Self-Supervised Predictive Learning: A Negative-Free Method for Sound Source Localization in Visual Scenes
(Supplementary Material)

Zengjie Song\textsuperscript{1}  Yuxi Wang\textsuperscript{1,3}  Junsong Fan\textsuperscript{1,2}  Tieniu Tan\textsuperscript{1,2}  Zhaoxiang Zhang\textsuperscript{1,2,3*}

\textsuperscript{1}Center for Research on Intelligent Perception and Computing, NLPR, CASIA
\textsuperscript{2}University of Chinese Academy of Sciences (UCAS)
\textsuperscript{3}Centre for Artificial Intelligence and Robotics, HKISI_CAS

\{zengjie.song, wangyuxi2016, fanjunsong2016, zhaoxiang.zhang\}@ia.ac.cn, tnt@nlpr.ia.ac.cn

This supplementary material contains four parts:

- Section A presents full derivations to formulate the representation update rules of predictive coding module (PCM).
- Section B provides more details on our implementation.
- Section C compares the localization performance of Attention [9] and HardWay [1] in original papers with our reproductions.
- Section D gives additional ablation and visualisation results.

A. Full Formulation of PCM

The PCM, proposed for audio and visual feature alignment, plays an important role in improving sound localization performance of SSPL. As shown in Figure S1, the key idea underlying PCM consists of three parts: (1) a feedback process (solid line) updates representations with the top-down predictions that originate from the visual feature; (2) a feedforward process (dashed line) also updates representations but with the bottom-up prediction errors that evolve from the audio feature; (3) a recursive modulation mechanism works to conduct the two processes alternatively. In the following, we first formulate the optimization objective of PCM, and then derive the representation update rules of the two processes, respectively, which are followed by a brief summary and a formal algorithm. Note that for applications of PCM, we only need to explicitly update representations according to the rules given in Eqs. (S10) to (S13), without performing derivations again.

Denote by \( f_a \) the audio feature, by \( f_v \) the visual feature, by \( r_l(t), l \in \{1, \ldots, L\}, t \in \{0, \ldots, T\} \) the representation of the \( l \)-th layer of PCM network at time step \( t \), and by \( W_{l,l-1} \) the feedback connection weights from layer \( l \) to layer \( l-1 \) (and vice versa for \( W_{l-1,l} \)).

**Optimization Objective.** At layer \( l \), PCM minimizes the following compound loss:

\[
\mathcal{L}_{PCM}^l = \frac{\alpha_l}{2} \left\| r_{l-1} - G((W_{l,l-1})^T r_l) \right\|_2^2 + \frac{\beta_l}{2} \left\| r_l - p_l \right\|_2^2
\]

(S1)

where the function \( G \) corresponds to a generative process, \( \alpha_l \) and \( \beta_l \) are scalars that control the weights of the two loss terms \( \mathcal{L}_1^l \) and \( \mathcal{L}_2^l \), and \( p_l = G((W_{l+1,l})^T r_{l+1}) \) is the prediction of \( r_l \).

Given the lower-level representation \( r_{l-1} \) and the top-down prediction \( p_l \), our goal is to estimate \( r_l \) so as to decrease the loss in Eq. (S1). Minimizing \( \mathcal{L}_1^l \) w.r.t. \( r_l \) leads to the representation that can be used to predict the lower level of representation \( r_{l-1} \), while minimizing \( \mathcal{L}_2^l \) w.r.t. \( r_l \) yields the representation that approximates the prediction signal \( p_l \) coming from a higher level. Therefore, the representation \( r_l \) associates lower- and higher-level information by reducing...
two prediction errors in $\mathcal{L}_1^t$ and $\mathcal{L}_2^t$. Minimizing losses at all layers can implicitly drive predictions at different levels to be mutually consistent [10].

Feedback Process. This process acts to update representations based on predictions from higher levels. Following [11], we set $g(x) = x$, and then employ gradient descent to minimize $\mathcal{L}_t^2$, resulting in update rules:

$$p_t(t) = (W_{t+1})^T r_{t+1}(t),$$  \hspace{1cm} \text{(S2)}$$

$$\frac{\partial \mathcal{L}_t^2}{\partial r_t(t)} = 2(r_t(t) - p_t(t)), \hspace{1cm} \text{(S3)}$$

$$r_t(t + 1) = r_t(t) - \eta_1 \frac{\partial \mathcal{L}_t^2}{\partial r_t(t)} = (1 - \eta_1 b_t) r_t(t) + \eta_1 p_t(t), \hspace{1cm} \text{(S4)}$$

where $\eta_1$ is a non-negative scalar governing learning. For simplicity, let $b_t = \eta_1 \beta_1$, and then Eq. (S4) is rewritten as follows:

$$r_t(t + 1) = (1 - b_t) r_t(t) + b_t p_t(t). \hspace{1cm} \text{(S5)}$$

PCM carries out the feedback updating from top layer $L$ to bottom layer 1, where the prediction of $r_L(t)$ at top layer is set as the visual feature, i.e., $p_L(t) = f_v$.

