Exercise Session 1

Learning at Scale: Supervised, Self-Supervised, and Beyond

Prepared by Xiuying Wei, Johann Wenckstern, Petr Grinberg

Overview

Additional Reading Materials. We recommend following three papers:
[1] Oord, Aaron van den, Yazhe Li, and Oriol Vinyals. "**Representation Learning with Contrastive Predictive Coding**." arXiv preprint arXiv:1807.03748 (2018).
[2] Chen, Ting, et al. "**A Simple Framework for Contrastive Learning of Visual Representations**." International Conference on Machine Learning (ICML). PMLR, 2020.
[3] Radford, Alec, et al. "**Learning Transferable Visual Models from Natural Language Supervision**." International Conference on Machine Learning. PMLR, 2021.

Task 1. InfoNCE in Contrastive Learning.

In contrastive learning, the goal is to learn useful representations without labels. For each condition c, the model is trained to identify the single positive sample among K distractors. For example, in instance discrimination for images, c is one augmented view of an image, x^+ is another independent augmentation of the same image, and the negatives $\{x_i^-\}$ are views of other images (e.g., from the same minibatch or from a memory queue). In vision language tasks, c is an image, x denotes captions, x^+ is the matched caption, and the negatives are captions of other images. A common objective for this is the InfoNCE loss. We will investigate it from a probabilistic perspective in this task.

Suppose that, given a condition c, we form a candidate set $X = \{x_0, \ldots, x_K\}$ where $x_0 = x^+$ is the positive sample and $x_1, \ldots, x_K$ are K negative samples drawn i.i.d. from $q_{\text{noise}}(x)$. The InfoNCE objective is defined as

$$\mathcal{L}_{\text{InfoNCE}} = -\sum_{(X,c)\in D} \left[\log \frac{\dfrac{p_{\text{data}}(x^+ \mid c)}{q_{\text{noise}}(x^+)}}{\sum_{j=0}^{K} \dfrac{p_{\text{data}}(x_j \mid c)}{q_{\text{noise}}(x_j)}} \right],$$

where the term inside the logarithm represents the probability of correctly identifying x^+ as the positive sample among all candidates.

In practice, we typically set $q_{\text{noise}}(x) = p_{\text{data}}(x)$, so the numerator simplifies to $\frac{p_{\text{data}}(x^+|c)}{p_{\text{data}}(x^+)}$, whose logarithm equals the pointwise mutual information. Then, we can encode x and c with neural networks to obtain embeddings, define a score to approximate this density ratio. Consequently, the InfoNCE loss encourages the model to learn effective representations that captures mutual information between x and c, by learning a higher estimated ratio (or score) for true positive pairs than for negative samples.

(a) **Connection to cross-entropy loss.** Choose logits whose exponentials match the relative weights of candidates in the InfoNCE numerator/denominator; then build connection between InfoNCE and cross-entropy loss.

(b) **Relation to NCE.** The local NCE loss is defined as

$$\mathcal{L}_{\text{NCE}} = \sum_{(X,c)\in D} \left[-\log \frac{p_{\text{data}}(x^+ \mid c)}{p_{\text{data}}(x^+ \mid c) + K\, q_{\text{noise}}(x^+)} - \sum_{j=1}^{K} \log \frac{K\, q_{\text{noise}}(x_j^-)}{p_{\text{data}}(x_j^- \mid c) + K\, q_{\text{noise}}(x_j^-)} \right].$$

Show that this objective can be seen as optimizing the same logits as the InfoNCE but with a binary cross-entropy loss.

(c) **Effect of K.** In the case $q_{\text{noise}}(x) = p_{\text{data}}(x)$, analyze the effect of the number of negative samples K on the InfoNCE loss.

Hint: In this case, the logarithm of the numerator equals the pointwise mutual information. Optimizing the InfoNCE loss corresponds to maximizing a lower bound on the mutual information.

