Supplementary Material for Borrowing Knowledge From Pre-trained Language Model: A New Data-efficient Visual Learning Paradigm

Here we include the general algorithm of our method (§ 1), dataset introduction and more implementation details (§ 2), complete results of the experiments shown in the main paper (§ 3), as well as using different language models (§ 4) and visualizations (§ 5). In section § 6 we provide the detailed derivation of Eq. (5) in the main paper. Finally, a failure case is discussed in § 7.

1. General Algorithm

Our method is generally applicable to various data-efficient training scenarios. Here we provide the general algorithm in Alg. 1 using labeled image data $(\boldsymbol{x}, y)$. We note that unlabeled data can also be utilized by assigning pseudo labels via vanilla pseudo-labeling techniques or more advanced approaches.

Algorithm 1: Language Guided Vision Training

Input: Labeled Data $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n}$; Concepts $\{\mathcal{W}_k\}_{k=1}^{K}$; Vision Backbone F; Pre-trained Language Model T; Prompt set $\{\mathcal{P}_q\}_{q=1}^{m}$; Hyper-parameters λ and τ; Max training iteration: I

Output: Language augmented Vision Model for the Task: $G \circ F$ (G is the task-specific classifier).

```
// Obtain text embedding
   distributions before training.
```

Combine $\mathcal{P}$ with $\mathcal{W}$ to obtain input texts $\{\mathcal{P}_1\mathcal{W}_k, \mathcal{P}_2\mathcal{W}_k, ..., \mathcal{P}_m\mathcal{W}_k\}_{k=1}^{K}$;

Send input texts to T and obtain normalized text embeddings $\{\boldsymbol{t}_1^{(k)}, \boldsymbol{t}_2^{(k)}, ..., \boldsymbol{t}_m^{(k)}\}_{k=1}^{K}$;

for $k = 1, 2, \cdots, K$ **do**

  Estimate text embedding distribution parameters $\boldsymbol{\mu}_k, \Sigma_k$ for concept $\mathcal{W}_k$ using Eq. (3);

end

```
// Train the vision model with
   language semantic guidance
```

. Initialize classifier G and projector H;

for $iter = 1, 2, \cdots, I$ **do**

  $\boldsymbol{f}_i \leftarrow F(\boldsymbol{x}_i)$;

  $\boldsymbol{p}_i \leftarrow G(\boldsymbol{f}_i),\ \boldsymbol{h}_i \leftarrow normalize(H(\boldsymbol{f}_i))$;

  Compute $\mathcal{L}_{emp}(\boldsymbol{p}_i, y_i)$ by Eq. (1);

  Compute $\mathcal{L}_{text}(\boldsymbol{h}_i, y_i)$ by Eq. (5);

  $\mathcal{L} \leftarrow \mathcal{L}_{emp} + \lambda\mathcal{L}_{text}$;

  Update model F, G, H by loss $\mathcal{L}$;

end

2. Dataset And Implementation Details

2.1. Datasets

FGVC Aircraft is a benchmarking dataset for aircraft visual categorization. The dataset contains 10,000 aircraft images, with 100 images for each of 100 different aircraft model variants. The data is split into 6,667 training images and 3,333 testing images.

Stanford Cars is a fine-grained dataset contains 16,185 images of 196 classes of cars. The data is split into 8,144 training images and 8,041 testing images, where each class has been split roughly in a 50-50 split. Classes are typically at the level of Make, Model, Year, e.g. 2012 Tesla Model S or 2012 BMW M3 coupe.

Caltech-UCSD Birds-200-2011 **(CUB-200-2011)** is the most widely-used dataset for fine-grained classification. It contains 11,788 images of 200 subcategories belonging to birds, 5,994 for training and 5,794 for testing. We only use the subcategory name for each bird to as concepts without using the fine-grained natural language descriptions.

CIFAR-100 consists of 60,000 32x32 color images. The 100 classes in the CIFAR-100 are grouped into 20 superclasses. There are 600 images per class, which is split into 500 training images and 100 testing images. Each image comes with a"fine" label (the class to which it belongs) and a"coarse" label (the superclass to which it belongs). We only use the "fine" label for classification.

