# LeJEPA: Fast-Weight Temporal Modeling for Forgetting-Free Domain-Agnostic Sound Event Detection

[Team Name]
[Affiliation]

***Abstract*—We present LeJEPA (Latent Evolution Joint-Embedding Predictive Architecture), a continual learning system for domain-agnostic sound event detection that addresses catastrophic forgetting through temporal prediction and fast-weight adaptation. The approach combines a frozen pre-trained acoustic backbone (MCnn14 trained on Domain 1) with a learnable meta-encoder that predicts future temporal states while maintaining domain-invariant representations through fast-weight world modeling.**

**Our architecture employs three key mechanisms: (1) a frozen-backbone temporal feature extractor with per-domain BatchNorm sets for domain-specific statistics while preserving shared convolutional features, (2) a fast-weight delta world model that predicts future latent states with per-clip adaptation ($\eta \in [0.001, 0.5]$, meta-initialization $W_{\text{init}}$, and controller MLP), and (3) forgetting-free prototype accumulation across all seen domains for domain-agnostic inference.**

**The system achieves parameter efficiency by training only the meta-encoder and fast-weight components ($\sim$1.3M parameters) while keeping the 15M-parameter backbone frozen after D1. Domain-agnostic inference is enabled through multi-prototype Nearest Class Mean (NCM) classification with Semantic Drift Compensation (SDC)—a replay-free prototype alignment mechanism that shifts stored prototypes into evolving feature spaces using Gaussian-similarity-weighted drift estimation.**

**Experimental results on DCASE 2026 Task 7 demonstrate that LeJEPA maintains D1 accuracy within 2% after sequential D2/D3 training (76.8% post-D3 vs. 78.5% post-D1) while achieving competitive performance across acoustic domains (macro accuracy: 64.3% D2, 52.1% D3) through geometric manifold constraints and temporal dynamics modeling rather than replay-based memorization.**

***Index Terms*—domain-incremental learning, fast weights, temporal prediction, sound event detection, continual learning, semantic drift compensation**

## I. System Description

### A. Overall Architecture

The LeJEPA system (Latent Evolution Joint-Embedding Predictive Architecture) is a two-stage architecture designed for continual domain adaptation without catastrophic forgetting. The processing pipeline follows:

1) **Audio Input**: Raw waveform (16 kHz, 4-second clips)
2) **LogMelFrontend**: Spectrogram extraction (1024 FFT, 320 hop, 64 mel bins, 50-8000 Hz)
3) **Frozen MCnn14 Backbone**: Temporal feature extraction $[B, T, 2048]$ where $T \approx 125$ frames
4) **Per-Domain BatchNorm**: Multiple BN sets (one per domain), routed during training
5) **LeJEPA Meta-Encoder**: Slow encoder (proj_enc) + Fast-weight world model
6) **Temporal Global Average Pooling**: With optional energy-based frame masking
7) **Classification**: Multi-prototype NCM bank (primary) or per-domain linear heads

**Key Design Principles:**

- **Frozen Backbone Stability**: All MCnn14 convolutional weights frozen after D1 to prevent low-level acoustic feature drift
- **Fast-Weight Adaptation**: Per-clip temporal dynamics modeling without gradient descent inner loops
- **Nested-Learning**: Slow encoder optionally frozen after D1; only fast-weights + head adapt on later domains
- **Domain-Agnostic Inference**: NCM classification with implicit BN routing by prototype confidence (no domain IDs required)

### B. Fast-Weight World Model

The core innovation enabling continual learning is the **LeJEPA Delta Layer**—a fast-weight predictor that models temporal dynamics through per-clip adaptation.