Feedforward Process. This process works to update representations by using prediction errors from lower layers. For layer $l$, the lower-layer prediction error $e_{l-1}$ is the difference between $r_{l-1}$ and $p_{l-1}$. We use gradient descent to minimize $\mathcal{L}_l^1$ w.r.t. $r_l$, leading to the following update rules:

$$e_{l-1}(t) = r_{l-1}(t) - p_{l-1}(t), \hspace{1cm} \text{(S6)}$$

$$\frac{\partial \mathcal{L}_l^1}{\partial r_l(t)} = -2W_{l-1} e_{l-1}(t), \hspace{1cm} \text{(S7)}$$

$$r_l(t + 1) = r_l(t) - \kappa_l \alpha_l \frac{\partial \mathcal{L}_l^1}{\partial r_l(t)} = r_l(t) + \kappa_l \alpha_l W_{l-1} e_{l-1}(t), \hspace{1cm} \text{(S8)}$$

where $\kappa_l$ is a non-negative scalar like $\eta_1$. We also set $\alpha_l = \kappa_l \alpha_l$ for simplicity. Similar to [11], we replace the feedback connection weights $W_{l-1}$ in Eq. (S8) with the transposed feedforward connection weights $(W_{l-1})^T$, and thus can endow PCM with more degrees of freedom to learn. Consequently the update rule in Eq. (S8) can be rewritten as a feedforward operation:

$$r_l(t + 1) = r_l(t) + \alpha_l(W_{l-1})^T e_{l-1}(t). \hspace{1cm} \text{(S9)}$$

In this process, PCM updates representations from bottom layer 1 to top layer $L$, where we let $p_0(t) = f_a$ and $p_b(t) = (W_{1,0})^T r_1(t)$.

Summary and Algorithm. So far we formulate PCM with the simple linear activation functions. To introduce non-linearity into PCM, a nonlinear activation function $\phi$ (e.g., ReLU [8] used in [11] or GELU [4] used in this work) is applied to the above update Eqs. (S5) and (S9). By taking the recursive computing into account, we summarize the two processes as follows.

Nonlinear feedback process ($l = L, L - 1, \ldots, 1$):

$$p_t(t) = (W_{t+1,l})^T r_{t+1}(t), \hspace{1cm} \text{(S10)}$$

$$r_t(t) \leftarrow \phi((1 - b_t)r_t(t) - 1 + b_t p_t(t)). \hspace{1cm} \text{(S11)}$$

Nonlinear feedforward process ($l = 1, 2, \ldots, L$):

$$e_{l-1}(t) = r_{l-1}(t) - p_{l-1}(t), \hspace{1cm} \text{(S12)}$$

$$r_l(t) \leftarrow \phi(r_l(t) + \alpha_l(W_{l-1})^T e_{l-1}(t)). \hspace{1cm} \text{(S13)}$$

The two processes are conducted alternatively such that all representations in PCM are refined progressively. Finally, we transform the top layer representation at last time step, $r_L(T)$, to a new visual feature, $f_v$, with dimension the same as $f_a$ by a $1 \times 1$ convolution. The representation learning of SSPL can proceed based on this $f_v$, instead of $f_a$ as used in the vanilla SSPL. We present main computing steps of PCM in Algorithm S1.

---

**Algorithm S1 Update Representations in PCM**

**Input:** $f_v$ and $f_a$

**Output:** $f_v$

1: for $t = 0$ to $T$ do

2: \quad if $t = 0$ then

3: \quad \quad initialize representations

4: \quad end if

5: \quad for $l = L$ to 1 do

6: \quad \quad if $l = L$ then

7: \quad \quad \quad $p_t(t) = f_v$

8: \quad \quad else

9: \quad \quad \quad compute prediction $p_t(t)$: Eq. (S10)

10: \quad \quad end if

11: \quad update representation $r_t(t)$: Eq. (S11)

