

000  
001  
002  
003  
004  
005  
006  
007  
008  
009  
010  
011  
012

# STOCHASTIC SPARSE SAMPLING: A FRAMEWORK FOR VARIABLE-LENGTH MEDICAL TIME SERIES CLASSIFICATION

013  
014  
015  
016  
017  
018  
019  
020  
021  
022  
023  
024  
025  
026  
027  
028  
029  
030

**Anonymous authors**  
Paper under double-blind review

## ABSTRACT

While the majority of time series classification research has focused on modeling fixed-length sequences, variable-length time series classification (VTSC) remains critical in healthcare, where sequence length may vary among patients and events. To address this challenge, we propose Stochastic Sparse Sampling (SSS), a novel VTSC framework developed for medical time series. SSS manages variable-length sequences by sparsely sampling fixed windows to compute local predictions, which are then aggregated and calibrated to form a global prediction. We apply SSS to the task of seizure onset zone (SOZ) localization, a critical VTSC problem requiring identification of seizure-inducing brain regions from variable-length electrophysiological time series. We evaluate our method on the Epilepsy iEEG Multicenter Dataset, a heterogeneous collection of intracranial electroencephalography (iEEG) recordings obtained from four independent medical centers. SSS demonstrates superior performance compared to state-of-the-art (SOTA) baselines across most medical centers, and superior performance on all out-of-distribution (OOD) unseen medical centers. Additionally, SSS naturally provides post-hoc insights into local signal characteristics related to the SOZ, by visualizing temporally averaged local predictions throughout the signal.

031  
032  
033  
034  
035  
036  
037  
038  
039  
040  
041  
042  
043  
044  
045  
046

## 1 INTRODUCTION

Artificial intelligence (AI) in medicine has received significant attention in recent years, with various applications to clinical diagnosis and treatment planning (Rajpurkar et al., 2022). Despite its advancements, the actual integration into everyday clinical practice remains limited, with much of it attributed to the challenges of handling the complexity and variability in medical data. One particularly challenging aspect of this variability lies in the nature of medical time series data. Variable-length time series are prevalent throughout many areas of healthcare, including heart rate monitoring, blood glucose measurements, and electrophysiological recordings where sequence length can vary dependent on the recording or length of an event (Agliari et al., 2020; Deutsch et al., 1994; Walther et al., 2023). Yet, the majority of time series classification (TSC) literature focuses solely on methods that process fixed-length sequences (Ismail Fawaz et al., 2019; Mohammadi Foumani et al., 2024).

At the same time, healthcare applications require greater interpretability from modern time series methods to expand their applicability in critical domains and accelerate clinical adoption (Amann et al., 2020). This interpretability is especially crucial in contexts where the relationship between pathology and signal characteristics is not well understood, as it can provide valuable insights for both clinicians and scientists. Recent studies in time series classification (TSC) have explored the explainability of specific signal segments, as opposed to full-signal analysis, which proves particularly useful for uncovering important characteristics

such as motifs, anomalies, or frequency patterns (Early et al., 2024; Crabbé & Van Der Schaar, 2021; Huang et al., 2024). However, there still remains a significant need for models with built-in interpretability in medical applications. Such methods would alleviate the burden of implementing both a base model and a specialized interpretability method—which may require more domain expertise—and may more effectively facilitate clinical adoption.

The need for variable-length time series classification (VSTC) methods with built-in interpretability is particularly relevant in seizure onset zone (SOZ) localization—the task of identifying brain regions from which seizures originate—as effective treatment requires analysis of variable-length signals (Balaji & Parhi, 2022). The World Health Organization (WHO) reports epilepsy affects over 50 million people globally, establishing it as one of the most common yet poorly understood neurological disorders (Organization et al., 2019; Stafstrom & Carmant, 2015). Additionally, one-third of patients do not respond to antiepileptic drugs, making surgery the last resort and accurate SOZ localization essential for effectively planning the operation. The process of SOZ identification involves a two-step procedure: initial implantation of electrodes in areas suspected to contain the SOZ, followed by recording and visual analysis of intracranial electroencephalography (iEEG) signals by medical experts. The task of SOZ localization reduces to classifying individual electrode recordings, representing different regions within the brain. Effective localization of the SOZ is challenging due to the absence of clinically validated biological markers and the variable-length nature of iEEG signals—consequently, surgical success rates range from 30% to 70% (Löscher et al., 2020; Li et al., 2021).

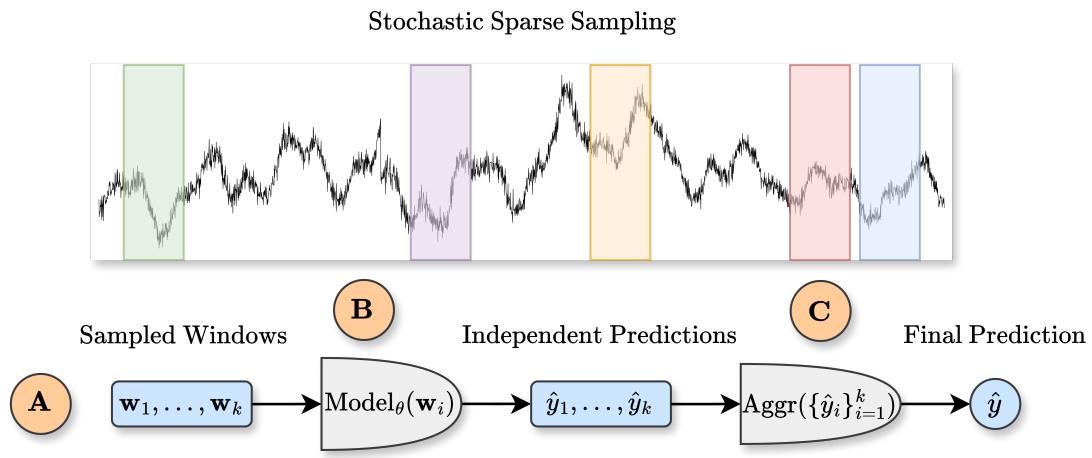


Figure 1: An overview of Stochastic Sparse Sampling (SSS) training procedure. **(A)** For a given time series, we sample windows of fixed-length at random throughout the signal. **(B)** Each window is processed independently by a local model with parameters  $\theta$ , outputting the local predictions  $\hat{y}_1, \dots, \hat{y}_k$ . **(C)** Local predictions are then fed through an aggregation function to form the final prediction  $\hat{y}$ .

**Contributions.** While our work primarily focuses on VTSC, we also evaluate our method’s performance on OOD data and explore its potential for providing local explanations. To this end, we propose Stochastic Sparse Sampling (SSS) a novel framework for VTSC developed for medical time series. The main contributions of our paper are listed as follows:

- **Robustness to long and variable-length sequences.** SSS samples fixed-length windows, and processes them independently through a single model. This prevents context overload in long sequences seen in infinite-context methods, and does not utilize padding, truncation, or interpolation

094 required by finite-context methods. By relying on a single local model, SSS utilizes far fewer  
 095 parameters compared to finite-context methods that traditionally ingest the entire signal, which  
 096 significantly reduces computational cost during training and the risk of overfitting over long  
 097 sequences.  
 098

- 099 • **Generalization to unseen patient populations.** SSS demonstrates strong performance on  
 100 out-of-distribution (OOD) data from unseen medical centers. When trained on data from one or  
 101 more medical centers and evaluated on a completely new center with a different patient population,  
 102 SSS outperforms all baselines in our comparisons. This result suggests SSS’s potential as a  
 103 foundation model for TSC, opening new avenues for research and clinical applications.
- 104 • **Explainability through local predictions:** Our method enhances model interpretability by directly  
 105 tying each output—a probability score for each window—to the overall prediction. This capability  
 106 is crucial in critical clinical settings, such as SOZ localization, which traditionally relies on visual  
 107 analysis. Given the significant risks associated with brain region removal, any proposal should be  
 108 designed to integrate within clinical workflows. Moreover, in the absence of universally recognized  
 109 biological markers for epilepsy, SSS offers the potential to further our understanding the SOZ and to  
 110 identify novel markers.
- 111 • **Compatibility with modern and classical backbones.** SSS integrates with any time series backbone.  
 112 This ensures that our approach leverages well-established frameworks now and into the future,  
 113 allowing for adaptability across a diverse array of contexts.

## 116 2 RELATED WORK

117 TSC methodologies can be broadly categorized into finite-context methods, which operate on fixed-length  
 118 input segments, and infinite-context methods, which handle variable-length sequences without being restricted  
 119 to a predetermined window size. For a formal treatment, please see Appendix A.5.