Office-Home is a standard dataset for domain adaptation and generalization which contains four different domains: Artistic (**Ar**), Clip Art (**Cl**), Product (**Pr**) and Real-world (**Re**). Each domain consists of roughly 4,000 images within the same 65 object categories found typically in office and home scenarios. We use each domain as the single source

Table 10: Complete table of accuracies (%) ↑ on DomainNet for unsupervised domain adaption. In each sub-table, the column-wise domains are selected as the single source domain and the row-wise domains are target domains (testing domain). In each super column, the methods use the same image model written on the top as backbone, where IN-1k/22k refers to using ImageNet-1k/22k pre-trained model. The first super row shows ERM results which serves as baseline, and the two rows below show our methods with $\text{CLIP}_{\text{text}}$ and Bert-L as language model, respectively.

Image Model: ResNet-50 (IN-1k)							
ERM	clp	inf	pnt	qdr	rel	skt	Avg.
clp	–	17.80	37.18	11.69	54.57	43.23	32.89
inf	36.37	–	32.27	3.47	49.88	28.21	30.04
pnt	44.03	18.24	–	6.41	58.56	37.83	33.01
qdr	11.33	1.18	1.89	–	5.64	8.94	5.80
rel	49.81	21.15	49.97	6.46	–	37.06	32.89
skt	52.88	15.56	39.51	14.05	51.23	–	34.65
Avg.	38.88	14.79	32.16	8.42	43.98	31.05	28.31
+$\text{CLIP}_{\text{text}}$	clp	inf	pnt	qdr	rel	skt	Avg.
clp	–	19.12	38.36	12.40	55.02	43.94	33.77
inf	37.65	–	34.78	5.12	50.83	30.86	31.85
pnt	44.56	18.59	–	6.09	58.12	37.98	33.07
qdr	13.47	1.38	1.94	–	6.06	9.61	6.49
rel	51.40	22.18	50.23	7.09	–	38.40	33.86
skt	54.64	16.01	40.44	14.43	51.18	–	35.34
Avg.	40.34	15.46	33.15	9.03	44.24	32.16	29.06
+Bert-L	clp	inf	pnt	qdr	rel	skt	Avg.
clp	–	18.93	38.33	12.76	55.00	43.91	33.79
inf	38.05	–	34.29	4.49	50.52	30.29	31.53
pnt	44.28	18.92	–	6.55	58.29	38.23	33.25
qdr	13.19	1.43	2.3	–	6.16	9.88	6.59
rel	51.28	22.19	50.34	7.57	–	38.54	33.98
skt	54.24	16.62	40.21	15.05	51.37	–	35.50
Avg.	40.21	15.62	33.09	9.28	44.27	32.17	**29.11**

Image Model: ConvNext-S (IN-1k)							
ERM	clp	inf	pnt	qdr	rel	skt	Avg.
clp	–	21.43	45.62	11.33	63.91	48.56	38.17
inf	44.40	–	42.41	6.11	60.50	36.08	37.90
pnt	48.80	20.70	–	5.78	66.00	42.05	36.67
qdr	27.29	4.24	11.27	–	20.56	17.94	16.26
rel	51.50	22.10	52.50	7.22	–	40.70	34.80
skt	58.00	19.30	48.30	13.67	61.23	–	40.10
Avg.	46.00	17.55	40.02	8.82	54.44	37.07	33.98
+$\text{CLIP}_{\text{text}}$	clp	inf	pnt	qdr	rel	skt	Avg.
clp	–	22.24	47.34	12.05	65.24	50.80	39.53
inf	46.40	–	44.98	6.63	62.05	39.47	39.91
pnt	49.80	22.04	–	6.36	65.52	44.20	37.58
qdr	29.89	5.10	14.82	–	24.97	20.08	18.97
rel	52.38	22.96	53.22	7.45	–	42.97	35.80
skt	60.00	20.38	49.11	14.45	62.71	–	41.33
Avg.	47.69	18.54	41.89	9.39	56.10	39.50	35.52
+Bert-L	clp	inf	pnt	qdr	rel	skt	Avg.
clp	–	22.53	47.30	12.18	65.26	50.82	39.62
inf	46.75	–	44.97	6.84	61.90	39.22	39.94
pnt	49.98	21.65	–	6.22	65.75	44.15	37.55
qdr	29.93	5.14	15.11	–	24.89	20.16	19.05
rel	52.75	23.23	53.21	7.61	–	42.96	35.95
skt	59.99	20.50	49.42	14.65	32.58	–	41.43
Avg.	47.88	18.61	42.00	9.50	56.08	39.46	**35.59**

