}}_t = i \mid \boldsymbol{x}_{1:t}) \tag{5}$$

Since $p(\mathrm{y}_t = i \mid \bar{\boldsymbol{x}}_{1:t}^i) \approx 1$ for the clean label, we can express each entry of the label noise matrix as:

$$\hat{\boldsymbol{Q}}(t)_{i,j} = p(\tilde{\mathrm{y}}_t = j \mid \bar{\boldsymbol{x}}_{1:t}^i) \tag{6}$$

We use a two-step approach, which we call AnchorTime, to get a plug-in estimate of $\hat{\boldsymbol{Q}}(t)$. We identify anchor points for each class $y \in \mathcal{Y}$ and $t = 1, \ldots, T$ and set each entry of $\hat{\boldsymbol{Q}}(t)_{i,j}$ by Eq. (6) (see Appendix D). This is analogous to the approach in static prediction tasks by Patrini et al. [26].

A.2 Discussion

All three methods improve performance in sequential classification tasks by accounting for temporal label noise (see e.g., Fig. 1 and Section 4). However, they each have their own strengths and limitations. Here we provide practical guidance for users to discriminate between methods:

- Joint Continuous Estimation (TENOR) imposes continuity across time steps – assuming that nearby points likely have similar noise levels. This assumption can improve reconstruction, and thus performance, in settings with multiple time steps as it reduces the effective number of model parameters. Conversely, it may also lead to misspecification in settings that exhibit discontinuity.
- Joint Discontinuous Estimation (VolMinTime) can handle discontinuous temporal noise processes, but requires fitting more parameters, which scales according to T – this can lead to computational challenges and overfitting, especially for long sequences.
- Plug-In estimation (AnchorTime) has a simpler optimization problem, useful when separate datasets are used in noise estimation and classifier training, but verifying anchor points is difficult [41].

B Proofs

In what follows, we use vector notation for completeness and clarity of exposition.

We make the following assumptions regarding the conditional time series distribution for the clean labels $p(\mathbf{y}_{1:T} \mid \mathbf{x}_{1:T})$ and noisy labels $p(\tilde{\mathbf{y}}_{1:T} \mid \mathbf{x}_{1:T})$:

We make the following assumptions about the clean data distribution:

Assumption 3. *The clean labels* y_t *at times* $t = 1, \ldots, T$ *are conditionally independent given the features observed up to time* t $\mathbf{x}_{1:t}$.

$$\mathrm{y}_t \perp\!\!\!\perp \mathrm{y}_{1:t-1} \mid \mathbf{x}_{1:t} \tag{7}$$

Assumption 4. *The clean labels* y_t *at time* t *is conditionally independent from* $\mathbf{x}_{t+1}$ *given* $\mathbf{x}_{1:t}$*:*

$$\boldsymbol{p}(\mathbf{y}_{1:T} \mid \mathbf{x}_{1:T}) = \prod_{t=1}^{\top} \boldsymbol{p}(\mathrm{y}_t \mid \mathbf{x}_{1:t}). \tag{8}$$

Assumption 5. *The noisy labels at time* t $\tilde{\mathbf{y}}_t$ *are conditionally independent of* $\mathbf{x}_{1:t}$ *given the clean labels* $\mathbf{y}_t$ *at time* t.

$$\boldsymbol{p}(\tilde{\mathrm{y}}_t \mid \mathbf{x}_{1:t}) = \sum_{c=1}^{C} q_t(\tilde{\mathrm{y}}_t \mid \mathrm{y}_t = c) p(\mathrm{y}_t = c \mid \mathbf{x}_{1:t}) \tag{9}$$

Note that the following property follows from the above assumptions:

$$\boldsymbol{p}(\tilde{\mathbf{y}}_{1:T} \mid \mathbf{x}_{1:T}) = \prod_{t=1}^{\top} \boldsymbol{q}_t(\tilde{\mathrm{y}}_t \mid \mathbf{x}_{1:t}). \tag{10}$$

Definitions We start by defining some of the quantities that will be important for our forward and backward proofs:

$\boldsymbol{p}(\mathrm{y}_t \mid \mathbf{x}_{1:t}) := [p(\mathrm{y}_t = c \mid \mathbf{x}_{1:t})]_{c=1:C}^{\top}, \in \mathbb{R}^{C\times 1}$ (Vector of probabilities for each label value, for the clean label distribution)