120 **Finite-context methods.** Finite-context methods are among the most commonly used approaches for TSC.  
 121 Transformer-based models have gained significant attention, with variations such as sparse attention, series  
 122 decomposition, and patching techniques (Vaswani et al., 2017; Kitaev et al., 2020; Zhou et al., 2021; Wu et al.,  
 123 2021; Zhou et al., 2022; Liu et al., 2022; Nie et al., 2023; Liu et al., 2024). Several temporal convolutional  
 124 networks (TCNs) have also been proposed, to capture temporal dependencies through dilated convolutions  
 125 and Inception-like architectures (Lai et al., 2018; Bai et al., 2018; Ismail Fawaz et al., 2020; Wu et al., 2022;  
 126 Luo & Wang, 2024). Recently, multilayer perceptrons (MLPs) and simple linear models have demonstrated  
 127 competitive performance as well (Chen et al., 2023; Zeng et al., 2023). Despite the significant rise of finite-  
 128 context methods, these methods are inherently limited in their ability to handle variable-length sequences, and  
 129 will require the use of either padding, truncation, or interpolation for VTSC. Furthermore, as the sequence  
 130 length increases, so does the number of model parameters, which leads to not only greater computational cost  
 131 but an increased risk of overfitting.

132 **Infinite-context methods.** The recurrent neural network (RNN) family includes several models capable of  
 133 ingesting variable-length time series (Rumelhart et al., 1986). Long-short term memory (LSTM) networks  
 134 introduce memory cells and gating mechanisms to better handle long-term dependencies (Hochreiter &  
 135 Schmidhuber, 1997). Gated recurrent units (GRUs) simplify the LSTM architecture while maintaining  
 136 similar performance (Bahdanau et al., 2014). State space models (SSMs) have gained recent attention,  
 137 with approaches such as S4 introducing structured parameterization to enable efficient computation over  
 138 long sequences, while still attempting to capture long-range dependencies (Gu et al., 2021). Building on  
 139 this, Mamba introduces a selective SSM that adapts to input dynamics, further improving processing of

long-range dependencies in time series data while maintaining linear time complexity with respect to sequence length (Gu & Dao, 2023). Despite these advancements, RNNs and SSMs can still struggle with retaining information in extremely long sequences and may be prone to vanishing or exploding gradients (Salehinejad et al., 2017). ROCKET offers an alternative, using random convolutional kernels to convert input into a fixed-length representation for VTSC, but at the cost of interpretability and potentially limited model expressivity (Dempster et al., 2020).

**SOZ localization methods.** Several recent proposals have been tailored specifically to SOZ localization. Functional connectivity graphs compute patient-specific channel metrics to capture brain connectivity patterns (Grattarola et al., 2022; Fang et al., 2024), offering insights into functional relationships associated with seizures. However, their reliance on intra-patient dynamics makes them unsuitable for a single model that can generalize across multi-patient, heterogeneous datasets. Alternatively, electrical stimulation methods that use intracranial electrodes (Johnson et al., 2022; Yang et al., 2024) can enhance localization accuracy through induced responses analyzed by TCNs and logistic regression models. Yet, these approaches require both fixed-length windows and the use of active stimulation. For our purpose of building a general model for SOZ localization, which can be applied to multiple patients (with a potentially varying number of channels) without electrical stimulation, we do not consider such approaches in our study.

### 3 METHOD

#### 3.1 VARIABLE-LENGTH TIME SERIES CLASSIFICATION

Consider a collection of time series  $X = \{(\mathbf{x}_t^{(1)})_{t=1}^{T_1}, \dots, (\mathbf{x}_t^{(n)})_{t=1}^{T_n}\}$  with labels  $Y = \{y^{(1)}, \dots, y^{(n)}\}$ , where each series  $i$  has sequence length  $T_i \in \mathbb{N}$ , and for each time point  $t$ , the vector  $\mathbf{x}_t^{(i)} \in \mathbb{R}^{M_i}$  has  $M_i \in \mathbb{N}$  channels. The goal of VTSC is to learn a classifier  $f_\theta$  which maps each series  $(\mathbf{x}_t^{(i)})_{t=1}^{T_i}$  to its corresponding class in  $\{1, \dots, K\}$  for  $K \in \mathbb{N}$  classes. We require that  $f_\theta$  can handle sequences of any length—that is, it has infinite context—since we assume that each  $T_i$  can be arbitrarily large at inference time. Otherwise, we must adjust a finite-context classifier using padding, truncation, or interpolation.

#### 3.2 STOCHASTIC SPARSE SAMPLING

##### 3.2.1 SPARSE TRAINING

Figure 1 provides an overview of SSS at train time. During each training epoch, SSS performs a sampling procedure without replacement to create each batch. Fix  $L \in \mathbb{N}$  as the window size and let  $\mathcal{W}$  be the collection of all windows with size  $L$  from all time series in  $X$ . Within any batch, a window is drawn from  $\mathcal{W}$ , where the probability of it originating from series  $i$  is set to  $p_i \approx T_i / (\sum_{j=1}^n T_j)$  for every  $i$ . More formally, for each  $i$ , let  $N_i$  be the random variable representing the number of windows from series  $i$  in a batch of size  $B$ . Then  $N_i \sim \text{Binomial}(B, p_i)$ , and consequently  $\mathbb{E}[N_i] = Bp_i$ . This proportional sampling ensures fair representation of each series based on its length, allowing longer sequences—which contain more information—to contribute more samples. By sampling only a subset of windows, SSS introduces sparsity into the training process, reducing computational cost found in finite-context methods, and the likelihood of context overload in infinite-context methods. Also note that by sampling with replacement, the model sees each window exactly once during a single training epoch.

After sampling a batch of windows  $\mathcal{W}_0 = \{\mathbf{w}_1, \dots, \mathbf{w}_B\}$ , each  $\mathbf{w}_b \in \mathcal{W}_0$  is processed independently by a local model  $f_\theta$  to obtain a local prediction  $\hat{y}_b = f_\theta(\mathbf{w}_b) \in [0, 1]^K$ , representing our probability distribution over  $K \in \mathbb{N}$  classes. The choice of  $f_\theta$  can be any time series backbone, in our experiments we select PatchTST (Nie et al., 2023). For each time series  $1 \leq i \leq n$ , denote:

$$\mathcal{W}_i = \{\mathbf{w} \in \mathcal{W}_0 \mid \mathbf{w} \text{ is from series } i\}, \quad (1)$$

as the collection of windows in the batch originating from series  $i$ , and let:

$$\mathcal{Y}_i = \{f_\theta(\mathbf{w}) \mid \mathbf{w} \in \mathcal{W}_i\}, \quad (2)$$

be the set of multiset of window probabilities. To obtain the global prediction for time series  $i$ , we aggregate window probabilities from all examples originating from it, given by:

$$\hat{y}^{(i)} = \text{Aggr}(\mathcal{Y}_i) = \sum_{\hat{y} \in \mathcal{Y}_i} \alpha(\hat{y}) \hat{y}, \quad (3)$$

where each  $\alpha(\hat{y}) \in [0, 1]$  represents the weight of window probability  $\hat{y}$  to the final output, satisfying  $\sum_{\hat{y} \in \mathcal{Y}_i} \alpha(\hat{y}) = 1$ ; that is,  $\text{Aggr}(\cdot)$  produces a convex combination over  $\hat{y}_1, \dots, \hat{y}_n$ . In our experiments, we use mean aggregation, i.e.,  $\alpha(\hat{y}) = \frac{1}{|\mathcal{Y}_i|}$  for all  $\hat{y} \in \mathcal{Y}_i$ , due to its simplicity and effectiveness for our current objectives. This formulation guarantees that  $\hat{y}^{(i)}$  remains a valid probability distribution over  $K$  classes (proof in Appendix A.3), and allows for potential of non-uniform aggregation functions, enabling weighting of window predictions based on factors such as prediction uncertainty, or frequency characteristics.