Solution

(a) **Connection to cross-entropy loss.** Define logits $s_j = \log p_{\text{data}}(x_j \mid c) - \log q_{\text{noise}}(x_j)$. Then

$$\mathcal{L}_{\text{InfoNCE}} = -\sum_{(X,c)\in D} \log \frac{\exp(s^+)}{\sum_{j=0}^{K} \exp(s_j)} = -\sum_{(X,c)\in D} \sum_{j=0}^{K} y_j \log \operatorname{softmax}(s)_j,$$

where y is the one-hot label for the true sample x^+ (i.e., for s^+). Thus this is exactly the multi-class softmax cross-entropy between y and the model distribution $P(\cdot \mid X, c)$.

In practice for contrastive learning, one example is $s_j = \langle f(x_j), f(c)\rangle/\tau$ (with temperature τ, network f), giving

$$-\sum_{(X,c)\in D} \log \frac{\exp\big(\langle f(x^+), f(c)\rangle/\tau\big)}{\sum_{j=0}^{K} \exp\big(\langle f(x_j), f(c)\rangle/\tau\big)}.$$

(b) **Relation to NCE.** In (local) NCE, each pair (x, c) is classified as *data* vs. *noise*. With K noises per positive and using the logits s_j defined in (a), we have:

$$\frac{p_{\text{data}}(x^+ \mid c)}{p_{\text{data}}(x^+ \mid c) + K\, q_{\text{noise}}(x^+)} = \sigma(s^+ - \log K), \tag{1}$$

$$\frac{K\, q_{\text{noise}}(x^-)}{p_{\text{data}}(x^- \mid c) + K\, q_{\text{noise}}(x^-)} = \sigma\big(-s^- + \log K\big) = 1 - \sigma(s^- - \log K), \tag{2}$$

with σ as the sigmoid function. Then, we have

$$\mathcal{L}_{\text{NCE}} = -\log \sigma(s^+ - \log K) - \sum_{j=1}^{K} \log\left(1 - \sigma(s_j^- - \log K)\right) \tag{3}$$
$$= -\big(y \log \sigma(z) + (1-y)\log\big(1-\sigma(z)\big)\big), \tag{4}$$

with $z = s - \log K$, $y \in \{0,1\}$ denote the binary label ($y=1$ for positive, $y=0$ for negative). Thus, NCE and InfoNCE optimize the same logits s_j; they differ only in how the likelihood is factored (sum of binary log-losses vs. one multi-class log-loss).

(c) **Effect of K.** For the case where $q_{\text{noise}}(x) = p_{\text{data}}(x)$, the mutual information is

$$I(x^+; c) = \sum_{x^+, c} p(x^+, c) \log \frac{p(x^+ \mid c)}{p(x^+)}.$$

If the labels are fully reliable, the InfoNCE estimator yields the lower bound (see Oord et al., 2018, Appendix A for proof)

$$I(x^+; c) \geq \log(K+1) - \mathcal{L}_{\text{InfoNCE}}.$$

By plugging the loss in the right-hand side, we can see that as K increases, the lower bound becomes tighter, which can often lead to better performance for models trained with the InfoNCE loss. However, in practice a very large K may introduce too much noise in the negatives and increases compute/memory cost, so K should be balanced with its temperature factor and batch design.

Task 2. Masked Language Modeling as Pseudo-Likelihood and -Perplexity.

Consider a sequence $x = (x_1, \ldots, x_T)$ from a data distribution $\mathcal{D}$. For masked language modeling (MLM), let us draw a random mask set $M \subseteq \{1, \ldots, T\}$ by sampling each position independently with probability $q \in (0,1)$. Let $x_{\setminus t}$ denote x with the token in position t hidden (or replaced) and other tokens visible. Let x_t be an actual token in the position t. The MLM training objective is

$$\mathcal{L}_{\text{MLM}}(\theta) = -\mathbb{E}_{x\sim\mathcal{D}}\, \mathbb{E}_M \left[\sum_{t\in M} \log p_\theta\big(x_t \mid x_{\setminus t}\big)\right]. \tag{5}$$

