# Appendix of Foundation Model is Efficient Multimodal Multitask Model Selector

**Anonymous Author(s)**
Affiliation
Address
email

## A Related Work

**Transferability Estimation.** Model selection is an important task in transfer learning. To perform model selection efficiently, methods based on designing transferability metrics have been extensively investigated. LEEP [1] pioneers to evaluate the transferability of source models by empirically estimating the joint distribution of pseudo-source labels and the target labels. But it can only handle classification tasks with supervised pre-trained models because the modeling of LEEP relies on the classifier of source models. Recent works propose several improvements over LEEP to overcome the limitation. For example, NLEEP [2] replaces pseudo-source labels with clustering indexes. Moreover, LogME [3], TransRate [4], and PACTran [5] directly measure the compatibility between model features and task labels. Although fast, these metrics can only be used on limited tasks such as classification and regression. This work deals with model selection in multi-task scenarios. We propose EMMS to evaluate the transferability of pre-trained models on various tasks.

**Label Embedding.** Label embedding represents a feature vector of task labels, which can be generated in various ways. The classical approach is to use one-hot encoding to represent the labels as sparse vectors, which is widely used in image classification. Another way is to transform labels into vectors by embedding layers. For example, an RNN module is employed to generate label representation in [6], which is encouraged to be compatible with input data vectors in text classification tasks. In addition, it is also common to treat the labels as words and use techniques such as word2vec [7] or GloVe [8] to learn vector representations of the labels. The main obstacle in the multi-task scenario is how to deal with diverse label formats. In this work, we follow the idea of word embedding and treat task labels as texts, which are then transformed into embeddings by publicly available foundation models [9, 10].

**Foundation Models.** CLIP [9] is the first known foundation model which learns good semantic matching between image and text. The text encoder of CLIP can perform zero-shot label prediction because it encodes rich text concepts of various image objects. By tokenizing multi-modal inputs into homogeneous tokens, recent work on foundation models such as OFA [11] and Uni-Perceiver [12] use a single encoder to learn multi-modal representations. In this work, we utilize the great capacity of foundation models in representing image-text concepts to generate label embedding. It is noteworthy that although foundation models can achieve good performance in various downstream tasks, they may not achieve good zero-shot performance on many tasks[13] and it is still computationally expensive to transfer a large model to the target task [14, 15]. On the contrary, a multi-task model selector can quickly select an optimal moderate-size pre-trained model that can generalize well in target tasks. In this sense, a multi-task model selector is complementary to foundation models.

## B Method

Here we derive in detail the regression with Unified Noisy Label Embeddings that appear in the method section of the text in Sec.B.1 and give complete proof of the convergence of the method in Sec.B.2.

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.

### B.1 Regression with Unified Noisy Label Embeddings

**Setup.** we assume that label embedding $z$ is a linear mapping of the model feature with additive Gaussian noise with a variance of $\sigma_0^2$, as given by $z = z_0 + \epsilon = w^T \hat{x} + \epsilon$ and $\epsilon \sim N(0, \sigma_0^2 I_L)$ where $z_0 = w^T \hat{x}$ is the regression prediction, $w \in \mathbb{R}^{D \times L}$ and $\epsilon$ are regression weights and regression error, respectively, and $I_L$ is a L-by-L identity matrix.

We assume that F-labels $\{z_k\}_{k=1}^K$ obtained from different foundation models are oracles that independently provide noisy estimates of the label embedding $z$. Formally, we have $P(z_k|z) = N(z, \sigma_k^2 I_L)$. Without loss of generality, we assume that $L = 1$

Then the joint probability over noisy labels for a fixed $n$, That is, for given $x^n$, we have:

$$P(z_1^n, \cdots, z_K^n | x^n, w) = \int P(z_1^n, \cdots, z_K^n | z, x^n, w) P(z|x^n, w) dz \tag{1}$$

Due to the independence between $z_k$ and $x$, using the real label $z$, we can rewrite it as:

$$P(z_1^n, \cdots, z_K^n | x^n, w) = \int P(z_1^n, \cdots, z_K^n | z, w) P(z|x^n, w) dz \tag{2}$$

And using the independencies among $z_k$, we have:

$$P(z_1^n, \cdots, z_K^n | z, w) = \prod_{k=1}^{K} P(z_K^n | z, \sigma_1^2, \cdots, \sigma_k^2) = \frac{1}{(2\pi)^{\frac{K}{2}} \prod_{k=1}^{K} \sigma_k} \exp^{-\sum_{k=1}^{K} \frac{(z_k^n - z)^2}{2\sigma_k^2}} \tag{3}$$

Due to $P(z_k|z) = N(z, \sigma_k^2 I_L)$, we can rewrite it as :

$$P(z_1^n, \ldots, z_K^n | x^n, w) = \int \frac{1}{(2\pi)^{\frac{K+1}{2}} \prod_{k=0}^{K} \sigma_k} \exp^{-\sum_{k=1}^{K} \frac{(z_K^n - z)^2}{2\sigma_k^2} - \frac{(z - z_0)^2}{2\sigma_0^2}} dy \tag{4}$$

which can be calculated as :

$$P(z_1^n, \ldots, z_K^n | x^n, w) = A_1 \int e^{-A_2 y^2 + A_3^n y - A_4^n} dz = A_1 \sqrt{\frac{\pi}{A_2}} e^{\frac{(A_3^n)^2}{4A_2} - A_4^n} \tag{5}$$

where $A_1 = \prod_{k=0}^{K} 1/\sigma_k$, $A_2 = \sum_{k=0}^{K} 1/2\sigma_k^2$, $A_3^n = \sum_{k=0}^{K} z_k^n/\sigma_k^2$, and $A_4^n = \sum_{k=0}^{K} (z_k^n)^2/2\sigma_k^2$

Consider the joint probability over all $N$ instances, we have:

$$P(z_1^n, \ldots, z_K^n | X, w) = \prod_{i=1}^{N} A_1 \sqrt{\frac{\pi}{A_2}} e^{\frac{(A_3^n)^2}{4A_2} - A_4^n} \tag{6}$$

where $X \in R^{N \times D}$ denotes the feature matrix, $N$ is the number of data points and $D$ is the number of features.

Then given $N$ data points, the negative log-likelihood is given by

$$-\mathcal{L} = \underbrace{-N \log A_1 + \frac{N}{2} \log A_2}_{\mathcal{L}_1} + \underbrace{\frac{1}{2} \sum_{n=1}^{N} (A_4^n - \frac{(A_3^n)^2}{4A_2})}_{\mathcal{L}_2} + \text{const} \tag{7}$$

where $\mathcal{L}_1$ and $\mathcal{L}_2$ are given by

$$\mathcal{L}_1 = \frac{N}{2} \log \sum_{k=0}^{K} \frac{1}{2\sigma_k^2} + N \sum_{k=0}^{K} \log \sigma_k, \quad \mathcal{L}_2 = \sum_{n=1}^{N} \{ \sum_{k=0}^{K} \frac{(z_k^n)^2}{\sigma_k^2} - \frac{(\sum_{k=0}^{K} z_k^n/\sigma_k^2)^2}{\sum_{k=1}^{K} 1/\sigma_k^2} \} \tag{8}$$