$\boldsymbol{p}(\tilde{\mathrm{y}}_t \mid \mathbf{x}_{1:t}) := [p(\mathrm{y}_t = c \mid \mathbf{x}_{1:t})]_{c=1:C}^{\top} \in \mathbb{R}^{C\times 1}$ (Vector of probabilities for each possible label value, for the noisy label distribution)

$\boldsymbol{h}_\theta(\boldsymbol{x}_{1:t}) = \boldsymbol{p}_\theta(\mathrm{y}_t \mid \mathbf{x}_{1:t} = \boldsymbol{x}_{1:t}) : \mathbb{R}^{d\times t} \to \mathbb{R}^C$ (Classifier that predicts label distribution at t given preceding observations)

$\boldsymbol{h}_\theta(\boldsymbol{x}_{1:t}) = \boldsymbol{\psi}^{-1}(\boldsymbol{g}_\theta(\boldsymbol{x}_{1:t}))$ (When h_θ is a deep network, g_θ is the final logits and $\boldsymbol{\psi} : \Delta^{C-1} \to \mathbb{R}^C$ represents an invertible link function whose inverse maps the logits to a valid probability; i.e. a softmax function). We thus assume that

$\boldsymbol{Q}_t := [q_t(\tilde{\mathrm{y}}_t = k \mid \mathrm{y}_t = j)]_{j,k} \in \mathbb{R}^{C\times C}$ (The temporal noise matrix at time t)

$\ell_t(y_t, \boldsymbol{h}_\theta(\boldsymbol{x}_{1:t})) = -\log p_\theta(\mathrm{y}_t = y_t \mid \mathbf{x}_{1:t} = \boldsymbol{x}_{1:t}) : \mathcal{Y} \times \mathbb{R}^C \to \mathbb{R}$ (loss at t)

$\ell_{\psi,t}(y_t, \boldsymbol{h}_\theta(\boldsymbol{x}_{1:t})) = \ell_t(y_t, \boldsymbol{\psi}^{-1}\boldsymbol{h}_\theta(\boldsymbol{x}_{1:t}))$ (A composite loss function is a loss function that uses the aid of a link function: $\boldsymbol{\psi}$)

$\boldsymbol{\ell}_t(\boldsymbol{h}_\theta(\boldsymbol{x}_{1:t})) = [\ell_t(c, \boldsymbol{h}_\theta(\boldsymbol{x}_{1:t})]_{c=1:C}^{\top} : \mathbb{R}^C \to \mathbb{R}^C$ (vector of NLL losses, for each possible value of the ground truth)

$\overrightarrow{\ell}_{t,\psi}(c, \boldsymbol{h}_\theta(\boldsymbol{x}_{1:t})) = \ell_t(c, \boldsymbol{Q}_t^{\top} \cdot \boldsymbol{\psi}^{-1}(\boldsymbol{g}_\theta))$

$\overrightarrow{\ell}_{seq,\psi}(\boldsymbol{y}_{1:T}, \boldsymbol{h}_\theta(\boldsymbol{x}_{1:t})) = \sum_{t=1}^{\top} \overrightarrow{\ell}_{t,\psi}(c, \boldsymbol{h}_\theta(\boldsymbol{x}_{1:t}))$

Theorem 2. $\operatorname{argmin}_\theta \mathbb{E}_{\tilde{\mathbf{y}}_{1:T},\mathbf{x}_{1:T}} \overrightarrow{\ell}_{seq,\psi}(\mathbf{y}_{1:T}, \boldsymbol{g}_\theta(\mathbf{x}_{1:T})) = \operatorname{argmin}_\theta \sum_{t=1}^{\top} \mathbb{E}_{\mathbf{y}_{1:t},\mathbf{x}_{1:t}} \ell_{t,\phi}(\mathbf{y}_{1:T}, \boldsymbol{g}_\theta(\mathbf{x}_{1:T}))$.

Proof. First, note that:

$$\begin{aligned}
\overrightarrow{\ell}_{t,\psi}(\mathrm{y}_t, \boldsymbol{h}_\theta(\mathbf{x}_{1:t})) &= \ell_t(\mathrm{y}_t, \boldsymbol{Q}_t^{\top}\boldsymbol{\psi}^{-1}(\boldsymbol{g}_\theta(\mathbf{x}_{1:t}))) && (11)\\
&= \ell_{\boldsymbol{\phi}_t,t}(\mathrm{y}_t, \boldsymbol{g}_\theta(\mathbf{x}_{1:t})), && (12)
\end{aligned}$$

where $\boldsymbol{\phi}_t^{-1} = \boldsymbol{\psi}^{-1} \circ \boldsymbol{Q}_t^{\top}$. Thus, $\boldsymbol{\phi}_t : \Delta^{C-1} \to \mathbb{R}^C$ is invertible, and is thus a proper composite loss [28].