Image Model: Swin-B (IN-22k)							
ERM	clp	inf	pnt	qdr	rel	skt	Avg.
clp	–	26.14	53.32	13.23	69.76	55.29	43.55
inf	55.66	–	51.29	8.00	68.68	45.90	45.91
pnt	58.04	25.73	–	6.42	69.47	46.59	41.25
qdr	35.74	6.00	21.80	–	33.13	23.53	24.24
rel	61.86	27.67	59.39	9.08	–	49.03	41.21
skt	64.91	22.54	53.65	15.69	67.37	–	43.83
Avg.	55.24	21.62	47.89	10.48	61.68	44.07	40.16
+$\text{CLIP}_{\text{text}}$	clp	inf	pnt	qdr	rel	skt	Avg.
clp	–	30.42	57.21	16.20	74.79	59.67	47.66
inf	60.61	–	55.32	10.38	73.32	50.69	50.06
pnt	61.80	29.41	–	10.30	74.00	52.10	45.52
qdr	40.50	10.26	28.51	–	44.97	28.93	30.63
rel	66.51	31.47	62.75	10.98	–	53.78	45.10
skt	69.43	26.27	57.54	18.90	72.27	–	48.88
Avg.	59.77	25.57	52.27	13.35	67.87	49.03	44.64
+Bert-L	clp	inf	pnt	qdr	rel	skt	Avg.
clp	–	30.48	57.27	16.21	74.82	59.66	47.69
inf	60.63	–	55.38	10.50	73.37	50.76	50.13
pnt	61.84	29.45	–	10.38	74.01	52.04	45.54
qdr	40.55	10.27	28.49	–	45.01	29.00	30.66
rel	66.57	31.50	62.80	11.08	–	53.76	45.14
skt	69.49	26.37	57.59	18.98	72.35	–	48.96
Avg.	59.82	25.61	52.31	13.43	67.91	49.04	**44.69**

and test model's performance on other domains

DomainNet is a challenging cross-domain benchmark. The whole dataset comprises ∼0.6 million images drawn from 345 categories in six diverse domains: Infograph (**inf**), Quickdraw (**qdr**), Real (**rel**), Sketch (**skt**), Clipart (**clp**), Painting (**pnt**).

2.2. Additional Implementation Details.

Text embedding estimation. We adopt the modified version of the 80 prompts provided in CLIP, which is adding "This is" before the original prompt to make the sentences more complete. For the text embeddings, we directly use the output embedding corresponding to the concept instead of the [cls] embedding or [eos] embedding. If the concept is divided into several tokens by the tokenizer, we simply take the average of the corresponding output. We find that using the text embeddings obtained in this way generally yields better results than using [cls] token embedding or [eos] token embedding. We think it is because special tokens in pre-trained language models focus more on representing the entire sentences rather than the specific object, making the output embeddings more likely to from "prompt" cluster. For the estimated covariance matrices, we use the diagonal form to ease the computation in loss function.

Training Details. In SSL, the trade-off parameters λ_s, λ_x and λ_u are set as 1.0 for all benchmarks. We adopt a mini-batch size of 20 for EfficientNet-B2 on CIFAR-100 and a batch size of 48 for the rest vision backbones. We assign new pseudo labels for all unlabeled training data at the beginning of each epoch. In Single-DG, the parameter λ_s and λ_x are set as 0.3 and 1.0 respectively, and the batch size is set as 64 for all experiments.

3. Full Results of The Experiments

Due to space limitation, we only provide the average results of several tasks and analysis in the main paper. Here we show the complete results of these experiments.