Since $\mathcal{L}_1$ is independent of input data, we focus on $\mathcal{L}_2$. To simplify the notation, we re-denote $\gamma_k = 1/\sigma_k^2$ and $\Gamma = \sum_{k=1}^{K} \gamma_k$. Using this notation, $\mathcal{L}_2$ can be rearranged as:

$$
\begin{aligned}
\mathcal{L}_2 &= \sum_{n=1}^{N} \{\gamma_0 z_0^2 + \sum_{k=1}^{K} \gamma_k (z_k^n)^2 - \frac{(\sum_{k=1}^{K} \gamma_k z_k^n + \gamma_0 z_0)^2}{\Gamma + \gamma_0}\} && (9)\\
&= \sum_{n=1}^{N} \{(\gamma_0 - \frac{\gamma_0^2}{\Gamma + \gamma_0}) z_0^2 - (\frac{2\Gamma\gamma_0}{\Gamma + \gamma_0} \sum_{k=1}^{K} \frac{\gamma_0}{\Gamma} z_k^n) z_0 + \sum_{k=1}^{K} \gamma_k (z_k^n)^2 - (\sum_{k=1}^{K} \gamma_k z_k^n)^2\} && (10)\\
&= \sum_{n=1}^{N} \{\frac{\Gamma\gamma_0}{\Gamma + \gamma_0} (z_0 - \sum_{k=1}^{K} \frac{\gamma_k}{\Gamma} z_k^n)^2 + \sum_{k=1}^{K} \gamma_k (z_k^n)^2 - (1 + \frac{\gamma_0}{\Gamma(\Gamma + \gamma_0)} (\sum_{k=1}^{K} \gamma_k z_k^n)^2\} && (11)\\
&= \sum_{n=1}^{N} \{\frac{\Gamma\gamma_0}{\Gamma + \gamma_0} (w^T \hat{x}^n - \sum_{k=1}^{K} \frac{\gamma_k}{\Gamma} z_k^n)^2 + \sum_{k=1}^{K} \gamma_k (z_k^n)^2 - (1 + \frac{\gamma_0}{\Gamma(\Gamma + \gamma_0)} (\sum_{k=1}^{K} \gamma_k z_k^n)^2\} && (12)
\end{aligned}
$$

Hence, the negative likelihood in Eqn.(7 can be written as

$$
-\mathcal{L} = \frac{\Gamma\gamma_0}{\Gamma + \gamma_0} \underbrace{\{\frac{1}{2} \sum_{i=1}^{N} (w^T \hat{x}^n - \sum_{k=1}^{K} \frac{\gamma_k}{\Gamma} z_k^n)^2\}}_{s(w,t)} + \mathcal{R}(\gamma_k) \tag{13}
$$

where $\mathcal{R}(\gamma_k) = \mathcal{L}_1 + \sum_{k=1}^{K} \gamma_k (z_k^n)^2 - (1 + \frac{\gamma_0}{\Gamma(\Gamma+\gamma_0)} (\sum_{k=1}^{K} \gamma_k z_k^n)^2$. The computational intractability of Eqn.(13) comes from the regularization term $\mathcal{R}(\gamma_k)$. Note that the coefficient $\frac{\Gamma\gamma_0}{\Gamma+\gamma_0} > 0$ and $\sum_{k=1}^{K} \frac{\gamma_k}{\Gamma} = 1$. By removing regularizer $\mathcal{R}(\gamma_k)$ and positive scale parameter $\frac{\Gamma\gamma_0}{\Gamma+\gamma_0}$, the minimization of negative log-likelihood can be approximately treated as a weighted linear square regression, as given by

$$
\min_{w \in \mathbb{R}^{D\times 1}, t \in \triangle^{K-1}} s(w,t) = \frac{1}{2} \|Xw - Zt\|_2^2 \tag{14}
$$

In Eqn.(14), $X \in \mathbb{R}^{N\times D}$ is the data matrix whose $n$-th row is model feature $(\hat{x}^n)^T$, $w \in \mathbb{R}^{D\times 1}$ are weight parameters, $Z \in \mathbb{R}^{N\times K}$ is F-Label matrix whose $k$-th column is the label embedding $z_k$, and $t \in \mathbb{R}^{K\times 1}$ satisfies that $1_K^T t = 1, t \geq 0$ which is a $(K-1)$-D simplex denoted as $\triangle^{K-1}$.

### B.2 Convergence Analysis and Proof Outline

We will prove the convergence property of the function value. Indeed, we demonstrate a stronger condition that the function value decreases after each round of iterations on $w$ and $t$. From the monotone convergence theorem, the convergence can thus be derived. For other convergence properties of alternating minimization, readers can refer to the literature [16], which can be of independent interest.

In the proof, we exploit the smoothness of the function and design a projection gradient descent method with sufficient decrease for the constraint optimization problem. The sufficient decrease in the unconstrained problem is a direct corollary.

**Definition 1.** *A function $f(x): \mathbb{R}^d \to \mathbb{R}$ is said to be $\beta$-smooth with constant $\beta$ if*

$$
|\nabla f(x) - \nabla f(y)| \leq \beta \|x - y\|, \forall x, y \in \mathbb{R}^d.
$$

**Lemma 1.** *Suppose $X$ is the simplex constraint, and $y \in \mathbb{R}^d$, $\Pi$ denotes the projection operator. Then the inequality holds:*

$$
(\Pi_X(y) - x)^T (\Pi_X(y) - y) \leq 0.
$$

*Proof.* For the projection $\Pi_X(y)$, it is a convex optimization problem and can be formulated to

$$
\min_x f(x) = \|x - y\|_2^2,
$$

where $x^T 1 = 1$ and $x > 0$. We denote $x^\star$ as the optimal solution to the problem. For the convex optimization problem, it holds for all $x \in \mathbb{R}^d$ that

$$
\nabla f(x^\star)^T (x^\star - x) \leq 0.
$$

Therefore we can derive

$$2(x^\star - y)^T(x^\star - x) \le 0.$$

Then this lemma is proved. $\square$

**Lemma 2.** *Let $f$ be the $\beta$-smooth function. For any $x, y \in \mathrm{dom}(f)$*

$$\left| f(x) - f(y) - \nabla f(y)^T(x-y) \right| \le \|x-y\|^2.$$

*Proof.*

$$\begin{aligned}
\left| f(x) - f(y) - \nabla f(y)^T(x-y) \right| &= \left| \int_0^1 \nabla f(y + t(x-y))^T(x-y)dt - \nabla f(y)^T(x-y) \right| \\
&\le \int_0^1 \|\nabla f(y+t(x-y)) - \nabla f(y)\| \|x-y\| dt \\
&\le \int_0^1 \beta t \|x-y\|^2 dt = \frac{\beta}{2}\|x-y\|^2.
\end{aligned}$$

