K-ICL: KERNEL-BASED PAIR MINING FOR INSTANCE-WISE CONTRASTIVE LEARNING OF TIME SERIES

Anonymous authors
Paper under double-blind review

ABSTRACT

Contrastive Learning (CL) is a Self-Supervised Learning (SSL) paradigm that learns representations by attracting *positive* pairs and repelling *negative* ones. SimCLR, a cornerstone CL method, forms positives through label-preserving augmentations of the same instance, and negatives from augmented views of different samples. This instance-wise approach is a key ingredient of many CL solutions, especially in computer vision, but suffers of a major limitation: it is affected by *ill-defined negatives* generated from instances sharing the same semantics. A second limitation specifically concerns time-series, where augmentation can alter the semantic of the instance, producing *ill-defined positives*, which has so far hindered SimCLR in the time-series domain. We address both limitations with K-ICL, a novel probabilistic pair-mining strategy for instance-wise CL that leverages a kernel to measure similarity between time series directly in the input space. Such similarity steers the sampling process towards *well-defined* positive and negative pairs, using only weak augmentations suitable for time-series. Experiments on time-series identification and classification show that K-ICL consistently outperforms SimCLR and its instance-wise extensions, in both few-class and unbalanced settings.

1 INTRODUCTION

Time series are ubiquitous in real-world applications and key in healthcare, cybersecurity, finance, and industrial manufacturing (Foumani et al., 2024). While sensors collect large amounts of time series, labels are scarce as annotation is costly and requires domain expertise. This motivates using Self-Supervised Learning (SSL) methods to learn transferable representations from unlabeled signals (Tonekaboni et al., 2021; Eldele et al., 2021; Yue et al., 2022). Contrastive Learning (CL), one of the most successful SSL paradigms, learns representations by training a Siamese network (Bertinetto et al., 2016) that aligns *positive* pairs and repels *negative* ones. Its most popular representative, SimCLR (Chen et al., 2020), forms *instance-wise* pairs on-the-fly from the minibatch, taking two augmented views from the same sample as a positive pair, and views of other samples as negatives, and trains the encoder with the InfoNCE loss (van den Oord et al., 2018). This instance-wise objective is a core component of many CL methods: more advanced frameworks build on the same instance-wise discrimination, and add on top objectives such as temporal-consistency losses (Yue et al., 2022; Eldele et al., 2021). Improving the pair construction of traditional instance-wise methods, such as SimCLR, is very impactful for the CL literature, given the widespread use of these components.

Instance-wise CL can create False Negatives (FNs) (or *ill-defined negatives*): pairs of distinct samples that belong to the same semantic class but are treated as negatives. Training on these pairs wrongly pushes their representations apart. This problem worsens when classes are few or imbalanced, as samples in a minibatch are more likely to share a class (Chuang et al., 2020; Huynh et al., 2022).

Despite this exposure to FNs, SimCLR and its extensions (Chen et al., 2020; He et al., 2020) succeed in many Computer Vision (CV) applications, where transformations such as rotations, crops, or color jitter preserve the original semantics. Transferring this SSL recipe to time series is challenging, as augmentations are highly domain-dependent: even simple transformations can corrupt discriminative temporal patterns and alter pair semantics, producing *ill-defined positives* (Wen et al., 2021; Luo et al., 2023; Pöppelbaum et al., 2022). As a consequence, SimCLR and its extensions must rely on weak augmentations on time series, which preserve semantics but convey little supervisory signal.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

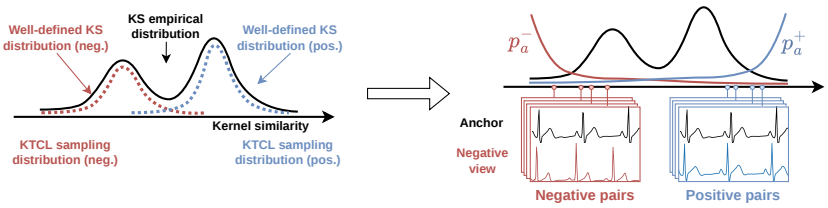


Figure 1: **Top:** the empirical distribution of minibatch kernel similarities (black) mixes two *unknown* distributions (blue and red dashed lines) of well-defined positive and negative pairs. **Bottom:** K-ICL builds two sampling distributions p_a^+ and p_a^- to draw well-defined pairs for a given anchor.

We propose K-ICL, a *kernel-guided pair-mining strategy* for instance-wise CL on time series, particularly beneficial under class imbalance. Our key intuition (Fig. 1) is to measure similarity between time series directly in the input space via a time series kernel. Kernels provide a similarity in the input space that captures structural properties or expert knowledge (Bianchi et al., 2019), reducing the need for strong, domain-specific augmentations. From the empirical distribution of the kernel similarities in each minibatch (black line in Fig. 1), we build two sampling distributions that mine *well-defined* positive pairs (high similarity) and negative pairs (low similarity). These act as surrogates for the unknown distributions of similarities in pairs from the same or different semantic classes (dashed lines in Fig. 1). For the input similarity we default to Global Alignment Kernel (GAK) as a simple alignment-aware choice, well-suited to the temporal misalignment in our datasets; K-ICL, however, supports any kernel κ.

K-ICL is a principled extension of SimCLR to the time-series domain. It is underpinned by a kernel-guided pair-mining strategy that replaces SimCLR's indiscriminate pair construction and integrates seamlessly into instance-wise CL pipelines for time-series. Its effectiveness comes from improving the quality of both the positive and negative pairs used for training: K-ICL significantly reduces the amount of ill-defined negatives, a problem particularly evident in class-imbalanced and few-class regimes, while additionally recovering informative positive pairs that do not originate from the same sample. We support K-ICL with a theoretical analysis (Sec. 3.3) showing that our positive miner is a sound surrogate for an ideal label-aware one, and we validate K-ICL through extensive experiments on time-series identification and classification, in balanced and imbalanced settings (Sec. 4.2), where K-ICL measurably lowers the FN rate of SimCLR, and yields consistently stronger representations than SimCLR and its instance-wise extensions.

2 Related Work

SimCLR (Chen et al., 2020) is one of the first and most relevant CL solutions, and represents a key ingredient of many state-of-the-art contrastive learning frameworks. A fundamental goal in *instance-wise* CL is to improve its pair construction by mining more informative positives and negatives. In CV, several strategies mitigate false negatives, either by debiasing and reweighting negative pairs (Chuang et al., 2020; Robinson et al., 2020) or by detecting and removing suspicious negatives through additional augmented views (Huynh et al., 2022; Xu et al., 2024). A complementary direction instead enriches the *positive* pairs, retrieving additional positives analysing the latent representations (Dwibedi et al., 2021). These strategies, however, either rely on strong label-preserving augmentations or on similarities computed in the evolving latent space, creating a circular dependency when embeddings are unreliable early in training. A few recent strategies derive pair relationships from input-space similarity. SoftCLT (Lee et al., 2024) introduces soft assignments for both instance-wise and temporal contrast; its instance-wise component weights pairs by input-space distance. AdaTS (Kara et al., 2026) derives cross-instance relationships through an ordinal-consistency objective. These instance-wise mining and weighting strategies are the closest alternatives to our method, which is the first using kernel similarity to guide explicit probabilistic pair sampling.

A second line of works in instance-wise CL improves SimCLR by focusing on the augmentations used to form positives. This direction is especially relevant for time series, where augmentations must preserve discriminant temporal patterns, which are highly domain-dependent (Wen et al., 2021; Luo et al., 2023). Indeed, the strong augmentations used in many CV methods are often unavailable, or

they perform poorly in the time series domain. To address this problem, SimCLR has been adapted to time series (Pöppelbaum et al., 2022) and extended through time-frequency consistency (Zhang et al., 2022), mixing-based augmentation (Wickstrøm et al., 2022), or DTW-based transformations (Yang et al., 2022). By acting on the augmentation aspect rather than on pair mining, these methods are complementary to K-ICL, and they could in principle be combined with our mining strategy. We consequently do not compare against them.

A third line uses the *internal temporal structure* of a series to define the learning task. TS2Vec (Yue et al., 2022), for example, combines instance-wise and temporal contrast at multiple temporal resolutions. Notably, such methods retain SimCLR's instance-wise InfoNCE loss as a core component and add temporal objectives on top of it. Thus, any improvement to instance-wise pair construction remain relevant to these. Other methods exploit specific temporal relationships: TNC (Tonekaboni et al., 2021) learns from locally stationary temporal neighborhoods; TS-TCC (Eldele et al., 2021) combines cross-view temporal prediction under weak and strong augmentations with contextual contrast; T-Loss (Franceschi et al., 2019) attracts the representations of a series and its subseries. Further approaches incorporate non-stationarity into contrastive learning (Wu et al., 2024) or learn separate trend and seasonal representations (Woo et al., 2022). These frameworks introduce temporal objectives, improve subseries construction, or present specialized network architectures, thus changing the representation-learning task itself. Also these contributions, referring to temporal representation learning, are complementary to K-ICL: the instance-wise objective in these frameworks could in principle incorporate our kernel-guided mining alongside their temporal objectives.

Finally, SIP-LDL (Jin et al., 2025) addresses false negatives under class imbalance through a *semi-supervised* framework using a few labeled samples. In contrast, K-ICL operates in *fully unsupervised* setting, using input-space kernel similarity to guide pair mining without class labels. Methods that require label supervision during representation learning therefore fall outside our comparison setting.

3 Methodology

3.1 Problem Formulation

Let $\mathcal{S} = \{\mathbf{S}_1, \ldots, \mathbf{S}_N\}$ be a set of unlabeled multivariate time series, where each $\mathbf{S}_i \in \mathbb{R}^{C\times T}$ is a chunk of length T with C channels. The goal of SSL is to train an encoder $\mathcal{E} : \mathbb{R}^{C\times T} \to \mathbb{R}^D$ that maps each time series to a representation $\mathbf{h}_i = \mathcal{E}(\mathbf{S}_i) \in \mathbb{R}^D$, which is useful for further training for downstream tasks such as identification and classification. We train $\mathcal{E}$ by *instance-wise* CL on mini-batches $\mathcal{B} \subset \mathcal{S}$, following the SimCLR paradigm (Chen et al., 2020): each time series is treated as an atomic instance, and the contrastive task operates on whole-instance representations $\mathbf{h}_i = \mathcal{E}(\mathbf{S}_i)$ rather than on their internal sub-segments. For each anchor $\mathbf{S}_a \in \mathcal{B}$, we consider candidate pairs of the form $\{(\mathbf{S}_a, A(\mathbf{S}_j)) \mid \mathbf{S}_j \in \mathcal{B}\}$, where $A(\cdot)$ is a (weak) augmentation operator.

We assume the (unknown) class y_i is a property of the instance $\mathbf{S}_i$ as a whole, to be reflected in its latent representation $\mathbf{h}_i$. Two instances should therefore form a positive pair if they share the same class and a negative pair otherwise, giving the ideal sets: $\mathcal{P}^* = \{(\mathbf{S}_i, \mathbf{S}_j) \mid y_i = y_j\}$, and $\mathcal{N}^* = \{(\mathbf{S}_i, \mathbf{S}_j) \mid y_i \neq y_j\}$, for positive and negative pairs, respectively. A *pair mining strategy* selects, for each anchor $\mathbf{S}_a$ and without access to labels, a collection of positive pairs $\mathcal{P}_a$ and negative pairs $\mathcal{N}_a$. In this unsupervised regime the mined pairs can be ill-defined: positives may connect different semantics (e.g., due to non-label-preserving augmentations), while negatives may include FNs when samples sharing the same semantics are paired. Our objective is to design a mining strategy that reduces ill-defined pairs, either positive or negative, in the challenging regime where only weak augmentations are available, and class imbalance or few classes make naive within-batch sampling unreliable. Concretely, K-ICL should produce $\mathcal{P}_a$ and $\mathcal{N}_a$ as close as possible to the ideal sets $\mathcal{P}^*$ and $\mathcal{N}^*$, using no labels.

3.2 K-ICL: Kernel Instance Contrastive Learning

K-ICL leverages a time-series kernel $\kappa(\cdot, \cdot)$ measuring similarities in the input space, to construct two distinct probability distributions: one for the candidate positive pairs (p_a^+) and one for the negative ones (p_a^-). Subsequently, we leverage these two distributions to mine well-defined positive and negative pairs within each mini-batch $\mathcal{B}$. The mined sets define a contrastive objective that pulls an anchor toward highly similar sequences and pushes it away from dissimilar ones, reducing ill-defined

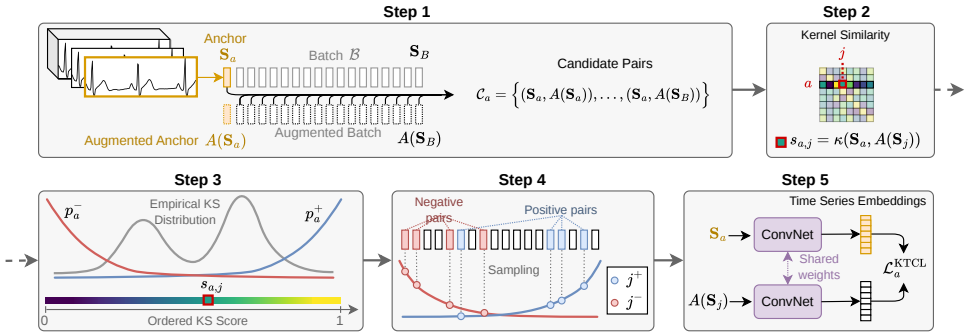


Figure 2: Schematic depiction of the five steps of the proposed K-ICL approach.

negatives without relying on strong augmentations. $\kappa(\cdot, \cdot)$ is used only to guide pair mining; the encoder is trained end-to-end with the InfoNCE loss, and we do not differentiate through the kernel.