---

**Algorithm 1** SSS Training Algorithm (Single Epoch)

---

**Input:** Time series  $X = \{(\mathbf{x}_t^{(1)})_{t=1}^{T_1}, \dots, (\mathbf{x}_t^{(n)})_{t=1}^{T_n}\}$ , labels  $Y = \{y^{(1)}, \dots, y^{(n)}\}$ , model  $f_\theta$  with parameters  $\theta$ , batch size  $B$   
**Output:** Updated model parameters  $\theta$   
 $\mathcal{W} \leftarrow$  Set of all windows from each series in  $X$   
**while**  $\mathcal{W} \neq \emptyset$  **do**  
    ▷ Sample  $B$  windows with probability  $T_i / (\sum_j T_j)$  from series  $i$ , for all  $i$   
     $\mathcal{W}_0 \leftarrow \text{SAMPLE}(\mathcal{W}, B)$   
    **for**  $i = 1, \dots, n$  **do**  
         $\mathcal{W}_i \leftarrow \{\mathbf{w} \in \mathcal{W}_0 \mid \mathbf{w} \text{ is from series } i\}$  ▷ Windows from series  $i$   
         $\mathcal{Y}_j \leftarrow \{f_\theta(\mathbf{w}) \mid \mathbf{w} \in \mathcal{W}_j\}$  ▷ Window probabilities for series  $i$   
         $\hat{y}^{(i)} \leftarrow \text{AGGREGATE}(\mathcal{Y}_j)$  ▷ Final probability for series  $i$   
         $\mathcal{L}_{\text{batch}} \leftarrow \frac{1}{n} \sum_{i=1}^n \mathcal{L}(\hat{y}^{(i)}, y^{(i)})$  ▷ Loss over the batch  
         $\theta \leftarrow \text{UPDATE}(\theta, \mathcal{L}_{\text{batch}})$  ▷ Update local model parameters  
         $\mathcal{W} \leftarrow \mathcal{W} \setminus \mathcal{W}_0$  ▷ Remove sampled windows from the pool  
**return**  $\theta$

---

### 3.2.2 INFERENCE

To derive the prediction for a time series  $(\mathbf{x}_t^{(i)})_{t=1}^{T_i}$  at inference time, we utilize all windows from the selected time series, to form its final prediction  $\hat{y}^{(i)}$ . Let  $\mathcal{W}_i$  be the collection of all windows from series  $i$ . We pass each window through the local model to obtain the multiset of window probabilities  $\mathcal{Y}_i$  as shown in Equation (2). Before the aggregation step, we utilize a calibrator  $g_\phi : [0, 1]^K \rightarrow [0, 1]^K$ , which adjusts each individual window probability to reduce the presence of noise, and define:

$$\tilde{\mathcal{Y}}_i = \{g_\phi(\hat{y}) \mid \hat{y} \in \mathcal{Y}_i\}, \quad (4)$$

which is then fed into the final prediction during the aggregation step  $\hat{y}^{(i)} = \text{Aggr}(\tilde{\mathcal{Y}}_i)$ . By calibrating the window probabilities before aggregation, we correct for biases or misesimations in the predicted probabilities, which occur when the output probabilities of the local models do not accurately reflect the true likelihood of the event. We consider isotonic regression and Venn-Abers predictors for our calibration method. It is important to note, that these calibration techniques do not alter underlying structure of  $f_\theta$  and do not utilize input features from the time series; rather, they adjust the output probabilities to mitigate the effect of noise during the aggregation step. For more information regarding calibration methods see Appendix A.4.

---

235    **4 EXPERIMENTS**  
236

237    **4.1 BASELINES**  
238

239    For our finite-context baselines, we include a variety of modern time series backbones including PatchTST,  
240    which uses subwindows as tokens in combination with the traditional Transformer architecture (Nie et al.,  
241    2023; Vaswani et al., 2017). TimesNet is a TCN architecture that models both interperiod and intraperiod  
242    dynamics by leveraging Fast Fourier Transform (FFT) features to slice the signal into multiple views, which  
243    are then processed through inception blocks (Wu et al., 2022; Ismail Fawaz et al., 2020). ModernTCN, is  
244    a recently proposed TCN which decouples temporal and channel information processing by using separate  
245    DWConv and ConvFFN modules for more efficient representation learning (Luo & Wang, 2024). DLinear  
246    is linear neural network, which has shown to outperform several modern Transformer-based architectures,  
247    utilizing traditional seasonal-trend decomposition techniques (Zeng et al., 2023). In our infinite-context  
248    baselines, we utilize ROCKET, which applies randomly initialized, fixed convolutional kernels to the input  
249    sequence. ROCKET compresses the resulting convolutional outputs to the maximum value and the proportion  
250    of positive values (PPV), where these features then fed to a linear classifier (Dempster et al., 2020). We also  
251    consider GRUs and an LSTM network, both of which are popular RNN frameworks designed to capture long-  
252    term dependencies in sequential data (Bahdanau et al., 2014; Hochreiter & Schmidhuber, 1997). Additionally,  
253    use the recent SSM architecture, Mamba, which utilizes selective state updates to enable efficient long-range  
254    dependency modeling (Gu & Dao, 2023). Further details regarding configurations and hyperparameter tuning  
255    for each baseline can be found in Appendix C.  
256

257    **4.2 DATASET**  
258

259    The Epilepsy iEEG Multicenter Dataset<sup>1</sup> consists of iEEG signals with SOZ clinical annotations from  
260    four medical centers including the Johns Hopkins Hospital (JHH), the National Institute of Health (NIH),  
261    University of Maryland Medical Center (UMMC), and University of Miami Jackson Memorial Hospital  
262    (UMH). Since UMH contained only a single patient with clinical SOZ annotations, we did not consider it in  
263    our main evaluations; however, we did use UMH within the multicenter evaluation in 1 and the training set for  
264    OOD experiments for SOZ localization on unseen medical centers in Table 2. We select the F1 score, Area  
265    Under the Receiver Operator Curve (AUC), and accuracy for our evaluation metrics. For summary statistics  
266    and information on the dataset see Appendix B.1.  
267

268    **4.3 UNIVARIATE VTSC**  
269

270    For each patient iEEG recording, the goal of SOZ localization is to determine the the correct of channels  
271    or electrodes which belong the seizure onset zone. This effectively reduces the task to univariate TSC.  
272    While several channel-dependent methods have been proposed for SOZ localization (see section 2), we focus  
273    primarily on channel-independent solutions for two key reasons: (1) they are more resilient to interpatient  
274    variability and are unaffected by factors like channel count and therefore can be applied in multiple hospital  
275    settings, and (2) they generalize better to domains beyond electrophysiological data, as they learn local signal  
276    characteristics rather than explicitly modeling functional connectivity between electrode sites, which may not  
277    be present in other medical time series.  
278

279    Table 1 summarizes our experimental results for SOZ localization on each individual medical center, along  
280    with training and evaluation on all medical centers. Within the multicenter evaluation, SSS outperforms all  
281    baselines for each evaluation metric. SSS also shows strong performance for the JHH and NIH centers, with  
282    comparable results in the UMMC center. We attribute this difference in performance for UMMC due to the  
283    fact that it is the only center where the sampling frequency of patient recordings can differ between patients  
284

---

285    <sup>1</sup><https://openneuro.org/datasets/ds003029/versions/1.0.7>

Table 1: SOZ localization. F1 score, AUC, and Accuracy are reported for each medical center, averaged over 5 seeds. For each center, we train and evaluate a separate model; the first column represents training and evaluation on all centers. **Bolded** values with \* and  $\dagger$  denote the best and second-best results, respectively.

Model	All			JHH			NIH			UMMC		
	F1	AUC	Acc.	F1	AUC	Acc.	F1	AUC	Acc.	F1	AUC	Acc.
<b>SSS (Ours)</b>	<b>0.7629*</b>	<b>0.7999*</b>	<b>72.35*</b>	<b>0.8187*</b>	<b>0.8851*</b>	<b>81.37*</b>	<b>0.6716*</b>	0.6853	<b>64.22<math>\dagger</math></b>	<b>0.7978<math>\dagger</math></b>	<b>0.8279<math>\dagger</math></b>	76.06
PatchTST (Nie et al., 2023)	<b>0.7097<math>\dagger</math></b>	<b>0.7852<math>\dagger</math></b>	66.83	<b>0.7419<math>\dagger</math></b>	<b>0.8045<math>\dagger</math></b>	71.82	0.6402	<b>0.7036<math>\dagger</math></b>	62.11	<b>0.8015*</b>	0.8121	77.58
TimesNet (Wu et al., 2022)	0.6897	0.7174	65.98	0.6891	0.8029	<b>73.64<math>\dagger</math></b>	0.5950	0.6806	<b>66.00*</b>	0.7821	0.8099	<b>77.06<math>\dagger</math></b>
ModernTCN (Luo & Wang, 2024)	0.6938	0.7305	68.42	0.6710	0.7508	67.73	0.5055	<b>0.7220*</b>	64.00	0.6371	0.8203	71.76
DLinear (Zeng et al., 2023)	0.6916	0.7044	68.41	0.6873	0.7395	66.36	0.6055	0.6405	59.50	0.7658	0.7729	77.05
ROCKET (Dempster et al., 2020)	0.6847	0.7481	69.27	0.6753	0.7752	69.09	<b>0.6520<math>\dagger</math></b>	0.6546	62.63	0.7686	0.7900	74.55
Mamba (Gu & Dao, 2023)	0.6452	0.7134	64.39	0.6456	0.6764	62.27	0.5974	0.6050	58.95	0.7900	<b>0.8424*</b>	76.36
GRUs (Bahdanau et al., 2014)	0.6948	0.7340	65.85	0.6140	0.6959	63.18	0.6171	0.6283	62.63	0.7920	0.8211	<b>77.27*</b>
LSTM (Hochreiter & Schmidhuber, 1997)	0.6709	0.7144	65.43	0.6571	0.6190	59.09	0.5657	0.5909	54.74	0.7604	0.8060	73.64

(250-1000 Hz), whereas JHH and NIH both have sampling frequencies of 1000 Hz. We also observe that in general, finite-context perform better on the chosen evaluation metrics in comparison to infinite-context methods, for in-distribution univariate VTSC.