3.1. Complete Results on DomainNet.

Table 10 shows the full results of our methods using $\text{CLIP}_{\text{text}}$ (not reported in the main paper) and Bert-L as language model, as well as the ERM baseline in our implementation. For $\text{CLIP}_{\text{text}}$, we refer to the text encoder that is jointly pre-trained with ViT-H/14 image encoder. We find it achieves slightly better results than the one jointly trained with ResNet-50 in CLIP.

3.2. Complete Results of the Ablation Study.

Table 11 shows the complete results of our ablation study on *Office-Home*. We can see that different language model is favorable in different tasks (source domain), while generally GPT2-L and mT5-L yields better results.

Table 11: Complete results of the ablation study in table 7 in the main paper. Target domain accuracy (%) ↑ for single domain generalization on *Office-Home* with different pre-trained language models. Backbone ResNet-50 and Swin-B are pre-trained on ImageNet-1k and 22k. For each backbone, the best results are marked in **bold** and the second best are underlined.

Image Model	Language Model	Source:Ar			Source:Cl			Source:Pr			Source:Rw			Avg.
		Cl	Pr	Rw	Ar	Pr	Rw	Ar	Cl	Rw	Ar	Cl	Pr	
ResNet-50 (IN-1k)	None (baseline)	43.71	67.60	73.78	51.03	60.90	63.32	52.73	38.81	72.21	64.80	44.17	76.89	59.16
	$\text{CLIP}_{\text{text}}$	47.18	70.27	76.53	57.40	**67.34**	68.68	57.24	43.14	**76.11**	68.77	48.12	79.66	63.37
	Bert-L	**47.97**	70.49	76.54	57.85	66.91	**69.22**	57.15	**44.01**	**76.11**	68.64	48.82	79.68	63.62
	mT5-L	47.40	70.06	76.96	57.15	67.18	69.18	**57.35**	43.92	75.95	69.22	**49.23**	**80.09**	63.64
	GPT2-L	47.58	**70.69**	**76.98**	**58.01**	66.86	69.13	**57.35**	43.69	75.90	**70.17**	49.19	79.68	**63.77**
Swin-B (IN-22k)	None (baseline)	70.32	84.97	88.99	82.52	85.91	87.68	80.21	67.59	88.56	83.80	68.05	90.15	81.56
	$\text{CLIP}_{\text{text}}$	73.08	86.12	89.68	84.72	87.97	89.04	83.36	69.27	90.21	85.00	70.91	91.58	83.41
	Bert-L	**73.26**	**86.75**	**90.34**	**85.21**	**88.24**	**89.92**	82.82	**70.49**	**90.96**	84.38	70.31	90.27	83.58
	mT5-L	73.05	86.25	90.05	84.89	87.73	89.29	**83.41**	69.84	90.44	85.38	71.05	91.72	83.59
	GPT2-L	73.22	86.73	90.20	85.13	88.17	89.74	83.35	70.06	90.73	**85.54**	**71.27**	**91.78**	**83.83**

Table 12: Complete results of the knowledge distillation comparison experiment in table 9 in the main paper. Vision models ConvNext-S, ConvNext-B, Swin-B and language models Bert-S, Bert-L are selected as different teachers for comparison. The best results are marked in **bold** and the second best are underlined.

Student Model	Teacher Model	Source:Ar			Source:Cl			Source:Pr			Source:Rw			Avg.
		Cl	Pr	Rw	Ar	Pr	Rw	Ar	Cl	Rw	Ar	Cl	Pr	
ResNet-50	None (baseline)	43.71	67.60	73.78	51.03	60.90	63.32	52.73	38.81	72.21	64.80	44.17	76.89	59.16
	ConvNext-S	45.38	68.01	75.83	53.70	63.84	65.89	54.51	41.46	73.90	66.00	46.32	78.55	61.12
	ConvNext-B	46.06	69.04	75.14	53.98	64.59	66.72	53.23	41.42	74.38	65.72	46.64	78.42	61.28
	Swin-B	45.99	68.94	75.71	53.74	64.46	66.11	53.82	41.48	74.37	65.66	46.84	78.59	61.31
	Bert-S	**48.11**	**70.67**	**76.70**	57.64	66.70	68.63	56.24	43.83	75.67	**69.84**	48.55	**80.20**	63.57
	Bert-L	47.97	70.49	76.54	**57.85**	**66.91**	**69.22**	**57.15**	**44.01**	**76.11**	68.64	**48.82**	79.68	**63.62**