K-ICL operates at the *instance* level, addressing the FN problem that standard instance-wise frameworks leave unaddressed, i.e., when multiple negative pairs share the same class. K-ICL relies on kernel similarities computed directly on the inputs, giving a stable notion of similarity throughout training and avoid the circular dependency of representation-based mining (Robinson et al., 2020).

3.2.1 Kernel-guided probabilistic pair mining

In the following, we describe in detail the five-step pipeline of K-ICL, which is depicted in Fig. 2.

Step 1. Batch construction and anchor selection Similar to SimCLR, K-ICL trains an encoder $\mathcal{E}$ within CL framework using mini-batches $\mathcal{B} \subset \mathcal{S}$. Within each batch, one time series is iteratively selected as the *anchor* $\mathbf{S}_a$, and a set of candidate pairs $\mathcal{C}_a$ is generated by applying the (weak) augmentations $A(\cdot)$ to each sample, $\mathcal{C}_a = \{(\mathbf{S}_a, A(\mathbf{S}_j)) \mid \mathbf{S}_j \in \mathcal{B}\}$. The set $\mathcal{C}_a$ contains $|\mathcal{B}|$ candidate pairs for the anchor $\mathbf{S}_a$. We keep the anchor $\mathbf{S}_a$ in its original form and pair it with augmented views of the other samples: preserving the anchor suits the weak-augmentation regime, since augmenting both views risks weakening an already fragile positive pair. We use only weak semantics-preserving perturbations (Wen et al., 2021) (shifting, scaling, jittering, time warping), applied identically to all compared methods. Details are reported in Appendix A.2.

Step 2. Kernel-based similarity computation For each anchor $\mathbf{S}_a$, we compute the input-space similarity against all the other samples in the minibatch by a kernel $\kappa(\cdot, \cdot)$ and define the set of similarities $\mathcal{K}_a = \{s_{a,j} = \kappa(\mathbf{S}_a, A(\mathbf{S}_j)), \forall j \in \mathcal{B}\}$. Note that $\mathcal{K}_a$ includes $s_{a,a} = \kappa(\mathbf{S}_a, A(\mathbf{S}_a))$, the similarity between the anchor and its own augmented view, which serves as a reference high-similarity (positive) pair. Crucially, the similarities in $\mathcal{K}_a$ are independent of the representations learned by $\mathcal{E}$: the kernel depends only on the inputs, not on the training procedure. This allows K-ICL to mine informative pairs since the first epoch, avoiding the vicious circle of selecting hard negatives from premature latent embeddings (Robinson et al., 2020). By default, the within-batch similarities are computed on the fly at each epoch, with no precomputation and no change to training. Since the kernel similarities do not depend on the encoder $\mathcal{E}$, they can optionally be cached and reused across epochs when the augmented views are fixed or cached, by simply tracking inputs to allow reuse while batches are shuffled (see Appendix A.5 for a discussion of computational cost).

Step 3. Construction of positive and negative probability distributions As depicted in Step 3 of Fig. 2, from the empirical distribution (gray curve) of the kernel similarity values in $\mathcal{K}_a$ we construct two discrete probability distributions over time-series similarities: one for mining positives (blue curve) and one for mining negatives (red curve). The positive mining distribution p_a^+ is a categorical distribution over $\mathcal{B}$ whose probability mass function is defined at values $s_{a,j} \in \mathcal{K}_a$ as follows:

$$p_{a,j}^+ = \frac{\exp(s_{a,j}/\tau_{\text{pos}})}{\sum_{k \in \mathcal{B}} \exp(s_{a,k}/\tau_{\text{pos}})}, j \in \mathcal{B}. \tag{1}$$

In practice, we consider $p^+_{a,j}$ as the probability of selecting $(\mathbf{S}_a, A(\mathbf{S}_j))$ as a positive pair, and the parameter $\tau_{\text{pos}} > 0$ controls how peaked the distribution is (smaller values concentrate probability mass on the most similar samples). Similarly, we define the negative mining distribution p^-_a as a categorical distribution with probability mass function:

$$p^-_{a,j} = \frac{\exp(-s_{a,j}/\tau_{\text{neg}})}{\sum_{k\in\mathcal{B}} \exp(-s_{a,k}/\tau_{\text{neg}})}, j \in \mathcal{B} \tag{2}$$

where $p^-_{a,j}$ denotes the probability of selecting $(\mathbf{S}_a, A(\mathbf{S}_j))$ as a negative, and $\tau_{\text{neg}} > 0$ controls the distribution sharpness. Note that, by construction, (1) and (2) are positive and sum to one. Moreover, $p^+_a(\cdot)$ increases as the kernel similarity between time-series becomes larger, while $p^-_a(\cdot)$ behaves in the opposite way (see Step 3 of Fig. 2).

Step 4. Probabilistic mining For each anchor $\mathbf{S}_a$, we draw i.i.d. k_{pos} candidate indices with replacement from the categorical distribution $(p^+_{a,j})_{j\in\mathcal{B}}$, $j^+_1, \ldots, j^+_{k_{\text{pos}}} \overset{\text{i.i.d.}}{\sim} \text{Categorical}(p^+_a)$, where each index $j \in \mathcal{B}$ is a distinct sampling outcome. We denote the resulting collection of sampled positive indices by $\{j^+_m\}_{m=1}^{k_{\text{pos}}}$, which is a multiset, since we sample with replacement. Keeping the self-pair available to the positive miner is intentional: under weak augmentations, it is the only universally valid positive, while the benefit of K-ICL comes from retrieving additional non-self positives as well. This yields the multiset of mined positive pairs $\mathcal{P}_a = \{(\mathbf{S}_a, A(\mathbf{S}_{j^+_m}))\}$, $m = 1, \ldots, k_{\text{pos}}$.

To explicitly prevent overlaps between mined positives and negatives, we mask out from p^-_a the self-pair index a (corresponding to $(\mathbf{S}_a, A(\mathbf{S}_a))$) and the set $\mathcal{I}^+_a$ containing the unique indices sampled in the positive set, and renormalize p^-_a as follows:

$$\tilde{p}^-_{a,j} = \begin{cases} 0 & \text{if } j \in \mathcal{I}^+_a \cup \{a\}, \\ \dfrac{p^-_{a,j}}{\sum_{k\notin\mathcal{I}^+_a\cup\{a\}} p^-_{a,k}} & \text{otherwise.} \end{cases}$$

Then, we similarly sample i.i.d. negative pairs from $\tilde{p}^-_a$ (with replacement) and obtain the mined negative pairs, $j^-_1, \ldots, j^-_{k_{\text{neg}}} \overset{\text{i.i.d.}}{\sim} \text{Categorical}(\tilde{p}^-_a)$. We denote the multiset of sampled negative indices by $\{j^-_m\}_{m=1}^{k_{\text{neg}}}$, from which we generate the multiset of K-ICL negative pairs $\mathcal{N}_a = \{(\mathbf{S}_a, A(\mathbf{S}_{j^-_m}))\}$, $m = 1, \ldots, k_{\text{neg}}$, with distinct-index set $\mathcal{I}^-_a$. We set $k_{\text{pos}} = k_{\text{neg}} = \mathcal{B} - 1$. Because sampling is probabilistic and with replacement, the effective mined sets adapt to the batch rather than being fixed: their composition and multiplicities follow the similarity scores, which keeps the procedure stable when a batch contains few true positives or negatives. By mixing easy and hard examples across training iterations, K-ICL is less sensitive to class imbalance than deterministic top-k or threshold-based mining. We analyze different kernel-based sampling rules in our ablation study.

Step 5. Contrastive learning optimization Pairs in $\mathcal{P}_a$ and $\mathcal{N}_a$ are used to train a Siamese encoder network $\mathcal{E}$ that produces latent embeddings, having $\mathbf{z}_a = \mathcal{E}(\mathbf{S}_a)$ and $\mathbf{z}_j = \mathcal{E}(A(\mathbf{S}_j)) \in \mathbb{R}^D$.

We optimize $\mathcal{E}$ with the following contrastive loss:

$$\mathcal{L}^{\text{KTCL}}_a = -\log \frac{\sum_{p\in\mathcal{I}^+_a} \exp(u_{a,p})}{\sum_{p\in\mathcal{I}^+_a} \exp(u_{a,p}) + \sum_{n\in\mathcal{I}^-_a} \exp(u_{a,n})}, \tag{3}$$

where $u_{aj} = \text{sim}(\mathbf{z}_a, \mathbf{z}_j)/\tau$, where $\text{sim}(\cdot,\cdot)$ is cosine similarity, and $\tau > 0$ is a temperature parameter. Let $\mathcal{I}^+_a$ and $\mathcal{I}^-_a$ denote the sets of *distinct* indices among the sampled positive and negative draws, so that repeated draws do not produce repeated terms to the loss. Our K-ICL loss follows the InfoNCE (van den Oord et al., 2018), pulling closer the representations of positive pairs while pushing apart those of negative pairs. However, K-ICL provides a different construction of these sets. In standard within-batch InfoNCE training, the denominator is built in an agnostic manner and accounts for both well-defined and ill-defined negatives, while the numerator accounts only for the single positive pair. In contrast, K-ICL achieves better distinction of positive ($\mathcal{P}_a$) and negative ($\mathcal{N}_a$) pairs. The

numerator and the denominator depend on the mined index collections $\mathcal{I}_a^+$ and $\mathcal{I}_a^-$ drawn from the kernel-derived distributions p_a^+ and p_a^-. We remark that, differently from SimCLR, we can sample multiple positive pairs per anchor, gathering additional information that is beneficial for CL training. This approach yields a more informed and adaptive contrastive objective and reduces the chance of pairing the anchor into ill-defined negatives under class imbalance.

3.3 Formal guarantees for the probabilistic miner

Here we show that positive miner in (1) is a surrogate for the ideal positive-only miner. Under the assumptions illustrated in Appendix B, let $w_{a,j}^+ = \exp(s_{a,j}/\tau_{\text{pos}})$ and let $W_{\mathcal{P}}$ and $W_{\mathcal{N}}$ denote the positive-softmax mass assigned to true positives and true negatives, respectively. We write μ_{sur} and μ_{pos} for the value of any bounded statistic under the surrogate and ideal positive-only miners.

Theorem 3.1 (Main guarantee) *If positives and negatives are separated in kernel-similarity by a margin $\gamma > 0$, then*

$$\frac{W_{\mathcal{N}}}{W_{\mathcal{P}}} \leq \frac{|\mathcal{N}_a^\star|}{|\mathcal{P}_a^\star|} \exp\left(-\frac{\gamma}{\tau_{\text{pos}}}\right), \tag{4}$$

and, for any bounded statistic f with $|f_i| \leq B$,

$$|\mu_{\text{sur}} - \mu_{\text{pos}}| \leq 2B \cdot \frac{W_{\mathcal{N}}}{W_{\mathcal{P}}} \leq 2B \cdot \frac{|\mathcal{N}_a^\star|}{|\mathcal{P}_a^\star|} \exp\left(-\frac{\gamma}{\tau_{\text{pos}}}\right). \tag{5}$$

Theorem 3.1 shows that the surrogate objective stays close to the ideal positive-only objective whenever kernel-guided mining assigns clearly larger scores to true positives than to true negatives. The dependence on γ/τ_{pos} makes the trade-off explicit: better separation improves the guarantee exponentially, while a smaller temperature sharpens the miner but also makes the bound more sensitive to poor separation. Appendix B provides the full assumptions, proofs, an overlap-aware extension, and an empirical convergence result for stochastic sampling.

4 Experiments

Our experiments show that K-ICL outperforms the SimCLR baseline and competing instance-wise mining and weighting strategies.

4.1 Experimental settings

Datasets. We consider datasets for two downstream tasks: *i*) closed-set user identification on electrocardiogram (ECG) or accelerometer signals and *ii*) multi-class time-series classification on a general-purpose time-series dataset. We assess ECG-based user identification performance on the CEBS dataset (García-González et al., 2013) from Physionet (Goldberger et al., 2000) and accelerometer-based user identification on the PAMAP dataset (Reiss, 2012), restricted to walking segments (8 users, one lacks walking data). We use three UCR benchmarks (PenDigits, JapaneseVowels, and CharacterTrajectories) (Dau et al., 2018) for time-series classification. Details on the datasets, preprocessing, and the evaluation protocol are deferred to Appendix A.

Figures of Merit. We assess the effectiveness of pair mining methods by the precision and recall of the mined positive and negative pairs. We also investigate the structure of the latent space learned by CL when using different mining methods. In particular, we quantify how much the embeddings learned by CL form compact and well-separated clusters that are aligned with the actual class distributions by computing the Adjusted Rand Index (ARI), the Normalized Mutual Information (NMI), and the accuracy of classifiers fine-tuned on a downstream task. For the latter, we freeze the encoder $\mathcal{E}$ and measure the accuracy of a k-Nearest Neighbors (k-NN) and a Multilayer Perceptron (MLP) classifier trained on a small amount of labeled samples per class. More details on the k-NN and MLP configurations are reported in Appendix A. We note that the encoder is trained fully unsupervised via CL, and labels are used only to evaluate the learned representations on the downstream task.

Baselines and competitors. Our comparisons isolate how pair construction and weighting over SimCLR affect representation quality within a shared whole-instance CL pipeline. We therefore

Table 1: Quality of the mined positive and negative sets, SimCLR vs. K-ICL. Prec.$^{\pm}$/Rec.$^{\pm}$ are precision/recall of positives (+) and negatives (-) (%, mean with SEM as subscript, over 10 runs). SimCLR's Prec.$^{+}$ and Rec.$^{-}$ are 100% by construction.