12: \quad end for

13: \quad for $l = 1$ to $L$ do \hspace{1cm} \text{feedback process}

14: \quad \quad if $l = 1$ then

15: \quad \quad \quad $e_{l-1}(t) = f_a - \phi((W_{l-1,l})^T r_t(t))$

16: \quad \quad \quad else

17: \quad \quad \quad obtain prediction error $e_{l-1}(t)$: Eq. (S12)

18: \quad \quad \quad end if

19: \quad \quad update representation $r_l(t)$: Eq. (S13)

20: \quad end for

21: end for

22: $f_v = \text{conv}_{1 \times 1}(r_L(T))$ \hspace{1cm} \text{transformed feature}
### B. Implementation Details

#### B.1. Architecture of PCM

For the feedback process of PCM, we use convolution layers \((\text{kernel size} = 3, \text{stride} = 1, \text{padding} = 1)\) followed by max pooling operation to reduce the spatial dimensionality of feature maps, while using \(1 \times 1\) convolutions to decrease the number of channels. As for the feedforward process, the transposed convolutions (a.k.a. deconvolutions) are utilized and feature maps are upsampled by the “bilinear” upsampling algorithm, provided in PyTorch. Besides, the number of convolution layers is \(L = 3\). From top layer \(L\) to bottom layer 1, the number of filters within each layer is 512, 512, and 128, respectively. The transposed convolution layers have the same setting. Moreover, we use GELU \([4]\) as the nonlinear activation function for both processes. To stabilize and accelerate training, we adopt the batch normalization \([6]\) before every non-linearity at each layer and at each time step, except the prediction of audio feature at bottom layer.

#### B.2. Training Details for SSPL

The AdamW \([7]\) optimizer is employed to train our model, where we set \((\beta_1, \beta_2) = (0.9, 0.999)\) and set weight decay to \(10^{-4}\). In practice, we find that better performance could be achieved if the learning rate for projection and prediction MLPs is greater than that for remaining model parts. The quantitative comparisons in the main text are partially based on our reproductions of two related methods: Attention \([9]\) and HardWay \([1]\). We reimplement them as faithfully as possible by following each corresponding paper. As show in Table S3, we are able to improve these two methods on SoundNet-Flickr by small and straightforward modifications. Specifically, we use Crop and Horizontal flip given in Table S2 to spatially augment images for Attention (vs. originally \(320 \times 320\) resizing), same as HardWay and our method. We also fine tune the learning rate and weight decay for these two competitors in order to achieve their best performance.

Table S4 compares our reproductions with original papers’ results on VGG-SS. In this case our reproductions are slightly lower than the original counterparts. Note that we had tried to adjust various hyper-parameters for HardWay training (e.g., learning rate, weight decay, batch size, and number of training epochs) for multiple times, but better performance than reproductions shown in Table S4 were not achieved. We contribute the performance discrepancy to the updated data in this benchmark. On the one hand, Chen et al. \([1]\) originally provides 5158 YouTube video IDs for testing, and users need to download, extract, and pre-process the designated audio and visual sources themselves. However, 466 videos (9%) are not available (removed or prohibited download) at the time of conducting our experiments, leading to 4692 image-audio pairs for final testing. On the other hand, as clarified by the authors on the official project

| Augmentation     | Parameter                                      |
|------------------|------------------------------------------------|
| Crop             | \(p = 1\)                                     |
|                  | output size of \(\text{Resize} = \text{int}(224 \times 1.1)\) |
|                  | interpolation method of \(\text{Resize} = \text{BICUBIC}\) |
|                  | crop size = 224                                |
| Horizontal flip  | \(p = 0.5\)                                   |
| Vertical flip    | \(p = 0.5\)                                   |
| Translation      | \(p = 1.0\)                                   |
|                  | maximum absolute fraction = \((0.2, 0.2)\)     |
| Rotation         | \(p = 1.0\)                                   |
|                  | angle \(\in \{0, 90, 180, 270\}\)            |
| Grayscale        | \(p = 0.2\)                                   |
| Color jittering  | \(p = 0.8\)                                   |
|                  | maximum brightness adjustment = 0.4           |
|                  | maximum contrast adjustment = 0.4             |
|                  | maximum saturation adjustment = 0.4           |
|                  | maximum hue adjustment = 0.1                  |
| Gaussian blur    | \(p = 0.5\)                                   |
|                  | \(\sigma \in \{0.1, 2.0\}\)                 |

Table S2. Parameters used to generate image augmentations. \(p\) denotes the probability that the corresponding operation will be performed.