The last inequality holds because $f$ is a $\beta$-smooth function. $\square$

**Lemma 3.** *Suppose the function $f$ is the $\beta$-smooth function, and $X$ is the simplex constraint. For any $x, y \in X$, let $x^+ = \Pi_X(x - \frac{1}{\beta}\nabla f(x))$ and $g_X(x) = \beta(x - x^+)$. Then the inequality holds*

$$f(x^+) - f(y) \le g_X(x)^T(x-y) - \frac{1}{2\beta}\|g_X(x)\|^2.$$

*Proof.* Using Lemma. 1, we have

$$(x^+ - (x - \frac{1}{\beta}\nabla f(x)))^T(x^+ - y) \le 0.$$

which is equivalent to

$$\nabla f(x)^T(x^+ - y) \le g_X(x)^T(x^+ - y).$$

By using Lemma. 2 and the fact $f(x^+) - f(y) = f(x^+) - f(x) + f(x) - f(y)$, we have

$$\begin{aligned}
f(x^+) - f(y) &\le \nabla f(x)^T(x^+ - x) + \frac{\beta}{2}\|x^+ - x\|^2 + \nabla f(x)^T(x-y) \\
&= \nabla f(x)^T(x^+ - y) + \frac{1}{2\beta}\|g_X(x)\|^2 \\
&\le g_X(x)^T(x^+ - y) + \frac{1}{2\beta}\|g_X(x)\|^2 \\
&= g_X(x)^T(x^+ - x + x - y) + \frac{1}{2\beta}\|g_X(x)\|^2 \\
&= g_X(x)^T(x^+ - x) + g_X(x)^T(x-y) + \frac{1}{2\beta}\|g_X(x)\|^2 \\
&= g_X(x)^T(x-y) - \frac{1}{\beta}\|g_X(x)\|^2 + \frac{1}{2\beta}\|g_X(x)\|^2 \\
&= g_X(x)^T(x-y) - \frac{1}{2\beta}\|g_X(x)\|^2.
\end{aligned}$$

$\square$

**Theorem 1.** *Suppose $s(w,t) = \frac{1}{2}\|Xw - Zt\|_F^2$ where $X \in \mathbb{R}^{N\times D}$, $Z \in \mathbb{R}^{N\times K}$, $w \in \mathbb{R}^{D\times 1}$ and $t \in \triangle^{K-1}$, the inner loop of $t$ in Algorithm lines* 7 - 10 *decreases after each iteration. Specifically, denote $\beta = 1/\|2Z^TZ\|$ and $t^+ = \Pi_{\triangle^{K-1}}(t - \beta\nabla s(w,t))$. For any $t \in \triangle^{K-1}$, $s(w,t^+) - s(w,t) \le -\frac{1}{2\beta}\|t - t^+\|^2 \le 0$.*

*Proof.* Since we fix $w$ to optimize $t$ at this point, we define $s(t) = s(w,t)$, thus, $\nabla s(t) = -2Z^T(Xw^\star - Zt)$. For any $t_1, t_2 \in \mathrm{dom}(s)$

$$\|\nabla s(t_1) - \nabla s(t_2)\| = \|2Z^TZt_1 - 2Z^TZt_2\| \leq \|2Z^TZ\|\|t_1 - t_2\|.$$

According to the definition 1, it shows that the $f(t)$ is $\beta$-smooth, where $\beta = \|2Z^TZ\|$. We denote $t \in \triangle^{K-1}$ to be the initial point and $t^+$ to be the result of one iteration of $t$, where $t^+ = \Pi_{\triangle^{K-1}}(t - \frac{1}{\beta}\nabla f(t))$. From Lemma 3, we can replace $x^+, y$ and $x$ with $t^+, t$, and $t$, repsectively. In this way, the inequality holds

$$0 \leq s(t^+) \leq s(t) - \frac{1}{2\beta}\|\beta(t - t^+)\|^2 \leq s(t)$$

□

Therefore, according to **Monotone convergence theorem**, the iterative optimization in the algorithm for $t$ is convergent

**Theorem 2.** *Suppose $s(w,t) = \frac{1}{2}\|Xw - Zt\|_2^2$ where $X \in \mathbb{R}^{N\times D}$, $Z \in \mathbb{R}^{N\times K}$, $w \in \mathbb{R}^{D\times 1}$ and $t \in \triangle^{K-1}$, the function value in Algorithm will be convergent. Specifically, denote $w^\star, t^\star$ as the result after one iteration of $w, t$ respectively, we have $0 \leq s(w^\star, t^\star) \leq s(w^\star, t) \leq s(w,t)$.*

*Proof.* In the first step, we denote $t \in \triangle^{K-1}$ is the initial point, then use gradient descent algorithm to calculate $w^\star$. Since the optimization problem for $w$ is a convex optimization problem and use lemma 2, the decreasing property for the gradient part can be derived. That is, for each $w \in \mathbb{R}^{D\times 1}$, we have $s(w^\star, t) \leq s(w,t)$. In the second step, we fix $w$ as $w^\star$, from Theorem 1, we have $s(w^\star, t^\star) \leq s(w^\star, t)$. Therefore, the value of $s(w,t)$ satisfies: $0 \leq s(w^\star, t^\star) \leq s(w^\star, t) \leq s(w,t)$, from **Monotone convergence theorem**, $s(w,t)$ converges to the limiting point. As shown above, the overall convergence of our algorithm is guaranteed. □

# C Experiment

In this section, we present more experimental results in Sec. C.1, detailed descriptions of datasets in Sec. C.2, pre-trained models and baselines in Sec. C.3, and ground-truth scores in Sec. C.4 in various target tasks. More ablation studies can be found in Sec. D.

**Foundation Models.** On image classification, image captioning, referring expression comprehension, and visual question answering, we use foundation models CLIP [9], BERT [17] and GPT-2 [10]. On text question answering, we use foundation models GPT-2 [10], BART [18], and ELECTRA [19]. CLIP was trained on a large dataset of images and their corresponding captions, which can understand the relationship between images and text. BERT is a pre-trained language model that can understand and generate natural language. GPT-2 was trained on a large corpus of text and can be fine-tuned for specific tasks such as text completion and text summarization. Bart is a sequence-to-sequence model, which is both auto-regressive and bidirectional. Electra is a different type of language model that key idea is to pre-train a generator model to produce fake data and shows promising results in various NLP tasks.

**Interpretation of weighted Kendall's tau.** The Kendall's $\tau$ represents the ratio of concordant pairs minus discordant pairs when enumerating all pairs of $\{T_m\}_{m=1}^M$ and $\{G_m\}_{m=1}^M$ as given by

$$\tau = \frac{2}{M(M-1)} \sum_{1\leq i<j\leq M} \mathrm{sgn}(G_i - G_j)\mathrm{sgn}(T_i - T_j) \tag{15}$$

where $\mathrm{sgn}(x)$ returns $-1$ if $x < 0$ and 1 otherwise. In this work, a weighted version of Kendall's $\tau$, denoted as $\tau_w$, is employed to assess transferability metrics considering that a top-performing model is always preferred for target tasks in transfer learning. In principle, a larger $\tau_w$ implies the transferability metric can rank pre-trained models better. And if a metric can rank top-performing models better, $\tau_w$ would be also larger. We also use other measurements to assess the performance of transferability metrics in Table 9 of Sec. D.