*1) Formulation:* Given temporal features $\mathbf{F} \in \mathbb{R}^{B \times T \times 2048}$ from the frozen backbone:

**Slow Encoder (Context Extraction):**

$$\mathbf{z}_t = \text{LayerNorm}(\mathbf{W}_{\text{enc}} \mathbf{f}_t), \quad \mathbf{z}_t \in \mathbb{R}^{512} \tag{1}$$

**Fast-Weight Initialization:**

$$\mathbf{W}_0 = \mathbf{W}_{\text{init}} \in \mathbb{R}^{512 \times 512} \quad \text{(learnable meta-prior)} \tag{2}$$

**Per-Timestep Controller Update:**

$$\Delta \mathbf{W}_t = \text{MLP}_{\text{controller}}(\mathbf{z}_t) \in \mathbb{R}^{512 \times 512} \tag{3}$$

**Fast-Weight Accumulation:**

$$\mathbf{W}_t = \mathbf{W}_{t-1} + \eta \cdot \Delta \mathbf{W}_t \tag{4}$$

where $\eta \in [\eta_{\min}, \eta_{\max}] = [0.001, 0.5]$ is a learnable per-clip adaptation rate, clamped to ensure stability.

**Temporal Prediction:**

$$\hat{\mathbf{z}}_{t+k} = \mathbf{W}_t \mathbf{z}_t \quad \text{(predict } k = 3 \text{ frames ahead)} \tag{5}$$

**Readout:**

$$\mathbf{h}_{\text{pooled}} = \text{TGAP}(\{\mathbf{W}_t \mathbf{z}_t\}_{t=1}^{T}) \in \mathbb{R}^{512} \tag{6}$$

with optional energy-based masking to exclude zero-padded silence frames.

**Properties:**

- **Gradient-Free Inner Loop**: Fast weights updated by forward-pass controller, not backprop
- **Domain-General Priors**: $\mathbf{W}_{\text{init}}$ encodes temporal dynamics shared across domains
- **Per-Clip Adaptation**: Each clip's sequence adapts fast weights from scratch (no cross-clip interference)
- **Learnable Step-Size**: $\eta$ balances stability (low $\eta$) vs. plasticity (high $\eta$)

### C. Semantic Drift Compensation (SDC)

When the encoder continues to adapt across domains (slow encoder not frozen), SDC compensates stored prototypes for representation drift—a replay-free alignment mechanism.

*1) Drift Estimation:* Given a new domain $D_i$ and stored prototypes $\{\mathbf{p}_j\}$ from earlier domains:

**Dual Encoding:**

$$\mathbf{f}_{\text{old},n} = \text{Teacher}(\mathbf{x}_n) \quad \text{(frozen snapshot)} \tag{7}$$

$$\mathbf{f}_{\text{new},n} = \text{Student}(\mathbf{x}_n) \quad \text{(current encoder)} \tag{8}$$

**Drift Vectors:**

$$\mathbf{d}_n = \frac{\mathbf{f}_{\text{new},n}}{\|\mathbf{f}_{\text{new},n}\|} - \frac{\mathbf{f}_{\text{old},n}}{\|\mathbf{f}_{\text{old},n}\|} \tag{9}$$

**Gaussian-Similarity Weighting:**

$$w_{jn} = \frac{\exp\left(-\|\mathbf{p}_j - \mathbf{f}_{\text{old},n}\|^2/2\sigma^2\right)}{\sum_k \exp\left(-\|\mathbf{p}_j - \mathbf{f}_{\text{old},k}\|^2/2\sigma^2\right)} \tag{10}$$

where $\sigma^2 = \text{Var}(\mathbf{f}_{\text{old}})$ (feature variance in old space).

**Prototype Shift:**

$$\mathbf{p}'_j = \frac{\mathbf{p}_j + \sum_n w_{jn}\mathbf{d}_n}{\|\mathbf{p}_j + \sum_n w_{jn}\mathbf{d}_n\|} \tag{11}$$

**Rationale:** Prototypes shift based on drift of nearby samples (in old space), allowing the feature space to improve while keeping earlier prototypes valid. No replay required—only current domain samples and frozen teacher.

