Self-supervised Representation Learning with Relative Predictive Coding

Yao-Hung Hubert Tsai[1]*, **Martin Q. Ma**[1]*, **Muqiao Yang**[1],
Han Zhao[23], **Ruslan Salakhutdinov**[1], **Louis-Philippe Morency**[1]
[1]Carnegie Mellon University, [2]D.E. Shaw & Co., [3] University of Illinois at Urbana-Champaign

Abstract

This paper introduces Relative Predictive Coding (RPC), a new contrastive representation learning objective that maintains a good balance among training stability, minibatch size sensitivity, and downstream task performance. The key to the success of RPC is two-fold. First, RPC introduces the relative parameters to regularize the objective for boundedness and low variance. Second, RPC contains no logarithm and exponential score functions, which are the main cause of training instability in prior contrastive objectives. We empirically verify the effectiveness of RPC on benchmark vision and speech self-supervised learning tasks.

Unsupervised learning has drawn tremendous attention recently because it can extract rich representations without label supervision. Self-supervised learning, as a subset of unsupervised learning, learns representations by allowing the data to provide supervision [10]. Among its mainstream strategies, contrastive self-supervised learning has been successful in different representation learning tasks [28, 20, 41, 19]. The idea of contrastive self-supervised learning is to learn latent representations such that related instances (e.g., patches from the same image; defined as *positive* pairs) will have representations within close distance, while unrelated instances (e.g., patches from two different images; defined as *negative* pairs) will have distant representations [2].

Table 1: Different contrastive learning objectives, grouped by measurements of distribution divergence. P_{XY} represents the distribution of related samples (positively-paired), and $P_X P_Y$ represents the distribution of unrelated samples (negatively-paired). $f(x, y) \in \mathcal{F}$ for $\mathcal{F}$ being any class of functions $f : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$.

Objective	Good Training Stability	Lower Minibatch Size Sensitivity	Good Downstream Performance
relating to KL-divergence between P_{XY} and $P_X P_Y$: J_{DV} [11], J_{NWJ} [25], and J_{CPC} [28]			
$J_{\text{DV}}(X,Y) := \sup_{f\in\mathcal{F}} \mathbb{E}_{P_{XY}}[f(x,y)] - \log(\mathbb{E}_{P_X P_Y}[e^{f(x,y)}])$	✗	✓	✗
$J_{\text{NWJ}}(X,Y) := \sup_{f\in\mathcal{F}} \mathbb{E}_{P_{XY}}[f(x,y)] - \mathbb{E}_{P_X P_Y}[e^{f(x,y)-1}]$	✗	✓	✗
$J_{\text{CPC}}(X,Y) := \sup_{f\in\mathcal{F}} \mathbb{E}_{(x,y_1)\sim P_{XY},\{y_j\}_{j=2}^N\sim P_Y}\left[\log\left(e^{f(x,y_1)}/\frac{1}{N}\sum_{j=1}^N e^{f(x,y_j)}\right)\right]$	✓	✗	✓
relating to JS-divergence between P_{XY} and $P_X P_Y$: J_{JS} [27]			
$J_{\text{JS}}(X,Y) := \sup_{f\in\mathcal{F}} \mathbb{E}_{P_{XY}}[-\log(1+e^{-f(x,y)})] - \mathbb{E}_{P_X P_Y}[\log(1+e^{f(x,y)})]$	✓	✓	✗
relating to Wasserstein-divergence between P_{XY} and $P_X P_Y$: J_{WPC} [29], with $\mathcal{F}_\mathcal{L}$ denoting the space of 1-Lipschitz functions			
$J_{\text{WPC}}(X,Y) := \sup_{f\in\mathcal{F}_\mathcal{L}} \mathbb{E}_{(x,y_1)\sim P_{XY},\{y_j\}_{j=2}^N\sim P_Y}\left[\log\left(e^{f(x,y_1)}/\frac{1}{N}\sum_{j=1}^N e^{f(x,y_j)}\right)\right]$	✓	✓	✗
relating to χ^2-divergence between P_{XY} and $P_X P_Y$: J_{RPC} (ours)			
$J_{\text{RPC}}(X,Y) := \sup_{f\in\mathcal{F}} \mathbb{E}_{P_{XY}}[f(x,y)] - \alpha\mathbb{E}_{P_X P_Y}[f(x,y)] - \frac{\beta}{2}\mathbb{E}_{P_{XY}}\left[f^2(x,y)\right] - \frac{\gamma}{2}\mathbb{E}_{P_X P_Y}\left[f^2(x,y)\right]$	✓	✓	✓

Prior work has formulated the contrastive learning objectives as maximizing the divergence between the distributions of related and unrelated instances. In this regard, different divergence measurement often leads to different loss function design. To illustrate this, we use an uppercase letter to denote a random variable (e.g., X), a lower case letter to denote the outcome of this random variable (e.g., x),

*Indicates equal contribution.

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

and a calligraphy letter to denote the sample space of this random variable (e.g., $\mathcal{X}$). In a contrastive learning setting with related and unrelated samples, we define related (or positively-paired) samples $(x, y) \sim P_{XY}$ with P_{XY} being the joint distribution of $X \times Y$, and unrelated (negatively-paired) samples $(x, y) \sim P_X P_Y$ with $P_X P_Y$ being the product of marginal distributions over $X \times Y$. Then, the category of modeling the Kullback-Leibler divergence $D_{\mathrm{KL}}(P_{XY} \,\|\, P_X P_Y)$ includes the Donsker-Varadhan objective (J_{DV} [11, 4]), the Nguyen-Wainright-Jordan objective (J_{NWJ} [25, 4]), and the Contrastive Predictive Coding (J_{CPC} [28]). The instance of modeling the Jensen-Shannon divergence $D_{\mathrm{JS}}(P_{XY} \,\|\, P_X P_Y)$ is the Jensen-Shannon f-GAN objective $\big(J_{\mathrm{JS}}$ [27, 15]$\big)$. The instance of modeling the Wasserstein divergence $D_{\mathrm{Wass}}(P_{XY} \,\|\, P_X P_Y)$ is the Wasserstein Predictive Coding $\big(J_{\mathrm{WPC}}$ [29]$\big)$. These objectives maximize the distribution divergence between P_{XY} and $P_X P_Y$, where we summarize them in Table 1. Prior work [2, 36] theoretically show that these self-supervised contrastive learning objectives leads to the representations that can work well on downstream tasks.

After discussing prior methods, we point out three challenges in their practical deployments: training stability, sensitivity to minibatch training size, and downstream task performance. These three challenges can hardly be handled well at the same time, and we highlight the conclusions in Table 1. ***Training Stability:*** The training stability highly relates to the variance of the objectives, where Song *et al.* [34] shows that J_{DV} and J_{NWJ} exhibit inevitable high variance due to their inclusion of exponential function. As pointed out by Tsai *et al.* [37], J_{CPC}, J_{WPC}, and J_{JS} have better training stability because J_{CPC} and J_{WPC} can be realized as a multi-class classification task and J_{JS} can be realized as a binary classification task. ***Sensitivity to minibatch training size:*** Among all the prior contrastive representation learning methods, J_{CPC} is known to be sensitive to the minibatch training size [29]. Taking a closer look at Table 1, J_{CPC} deploys an instance selection such that y_1 should be selected from $\{y_1, y_2, \cdots, y_N\}$, with $(x, y_1) \sim P_{XY}$, $(x, y_{j>1}) \sim P_X P_Y$ with N being the minibatch size. Previous work [31, 34, 6, 5] showed that a large N results in a more challenging instance selection and forces J_{CPC} to have a better contrastiveness of y_1 (related instance for x) against $\{y_j\}_{j=2}^N$ (unrelated instance for x). J_{DV}, J_{NWJ}, and J_{JS} do not consider the instance selection, and J_{WPC} reduces the minibatch training size sensitivity by enforcing 1-Lipschitz constraint. ***Downstream Task Performance:*** The downstream task performance is what we care the most among all the three challenges. J_{CPC} has been the most popular objective as it manifests superior performance over the other alternatives [38, 37, 36]. We note that although J_{WPC} shows better performance on Omniglot [22] and CelebA [24] datasets, we empirically find it not generalizing well to CIFAR-10/-100 [21] and ImageNet [33].

1 Relative Predictive Coding

In this paper, we present Relative Predictive Coding (RPC), which achieves a good balance among the three challenges mentioned above:

$$\mathrm{J}_{\mathrm{RPC}}(X, Y) := \sup_{f \in \mathcal{F}} \mathbb{E}_{P_{XY}}[f(x,y)] - \alpha \mathbb{E}_{P_X P_Y}[f(x,y)] - \tfrac{\beta}{2}\mathbb{E}_{P_{XY}}\left[f^2(x,y)\right] - \tfrac{\gamma}{2}\mathbb{E}_{P_X P_Y}\left[f^2(x,y)\right], \quad (1)$$

where $\alpha > 0$, $\beta > 0$, $\gamma > 0$ are hyper-parameters and we define them as *relative parameters*. Intuitively, J_{RPC} contains no logarithm or exponential, potentially preventing unstable training due to numerical issues. Now, we discuss the roles of α, β, γ. At a first glance, α acts to discourage the scores of P_{XY} and $P_X P_Y$ from being close, and β/γ acts as a ℓ_2 regularization coefficient to stop f from becoming large. For a deeper analysis, the relative parameters act to regularize our objective for boundedness and low variance. To show this claim, we first present the following lemma:

Lemma 1 (Optimal Solution for J_{RPC}) *Let $r(x, y) = \frac{p(x,y)}{p(x)p(y)}$ be the density ratio. J_{RPC} has the optimal solution $f^*(x, y) = \frac{r(x,y)-\alpha}{\beta\, r(x,y)+\gamma} := r_{\alpha,\beta,\gamma}(x, y)$ with $-\frac{\alpha}{\gamma} \leq r_{\alpha,\beta,\gamma} \leq \frac{1}{\beta}$.*

Lemma 1 suggests that J_{RPC} achieves its supreme value at the ratio $r_{\alpha,\beta,\gamma}(x, y)$ indexed by the relative parameters α, β, γ (i.e., we term $r_{\alpha,\beta,\gamma}(x, y)$ as the relative density ratio). We note that $r_{\alpha,\beta,\gamma}(x, y)$ is an increasing function w.r.t. $r(x, y)$ and is nicely bounded even when $r(x, y)$ is large. We will now show that the bounded $r_{\alpha,\beta,\gamma}$ suggests the empirical estimation of J_{RPC} has boundeness and low variance. In particular, let $\{x_i, y_i\}_{i=1}^n$ be n samples drawn uniformly at random from P_{XY} and $\{x'_j, y'_j\}_{j=1}^m$ be m samples drawn uniformly at random from $P_X P_Y$. Then, we use neural networks to empirically estimate J_{RPC} as $\hat{J}^{m,n}_{\mathrm{RPC}}$:

Definition 1 ($\hat{J}^{m,n}_{\mathrm{RPC}}$, empirical estimation of J_{RPC}) *We parametrize f via a family of neural networks $\mathcal{F}_\Theta := \{f_\theta : \theta \in \Theta \subseteq \mathbb{R}^d\}$ where $d \in \mathbb{N}$ and Θ is compact. Then, $\hat{J}^{m,n}_{\mathrm{RPC}} = \sup_{f_\theta \in \mathcal{F}_\Theta} \frac{1}{n}\sum_{i=1}^n f_\theta(x_i, y_i) - \frac{1}{m}\sum_{j=1}^m \alpha f_\theta(x'_j, y'_j) - \frac{1}{n}\sum_{i=1}^n \frac{\beta}{2} f^2_\theta(x_i, y_i) - \frac{1}{m}\sum_{j=1}^m \frac{\gamma}{2} f^2_\theta(x'_j, y'_j)$.*

Proposition 1 (Boundedness and variance f $\hat{J}^{m,n}_{\mathrm{RPC}}$, informal) $0 \leq J_{\mathrm{RPC}} \leq \frac{1}{2\beta} + \frac{\alpha^2}{2\gamma}$. *With probability at least $1-\delta$, $|J_{\mathrm{RPC}} - \hat{J}^{m,n}_{\mathrm{RPC}}| = O(\sqrt{\frac{d+\log(1/\delta)}{n'}})$, where $n' = \min\{n, m\}$. Also, There exist universal constants c_1 and c_2 that depend only on α, β, γ, such that $\mathrm{Var}[\hat{J}^{m,n}_{\mathrm{RPC}}] = O(\frac{c_1}{n} + \frac{c_2}{m})$.*

From the proposition, when m and n are large, i.e., the sample sizes are large, $\hat{J}^{m,n}_{\mathrm{RPC}}$ is bounded, and its variance vanishes to 0. First, the boundedness of $\hat{J}^{m,n}_{\mathrm{RPC}}$ suggests $\hat{J}^{m,n}_{\mathrm{RPC}}$ will not grow to extremely large or small values. Prior contrastive learning objectives with good training stability (e.g., J_{CPC}/J_{JS}/J_{WPC}) also have the boundedness of their objective values. For instance, the empirical estimation of J_{CPC} is less than $\log N$ (where N is the batch size) [31]. Second, the upper bound of the variance implies the training of $\hat{J}^{m,n}_{\mathrm{RPC}}$ can be stable, and in practice we observe a much smaller value than the stated upper bound. On the contrary, the empirical estimations of J_{DV} and J_{NWJ} exhibit inevitable variances that grow exponentially with the true $D_{\mathrm{KL}}(P_{XY}\|P_X P_Y)$ [34]. Lastly, we use empirical experiments to support the claim of low minibatch size sensitivity and good performance.

2 Experiments

We conduct benchmark self-supervised representation learning tasks spanning visual object classification and speech recognition. They are designed to discuss the three challenges of the contrastive representation learning objectives: downstream task performance, training stability, and mini-batch size sensitivity. We also provide an ablation study on the choices of the relative parameters in J_{RPC}. For the visual objective classification, we consider CIFAR-10/-100 [21], STL-10 [9], and ImageNet [33]. For the speech recognition, we consider LibriSpeech-100h [30] dataset, which contains 100 hours of 16kHz English speech from 251 speakers with 41 types of phonemes. For the vision experiments, we follow the setup from SimCLRv2 [7], which considers visual object recognition as its downstream task. For the speech experiments, we follow the setup from prior work [28, 32], which consider phoneme classification and speaker identification as the downstream tasks. We fairly compare J_{RPC} with other contrastive learning objectives. Particularly, across different objectives, we fix the network, learning rate, optimizer, and batch size by using the default configurations suggested by the original implementations from prior work [7, 32]. The only difference will be the objective itself. We defer experimental details, including choices of relative parameters, in the Appendix. In general, we found that a small β and large γ in RPC would result in the best performance across different tasks. We would like to point out that we do not apply hidden embedding normalization as in prior work [35, 6, 7]. Specifically, in Table 1, prior work designs $f(x, y) = \frac{x^\top y}{\|x\|\cdot\|y\|\cdot\tau}$ with $\|x\|$ ($\|y\|$) representing the ℓ_2-norm of x (y) and τ is a hyper-parameter denoting the temperature. While J_{RPC} considers $f(x, y) = x^\top y/\tau$ without normalizing x and y. This normalization leads to better training stability and performance boost for prior method, while we find it not required for our method.

Downstream Task Performances on Vision and Speech For the downstream task performance in the vision domain, Table 2 shows that the proposed J_{RPC} outperforms other objectives on all datasets. Using J_{RPC} on the largest network (last row in Table 2, where we use a ResNet [14] with depth of 152, channel width of 2 and selective kernels [23]), the performance jumps from 77.80% of J_{CPC} to 78.40% of J_{RPC}. Regarding speech representation learning, the downstream performance for phoneme and speaker classification are shown in Table 3. J_{RPC} improves the phoneme classification results with around 4.5 percent and the speaker classification results with around 0.3 percent. We empirically observe that small βs and large γs lead to the best performance.

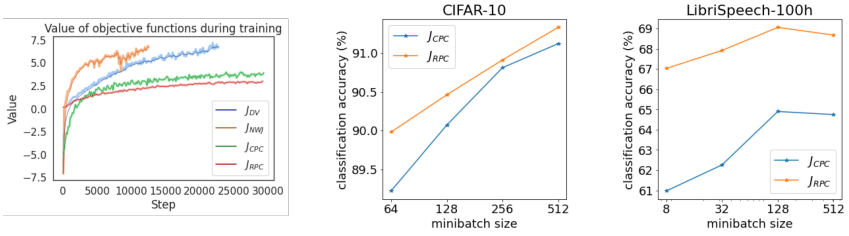
Training Stability We provide empirical training stability comparisons on J_{DV}, J_{NWJ}, J_{CPC} and J_{RPC} by plotting the values of the objectives as the training step increases. We apply J_{DV}, J_{NWJ}, J_{CPC} and J_{RPC} to the SimCLRv2 framework and train on the CIFAR-10 dataset. From our experiments, J_{DV} and J_{NWJ} soon explode to NaN value and disrupt training (shown as early

Table 2: Top-1 accuracy (%) for visual object recognition results. J_{DV} and J_{NWJ} are not reported on ImageNet due to numerical instability. ResNet depth, width and Selective Kernel (SK) configuration for each setting are provided in ResNet depth+width+SK column. A slight drop of J_{CPC} performance compared to prior work [7] is because we only train for 100 epochs rather than 800 due to the fact that we train on cloud TPU and running 800 epochs uninterruptedly is very expensive. Also, we did not employ a memory buffer [13] that are used to store negative samples. We provide the results from fully supervised models as a comparison [6, 7].

Dataset	ResNet Depth+Width+SK	Self-supervised						Supervised
		J_{DV}	J_{NWJ}	J_{JS}	J_{WPC}	J_{CPC}	J_{RPC}	
CIFAR-10	18 + 1× + No SK	91.10	90.54	83.55	80.02	91.12	**91.46**	93.12
CIFAR-10	50 + 1× + No SK	92.23	92.67	87.34	85.93	93.42	**93.57**	95.70
CIFAR-100	18 + 1× + No SK	77.10	77.27	74.02	72.16	77.36	**77.98**	79.11
CIFAR-100	50 + 1× + No SK	79.02	78.52	75.31	73.23	79.31	**79.89**	81.20
STL-10	50 + 1× + No SK	82.25	81.17	79.07	76.50	83.40	**84.10**	71.40
ImageNet	50 + 1× + SK	-	-	66.21	62.10	73.48	**74.43**	78.50
ImageNet	152 + 2× + SK	-	-	71.12	69.51	77.80	**78.40**	80.40

Table 3: Accuracy (%) for LibriSpeech-100h phoneme and speaker classification results. We also provide the results from fully supervised model as a comparison [28].

Task Name	Self-supervised				Supervised
	J_{CPC}	J_{DV}	J_{NWJ}	J_{RPC}	
Phoneme classification	64.6	61.27	62.09	**69.06**	74.6
Speaker classification	97.4	95.36	95.89	**97.68**	98.5



(a) (b) (c)

Figure 1: (a) Empirical values of J_{DV}, J_{NWJ}, J_{CPC} and J_{RPC} performing visual objective recognition on CIFAR-10. J_{DV} and J_{NWJ} soon explode to NaN values and stop the training (shown as early stopping in the figure), while J_{CPC} and J_{RPC} are more stable. Performance comparison of J_{CPC} and J_{RPC} on (b) CIFAR-10 and (c) LibriSpeech-100h with different minibatch sizes, showing that the performance of J_{RPC} is less sensitive to minibatch size change compared to J_{CPC}.

stopping in Figure 1a; extremely large values are not plotted due to scale constraints). On the other hand, J_{RPC} and J_{CPC} has low variance, and both enjoy stable training. As a result, performances using the representations learned from unstable J_{DV} and J_{NWJ} suffer in downstream task, while representations learned by J_{RPC} and J_{CPC} work much better.

Minibatch Size Sensitivity We then provide the analysis on the effect of minibatch size on J_{RPC} and J_{CPC}, since J_{CPC} is known to be sensitive to minibatch size [31]. We train SimCLRv2 [7] on CIFAR-10 and the model from prior work [32] on LibriSpeech-100h using J_{RPC} and J_{CPC} with different minibatch sizes. From Figure 1b and 1c, we can observe that both J_{RPC} and J_{CPC} achieve their optimal performance at a large minibatch size. However, when the minibatch size decreases, the performance of J_{CPC} shows slightly higher sensitivity and suffers a little more when the number of minibatch samples is small. Overall, the result suggests that the proposed method might be less sensitive to the change of minibatch size compared to J_{CPC} given the same training settings.

Effect of Relative Parameters We study the effect of different combinations of relative parameters in J_{RPC} by showing downstream performance on visual object recognition. We train SimCLRv2 on CIFAR-10 with different combination of α, β and γ in J_{RPC}. We find that in Equation 1, a small β (in a scale of 10^{-3}) and a large γ (close to 1.0) are crucial for training a high-performing encoder which extracts useful information for downstream tasks. $\alpha = 0$ influences less to the downstream performance unless being equal to 0. The major reason is that in the proposed J_{RPC}, β and γ serve as regularization coefficients. We argue that α is less effective because it only regularizes a linear function of the negative terms, thus the gradients during back-propagation will be less dominant than the gradients from the quadratic term regularized by β. We report the relative parameters that achieve the best downstream performance in the last row. Next, we report the empirical variance $\mathrm{Var}(\hat{J}^{m,n}_{\mathrm{RPC}})$ using different sets of relative parameters. We observe that a small β is the most significant factor for a small empirical variance; γ also hugely impacts the empirical variance, but not as much as β. Effect of α is relatively small. Although some empirical variances seem to be large, they are still significantly smaller than the scale of variance of J_{DV} and J_{NWJ}, which grows to the scale of 10^{23} and eventually explodes under the same training settings.