	Dataset	SimCLR				K-ICL			
		Prec.$^{+}$	Rec.$^{+}$	Prec.$^{-}$	Rec.$^{-}$	Prec.$^{+}$	Rec.$^{+}$	Prec.$^{-}$	Rec.$^{-}$
Balanced (5 classes)	CEBS	$100.0_{0.0}$	$14.0_{0.0}$	$80.1_{0.0}$	$100.0_{0.0}$	$73.1_{2.1}$	$64.8_{3.4}$	$98.4_{0.6}$	$47.6_{1.3}$
	PAMAP	$100.0_{0.0}$	$13.9_{0.0}$	$80.0_{0.0}$	$100.0_{0.0}$	$81.8_{1.0}$	$60.6_{1.8}$	$97.9_{0.5}$	$53.8_{0.7}$
	PenDigits	$100.0_{0.0}$	$14.0_{0.0}$	$80.2_{0.0}$	$100.0_{0.0}$	$59.6_{1.3}$	$72.6_{1.4}$	$94.9_{0.6}$	$42.5_{0.4}$
	Jap.Vowels	$100.0_{0.0}$	$13.6_{0.2}$	$79.4_{0.3}$	$100.0_{0.0}$	$74.5_{1.5}$	$50.4_{1.0}$	$95.1_{0.3}$	$50.4_{0.8}$
	Char.Traj.	$100.0_{0.0}$	$14.1_{0.0}$	$80.3_{0.0}$	$100.0_{0.0}$	$80.1_{2.8}$	$68.5_{1.2}$	$96.8_{0.4}$	$50.1_{1.3}$
Unbalanced (10:1)	CEBS	$100.0_{0.0}$	$9.5_{0.1}$	$69.1_{0.2}$	$100.0_{0.0}$	$71.5_{1.3}$	$62.9_{2.0}$	$95.2_{1.3}$	$50.2_{1.1}$
	PAMAP	$100.0_{0.0}$	$7.8_{0.1}$	$61.7_{0.5}$	$100.0_{0.0}$	$85.5_{0.5}$	$54.1_{2.2}$	$92.7_{1.4}$	$65.5_{0.8}$
	PenDigits	$100.0_{0.0}$	$9.5_{0.0}$	$69.4_{0.0}$	$100.0_{0.0}$	$60.4_{0.9}$	$70.8_{1.8}$	$90.5_{1.7}$	$44.3_{0.7}$
	Jap.Vowels	$100.0_{0.0}$	$11.4_{0.4}$	$74.5_{0.9}$	$100.0_{0.0}$	$71.9_{1.4}$	$56.2_{1.5}$	$91.8_{1.1}$	$51.1_{0.9}$
	Char.Traj.	$100.0_{0.0}$	$9.4_{0.1}$	$68.9_{0.2}$	$100.0_{0.0}$	$80.2_{1.5}$	$64.8_{2.0}$	$90.4_{2.6}$	$53.9_{1.8}$

evaluate instance-wise mining and weighting strategies using the same encoder, weak-augmentation protocol, and downstream evaluation. Methods centered on augmentation design or training leveraging the temporal structure, including TS2Vec (Yue et al., 2022) and TS-TCC (Eldele et al., 2021), introduce additional components to this pipeline. Comparing K-ICL with these complete frameworks would therefore prevent us from isolating the contribution of pair mining. Conversely, removing their defining augmentation or temporal mechanisms to fit our protocol would change the methods being evaluated. We consequently exclude these complete frameworks from this controlled comparison.

As references, we include *Base* (a randomly initialized encoder), an *AutoEncoder* trained to reconstruct the input time series, and *Oracle*, an encoder that follows the same pipeline as K-ICL, but replaces unsupervised mining with the label-aware ideal positive/negative sets $\mathcal{P}^*$ and $\mathcal{N}^*$. It therefore acts as an upper bound on pair selection within the same contrastive pipeline.

We further compare against SimCLR and instance-wise CL strategies designed to reduce the impact of FNs during training; some approaches filter out or reweight likely FNs (Chuang et al., 2020; Robinson et al., 2020), while others promote attracting samples that are likely from the same semantic class (Xu et al., 2024; Huynh et al., 2022). We tested both (Xu et al., 2024) and (Huynh et al., 2022) in their *FN Attraction* and *FN Elimination* modalities, denoted as *-Attr* and *-El* in the tables. Closest to K-ICL, SoftCLT (Lee et al., 2024) also derives pair relationships from input-space distances to differently weigh pairs during training. SoftCLT adopts an instance-wise loss, which compares whole instances as in our setting, and a temporal loss, which contrasts representations at different timestamps within a series. As our pipeline operates on whole instances, we adopt only the instance-wise loss, denoted *SoftCLT-I* (SoftCLT's instance-wise soft-assignment). AdaTS (Kara et al., 2026) is not included, as no runnable official implementation is available.

4.2 Results

Contrastive Learning in Class-Balanced Settings. We first evaluate K-ICL in a *class-balanced* setting with five equally represented classes per dataset. Although balanced, this regime already exposes the false-negative problem: with few classes, each batch contains several samples sharing the anchor's class, so a fixed fraction of SimCLR's negatives are ill-defined. As reported in Tab. 1, SimCLR has, by construction, perfect positive precision and negative recall, but its negative precision is only around 80% across datasets, i.e. roughly one in five mined negatives is a false negative, while its positive recall is very low ($\sim 14\%$), as it never recovers same-class samples beyond the anchor's own augmentation. K-ICL instead reaches negative precision above 94% on every dataset and substantially increases positive recall by retrieving additional cross-instance positive pairs, which reduce reliance on possibly wrong strong augmentations. The effective mining strategy of K-ICL translates into consistently better representations (Tab. 2): K-ICL obtains the best clustering (ARI, NMI) and k-NN scores on all considered datasets, improving over SimCLR by a large margin (e.g. ARI from .53 to .72 on CEBS and from .40 to .58 on PenDigits) and approaching the label-aware Oracle. The FN-mitigation alternatives achieve competitive results in this balanced regime, but K-ICL still outperforms them on all metrics but the MLP accuracy, where the gap narrows as a high-capacity classifier partly compensates for differences in representation quality.

Table 2: Clustering and classification metrics in the balanced setting with 5 classes. Values are mean with SEM as subscript, over 10 runs for each dataset. Best per column in **bold**, second-best underlined; the supervised Oracle is excluded from the ranking.

Method	CEBS				PAMAP				PenDigits				Jap.Vowels				Char.Traj.			
	ARI	NMI	kNN	MLP	ARI	NMI	kNN	MLP	ARI	NMI	kNN	MLP	ARI	NMI	kNN	MLP	ARI	NMI	kNN	MLP
Base	$.15_{.04}$	$.23_{.06}$	$.41_{.04}$	$.27_{.06}$	$.40_{.06}$	$.54_{.05}$	$.64_{.03}$	$.41_{.09}$	$.27_{.03}$	$.36_{.03}$	$.58_{.03}$	$.68_{.02}$	$.21_{.03}$	$.35_{.03}$	$.53_{.04}$	$.59_{.04}$	$.63_{.03}$	$.76_{.03}$	$.86_{.04}$	$.91_{.02}$
SimCLR	$.53_{.06}$	$.65_{.05}$	$.79_{.04}$	$.90_{.03}$	$.68_{.05}$	$.77_{.03}$	$.82_{.03}$	$.84_{.03}$	$.40_{.03}$	$.48_{.03}$	$.69_{.02}$	$.77_{.02}$	$.28_{.03}$	$.43_{.04}$	$.58_{.03}$	$.71_{.03}$	$.64_{.07}$	$.76_{.05}$	$.89_{.03}$	$.94_{.01}$
Chuang et al. (2020)	$.44_{.04}$	$.54_{.04}$	$.73_{.05}$	$.88_{.02}$	$.40_{.05}$	$.53_{.05}$	$.63_{.03}$	$.77_{.03}$	$.48_{.05}$	$.54_{.04}$	$.70_{.03}$	$.80_{.03}$	$.29_{.02}$	$.45_{.02}$	$.69_{.03}$	$.72_{.02}$	$.54_{.03}$	$.70_{.02}$	$.85_{.04}$	$.96_{.01}$
Robinson et al. (2020)	$.12_{.03}$	$.18_{.03}$	$.54_{.04}$	$.77_{.03}$	$.11_{.02}$	$.18_{.02}$	$.43_{.02}$	$.61_{.04}$	$.14_{.02}$	$.21_{.02}$	$.47_{.02}$	$.65_{.02}$	$.13_{.02}$	$.26_{.02}$	$.47_{.02}$	$.59_{.03}$	$.18_{.03}$	$.34_{.03}$	$.56_{.03}$	$.72_{.03}$
Xu et al. (2024)-*Attr*	$.68_{.04}$	$.75_{.04}$	$.85_{.04}$	$.92_{.02}$	$.59_{.04}$	$.72_{.04}$	$.79_{.03}$	$.94_{.02}$	$.43_{.05}$	$.50_{.04}$	$.67_{.03}$	$.77_{.03}$	$.28_{.04}$	$.44_{.03}$	$.66_{.04}$	$.71_{.04}$	$.66_{.05}$	$.78_{.03}$	$.84_{.03}$	$.95_{.01}$
Xu et al. (2024)-*EI*	$.69_{.04}$	$.77_{.03}$	$.84_{.03}$	$.92_{.02}$	$.64_{.06}$	$.77_{.04}$	$.83_{.04}$	$.94_{.02}$	$.41_{.04}$	$.50_{.03}$	$.68_{.03}$	$.80_{.02}$	$.28_{.03}$	$.44_{.03}$	$.63_{.03}$	$.76_{.03}$	$.65_{.07}$	$.75_{.05}$	$.86_{.03}$	$.96_{.01}$
Huynh et al. (2022)-*Attr*	$.63_{.05}$	$.72_{.04}$	$.87_{.02}$	$\mathbf{.93_{.01}}$	$.66_{.02}$	$.79_{.01}$	$.88_{.02}$	$.97_{.01}$	$.46_{.04}$	$.54_{.03}$	$.71_{.02}$	$.80_{.03}$	$.33_{.02}$	$.48_{.02}$	$.64_{.02}$	$.71_{.03}$	$.59_{.07}$	$.72_{.04}$	$.86_{.03}$	$\mathbf{.97_{.01}}$
Huynh et al. (2022)-*EI*	$.57_{.07}$	$.67_{.06}$	$.82_{.04}$	$.92_{.02}$	$.67_{.07}$	$.77_{.04}$	$.87_{.03}$	$.93_{.02}$	$.40_{.03}$	$.46_{.03}$	$.67_{.02}$	$.77_{.03}$	$.29_{.03}$	$.44_{.04}$	$.60_{.03}$	$.69_{.03}$	$.62_{.06}$	$.75_{.05}$	$.86_{.02}$	$.96_{.01}$
AutoEncoder	$.33_{.04}$	$.44_{.03}$	$.59_{.03}$	$.58_{.03}$	$.54_{.03}$	$.69_{.03}$	$.83_{.01}$	$.89_{.01}$	$.13_{.02}$	$.19_{.03}$	$.42_{.03}$	$.45_{.07}$	$.36_{.04}$	$.53_{.04}$	$.66_{.05}$	$.67_{.05}$	$.63_{.06}$	$.75_{.04}$	$.88_{.04}$	$.84_{.04}$
SoftCLT-I Lee et al. (2024)	$.56_{.05}$	$.67_{.05}$	$.80_{.05}$	$.88_{.02}$	$.60_{.05}$	$.73_{.03}$	$.84_{.02}$	$.90_{.03}$	$.39_{.03}$	$.46_{.02}$	$.64_{.02}$	$.76_{.02}$	$.37_{.04}$	$.49_{.04}$	$.68_{.04}$	$.79_{.03}$	$.62_{.04}$	$.76_{.03}$	$.87_{.03}$	$.94_{.02}$
K-ICL	$\mathbf{.72_{.03}}$	$\mathbf{.80_{.02}}$	$\mathbf{.90_{.02}}$	$.92_{.02}$	$\mathbf{.82_{.04}}$	$\mathbf{.89_{.03}}$	$\mathbf{.94_{.02}}$	$\mathbf{.98_{.01}}$	$\mathbf{.58_{.05}}$	$\mathbf{.65_{.04}}$	$\mathbf{.78_{.02}}$	$\mathbf{.85_{.02}}$	$\mathbf{.47_{.04}}$	$\mathbf{.63_{.03}}$	$\mathbf{.78_{.02}}$	$\mathbf{.84_{.02}}$	$\mathbf{.78_{.05}}$	$\mathbf{.86_{.03}}$	$\mathbf{.90_{.03}}$	$\mathbf{.97_{.01}}$
Oracle	$.94_{.03}$	$.96_{.02}$	$.99_{.01}$	$.99_{.00}$	$.99_{.00}$	$.99_{.00}$	$1.00_{.00}$	$1.00_{.00}$	$.69_{.02}$	$.75_{.02}$	$.88_{.02}$	$.91_{.01}$	$.64_{.04}$	$.76_{.02}$	$.92_{.01}$	$.94_{.01}$	$.93_{.03}$	$.96_{.02}$	$.98_{.01}$	$.99_{.00}$

Table 3: Clustering and classification metrics in the unbalanced setting (minority classes at $1/10$ of the dominant one). Values are mean with SEM as subscript, over 10 runs. Best per column in **bold**, second-best underlined; the supervised Oracle is excluded from the ranking.