3.3. Complete Results of The Knowledge Distillation Comparison Experiment.

Here we discuss the knowledge distillation experiment in section 4.4 more thoroughly and report the complete experiment results of table 9 in table 12. Classic knowledge distillation involves a large vision model as teacher and an actual deployed small model as student. In the comparison, we adopt the standard response-based distillation method [1], where teacher model is first fine-tuned on the interested task and then use its logits as guidance to improve the student model. Denote the teacher logit and student logit as $\boldsymbol{p}_t$ and $\boldsymbol{p}_s$, then the overall knowledge distillation objective for student model has the following form:

$$\mathcal{L}_{kd} = \mathcal{L}_{emp}(\boldsymbol{p}_s, y) + \lambda \mathcal{L}_R(\boldsymbol{p}_t, \boldsymbol{p}_s), \tag{8}$$

where $\mathcal{L}_R$ usually is the KL-divergence loss and λ is a trade-off hyperparameter.

It is well-believed that visual knowledge distillation improves the student model via the transfer of "dark knowledge", which refers to the latent semantic relationship between categories that can be reflected through the teacher model's logits. Similarly, the text embedding space proposed in our method is semantically rich and, through our proposed feature alignment objective, is able to transfer the semantic relationship to vision model. To compare their effectiveness, we train three large vision model of different architecture and size and conduct standard visual knowledge distillation on ResNet-50 using labeled data from a single domain on *Office-Home*. For our method, we denote the pre-trained language model as teacher and the interested vision model (ResNet-50 as well) as student. We include the results of our method using a smaller-sized language model Bert-S [2] to examine the influence of teacher model size.

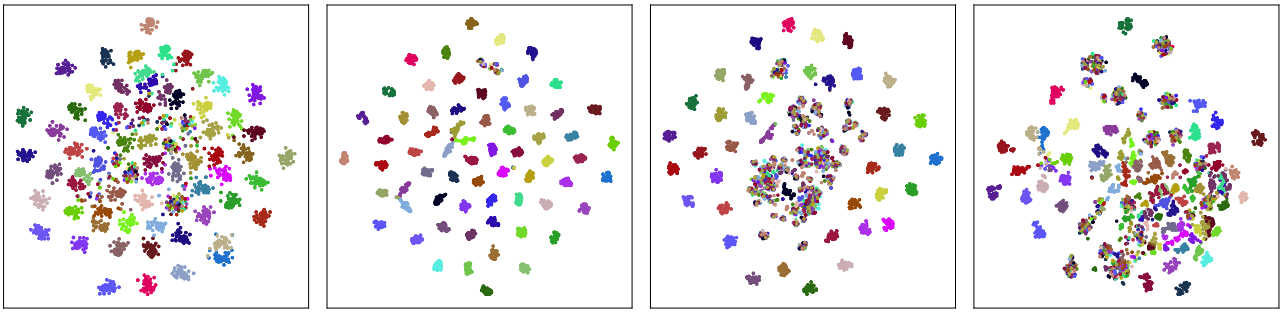
As shown in table 12, when examine the cross-domain generalization performance, we observe significant advantage of using language model as teacher over vision model. Specifically, three vision teacher model achieves similar and limited improvements on ResNet-50, while the language teacher model in our method doubles the accuracy improvement. We attribute this improvement to the more direct semantic transfer in our method.

4. Additional Results With Different Language Models

Here we show more results of using different language models to enhance vision model on *FGVC Aircraft*, *Stanford Cars*, *CUB-200* and *Office-Home*. The results are shown in table 13. We can see that all the language models achieves comparable performances, which validate that our method is able to borrow knowledge from a variety of choices. It is also advised for practitioner to compare different pre-trained language models on the interested task to

Table 13: Accuracy (%) ↑ of our method on four benchmarks with different pre-trained language models. The best results are marked in **bold** and the second best are underlined.