#### 4.4 OUT-OF-DISTRIBUTION VTSC

Table 2 reports our results for SOZ localization in the OOD setting. At train time, from the collection of four medical center datasets, we omit one and train on the remaining three. At inference time, we test solely on the omitted medical center to gauge how well each method performs OOD. For iEEG signals from epilepsy patients, inter-patient variability can be significant due to differences in placement of electrodes in the brain, and the inherent heterogeneity of epileptogenic networks across individuals. Thus, even among medical time series, this can be one of the most challenging tasks to perform OOD. SSS outperforms each baseline on each unseen medical center, often by a considerable margin when compared to finite-context methods.

Table 2: Out-of-Distribution SOZ localization. F1 score, AUC, and Accuracy are reported for unseen medical centers, averaged over 5 seeds. For each center, we train on all other centers and evaluate on the selected center. **Bolded** values with \* and  $\dagger$  denote the best and second-best results, respectively.

Model	JHH			NIH			UMMC		
	F1	AUC	Acc.	F1	AUC	Acc.	F1	AUC	Acc.
<b>SSS (Ours)</b>	<b>0.6981*</b>	<b>0.6590*</b>	<b>57.80*</b>	<b>0.6492*</b>	<b>0.6092*</b>	<b>54.73<math>\dagger</math></b>	<b>0.7243*</b>	<b>0.8048*</b>	<b>72.42*</b>
PatchTST (Nie et al., 2023)	0.6175	0.5267	50.46	0.5986	0.4829	48.17	0.5067	0.5274	57.63
TimesNet (Wu et al., 2022)	0.5261	0.4501	47.00	0.4461	0.4407	45.85	0.3177	0.3108	46.14
ModernTCN (Luo & Wang, 2024)	0.4934	0.4970	49.54	0.4019	0.4651	48.71	0.3804	0.4474	50.55
DLinear (Zeng et al., 2023)	0.4205	0.4775	47.25	0.5090	0.4945	50.54	0.5602	0.5236	56.00
ROCKET (Dempster et al., 2020)	0.5784	0.5777	<b>56.71<math>\dagger</math></b>	0.5051	0.5522	52.91	0.5608	0.5941	58.36
Mamba (Gu & Dao, 2023)	0.5790	<b>0.5835<math>\dagger</math></b>	55.68	<b>0.6183<math>\dagger</math></b>	<b>0.5767<math>\dagger</math></b>	<b>55.69*</b>	0.5715	0.5953	55.76
GRUs (Bahdanau et al., 2014)	0.5779	0.4868	48.80	0.5824	0.5588	53.66	<b>0.6689<math>\dagger</math></b>	<b>0.7645<math>\dagger</math></b>	<b>69.30<math>\dagger</math></b>
LSTM (Hochreiter & Schmidhuber, 1997)	<b>0.6362<math>\dagger</math></b>	0.5165	50.92	0.5774	0.5678	53.87	0.6581	0.6616	62.36

In comparison to Table 1, we observe the opposite trend between finite- and infinite-context methods, whereas infinite-context methods seem to perform better OOD. In contrast, SSS demonstrates robust performance both in distribution (where finite-context methods excel) and OOD (where infinite-context methods excel).

#### 4.5 QUALITATIVE VISUALIZATION

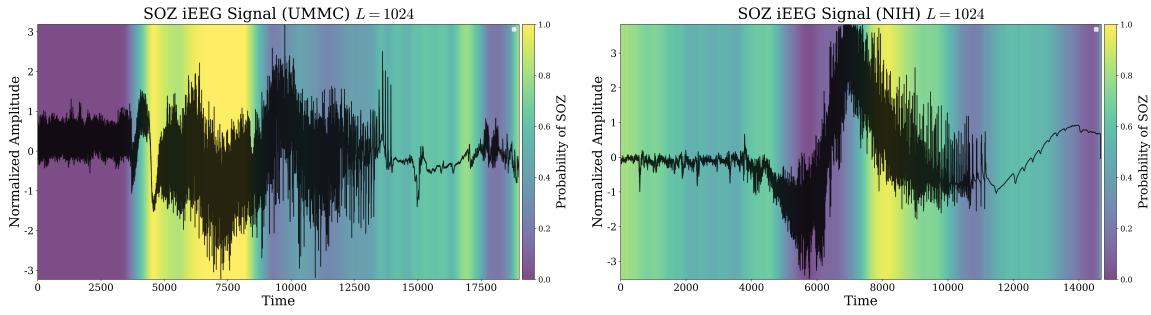


Figure 2: Visualization of SSS window probabilities throughout iEEG channels at inference time, using the PatchTST backbone with window size 1024. The heatmap represents locally averaged window probabilities over time, with color intensity being proportional to the likelihood of the channel belonging to the SOZ.

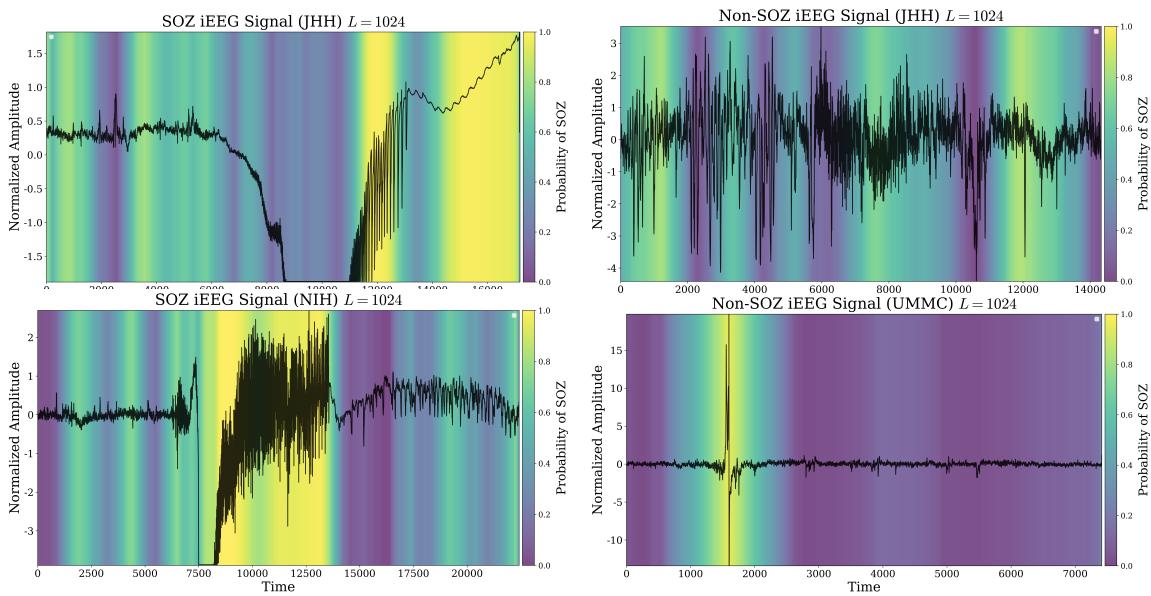


Figure 3: Visualization of SSS window probabilities for OOD iEEG channels at inference time, using the PatchTST backbone with window size 1024. The heatmap represents locally averaged window probabilities over time, with color intensity being proportional to the likelihood of the channel belonging to the SOZ.

376  
377

## 5 DISCUSSION

378  
379

### 5.1 REVIEW OF RESULTS

380  
381  
382  
383  
384  
385

In general, we observe that SSS outperforms modern finite-context and infinite-context methods both in-distribution and OOD for univariate VTSC. SOZ localization presents a significant challenge due to intra-patient and inter-patient variability, and with our evaluation the collection of 3 heterogeneous datasets, serving rigorous testbed for assessing the generalizability of SSS’s capabilities for learning local signal characteristics in medical time series. While our experiments focus on univariate VTSC, as outlined in section 3.2.1 and 3.2.2, SSS can be easily applied to the multivariate setting.