Method	CEBS				PAMAP				PenDigits				Jap.Vowels				Char.Traj.			
	ARI	NMI	kNN	MLP	ARI	NMI	kNN	MLP	ARI	NMI	kNN	MLP	ARI	NMI	kNN	MLP	ARI	NMI	kNN	MLP
Base	$.11_{.02}$	$.25_{.04}$	$.28_{.03}$	$.14_{.03}$	$.39_{.03}$	$.58_{.03}$	$.55_{.03}$	$.24_{.07}$	$.26_{.02}$	$.41_{.02}$	$.50_{.02}$	$.59_{.02}$	$.16_{.02}$	$.35_{.02}$	$.41_{.02}$	$.46_{.03}$	$.54_{.04}$	$.74_{.03}$	$.74_{.02}$	$.80_{.02}$
SimCLR	$.50_{.03}$	$.65_{.02}$	$.67_{.02}$	$.73_{.02}$	$.55_{.04}$	$.72_{.03}$	$.69_{.04}$	$.78_{.04}$	$.36_{.02}$	$.52_{.02}$	$.58_{.01}$	$.68_{.01}$	$.23_{.02}$	$.43_{.02}$	$.49_{.02}$	$.58_{.02}$	$.58_{.03}$	$.75_{.02}$	$.79_{.02}$	$.89_{.01}$
Chuang et al. (2020)	$.37_{.04}$	$.54_{.03}$	$.61_{.03}$	$.70_{.03}$	$.32_{.03}$	$.50_{.02}$	$.55_{.03}$	$.68_{.04}$	$.39_{.02}$	$.55_{.02}$	$.61_{.02}$	$.69_{.02}$	$.22_{.02}$	$.44_{.02}$	$.50_{.02}$	$.57_{.02}$	$.52_{.03}$	$.72_{.02}$	$.78_{.02}$	$.89_{.01}$
Robinson et al. (2020)	$.26_{.02}$	$.42_{.02}$	$.54_{.02}$	$.66_{.02}$	$.19_{.02}$	$.32_{.02}$	$.39_{.01}$	$.51_{.03}$	$.25_{.01}$	$.40_{.01}$	$.52_{.01}$	$.65_{.01}$	$.22_{.01}$	$.42_{.02}$	$.49_{.02}$	$.56_{.02}$	$.32_{.02}$	$.57_{.02}$	$.68_{.02}$	$.78_{.02}$
Xu et al. (2024)-*Attr*	$.47_{.04}$	$.64_{.02}$	$.63_{.03}$	$.67_{.03}$	$.54_{.02}$	$.70_{.02}$	$.71_{.02}$	$.78_{.02}$	$.38_{.02}$	$.54_{.02}$	$.59_{.02}$	$.70_{.02}$	$.23_{.02}$	$.43_{.02}$	$.51_{.02}$	$.55_{.03}$	$.59_{.03}$	$.78_{.02}$	$\mathbf{.85_{.02}}$	$.91_{.02}$
Xu et al. (2024)-*EI*	$.48_{.02}$	$.65_{.02}$	$.66_{.03}$	$.72_{.03}$	$.54_{.05}$	$.71_{.04}$	$.70_{.04}$	$.76_{.04}$	$.40_{.02}$	$.55_{.02}$	$.62_{.02}$	$.72_{.02}$	$.23_{.02}$	$.46_{.02}$	$.50_{.02}$	$.58_{.02}$	$.62_{.04}$	$.78_{.02}$	$.83_{.02}$	$.91_{.01}$
Huynh et al. (2022)-*Attr*	$.54_{.03}$	$.69_{.02}$	$\mathbf{.71_{.03}}$	$\mathbf{.77_{.03}}$	$.55_{.03}$	$.73_{.02}$	$.70_{.02}$	$.79_{.03}$	$.39_{.02}$	$.55_{.02}$	$.60_{.01}$	$.72_{.01}$	$.21_{.01}$	$.43_{.02}$	$.45_{.01}$	$.54_{.03}$	$.66_{.04}$	$.81_{.02}$	$.84_{.02}$	$.92_{.01}$
Huynh et al. (2022)-*EI*	$.53_{.03}$	$.69_{.02}$	$.68_{.02}$	$.75_{.02}$	$.53_{.03}$	$.70_{.03}$	$.69_{.02}$	$.79_{.03}$	$.41_{.01}$	$.55_{.01}$	$.60_{.02}$	$.70_{.01}$	$.23_{.03}$	$.44_{.02}$	$.49_{.02}$	$.56_{.02}$	$.60_{.02}$	$.78_{.01}$	$.82_{.02}$	$.90_{.01}$
AutoEncoder	$.31_{.02}$	$.49_{.02}$	$.50_{.02}$	$.50_{.02}$	$.47_{.07}$	$.66_{.05}$	$.66_{.04}$	$.78_{.04}$	$.10_{.01}$	$.21_{.01}$	$.30_{.02}$	$.35_{.07}$	$.17_{.01}$	$.39_{.01}$	$.36_{.03}$	$.42_{.02}$	$.58_{.03}$	$.77_{.02}$	$.82_{.01}$	$.73_{.07}$
SoftCLT-I Lee et al. (2024)	$.50_{.05}$	$.65_{.04}$	$.66_{.04}$	$.73_{.03}$	$.57_{.05}$	$.72_{.03}$	$.71_{.04}$	$.83_{.04}$	$.39_{.01}$	$.54_{.01}$	$.61_{.02}$	$.72_{.01}$	$.27_{.03}$	$.47_{.02}$	$.52_{.02}$	$\mathbf{.61_{.02}}$	$.56_{.03}$	$.75_{.02}$	$.80_{.02}$	$.88_{.02}$
K-ICL	$\mathbf{.54_{.03}}$	$\mathbf{.70_{.02}}$	$\mathbf{.71_{.03}}$	$\mathbf{.77_{.03}}$	$\mathbf{.70_{.04}}$	$\mathbf{.84_{.02}}$	$\mathbf{.83_{.03}}$	$\mathbf{.95_{.02}}$	$\mathbf{.46_{.02}}$	$\mathbf{.61_{.01}}$	$\mathbf{.65_{.02}}$	$\mathbf{.74_{.01}}$	$\mathbf{.30_{.03}}$	$\mathbf{.49_{.02}}$	$\mathbf{.56_{.02}}$	$.60_{.03}$	$\mathbf{.71_{.02}}$	$\mathbf{.84_{.01}}$	$\mathbf{.85_{.02}}$	$\mathbf{.94_{.01}}$
Oracle	$.70_{.03}$	$.81_{.02}$	$.87_{.01}$	$.91_{.01}$	$.88_{.03}$	$.93_{.02}$	$.96_{.01}$	$.97_{.01}$	$.52_{.03}$	$.66_{.02}$	$.72_{.02}$	$.81_{.02}$	$.33_{.03}$	$.55_{.02}$	$.60_{.02}$	$.68_{.02}$	$.85_{.02}$	$.92_{.01}$	$.96_{.01}$	$.97_{.01}$

Contrastive Learning in Class-Unbalanced Settings. We next consider a *class-unbalanced* setting with ten classes (eight for PAMAP and nine for JapaneseVowels, i.e., all their available classes), one holding ten times as many samples as each of the others (10:1), accounting for more than half of the training set. When the anchor belongs to the dominant class, most of its in-batch negatives share the same unknown label, significantly increasing the proportion of ill-defined negatives: Tab. 1 shows SimCLR's negative precision dropping to 62–74% in the unbalanced case, which K-ICL restores to above 90% while also significantly improving positive recall. Tab. 3 shows that K-ICL achieves the best clustering and k-NN performance on all five datasets, with the largest gains on PAMAP (ARI from .55 to .70) and on CharacterTrajectories (ARI from .58 to .71), resulting in the best or tied-best on MLP in most cases. Here, several competitors degrade under imbalance: the hard-negative weighting of (Robinson et al., 2020) is particularly penalized, as emphasizing negatives close to the anchor up-weights same-class samples (FN) precisely where they are most frequent, dropping below even SimCLR in term of performance. SoftCLT-I softens rather than removes ambiguous pairs and remains competitive only on some datasets (e.g. on JapaneseVowels). The selective mining of K-ICL concentrates the signal on confidently well-defined pairs and yields the strongest overall results.

4.3 Ablation Studies

We ablate the three components of K-ICL's mining, namely the kernel, the sampling temperatures and the sampling rule, in the unbalanced (10:1) setting, where the FN problem is most evident.

Choice of Kernel. K-ICL is not tied to a single kernel, but benefits from any similarity capturing temporal alignment. Tab. 4 compares Euclidean, DTW and GAK kernels: the two alignment-aware kernels (DTW, GAK) perform comparably and each is best on some datasets, while Euclidean, ignoring misalignment, is the poorest in terms of pair mining (Appendix A.3) and yields the worst representations. Pair mining in K-ICL is thus kernel-agnostic, even though its effectiveness still depends on the adopted kernel similarity. We adopt GAK by default, as it captures alignment while staying well-calibrated across datasets.

Sampling Rule. Tab. 5 replaces probabilistic sampling with two deterministic rules on the same GAK similarities: *Thresh* (positives if similarity > 0.5) and *RankHalf* (top half of the batch ranked by kernel similarity as positives, bottom half as negatives). *Thresh* is inconsistent, as similarities vary in scale across datasets and no single threshold fits them all a priori. *RankHalf* is competitive here, as it recovers many same-class positives. This is because, with one dominant class filling roughly half

Table 4: K-ICL with the three kernels, Euclidean, DTW and GAK, in the balanced and unbalanced settings. Per-dataset results are mean with SEM over 10 runs; the *Average* row is the mean across the five datasets (no SEM). Best kernel per metric, on the average, in **bold**.

	Dataset	Euclidean				DTW				GAK			
		ARI	NMI	kNN	MLP	ARI	NMI	kNN	MLP	ARI	NMI	kNN	MLP
Balanced (5 classes)	CEBS	$.54_{.07}$	$.62_{.06}$	$.73_{.05}$	$.81_{.03}$	$.71_{.04}$	$.79_{.03}$	$.89_{.02}$	$.93_{.01}$	$.72_{.03}$	$.80_{.02}$	$.90_{.02}$	$.92_{.02}$
	PAMAP	$.78_{.04}$	$.87_{.03}$	$.92_{.02}$	$.96_{.02}$	$.79_{.03}$	$.88_{.01}$	$.94_{.01}$	$.98_{.01}$	$.82_{.04}$	$.89_{.03}$	$.94_{.02}$	$.98_{.01}$
	PenDigits	$.49_{.03}$	$.59_{.02}$	$.76_{.02}$	$.83_{.02}$	$.49_{.05}$	$.58_{.03}$	$.77_{.02}$	$.82_{.03}$	$.58_{.05}$	$.65_{.04}$	$.78_{.02}$	$.85_{.02}$
	Jap.Vowels	$.44_{.04}$	$.59_{.03}$	$.76_{.02}$	$.82_{.02}$	$.47_{.03}$	$.60_{.03}$	$.80_{.02}$	$.84_{.02}$	$.47_{.04}$	$.63_{.03}$	$.78_{.02}$	$.84_{.02}$
	Char.Traj.	$.73_{.05}$	$.82_{.03}$	$.90_{.01}$	$.94_{.01}$	$.76_{.04}$	$.84_{.03}$	$.92_{.02}$	$.97_{.01}$	$.78_{.05}$	$.86_{.03}$	$.90_{.03}$	$.97_{.01}$
	Average	.60	.70	.81	.87	.64	.74	.86	**.91**	**.67**	**.76**	**.86**	**.91**
Unbalanced (10:1)	CEBS	$.39_{.02}$	$.57_{.02}$	$.56_{.02}$	$.62_{.03}$	$.57_{.03}$	$.71_{.02}$	$.72_{.02}$	$.79_{.02}$	$.54_{.03}$	$.70_{.02}$	$.71_{.03}$	$.77_{.03}$
	PAMAP	$.67_{.04}$	$.81_{.02}$	$.81_{.02}$	$.88_{.03}$	$.73_{.04}$	$.84_{.03}$	$.88_{.04}$	$.90_{.03}$	$.70_{.04}$	$.84_{.02}$	$.83_{.03}$	$.95_{.02}$
	PenDigits	$.46_{.01}$	$.62_{.01}$	$.67_{.01}$	$.75_{.01}$	$.42_{.01}$	$.58_{.01}$	$.63_{.01}$	$.74_{.01}$	$.46_{.02}$	$.61_{.01}$	$.65_{.02}$	$.74_{.01}$
	Jap.Vowels	$.27_{.02}$	$.49_{.02}$	$.55_{.02}$	$.62_{.02}$	$.28_{.02}$	$.49_{.02}$	$.57_{.02}$	$.64_{.02}$	$.30_{.03}$	$.49_{.02}$	$.56_{.02}$	$.60_{.03}$
	Char.Traj.	$.70_{.02}$	$.83_{.01}$	$.83_{.02}$	$.90_{.02}$	$.71_{.03}$	$.84_{.02}$	$.85_{.03}$	$.93_{.01}$	$.71_{.02}$	$.84_{.01}$	$.85_{.02}$	$.94_{.01}$
	Average	.50	.66	.68	.75	**.54**	.69	**.73**	**.80**	**.54**	**.70**	.72	**.80**

Table 5: Ablation of the mining rule in the unbalanced (10:1) setting. Unlike K-ICL probabilistic sampling (τ^+=τ^-=0.1), RankHalf takes the top half of the batch by GAK similarity as positives and the bottom half as negatives, and Thresh takes GAK>0.5 as positives and GAK<0.5 as negatives. Mean with SEM as subscript over 10 runs.