Vision Model	Language Model	Aircraft			Stanford Cars			CUB-200			OH	Avg.
		15%	30%	50%	15%	30%	50%	15%	30%	50%		
ResNet-50	None (baseline)	39.57	57.46	67.93	36.77	60.63	75.10	45.25	59.68	70.12	59.16	57.17
	$\text{CLIP}_{\text{text}}$	71.08	83.05	87.34	79.49	88.07	91.28	65.00	74.78	80.31	63.37	78.38
	Bert-L	71.05	83.41	87.22	79.34	88.78	91.46	65.96	**75.70**	80.91	63.62	78.75
	Deberta-L	**71.17**	**83.71**	**87.40**	79.44	88.10	91.53	66.00	75.65	80.69	63.74	78.74
	T5-L	70.51	83.29	86.98	79.79	88.88	91.75	**66.14**	75.25	80.07	63.62	78.63
	mT5-L	70.39	83.02	87.37	**79.93**	**88.91**	**91.77**	65.98	75.68	**81.08**	63.64	**78.78**
	GPT-2	70.18	82.96	86.71	79.23	88.61	**91.77**	66.02	75.53	80.88	**63.77**	78.57



CLIP Deberta-L mT5-L GPT2-L

Figure 5: T-SNE visualization of the text embedding space of different language models on *Office-Home*.

fully exploit the flexibility of our method.

5. Additional Visualizations

In this section, we show more visualizations of the generated text embedding space to deepen our understanding of its property. We first show the t-SNE visualization results of different language model. Then we provide heatmap of concept similarities on two extra datasets.

5.1. T-SNE Visualization.

Besides the t-SNE figure of Bert-L text embedding space, we illustrate here four other text embedding spaces generated by $\text{CLIP}_{\text{text}}$, Deberta-L, mT5-L and GPT2-L respectively. As shown in Fig. 5, it is interesting to discover that text embedding spaces generated by different pre-trained language model varies significantly. $\text{CLIP}_{\text{text}}$ forms less compact "concept clusters" and thus results in smaller separation between different concepts. Other language models form more compact clusters for each concept, yet they also form "prompt clusters".

To explain why mT5-L has more prompt cluster than Deberta-L while achieving better results in table 13, we must consider the proposed distribution estimation process. In fact, compared to most text embeddings that are clustered based on the same category, those few outliers cannot have notable effect on the estimated parameters.

5.2. Cosine Similarity Between Categories.

Similar to the text embedding similarity heatmap in Fig. 4 in the main paper, we plot the cosine similarity heatmap on two additional datasets: a fine-grained dataset *CUB-200-2011* and a coarse-grained dataset *CIFAR-100*. The cosine similarity is calculated between each two concepts using the corresponding estimated mean embeddings. We normalize the whole similarity matrix to range $[0, 1]$ to better illustrate the similarity relationship among concepts.

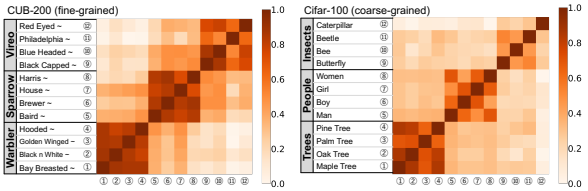


Figure 6: Normalized cosine similarity map between mean embeddings of different concepts on two datasets: fine-grained *CUB-200-2011* and coarse-grained *CIFAR-100*. All embeddings are generated by pre-trained Bert-L.

As shown in Fig. 6, in each dataset, we manually select 12 categories that belongs to three different superclasses. In *CUB-200*, we randomly choose four bird subcategories under "Warbler", "Sparrow" and "Vireo". In *CIFAR-100*, we select three superclass defined in the dataset: "Trees", "People" and "Insects". We can observe that the text em-

beddings within the same group are more similar to each other than to those outside the group.