Table 4: Accuracy comparison under different relative parameters for visual object recognition classification results, showing that a small β and a large γ are crucial for better performance.

α	β	γ	Acc (%)	Var ($\hat{J}^{m,n}_{\mathrm{RPC}}$)
0.001	0.001	0.001	87.00	5120.370
1.000	0.001	0.001	87.50	6096.360
0.001	1.000	0.001	76.90	0.002
0.001	0.001	1.000	91.20	46.500
0.001	1.000	1.000	84.02	0.003
1.000	0.001	1.000	91.30	72.540
1.000	1.000	0.001	76.45	0.002
1.000	1.000	1.000	85.23	0.003
1.000	0.050	1.000	91.46	1.620

3 Conclusion

In this work, we present RPC, the Relative Predictive Coding, that achieves a good balance among the three challenges when modeling a contrastive learning objective: training stability, sensitivity to minibatch size, and downstream task performance. We believe this work brings an appealing option for training self-supervised models and inspires future work to design objectives for balancing the aforementioned three challenges. In the future, we are interested in applying RPC in other application domains and developing more principled approaches for better representation learning.

Acknowledgments

This work was supported in part by the DARPA grants FA875018C0150 HR00111990016, NSF IIS1763562, NSF Awards #1750439 #1722822, National Institutes of Health, and Apple. We would also like to acknowledge NVIDIA's GPU support. We would like to thank Google Tensorflow Research Cloud program for their very generous TPU support.

References

[1] Martin Anthony and Peter L Bartlett. *Neural network learning: Theoretical foundations*. cambridge university press, 2009.

[2] Sanjeev Arora, Hrishikesh Khandeparkar, Mikhail Khodak, Orestis Plevrakis, and Nikunj Saunshi. A theoretical analysis of contrastive unsupervised representation learning. *arXiv preprint arXiv:1902.09229*, 2019.

[3] Peter L Bartlett. The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network. *IEEE transactions on Information Theory*, 44(2):525–536, 1998.

[4] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeswar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and R Devon Hjelm. Mine: mutual information neural estimation. *arXiv preprint arXiv:1801.04062*, 2018.

[5] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *arXiv preprint arXiv:2006.09882*, 2020.

[6] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

[7] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey Hinton. Big self-supervised models are strong semi-supervised learners. *arXiv preprint arXiv:2006.10029*, 2020.

[8] Ching-Yao Chuang, Joshua Robinson, Lin Yen-Chen, Antonio Torralba, and Stefanie Jegelka. Debiased contrastive learning. *arXiv preprint arXiv:2007.00224*, 2020.

[9] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223, 2011.

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[11] Monroe D Donsker and SR Srinivasa Varadhan. Asymptotic evaluation of certain markov process expectations for large time, i. *Communications on Pure and Applied Mathematics*, 28(1):1–47, 1975.

[12] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in neural information processing systems*, pages 5767–5777, 2017.

[13] 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*, pages 9729–9738, 2020.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[15] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*, 2018.

[16] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[17] K Hornik, M Stinchcombe, and H White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.

[18] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[19] Thomas Kipf, Elise van der Pol, and Max Welling. Contrastive learning of structured world models. *arXiv preprint arXiv:1911.12247*, 2019.

[20] Lingpeng Kong, Cyprien de Masson d'Autume, Wang Ling, Lei Yu, Zihang Dai, and Dani Yogatama. A mutual information maximization perspective of language representation learning. *arXiv preprint arXiv:1910.08350*, 2019.

[21] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[22] Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

[23] Xiang Li, Wenhai Wang, Xiaolin Hu, and Jian Yang. Selective kernel networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 510–519, 2019.

[24] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision*, pages 3730–3738, 2015.

[25] XuanLong Nguyen, Martin J Wainwright, and Michael I Jordan. Estimating divergence functionals and the likelihood ratio by convex risk minimization. *IEEE Transactions on Information Theory*, 56(11):5847–5861, 2010.

[26] Frank Nielsen and Richard Nock. On the chi square and higher-order chi distances for approximating f-divergences. *IEEE Signal Processing Letters*, 21(1):10–13, 2013.

[27] Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. In *Advances in neural information processing systems*, pages 271–279, 2016.

[28] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[29] Sherjil Ozair, Corey Lynch, Yoshua Bengio, Aaron Van den Oord, Sergey Levine, and Pierre Sermanet. Wasserstein dependency measure for representation learning. In *Advances in Neural Information Processing Systems*, pages 15604–15614, 2019.

[30] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210. IEEE, 2015.

[31] Ben Poole, Sherjil Ozair, Aaron van den Oord, Alexander A Alemi, and George Tucker. On variational bounds of mutual information. *arXiv preprint arXiv:1905.06922*, 2019.

[32] Morgane Rivière, Armand Joulin, Pierre-Emmanuel Mazaré, and Emmanuel Dupoux. Unsupervised pretraining transfers well across languages. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7414–7418. IEEE, 2020.

[33] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[34] Jiaming Song and Stefano Ermon. Understanding the limitations of variational mutual information estimators. *arXiv preprint arXiv:1910.06222*, 2019.

[35] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. *arXiv preprint arXiv:1906.05849*, 2019.

[36] Yao-Hung Hubert Tsai, Yue Wu, Ruslan Salakhutdinov, and Louis-Philippe Morency. Self-supervised learning from a multi-view perspective. *arXiv preprint arXiv:2006.05576*, 2020.

[37] Yao-Hung Hubert Tsai, Han Zhao, Makoto Yamada, Louis-Philippe Morency, and Ruslan Salakhutdinov. Neural methods for point-wise dependency estimation. *arXiv preprint arXiv:2006.05553*, 2020.

[38] Michael Tschannen, Josip Djolonga, Paul K Rubenstein, Sylvain Gelly, and Mario Lucic. On mutual information maximization for representation learning. *arXiv preprint arXiv:1907.13625*, 2019.

[39] Aad W Van der Vaart. *Asymptotic statistics*, volume 3. Cambridge university press, 2000.

[40] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[41] Petar Velickovic, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. In *ICLR (Poster)*, 2019.

[42] Yang You, Igor Gitman, and Boris Ginsburg. Large batch training of convolutional networks. *arXiv preprint arXiv:1708.03888*, 2017.

A Proof of Lemma 1 in the Main Text

Lemma 2 (Optimal Solution for J_{RPC}, restating Lemma 1 in the main text) *Let*

$$J_{\mathrm{RPC}}(X,Y) := \sup_{f\in\mathcal{F}} \mathbb{E}_{P_{XY}}[f(x,y)] - \alpha \mathbb{E}_{P_X P_Y}[f(x,y)] - \frac{\beta}{2}\mathbb{E}_{P_{XY}}\left[f^2(x,y)\right] - \frac{\gamma}{2}\mathbb{E}_{P_X P_Y}\left[f^2(x,y)\right]$$

and $r(x,y) = \frac{p(x,y)}{p(x)p(y)}$ *be the density ratio.* J_{RPC} *has the optimal solution*

$$f^*(x,y) = \frac{r(x,y)-\alpha}{\beta\, r(x,y)+\gamma} := r_{\alpha,\beta,\gamma}(x,y) \text{ with } -\frac{\alpha}{\gamma} \le r_{\alpha,\beta,\gamma} \le \frac{1}{\beta}.$$

Proof: The second-order functional derivative of the objective is

$$-\beta dP_{X,Y} - \gamma dP_X P_Y,$$

which is always negative. The negative second-order functional derivative implies the objective has a supreme value. Then, take the first-order functional derivative $\frac{\partial J_{\mathrm{RPC}}}{\partial m}$ and set it to zero:

$$dP_{X,Y} - \alpha \cdot dP_X P_Y - \beta \cdot f(x,y) \cdot dP_{X,Y} - \gamma \cdot f(x,y) \cdot dP_X P_Y = 0.$$

We then get

$$f^*(x,y) = \frac{dP_{X,Y} - \alpha \cdot dP_X P_Y}{\beta \cdot dP_{X,Y} + \gamma \cdot dP_X P_Y} = \frac{p(x,y) - \alpha p(x)p(y)}{\beta p(x,y) + \gamma p(x)p(y)} = \frac{r(x,y)-\alpha}{\beta r(x,y)+\gamma}.$$

Since $0 \le r(x,y) \le \infty$, we have $-\frac{\alpha}{\gamma} \le \frac{r(x,y)-\alpha}{\beta r(x,y)+\gamma} \le \frac{1}{\beta}$. Hence,

$$\forall \beta \neq 0, \gamma \neq 0, f^*(x,y) := r_{\alpha,\beta,\gamma}(x,y) \text{ with } -\frac{\alpha}{\gamma} \le r_{\alpha,\beta,\gamma} \le \frac{1}{\beta}.$$

□

B Relation between J_{RPC} and D_{χ^2}

In this subsection, we aim to show the following: 1) $D_{\chi^2}(P_{XY} \,\|\, P_X P_Y) = \mathbb{E}_{P_X P_Y}[r^2(x,y)] - 1$; and 2) $J_{\mathrm{RPC}}(X,Y) = \frac{\beta+\gamma}{2}\mathbb{E}_{P'}[r^2_{\alpha,\beta,\gamma}(x,y)]$ by having $P' = \frac{\beta}{\beta+\gamma}P_{XY} + \frac{\gamma}{\beta+\gamma}P_X P_Y$ as the mixture distribution of P_{XY} and $P_X P_Y$.

Lemma 3 $D_{\chi^2}(P_{XY} \,\|\, P_X P_Y) = \mathbb{E}_{P_X P_Y}[r^2(x,y)] - 1$

Proof: By definition [26],

$$\begin{aligned} D_{\chi^2}(P_{XY} \,\|\, P_X P_Y) &= \int \frac{\left(dP_{XY}\right)^2}{dP_X P_Y} - 1 = \int \left(\frac{dP_{XY}}{dP_X P_Y}\right)^2 dP_X P_Y - 1 \\ &= \int \left(\frac{p(x,y)}{p(x)p(y)}\right)^2 dP_X P_Y - 1 = \int r^2(x,y) dP_X P_Y - 1 \\ &= \mathbb{E}_{P_X P_Y}[r^2(x,y)] - 1. \end{aligned}$$

□

Lemma 4 *Defining* $P' = \frac{\beta}{\beta+\gamma}P_{XY} + \frac{\gamma}{\beta+\gamma}P_X P_Y$ *as a mixture distribution of* P_{XY} *and* $P_X P_Y$, $J_{\mathrm{RPC}}(X,Y) = \frac{\beta+\gamma}{2}\mathbb{E}_{P'}[r^2_{\alpha,\beta,\gamma}(x,y)]$.