Table 1: Comparison of different transferability metrics on VQA models in rank correlation $\tau_w$ with the ground truth and the wall-clock time. The LogME denotes using LogME with F-Label. Our proposed EMMS performs better than PACTran head over 3 target tasks with much less time.

| | DAQUAR | COCO-QA | CLEVR | DAQUAR | COCO-QA | CLEVR |
|---|---|---|---|---|---|---|
| | Weighted Kendall's tau $\tau_w$ | | | Wall-Clock Time (s) | | |
| LogME | 0.586 | 0.591 | 0.281 | 116.72 | 716.35 | 4665.06 |
| PACTran(Dir) | 0.671 | 0.296 | 0.347 | 633.16 | 1169.91 | 428.03 |
| PACTran(Gam) | 0.595 | 0.419 | 0.319 | 614.23 | 1061.72 | 428.49 |
| PACTran(Gau) | 0.478 | 0.378 | 0.415 | 637.39 | 1075.88 | 418.34 |
| EMMS | **0.712** | **0.812** | **0.804** | **50.54** | **263.72** | **274.56** |

Table 2: Comparison of different transferability metrics on CNN models regarding $\tau_w$ and the wall-clock time where EMMS(One) denotes EMMS with the one-hot label. Our proposed EMMS achieves the best transfer-ability assessment over 11 target tasks and exhibits higher efficiency than NLEEP.

| | Aircraft | Caltech | Cars | CF-10 | CF-100 | DTD | Flowers | Food | Pets | SUN | VOC | Avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Weighted Kendall's tau $\tau_w$ | | | | | | | | | | | |
| LEEP | -0.234 | 0.605 | 0.367 | 0.824 | 0.677 | 0.486 | -0.243 | 0.491 | 0.389 | 0.722 | 0.371 | 0.409 |
| LogME | 0.506 | 0.435 | **0.576** | 0.852 | 0.677 | 0.647 | 0.111 | 0.385 | 0.411 | 0.487 | 0.669 | 0.509 |
| NLEEP | -0.41 | **0.614** | 0.265 | 0.818 | 0.805 | **0.796** | 0.122 | 0.214 | **0.753** | **0.925** | 0.687 | 0.611 |
| TransRate | 0.172 | 0.269 | 0.172 | 0.513 | 0.197 | 0.336 | -0.176 | -0.071 | 0.173 | 0.612 | 0.651 | 0.236 |
| EMMS(One) | 0.481 | 0.546 | 0.304 | 0.963 | 0.804 | 0.701 | 0.498 | 0.588 | 0.574 | 0.638 | 0.707 | 0.618 |
| EMMS | **0.556** | 0.562 | 0.565 | **0.963** | **0.840** | 0.720 | **0.498** | **0.608** | 0.604 | 0.667 | **0.735** | **0.664** |
| | Wall-Clock Time (s) | | | | | | | | | | | |
| LEEP | 5.1 | 4.9 | 8.3 | 22.3 | 23.8 | 3.5 | 3.8 | 37.1 | 3.9 | 21.1 | 4.8 | 10.4 |
| LogME | 30.36 | 31.24 | 56.26 | 90.34 | 188.3 | 15.16 | 22.27 | 334.53 | 17.55 | 180.01 | 20.05 | 289.64 |
| NLEEP | 253.8 | 488.7 | 973.8 | 1.1e4 | 1.7e4 | 146.0 | 294.0 | 2.0e4 | 580.8 | 8.6e3 | 678.8 | 5455.9 |
| TransRate | 147.90 | 163.41 | 300.29 | 65.25 | 193.64 | 75.48 | 166.24 | 195.92 | 60.53 | 430.33 | 18.72 | 165.24 |
| EMMS(One) | 17.43 | 20.53 | 35.22 | 70.01 | 78.24 | 12.75 | 18.04 | 116.23 | 15.04 | 70.98 | 18.42 | 42.99 |
| EMMS | 65.85 | 63.49 | 79.79 | 245.49 | 295.37 | 46.38 | 63.52 | 417.80 | 59.64 | 173.59 | 64.60 | 143.2 |

## C.1 More Experiments

### C.1.1 Performance on Visual Question Answering

To further demonstrate the generality of EMMS in multi-model tasks, we show how EMMS can work for VQA. We follow previous practice ( [5]) which treats VQA as a classification task (vocab-based VQA). That is, we construct a vocabulary based on the top answers in the training sets and classify them into some of those labels. The models to be selected and the architecture is the same as in the image captioning .

**Performance and wall-clock time comparison.** As shown in Table.1, EMMS is clearly ahead of PACTran in terms of results and time, proving that EMMS has the ability to handle multi-modal tasks very well. We can find that EMMS outperforms PACTran on all datasets. In particular, EMMS achieves 93.8% and 93.7% gain over PACTran on the COCO-QA and CLEVR datasets with rank correlation $\tau_w$ while reducing time consumption by 75.1% and 34.3% respectively compared to Pactran. This indicates that EMMS performs well on both ordinary VQA datasets(DAQUAR, COCO-QA) as well as VQA datasets(CLEVR) that focus on inference capabilities.

### C.1.2 Performance on Image Classification with CNN Models

**Performance and wall-clock time comparison.** We compare EMMS with previous LEEP, NLEEP, LogME, and TransRate. As shown in Table.2, our EMMS achieve the best average $\tau_w$ on 11 target datasets and the best $\tau_w$ on 6 target datasets. Compared to NLEEP, which is the most effective other than EMMS, we have almost 1/40 of the time of NLEEP.

### C.2 Descriptions of Datasets

#### C.2.1 Image Classification

For image classification, we adopt 11 classification benchmarks , including FGVC Aircraft [20], Caltech-101 [21], Stanford Cars [22], CIFAR-10 [23], CIFAR-100 [23], DTD [24], Oxford 102 Flowers [25], Food-101 [26], Oxford-IIIT Pets [27], SUN397 [28], and VOC2007 [29]. These datasets cover a broad range of classification tasks, which include scene, texture, and coarse/fine-grained image classification, which are widely used in transfer learning. In particular, CF10 and VOC2007 are typical coarse-grained classification datasets, Aircraft, and Cars are typical fine-grained classification datasets, and CF100 contains both coarse- and fine-grained classifications.

#### C.2.2 Image Captioning

For image captioning, We use Flickr8k [30], Flickr30k [31], FlickrStyle10K-Humor [32], FlickrStyle10K-Romantic [32] and RSICD [33]. Among them, Flickr8k and Flickr30k have commonly used image captioning datasets for natural images and have no emotional color; RSICD is a commonly used image captioning dataset in remote sensing; Flickr10k-H and Flickr10k-R are also image captioning datasets for natural images, but their images are depicted with humorous and romantic emotional colors, respectively.

#### C.2.3 Visual Question Answering

For visual question answering, we apply COCOQA [34], DAQUAR [35] and CLEVR [36].Among them, DAQUAR is an early VQA dataset on real images; CLEVR is a synthetic dataset, which is a visual scene composed of some simple geometric shapes, focusing on evaluating the inference ability of VQA models; the questions and answers of COCO-QA are generated by NLP algorithms, and the images are from the COCO dataset, which is also a commonly used VQA dataset.

#### C.2.4 Text Question Answering

For text question answering, we separately use SQuAD1.1 [37] ,SQuAD2.0 [38], which are collections of question-answer pairs derived from Wikipedia articles and are widely used in text question answer.

#### C.2.5 Referring Expression Comprehension

For referring expression comprehension, we separately use RefCOCO [39], RefCOCO+ [39] and RefCOCOg [40].Specifically, RefCOCO includes instances where there is only one object of its kind in the image, while RefCOCO+ includes instances where multiple objects of the same kind exist in the image.