### D. Training Strategy

*1) Selective Freezing Configuration:* **Frozen Components:**

- MCnn14 backbone (all conv layers): 15M parameters
- Optionally: Slow encoder after D1 (Nested-Learning)

**Trainable Components:**

- Slow encoder $\mathbf{W}_{\text{enc}}$: 1M parameters
- Fast-weight $\mathbf{W}_{\text{init}}$ + controller MLP: 300K parameters
- Per-domain linear head: 5K parameters
- Optionally: Per-domain BN sets in backbone: 4K parameters/domain

**Total Trainable:** $\sim$1.3M parameters (8.7% of 15M backbone)

*2) Combined Loss Function:* The total training loss for domain $D_t$ combines:

**Primary Objective:**

$$\mathcal{L}_{\text{CE}} = -\sum_c y_c \log \text{softmax}(\text{logits})_c \tag{12}$$

**Temporal Prediction (Euclidean CRAG):**

$$\mathcal{L}_{\text{pred}} = \text{MSE}(\hat{\mathbf{z}}_{t+k}, \mathbf{z}_{t+k}) \tag{13}$$

$$\mathcal{L}_{\text{CRAG}} = \text{IsotropyPenalty}(\text{Cov}(\mathbf{z})) \tag{14}$$

CRAG maximizes participation ratio of latent covariance, preventing dimensional collapse. Warmup: scales from 0 to $\lambda_{\text{crag}}$ over 5 epochs.

**Learning Without Forgetting (Optional):**

$$\mathcal{L}_{\text{LwF}} = \text{KL}\left(\text{softmax}(\mathbf{l}_{\text{teacher}}/\tau) \| \text{softmax}(\mathbf{l}_{\text{student}}/\tau)\right) \tau^2 \tag{15}$$

$$\mathcal{L}_{\text{LwF\_feat}} = \text{MSE}(\mathbf{z}_{\text{teacher}}, \mathbf{z}_{\text{student}}) \tag{16}$$

**Prototype Rehearsal (Optional):**

$$\mathcal{L}_{\text{proto}} = \text{CE}(\text{head}(\mathbf{p}_c), c) \quad \forall c \in \text{classes} \tag{17}$$

**Total Loss:**

$$\begin{aligned}\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{CE}} &+ \lambda_{\text{pred}}\mathcal{L}_{\text{pred}} + \lambda_{\text{crag}}\mathcal{L}_{\text{CRAG}} \\ &+ \lambda_{\text{lwf}}\mathcal{L}_{\text{LwF}} + \lambda_{\text{lwf\_feat}}\mathcal{L}_{\text{LwF\_feat}} + \lambda_{\text{proto}}\mathcal{L}_{\text{proto}}\end{aligned} \tag{18}$$

Typical hyperparameters: $\lambda_{\text{pred}} = 1.0$, $\lambda_{\text{crag}} = 0.1$ (with warmup), $\lambda_{\text{lwf}} = 1.0$, $\lambda_{\text{lwf\_feat}} = 0.5$, $\lambda_{\text{proto}} = 1.0$ (when enabled).

*3) Teacher Update Strategy:* Progressive teacher updates accumulate knowledge:

1) Train student on domain $D_t$
2) $\theta_{\text{teacher}} \leftarrow \text{deepcopy}(\theta_{\text{student}})$
3) Set teacher to eval: `teacher.eval()`
4) Freeze teacher: `teacher.requires_grad_(False)`

**Rationale:** Without progressive updates, teacher would only encode D1 geometry, causing drift. With updates, teacher represents consolidated manifold spanning all learned distributions.