Thus, as shown in Patrini et al. [26]:

$$\begin{aligned}
\operatorname*{argmin}_\theta \mathbb{E}_{\tilde{\mathrm{y}}_t,\mathbf{x}_{1:t}} \ell_{\boldsymbol{\phi},t}(\mathrm{y}_t, \boldsymbol{g}_\theta(\mathbf{x}_{1:t})) &= \operatorname*{argmin}_\theta \mathbb{E}_{\tilde{\mathrm{y}}_t|\mathbf{x}_{1:t}} \ell_{\boldsymbol{\phi}_t,t}(\mathrm{y}_t, \boldsymbol{g}_\theta(\mathbf{x}_{1:t})) && (13)\\
&= \boldsymbol{\phi}_t(\boldsymbol{p}(\tilde{\mathrm{y}}_t \mid \mathbf{x}_{1:t})) \qquad \text{(property of proper composite losses)}\\
&= \boldsymbol{\psi}((\boldsymbol{Q}_t^{-1})^{\top}\boldsymbol{p}(\tilde{\mathrm{y}}_t \mid \mathbf{x}_{1:t}))) && (14)\\
&= \boldsymbol{\psi}(\boldsymbol{p}(\mathrm{y}_t \mid \mathbf{x}_{1:t})) && (15)
\end{aligned}$$

The above holds for the minimizer at a single time step, not the sequence as a whole. To find the minimizer of the loss over the entire sequence:

$$\operatorname*{argmin}_{\theta} \mathbb{E}_{\mathbf{x}_{1:T},\tilde{\mathbf{y}}_{1:T}} \overrightarrow{\ell}_{seq,\psi}(\tilde{\mathbf{y}}_{1:T}, \boldsymbol{g}_\theta(\mathbf{x}_{1:T})) = \operatorname*{argmin}_{\theta} \mathbb{E}_{\tilde{\mathbf{y}}_{1:T}|\mathbf{x}_{1:T}} \overrightarrow{\ell}_{seq,\psi}(\tilde{\mathbf{y}}_{1:T}, \boldsymbol{g}_\theta(\mathbf{x}_{1:T})) \tag{16}$$

$$= \operatorname*{argmin}_{\theta} \mathbb{E}_{\tilde{\mathbf{y}}_{1:T}|\mathbf{x}_{1:T}} \sum_{t=1}^{\top} \overrightarrow{\ell}_{t,\psi}(\tilde{\mathrm{y}}_t, \boldsymbol{g}_\theta(\mathbf{x}_{1:t})) \tag{17}$$

$$= \operatorname*{argmin}_{\theta} \sum_{t=1}^{\top} \mathbb{E}_{\tilde{\mathbf{y}}_{1:T}|\mathbf{x}_{1:T}} \overrightarrow{\ell}_{t,\psi}(\tilde{\mathrm{y}}_t, \boldsymbol{g}_\theta(\mathbf{x}_{1:t})) \tag{18}$$

$$= \operatorname*{argmin}_{\theta} \sum_{t=1}^{\top} \mathbb{E}_{\tilde{\mathrm{y}}_t|\mathbf{x}_{1:t}} \overrightarrow{\ell}_{t,\psi}(\tilde{\mathrm{y}}_t, \boldsymbol{g}_\theta(\mathbf{x}_{1:t})) \tag{19}$$

$$= \operatorname*{argmin}_{\theta} \sum_{t=1}^{\top} \mathbb{E}_{\tilde{\mathrm{y}}_t|\mathbf{x}_{1:t}} \ell_{t,\phi}(\tilde{\mathrm{y}}_t, \boldsymbol{g}_\theta(\mathbf{x}_{1:t})) \tag{20}$$