(a) **Connection to pseudo-likelihood.** Define the (negative) pseudo log-likelihood (NPLL):

$$\text{NPLL}(\theta) = -\mathbb{E}_{x\sim\mathcal{D}} \left[\sum_{t=1}^{T} \log p_\theta\big(x_t \mid x_{\setminus t}\big)\right]. \tag{6}$$

Show that under independent Bernoulli masking at rate q, i.e., $M \overset{iid}{\sim} \text{Bern}(q)$:

$$\mathbb{E}_M \left[\sum_{t\in M} \log p_\theta(x_t \mid x_{\setminus t})\right] = q \sum_{t=1}^{T} \log p_\theta(x_t \mid x_{\setminus t}), \tag{7}$$

and hence $\mathcal{L}_{\text{MLM}}(\theta) = q \cdot \text{NPLL}(\theta)$.

Hint: Use indicators $\mathbf{1}_{\{t\in M\}}$ and linearity of expectation.

(b) **Pseudo-perplexity and an unbiased estimator.** Define the pseudo-perplexity (PPPL) for a sequence x by

$$\text{PPPL}(x) \;=\; \exp\left(\frac{1}{T}\sum_{t=1}^{T} -\log p_\theta\big(x_t \mid x_{\setminus t}\big)\right). \tag{8}$$

(i) Show that $\log \text{PPPL}(x)$ equals the average token-wise NPLL.

(ii) Propose a practical *unbiased* single-pass estimator of $S(x) = \sum_{t=1}^{T} -\log p_\theta(x_t \mid x_{\setminus t})$ by sampling one index $U \sim \text{Unif}\{1,\ldots,T\}$ and evaluating only $-\log p_\theta(x_U \mid x_{\setminus U})$. Then, prove that it is indeed unbiased.

(c) **Relation to autoregressive maximum likelihood.** An autoregressive (AR) model maximizes

$$\log p_\theta(x) = \sum_{t=1}^{T} \log p_\theta(x_t \mid x_{<t}). \tag{9}$$

(i) Explain why AR likelihood can be evaluated exactly, whereas MLM/pseudo-likelihood generally cannot yield a normalized joint $p_\theta(x)$.

(ii) *Bonus:* State a modeling assumption(s) under which minimizing $\text{NPLL}(\theta)$ is statistically consistent. That is, if θ_n denotes the estimator from n samples, then $\text{NPLL}(\theta_n) \to \text{NPLL}(\theta^\star)$ and $\theta_n \to \theta^\star$ as $n \to \infty$. Sketch the proof of consistency.
Hint: Consider applying results from M-estimation theory.

(iii) Give one advantage and one limitation of MLM vs. AR for downstream tasks.

Bonus (BERT 80/10/10). In BERT, of the selected tokens, 80% are replaced by `[MASK]`, 10% by a random token, and 10% are left unchanged. Argue how this reduces train–test mismatch and prevents over-reliance on `[MASK]`; predict qualitative effects of using 100% or 0% `[MASK]`.

Solution

(a) **Connection to pseudo-likelihood.** Let $I_t := \mathbf{1}_{\{t\in M\}}$ with $I_t \overset{iid}{\sim} \text{Bern}(q)$, independent of x. Then

$$\begin{aligned}\mathbb{E}_M\big[\textstyle\sum_{t\in M} \log p_\theta(x_t \mid x_{\setminus t})\big] &= \mathbb{E}_M\Big[\textstyle\sum_{t=1}^{T} I_t \,\log p_\theta(x_t \mid x_{\setminus t})\Big] = \\ &= \textstyle\sum_{t=1}^{T} \mathbb{E}[I_t]\,\log p_\theta(x_t \mid x_{\setminus t}) = q\textstyle\sum_{t=1}^{T} \log p_\theta(x_t \mid x_{\setminus t}).\end{aligned}$$