### C.3 Pre-trained Models and Baselines

#### C.3.1 Image Classification

**Pre-trained Models.** For **CNN-based** models, We select 11 widely-used CNN models including ResNet-34 [41], ResNet-50 [41], ResNet-101 [41], ResNet-152 [41], DenseNet-121 [42], DenseNet-169 [42], DenseNet-201 [42], MNet-A1 [43], MobileNetV2 [44], GoogleNet [45], and InceptionV3 [46]. All these models are trained on ImageNet dataset [47], which are widely used within the field of migration learning. For **ViT-based** models, we collect 10 ViT models including ViT-T [48], ViT-S [48], ViT-B [48], DINO-S [49], MoCov3-S [50] , PVTv2-B2 [51], PVT-T [51], PVT-S [51], PVT-M [51], and Swin-T [52], which are widely used in various vision tasks. Besides, we append EMMS with one-hot label, which degenerates to a linear regression whose label is the one-hot vector. We fine-tune these models on the 11 target datasets to obtain the ground truth.

**Comparison Baselines.** Here we use some of the latest methods as baselines, including LEEP [1], NLEEP [2], LogME [3], and TransRate [4], which have been experimented with model selection on image classification tasks.

### C.3.2 Image Captioning

**Pre-trained Models.** We use a classic and effective image captioning model architecture, which contains an image encoder and a language encoder to extract the features of the image and the corresponding caption, then fuses the image feature and the text feature and input it to the classifier. We aim to choose the best combination of image encoder and language encoder. Besides, We finetune each model in COCO Caption [53] and use these as the pre-trained models.

Specifically, We separately use ViT-B [48],Swin-B [52], Swinv2-B [54] as image encoder and Bert [17], Roberta [55], Bart [18] as language encoder, and use VisionEncoderDecoderModel from HuggingFace as the model architecture. Following the setting in PACTran [5], We finetune the model in COCO Caption [53] and use these as the pre-trained models. Following common practice( [56]) , we treat image captioning as a vocab-based classification task. That is we use a vocabulary and classify the caption into the index of some words in the vocabulary. Afterward, training is done according to the classification task criteria.

**Comparison Baselines.** In this common setup, each caption is converted to a matrix $Y \in R^{L \times N}$, where $L$ denotes the length of the caption after padding or truncation and $N$ denotes the size of the vocabulary, and each row in the matrix is a one-hot vector. Since $N$ is generally very large, Existing model selection metrics do not scale to this case due to the huge amount of time spent. The only baseline we use is to model the fused feature with F-label using LogME since only LogME can handle the regression task. Here we calculate the average $\tau_w$ and time of it with $K$ single F-label from $K$ foundation models we use respectively.

### C.3.3 Visual Question Answering

**Pre-trained Models.** The model architecture and the model selection settings are the same as in the image captioning, Following the setting in PACTran [5], here we use the model after finetune on VQA-v2 [56] as the pre-trained model waiting for selection and treat VQA as a vocab-based classification task.

**Comparison Baselines.** Here we calculate the average $\tau_w$ and time of it with $K$ single F-label from $K$ foundation models we use respectively. And in addition to that, the three methods proposed in PACTran [5] are added here, which are the only methods currently applied to VQA tasks.

### C.3.4 Text Question Answering

**Pre-trained Models.** The selected models include BERT-Large [17], RoBERTa-Large [55], XLNet-Large [57], DeBERTa [58] (XLarge), DeBERTa-V2 [58] (XLarge and XXLarge), DeBERTa-V3 [59] (Base, Small, XSmall). More specifically, we simultaneously input the question and passage into the aforementioned models, utilizing the distinctive symbol `[SEP]` to demarcate them. By stacking the predicted head onto each model, we could further fine-tune the model such that it can predict the start and end positions of the answer within the passage. This is achieved by using two binary classifiers, where one is dedicated to identifying the start position and the other to pinpointing the end.

**Comparison Baselines.** Here we calculate the average $\tau_w$ and time of it with F-labels from $K$ foundation models respectively.

### C.3.5 Referring Expression Comprehension

**Pre-trained Models.** The candidate multi-modal architectures considered for REC task incorporate Blip [60], ALBEF [61], CLIP [9] (ViT-B-32, ViT-B-16, ViT-L-14, ViT-L-14-336, RN50), OFA [62] (Base, Large, Huge). In practice, we respectively extract the visual and textual representations from each of these models and feed them into a multi-modal interaction module followed by a stacked detection head, and further fine-tune the model to generate the ground truth of model selection.

**Comparison Baselines.** Here we calculate the average $\tau_w$ and time of LogME with $K$ single F-label from $K$ foundation models we use respectively.

Table 3: The fine-tuning accuracy of supervised CNN models on 11 target tasks.

| | Aircraft | Caltech | Cars | CF-10 | CF-100 | DTD | Flowers | Food | Pets | SUN | VOC |
|---|---|---|---|---|---|---|---|---|---|---|---|
| ResNet-34 | 84.06 | 91.15 | 88.63 | 96.12 | 81.94 | 72.96 | 95.2 | 81.99 | 93.5 | 61.02 | 84.6 |
| ResNet-50 | 84.64 | 91.98 | 89.09 | 96.28 | 82.8 | 74.72 | 96.26 | 84.45 | 93.88 | 63.54 | 85.8 |
| ResNet-101 | 85.53 | 92.38 | 89.47 | 97.39 | 84.88 | 74.8 | 96.53 | 85.58 | 93.92 | 63.76 | 85.68 |
| ResNet-152 | 86.29 | 93.1 | 89.88 | 97.53 | 85.66 | 76.44 | 96.86 | 86.28 | 94.42 | 64.82 | 86.32 |
| DenseNet-121 | 84.66 | 91.5 | 89.34 | 96.45 | 82.75 | 74.18 | 97.02 | 84.99 | 93.07 | 63.26 | 85.28 |
| DenseNet-169 | 84.19 | 92.51 | 89.02 | 96.77 | 84.26 | 74.72 | 97.32 | 85.84 | 93.62 | 64.1 | 85.77 |
| DenseNet-201 | 85.38 | 93.14 | 89.44 | 97.02 | 84.88 | 76.04 | 97.1 | 86.71 | 94.03 | 64.57 | 85.67 |
| MNet-A1 | 66.48 | 89.34 | 72.58 | 92.59 | 72.04 | 70.12 | 95.39 | 71.35 | 91.08 | 56.56 | 81.06 |
| MobileNetV2 | 79.68 | 88.64 | 86.44 | 94.74 | 78.11 | 71.72 | 96.2 | 81.12 | 91.28 | 60.29 | 82.8 |
| Googlenet | 80.32 | 90.85 | 87.76 | 95.54 | 79.84 | 72.53 | 95.76 | 79.3 | 91.38 | 59.89 | 82.58 |
| InceptionV3 | 80.15 | 92.75 | 87.74 | 96.18 | 81.49 | 72.85 | 95.73 | 81.76 | 92.14 | 59.98 | 83.84 |

Table 4: The fine-tuning accuracy of vision transformer models on 11 target tasks.