Proof: Plug in the optimal solution $f^*(x,y) = \frac{dP_{X,Y} - \alpha \cdot dP_X P_Y}{\beta \cdot dP_{X,Y} + \gamma \cdot dP_X P_Y}$ (see Lemma 2) into J_{RPC}:

$$\begin{aligned}
J_{\mathrm{RPC}} &= \mathbb{E}_{P_{XY}}[f^*(x,y)] - \alpha \mathbb{E}_{P_X P_Y}[f^*(x,y)] - \frac{\beta}{2}\mathbb{E}_{P_{XY}}\left[f^{*2}(x,y)\right] - \frac{\gamma}{2}\mathbb{E}_{P_X P_Y}\left[f^{*2}(x,y)\right] \\
&= \int f^*(x,y) \cdot \Big(dP_{XY} - \alpha \cdot dP_X P_Y\Big) - \frac{1}{2} f^{*2}(x,y) \cdot \Big(\beta \cdot dP_{XY} + \gamma \cdot dP_X P_Y\Big) \\
&= \int \frac{dP_{X,Y} - \alpha \cdot dP_X P_Y}{\beta \cdot dP_{X,Y} + \gamma \cdot dP_X P_Y}\Big(dP_{XY} - \alpha \cdot dP_X P_Y\Big) - \frac{1}{2}\Big(\frac{dP_{X,Y} - \alpha \cdot dP_X P_Y}{\beta \cdot dP_{X,Y} + \gamma \cdot dP_X P_Y}\Big)^2\Big(\beta \cdot dP_{XY} + \gamma \cdot dP_X P_Y\Big) \\
&= \frac{1}{2}\int \Big(\frac{dP_{X,Y} - \alpha \cdot dP_X P_Y}{\beta \cdot dP_{X,Y} + \gamma \cdot dP_X P_Y}\Big)^2\Big(\beta \cdot dP_{XY} + \gamma \cdot dP_X P_Y\Big) \\
&= \frac{\beta+\gamma}{2}\int \Big(\frac{dP_{X,Y} - \alpha \cdot dP_X P_Y}{\beta \cdot dP_{X,Y} + \gamma \cdot dP_X P_Y}\Big)^2\Big(\frac{\beta}{\beta+\gamma} \cdot dP_{XY} + \frac{\gamma}{\beta+\gamma} \cdot dP_X P_Y\Big).
\end{aligned}$$

Since we define $r_{\alpha,\beta,\gamma} = \frac{dP_{X,Y} - \alpha \cdot dP_X P_Y}{\beta \cdot dP_{X,Y} + \gamma \cdot dP_X P_Y}$ and $P' = \frac{\beta}{\beta+\gamma}P_{XY} + \frac{\gamma}{\beta+\gamma}P_X P_Y$,

$$J_{\mathrm{RPC}} = \frac{\beta+\gamma}{2}\mathbb{E}_{P'}[r^2_{\alpha,\beta,\gamma}(x,y)].$$

□

C Proof of Proposition 1 in the Main Text

The proof contains two parts: showing $0 \le J_{\mathrm{RPC}} \le \frac{1}{2\beta} + \frac{\alpha^2}{2\gamma}$ (see Section C.1) and $\hat{J}^{m,n}_{\mathrm{RPC}}$ is a consistent estimator for J_{RPC} (see Section C.2).

C.1 Boundness of J_{RPC}

Lemma 5 (Boundness of J_{RPC}) $0 \le J_{\mathrm{RPC}} \le \frac{1}{2\beta} + \frac{\alpha^2}{2\gamma}$

Proof: Lemma 4 suggests $J_{\mathrm{RPC}}(X,Y) = \frac{\beta+\gamma}{2}\mathbb{E}_{P'}[r^2_{\alpha,\beta,\gamma}(x,y)]$ with $P' = \frac{\beta}{\beta+\gamma}P_{XY} + \frac{\gamma}{\beta+\gamma}P_X P_Y$ as the mixture distribution of P_{XY} and $P_X P_Y$. Hence, it is obvious $J_{\mathrm{RPC}}(X,Y) \ge 0$.

We leverage the intermediate results in the proof of Lemma 4:

$$\begin{aligned}
J_{\mathrm{RPC}}(X,Y) &= \frac{1}{2}\int \Big(\frac{dP_{X,Y} - \alpha \cdot dP_X P_Y}{\beta \cdot dP_{X,Y} + \gamma \cdot dP_X P_Y}\Big)^2\Big(\beta \cdot dP_{XY} + \gamma \cdot dP_X P_Y\Big) \\
&= \frac{1}{2}\int dP_{X,Y}\Big(\frac{dP_{X,Y} - \alpha \cdot dP_X P_Y}{\beta \cdot dP_{X,Y} + \gamma \cdot dP_X P_Y}\Big) - \frac{\alpha}{2}\int dP_X P_Y\Big(\frac{dP_{X,Y} - \alpha \cdot dP_X P_Y}{\beta \cdot dP_{X,Y} + \gamma \cdot dP_X P_Y}\Big) \\
&= \frac{1}{2}\mathbb{E}_{P_{XY}}[r_{\alpha,\beta,\gamma}(x,y)] - \frac{\alpha}{2}\mathbb{E}_{P_{XY}}[r_{\alpha,\beta,\gamma}(x,y)].
\end{aligned}$$

Since $-\frac{\alpha}{\gamma} \le r_{\alpha,\beta,\gamma} \le \frac{1}{\beta}$, $J_{\mathrm{RPC}}(X,Y) \le \frac{1}{2\beta} + \frac{\alpha^2}{2\gamma}$. □

C.2 Consistency

We first recall the definition of the estimation of J_{RPC}:

Definition 2 ($\hat{J}^{m,n}_{\mathrm{RPC}}$, empirical estimation of J_{RPC}, restating Definition 1 in the main text) *We parametrize f via a family of neural networks $\mathcal{F}_\Theta := \{f_\theta : \theta \in \Theta \subseteq \mathbb{R}^d\}$ where $d \in \mathbb{N}$ and Θ is compact. Let $\{x_i, y_i\}_{i=1}^n$ be n samples drawn uniformly at random from P_{XY} and $\{x'_j, y'_j\}_{j=1}^m$ be m samples drawn uniformly at random from $P_X P_Y$. Then,*

$$\hat{J}^{m,n}_{\mathrm{RPC}} = \sup_{f_\theta \in \mathcal{F}_\Theta} \frac{1}{n}\sum_{i=1}^n f_\theta(x_i, y_i) - \frac{1}{m}\sum_{j=1}^m \alpha f_\theta(x'_j, y'_j) - \frac{1}{n}\sum_{i=1}^n \frac{\beta}{2} f^2_\theta(x_i, y_i) - \frac{1}{m}\sum_{j=1}^m \frac{\gamma}{2} f^2_\theta(x'_j, y'_j).$$

Our goal is to show that $\hat{J}^{m,n}_{\mathrm{RPC}}$ is a consistent estimator for J_{RPC}. We begin with the following definition:

$$\hat{J}^{m,n}_{\mathrm{RPC},\theta} := \frac{1}{n}\sum_{i=1}^{n} f_\theta(x_i, y_i) - \frac{1}{m}\sum_{j=1}^{m} \alpha f_\theta(x'_j, y'_j) - \frac{1}{n}\sum_{i=1}^{n} \frac{\beta}{2} f^2_\theta(x_i, y_i) - \frac{1}{m}\sum_{j=1}^{m} \frac{\gamma}{2} f^2_\theta(x'_j, y'_j) \quad (2)$$

and

$$\mathbb{E}\Big[\hat{J}_{\mathrm{RPC},\theta}\Big] := \mathbb{E}_{P_{XY}}[f_\theta(x,y)] - \alpha\mathbb{E}_{P_X P_Y}[f_\theta(x,y)] - \frac{\beta}{2}\mathbb{E}_{P_{XY}}[f^2_\theta(x,y)] - \frac{\gamma}{2}\mathbb{E}_{P_X P_Y}[f^2_\theta(x,y)]. \quad (3)$$

Then, we follow the steps:

- The first part is about estimation. We show that, with high probability, $\hat{J}^{m,n}_{\mathrm{RPC},\theta}$ is close to $\mathbb{E}\Big[\hat{J}_{\mathrm{RPC},\theta}\Big]$, for any given θ.
- The second part is about approximation. We will apply the universal approximation lemma of neural networks [17] to show that there exists a network θ^* such that $\mathbb{E}\Big[\hat{J}_{\mathrm{RPC},\theta^*}\Big]$ is close to J_{RPC}.

Part I - Estimation: With high probability, $\hat{J}^{m,n}_{\mathrm{RPC},\theta}$ is close to $\mathbb{E}\Big[\hat{J}_{\mathrm{RPC},\theta}\Big]$, for any given θ. Throughout the analysis on the uniform convergence, we need the assumptions on the boundness and smoothness of the function f_θ. Since we show the optimal function f is bounded in J_{RPC}, we can use the same bounded values for f_θ without losing too much precision. The smoothness of the function suggests that the output of the network should only change slightly when only slightly perturbing the parameters. Specifically, the two assumptions are as follows:

Assumption 1 (boundness of f_θ) *There exist universal constants such that $\forall f_\theta \in \mathcal{F}_\Theta$, $C_L \le f_\theta \le C_U$. For notations simplicity, we let $M = C_U - C_L$ be the range of f_θ and $U = \max\{|C_U|, |C_L|\}$ be the maximal absolute value of f_θ. In the paper, we can choose to constrain that $C_L = -\frac{\alpha}{\gamma}$ and $C_U = \frac{1}{\beta}$ since the optimal function f^* has $-\frac{\alpha}{\gamma} \le f^* \le \frac{1}{\beta}$.*

Assumption 2 (smoothness of f_θ) *There exists constant $\rho > 0$ such that $\forall (x,y) \in (\mathcal{X} \times \mathcal{Y})$ and $\theta_1, \theta_2 \in \Theta$, $|f_{\theta_1}(x,y) - f_{\theta_2}(x,y)| \le \rho|\theta_1 - \theta_2|$.*

Now, we can bound the rate of uniform convergence of a function class in terms of covering number [3]:

Lemma 6 (Estimation) *Let $\epsilon > 0$ and $\mathcal{N}(\Theta, \epsilon)$ be the covering number of Θ with radius ϵ. Then,*

$$\Pr\left(\sup_{f_\theta \in \mathcal{F}_\Theta} \Big|\hat{J}^{m,n}_{\mathrm{RPC},\theta} - \mathbb{E}\Big[\hat{J}_{\mathrm{RPC},\theta}\Big]\Big| \ge \epsilon\right)$$
$$\le 2\mathcal{N}(\Theta, \frac{\epsilon}{4\rho\big(1+\alpha+2(\beta+\gamma)U\big)})\left(\exp\Big(-\frac{n\epsilon^2}{32M^2}\Big) + \exp\Big(-\frac{m\epsilon^2}{32M^2\alpha^2}\Big) + \exp\Big(-\frac{n\epsilon^2}{32U^2\beta^2}\Big) + \exp\Big(-\frac{m\epsilon^2}{32U^2\gamma^2}\Big)\right).$$

Proof: For notation simplicity, we define the operators

- $P(f) = \mathbb{E}_{P_{XY}}[f(x,y)]$ and $P_n(f) = \frac{1}{n}\sum_{i=1}^{n} f(x_i, y_i)$
- $Q(f) = \mathbb{E}_{P_X P_Y}[f(x,y)]$ and $Q_m(f) = \frac{1}{m}\sum_{j=1}^{m} f(x'_j, y'_j)$

Hence,

$$\begin{aligned}
&\Big|\hat{J}^{m,n}_{\mathrm{RPC},\theta} - \mathbb{E}\Big[\hat{J}_{\mathrm{RPC},\theta}\Big]\Big| \\
=&\left|P_n(f_\theta) - P(f_\theta) - \alpha Q_m(f_\theta) + \alpha Q(f_\theta) - \beta P_n(f^2_\theta) + \beta P(f^2_\theta) - \gamma Q_m(f^2_\theta) + \gamma Q(f^2_\theta)\right| \\
\le&\left|P_n(f_\theta) - P(f_\theta)\right| + \alpha\left|Q_m(f_\theta) - Q(f_\theta)\right| + \beta\left|P_n(f^2_\theta) - P(f^2_\theta)\right| + \gamma\left|Q_m(f^2_\theta) - Q(f^2_\theta)\right|
\end{aligned}$$

Let $\epsilon' = \frac{\epsilon}{4\rho\left(1+\alpha+2(\beta+\gamma)U\right)}$ and $T := \mathcal{N}(\Theta, \epsilon')$. Let $C = \{f_{\theta_1}, f_{\theta_2}, \cdots, f_{\theta_T}\}$ with $\{\theta_1, \theta_2, \cdots, \theta_T\}$ be such that $B_\infty(\theta_1, \epsilon'), \cdots, B_\infty(\theta_T, \epsilon')$ are ϵ' cover. Hence, for any $f_\theta \in \mathcal{F}_\Theta$, there is an $f_{\theta_k} \in C$ such that $\|\theta - \theta_k\|_\infty \leq \epsilon'$.

Then, for any $f_{\theta_k} \in C$:

$$
\begin{aligned}
&\left|\hat{J}^{m,n}_{\mathrm{RPC},\theta} - \mathbb{E}\left[\hat{J}_{\mathrm{RPC},\theta}\right]\right| \\
\leq& \left|P_n(f_\theta) - P(f_\theta)\right| + \alpha\left|Q_m(f_\theta) - Q(f_\theta)\right| + \beta\left|P_n(f_\theta^2) - P(f_\theta^2)\right| + \gamma\left|Q_m(f_\theta^2) - Q(f_\theta^2)\right| \\
\leq& \left|P_n(f_{\theta_k}) - P(f_{\theta_k})\right| + \left|P_n(f_\theta) - P_n(f_{\theta_k})\right| + \left|P(f_\theta) - P(f_{\theta_k})\right| \\
&+ \alpha\Big(\left|Q_m(f_{\theta_k}) - Q(f_{\theta_k})\right| + \left|Q_m(f_\theta) - Q_m(f_{\theta_k})\right| + \left|Q(f_\theta) - Q(f_{\theta_k})\right|\Big) \\
&+ \beta\Big(\left|P_n(f^2_{\theta_k}) - P(f^2_{\theta_k})\right| + \left|P_n(f^2_\theta) - P_n(f^2_{\theta_k})\right| + \left|P(f^2_\theta) - P(f^2_{\theta_k})\right|\Big) \\
&+ \gamma\Big(\left|Q_m(f^2_{\theta_k}) - Q(f^2_{\theta_k})\right| + \left|Q_m(f^2_\theta) - Q_m(f^2_{\theta_k})\right| + \left|Q(f^2_\theta) - Q(f^2_{\theta_k})\right|\Big) \\
\leq& \left|P_n(f_{\theta_k}) - P(f_{\theta_k})\right| + \rho\|\theta - \theta_k\| + \rho\|\theta - \theta_k\| \\
&+ \alpha\Big(\left|Q_m(f_{\theta_k}) - Q(f_{\theta_k})\right| + \rho\|\theta - \theta_k\| + \rho\|\theta - \theta_k\|\Big) \\
&+ \beta\Big(\left|P_n(f^2_{\theta_k}) - P(f^2_{\theta_k})\right| + 2\rho U\|\theta - \theta_k\| + 2\rho U\|\theta - \theta_k\|\Big) \\
&+ \gamma\Big(\left|Q_m(f^2_{\theta_k}) - Q(f^2_{\theta_k})\right| + 2\rho U\|\theta - \theta_k\| + 2\rho U\|\theta - \theta_k\|\Big) \\
=& \left|P_n(f_{\theta_k}) - P(f_{\theta_k})\right| + \alpha\left|Q_m(f_{\theta_k}) - Q(f_{\theta_k})\right| + \beta\left|P_n(f^2_{\theta_k}) - P(f^2_{\theta_k})\right| + \gamma\left|Q_m(f^2_{\theta_k}) - Q(f^2_{\theta_k})\right| \\
&+ 2\rho\big(1 + \alpha + 2(\beta+\gamma)U\big)\|\theta - \theta_k\| \\
\leq& \left|P_n(f_{\theta_k}) - P(f_{\theta_k})\right| + \alpha\left|Q_m(f_{\theta_k}) - Q(f_{\theta_k})\right| + \beta\left|P_n(f^2_{\theta_k}) - P(f^2_{\theta_k})\right| + \gamma\left|Q_m(f^2_{\theta_k}) - Q(f^2_{\theta_k})\right| + \frac{\epsilon}{2},
\end{aligned}
$$

where

- $\left|P_n(f_\theta) - P_n(f_{\theta_k})\right| \leq \rho\|\theta - \theta_k\|$ due to Assumption 2, and the result also applies for $\left|P(f_\theta) - P(f_{\theta_k})\right|$, $\left|Q_m(f_\theta) - Q_m(f_{\theta_k})\right|$, and $\left|Q(f_\theta) - Q(f_{\theta_k})\right|$.
- $\left|P_n(f^2_\theta) - P_n(f^2_{\theta_k})\right| \leq 2\|f_\theta\|_\infty\rho\|\theta - \theta_k\| \leq 2\rho U\|\theta - \theta_k\|$ due to Assumptions 1 and 2. The result also applies for $\left|P(f^2_\theta) - P(f^2_{\theta_k})\right|$, $\left|Q_m(f^2_\theta) - Q_m(f^2_{\theta_k})\right|$, and $\left|Q(f^2_\theta) - Q(f^2_{\theta_k})\right|$.

Hence,

$$
\begin{aligned}
&\Pr\left(\sup_{f_\theta \in \mathcal{F}_\Theta}\left|\hat{J}^{m,n}_{\mathrm{RPC},\theta} - \mathbb{E}\left[\hat{J}_{\mathrm{RPC},\theta}\right]\right| \geq \epsilon\right) \\
\leq& \Pr\left(\max_{f_{\theta_k} \in C}\left|P_n(f_{\theta_k}) - P(f_{\theta_k})\right| + \alpha\left|Q_m(f_{\theta_k}) - Q(f_{\theta_k})\right| + \beta\left|P_n(f^2_{\theta_k}) - P(f^2_{\theta_k})\right| + \gamma\left|Q_m(f^2_{\theta_k}) - Q(f^2_{\theta_k})\right| + \frac{\epsilon}{2} \geq \epsilon\right) \\
=& \Pr\left(\max_{f_{\theta_k} \in C}\left|P_n(f_{\theta_k}) - P(f_{\theta_k})\right| + \alpha\left|Q_m(f_{\theta_k}) - Q(f_{\theta_k})\right| + \beta\left|P_n(f^2_{\theta_k}) - P(f^2_{\theta_k})\right| + \gamma\left|Q_m(f^2_{\theta_k}) - Q(f^2_{\theta_k})\right| \geq \frac{\epsilon}{2}\right) \\
\leq& \sum_{k=1}^{T}\Pr\left(\left|P_n(f_{\theta_k}) - P(f_{\theta_k})\right| + \alpha\left|Q_m(f_{\theta_k}) - Q(f_{\theta_k})\right| + \beta\left|P_n(f^2_{\theta_k}) - P(f^2_{\theta_k})\right| + \gamma\left|Q_m(f^2_{\theta_k}) - Q(f^2_{\theta_k})\right| \geq \frac{\epsilon}{2}\right) \\
\leq& \sum_{k=1}^{T}\Pr\left(\left|P_n(f_{\theta_k}) - P(f_{\theta_k})\right| \geq \frac{\epsilon}{8}\right) + \Pr\left(\alpha\left|Q_m(f_{\theta_k}) - Q(f_{\theta_k})\right| \geq \frac{\epsilon}{8}\right) \\
&\qquad + \Pr\left(\beta\left|P_n(f^2_{\theta_k}) - P(f^2_{\theta_k})\right| \geq \frac{\epsilon}{8}\right) + \Pr\left(\gamma\left|Q_m(f^2_{\theta_k}) - Q(f^2_{\theta_k})\right| \geq \frac{\epsilon}{8}\right).
\end{aligned}
$$

With Hoeffding's inequality,

- $\Pr\left(|P_n(f_{\theta_k}) - P(f_{\theta_k})| \geq \frac{\epsilon}{8}\right) \leq 2\exp\Big(-\frac{n\epsilon^2}{32M^2}\Big)$
- $\Pr\left(\alpha\,|Q_m(f_{\theta_k}) - Q(f_{\theta_k})| \geq \frac{\epsilon}{8}\right) \leq 2\exp\Big(-\frac{m\epsilon^2}{32M^2\alpha^2}\Big)$
- $\Pr\left(\beta\,|P_n(f^2_{\theta_k}) - P(f^2_{\theta_k})| \geq \frac{\epsilon}{8}\right) \leq 2\exp\Big(-\frac{n\epsilon^2}{32U^2\beta^2}\Big)$
- $\Pr\left(\gamma\,|Q_m(f^2_{\theta_k}) - Q(f^2_{\theta_k})| \geq \frac{\epsilon}{8}\right) \leq 2\exp\Big(-\frac{m\epsilon^2}{32U^2\gamma^2}\Big)$