Taking expectation over $x \sim \mathcal{D}$ yields

$$\begin{aligned}\mathcal{L}_{\text{MLM}}(\theta) &= -\,\mathbb{E}_{x\sim\mathcal{D}}\,\mathbb{E}_M\big[\textstyle\sum_{t\in M} \log p_\theta(x_t \mid x_{\setminus t})\big] \\ &= -\,q\,\mathbb{E}_{x\sim\mathcal{D}}\Big[\textstyle\sum_{t=1}^{T} \log p_\theta(x_t \mid x_{\setminus t})\Big] = q \cdot \text{NPLL}(\theta).\end{aligned}$$

(b) **Pseudo-perplexity and an unbiased estimator.**

(i) By definition,

$$\log \mathrm{PPPL}(x) = \log \exp\left(\frac{1}{T}\sum_{t=1}^{T} -\log p_\theta(x_t \mid x_{\setminus t})\right) = \frac{1}{T}\sum_{t=1}^{T} -\log p_\theta(x_t \mid x_{\setminus t}),$$

which is the average token-wise NPLL.

(ii) A single-pass unbiased estimator of $S(x) = \sum_{t=1}^{T} -\log p_\theta(x_t \mid x_{\setminus t})$ is

$$\widehat{S}(x) = T \cdot \big(-\log p_\theta(x_U \mid x_{\setminus U})\big), \quad \text{where } U \sim \mathrm{Unif}\{1, \ldots, T\}.$$

Intuition: We estimate the sum of T terms by sampling one uniformly at random and scaling by T.

Unbiased-ness follows by linearity of expectation:

$$\begin{aligned}\mathbb{E}_U\big[T \cdot \big(-\log p_\theta(x_U \mid x_{\setminus U})\big)\big] &= \textstyle\sum_{t=1}^{T} \Pr(U = t)\, T \cdot \big(-\log p_\theta(x_t \mid x_{\setminus t})\big) = \\ &= \textstyle\sum_{t=1}^{T} \frac{1}{T}\, T \cdot \big(-\log p_\theta(x_t \mid x_{\setminus t})\big) = S(x).\end{aligned}$$

(c) **Relation to autoregressive maximum likelihood.**

(i) For an autoregressive (AR) model,

$$\log p_\theta(x) = \sum_{t=1}^{T} \log p_\theta(x_t \mid x_{<t}) \Rightarrow p_\theta(x) = \prod_{t=1}^{T} p_\theta(x_t \mid x_{<t}),$$

where each factor is directly given by the model, so we can compute likelihood exactly.

For the MLM/pseudo-likelihood, if the conditionals came from a joint normalized $p_\theta(x)$, they would satisfy:

$$p_\theta(x_t|x_{\setminus t}) = \frac{p_\theta(x)}{\sum_{x'_t} p_\theta(x_1, \ldots, x_{t-1}, x'_t, x_{t+1}, \ldots, x_T)} \tag{*}$$

(i.e., marginalizing out position t from the joint). But there is no guarantee that such a joint exists. We model the conditional distributions directly without constraints to ensure compatibility.

Here's a counterexample showing incompatibility. Consider $T = 2$ with tokens u, v. Suppose the MLM model defines:

$$p_\theta(x_1 = u|x_2 = v) = 1 \tag{10}$$
$$p_\theta(x_2 = v|x_1 = u) = 0 \tag{11}$$

If these came from a joint $p_\theta(x_1, x_2)$, we would need:

$$p_\theta(u, v) = p_\theta(x_1 = u|x_2 = v) \cdot p_\theta(x_2 = v) = 1 \cdot p_\theta(v)$$

and also:

$$p_\theta(u, v) = p_\theta(x_2 = v|x_1 = u) \cdot p_\theta(x_1 = u) = 0 \cdot p_\theta(u) = 0$$

This gives $p_\theta(v) = 0$, but then the first conditional wouldn't be well-defined (division by zero in the marginal). Therefore, no joint distribution exists that yields these conditionals.