| | Aircraft | Caltech | Cars | CF-10 | CF-100 | DTD | Flowers | Food | Pets | SUN | VOC |
|---|---|---|---|---|---|---|---|---|---|---|---|
| ViT-T | 71.26 | 89.39 | 82.09 | 96.52 | 81.58 | 71.86 | 95.5 | 81.96 | 91.44 | 58.4 | 83.1 |
| ViT-S | 73.12 | 92.7 | 86.72 | 97.69 | 86.62 | 75.08 | 96.79 | 86.26 | 94.02 | 64.76 | 86.62 |
| ViT-B | 78.39 | 93.47 | 89.26 | 98.56 | 89.96 | 77.66 | 97.98 | 88.96 | 94.61 | 68.62 | 87.88 |
| PVTv2-B2 | 84.14 | 93.13 | 90.6 | 97.96 | 88.24 | 77.16 | 97.89 | 88.67 | 93.86 | 66.44 | 86.44 |
| PVT-T | 69.76 | 90.04 | 84.1 | 94.87 | 75.26 | 72.92 | 95.8 | 83.78 | 91.48 | 61.86 | 84.6 |
| PVT-S | 75.2 | 93.02 | 87.61 | 97.34 | 86.2 | 75.77 | 97.32 | 86.98 | 94.13 | 65.78 | 86.62 |
| PVT-M | 76.7 | 93.75 | 87.66 | 97.93 | 87.36 | 77.1 | 97.36 | 85.56 | 94.48 | 67.22 | 87.36 |
| Swin-T | 81.9 | 91.9 | 88.93 | 97.34 | 85.97 | 77.04 | 97.4 | 86.67 | 94.5 | 65.51 | 87.54 |
| MoCov3-S | 76.04 | 89.84 | 82.18 | 97.92 | 85.84 | 71.88 | 93.89 | 82.84 | 90.44 | 60.6 | 81.84 |
| DINO-S | 72.18 | 86.76 | 79.81 | 97.96 | 85.66 | 75.96 | 95.96 | 85.69 | 92.59 | 64.14 | 84.8 |

## C.4 Fine-tuning Score on Various Target Tasks

### C.4.1 Image Classification

**Fine-tuning Details.** The ground truth of the problem of pre-trained model ranking is to fine-tune all pre-trained models with a hyper-parameters sweep on target datasets. Given the model and the target dataset, two of the most important parameters would be learning rate and weight decay in optimizing the model [63]. Therefore, we carefully fine-tune pre-trained models with a grid search of learning rate in $\{1e-1, 1e-2, 1e-3, 1e-4\}$ and weight decay in $\{1e-3, 1e-4, 1e-5, 1e-6, 0\}$. And using SGD optimizer. After determining the best hyper-parameters candidate, we fine-tune the pre-trained model on the target dataset with the candidate and then obtain the test accuracy as the ground truth. We use a Tesla V100 with a batch size of $128$ to perform finetuning. All input images are resized to $224 \times 224$. To avoid random error, we repeat the above fine-tuning procedure three times and take an average to obtain the final fine-tuning accuracy. For reference, we list the fine-tuning accuracy of supervised CNN models in Table.3, and vision transformer models in Table 4, respectively.

### C.4.2 Image Captioning and Visual Question Answering

**Fine-tuning Details.** The setting of finetune here is approximately the same as in image classification. We carefully fine-tune pre-trained models with a grid search of learning rate in $\{1e-4, 1e-5, 1e-6\}$ and weight decay in $\{1e-4, 1e-5, 1e-6\}$. And using AdamW optimizer. After determining the best hyper-parameters candidate, we fine-tune the pre-trained model on the target dataset with the candidate and then obtain the test BLEU-4 and accuracy as the ground truth. However, since Flickr10k-H and Flickr10k-R do not provide a test set, we use a 6:1 ratio to divide the original training set of 7000 images into a training set and a test set. For visual question answering, Due to the lack of a test set for CLEVR dataset, we also assign its training set as training set and test set in the ratio of 6:1. We use an Nvidia A100 with a batch size of $64$ to perform finetuning. All input images are resized to $224 \times 224$. To avoid random error, we repeat the above fine-tuning procedure three times and take an average to obtain the final fine-tuning accuracy. For inference, We use BLEU-4 as the score for the model with image captioning and accurarcy as the score for the model with VQA. we list result of image captioning models in Table.5, and visual question answering models in Table 6, respectively.

Table 5: The fine-tuning BLEU-4 of image captioning models on 5 target tasks.

| | F8k | F30k | RSD | F10k-H | F10k-R |
|---|---|---|---|---|---|
| Vit-Bert | 18.51 | 26.65 | 31.39 | 5.31 | 5.18 |
| Vit-Roberta | 20.53 | 23.70 | 29.92 | 5.88 | 5.48 |
| Vit-Bart | 21.90 | 25.13 | 31.35 | 5.75 | 5.53 |
| Swinvit-Bert | 22.91 | 26.61 | 33.54 | 6.24 | 5.67 |
| Swinvit-Roberta | 23.99 | 28.84 | 33.07 | 7.11 | 5.49 |
| Swinvit-Bart | 24.68 | 28.03 | 32.99 | 6.10 | 5.95 |
| Swin2vit-Bert | 25.69 | 31.33 | 35.45 | 5.86 | 5.49 |
| Swin2vit-Roberta | 23.40 | 28.81 | 36.22 | 6.80 | 7.13 |
| Swin2vit-Bart | 26.24 | 30.35 | 34.72 | 7.90 | 5.96 |

Table 6: The fine-tuning accuracy of visual question answering models on 3 target tasks.

| | DAQUAR | COCO-QA | CLEVR |
|---|---|---|---|
| Vit-Bert | 25.01 | 55.11 | 59.29 |
| Vit-Roberta | 26.38 | 57.30 | 62.80 |
| Vit-Bart | 26.30 | 59.60 | 64.98 |
| Swinvit-Bert | 28.05 | 61.72 | 68.25 |
| Swinvit-Roberta | 27.75 | 62.81 | 66.09 |
| Swinvit-Bart | 27.06 | 60.62 | 67.17 |
| Swin2vit-Bert | 26.45 | 63.1 | 67.4 |
| Swin2vit-Roberta | 26.33 | 66.54 | 65.91 |
| Swin2vit-Bart | 26.25 | 64.4 | 70.34 |

### C.4.3 Text Question Answering

**Fine-tuning Details.** The accuracy of most models in TQA is provided by DeBERTa [58, 59], except for DeBERTa-V3 [59](Base, Small, XSmall). Following the setting of Bert [17], we finetune these models with a batch size of 24 for 2 epochs. We use AdamW optimizer with an initial learning rate of $3e-5$, polynomial decay. The Dev F1 score is used for pre-trained model ranking. All experiments are implemented on an NVIDIA Tesla A100 GPU. The finetune accuracy is shown in Table 7.