Dataset	K-ICL				RankHalf				Thresh			
	ARI	NMI	kNN	MLP	ARI	NMI	kNN	MLP	ARI	NMI	kNN	MLP
CEBS	$.54_{.03}$	$.70_{.02}$	$.71_{.03}$	$.77_{.03}$	$.51_{.03}$	$.68_{.02}$	$.68_{.03}$	$.74_{.03}$	$.54_{.02}$	$.70_{.02}$	$.72_{.02}$	$.78_{.02}$
PAMAP	$.70_{.04}$	$.84_{.02}$	$.83_{.03}$	$.95_{.02}$	$.75_{.03}$	$.86_{.02}$	$.87_{.02}$	$.93_{.02}$	$.67_{.03}$	$.82_{.02}$	$.81_{.02}$	$.90_{.03}$
PenDigits	$.46_{.02}$	$.61_{.01}$	$.65_{.02}$	$.74_{.01}$	$.46_{.02}$	$.61_{.02}$	$.63_{.02}$	$.73_{.02}$	$.43_{.01}$	$.60_{.01}$	$.63_{.01}$	$.74_{.02}$
JapaneseVowels	$.30_{.03}$	$.49_{.02}$	$.56_{.02}$	$.60_{.03}$	$.33_{.03}$	$.53_{.02}$	$.60_{.02}$	$.64_{.02}$	$.28_{.02}$	$.50_{.02}$	$.60_{.02}$	$.60_{.03}$
CharacterTrajectories	$.71_{.02}$	$.84_{.01}$	$.85_{.02}$	$.94_{.01}$	$.69_{.03}$	$.83_{.02}$	$.85_{.02}$	$.90_{.01}$	$.70_{.03}$	$.84_{.02}$	$.87_{.02}$	$.93_{.01}$

of each batch, an even positive/negative split happens to suit the majority-class anchors: this match is tied to the specific imbalance ratio and would not hold under a different class balance. K-ICL instead achieves the highest negative precision across datasets (Appendix A.4.1) and adapts its split to the batch composition through the similarity distribution, making probabilistic sampling the robust choice without assuming a fixed threshold or a fixed positive/negative proportion.

Sampling Temperatures. The temperatures τ^+, τ^- (set to 0.1 by default) control the precision–recall trade-off of the mined pairs: a very low value concentrates sampling on the nearest neighbors, maximizing precision but collapsing recall, while a high value flattens the distribution, restoring recall at the cost of precision and possibly introducing more ill-defined pairs. The intermediate default ($\tau^+ = \tau^- = 0.1$) balances the two and yields the best or near-best downstream performance across datasets. We use a single shared temperature for simplicity, but τ^+ and τ^- may be set independently to tune the trade-off per task. Detailed results and discussion are provided in Appendix A.4.2.

5 Conclusions

In this work, we introduced K-ICL, a *kernel-guided pair-mining strategy* for *instance-wise* CL on time-series, particularly beneficial under class imbalance. K-ICL probabilistically selects well-defined positive and negative pairs within each batch by leveraging a kernel similarity computed directly in the input space, using only weak augmentations. Its effectiveness comes from improving the quality of pairs on both sides: it reduces false negatives, while recovering informative cross-instance positives. Our formal analysis links surrogate mining error to the separation between true positives and true negatives in kernel-similarity space, and our experiments show that K-ICL outperforms the considered baselines and competing instance-wise mining strategies on both user identification and classification tasks. Since K-ICL acts at the level of pair selection, it could be in principle plugged into advanced CL frameworks to improve their instance-wise component. Thus, a natural next step consists in combining K-ICL with intra-segment temporal contrastive objectives and to study how the two combined improve the effectiveness of the learned represenation. Future work also aim at incorporating soft latent-space guidance to refine embeddings, while carefully mitigating the risk of detrimental circular dependencies.

AI USE STATEMENT

We used generative AI tools in two limited roles: to polish the writing of text and to assist with code development and refinement. We did not use generative AI for ideation, for producing the scientific content or claims, or for verifying the validity of our findings. All AI-assisted work was reviewed by the authors: LLM-generated code was read, tested, and verified for correctness before use, and AI-assisted text edits were restricted to stylistic improvements of content written by authors, which we checked to ensure no external material was introduced. We take full responsibility for the final content of this work, including all text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work proposes a kernel-based pair-mining strategy for instance-wise contrastive learning on time series and does not raise notable ethical concerns. All datasets used are publicly available. The biometric identification data are standard public benchmarks used here only for representation-learning evaluation. Our method does not target individuals, generate sensitive content, or introduce risks beyond those inherent to self-supervised representation learning in general.

REPRODUCIBILITY STATEMENT

We provide the information needed to reproduce our results in the main text and appendix: the full method (Sec. 3), the datasets and their preprocessing (Appendix A), the complete experimental protocol and hyperparameters (Appendix A), and the assumptions and full proofs of our theoretical results (Appendix B). All datasets used are publicly available. Our experiments are seeded and can be independently re-run from the reported configurations. We further provide the implementation of K-ICL as supplementary material, and will release the complete codebase publicly upon acceptance.

REFERENCES

Mustafa Gokce Baydogan and George Runger. Time series representation and similarity based on local autopatterns. *Data Mining and Knowledge Discovery*, 30(2):476–509, 2016.

Donald J. Berndt and James Clifford. Using dynamic time warping to find patterns in time series. In *Knowledge Discovery in Databases: Papers from the 1994 AAAI Workshop*, pp. 359–370. AAAI Press, 1994.

Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr. Fully-convolutional siamese networks for object tracking. In *European conference on computer vision*, pp. 850–865. Springer, 2016.

Filippo Maria Bianchi, Lorenzo Livi, Karl Øyvind Mikalsen, Michael Kampffmeyer, and Robert Jenssen. Learning representations of multivariate time series with missing data. *Pattern Recognition*, 96:106973, 2019.

Filippo Maria Bianchi, Simone Scardapane, Sigurd Løkse, and Robert Jenssen. Reservoir computing approaches for representation and classification of multivariate time series. *IEEE transactions on neural networks and learning systems*, 32(5):2169–2179, 2020.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PmLR, 2020.

Ching-Yao Chuang, Joshua Robinson, Yen-Chen Lin, Antonio Torralba, and Stefanie Jegelka. Debiased contrastive learning. *Advances in neural information processing systems*, 33:8765–8775, 2020.

Marco Cuturi. Fast global alignment kernels. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*, pp. 929–936. Omnipress, 2011. URL https://icml.cc/2011/papers/489_icmlpaper.pdf.

Hoang Anh Dau, Eamonn Keogh, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, Yanping, Bing Hu, Nurjahan Begum, Anthony Bagnall, Abdullah Mueen, Gustavo Batista, and Hexagon-ML. The ucr time series classification archive, October 2018. https://www.cs.ucr.edu/~eamonn/time_series_data_2018/.

Giovanni De Felice, John Goulermas, and Vladimir Gusev. Time series kernels based on nonlinear vector autoregressive delay embeddings. *Advances in Neural Information Processing Systems*, 36: 37230–37251, 2023.

Angus Dempster, Daniel F Schmidt, and Geoffrey I Webb. Hydra: Competing convolutional kernels for fast and accurate time series classification. *Data Mining and Knowledge Discovery*, 37: 1779–1805, 2023.

Debidatta Dwibedi, Yusuf Aytar, Jonathan Tompson, Pierre Sermanet, and Andrew Zisserman. With a little help from my friends: Nearest-neighbor contrastive learning of visual representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9568–9577, 2021. doi: 10.1109/ICCV48922.2021.00945. URL https://doi.org/10.1109/ICCV48922.2021.00945.

Emadeldeen Eldele, Mohamed Ragab, Zhenghua Chen, Min Wu, Chee Keong Kwoh, Xiaoli Li, and Cuntai Guan. Time-series representation learning via temporal and contextual contrasting. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 2352–2359. ijcai.org, 2021. doi: 10.24963/IJCAI.2021/324. URL https://doi.org/10.24963/ijcai.2021/324.

Navid Mohammadi Foumani, Lynn Miller, Chang Wei Tan, Geoffrey I. Webb, Germain Forestier, and Mahsa Salehi. Deep learning for time series classification and extrinsic regression: A current survey. *ACM Computing Surveys*, 56(9):217:1–217:45, 2024. doi: 10.1145/3649448. URL https://doi.org/10.1145/3649448.

Jean-Yves Franceschi, Aymeric Dieuleveut, and Martin Jaggi. Unsupervised scalable representation learning for multivariate time series. *Advances in neural information processing systems*, 32, 2019.

Miguel A García-González, Ariadna Argelagós-Palau, Mireya Fernández-Chimeno, and Juan Ramos-Castro. A comparison of heartbeat detectors for the seismocardiogram. In *Computing in Cardiology 2013*, pp. 461–464. IEEE, 2013.

Ary L Goldberger, 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.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9729–9738, 2020.

Tri Huynh, Simon Kornblith, Matthew R Walter, Michael Maire, and Maryam Khademi. Boosting contrastive self-supervised learning with false negative cancellation. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pp. 2785–2795, 2022.

Xiyuan Jin, Jing Wang, Xiaoyu Ou, Lei Liu, and Youfang Lin. Time-series contrastive learning against false negatives and class imbalance. *IEEE Transactions on Neural Networks and Learning Systems*, 36(9):16946–16959, 2025. doi: 10.1109/TNNLS.2025.3568387. URL https://doi.org/10.1109/TNNLS.2025.3568387.

Marco-Blanco Jorge and Cuevas Rubén. Time series clustering with random convolutional kernels. *Data Mining and Knowledge Discovery*, 38(4):1862–1888, 2024.

Denizhan Kara, Tomoyoshi Kimura, Jinyang Li, Bowen He, Yizhuo Chen, Yigong Hu, Hongjue Zhao, Shengzhong Liu, and Tarek Abdelzaher. Adats: Learning adaptive time series representations via dynamic soft contrasts. *Advances in Neural Information Processing Systems*, 38:74779–74805, 2026.

Seunghan Lee, Taeyoung Park, and Kibok Lee. Soft contrastive learning for time series. In *The Twelfth International Conference on Learning Representations*, 2024.

Jessica Lin, Eamonn Keogh, Stefano Lonardi, and Bill Chiu. A symbolic representation of time series, with implications for streaming algorithms. In *Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery*, pp. 2–11, 2003.

Dongsheng Luo, Wei Cheng, Yingheng Wang, Dongkuan Xu, Jingchao Ni, Wenchao Yu, Xuchao Zhang, Yanchi Liu, Yuncong Chen, Haifeng Chen, et al. Time series contrastive learning with information-aware augmentations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 4534–4542, 2023.

Karl Øyvind Mikalsen, Filippo Maria Bianchi, Cristina Soguero-Ruiz, and Robert Jenssen. Time series cluster kernel for learning similarities between multivariate time series with missing data. *Pattern Recognition*, 76:569–581, 2018.

Johannes Pöppelbaum, Gavneet Singh Chadha, and Andreas Schwung. Contrastive learning based self-supervised time-series analysis. *Applied Soft Computing*, 117:108397, 2022.

Attila Reiss. PAMAP2 Physical Activity Monitoring. UCI Machine Learning Repository, 2012. DOI: https://doi.org/10.24432/C5NW2H.

Cédric Richard, José Carlos M Bermudez, and Paul Honeine. Online prediction of time series data with kernels. *IEEE Transactions on Signal Processing*, 57(3):1058–1067, 2008.

Joshua Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. Contrastive learning with hard negative samples. *arXiv preprint arXiv:2010.04592*, 2020.

Patrick Schäfer and Mikael Högqvist. Sfa: a symbolic fourier approximation and index for similarity search in high dimensional datasets. In *Proceedings of the 15th international conference on extending database technology*, pp. 516–527, 2012.

Romain Tavenard, Johann Faouzi, Gilles Vandewiele, Felix Divo, Guillaume Androz, Chester Holtz, Marie Payne, Roman Yurchak, Marc Rußwurm, Kushal Kolar, et al. Tslearn, a machine learning toolkit for time series data. *Journal of machine learning research*, 21(118):1–6, 2020.

Sana Tonekaboni, Danny Eytan, and Anna Goldenberg. Unsupervised representation learning for time series with temporal neighborhood coding. In *International Conference on Learning Representations (ICLR)*, 2021. URL https://openreview.net/forum?id=8qDwejCuCN.

Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748, 2018. URL http://arxiv.org/abs/1807.03748.

Qingsong Wen, Liang Sun, Fan Yang, Xiaomin Song, Jingkun Gao, Xue Wang, and Huan Xu. Time series data augmentation for deep learning: A survey. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 4653–4660. ijcai.org, 2021. doi: 10.24963/IJCAI.2021/631. URL https://doi.org/10.24963/ijcai.2021/631.

Kristoffer Wickstrøm, Michael Kampffmeyer, Karl Øyvind Mikalsen, and Robert Jenssen. Mixing up contrastive learning: Self-supervised representation learning for time series. *Pattern Recognition Letters*, 155:54–61, 2022.

Gerald Woo, Chenghao Liu, Doyen Sahoo, Akshat Kumar, and Steven Hoi. CoST: Contrastive learning of disentangled seasonal-trend representations for time series forecasting. In *International Conference on Learning Representations*, 2022.

Yu Wu, Ting Dang, Dimitris Spathis, Hong Jia, and Cecilia Mascolo. Statiocl: contrastive learning for time series via non-stationary and temporal contrast. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pp. 2575–2584, 2024.

Lingling Xu, Haoran Xie, Fu Lee Wang, Xiaohui Tao, Weiming Wang, and Qing Li. Contrastive sentence representation learning with adaptive false negative cancellation. *Information Fusion*, 102:102065, 2024.