As the minimizer of the sum will be the function that minimizes each element of the sum, then $\operatorname{argmin}_\theta \mathbb{E}_{\tilde{\mathbf{y}}_{1:T},\mathbf{x}_{1:T}} \overrightarrow{\ell}_{seq,\psi}(\mathbf{y}_{1:T}, \boldsymbol{g}_\theta(\mathbf{x}_{1:T})) = \boldsymbol{\psi}(\boldsymbol{p}(\mathbf{y}_{1:T} \mid \mathbf{x}_{1:T}))$. Note that the $\operatorname{argmin}_\theta \sum_{t=1}^{\top} \mathbb{E}_{\mathbf{y}_{1:t},\mathbf{x}_{1:t}} \ell_{t,\phi}(\mathbf{y}_{1:T}, \boldsymbol{g}_\theta(\mathbf{x}_{1:T})) = \boldsymbol{\psi}(\boldsymbol{p}(\mathbf{y}_{1:T} \mid \mathbf{x}_{1:T}))$, because the minimizer of the NLL is the data distribution. Thus, $\operatorname{argmin}_\theta \mathbb{E}_{\tilde{\mathbf{y}}_{1:T},\mathbf{x}_{1:T}} \overrightarrow{\ell}_{seq,\psi}(\mathbf{y}_{1:T}, \boldsymbol{g}_\theta(\mathbf{x}_{1:T})) = \operatorname{argmin}_\theta \sum_{t=1}^{\top} \mathbb{E}_{\mathbf{y}_{1:t},\mathbf{x}_{1:t}} \ell_{t,\phi}(\mathbf{y}_{1:T}, \boldsymbol{g}_\theta(\mathbf{x}_{1:T}))$.

□

C TENOR Learning Algorithm

We summarize the augmented Lagrangian approach to solving the TENOR objective in Algorithm 1

Algorithm 1 TENOR Learning Algorithm

Input: Noisy Training Dataset D, hyperparameters γ and η
Output: Model θ, Temporal Noise Function ω

```
c ← 1 and λ ← 1
for k = 1, 2, 3, ..., do
    θ^k, ω^k = arg min_{θ,ω} L(θ, ω)        ▷ Computed with SGD using the Adam optimizer
    λ ← λ + c * R_t(θ^k, ω^k)               ▷ Update Lagrange multiplier
    if k > 0 and R_t(θ^k, ω^k) > γ R_t(θ^{k-1}, ω^{k-1}) then
        c ← ηc
    else
        c ← c
    end if
    if R_t(θ^k, ω^k) == 0 then
        break
    end if
end for
```

For all experiments we set $\lambda = 1$,$c = 1$, $\gamma = 2$, and $\eta = 2$. k and the maximum number of SGD iterations are set to 15 and 10, respectively. This is to ensure that the total number of epochs is 150, which is the max number of epochs used for all experiments.

D AnchorTime Procedure

1. Fit a probabilistic classifier to predict noisy labels from the observed data.
2. For each class $y \in \mathcal{Y}$ and time $t \in [1 \dots T]$:
   i Identify anchor points for class y: $\bar{\boldsymbol{x}}_t^j = \arg\max_{\boldsymbol{x}_t} p(\tilde{\mathrm{y}}_t = y \mid \boldsymbol{x}_{1:t})$.
   ii Set $\hat{\boldsymbol{Q}}(t)_{y,y'}$ as the probability of classifier predicting class y' at time t given $\bar{\boldsymbol{x}}_t^j$.

E Experimental Details

Our code is available in an anonymized repository.

Dataset	Classification Task	n	d	T
`eeg_eye` [30]	Eye Open vs Eye Closed	299	14	50
`eeg_sleep` [11]	Sleep vs Awake	964	7	100
`har` [29]	Walking vs Not Walking	192	9	50
`har70` [23]	Walking vs Not Walking	444	6	100
`synth`	[describe model in notation]	1,000	50	100

Table 2: Datasets used in the experiments. Classification tasks, number of samples (n), dimensionality at each time step (d), and sequence length (T) are shown.

E.1 Dataset Details

Synthetic We generate data for binary and multiclass classification with $n = 1\,000$ samples and $d = 50$ features over $T = 100$ time steps. We generate the class labels and obvservations for each time step using a Hidden Markov Model (HMM). The transition matrix generating the markov chain is uniform ensuring an equal likelihood of any state at any given time. We corrupted them using multidimensional (50) Gaussian emissions. The mean of the gaussian for state/class c is set to c with variance 1.5 (i.e. class 1 has mean 1 and variance 1.5). The high-dimensionality and overlap in feature-space between classes makes this a sufficiently difficult task, especially under label noise. We use a batchsize of 256

HAR from UC Irvine [29] consists of inertial sensor readings of 30 adult subjects performing activities of daily living. The sensor signals are already preprocessed and a vector of features at each time step are provided. We apply z-score normalization at the participant-level, then split the dataset into subsequences of a fixed size 50. We use a batchsize of 64.