Table 7: The standard metric the Dev F1 score of text question answering models on 2 target tasks.

| | SQu1.1 | SQu2.0 |
|---|---|---|
| BERT-Large | 90.9 | 81.8 |
| RoBERTa-Large | 94.6 | 89.4 |
| XLNet-Large | 95.1 | 90.6 |
| DeBERTa-Large | 95.5 | 90.7 |
| DeBERTa-V2-XLarge | 95.8 | 91.4 |
| DeBERTa-V2-XXLarge | 96.1 | 92.2 |
| DeBERTa-V3-Base | 93.9 | 88.4 |
| DeBERTa-V3-Small | 89.8 | 82.9 |
| DeBERTa-V3-XSmall | 91.5 | 84.8 |

Table 8: The standard metric Acc@0.5 of referring expression comprehension models on 3 target tasks.

| | RefCOCO | RefCOCO+ | RefCOCOg |
|---|---|---|---|
| Blip | 88.67 | 84.68 | 85.08 |
| ALBEF | 87.98 | 82.20 | 82.89 |
| CLIP-ViT-B-32 | 83.20 | 74.56 | 76.98 |
| CLIP-ViT-B-16 | 87.35 | 80.12 | 81.69 |
| CLIP-ViT-L-14 | 90.17 | 86.09 | 87.13 |
| CLIP-ViT-L-14-336 | 91.67 | 87.60 | 87.89 |
| CLIP-RN50 | 84.69 | 76.72 | 79.39 |
| OFA-Base | 88.48 | 81.39 | 82.29 |
| OFA-Large | 90.05 | 85.80 | 85.89 |
| OFA-Huge | 92.04 | 87.86 | 88.07 |

### C.4.4 Referring Expression Comprehension

**Fine-tuning Details.** For referring expression comprehension, the standard metric Acc@0.5 on the validation set is used as the ground truth. For finetuning, we use a batch size of 128 with a resolution of $512 \times 512$ for each image. We finetune the models on each dataset for 12 epochs with a learning rate of $\{3e-5, 5e-5\}$ and weight decay in $\{1e-3, 1e-5\}$ using Adam optimizer. The best performance on the validation set for each task is reported among these hyper-parameters. Table 8 shows the performance of referring expression comprehension models.

Table 10: The effect of Label Embedding in EMMS. Three variants of EMMS are considered: (1) EMMS with one-hot label; (2) EMMS with single F-Label; (3) EMMS with multiple F-Labels which is the original. We see that label embedding brings some performance improvement to EMMS

| | Aircraft | Caltech | Cars | CF-10 | CF-100 | DTD | Flowers | Food | Pets | SUN | VOC | Avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | Weighted Kendall's tau $\tau_w$ | | | | | | |
| (1) | 0.481 | 0.546 | 0.304 | 0.963 | 0.804 | 0.701 | 0.498 | 0.588 | 0.574 | 0.638 | 0.707 | 0.618 |
| (2) | 0.531 | 0.562 | 0.426 | 0.952 | 0.804 | 0.720 | 0.481 | 0.602 | 0.535 | 0.667 | 0.726 | 0.636 |
| (3) | **0.556** | **0.562** | **0.565** | **0.963** | **0.840** | **0.720** | **0.498** | **0.608** | **0.604** | **0.667** | **0.735** | **0.664** |

## D More Ablation Analysis

**The Efftiveness of EMMS under Various Measurements.** In addition to weighted Kendall's tau, we employ various other measures to evaluate our EMMS. These include Kendall's tau ($\tau$), Pearson's correlation ($r$), weighted Pearson's correlation ($r_w$), and top-$k$ relative accuracy, denoted as Rel@$k$, which represents the ratio between the best fine-tuning accuracy achieved on the downstream task using the top-k ranked models and the best fine-tuning precision achieved with all models. We test

Table 9: EMMS under different measurements of transferability assessment. The results are obtained on Flickr8k and RSICD datasets with image captioning task and Aircraft and DTD datasets with image classification task with ViT-based models. EMMS outperforms LogME and other baselines under various measures.

| Data | Method | Rel@1 | Rel@3 | $r$ | $r_w$ | $\tau$ | $\tau_w$ | Data | Method | Rel@1 | Rel@3 | $r$ | $r_w$ | $\tau$ | $\tau_w$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| F8k | LogME | 0.928 | **1.0** | 0.735 | 0.799 | 0.537 | 0.483 | RSD | LogME | 0.957 | **1.0** | 0.727 | 0.708 | 0.518 | 0.501 |
| | EMMS | **1.0** | **1.0** | **0.741** | **0.823** | **0.667** | **0.660** | | EMMS | **1.0** | **1.0** | **0.783** | **0.765** | **0.611** | **0.705** |
| Aircraft | LogME | 0.852 | 0.993 | 0.407 | 0.060 | 0.378 | 0.299 | DTD | LogME | **0.992** | **1.0** | 0.641 | 0.694 | 0.556 | 0.569 |
| | TransRate | **0.926** | **0.967** | 0.457 | 0.499 | 0.289 | 0.244 | | TransRate | **0.992** | **1.0** | 0.607 | 0.676 | 0.422 | 0.533 |
| | EMMS | **0.926** | **0.967** | **0.622** | **0.608** | **0.511** | **0.481** | | EMMS | **0.992** | **1.0** | **0.704** | **0.785** | **0.644** | **0.621** |

the robustness of our transferability metrics to different measurements on the Flickr8k and RSICD datasets for image captioning tasks, as shown in Table 9. Our EMMS consistently outperforms the previous transferability metric, including LogME and TransRate. Under the aforementioned measurements, demonstrating the superiority of our EMMS.

**The Effect of Label Embedding** In some multimodal tasks or text tasks, including image captioning or text question answering. Label emebdding directly affects the applicability of existing model selection metric to these tasks. In addition, even in classification tasks, the use of F-Label can also bring improvements in results. Here we focus on the comparison between label embedding and direct one-hot vectors for image classification tasks in CNN-based models. As shown in Table 10, the use of F-Label can bring performance improvement compared to One-Hot vector, the average $\tau_w$ increase from 0.618 to 0.636; furthermore, the use of multiple F-Label also brings some improvement compared to the average of single F-Label with $\tau_w$ increasing from 0.636 to 0.664.

**The Effect of Computational Speedup.** Here we experimentally demonstrate the effect of our accelerated algorithm. As shown in Table 11, the algorithm is similar to the in-accelerated version in terms of results, but much shorter in terms of the wall-clock time.

**The Wall-clock Time of Label Embedding.** For classification tasks, since the maximum number of categories is often only a few hundred, Label Embedding is very fast. Here we focus on documenting the time required for multimodal tasks, e.g. image captioning, text question answering, and referring expression comprehension, where label embedding is more time-consuming. For each task, we use 8 Nvidia A100 GPUs for label embedding, with a batch size of 512 for each GPU. The running time of label embedding for image captioning, text question answering, and referring expression comprehension is shown in Table 12.

Table 11: The effect of computational speedup in image classification with ViT models. We can see that the accelerated version of the algorithm achieves a significant reduction in time while guaranteeing results. Two variants of EMMS are considered: (1) EMMS with normal algorithm; (2) EMMS with fast algorithm;

| | Aircraft | Caltech | Cars | CF-10 | CF-100 | DTD | Flowers | Food | Pets | SUN | VOC | Avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Weighted Kendall's tau $\tau_w$ | | | | | | | | | | | |
| (1) | **0.564** | **0.463** | 0.706 | 0.718 | 0.745 | 0.589 | **0.592** | 0.531 | **0.755** | 0.532 | 0.730 | 0.629 |
| (2) | 0.481 | 0.444 | **0.706** | **0.718** | **0.745** | **0.621** | 0.562 | **0.673** | 0.740 | **0.619** | **0.730** | **0.639** |
| | Wall-Clock Time (s) | | | | | | | | | | | |
| (1) | 102.06 | 114.72 | 177.25 | 718.34 | 724.5 | 50.24 | 87.28 | 944.57 | 83.37 | 336.92 | 104.9 | 313.10 |
| (2) | **21.31** | **17.23** | **28.06** | **154.61** | **182.11** | **13.87** | **15.95** | **265.99** | **17.93** | **63.86** | **16.63** | **72.55** |

Table 12: The wall-clock time (s) of label embedding in image captioning on 5 target tasks, text question answering on 2 target tasks, and referring expression comprehension on 3 target tasks,respectively.

| Task | Image Captioning | | | | | Text QA | | Referring EC | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Dataset | F8k | F30k | RSD | F10k-H | F10k-R | SQuAD1.1 | SQuAD2.0 | RefCOCO | RefCOCO+ | RefCOCOg |
| Time | 14.56 | 89.31 | 18.92 | 3.37 | 3.13 | 35.67 | 53.87 | 49.19 | 48.88 | 31.63 |

## References


[1] Cuong Nguyen, Tal Hassner, Matthias Seeger, and Cedric Archambeau. Leep: A new measure to evaluate transferability of learned representations. In *International Conference on Machine Learning*, pages 7294–7305. PMLR, 2020.