Xinyu Yang, Zhenguo Zhang, and Rongyi Cui. Timeclr: A self-supervised contrastive learning framework for univariate time series representation. *Knowledge-Based Systems*, 245:108606, 2022.

Zhihan Yue, Yujing Wang, Juanyong Duan, Tianmeng Yang, Congrui Huang, Yunhai Tong, and Bixiong Xu. Ts2vec: Towards universal representation of time series. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 8980–8987. AAAI Press, 2022. doi: 10.1609/AAAI.V36I8.20881. URL https://doi.org/10.1609/aaai.v36i8.20881.

Xiang Zhang, Ziyuan Zhao, Theodoros Tsiligkaridis, and Marinka Zitnik. Self-supervised contrastive pre-training for time series via time-frequency consistency. *Advances in neural information processing systems*, 35:3988–4003, 2022.

A Additional Experimental Details

A.1 Tasks, datasets, and preprocessing

User identification. We established a user identification benchmark based on classifiers. Each classifier comprises an encoder $\mathcal{E}$ trained via CL on unlabeled biometric time series and a classification head (either an MLP or a k-NN classifier with $k = 5$) fine-tuned on a small training set of 5 labeled samples per user. This is a *closed-set* identification task: the same users appear in training and testing, but the evaluated windows are disjoint from the training ones. The trained classifiers are evaluated on a separate test set.

We test ECG-based identification on the publicly available CEBS dataset (García-González et al., 2013) from Physionet (Goldberger et al., 2000), using the Basal-state recordings (the five minutes preceding music listening), excluding two users whose traces contain anomalous recordings. Each continuous ECG series is segmented into fixed-length *non-overlapping* windows, each spanning at least two complete cardiac cycles, which preserves biometric information while giving a consistent input representation. We test accelerometer-based identification on the PAMAP dataset (Reiss, 2012), recorded from multiple users performing different activities via inertial sensors, restricting to the walking activity: this leaves 8 users with sufficient walking recordings.

Windows are split randomly into non-overlapping training and test sets, with all users present in both (closed-set identification). The test sets are balanced while the training sets are not, which justifies using *accuracy* as our primary metric. This benchmark lets us evaluate K-ICL across sensing modalities and levels of subject variability.

Classification. We also established a multi-class time-series classification benchmark of three datasets from the UCR Archive (Dau et al., 2018): PenDigits, JapaneseVowels, and CharacterTrajectories, spanning different sensing modalities, temporal dynamics, and class cardinalities. We adopt the same training and testing procedure and model architectures as for user identification.

Experimental settings. For all datasets and both tasks, we consider two settings. In the *balanced* setting we use 5 randomly selected classes per run, with equal representation. In the *unbalanced* setting we use 10 classes, or all available classes when fewer than 10 (8 for PAMAP, 9 for JapaneseVowels), with one class overrepresented at ten times the samples of each of the others. During training, each minority class retains at least 5 samples: when a class is too small for a clean 10:1 ratio, this floor makes the realized imbalance milder, which is why SimCLR's negative precision does not always match the value predicted by an idealized 10:1 construction (e.g., staying above 70% on JapaneseVowels). Table 6 summarizes the class counts per dataset and setting.

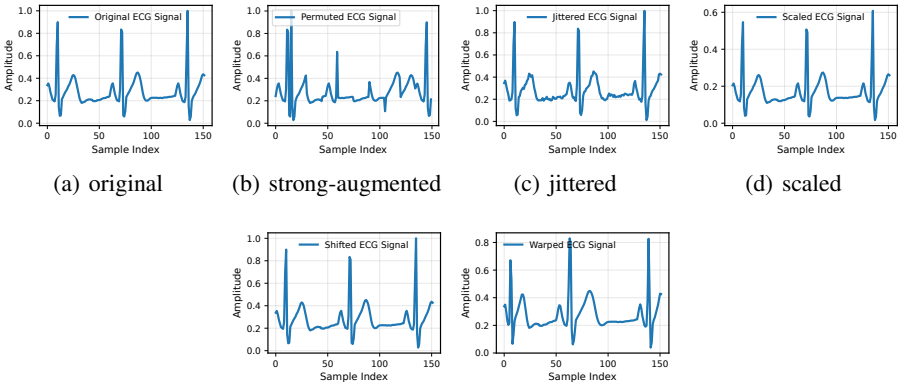
Table 6: Class counts used per dataset in the balanced and unbalanced settings. In the balanced setting each class is equally represented. In the unbalanced (10:1) setting one dominant class holds ten times the samples of each minority class. When a dataset has fewer than 10 classes, all of them are used in the unbalanced setting.

Dataset	Total classes	Balanced	Unbalanced
CEBS	18	5	10
PAMAP	8	5	8
PenDigits	10	5	10
JapaneseVowels	9	5	9
CharacterTrajectories	20	5	10

A.2 Augmentations

As motivated in the main text, time-series augmentations easily violate label preservation, so we restrict K-ICL and all baselines to the weak, semantics-preserving perturbations of (Wen et al., 2021). We use four transformations: *shifting*, a temporal displacement of at most five samples; *scaling*, a global rescaling of the signal amplitude by a factor drawn uniformly from $[0.8, 1.2]$; *jittering*, additive zero-mean Gaussian noise with $\sigma = 0.01$ on the normalised signal, modelling sensor variability; and *time warping*, a smooth local stretching or compression of the time axis obtained by interpolating

four control points drawn from $\mathcal{N}(1, 0.2)$ and renormalising the resulting cumulative map, modelling small speed variations. For each anchor one of the four transformations is drawn uniformly at random and applied to produce the augmented view, while the anchor itself is kept unmodified. These perturbations are not universally valid across all time-series domains, but we use them because they are weak and task-compatible for the considered datasets. When a domain admits only a subset (or different augmentations), K-ICL can be instantiated with that subset without changing the mining mechanism. Fig. 3 illustrates their effect on an ECG instance (c–f), alongside the original signal (a) and, for contrast, a strong segment-permutation augmentation (b) that disrupts the temporal semantics and that we therefore avoid.



(a) original (b) strong-augmented (c) jittered (d) scaled

(e) shifted (f) time warped

Figure 3: Original time series (a) vs. strong augmented (segment-wise permuted) time series (b) vs. weak augmented time series, namely jittered (c), scaled (d), shifted (e) and time warped (f).

A.3 Time Series Kernels

Time-series kernels provide an explicit similarity in the input space (Bianchi et al., 2019). Formally, a kernel is a positive semi-definite similarity function that compares two time series, which can be interpreted as an inner product in a (possibly high-dimensional) feature space (Richard et al., 2008). In our notation, a kernel is a function $\kappa : \mathbb{R}^{C\times T} \times \mathbb{R}^{C\times T} \to \mathbb{R}^{+}$ that computes a similarity score $s_{ij} = \kappa(\mathbf{S}_i, \mathbf{S}_j)$ between two time series.

Some kernels are built upon (dis)similarity measures that account for meaningful invariances of time series. One of the most popular dissimilarity measures is the Dynamic Time Warping (DTW), an alignment-based distance that is invariant to temporal translations and distortions (Berndt & Clifford, 1994). Building on DTW alignments, the GAK kernel aggregates similarities over all possible alignments while remaining positive semi-definite (Cuturi, 2011). Beyond DTW-based kernels, many alternatives exist, including kernels for multivariate time series with missing data (Mikalsen et al., 2018), pattern- and symbol-based similarities (Lin et al., 2003; Baydogan & Runger, 2016; Schäfer & Högqvist, 2012), kernels derived from dynamical-system embeddings (De Felice et al., 2023), reservoir computing (Bianchi et al., 2020), and efficient convolutional/random-kernel schemes for time-series classification (Dempster et al., 2023; Jorge & Rubén, 2024).

In our work, we leverage kernels to guide the training of a deep learning model, combining the structure-aware, input-space similarity induced by kernel methods with the representation-learning flexibility of deep neural networks. Our approach is in contrast with the mainstream approach in Deep learning where similarity is implicitly assessed in the embedding space. Kernels provide a fixed input-space notion of similarity that can encode structural biases or expert knowledge.

While deep learning models are data hungry but efficient at inference time, kernel methods are sample-efficient but computationally expensive, especially at inference time. This motivates the development of a hybrid approach that retains the flexibility of deep encoders while leveraging kernel similarities as a sample-efficient and reliable supervisory signal over sequences, e.g., to guide pair selection in CL.

Kernel instantiations. All our experiments use GAK as the similarity κ, and we ablate it against two simpler alternatives. All three produce a similarity in $[0, 1]$ that is used identically by our sampling rule, so the ablation isolates the notion of similarity itself. For GAK we use the implementation of *tslearn* (Tavenard et al., 2020), whose bandwidth σ is the only hyperparameter: we estimate it with the median heuristic of Cuturi (2011) on the training set, which requires no labels and no tuning. DTW uses the multivariate alignment distance mapped to a similarity by $\exp(-\gamma\, d_{\mathrm{DTW}})$, with γ fixed by the same median heuristic. The Euclidean variant flattens each series and applies a min–max-inverted pairwise Euclidean distance.

Kernels' mining. Tab. 7 reports the quality of the pairs each kernel mines. The two alignment-aware kernels consistently mine cleaner pairs than the Euclidean one: GAK reaches the highest positive precision (Prec.$^+$) and negative precision (Prec.$^-$) on almost every dataset and setting, with DTW second and Euclidean clearly last. This confirms that capturing temporal alignment yields a similarity that better separates true positives from true negatives, which is precisely what the mining relies on. The gap is most pronounced on the negative side under imbalance, where GAK's Prec.$^-$ stays high while the Euclidean kernel's drops sharply, indicating that alignment-aware similarity is especially effective at avoiding false negatives in the regime K-ICL targets. The higher recall of the weaker kernels on some datasets reflects their less selective mining rather than better pairs, and does not translate into better downstream performance (Tab. 4).

Table 7: Quality of the positive and negative sets mined by K-ICL with the three kernels, Euclidean, DTW and GAK, in the balanced setting with 5 classes and in the unbalanced setting (minority classes at 1/10 of the dominant one). Prec.$^+$/Rec.$^+$ are precision and recall of the positives, Prec.$^-$/Rec.$^-$ of the negatives (%, mean with SEM as subscript over 10 runs). Best kernel per metric in **bold**.

	Dataset	Euclidean				DTW				GAK			
		Prec.$^+$	Rec.$^+$	Prec.$^-$	Rec.$^-$	Prec.$^+$	Rec.$^+$	Prec.$^-$	Rec.$^-$	Prec.$^+$	Rec.$^+$	Prec.$^-$	Rec.$^-$
Balanced (5 classes)	CEBS	$48.8_{1.1}$	$49.0_{1.8}$	$90.7_{1.0}$	$43.0_{0.8}$	$55.0_{1.5}$	$64.2_{1.4}$	$95.3_{0.7}$	$47.0_{0.3}$	$\mathbf{73.1_{2.1}}$	$\mathbf{64.8_{3.4}}$	$\mathbf{98.4_{0.6}}$	$\mathbf{47.6_{1.3}}$
	PAMAP	$56.7_{1.3}$	$\mathbf{73.5_{1.0}}$	$97.5_{0.5}$	$39.5_{0.8}$	$64.2_{1.0}$	$69.9_{0.8}$	$97.3_{0.3}$	$49.7_{0.4}$	$\mathbf{81.8_{1.0}}$	$60.6_{1.8}$	$\mathbf{97.9_{0.5}}$	$\mathbf{53.8_{0.7}}$
	PenDigits	$\mathbf{67.2_{1.3}}$	$61.1_{1.2}$	$94.1_{0.6}$	$46.2_{0.4}$	$61.1_{1.1}$	$57.4_{1.0}$	$92.8_{0.5}$	$\mathbf{53.2_{0.3}}$	$59.6_{1.3}$	$\mathbf{72.6_{1.4}}$	$\mathbf{94.9_{0.6}}$	$42.5_{0.4}$
	JapaneseVowels	$56.5_{1.0}$	$\mathbf{57.2_{1.1}}$	$94.2_{0.3}$	$44.7_{0.7}$	$55.1_{0.7}$	$50.8_{0.7}$	$92.0_{0.3}$	$\mathbf{52.4_{0.4}}$	$\mathbf{74.5_{1.5}}$	$50.4_{1.0}$	$\mathbf{95.1_{0.3}}$	$50.4_{0.8}$
	CharacterTrajectories	$64.0_{1.5}$	$\mathbf{68.6_{1.4}}$	$96.2_{0.4}$	$44.9_{0.8}$	$66.4_{1.7}$	$\mathbf{68.7_{1.7}}$	$95.6_{0.5}$	$\mathbf{51.6_{0.6}}$	$\mathbf{80.1_{2.8}}$	$68.5_{1.2}$	$\mathbf{96.8_{0.4}}$	$50.1_{1.3}$
Unbalanced (10:1)	CEBS	$52.7_{0.7}$	$53.2_{1.0}$	$80.8_{1.5}$	$44.4_{0.6}$	$58.8_{0.7}$	$62.6_{0.6}$	$87.2_{1.3}$	$49.8_{0.5}$	$\mathbf{71.5_{1.3}}$	$\mathbf{62.9_{2.0}}$	$\mathbf{95.2_{1.3}}$	$\mathbf{50.2_{1.1}}$
	PAMAP	$65.8_{0.7}$	$\mathbf{70.4_{1.6}}$	$\mathbf{94.0_{1.8}}$	$41.5_{1.3}$	$72.9_{0.3}$	$63.5_{1.5}$	$90.9_{1.4}$	$57.6_{0.4}$	$\mathbf{85.5_{0.5}}$	$54.1_{2.2}$	$92.7_{1.4}$	$\mathbf{65.5_{0.8}}$
	PenDigits	$\mathbf{66.3_{0.9}}$	$63.3_{1.5}$	$89.7_{1.9}$	$47.9_{0.6}$	$64.4_{1.0}$	$59.1_{1.1}$	$86.0_{1.7}$	$\mathbf{56.2_{0.8}}$	$60.4_{0.9}$	$\mathbf{70.8_{1.8}}$	$\mathbf{90.5_{1.7}}$	$44.3_{0.7}$
	JapaneseVowels	$56.0_{1.2}$	$\mathbf{61.0_{1.0}}$	$90.3_{0.9}$	$45.7_{0.9}$	$56.0_{1.2}$	$55.4_{0.7}$	$87.3_{1.0}$	$\mathbf{53.9_{0.5}}$	$\mathbf{71.9_{1.4}}$	$56.2_{1.5}$	$\mathbf{91.8_{1.1}}$	$51.1_{0.9}$
	CharacterTrajectories	$64.1_{0.7}$	$67.0_{1.6}$	$88.4_{2.5}$	$46.0_{1.0}$	$69.3_{0.7}$	$\mathbf{67.2_{1.2}}$	$88.8_{2.2}$	$\mathbf{56.2_{0.8}}$	$\mathbf{80.2_{1.5}}$	$64.8_{2.0}$	$\mathbf{90.4_{2.6}}$	$53.9_{1.8}$