(ii) For statistical consistency of minimizing NPLL(θ), we need different assumptions than for joint distributions.

Key point: We do NOT assume the conditionals come from a joint distribution (which would contradict part (i)).

Sufficient assumptions for consistency:

- The data is generated i.i.d. from some true distribution with conditionals $p^*(x_t \mid x_{\setminus t})$
- The model class is correctly specified: $\exists \theta^* \in \Theta$ such that $p_{\theta^*}(x_t \mid x_{\setminus t}) = p^*(x_t \mid x_{\setminus t})$ for all t
- Identifiability: if $p_\theta(x_t \mid x_{\setminus t}) = p_{\theta'}(x_t \mid x_{\setminus t})$ for all t and almost all x, then $\theta = \theta'$
- Standard regularity conditions (e.g., compact parameter space, continuous likelihood)

Sketch of consistency proof: Define the empirical estimator from n samples $x^{(1)}, \ldots, x^{(n)}$:

$$\theta_n = \arg\max_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} \sum_{t=1}^{T} \log p_\theta(x_t^{(i)} \mid x_{\setminus t}^{(i)})$$

The population objective is:

$$\mathcal{Q}(\theta) = \mathbb{E}_{x \sim p^*}\left[\sum_{t=1}^{T} \log p_\theta(x_t \mid x_{\setminus t})\right]$$

For each position t, by the KL divergence inequality:

$$\mathbb{E}_{x \sim p^*}[\log p_\theta(x_t \mid x_{\setminus t})] \leq \mathbb{E}_{x \sim p^*}[\log p^*(x_t \mid x_{\setminus t})]$$

with equality if and only if $p_\theta(x_t \mid x_{\setminus t}) = p^*(x_t \mid x_{\setminus t})$ almost surely.

Therefore $\mathcal{Q}(\theta) \leq \mathcal{Q}(\theta^*)$ with equality only at $\theta = \theta^*$ (by identifiability).

By the law of large numbers, the empirical objective converges to the population objective uniformly over Θ. Since θ^* is the unique maximizer of $\mathcal{Q}(\theta)$, standard M-estimation theory gives us $\theta_n \to \theta^*$ in probability as $n \to \infty$.

Remark: The assumption of correct specification can be relaxed. Even if no θ^* exactly matches the true conditionals, consistency still holds for the θ^* that minimizes the KL divergence to the true conditionals, making pseudo-likelihood robust to model misspecification.

Note: Knowledge of M-estimation theory is not relevant for the exam.

(iii) *One advantage of MLM:* It supports bidirectional context, making it natural for tasks that require information from both directions (e.g., fill-in-the-blank or fill-in-the-middle). Training with bidirectional context often yields strong representations for fill-in-the-blank and encoder-style downstream tasks. One limitation: to enable generation, an MLM typically needs an encoder–decoder architecture, whereas an autoregressive (AR) model requires only a decoder. Moreover, under a strict MLM objective, only the masked tokens contribute to the loss in each forward pass, so token usage is less efficient than in AR models.

Bonus (BERT 80/10/10). Among the tokens chosen for prediction, replacing 80% with `[MASK]`, 10% with a random token, and leaving 10% unchanged reduces train–test mismatch and discourages over-reliance on the `[MASK]` because:

- The 10% unchanged positions force the model to sometimes predict when the true token is visible, mitigating the fact that `[MASK]` never appears at test time.
- The 10% random replacements inject noise so the model cannot treat `[MASK]` as the sole prediction signal.

Qualitatively, using 100% `[MASK]` increases train–test mismatch and risks overfitting to the special token; using 0% `[MASK]` makes many targets trivially copyable (identity shortcut), weakening the learning signal, harming generalization, and reducing model understanding of the language.

Task 3. Exploring Contrastive Learning with SimCLR.

See Task A in the Jupyter notebook.

Task 4. Exploring the Scaling Behaviour of LMs with a Series of Pythia Models.

See Task B in the Jupyter notebook.