386  
387  
388  
389  
390  
391  
392  
393  
394  
395

Our results from the multicenter evaluation Table 1 indicate that SSS benefits from a diversity of data distributions and volume of training examples, given by the magnitude in performance differences, when compared to single cluster results. Furthermore, our OOD experiments from Table 2 suggest that SSS may be learning local signal characteristics present in different patient populations, leading to a advantage over finite-context and infinite-context methods. Our visualizations of SSS’s predictions OOD in Figure 3 supports this notion, as there exist clear qualitative differences in locally averaged window probabilities with respect to anomalous signal characteristics, such as spikes or increases in amplitude or frequency. Figure 2 shows SSS’s predictions in-distribution which also suggest a form of implicit semantic segmentation for anomalous local regions of the signal with respect to the SOZ probability. More analysis is needed to further solidify our understanding, which would benefit from a rigorous explainability study in future works.

396  
397

### 5.2 CONCLUSION

398  
399  
400  
401  
402  
403  
404  
405  
406  
407

To conclude, this work introduces novel VTSC framework, Stochastic Sparse Sampling (SSS), specifically tailored for medical time series applications. SSS blends the best of both worlds between finite-context methods (enabling usage of finite-context backbones) while allowing sampling of the entire signal in a computationally efficient manner that is less prone to context overload from infinite-context methods. SSS learns local signal characteristics, which provides the added benefit of inherent interpretability, and provides superior performance to the SOTA in-distritbuion and OOD for unseen medical centers. For future work, it would be valuable to: (1) benchmark SSS across a wider variety of variable-length medical time series, (2) provide further rigorous post-hoc insights into the window probability distribution given by SSS, and (3) potentially include uncertainty estimates within the aggregation function to highlight anomalous regions of the signal.

408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422

423 REFERENCES  
424

- 425 Elena Agliari, Adriano Barra, Orazio Antonio Barra, Alberto Fachechi, Lorenzo Franceschi Vento, and Luciano Moretti.  
426 Detecting cardiac pathologies via machine learning on heart-rate variability time series and related markers. *Scientific*  
427 *Reports*, 10(1):8845, 2020.
- 428 Julia Amann, Alessandro Blasimme, Effy Vayena, Dietmar Frey, Vince I Madai, and Precise4Q Consortium. Explainability  
429 for artificial intelligence in healthcare: a multidisciplinary perspective. *BMC medical informatics and decision making*,  
430 20:1–9, 2020.
- 431 Anastasios N Angelopoulos and Stephen Bates. A gentle introduction to conformal prediction and distribution-free  
432 uncertainty quantification. *arXiv preprint arXiv:2107.07511*, 2021.
- 433 Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and  
434 translate. *arXiv preprint arXiv:1409.0473*, 2014.
- 435 Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks  
436 for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.
- 437 Sai Sanjay Balaji and Keshab K Parhi. Seizure onset zone identification from ieeg: A review. *IEEE Access*, 10:  
438 62535–62547, 2022.
- 439 Si-An Chen, Chun-Liang Li, Nate Yoder, Sercan O Arik, and Tomas Pfister. Tsmixer: An all-mlp architecture for time  
440 series forecasting. *arXiv preprint arXiv:2303.06053*, 2023.
- 441 Jonathan Crabbé and Mihaela Van Der Schaar. Explaining time series predictions with dynamic masks. In *International*  
442 *Conference on Machine Learning*, pp. 2166–2177. PMLR, 2021.
- 443 Angus Dempster, François Petitjean, and Geoffrey I Webb. Rocket: exceptionally fast and accurate time series classifica-  
444 tion using random convolutional kernels. *Data Mining and Knowledge Discovery*, 34(5):1454–1495, 2020.
- 445 T Deutsch, ED Lehmann, ER Carson, AV Roudsari, KD Hopkins, and PH Sönksen. Time series analysis and control of  
446 blood glucose levels in diabetic patients. *Computer Methods and Programs in Biomedicine*, 41(3-4):167–182, 1994.
- 447 Joseph Early, Gavin KC Cheung, Kurt Cutajar, Hanting Xie, Jas Kandola, and Niall Twomey. Inherently interpretable  
448 time series classification via multiple instance learning. In *The Twelfth International Conference on Learning*  
449 *Representations*, 2024.
- 450 Chunying Fang, Xingyu Li, Meng Na, Wenhao Jiang, Yuankun He, Aowei Wei, Jie Huang, and Ming Zhou. Epilepsy  
451 lesion localization method based on brain function network. *Frontiers in Human Neuroscience*, 18:1431153, 2024.
- 452 Daniele Grattarola, Lorenzo Livi, Cesare Alippi, Richard Wennberg, and Taufik A Valiante. Seizure localisation with  
453 attention-based graph neural networks. *Expert systems with applications*, 203:117330, 2022.
- 454 Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint*  
455 *arXiv:2312.00752*, 2023.
- 456 Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. *arXiv*  
457 *preprint arXiv:2111.00396*, 2021.
- 458 Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- 459 Qi Huang, Wei Chen, Thomas Bäck, and Niki van Stein. Shapelet-based model-agnostic counterfactual local explanations  
460 for time series classification. *arXiv preprint arXiv:2402.01343*, 2024.
- 461 Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning  
462 for time series classification: a review. *Data mining and knowledge discovery*, 33(4):917–963, 2019.

- 470 Hassan Ismail Fawaz, Benjamin Lucas, Germain Forestier, Charlotte Pelletier, Daniel F Schmidt, Jonathan Weber,  
 471 Geoffrey I Webb, Lhassane Idoumghar, Pierre-Alain Muller, and François Petitjean. Inceptiontime: Finding alexnet for  
 472 time series classification. *Data Mining and Knowledge Discovery*, 34(6):1936–1962, 2020.
- 473 Graham W Johnson, Leon Y Cai, Derek J Doss, Jasmine W Jiang, Aarushi S Negi, Saramati Narasimhan, Danika L Paulo,  
 474 Hernán FJ González, Shawniqua Williams Roberson, Sarah K Bick, et al. Localizing seizure onset zones in surgical  
 475 epilepsy with neurostimulation deep learning. *Journal of neurosurgery*, 138(4):1002–1007, 2022.
- 476 Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. Reversible instance nor-  
 477 malization for accurate time-series forecasting against distribution shift. In *International Conference on Learning  
 478 Representations*, 2021.
- 479 Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- 480 Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. In *International Conference  
 481 on Learning Representations*, 2020. URL <https://openreview.net/forum?id=rkgNKKHtvB>.
- 482 Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with  
 483 deep neural networks. In *The 41st international ACM SIGIR conference on research & development in information  
 484 retrieval*, pp. 95–104, 2018.
- 485 Adam Li, Chester Huynh, Zachary Fitzgerald, Iahn Cajigas, Damian Brusko, Jonathan Jagid, Angel O Claudio, Andres M  
 486 Kanner, Jennifer Hopp, Stephanie Chen, et al. Neural fragility as an eeg marker of the seizure onset zone. *Nature  
 487 neuroscience*, 24(10):1465–1474, 2021.
- 488 Shizhan Liu, Hang Yu, Cong Liao, Jianguo Li, Weiyao Lin, Alex X Liu, and Schahram Dustdar. Pyraformer: Low-  
 489 complexity pyramidal attention for long-range time series modeling and forecasting. In *International Conference on  
 490 Learning Representations*, 2022.
- 491 Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted  
 492 transformers are effective for time series forecasting. In *International Conference on Learning Representations*, 2024.
- 493 Wolfgang Löscher, Heidrun Potschka, Sanjay M Sisodiya, and Annamaria Vezzani. Drug resistance in epilepsy: clinical  
 494 impact, potential mechanisms, and new innovative treatment options. *Pharmacological reviews*, 72(3):606–638, 2020.
- 495 Donghao Luo and Xue Wang. Moderntcn: A modern pure convolution structure for general time series analysis. In *The  
 496 Twelfth International Conference on Learning Representations*, 2024.
- 497 Navid Mohammadi Foumani, Lynn Miller, Chang Wei Tan, Geoffrey I Webb, Germain Forestier, and Mahsa Salehi. Deep  
 498 learning for time series classification and extrinsic regression: A current survey. *ACM Computing Surveys*, 56(9):1–45,  
 499 2024.
- 500 Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term  
 501 forecasting with transformers. In *International Conference on Learning Representations*, 2023.
- 502 World Health Organization et al. *Epilepsy: a public health imperative*. World Health Organization, 2019.
- 503 Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming  
 504 Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library.  
 505 *Advances in neural information processing systems*, 32, 2019.
- 506 Pranav Rajpurkar, Emma Chen, Oishi Banerjee, and Eric J Topol. Ai in health and medicine. *Nature medicine*, 28(1):  
 507 31–38, 2022.
- 508 David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation,  
 509 parallel distributed processing, explorations in the microstructure of cognition, ed. de rumelhart and j. mcclelland. vol.  
 510 1. 1986. *Biometrika*, 71(599-607):6, 1986.
- 511 Hojjat Salehinejad, Sharan Sankar, Joseph Barfett, Errol Colak, and Shahrokh Valaei. Recent advances in recurrent  
 512 neural networks. *arXiv preprint arXiv:1801.01078*, 2017.