A.4 Encoder $\mathcal{E}$ and classifier architecture

As in traditional CL, we adopt a Siamese network with two identical and shared encoders $\mathcal{E}$ that are jointly trained, as shown in Fig. 2. Each encoder has three 1D convolutional layers (kernel size 3, stride 2, padding 1), mapping the C input channels to 4, 8, and 16 feature maps and halving the temporal resolution at each step, each followed by a `ReLU`. The resulting feature maps are flattened and projected by a fully connected layer to a fixed embedding of size $D = 16$. The input size depends on the dataset.

During contrastive pretraining, the embedding passes through a projection head, a two-layer `MLP` with `ReLU` that preserves dimensionality, on whose output the loss is computed. The projection head is discarded afterwards: clustering and the downstream classifiers operate on the $D = 16$ encoder embedding, as is standard in CL. This architecture is used identically across all compared pair-mining methods. Since K-ICL acts entirely at the mining stage, it is in principle agnostic to the encoder choice, and fixing the encoder isolates the effect of the mining rule.

Training setup. All methods are trained with the same protocol: `Adam` optimizer with learning rate 5×10^{-4}, batch size 32, and 500 epochs. We apply no early stopping and no checkpoint selection: the last-epoch encoder is evaluated, so that no method benefits from implicit label-based model selection, keeping the comparison fair and label-free given our few-label regime. Each configuration is run 10 times with different seeds: the seed controls the train/test split, initialization, and augmentations, and all methods within a run share the same split. We report the mean and standard error over runs. For downstream evaluation we fit, on the frozen embeddings, a k-nearest-neighbour classifier ($k = 5$) and a single-hidden-layer `MLP` (100 neurons). Clustering metrics use k-means with the number of clusters set to the number of classes.

A.4.1 SAMPLING RULE ABLATION

Table 8: Quality of the mined positive and negative sets for the mining-rule ablation in the unbalanced setting (minority classes at 1/10 of the dominant one). K-ICL samples positives and negatives probabilistically from the GAK similarities ($\tau^+=\tau^-=0.1$); RankHalf instead takes the top half of the batch by GAK similarity as positives and the bottom half as negatives; Thresh takes GAK$>$0.5 as positives and GAK$<$0.5 as negatives. Prec.$^+$/Rec.$^+$ are precision and recall of the positives, Prec.$^-$/Rec.$^-$ of the negatives (%, mean with SEM as subscript over 10 runs; each column is computed on the same runs, hence the same data splits). Best per metric in **bold**.

Dataset	K-ICL				RankHalf				Thresh			
	Prec.$^+$	Rec.$^+$	Prec.$^-$	Rec.$^-$	Prec.$^+$	Rec.$^+$	Prec.$^-$	Rec.$^-$	Prec.$^+$	Rec.$^+$	Prec.$^-$	Rec.$^-$
CEBS	$71.5_{1.3}$	$62.9_{2.0}$	$\mathbf{95.2_{1.3}}$	$50.2_{1.1}$	$53.3_{0.8}$	$\mathbf{88.3_{0.8}}$	$87.2_{0.9}$	$70.9_{0.9}$	$\mathbf{72.2_{2.8}}$	$78.9_{4.0}$	$86.7_{2.3}$	$\mathbf{87.6_{1.9}}$
PAMAP	$85.5_{0.5}$	$54.1_{2.2}$	$\mathbf{92.7_{1.4}}$	$65.5_{0.8}$	$64.2_{0.3}$	$\mathbf{86.6_{0.8}}$	$83.6_{1.1}$	$78.9_{0.5}$	$\mathbf{91.8_{1.0}}$	$63.6_{3.2}$	$75.3_{2.2}$	$\mathbf{98.1_{0.3}}$
PenDigits	$\mathbf{60.4_{0.9}}$	$70.8_{1.8}$	$90.5_{1.7}$	$44.3_{0.7}$	$51.1_{1.5}$	$84.9_{1.3}$	$85.6_{1.5}$	$\mathbf{68.8_{1.6}}$	$47.7_{0.7}$	$\mathbf{91.3_{1.8}}$	$\mathbf{91.9_{2.2}}$	$55.4_{2.3}$
JapaneseVowels	$71.9_{1.4}$	$56.2_{1.5}$	$\mathbf{91.8_{1.1}}$	$51.1_{0.9}$	$46.5_{1.5}$	$\mathbf{89.3_{1.0}}$	$90.9_{1.0}$	$67.1_{1.1}$	$\mathbf{75.7_{1.8}}$	$72.2_{2.7}$	$87.0_{1.5}$	$\mathbf{90.4_{1.4}}$
CharacterTrajectories	$80.2_{1.5}$	$64.8_{2.0}$	$\mathbf{90.4_{2.6}}$	$53.9_{1.8}$	$53.4_{2.0}$	$\mathbf{87.2_{1.8}}$	$86.9_{2.0}$	$71.1_{2.2}$	$\mathbf{82.9_{2.4}}$	$80.3_{3.6}$	$87.7_{2.3}$	$\mathbf{92.9_{2.4}}$

Tab. 8 reports the quality of the pairs mined by each rule, complementing the downstream results in the main text. K-ICL reaches the highest negative precision (Prec.$^-$) on four of the five datasets, confirming that its probabilistic sampling most effectively avoids false negatives. The deterministic rules instead have higher positive recall: both RankHalf and Thresh recover more same-class positives (higher Rec.$^+$), but at the cost of substantially lower negative precision, admitting more false negatives into the loss. RankHalf's high recall is a direct consequence of its fixed even split, which, under the 10:1 imbalance where the dominant class fills roughly half of each batch, happens to place many true positives in the "positive" half. Thresh is highly unstable across datasets: a fixed 0.5 cutoff suits the similarity scale of some datasets, but is poorly calibrated for others, so its precision and recall vary considerably. Overall, K-ICL is the only rule that keeps negative precision high without relying on a fixed proportion or threshold, and its cleaner mining translates into the strongest downstream results.

A.4.2 SAMPLING TEMPERATURES

Table 9: Ablation of the sampling temperatures in the unbalanced setting (minority classes at 1/10 of the dominant one). K-ICL uses $\tau^+=\tau^-=0.1$ in its probabilistic sampling; the two ablations keep the sampling unchanged and set both temperatures to 0.01 or 0.5. Results are reported as mean with SEM as subscript over 10 runs. Each column is computed on the same runs, hence the same data splits. Best per metric in **bold**.

Dataset	K-ICL (τ=0.1)				τ=0.01				τ=0.5			
	ARI	NMI	kNN	MLP	ARI	NMI	kNN	MLP	ARI	NMI	kNN	MLP
CEBS	$\mathbf{.54_{.03}}$	$\mathbf{.70_{.02}}$	$.71_{.03}$	$.77_{.03}$	$.40_{.02}$	$.61_{.01}$	$.57_{.02}$	$.60_{.02}$	$\mathbf{.54_{.02}}$	$\mathbf{.70_{.01}}$	$\mathbf{.71_{.02}}$	$\mathbf{.78_{.02}}$
PAMAP	$\mathbf{.70_{.04}}$	$\mathbf{.84_{.02}}$	$\mathbf{.83_{.03}}$	$\mathbf{.95_{.02}}$	$.68_{.05}$	$.81_{.04}$	$.81_{.05}$	$.85_{.04}$	$.68_{.05}$	$.81_{.03}$	$.79_{.03}$	$.86_{.03}$
PenDigits	$\mathbf{.46_{.02}}$	$\mathbf{.61_{.01}}$	$\mathbf{.65_{.02}}$	$\mathbf{.74_{.01}}$	$.41_{.01}$	$.58_{.01}$	$.59_{.02}$	$.71_{.01}$	$.43_{.01}$	$.59_{.01}$	$.62_{.01}$	$.71_{.02}$
JapaneseVowels	$.30_{.03}$	$.49_{.02}$	$.56_{.02}$	$.60_{.03}$	$.23_{.02}$	$.45_{.02}$	$.49_{.02}$	$.56_{.02}$	$\mathbf{.33_{.02}}$	$\mathbf{.53_{.01}}$	$\mathbf{.59_{.02}}$	$\mathbf{.63_{.02}}$
CharacterTrajectories	$\mathbf{.71_{.02}}$	$\mathbf{.84_{.01}}$	$\mathbf{.85_{.02}}$	$\mathbf{.94_{.01}}$	$.58_{.03}$	$.77_{.02}$	$.80_{.02}$	$.86_{.02}$	$.66_{.03}$	$.81_{.02}$	$\mathbf{.85_{.02}}$	$.91_{.01}$

Table 10: Quality of the mined positive and negative sets for the temperature ablation in the unbalanced setting (minority classes at 1/10 of the dominant one). K-ICL uses $\tau^+=\tau^-=0.1$ in its probabilistic sampling; the two ablations keep the sampling unchanged and set both temperatures to 0.01 or 0.5. Prec.$^+$/Rec.$^+$ are precision and recall of the positives, Prec.$^-$/Rec.$^-$ of the negatives (%, mean with SEM as subscript over 10 runs; each column is computed on the same runs, hence the same data splits). Best per metric in **bold**.

Dataset	K-ICL (τ=0.1)				τ=0.01				τ=0.5			
	Prec.$^+$	Rec.$^+$	Prec.$^-$	Rec.$^-$	Prec.$^+$	Rec.$^+$	Prec.$^-$	Rec.$^-$	Prec.$^+$	Rec.$^+$	Prec.$^-$	Rec.$^-$
CEBS	$71.5_{1.3}$	$62.9_{2.0}$	$95.2_{1.3}$	$\mathbf{50.2_{1.1}}$	$\mathbf{97.7_{0.2}}$	$29.1_{0.4}$	$\mathbf{99.4_{0.5}}$	$26.5_{1.9}$	$41.1_{0.5}$	$\mathbf{78.9_{0.5}}$	$83.8_{1.2}$	$34.8_{0.6}$
PAMAP	$85.5_{0.5}$	$54.1_{2.2}$	$92.7_{1.4}$	$\mathbf{65.5_{0.8}}$	$\mathbf{98.7_{0.1}}$	$24.1_{0.4}$	$\mathbf{100.0_{0.0}}$	$33.9_{2.0}$	$49.8_{0.6}$	$\mathbf{76.8_{0.6}}$	$79.1_{1.5}$	$38.9_{0.8}$
PenDigits	$60.4_{0.9}$	$70.8_{1.8}$	$90.5_{1.7}$	$\mathbf{44.3_{0.7}}$	$\mathbf{98.9_{0.3}}$	$32.6_{0.5}$	$\mathbf{96.1_{0.8}}$	$10.2_{0.3}$	$38.4_{0.3}$	$\mathbf{75.8_{0.3}}$	$79.8_{0.8}$	$31.4_{0.4}$
JapaneseVowels	$71.9_{1.4}$	$56.2_{1.5}$	$91.8_{1.1}$	$\mathbf{51.1_{0.9}}$	$\mathbf{99.9_{0.0}}$	$27.1_{0.1}$	$\mathbf{98.1_{0.6}}$	$11.4_{0.4}$	$33.6_{1.1}$	$\mathbf{77.6_{0.3}}$	$84.0_{0.9}$	$31.0_{0.4}$
CharacterTrajectories	$80.2_{1.5}$	$64.8_{2.0}$	$\mathbf{90.4_{2.6}}$	$\mathbf{53.9_{1.8}}$	$\mathbf{99.2_{0.1}}$	$29.3_{0.3}$	$90.2_{4.6}$	$13.8_{1.2}$	$41.1_{0.6}$	$\mathbf{78.7_{0.6}}$	$82.1_{1.3}$	$34.4_{0.8}$

Tab. 9 and 10 report the downstream and mining metrics as the sampling temperature varies around our default of $\tau^+ = \tau^- = 0.1$. The mining table makes the precision–recall trade-off explicit. At

$\tau = 0.01$ the sampling concentrates on the nearest neighbors, driving positive precision above 97% on every dataset (up to 99.9% on JapaneseVowels) but collapsing recall (positive recall around 30%, negative recall as low as 10%): the loss then sees only a few repeated pairs and the representation degrades, as seen in the drop across all downstream metrics. At $\tau = 0.5$ the distribution flattens, restoring recall but roughly halving positive precision (to 33–50%) and admitting many ill-defined pairs. The default $\tau = 0.1$ balances the two and attains the best or near-best downstream performance on every dataset, confirming that an intermediate temperature is what makes probabilistic mining effective.