#### B.3. Image Augmentations for SSPL

As shown in Table S2, a total of 8 image augmentations are considered in our method. We follow HardWay \([1]\) to select and set the first two augmentations: cropping with \(224 \times 224\) resizing and horizontal flip. Then, we verify the effectiveness of other three spatial augmentations that are widely used in self-supervised visual representation learning \([3, 5]\), i.e., vertical flip, translation, and rotation. Additionally, since our work draws inspiration from SimSiam \([2]\), we also take into account its augmentation strategies: grayscale, color jittering, and Gaussian blur, while keeping their settings the same as SimSiam.

| Training set | SSPL (w/o PCM) | SSPL (w/ PCM) |
|--------------|----------------|---------------|
|              | \(lr_1\)      | \(lr_2\)      | \(lr_1\) | \(lr_2\) |
| SoundNet-Flickr | \(2 \times 10^{-3}\) | \(5 \times 10^{-4}\) | \(5 \times 10^{-5}\) | \(2 \times 10^{-5}\) |
| VGG-Sound    | \(1 \times 10^{-2}\) | \(5 \times 10^{-3}\) | \(5 \times 10^{-5}\) | \(2 \times 10^{-5}\) |

Table S1. Learning rate settings. \(lr_1\) denotes the learning rate for projection and prediction MLPs, \(lr_2\) for remaining model parts.
To balance between performance and time complexity, we set $T = 5$ during training.

D.3. Effect of False Negatives on Localization

As discussed in the main text, learning with false negatives can induce ambiguity in localization results. In this section, we give more examples to empirically illustrate this effect. As shown in Figure S2, when the false negatives are allowed to take part in contrastive learning, sounding objects are easily ignored in final localization maps (method A). Although learning with true positive and negative samples harvests accurate localization, it requires class label to direct negative sampling (method B). By contrast, our method is able to obtain consistent localization among different image-audio pairs, without using negatives and labels at all (method C).

D.4. Additional Qualitative Comparisons

In Figure S3, we illustrate more localization examples from Attention [9], HardWay [1], and our method SSPL on two standard benchmarks: SoundNet-Flickr and VGG-SS. Qualitative evaluation results show that our method can localize the full extent of sounding objects, especially for SSPL (w/ PCM) that yields more accurate localization by ignoring background noise.

References

[1] Honglie Chen, Weidi Xie, Triantafyllos Afouras, Arsha Nagrani, Andrea Vedaldi, and Andrew Zisserman. Localizing visual sounds the hard way. In CVPR, pages 16867–16876, 2021.
[2] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In CVPR, pages 15750–15758, 2021.
[3] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In ICLR, 2018.
Figure S2. **Visualisation of the effect of false negatives on sound localization.** A denotes the training strategy that uses both true and false negatives to perform contrastive learning; B indicates the method where false negatives do not take part in contrastive learning, but requiring class label to direct negative sampling; C corresponds to our self-supervised method that only explores audio-visual positive pairs during learning. Here the false negatives are other videos' sounds that belong to the same category as the positive one. The images marked with red rectangle illustrate ambiguous localization results of method A. Models are trained on MUSIC [12].
Figure S3. **Qualitative comparisons.** In each panel, the first column shows images accompanied with annotations, and remaining columns represent the predicted localization of sounding objects. Here the attention map or similarity map produced by different methods is visualized as the localization map. Note that for SoundNet-Flickr the bounding boxes are derived from multiple annotators.

[4] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (GELUs). *arXiv preprint arXiv:1606.08415*, 2016.

[5] Minsong Ki, Youngjung Uh, Junsuk Choe, and Hyeran Byun. Contrastive attention maps for self-supervised localization. In *ICCV*, pages 2803–2812, 2021.

[6] César Laurent, Gabriel Pereyra, Philémon Brakel, Ying Zhang, and Yoshua Bengio. Batch normalized recurrent neural networks. In *ICASSP*, pages 2657–2661, 2016.

[7] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

[8] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, pages 807–814, 2010.

[9] Arda Senocak, Tae-Hyun Oh, Junsik Kim, Ming-Hsuan Yang, and In So Kweon. Learning to localize sound source in visual scenes. In *CVPR*, pages 4358–4366, 2018.

[10] Michael W Spratling. A review of predictive coding algorithms. *Brain and Cognition*, 112:92–97, 2017.

[11] Haiguang Wen, Kuan Han, Junxing Shi, Yizhen Zhang, Eugenio Culurciello, and Zhongming Liu. Deep predictive coding network for object recognition. In *ICML*, pages 5266–5275, 2018.

[12] Hang Zhao, Chuang Gan, Andrew Rouditchenko, Carl Vondrick, Josh McDermott, and Antonio Torralba. The sound of pixels. In *ECCV*, pages 570–586, 2018.