To conclude,

$$\Pr\left(\sup_{f_\theta \in \mathcal{F}_\Theta} \left|\hat{J}^{m,n}_{\mathrm{RPC},\theta} - \mathbb{E}\Big[\hat{J}_{\mathrm{RPC},\theta}\Big]\right| \geq \epsilon\right)$$
$$\leq 2\mathcal{N}(\Theta, \frac{\epsilon}{4\rho\big(1+\alpha+2(\beta+\gamma)U\big)})\left(\exp\Big(-\frac{n\epsilon^2}{32M^2}\Big) + \exp\Big(-\frac{m\epsilon^2}{32M^2\alpha^2}\Big) + \exp\Big(-\frac{n\epsilon^2}{32U^2\beta^2}\Big) + \exp\Big(-\frac{m\epsilon^2}{32U^2\gamma^2}\Big)\right).$$

□

Part II - Approximation: Neural Network Universal Approximation. We leverage the universal function approximation lemma of neural network

Lemma 7 (Approximation [17]) *Let $\epsilon > 0$. There exists $d \in \mathbb{N}$ and a family of neural networks $\mathcal{F}_\Theta := \{f_\theta : \theta \in \Theta \subseteq \mathbb{R}^d\}$ where Θ is compact, such that* $\inf_{f_\theta \in \mathcal{F}_\Theta} \left|\mathbb{E}\Big[\hat{J}_{\mathrm{RPC},\theta}\Big] - J_{\mathrm{RPC}}\right| \leq \epsilon$.

Part III - Bringing everything together. Now, we are ready to bring the estimation and approximation together to show that there exists a neural network θ^* such that, with high probability, $\hat{J}^{m,n}_{\mathrm{RPC},\theta}$ can approximate J_{RPC} with $n' = \min\{n, m\}$ at a rate of $O(1/\sqrt{n'})$:

Proposition 2 *With probability at least $1-\delta$, $\exists \theta^* \in \Theta$, $|J_{\mathrm{RPC}} - \hat{J}^{m,n}_{\mathrm{RPC},\theta}| = O(\sqrt{\frac{d+\log(1/\delta)}{n'}})$, where $n' = \min\{n, m\}$.*

Proof: The proof follows by combining Lemma 6 and 7.

First, Lemma 7 suggests, $\exists \theta^* \in \Theta$,

$$\left|\mathbb{E}\Big[\hat{J}_{\mathrm{RPC},\theta^*}\Big] - J_{\mathrm{RPC}}\right| \leq \frac{\epsilon}{2}.$$

Next, we perform analysis on the estimation error, aiming to find n, m and the corresponding probability, such that

$$\left|\hat{J}^{m,n}_{\mathrm{RPC},\theta} - \mathbb{E}\Big[\hat{J}_{\mathrm{RPC},\theta^*}\Big]\right| \leq \frac{\epsilon}{2}.$$

Applying Lemma 6 with the covering number of the neural network: $\Big(\mathcal{N}(\Theta,\epsilon) = O\Big(\exp\big(d\log(1/\epsilon)\big)\Big)$ [1]$\Big)$ and let $n' = \min\{n, m\}$:

$$\Pr\left(\sup_{f_\theta \in \mathcal{F}_\Theta} \left|\hat{J}^{m,n}_{\mathrm{RPC},\theta} - \mathbb{E}\Big[\hat{J}_{\mathrm{RPC},\theta}\Big]\right| \geq \frac{\epsilon}{2}\right)$$
$$\leq 2\mathcal{N}(\Theta, \frac{\epsilon}{8\rho\big(1+\alpha+2(\beta+\gamma)U\big)})\left(\exp\Big(-\frac{n\epsilon^2}{128M^2}\Big) + \exp\Big(-\frac{m\epsilon^2}{128M^2\alpha^2}\Big) + \exp\Big(-\frac{n\epsilon^2}{128U^2\beta^2}\Big) + \exp\Big(-\frac{m\epsilon^2}{128U^2\gamma^2}\Big)\right)$$
$$= O\Big(\exp\big(d\log(1/\epsilon) - n'\epsilon^2\big)\Big),$$

where the big-O notation absorbs all the constants that do not require in the following derivation. Since we want to bound the probability with $1-\delta$, we solve the ϵ such that

$$\exp\big(d\log(1/\epsilon) - n'\epsilon^2\big) \leq \delta.$$

With $\log(x) \leq x - 1$,

$$n'\epsilon^2 + d(\epsilon - 1) \geq n'\epsilon^2 + d\log\epsilon \geq \log(1/\delta),$$

where this inequality holds when

$$\epsilon = O\left(\sqrt{\frac{d + \log(1/\delta)}{n'}}\right).$$

□

D Proof of Proposition in the Main Text - From an Asymptotic Viewpoint

Here, we provide the variance analysis on $\hat{J}^{m,n}_{\mathrm{RPC}}$ via an asymptotic viewpoint. First, assuming the network is correctly specified, and hence there exists a network parameter θ^* satisfying $f^*(x,y) = f_{\theta^*}(x,y) = r_{\alpha,\beta,\gamma}(x,y)$. Then we recall that $\hat{J}^{m,n}_{\mathrm{RPC}}$ is a consistent estimator of J^{RPC} (see Proposition 2), and under regular conditions, the estimated network parameter $\hat{\theta}$ in $\hat{J}^{m,n}_{\mathrm{RPC}}$ satisfying the asymptotic normality in the large sample limit (see Theorem 5.23 in [39]). We recall the definition of $\hat{J}^{m,n}_{\mathrm{RPC},\theta}$ in equation 2 and let $n' = \min\{n, m\}$, the asymptotic expansion of $\hat{J}^{m,n}_{\mathrm{RPC}}$ has

$$\begin{aligned}
\hat{J}^{m,n}_{\mathrm{RPC},\theta^*} &= \hat{J}^{m,n}_{\mathrm{RPC},\hat{\theta}} + \dot{\hat{J}}^{m,n}_{\mathrm{RPC},\hat{\theta}}(\theta^* - \hat{\theta}) + o(\|\theta^* - \hat{\theta}\|) \\
&= \hat{J}^{m,n}_{\mathrm{RPC},\hat{\theta}} + \dot{\hat{J}}^{m,n}_{\mathrm{RPC},\hat{\theta}}(\theta^* - \hat{\theta}) + o_p(\frac{1}{\sqrt{n'}}) \\
&= \hat{J}^{m,n}_{\mathrm{RPC},\hat{\theta}} + o_p(\frac{1}{\sqrt{n'}}),
\end{aligned} \tag{4}$$

where $\dot{\hat{J}}^{m,n}_{\mathrm{RPC},\hat{\theta}} = 0$ since $\hat{\theta}$ is the estimation from $\hat{J}^{m,n}_{\mathrm{RPC}} = \sup_{f_\theta \in \mathcal{F}_\Theta} \hat{J}^{m,n}_{\mathrm{RPC},\theta}$.

Next, we recall the definition in equation 3:

$$\mathbb{E}[\hat{J}_{\mathrm{RPC},\hat{\theta}}] = \mathbb{E}_{P_{XY}}[f_{\hat{\theta}}(x,y)] - \alpha\mathbb{E}_{P_X P_Y}[f_{\hat{\theta}}(x,y)] - \frac{\beta}{2}\mathbb{E}_{P_{XY}}[f^2_{\hat{\theta}}(x,y)] - \frac{\gamma}{2}\mathbb{E}_{P_X P_Y}[f^2_{\hat{\theta}}(x,y)].$$

Likewise, the asymptotic expansion of $\mathbb{E}[\hat{J}_{\mathrm{RPC},\theta}]$ has

$$\begin{aligned}
\mathbb{E}[\hat{J}_{\mathrm{RPC},\hat{\theta}}] &= \mathbb{E}[\hat{J}_{\mathrm{RPC},\theta^*}] + \mathbb{E}[\dot{\hat{J}}_{\mathrm{RPC},\theta^*}](\hat{\theta} - \theta^*) + o(\|\hat{\theta} - \theta^*\|) \\
&= \mathbb{E}[\hat{J}_{\mathrm{RPC},\theta^*}] + \mathbb{E}[\dot{\hat{J}}_{\mathrm{RPC},\theta^*}](\hat{\theta} - \theta^*) + o_p(\frac{1}{\sqrt{n'}}) \\
&= \mathbb{E}[\hat{J}_{\mathrm{RPC},\theta^*}] + o_p(\frac{1}{\sqrt{n'}}),
\end{aligned} \tag{5}$$

where $\mathbb{E}[\dot{\hat{J}}_{\mathrm{RPC},\theta^*}] = 0$ since $\mathbb{E}[\hat{J}_{\mathrm{RPC},\theta^*}] = J_{\mathrm{RPC}}$ and θ^* satisfying $f^*(x,y) = f_{\theta^*}(x,y)$.

Combining equations 4 and 5:

$$
\begin{aligned}
\hat{J}^{m,n}_{\mathrm{RPC},\hat{\theta}} - \mathbb{E}[\hat{J}_{\mathrm{RPC},\hat{\theta}}] =& \hat{J}^{m,n}_{\mathrm{RPC},\theta^*} - J_{\mathrm{RPC}} + o_p(\frac{1}{\sqrt{n'}}) \\
=& \frac{1}{n}\sum_{i=1}^{n} f^*_\theta(x_i, y_i) - \alpha\frac{1}{m}\sum_{j=1}^{m} f^*_\theta(x'_j, y'_j) - \frac{\beta}{2}\frac{1}{n}\sum_{i=1}^{n} f^2_{\theta^*}(x_i, y_i) - \frac{\gamma}{2}\frac{1}{m}\sum_{j=1}^{m} f^2_{\theta^*}(x'_j, y'_j) \\
& - \mathbb{E}_{P_{XY}}[f^*(x,y)] + \alpha\mathbb{E}_{P_X P_Y}[f^*(x,y)] + \frac{\beta}{2}\mathbb{E}_{P_{XY}}\left[f^{*2}(x,y)\right] + \frac{\gamma}{2}\mathbb{E}_{P_X P_Y}\left[f^{*2}(x,y)\right] + o_p(\frac{1}{\sqrt{n'}}) \\
=& \frac{1}{n}\sum_{i=1}^{n} r_{\alpha,\beta,\gamma}(x_i, y_i) - \alpha\frac{1}{m}\sum_{j=1}^{m} r_{\alpha,\beta,\gamma}(x'_j, y'_j) - \frac{\beta}{2}\frac{1}{n}\sum_{i=1}^{n} r^2_{\alpha,\beta,\gamma}(x_i, y_i) - \frac{\gamma}{2}\frac{1}{m}\sum_{j=1}^{m} r^2_{\alpha,\beta,\gamma}(x'_j, y'_j) \\
& - \mathbb{E}_{P_{XY}}[r_{\alpha,\beta,\gamma}(x,y)] + \alpha\mathbb{E}_{P_X P_Y}[r_{\alpha,\beta,\gamma}(x,y)] + \frac{\beta}{2}\mathbb{E}_{P_{XY}}\left[r^2_{\alpha,\beta,\gamma}(x,y)\right] + \frac{\gamma}{2}\mathbb{E}_{P_X P_Y}\left[r^2_{\alpha,\beta,\gamma}(x,y)\right] \\
& + o_p(\frac{1}{\sqrt{n'}}) \\
=& \frac{1}{\sqrt{n}}\cdot\frac{1}{\sqrt{n}}\sum_{i=1}^{n}\left(r_{\alpha,\beta,\gamma}(x_i, y_i) - \frac{\beta}{2}r^2_{\alpha,\beta,\gamma}(x_i, y_i) - \mathbb{E}_{P_{XY}}\left[r_{\alpha,\beta,\gamma}(x,y) - \frac{\beta}{2}r^2_{\alpha,\beta,\gamma}(x,y)\right]\right) \\
& - \frac{1}{\sqrt{m}}\cdot\frac{1}{\sqrt{m}}\sum_{j=1}^{m}\left(\alpha r_{\alpha,\beta,\gamma}(x'_j, y'_j) + \frac{\gamma}{2}r^2_{\alpha,\beta,\gamma}(x'_j, y'_j) - \mathbb{E}_{P_X P_Y}\left[\alpha r_{\alpha,\beta,\gamma}(x,y) + \frac{\gamma}{2}r^2_{\alpha,\beta,\gamma}(x,y)\right]\right) \\
& + o_p(\frac{1}{\sqrt{n'}}).
\end{aligned}
$$

Therefore, the asymptotic Variance of $\hat{J}^{m,n}_{\mathrm{RPC}}$ is

$$
\mathrm{Var}[\hat{J}^{m,n}_{\mathrm{RPC}}] = \frac{1}{n}\mathrm{Var}_{P_{XY}}[r_{\alpha,\beta,\gamma}(x,y) - \frac{\beta}{2}r^2_{\alpha,\beta,\gamma}(x,y)] + \frac{1}{m}\mathrm{Var}_{P_X P_Y}[\alpha r_{\alpha,\beta,\gamma}(x,y) + \frac{\gamma}{2}r^2_{\alpha,\beta,\gamma}(x,y)] + o(\frac{1}{n'}).
$$

First, we look at $\mathrm{Var}_{P_{XY}}[r_{\alpha,\beta,\gamma}(x,y) - \frac{\beta}{2}r^2_{\alpha,\beta,\gamma}(x,y)]$. Since $\beta > 0$ and $-\frac{\alpha}{\gamma} \le r_{\alpha,\beta,\gamma} \le \frac{1}{\beta}$, simple calculation gives us $-\frac{2\alpha\gamma+\beta\alpha^2}{2\gamma^2} \le r_{\alpha,\beta,\gamma}(x,y) - \frac{\beta}{2}r^2_{\alpha,\beta,\gamma}(x,y) \le \frac{1}{2\beta}$. Hence,

$$
\mathrm{Var}_{P_{XY}}[r_{\alpha,\beta,\gamma}(x,y) - \frac{\beta}{2}r^2_{\alpha,\beta,\gamma}(x,y)] \le \max\left\{\left(\frac{2\alpha\gamma+\beta\alpha^2}{2\gamma^2}\right)^2, \left(\frac{1}{2\beta}\right)^2\right\}.
$$

Next, we look at $\mathrm{Var}_{P_X P_Y}[\alpha r_{\alpha,\beta,\gamma}(x,y) + \frac{\gamma}{2}r^2_{\alpha,\beta,\gamma}(x,y)]$. Since $\alpha \ge 0, \gamma > 0$ and $-\frac{\alpha}{\gamma} \le r_{\alpha,\beta,\gamma} \le \frac{1}{\beta}$, simple calculation gives us $-\frac{\alpha^2}{2\gamma} \le \alpha r_{\alpha,\beta,\gamma}(x,y) + \frac{\gamma}{2}r^2_{\alpha,\beta,\gamma}(x,y) \le \frac{2\alpha\beta+\gamma}{2\beta^2}$. Hence,

$$
\mathrm{Var}_{P_X P_Y}[\alpha r_{\alpha,\beta,\gamma}(x,y) + \frac{\gamma}{2}r^2_{\alpha,\beta,\gamma}(x,y)] \le \max\left\{\left(\frac{\alpha^2}{2\gamma}\right)^2, \left(\frac{2\alpha\beta+\gamma}{2\beta^2}\right)^2\right\}.
$$

Combining everything together, we restate the Proposition 2 in the main text:

Proposition 3 (Asymptotic Variance of $\hat{J}^{m,n}_{\mathrm{RPC}}$)

$$
\begin{aligned}
\mathrm{Var}[\hat{J}^{m,n}_{\mathrm{RPC}}] &= \frac{1}{n}\mathrm{Var}_{P_{XY}}[r_{\alpha,\beta,\gamma}(x,y) - \frac{\beta}{2}r^2_{\alpha,\beta,\gamma}(x,y)] + \frac{1}{m}\mathrm{Var}_{P_X P_Y}[\alpha r_{\alpha,\beta,\gamma}(x,y) + \frac{\gamma}{2}r^2_{\alpha,\beta,\gamma}(x,y)] + o(\frac{1}{n'}) \\
&\le \frac{1}{n}\max\left\{\left(\frac{2\alpha\gamma+\beta\alpha^2}{2\gamma^2}\right)^2, \left(\frac{1}{2\beta}\right)^2\right\} + \frac{1}{m}\max\left\{\left(\frac{\alpha^2}{2\gamma}\right)^2, \left(\frac{2\alpha\beta+\gamma}{2\beta^2}\right)^2\right\} + o(\frac{1}{n'})
\end{aligned}
$$

E Proof of Proposition in the Main Text - From Boundness of f_θ

As discussed in Assumption 1, for the estimation $\hat{J}^{m,n}_{\mathrm{RPC}}$, we can bound the function f_θ in $\mathcal{F}_\Theta$ within $[-\frac{\alpha}{\gamma}, \frac{1}{\beta}]$ without losing precision. Then, re-arranging $\hat{J}^{m,n}_{\mathrm{RPC}}$:

$$\sup_{f_\theta \in \mathcal{F}_\Theta} \frac{1}{n}\sum_{i=1}^{n} f_\theta(x_i, y_i) - \frac{1}{m}\sum_{j=1}^{m} \alpha f_\theta(x'_j, y'_j) - \frac{1}{n}\sum_{i=1}^{n} \frac{\beta}{2} f^2_\theta(x_i, y_i) - \frac{1}{m}\sum_{j=1}^{m} \frac{\gamma}{2} f^2_\theta(x'_j, y'_j)$$

$$\sup_{f_\theta \in \mathcal{F}_\Theta} \frac{1}{n}\sum_{i=1}^{n} \Big(f_\theta(x_i, y_i) - \frac{\beta}{2} f^2_\theta(x_i, y_i) \Big) + \frac{1}{m}\sum_{j=m}^{n} \Big(\alpha f_\theta(x'_j, y'_j) + \frac{\gamma}{2} f^2_\theta(x'_j, y'_j) \Big)$$

Then, since $-\frac{\alpha}{\gamma} \leq f_\theta(\cdot,\cdot) \leq \frac{1}{\beta}$, basic calculations give us

$$-\frac{2\alpha\gamma + \beta\alpha^2}{2\gamma^2} \leq f_\theta(x_i, y_i) - \frac{\beta}{2} f^2_\theta(x_i, y_i) \leq \frac{1}{2\beta} \text{ and } -\frac{\alpha^2}{2\gamma} \leq \alpha f_\theta(x'_j, y'_j) + \frac{\gamma}{2} f^2_\theta(x'_j, y'_j) \leq \frac{2\alpha\beta + \gamma}{2\beta^2}.$$

The resulting variances have

$$\mathrm{Var}[f_\theta(x_i, y_i) - \frac{\beta}{2} f^2_\theta(x_i, y_i)] \leq \max\left\{ \Big(\frac{2\alpha\gamma + \beta\alpha^2}{2\gamma^2}\Big)^2, \Big(\frac{1}{2\beta}\Big)^2 \right\}$$

and

$$\mathrm{Var}[\alpha f_\theta(x'_j, y'_j) + \frac{\gamma}{2} f^2_\theta(x'_j, y'_j)] \leq \max\left\{ \Big(\frac{\alpha^2}{2\gamma}\Big)^2, \Big(\frac{2\alpha\beta + \gamma}{2\beta^2}\Big)^2 \right\}.$$

Taking the mean of m, n independent random variables gives the result:

Proposition 4 (Variance of $\hat{J}^{m,n}_{\mathrm{RPC}}$)

$$\mathrm{Var}[\hat{J}^{m,n}_{\mathrm{RPC}}] \leq \frac{1}{n}\max\left\{ \Big(\frac{2\alpha\gamma + \beta\alpha^2}{2\gamma^2}\Big)^2, \Big(\frac{1}{2\beta}\Big)^2 \right\} + \frac{1}{m}\max\left\{ \Big(\frac{\alpha^2}{2\gamma}\Big)^2, \Big(\frac{2\alpha\beta + \gamma}{2\beta^2}\Big)^2 \right\}.$$

F Experimental setup

We briefly discuss the training and evaluation details into three modules: 1) related and unrelated data construction, 2) pre-training, and 3) fine-tuning and evaluation. In the vision experiment, we construct the related images by applying different augmentations on the same image. Hence, when $(x, y) \sim P_{XY}$, x and y are the same image with different augmentations. The unrelated images are two randomly selected samples. In the speech experiment, we define the current latent feature (feature at time t) and the future samples (samples at time $> t$) as related data. In other words, the feature in the latent space should contain information that can be used to infer future time steps. A latent feature and randomly selected samples would be considered as unrelated data. The pre-training stage refers to the self-supervised training by a contrastive learning objective. We use neural networks to parametrize the function f in the definition of J_{RPC} using the constructed related and unrelated data. Convolutional neural networks are used for vision experiments. Transformers [40] and LSTMs [16] are used for speech experiments. After the pre-training stage, we fix the parameters in the pre-trained networks and add a small fine-tuning network on top of them. Then, we fine-tune this small network with the downstream labels in the data's training split. For the fine-tuning network, both vision and speech experiments consider multi-layer perceptrons. Last, we evaluate the fine-tuned representations on the data's test split.