A.5 Computational cost

All experiments were run on a single NVIDIA RTX 6000 Ada Generation GPU. Since the kernel is used only for pair mining and lies outside the computational graph, K-ICL adds no backpropagation memory cost to the encoder, and the only overhead is computing the within-batch similarities. This overhead need not recur every epoch: when augmentations are fixed, the similarities can be cached and reused across epochs, either per batch (if the batch composition is kept fixed) or, more flexibly, keyed by sample identity so that reuse is possible even when batches are shuffled. If the augmented views are precomputed before training, the similarities can likewise be precomputed once and removed entirely from the optimization loop.

B Formal Guarantees for Probabilistic Mining

This appendix provides the full assumptions and proofs underlying Theorem 3.1. For a fixed anchor a, let $\mathcal{P}_a^\star$ and $\mathcal{N}_a^\star$ denote the unknown sets of true positives and true negatives, respectively.

Assumptions.

A1 (finite candidate sets). For a fixed anchor a and minibatch $\mathcal{B}$, the true-positive and true-negative index sets $\mathcal{P}_a^\star, \mathcal{N}_a^\star \subseteq \mathcal{B}$ are finite, and $\mathcal{P}_a^\star \neq \emptyset$.

A2 (score separation). Let $s_{a,j} = \kappa(\mathbf{S}_a, A(\mathbf{S}_j))$. There exist $m_{\text{pos}}, m_{\text{neg}}, \gamma \in \mathbb{R}$ such that

$$\forall i \in \mathcal{P}_a^\star,\ m_{\text{pos}} \le s_{a,i}, \quad \forall j \in \mathcal{N}_a^\star,\ s_{a,j} \le m_{\text{neg}}, \quad m_{\text{neg}} \le m_{\text{pos}} - \gamma.$$

A3 (positive temperature). The positive-mining temperature is strictly positive, i.e., $\tau_{\text{pos}} > 0$.

A4 (bounded statistic). For each anchor a and candidate index i, let f_i denote a scalar statistic associated with pair (a, i), such as embedding-space cosine similarity ($f_i = \text{sim}(\mathbf{z}_a, \mathbf{z}_i)$), a normalized or clipped kernel similarity based on $s_{a,i}$, or an indicator score (e.g., $f_i = \mathbf{1}\{s_{a,i} \ge \theta\}$). We assume a fixed global constant $B \ge 0$ such that $|f_i| \le B$ on all relevant candidate indices. In practice, this is straightforward: $B = 1$ for cosine similarity and indicator scores; $B = 1$ for kernel scores normalized or clipped to $[0, 1]$ or $[-1, 1]$; and for unbounded raw scores g_i one can define a clipped statistic $f_i = \text{clip}(g_i, -B, B)$ so A4 holds by construction.

A5 (i.i.d. sampling with replacement). Sampled indices are i.i.d. draws from a target sampling distribution over candidate indices, i.e., for a fixed anchor a, the sampled positive indices satisfy $j_1^+, \ldots, j_{k_{\text{pos}}}^+ \overset{\text{i.i.d.}}{\sim} p_a^+$ and the sampled negative indices satisfy $j_1^-, \ldots, j_{k_{\text{neg}}}^- \overset{\text{i.i.d.}}{\sim} \tilde{p}_a^-$, with replacement in both cases.

A6 ($L1$ convergence of the empirical sampling distribution). If p denotes a target sampling distribution and $\hat{p}_n(i) := \frac{1}{n}\sum_{t=1}^n \mathbf{1}\{X_t = i\}$ is the empirical sampling distribution, then

$$\Delta_n := \sum_i |\hat{p}_n(i) - p(i)| \to 0.$$

Equivalently, the total variation satisfies $\text{TV}(\hat{p}_n, p) = \frac{1}{2}\Delta_n \to 0$. In our finite discrete setting with i.i.d. sampling (A5), this is the standard LLN behavior.

A7 (relaxed separation with overlap). To model overlapping positive and negative kernel-similarity tails, define

$$\mathcal{N}_{a,\text{ov}}^\star := \{j \in \mathcal{N}_a^\star : s_{a,j} > m_{\text{pos}} - \gamma\}.$$

Assume that its positive-softmax mass is bounded relative to positive mass:

$$\frac{\sum_{j\in\mathcal{N}^\star_{a,\mathrm{ov}}} w^+_{a,j}}{W_\mathcal{P}} \le \epsilon_{\mathrm{ov}},$$

for some $\epsilon_{\mathrm{ov}} \ge 0$.

Lemma B.1 (Separation controls off-target weight ratio) *For a fixed anchor a, define*

$$w^+_{a,j} := \exp\left(\frac{s_{a,j}}{\tau_{\mathrm{pos}}}\right), \qquad W_\mathcal{N} := \sum_{j\in\mathcal{N}^\star_a} w^+_{a,j}, \qquad W_\mathcal{P} := \sum_{i\in\mathcal{P}^\star_a} w^+_{a,i}.$$

*Under **A1–A3**,*

$$\rho := \frac{W_\mathcal{N}}{W_\mathcal{P}} \le \frac{|\mathcal{N}^\star_a|}{|\mathcal{P}^\star_a|} \exp\left(-\frac{\gamma}{\tau_{\mathrm{pos}}}\right). \tag{6}$$

Proof. By **A2**, for every $j \in \mathcal{N}^\star_a$,

$$s_{a,j} \le m_{\mathrm{neg}} \le m_{\mathrm{pos}} - \gamma.$$

By **A3**, $\tau_{\mathrm{pos}} > 0$, so dividing by τ_{pos} preserves inequalities, and monotonicity of exp gives

$$w^+_{a,j} = \exp\left(\frac{s_{a,j}}{\tau_{\mathrm{pos}}}\right) \le \exp\left(\frac{m_{\mathrm{neg}}}{\tau_{\mathrm{pos}}}\right).$$

Summing over $j \in \mathcal{N}^\star_a$,

$$W_\mathcal{N} \le |\mathcal{N}^\star_a| \exp\left(\frac{m_{\mathrm{neg}}}{\tau_{\mathrm{pos}}}\right).$$

Again by **A2**, for every $i \in \mathcal{P}^\star_a$,

$$w^+_{a,i} = \exp\left(\frac{s_{a,i}}{\tau_{\mathrm{pos}}}\right) \ge \exp\left(\frac{m_{\mathrm{pos}}}{\tau_{\mathrm{pos}}}\right),$$

hence

$$W_\mathcal{P} \ge |\mathcal{P}^\star_a| \exp\left(\frac{m_{\mathrm{pos}}}{\tau_{\mathrm{pos}}}\right).$$

By **A1**, $\mathcal{P}^\star_a \neq \emptyset$, so $W_\mathcal{P} > 0$ and the ratio is well-defined. Therefore

$$\frac{W_\mathcal{N}}{W_\mathcal{P}} \le \frac{|\mathcal{N}^\star_a|}{|\mathcal{P}^\star_a|} \exp\left(\frac{m_{\mathrm{neg}} - m_{\mathrm{pos}}}{\tau_{\mathrm{pos}}}\right) \le \frac{|\mathcal{N}^\star_a|}{|\mathcal{P}^\star_a|} \exp\left(-\frac{\gamma}{\tau_{\mathrm{pos}}}\right),$$

where the last step uses $m_{\mathrm{neg}} - m_{\mathrm{pos}} \le -\gamma$ from **A2**. □

Overlap-aware extension. If strict separation is relaxed through **A7**, define

$$\mathcal{N}^\star_{a,\mathrm{cl}} := \mathcal{N}^\star_a \setminus \mathcal{N}^\star_{a,\mathrm{ov}}.$$

Then, under **A1**, **A3**, the positive lower-band part of **A2**, and **A7**,

$$\rho \le \frac{|\mathcal{N}^\star_a|}{|\mathcal{P}^\star_a|} \exp\left(-\frac{\gamma}{\tau_{\mathrm{pos}}}\right) + \epsilon_{\mathrm{ov}}. \tag{7}$$

Indeed, we can split $W_\mathcal{N}$ into clean and overlap parts:

$$\frac{W_\mathcal{N}}{W_\mathcal{P}} = \frac{\sum_{j\in\mathcal{N}^\star_{a,\mathrm{cl}}} w^+_{a,j}}{W_\mathcal{P}} + \frac{\sum_{j\in\mathcal{N}^\star_{a,\mathrm{ov}}} w^+_{a,j}}{W_\mathcal{P}}.$$

For $j \in \mathcal{N}^\star_{a,\mathrm{cl}}$, by definition $s_{a,j} \le m_{\mathrm{pos}} - \gamma$, which yields the exponential term by the same denominator lower-bound argument as above. The overlap term is controlled directly by **A7**.

Theorem B.2 (Bounded-statistic gap) *Using the same weights as in Lemma B.1, define*

$$n_\mathcal{P} := \sum_{i\in\mathcal{P}^\star_a} w^+_{a,i} f_i, \qquad n_\mathcal{N} := \sum_{j\in\mathcal{N}^\star_a} w^+_{a,j} f_j,$$

and

$$\mu_{\mathrm{sur}} := \frac{n_\mathcal{P} + n_\mathcal{N}}{W_\mathcal{P} + W_\mathcal{N}}, \qquad \mu_{\mathrm{pos}} := \frac{n_\mathcal{P}}{W_\mathcal{P}}.$$

*Under **A1–A4** and Lemma B.1,*

$$|\mu_{\mathrm{sur}} - \mu_{\mathrm{pos}}| \le 2B \cdot \frac{W_\mathcal{N}}{W_\mathcal{P}} \le 2B \cdot \frac{|\mathcal{N}^\star_a|}{|\mathcal{P}^\star_a|} \exp\left(-\frac{\gamma}{\tau_{\mathrm{pos}}}\right). \tag{8}$$

Proof. By **A4** ($|f_i| \le B$),

$$|n_{\mathcal{P}}| \le B\, W_{\mathcal{P}}, \qquad |n_{\mathcal{N}}| \le B\, W_{\mathcal{N}}.$$

By **A1** and **A3**, $W_{\mathcal{P}} > 0$, so

$$\mu_{\text{sur}} - \mu_{\text{pos}} = \frac{n_{\mathcal{N}} - (W_{\mathcal{N}}/W_{\mathcal{P}})\, n_{\mathcal{P}}}{W_{\mathcal{P}} + W_{\mathcal{N}}}.$$

Hence, by triangle inequality,

$$|\mu_{\text{sur}} - \mu_{\text{pos}}| \le \frac{|n_{\mathcal{N}}| + (W_{\mathcal{N}}/W_{\mathcal{P}})|n_{\mathcal{P}}|}{W_{\mathcal{P}} + W_{\mathcal{N}}} \le 2B \cdot \frac{W_{\mathcal{N}}}{W_{\mathcal{P}}}.$$

Applying Lemma B.1 yields equation 8. □

Overlap-aware bound. If strict separation fails but **A7** holds, then equation 7 gives

$$|\mu_{\text{sur}} - \mu_{\text{pos}}| \le 2B \left(\frac{|\mathcal{N}_a^\star|}{|\mathcal{P}_a^\star|} \exp\left(-\frac{\gamma}{\tau_{\text{pos}}} \right) + \epsilon_{\text{ov}} \right). \tag{9}$$

Theorem B.3 (Empirical convergence) *Let $\mathcal{S}$ be a finite index set for one fixed anchor and one mining side, let p denote the corresponding target sampling distribution on $\mathcal{S}$, and let*

$$\hat{p}_n(i) := \frac{1}{n} \sum_{t=1}^{n} \mathbf{1}\{X_t = i\}, \qquad i \in \mathcal{S}.$$

Define

$$\Delta_n := \sum_{i \in \mathcal{S}} |\hat{p}_n(i) - p(i)|, \qquad \mu_n := \sum_{i \in \mathcal{S}} \hat{p}_n(i) f_i, \qquad \mu := \sum_{i \in \mathcal{S}} p(i) f_i.$$

*Under **A4–A6**,*

$$|\mu_n - \mu| \le B\, \Delta_n = B \sum_{i \in \mathcal{S}} |\hat{p}_n(i) - p(i)|, \tag{10}$$

and consequently

$$\mu_n \to \mu. \tag{11}$$

Proof. For each n,

$$\mu_n - \mu = \sum_{i \in \mathcal{S}} (\hat{p}_n(i) - p(i))\, f_i.$$

By triangle inequality and **A4**,

$$|\mu_n - \mu| \le \sum_{i \in \mathcal{S}} |\hat{p}_n(i) - p(i)|\, |f_i| \le B \sum_{i \in \mathcal{S}} |\hat{p}_n(i) - p(i)| = B\, \Delta_n,$$

which proves equation 10. Then **A6** gives $\Delta_n \to 0$, hence equation 11. Assumption **A5** specifies the i.i.d. sampling mechanism under which **A6** holds in the present finite setting. □

Discussion. Theorem B.2 shows that the statistic induced by the surrogate miner stays close to the ideal positive-only statistic whenever the softmax mass assigned to true negatives is small relative to the mass assigned to true positives. Lemma B.1 makes this dependence explicit in terms of the separation margin γ and the positive-mining temperature τ_{pos}, which is the compact guarantee reported in the main text. Theorem B.3 complements this finite-batch view with an asymptotic statement showing that the empirical statistics produced by repeated stochastic sampling converge to their target quantities.