- 517 Mervyn J Silvapulle and Pranab Kumar Sen. *Constrained statistical inference: Order, inequality, and shape constraints.*  
 518 John Wiley & Sons, 2011.
- 519
- 520 Carl E Stafstrom and Lionel Carmant. Seizures and epilepsy: an overview for neuroscientists. *Cold Spring Harbor  
 521 perspectives in medicine*, 5(6):a022426, 2015.
- 522 Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia  
 523 Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
- 524 Vladimir Vovk and Ivan Petej. Venn-abers predictors. In *Proceedings of the Thirtieth Conference on Uncertainty in  
 525 Artificial Intelligence*, pp. 829–838, 2014.
- 526
- 527 Dominik Walther, Johannes Viehweg, Jens Haueisen, and Patrick Mäder. A systematic comparison of deep learning  
 528 methods for eeg time series analysis. *Frontiers in Neuroinformatics*, 17:1067095, 2023.
- 529 Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-  
 530 correlation for long-term series forecasting. *Advances in Neural Information Processing Systems*, 34:22419–22430,  
 531 2021.
- 532 Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation  
 533 modeling for general time series analysis. *arXiv preprint arXiv:2210.02186*, 2022.
- 534 Bowen Yang, Baotian Zhao, Chao Li, Jiajie Mo, Zhihao Guo, Zilin Li, Yuan Yao, Xiuliang Fan, Du Cai, Lin Sang,  
 535 et al. Localizing seizure onset zone by a cortico-cortical evoked potentials-based machine learning approach in focal  
 536 epilepsy. *Clinical Neurophysiology*, 158:103–113, 2024.
- 537
- 538 Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? 2023.
- 539
- 540 Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond  
 541 efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial  
 intelligence*, volume 35, pp. 11106–11115, 2021.
- 542
- 543 Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. FEDformer: Frequency enhanced  
 544 decomposed transformer for long-term series forecasting. In *Proc. 39th International Conference on Machine Learning  
 (ICML 2022)*, 2022.
- 545
- 546
- 547
- 548
- 549
- 550
- 551
- 552
- 553
- 554
- 555
- 556
- 557
- 558
- 559
- 560
- 561
- 562
- 563

## 564 A STOCHASTIC SPARSE SAMPLING

### 565 A.1 SAMPLING IMPLEMENTATION

566 Let  $X = \{(\mathbf{x}_t^{(1)})_{t=1}^{T_1}, \dots, (\mathbf{x}_t^{(n)})_{t=1}^{T_n}\}$  be the collection of variable-length time series. To achieve the sampling  
 567 procedure outlined in section 3.2, we first construct the set all windows  $\mathcal{W}$  by performing the slicing window  
 568 method over each individual time series in  $X$ . For a window size  $L$ , with window stride  $S$ , and a time  
 569 series  $i$  with sequence length  $T_i$ , we obtain  $A_i = \lfloor \frac{T_i-L}{S} \rfloor + 1$  windows. Note that  $A_i \propto T_i$  for each  $i$ .  
 570 During training, we convert  $\mathcal{W}$  to a PyTorch dataset and use the native PyTorch dataloader with batch size  
 571  $B$ . When a batch is sampled, windows are drawn uniformly from  $\mathcal{W}$ , and thus for each  $i$ , the probability of  
 572 observing a window from series  $i$  is  $p_i = A_i / (\sum_{j=1}^n A_j) \approx T_i / (\sum_{j=1}^n T_j)$ . If  $N_i$  represents the number of  
 573 windows in the batch from series  $i$ , then we achieve the desired sampling property of  $N_i \sim \text{Binomial}(B, p_i)$   
 574 where  $p_i \approx T_i / (\sum_{j=1}^n T_j)$ . Note that this procedure uses sampling *without* replacement; one may consider  
 575 replacement, however, we did not experiment with this and leave modifications with more complex sampling  
 576 procedures as a future direction.

### 577 A.2 ABLATIONS

#### 578 A.2.1 BATCH SIZE

579 Table 3 reports mean F1 score, AUC, and accuracy  
 580 (%) with standard deviations, are reported over 5  
 581 seeds, for the evaluation on all medical centers. Each  
 582 experiment uses the best configuration described in  
 583 Table 6. While the aggregation function in Equation  
 584 (3) may benefit from a higher number of samples  
 585 within the batch (due to mean approximation), we  
 586 observe that the performance of SSS remains rela-  
 587 tively constant across various batch sizes, and thus  
 588 does not require large batch sizes to achieve ade-  
 589 quate performance.

#### 590 A.2.2 WINDOW SIZE

591 Table 4 follows the same experimental setup as Table  
 592 3, but varies over the window size  $L$  instead of batch  
 593 size. While the performance of  $L = 512$  and  $L =$   
 594 1024 remain relatively on par, we notice that the  
 595 F1 score drops significantly for  $L = 2048$  along  
 596 with all other metrics. This suggests that a large  
 597 receptive field may not be advantageous, and that  
 598 SSS benefits from processing localized areas of the  
 599 signal. Indeed, as  $L$  increases we expect to reach a similar performance to the finite-context PatchTST  
 600 baseline, with a decrease in performance as a result.

#### 601 A.2.3 CALIBRATION

602 Table 4 follows the same experimental setup as Table 3, but varies over the window size  $L$  instead of batch  
 603 size. While the performance of  $L = 512$  and  $L = 1024$  remain relatively on par, we notice that the F1  
 604 score drops significantly for  $L = 2048$  along with all other metrics. This suggests that a large receptive field

Table 3: Performance of SSS over various batch sizes.

Batch Size	F1 Score	AUC	Acc. (%)
128	$0.7563 \pm 0.02717$	$0.8139 \pm 0.04302$	$70.79 \pm 2.323$
512	$0.7498 \pm 0.02354$	$0.7965 \pm 0.03526$	$69.46 \pm 3.109$
2048	$0.7651 \pm 0.03568$	$0.8194 \pm 0.05166$	$71.40 \pm 6.078$
4096	$0.7441 \pm 0.02987$	$0.7979 \pm 0.06818$	$68.09 \pm 4.848$
8192	$0.7629 \pm 0.02829$	$0.7999 \pm 0.05331$	$72.35 \pm 4.965$

Table 4: Performance of SSS over various window sizes  $L$ .

$L$	F1 Score	AUC	Acc. (%)
512	$0.7567 \pm 0.01075$	$0.8141 \pm 0.03054$	$70.62 \pm 2.165$
1024	$0.7629 \pm 0.02829$	$0.7999 \pm 0.05331$	$72.35 \pm 4.965$
2048	$0.7334 \pm 0.03003$	$0.7719 \pm 0.04762$	$68.52 \pm 3.353$

may not be advantageous, and that SSS benefits from processing localized areas of the signal. Indeed, as  $L$  increases we expect to reach a similar performance to the finite-context PatchTST baseline, with a decrease in performance as a result.

### A.3 CONVEX AGGREGATION

**Theorem 1** (Probability Distribution Guarantee). *Fix  $K, n \in \mathbb{N}$ . Suppose  $\alpha_1, \dots, \alpha_n \geq 0$  satisfies  $\sum_{i=1}^n \alpha_i = 1$ , and  $\mathbf{v}_1, \dots, \mathbf{v}_n \in [0, 1]^K$  each satisfy  $\sum_{j=1}^K v_{ik} = 1$  for  $1 \leq i \leq n$ . That is, each  $\mathbf{v}_i = (v_{i1}, v_{i2}, \dots, v_{iK})^T$  represents a valid discrete probability distribution over  $K$  classes. Then the convex combination:*

$$\mathbf{y} = \sum_{i=1}^n \alpha_i \mathbf{v}_i, \quad (5)$$

*also represents a valid discrete probability distribution, satisfying  $\sum_{j=1}^K y_j = 1$ .*

*Proof.* By construction,  $y_j = \sum_{i=1}^n \alpha_i v_{ij}$  for each entry  $1 \leq j \leq K$ . Then  $y_j \geq 0$ , since for each  $1 \leq i \leq n$  and  $1 \leq j \leq K$  we are given that  $\alpha_i \geq 0$  and  $v_{ij} \geq 0$ . Furthermore, we can write:

$$\begin{aligned} \sum_{j=1}^n y_j &= \sum_{j=1}^K \sum_{i=1}^n \alpha_i v_{ij} \\ &= \sum_{i=1}^n \sum_{j=1}^K \alpha_i v_{ij} && \text{(Swap summation order)} \\ &= \sum_{i=1}^n \alpha_i \sum_{j=1}^K v_{ij} \\ &= \sum_{i=1}^n \alpha_i && (\sum_{j=1}^K v_{ij} = 1 \text{ for all } i) \\ &= 1 && (\sum_{i=1}^n \alpha_i = 1) \end{aligned}$$