[2] Yandong Li, Xuhui Jia, Ruoxin Sang, Yukun Zhu, Bradley Green, Liqiang Wang, and Boqing Gong. Ranking neural checkpoints. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2663–2673, 2021.

[3] Kaichao You, Yong Liu, Jianmin Wang, and Mingsheng Long. Logme: Practical assessment of pre-trained models for transfer learning. In *International Conference on Machine Learning*, pages 12133–12143. PMLR, 2021.

[4] Long-Kai Huang, Junzhou Huang, Yu Rong, Qiang Yang, and Ying Wei. Frustratingly easy transferability estimation. In *International Conference on Machine Learning*, pages 9201–9225. PMLR, 2022.

[5] Nan Ding, Xi Chen, Tomer Levinboim, Soravit Changpinyo, and Radu Soricut. Pactran: Pac-bayesian metrics for estimating the transferability of pretrained models to classification tasks. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXIV*, pages 252–268. Springer, 2022.

[6] Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Interspeech*, volume 2, pages 1045–1048. Makuhari, 2010.

[7] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[8] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[9] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[10] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[11] Peng Wang, An Yang, Rui Men, Junyang Lin, Shuai Bai, Zhikang Li, Jianxin Ma, Chang Zhou, Jingren Zhou, and Hongxia Yang. Unifying architectures, tasks, and modalities through a simple sequence-to-sequence learning framework. *arXiv preprint arXiv:2202.03052*, 2022.

[12] Xizhou Zhu, Jinguo Zhu, Hao Li, Xiaoshi Wu, Hongsheng Li, Xiaohua Wang, and Jifeng Dai. Uni-perceiver: Pre-training unified architecture for generic perception for zero-shot and few-shot tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16804–16815, 2022.

[13] Chengzhi Mao, Scott Geng, Junfeng Yang, Xin Wang, and Carl Vondrick. Understanding zero-shot adversarial robustness for large-scale models. *arXiv preprint arXiv:2212.07016*, 2022.

[14] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR, 2019.

[15] Peng Gao, Shijie Geng, Renrui Zhang, Teli Ma, Rongyao Fang, Yongfeng Zhang, Hongsheng Li, and Yu Qiao. Clip-adapter: Better vision-language models with feature adapters. *arXiv preprint arXiv:2110.04544*, 2021.

[16] Charles L Byrne. Alternating minimization and alternating projection algorithms: A tutorial. *Sciences New York*, pages 1–41, 2011.

[17] 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.

[18] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

[19] Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*, 2020.

[20] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013.

[21] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *2004 conference on computer vision and pattern recognition workshop*, pages 178–178. IEEE, 2004.

[22] Jonathan Krause, Jia Deng, Michael Stark, and Li Fei-Fei. Collecting a large-scale dataset of fine-grained cars. 2013.

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

[24] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3606–3613, 2014.

[25] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729. IEEE, 2008.

[26] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101–mining discriminative components with random forests. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VI 13*, pages 446–461. Springer, 2014.

[27] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016 ieee conference on computer vision and pattern recognition (cvpr). *Las Vegas, NV, USA*, 1:770–78, 2016.

[28] Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 3485–3492. IEEE, 2010.

[29] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88:303–338, 2010.

[30] Cyrus Rashtchian, Peter Young, Micah Hodosh, and Julia Hockenmaier. Collecting image annotations using amazon's mechanical turk. In *Proceedings of the NAACL HLT 2010 workshop on creating speech and language data with Amazon's Mechanical Turk*, pages 139–147, 2010.

[31] Bryan A Plummer, Liwei Wang, Chris M Cervantes, Juan C Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *Proceedings of the IEEE international conference on computer vision*, pages 2641–2649, 2015.

[32] Chuang Gan, Zhe Gan, Xiaodong He, Jianfeng Gao, and Li Deng. Stylenet: Generating attractive visual captions with styles. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3137–3146, 2017.

[33] Xiaoqiang Lu, Binqiang Wang, Xiangtao Zheng, and Xuelong Li. Exploring models and data for remote sensing image caption generation. *IEEE Transactions on Geoscience and Remote Sensing*, 56(4):2183–2195, 2017.

[34] Mengye Ren, Ryan Kiros, and Richard Zemel. Exploring models and data for image question answering. *Advances in neural information processing systems*, 28, 2015.

[35] Mateusz Malinowski and Mario Fritz. A multi-world approach to question answering about real-world scenes based on uncertain input. *Advances in neural information processing systems*, 27, 2014.

[36] Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2901–2910, 2017.

[37] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.

[38] Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for squad. *arXiv preprint arXiv:1806.03822*, 2018.

[39] Licheng Yu, Patrick Poirson, Shan Yang, Alexander C Berg, and Tamara L Berg. Modeling context in referring expressions. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*, pages 69–85. Springer, 2016.

[40] Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L Yuille, and Kevin Murphy. Generation and comprehension of unambiguous object descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 11–20, 2016.

[41] 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.

[42] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[43] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. Mnasnet: Platform-aware neural architecture search for mobile. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2820–2828, 2019.

[44] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[45] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[46] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[47] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[48] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[49] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021.

[50] X Chen, S Xie, and K He. An empirical study of training self-supervised visual transformers. arxiv e-prints. *arXiv preprint arXiv:2104.02057*, 2021.

[51] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 568–578, 2021.

[52] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[53] Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*, 2015.

[54] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, et al. Swin transformer v2: Scaling up capacity and resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12009–12019, 2022.

[55] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[56] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6904–6913, 2017.

[57] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32, 2019.

[58] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*, 2020.

[59] Pengcheng He, Jianfeng Gao, and Weizhu Chen. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*, 2021.

[60] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International Conference on Machine Learning*, pages 12888–12900. PMLR, 2022.

[61] Junnan Li, Ramprasaath Selvaraju, Akhilesh Gotmare, Shafiq Joty, Caiming Xiong, and Steven Chu Hong Hoi. Align before fuse: Vision and language representation learning with momentum distillation. *Advances in neural information processing systems*, 34:9694–9705, 2021.

[62] Peng Wang, An Yang, Rui Men, Junyang Lin, Shuai Bai, Zhikang Li, Jianxin Ma, Chang Zhou, Jingren Zhou, and Hongxia Yang. Ofa: Unifying architectures, tasks, and modalities through a simple sequence-to-sequence learning framework. In *International Conference on Machine Learning*, pages 23318–23340. PMLR, 2022.

[63] Hao Li, Pratik Chaudhari, Hao Yang, Michael Lam, Avinash Ravichandran, Rahul Bhotika, and Stefano Soatto. Rethinking the hyperparameters for fine-tuning. *arXiv preprint arXiv:2002.11770*, 2020.