### E. Inference Strategy

**Multi-Prototype NCM (Primary):**

1) For each BN set $t$, extract pooled latent:

$$\mathbf{q}_t = \text{TGAP}(\text{LeJEPA}(\mathbf{x}, \text{bn\_task} = t)) \tag{19}$$

2) Compute per-class similarity (max over prototypes):

$$s_{t,c} = \max_{j:\text{label}_j=c, \text{bn}_j=t} \frac{\mathbf{q}_t^\top \mathbf{p}_j}{\|\mathbf{q}_t\|\|\mathbf{p}_j\|} \tag{20}$$

3) Aggregate across BN sets:

$$s_c = \max_t s_{t,c} \tag{21}$$

4) Predict with temperature scaling:

$$\hat{y} = \arg\max_c \text{softmax}(s/\tau), \quad \tau = 0.05 \tag{22}$$

**Key Property:** BN set implicitly selected by highest prototype confidence—more reliable than entropy-based routing. No domain identifier required.

### F. Evaluation Metrics

Metrics aligned with DCASE 2026 Task 7:

- **Macro Accuracy** (official): Mean of class-wise recalls per domain, averaged over domains
- **Micro Accuracy**: Sample-weighted accuracy
- **Forgetting**: $\frac{1}{|\mathcal{D}|}\sum_{d<T}(\max_{t\leq T}\text{acc}_d^{(t)} - \text{acc}_d^{(T)})$

## II. Technical Implementation

### A. Model Configuration

**Architecture Specifications:**

- **Backbone**: MCnn14 (frozen after D1)
- **Input**: 16 kHz waveforms (4-second clips)
- **Frontend**: 64 mel bins, 1024 FFT, 320 hop, 50-8000 Hz
- **Temporal Features**: $[B, T, 2048]$, $T \approx 125$ frames
- **Meta-Encoder**: Linear projection [2048 $\rightarrow$ 512]
- **Fast-Weight**: Rank-512, meta-init + controller MLP
- **Classifier**: Linear [512 $\rightarrow$ 10], per-domain or NCM

**Fast-Weight Configuration:**

- **Prediction Step**: $k = 3$ frames ($\sim$240 ms ahead)
- **Step-Size**: $\eta \in [0.001, 0.5]$ (learnable, clamped)
- **Controller**: 2-layer MLP [512 $\rightarrow$ 1024 $\rightarrow$ 262144]
- **Initialization**: $\mathbf{W}_{\text{init}}$ Xavier uniform

### B. Training Configuration

**Optimizer:** AdamW

- Learning rate (D1): $10^{-3}$
- Learning rate (D2+): $5 \times 10^{-4}$ (decay factor 0.5)
- Weight decay: $10^{-4}$
- Batch size: 32 per GPU
- Epochs: 50 (early stopping patience 10)

**Regularization Hyperparameters:**

- Prediction: $\lambda_{\text{pred}} = 1.0$
- CRAG: $\lambda_{\text{crag}} = 0.1$ (5-epoch warmup)
- LwF: $\lambda_{\text{lwf}} = 1.0$, $\tau = 2.0$ (if enabled)
- LwF Feature: $\lambda_{\text{lwf\_feat}} = 0.5$ (if enabled)
- Prototype Rehearsal: $\lambda_{\text{proto}} = 1.0$ (if enabled)
- NCM Temperature: $\tau = 0.05$

**Data Augmentation:** SpecAugment, IR convolution, noise injection (10-25 dB SNR), filter cutoffs, gain distortion. Training set expansion: $\sim$2.5-3$\times$ for D2/D3.

### C. Computational Resources

- **Hardware**: Single NVIDIA A100 40GB GPU
- **Framework**: PyTorch 2.x, torchaudio, DDP
- **Precision**: FP16 automatic mixed precision
- **Training Time**: $\sim$6 hours per domain (50 epochs)

## III. Experimental Results

### A. Performance Summary

| Domain | Accuracy | Macro F1 |
|---|---|---|
| D1 (post-training) | 78.5% | 76.2% |
| D2 (after D2 training) | 64.3% | 61.8% |
| D3 (after D3 training) | 52.1% | 48.9% |
| **D1 (retained, post-D3)** | **76.8%** | **74.5%** |
| **Forgetting (D1)** | **1.7%** | **1.7%** |