We include the relative parameters that lead to the highest accuracy for each dataset in Table 5. In general, we found that a small β and large γ in J_{RPC} would result in the best performance across different tasks. We also included the usage of hidden normalization and temperature parameter τ in Table 6. We notice that adding hidden normalization will stabilize training and increase performance for prior objectives, but J_{RPC} might not need hidden normalization to achieve these two.

G Implementation of Experiments

Domain	Dataset	Objective(s)	Hidden Normalization	Temperature τ
Visual	CIFAR-10	$J_{\mathrm{DV}}/J_{\mathrm{NWJ}}/J_{\mathrm{JS}}/J_{\mathrm{WPC}}/J_{\mathrm{CPC}}$	✓	0.5
Visual	CIFAR-10	J_{RPC}	✗	128
Visual	CIFAR-100	$J_{\mathrm{DV}}/J_{\mathrm{NWJ}}/J_{\mathrm{JS}}/J_{\mathrm{WPC}}/J_{\mathrm{CPC}}$	✓	0.5
Visual	CIFAR-100	J_{RPC}	✗	128
Visual	STL-10	$J_{\mathrm{DV}}/J_{\mathrm{NWJ}}/J_{\mathrm{JS}}/J_{\mathrm{WPC}}/J_{\mathrm{CPC}}$	✓	0.5
Visual	STL-10	J_{RPC}	✗	128
Visual	ImageNet	$J_{\mathrm{JS}}/J_{\mathrm{WPC}}/J_{\mathrm{CPC}}$	✓	0.1
Visual	ImageNet	J_{RPC}	✗	32
Speech	Librispeech	$J_{\mathrm{DV}}/J_{\mathrm{NWJ}}/J_{\mathrm{CPC}}$	✗	1
Speech	Librispeech	J_{RPC}	✗	1

Table 6: Hidden norm usage and temperature parameters used in each set of experiments.

For visual representation learning, we follow the implementation in https://github.com/google-research/simclr. For speech representation learning, we follow the implementation in https://github.com/facebookresearch/CPC_audio. For MI estimation, we follow the implementation in https://github.com/yaohungt/Pointwise_Dependency_Neural_Estimation/tree/master/MI_Est_and_CrossModal..

Dataset	α	β	γ
CIFAR-10	1.0	0.005	1.0
CIFAR-100	2.0	0.01	1.0
STL-10	1.0	0.005	1.0
ImageNet	3.0	0.01	0.5
Librispeech	1.0	0.2	1.0

Table 5: Relative parameters that lead to the best performance for each dataset.

H Relative Predictive Coding on Vision

The whole pipeline of pretraining contains the following steps [7]: First, a stochastic data augmentation will transform one image sample x_k to two different but correlated augmented views, x'_{2k-1} and x'_{2k}. Then a base encoder $f(\cdot)$ implemented using ResNet [14] will extract representations from augmented views, creating representations h_{2k-1} and h_{2k}. Later a small neural network $g(\cdot)$ called projection head will map h_{2k-1} and h_{2k} to z_{2k-1} and z_{2k} in a different latent space. For each minibatch of N samples, there will be $2N$ views generated. For each image x_k there will be one positive pair x'_{2k-1} and x'_{2k} and $2(N-1)$ negative samples. The RPC loss between a pair of positive views, x'_i and x'_j (augmented from the same image) , can be calculated by the substitution $f_\theta(x'_i, x'_j) = (z_i \cdot z_j)/\tau = s_{i,j}$ (τ is a hyperparameter) to the definition of RPC:

$$\ell^{\mathrm{RPC}}_{i,j} = -(s_{i,j} - \frac{\alpha}{2(N-1)} \sum_{k=1}^{2N} \mathbf{1}_{[\mathrm{k}\neq\mathrm{i}]} s_{i,k} - \frac{\beta}{2} s^2_{i,j} - \frac{\gamma}{2 \cdot 2(N-1)} \sum_{k=1}^{2N} \mathbf{1}_{[\mathrm{k}\neq\mathrm{i}]} s^2_{i,k}) \tag{6}$$

For losses other than RPC, a hidden normalization of $s_{i,j}$ is often required by replacing $z_i \cdot z_j$ with $(z_i \cdot z_j)/|z_i||z_j|$. CPC and WPC adopt this, while other objectives needs it to help stabilize training variance. RPC does not need this normalization.

I CIFAR-10/-100 and ImageNet Experiments Details

ImageNet Following the settings in [6, 7], we train the model on Cloud TPU with 128 cores, with a batch size of $4,096$ and global batch normalization [2] [18]. The largest model we train is a 152-layer ResNet with selective kernels (SK) [23] and $2\times$ wider channels. We use the LARS optimizer [42] with momentum 0.9. The learning rate linearly increases for the first 20 epochs, reaching a maximum of 6.4, then decayed with cosine decay schedule. The weight decay is 10^{-4}. A MLP projection head $g(\cdot)$ with three layers is used on top of the ResNet encoder. Unlike SimCLRv2 [7], we do not use a memory buffer, and train the model for only 100 epochs rather than 800 epochs due to computational

[2] For WPC [29], the global batch normalization during pretraining is disabled since we enforce 1-Lipschitz by gradient penalty [12].

constraints. These two options slightly reduce CPC's performance benchmark for about 2% with the exact same setting. The unsupervised pre-training is followed by a supervised fine-tuning. Following SimCLRv2 [6, 7], we fine-tune the 3-layer $g(\cdot)$ for the downstream tasks. We use learning rates 0.16 and 0.064 for standard 50-layer ResNet and larger 152-layer ResNet respectively, and weight decay and learning rate warmup are removed. Different from SimCLRv2 [7], we use a batch size of $4,096$, and we do not use global batch normalization for fine-tuning. For J_{RPC} we disable hidden normalization and use a temperature $\tau = 32$. For all other objectives, we use hidden normalization and $\tau = 0.1$ following previous work [7].

CIFAR-10/-100 Following the settings in [6], we train the model on a single GPU, with a batch size of 512 and global batch normalization [18]. We use ResNet [14] of depth 18 and depth 50, and does not use Selective Kernel [23] or a multiplied width size. We use the LARS optimizer [42] with momentum 0.9. The learning rate linearly increases for the first 20 epochs, reaching a maximum of 6.4, then decayed with cosine decay schedule. The weight decay is 10^{-4}. A MLP projection head $g(\cdot)$ with three layers is used on top of the ResNet encoder. Unlike SimCLRv2 [7], we do not use a memory buffer. We train the model for 1000 epochs. The unsupervised pre-training is followed by a supervised fine-tuning. Following SimCLRv2 [6, 7], we fine-tune the 3-layer $g(\cdot)$ for the downstream tasks. We use learning rates 0.16 for standard 50-layer ResNet , and weight decay and learning rate warmup are removed. For J_{RPC} we disable hidden normalization and use a temperature $\tau = 128$. For all other objectives, we use hidden normalization and $\tau = 0.5$ following previous work [7].

STL-10 We also perform the pre-training and fine-tuning on STL-10 [9] using the model proposed in prior work [8], which indirectly approximate the distribution of negative samples so that the objective is *debiased*. However, their implementation of contrastive learning is consistent with SimCLRv2 [6]. We use a ResNet with depth 50 as an encoder for pre-training, with Adam optimizer, learning rate 0.001 and weight decay 10^{-6}. The temperature τ is set to 0.5 for all objectives other than J_{RPC}, which disables hidden normalization and use $\tau = 128$. The downstream task performance increases from 83.4% of J_{CPC} to 84.1% of J_{RPC}.

J Relative Predictive Coding on Speech

For speech representation learning, we adopt the general architecture from prior work [28, 32]. Given an input signal $x_{1:T}$ with T time steps, we first pass it through an encoder ϕ_θ parametrized by θ to produce a sequence of hidden representations $\{h_{1:T}\}$ where $h_t = \phi_\theta(x_t)$. After that, we obtain the contextual representation c_t at time step t with a sequential model ψ_ρ parametrized by ρ: $c_t = \psi_\rho(h_1, \ldots, h_t)$, where c_t contains context information before time step t. For unsupervised pre-training, we use a multi-layer convolutional network as the encoder ϕ_θ, and an LSTM with hidden dimension 256 as the sequential model ψ_ρ. Here, the contrastiveness is between the positive pair (h_{t+k}, c_t) where k is the number of time steps ahead, and the negative pairs (h_i, c_t), where h_i is randomly sampled from N, a batch of hidden representation of signals assumed to be unrelated to c_t. The scoring function f based on Equation 1 at step t and look-ahead k will be $f_k = f_k(h, c_t) = \exp((h)^\top W_k c_t)$, where W_k is a learnable linear transformation defined separately for each $k \in \{1, ..., K\}$ and K is predetermined as 12 time steps. The loss in Equation 1 will then be formulated as:

$$\ell^{\mathrm{RPC}}_{t,k} = -(f_k(h_{t+k}, c_t) - \frac{\alpha}{|N|}\sum_{h_i \in N} f_k(h_i, c_t) - \frac{\beta}{2} f_k^2(h_{t+k}, c_t) - \frac{\gamma}{2|N|}\sum_{h_i \in N} f_k^2(h_i, c_t)) \quad (7)$$

As shown in Table 3, J_{RPC} has better downstream task performance, and is closer to the performance from a fully supervised model.

K Empirical Observations on Variance and Minibatch Size

Training Stability Experiment Setup We perform the variance comparison of J_{DV}, J_{NWJ} and the proposed J_{RPC}. The empirical experiments are performed using SimCLRv2 [7] on CIFAR-10 dataset. We use a ResNet of depth 18, with batch size of 512. We train each objective with 30K training steps and record their value. In Figure 1(a), we use a temperature $\tau = 128$ for all objectives. Unlike other experiments, where hidden normalization is applied to other objectives, we remove

hidden normarlization for all objectives due to the reality that objectives after normalization does not reflect their original values. From Figure 1(a), J_{RPC} enjoys lower variance and more stable training compared to J_{DV} and J_{NWJ}.

Then we use the pre-trained representation to perform downstream tasks. Since we train for only 30K steps, the results are not comparable with the results provided in the main text. However, the downstream task performance using representations learned by J_{RPC} and J_{CPC} significantly outperform those learned by the exploding J_{DV} and J_{NWJ} (more than 40% relative improvement from J_{DV} and J_{NWJ}).

Minibatch Size Experimental Setup We perform experiments on the effect of batch size on downstream performances for different objective. The experiments are performed using SimCLRv2 [7] on CIFAR-10 dataset, as well as the model from prior work [32] on LibriSpeech-100h dataset [30]. For vision task, we use the default temperature $\tau = 0.5$ from SimCLRv2 [7] and hidden normalization for J_{CPC}. For J_{RPC} in vision and speech tasks we use a temperature of $\tau = 128$ and do not use hidden normalization.