It follows that since each  $y_j \geq 0$  and  $\sum_{j=1}^K y_j = 1$ , then  $\mathbf{y}$  represents a valid discrete probability distribution over  $K$  classes.  $\square$

### A.4 CALIBRATION

Let  $\hat{y}_1, \dots, \hat{y}_n \in [0, 1]$  be the uncalibrated window probabilities, each with a corresponding binary label  $y_1, \dots, y_n \in \{0, 1\}$  derived from the label of the time series; that is, if  $\hat{y}_i = f_\theta(\mathbf{w}_i)$  for a window  $\mathbf{w}_i$ , then the window label  $y_i$  is inherited from the global time series it was sampled from. The goal of probability calibration is to transform each  $\hat{y}_i$  into  $\tilde{y}_i = g_\phi(\hat{y}_i)$ , such that  $\tilde{y}_i$  represents true likelihood of a class. Within this context, probability calibration can help mitigate the impact of temporal fluctuations and local anomalies by adjusting probabilities for each individual windows. Note that while

Table 5: Performance of SSS with different calibration methods.

Calibration Method	F1 Score	AUC	Acc. (%)
Isotonic Regression	$0.7629 \pm 0.02829$	$0.7999 \pm 0.05331$	$72.35 \pm 4.965$
Venn-ABERS	$0.7637 \pm 0.02704$	$0.8003 \pm 0.05308$	$72.47 \pm 4.773$
No Calibration	$0.7291 \pm 0.06909$	$0.7830 \pm 0.03844$	$69.93 \pm 4.635$

658 calibration may yield more refined probability estimates with reduced noise, it is an integral intermediate step  
 659 rather than a global optimization procedure on the final probabilities. For each calibration we considered, we  
 660 provide a short description below.

661 **Isotonic regression** is a nonparametric method that fits a weighted least-squares model subject to monotonicity  
 662 constraints (Silvapulle & Sen, 2011). Formally, this can be stated as a quadratic program (QP) given by:

$$664 \min_g \sum_{i=1}^n w_i (\tilde{y}_i - y_i)^2 \text{ subject to } \tilde{y}_i \leq \tilde{y}_j \text{ for all } i, j \text{ where } \hat{y}_i \leq \hat{y}_j. \quad (6)$$

668 where  $\tilde{y}_i = g(\hat{y}_i)$  and  $w_i \geq 0$  are weights assigned to each datapoint, which are often each set to  $w_i = 1$  to  
 669 provide equal importance over all inputs. The monotonicity constraint ensures that uncalibrated probabilities  
 670 will always map to equal or higher calibrated probabilities. Due its nonparametric nature, isotonic regression  
 671 can adapt to various probability distributions across diverse datasets. However, this flexibility comes at a  
 672 cost as it is also prone to overfitting on smaller datasets, potentially adjusting to noise rather than properly  
 673 calibrating its inputs.

674 **Venn-Abers predictors** is based on the concept of isotonic regression but extends it to ensure validity within  
 675 the framework of conformal prediction, which provides uncertainty estimates with distribution-free theoretical  
 676 guarantees (Vovk & Petej, 2014; Angelopoulos & Bates, 2021). For an uncalibrated probability  $\hat{y}$ , two  
 677 isotonic calibrators are trained:

$$679 p_0 = g_0(\hat{y}) \text{ and } p_1 = g_1(\hat{y}) \quad (7)$$

680 where  $g_0$  and  $g_1$  are isotonic functions derived from augmented sets. These sets include  $(\hat{y}, 0)$  and  $(\hat{y}, 1)$   
 681 respectively, alongside all other uncalibrated probabilities and their respective labels. The values  $p_0$  and  $p_1$   
 682 represent likelihoods for class 0 and class 1, while the interval  $[p_0, p_1]$  provides an uncertainty estimate of  
 683 where the true probability resides. The final calibrated probability is then given by:

$$685 \tilde{y} = \frac{p_1}{1 - p_0 + p_1}. \quad (8)$$

687 Venn-Abers predictors provide guaranteed validity in terms of calibration, meaning the predicted probabilities  
 688 closely match empirical frequencies. While we do not explicitly utilize the uncertainty interval  $[p_0, p_1]$  (only  
 689 the calibrated score  $\tilde{y}$ ), this method can be effective in scenarios requiring risk assessment or critical tasks  
 690 where rigorous uncertainty estimates are crucial. We leave this as a future direction to implement conformal  
 691 prediction within the context of SSS, to provide uncertainty guarantees based off of window predictions,  
 692 which may be useful for post-hoc interpretability. In comparison to isotonic regression, Venn-Abers can be  
 693 more computationally intensive, as it requires fitting two isotonic functions simultaneously.

## 695 A.5 FINITE-CONTEXT & INFINITE-CONTEXT METHODS

697 **Definition 2.** Let  $\mathcal{X}$  be a vector space over  $\mathbb{R}$  and  $f_\theta : \mathcal{X} \rightarrow \mathcal{Y}$  be a model with parameters  $\theta$  and output  
 698 space  $\mathcal{Y}$ . We say that  $f_\theta$  has **finite-context** if  $\mathcal{X}$  is finite-dimensional, that is, there exists some  $n \in \mathbb{N}$  such  
 699 that  $\mathcal{X} \cong \mathbb{R}^n$  as vector spaces. Whereas  $f_\theta$  is said to have **infinite-context** if  $\mathcal{X} = \mathbb{R}^{(\infty)}$  is the space of real  
 700 number sequences with finite support<sup>2</sup>.

702 Note that this definition refers to the *native* capabilities of  $f_\theta$ , without the usage of data manipulation  
 703 techniques such as padding, truncation, and interpolation. We utilize this formalization to separate our  
 704 baselines, so that we may better understand the advantages and limitations of both.

705 Table 6: Hyperparameter search space for SSS (with PatchTST backbone). Best configuration is highlighted  
 706 in red.  
 707

708           Parameter	709           Search Values
710 $d_{\text{model}}$	{16, 32, 64}
711 $d_{\text{ff}}$	{32, 64, 128}
712           num_heads	{2, 4, 8}
713           num_enc_layers	{1, 2, 3}
714           lr	{ $10^{-4}$ , $10^{-5}$ }
715 $L$	{512, 1024, 2048}
716           batch_size	{2048, 4096, 8192}
717 $g_{\phi}$	{isotonic regression, Venn-Abers predictors}

## 718           A.6 SSS IMPLEMENTATION AND CONFIGURATIONS

## 719           B DATASET AND PREPROCESSING

### 720           B.1 DATASET

721 Table 7: iEEG Multicenter Dataset Summary: For each medical center, we report the total number of patients  
 722 recorded ( $n$ ), the number of patients with seizure onset zone (SOZ) annotations ( $n_{\text{SOZ}}$ ), the number of time  
 723 series recordings ( $n_{\text{ts}}$ ), the percentage of time series labeled as SOZ ( $p_{\text{SOZ}}$ ), the type of iEEG method used  
 724 (e.g., electrocorticography, ECoG), the sampling frequency (Hz) (noting that some recordings may vary), and  
 725 post-operative patient outcomes following SOZ surgical resection.

726           Medical Center	727 $n$	728 $n_{\text{SOZ}}$	729 $n_{\text{ts}}$	730 $p_{\text{SOZ}}$	731           iEEG Type	732           Frequency (Hz)	733           Patient Outcomes
JHH	7	3	1458	7.48%	ECoG	1000	No
NIH	14	11	3057	12.23%	ECoG	1000	Yes
UMMC	9	9	2967	5.56%	ECoG	250-1000	Yes
UMF	5	1	129	25.58%	ECoG	1000	No

734 Table 7 provides an overview of the iEEG Multicenter Dataset. For each cluster, we filter out patients ( $n$ )  
 735 who have SOZ annotations ( $n_{\text{SOZ}}$ ). All channels for all patients are group together into one dataset per  
 736 medical center, where  $n_{\text{ts}}$  indicates the number of examples. However, due to the heavy imbalance between  
 737 SOZ-labeled time series and non-SOZ labeled time series, the number of examples used for training and  
 738 validation decreases significantly once we employ class balancing, resulting in  $\lfloor 2 \cdot p_{\text{SOZ}} \cdot n_{\text{ts}} \rfloor$  examples for  
 739 each medical center, which is then split into training, validation, and testing.