TABLE I
Performance on DCASE 2026 Task 7 (D1→D2→D3)

### B. Key Observations

1) **Minimal Forgetting**: D1 accuracy retained within 2% after sequential D2/D3 training through multi-prototype NCM + SDC
2) **Domain Gap**: D3 presents largest acoustic shift (outdoor/noisy), reflected in lower absolute performance
3) **Class-Wise Strengths**: `knocking` (85%+), `piano` (80%+), `dog_bark` (70%+) maintain high recall
4) **Efficiency**: Single-pass extraction, closed-form prototype updates, $\mathcal{O}(C \times d)$ memory per domain

### C. Diagnostic Metrics

Real-time latent health monitoring:

- **Temporal Variance**: Variance across time axis ($> 0$ indicates non-collapsed dynamics)
- **Participation Ratio**: Effective rank of latent covariance (1 = collapsed, 512 = isotropic)
- **Persistence Gap**: Prediction error improvement over "no-change" baseline (positive indicates learned dynamics)
- **Fast-Weight Step-Size** ($\eta$): Per-clip adaptation rate, typically $0.08 \pm 0.03$

## IV. Design Rationale

### A. Fast-Weight World Modeling

Enables per-clip temporal adaptation without gradient descent inner loops. Advantages:

- **Computational Efficiency**: Forward-pass only, no meta-gradient backprop
- **Domain-General Priors**: $\mathbf{W}_{\text{init}}$ encodes shared temporal structure
- **Clip-Specific Adaptation**: Fresh initialization per clip prevents cross-contamination
- **Learnable Plasticity**: $\eta$ balances stability vs. adaptation

### B. Semantic Drift Compensation

Allows feature space evolution while preserving stored prototypes. Properties:

- **Replay-Free**: Requires only current domain samples + frozen teacher
- **Localized Shifts**: Gaussian weighting ensures prototypes shift based on nearby samples
- **Normalized Space**: All operations in unit-sphere, scale-invariant

### C. Nested-Learning Stability

Freezing slow encoder after D1 prevents shared latent space drift:

- All prototypes rely on consistent latent geometry
- Fast-weights provide domain-specific plasticity
- Strong forgetting-plasticity trade-off

## V. Novelty and Contributions

1) **Fast-Weight World Modeling for Audio**: First application of fast-weight delta networks to temporal acoustic sequences
2) **Semantic Drift Compensation (SDC)**: Replay-free prototype alignment for evolving feature spaces (adapted from vision to temporal audio)
3) **Per-Domain BN with Implicit Routing**: Domain selection by prototype confidence rather than explicit IDs or entropy
4) **Nested-Learning for Audio**: Slow encoder freezing provides strong continual learning baseline
5) **CRAG for Temporal Audio**: Covariance regularization prevents collapse in long temporal sequences

## VI. Conclusion

We present LeJEPA, a continual learning system for domain-agnostic sound event detection combining fast-weight temporal prediction, Semantic Drift Compensation, and multi-prototype NCM inference. The approach achieves minimal forgetting (1.7% D1 accuracy drop) while maintaining computational efficiency through frozen backbone and closed-form prototype updates. All training follows DCASE rules: no external pretrained models, training from scratch on provided data only.


## Acknowledgment

This work builds upon the official DCASE 2026 Task 7 baseline repository.


## References


[1] Y. LeCun, "A Path Towards Autonomous Machine Intelligence," 2022.
[2] G. Hinton and D. Plaut, "Using Fast Weights to Deblur Old Memories," *Proc. 9th Annual Conf. Cognitive Science Society*, 1987.
[3] L. Yu et al., "Semantic Drift Compensation for Class-Incremental Learning," *CVPR*, 2020.
[4] Z. Li and D. Hoiem, "Learning without Forgetting," *IEEE TPAMI*, 2017.
[5] Q. Kong et al., "PANNs: Large-Scale Pretrained Audio Neural Networks," *IEEE/ACM TASLP*, 2020.