HAR70 from UC Irvine [23] consists of inertial sensor readings of 18 elderly subjects performing activities of daily living. The sensor signals are already preprocessed and a vector of features at each time step are provided. We apply z-score normalization at the participant-level, then split the dataset into subsequences of a fixed size 100. We use a batchsize of 256.

EEG SLEEP from Physionet [11] consists of EEG data measured from 197 different whole nights of sleep observation, including awake periods at the start, end, and intermittently. We apply z-score normalization at the whole night-level. Then downsample the data to have features and labels each minute, as EEG data is sampled at 100Hz and labels are sampled at 1Hz. We then split the data into subsequences of a fixed size 100. We use a batchsize of 512.

EEG EYE from UC Irvine [30] consists of data measured from one continuous participant tasked with opening and closing their eyes while wearing a headset to measure their EEG data . We apply z-score normalization for the entire sequence, remove outliers (>5 SD away from mean), and split into subsequences of a fixed size 50. We use a batchsize of 128.

E.2 Specific Implementation Details

GRU the GRU $\boldsymbol{r} : \mathbb{R}^d \times \mathbb{Z} \to \mathbb{R}^C \times \mathbb{Z}$ produces an *output vector* such that the output of $\boldsymbol{r}(\boldsymbol{x}_t, \boldsymbol{z}_{t-1})$ is our model for $\boldsymbol{h}_\theta(\boldsymbol{x}_{1:t})$, and a *hidden state* $\boldsymbol{z}_t \in \mathbb{Z}$ that summarizes $\boldsymbol{x}_{1:t}$. We use a softmax activation on the output vector of the GRU to make it a valid parameterization of $\boldsymbol{p}_\theta(y_t \mid \boldsymbol{x}_{1:t})$. The GRU has a single hidden layer with a 32 dimension hidden state.

TENOR TENOR uses an additional fully-connected neural network with 10 hidden layers that outputs a $C * C$-dimensional vector to represent each entry of a flattened $\hat{\boldsymbol{Q}}_t$. To ensure the output of this network is valid for Def. 1, we reshape the prediction to be $C \times C$, apply a row-wise softmax function, add this to the identity matrix to ensure diagonal dominance, then rescale the rows to be

row-stochastic. These operations are all differentiable, ensuring we can optimize this network with standard backpropagation.

VolMinNet and VolMinTime We do a similar parameterization for VolMinNet and VolMinTime, using a set of differentiable weights to represent the entries of $\boldsymbol{Q}_t$ rather than a neural network.

Anchor and AnchorTime Patrini et al. [26] show that in practice taking the 97th percentile anchor points rather than the maximum yield better results, so we use that same approach in our experiments. They also describe a two-stage approach: 1) estimate the anchor points after a warmup period 2) use the anchor points to train the classifier with forward corrected loss. We set the warmup period to 25 epochs.

E.3 Experimental Parameters

Given that the learning algorithm only has access to a noisy training dataset and performance is evaluated on a clean test set, a validation set must be drawn from clean test data or by manually cleaning the noisy training dataset which may be impractical. This makes hyperparameter tuning difficult in noisy label learning. As the optimal set of hyperparameters within each could vary for each method, noise type, amount of noise, and dataset, this represents a difficult task. To be fair for our experimental evaluations, we use the same set of hyperparameters for experiment, and only manually set batch size for each dataset.

Each model was trained for 150 epochs using the adam optimizer with default parameters and a learning rate of 0.01.

For VolMinNet, VolMinTime, and TENOR we use adam optimizer with default parameters and a learning rate of 0.01 to optimize each respective $\hat{\boldsymbol{Q}}_t$-estimation technique. λ was set to $1e-4$ for VolMinNet and VolMinTime for all experiments, based on what was published previously [18] .

E.4 Noise Injection

To the best of our knowledge there are no noisy label time series datasets (i.e.: standardized datasets with both clean and noisy labels) to evaluate our methods. In line with prior experimental approaches, we propose a noise injection strategy which assumes some temporal noise function that can give us a noisy distribution to evaluate from. We deliberately pick a wide variety of noise types, varying the amount and functional form of time-dependent noise, including static noise setting (uniform noise at every time, akin to what baseline methods assume), and class-dependent noise structure Fig. 3.

Figure 3: Temporal functions that can be specified using a temporal label noise function $\boldsymbol{Q}(t)$. We present six examples for binary classification task (from top-left clockwise): time independent, exponential decay sinusoidal noise, mixed class-dependent noise, linear decay noise, sigmoid increasing noise. Each plot shows the off-diagonal entries of various parameterized forms of $\boldsymbol{Q}(t)$.