### 740           B.2 DATA PREPROCESSING

741 Each patient recording contains multiple channels corresponding to individual electrodes from the iEEG  
 742 device. During preprocessing, for all patients, we extract all channels and balance the dataset to have an equal  
 743 number of SOZ and non-SOZ channels. After, we partition this dataset into train, validation, and test channels  
 744 with a 70%/10%/20% split respectively, and ensure that during the window sampling phase of SSS there is  
 745 no temporal leakage from the test set. Each channel, or univariate time series, is  $z$ -score normalized to have  
 746 zero mean and unit standard deviation.

747           <sup>2</sup>Every sequence in  $\mathbb{R}^{(\infty)}$  must have finitely many non-zero terms.

752 Due to the extremely long sequence length of several channels, we required downsampling to fit the dataset  
 753 into memory (250GB RAM). To achieve this, for each channel we applied a 1D average pooling layer with  
 754 `kernel_size=24` and `kernel_stride=12` before feeding it to the baseline model or before performing  
 755 the window sampling procedure for SSS at train-time.

756 Finite-context methods required either padding, truncation, or interpolation due to fit each sequence into its  
 757 limited context window. For each finite-context method we perform a combination of padding and truncation  
 758 according to the chosen window size  $L$ : if the sequence length of the original time series exceeded  $L$ , it was  
 759 truncated to obtain the first  $L$  time points, otherwise if it was less than  $L$ , it was padded with zeros at the end  
 760 of the sequence to ensure a sequence length of  $L$ .

### 762 B.3 EVALUATION METRICS

#### 764 B.3.1 F1

765 The F1 score is defined as the harmonic mean of precision and recall, given by:

$$767 \quad F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}, \quad (9)$$

769 where,

$$771 \quad \text{Precision} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Positives (FP)}}, \quad (10)$$

773 and,

$$775 \quad \text{Recall} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Negatives (FN)}}. \quad (11)$$

777 We select the F1 score as our primary evaluation metric for SOZ localization for the following reasons.  
 778 The F1 score balances the need to correctly identify all regions of the SOZ (recall) with the need to avoid  
 779 misclassifying healthy regions as SOZ (precision). This balance is crucial in surgical planning, where both  
 780 missing SOZs and unnecessarily removing healthy tissue can have severe consequences. Unlike accuracy,  
 781 which overlooks the difference between false positives and false negatives, the F1 score provides a more  
 782 nuanced evaluation by considering both, making it well-suited in clinical contexts such as SOZ localization.

#### 783 B.3.2 AUC

785 We complement the F1 score with the Area Under the Receiver Operating Characteristic curve (AUC), defined  
 786 by:

$$787 \quad \text{AUC} = \int_0^1 \text{TPR}(t) \cdot \frac{d\text{FPR}(t)}{dt} dt \quad (12)$$

790 While the F1 score provides insight into the balance between precision and recall at a specific threshold,  
 791 AUC assesses the model's overall discriminative ability across all thresholds. This threshold-independent  
 792 evaluation is relevant for critical scenarios where the threshold maybe be adjusted from 0.5, which is not  
 793 common in clinical settings.

## 794 C IMPLEMENTATION AND EXPERIMENTAL CONFIGURATIONS

797 For each baseline, we perform grid search and optimize with respect to best accuracy score on the evaluation  
 798 for all medical centers.  $L$  refers to the window size parameter,  $d_{\text{model}}$  is the model dimension, and  $d_{\text{ff}}$

is the dimension of the feed-forward network. The grid search parameters for each baseline are shown below; for information on the implementation of SSS, see Appendix A.6. In all experiments, we train using the Adam optimizer (Kingma & Ba, 2014), for 50 epochs, with cosine learning rate annealing (one cycle with 50 epochs in length) which adjusts the learning rate down by two orders of magnitude (e.g.,  $10^{-4}$  to  $10^{-6}$ ) by the last epoch. We also implement early stopping with a patience of 15, and apply learnable instance normalization (Kim et al., 2021) for each input. For most of the baselines we use a dropout rate of 0.2 – 0.3, and weight decay to  $10^{-4}$  –  $10^{-5}$ , but do not explicitly tune these parameters in our grid search. For finite-context methods we set the batch size to the entire dataset (596 individual univariate time series for all clusters), whereas infinite-context methods required batch size of 1 due to their variable-length. The code for SSS and baseline implementations is available at the following anonymous link: <https://anonymous.4open.science/r/ssss-0D75/>.

**PatchTST:** We adapt the official implementation [github.com/yuqinie98/PatchTST](https://github.com/yuqinie98/PatchTST), but swap out the attention module with the native PyTorch `torch.nn.MultiheadAttention` module.

Table 8: Hyperparameter search space for PatchTST. Best configuration is highlighted in red.

Parameter	Search Values
$d_{\text{model}}$	{16, 32, 64}
$d_{\text{ff}}$	{32, 64, 128}
num_heads	{2, 4, 8}
num_enc_layers	{1, 2, 3}
lr	{ $10^{-4}$ , $10^{-5}$ }
$L$	{1000, 3000, 5000, 10000}

**TimesNet:** We use the official implementation [github.com/thuml/Time-Series-Library](https://github.com/thuml/Time-Series-Library).

Table 9: Hyperparameter search space for TimesNet. Best configuration is highlighted in red.

Parameter	Search Values
$d_{\text{model}}$	{16, 32, 64}
$d_{\text{ff}}$	{32, 64, 128}
num_kernels	{4, 6}
top_k	{3, 5}
num_enc_layers	{1, 2}
lr	{ $10^{-4}$ , $10^{-5}$ }
$L$	{1000, 3000, 5000, 10000}

846 **ModernTCN:** We use the official implementation [github.com/luodhhh/ModernTCN](https://github.com/luodhhh/ModernTCN).  
 847

848 Table 10: Hyperparameter search space for ModernTCN. Best configuration is highlighted in red.  
 849

Parameter	Search Values
lr	{ $10^{-4}$ , $10^{-5}$ }
$d_{model}$	{16, $\textcolor{red}{32}$ , 64}
num_enc_layers	{1, 2}
large_size_kernel	{9, 13, 21, 51}
small_size_kernel	5
dw_dims	{128, 256}
ffn_ratio	{1, 4}
$L$	{1000, 3000, 5000, $\textcolor{red}{10000}$ }

860 **DLinear:** We use the implementation from [github.com/thuml/Time-Series-Library](https://github.com/thuml/Time-Series-Library).  
 861

862 Table 11: Hyperparameter search space for DLinear. Best configuration is highlighted in red.  
 863

Parameter	Search Values
moving_avg	{10, $\textcolor{red}{25}$ }
lr	{ $10^{-4}$ , $10^{-5}$ , $10^{-6}$ }
$L$	{1000, 3000, 5000, $\textcolor{red}{10000}$ }

864 **ROCKET:** We use the official implementation [github.com/angus924/rocket](https://github.com/angus924/rocket) and follow the standard implementation of 10,000 kernels.  
 865

866 Table 12: Hyperparameter search space for ROCKET. Best configuration is highlighted in red.  
 867

Parameter	Search Values
num_kernels	{ $\textcolor{red}{10000}$ }
lr	{ $10^{-4}$ , $10^{-5}$ , $10^{-6}$ }

868 **Mamba:** We use the package `mambapy` which builds upon the official Mamba implementation. We  
 869

870 Table 13: Hyperparameter search space for Mamba. Best configuration is highlighted in red.  
 871

Parameter	Search Values
lr	{ $10^{-4}$ , $10^{-5}$ , $10^{-6}$ }
$d_{model}$	{16, 32, 64}
num_enc_layers	{1, $\textcolor{red}{2}$ , 3}

880 also employ patching from (Nie et al., 2023), which we observed led to greater performance, with  
 881  $\text{patch\_size} = 64$  and  $\text{patch\_stride} = 16$ .  
 882

883 **GRUs:** We utilized the native PyTorch module `torch.nn.GRU`.  
 884

893           Table 14: Hyperparameter search space for GRUs. Best configuration is highlighted in red.  
 894  
 895

Parameter	Search Values
lr	$\{10^{-4}, 10^{-5}, 10^{-6}\}$
$d_{\text{model}}$	$\{16, 32, 64\}$
num_enc_layers	$\{1, 2, 3\}$
bidirectional	{True, False}

901

902       C.1 COMPUTATIONAL RESOURCES  
 903

904       Our experiments were conducted on 4x NVIDIA RTX 6000 Ada Generation GPUs using the PyTorch  
 905       framework with CUDA Paszke et al. (2019). Although we have not tracked explicitly the amount of consumed  
 906       GPU hours, all experiments can be conducted for 5 seeds no more than 2 – 3 hours with a similar setup.

907

908

909

910

911

912

913

914

915

916

917

918

919

920

921

922

923

924

925

926

927

928

929

930

931

932

933

934

935

936

937

938

939