6. Derivation of the Loss Upper Bound.

In this section, we provide the detailed derivation of the upper bound of $\mathcal{L}_{text}^{\infty}$ (Eq. (5) in the main paper).

$$\mathcal{L}_{text}^{\infty} = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}_{\boldsymbol{t}^{(y_i)}}\left[-\log\frac{e^{\tau \boldsymbol{h}_i^\top \boldsymbol{t}^{(y_i)}}}{e^{\tau \boldsymbol{h}_i^\top \boldsymbol{t}^{(y_i)}} + \sum_{k\neq y_i}^{K}\mathbb{E}_{\boldsymbol{t}^{(k)}} e^{\tau \boldsymbol{h}_i^\top \boldsymbol{t}^{(k)}}}\right] \tag{9}$$

$$= \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}\left[\log(e^{\tau \boldsymbol{h}_i^\top \boldsymbol{t}^{(y_i)}} + \sum_{k\neq y_i}^{K}\mathbb{E}[e^{\tau \boldsymbol{h}_i^\top \boldsymbol{t}^{(k)}}]) - \log(e^{\tau \boldsymbol{h}_i^\top \boldsymbol{t}^{(y_i)}})\right] \tag{10}$$

$$\leq \frac{1}{n}\sum_{i=1}^{n}\log(\mathbb{E}\left[e^{\tau \boldsymbol{h}_i^\top \boldsymbol{t}^{(y_i)}} + \sum_{k\neq y_i}^{K}\mathbb{E}[e^{\tau \boldsymbol{h}_i^\top \boldsymbol{t}^{(k)}}]\right]) - \mathbb{E}\left[\tau \boldsymbol{h}_i^\top \boldsymbol{t}^{(y_i)}\right] \tag{11}$$

$$= \frac{1}{n}\sum_{i=1}^{n}\log(\sum_{k=1}^{K}\mathbb{E}_{\boldsymbol{t}^{(k)}}\left[e^{\tau \boldsymbol{h}_i^\top \boldsymbol{t}^{(k)}}\right]) - \tau \boldsymbol{h}_i^\top \boldsymbol{\mu}^{(y_i)} \tag{12}$$

$$= \frac{1}{n}\sum_{i=1}^{n}\log(\sum_{k=1}^{K}e^{\tau \boldsymbol{h}_i^\top \boldsymbol{\mu}^{(k)} + \tau^2 \boldsymbol{h}_i^\top \boldsymbol{\Sigma}^{(k)} \boldsymbol{h}_i/2}) - \tau \boldsymbol{h}_i^\top \boldsymbol{\mu}^{(y_i)} \tag{13}$$

$$= \frac{1}{n}\sum_{i=1}^{n}\left[-\log\frac{e^{\mathcal{F}(\boldsymbol{h}_i, y_i)}}{\sum_{k=1}^{K} e^{\mathcal{F}(\boldsymbol{h}_i, k)}} + \mathcal{F}(\boldsymbol{h}_i, y_i) - \tau \boldsymbol{h}_i^\top \boldsymbol{\mu}^{(y_i)}\right] \tag{14}$$

$$= \frac{1}{n}\sum_{i=1}^{n}\left[-\log\frac{e^{\mathcal{F}(\boldsymbol{h}_i, y_i)}}{\sum_{k=1}^{K} e^{\mathcal{F}(\boldsymbol{h}_i, k)}} + \tau^2 \boldsymbol{h}_i^\top \boldsymbol{\Sigma}^{(y_i)} \boldsymbol{h}_i/2\right]. \tag{15}$$

We denote $\mathcal{F}(\boldsymbol{h}, y) \triangleq \tau \boldsymbol{h}^\top \boldsymbol{\mu}^{(y)} + \tau^2 \boldsymbol{h}^\top \boldsymbol{\Sigma}^{(y)} \boldsymbol{h}/2$ for simplicity. The inequality in Eq. (11) comes from the Jensen's inequality for convex function: $\mathbb{E}[\log(X)] \leq \log(\mathbb{E}X)$, and the moment generation function for Gaussian variable $X \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$: $\mathbb{E}[e^{\boldsymbol{h}^\top X}] = e^{\boldsymbol{h}^\top \boldsymbol{\mu} + \boldsymbol{h}^\top \Sigma \boldsymbol{h}/2}$ is applied to obtain Eq. (13).

7. Failure Case.

Here we show a failure case of our method. It happens when we use the output embedding of the [eos] token (at the end of the sencentense) in GPT2-L as the text embedding. Note that unlike Bert or Deberta that use bidirectional encoding process which allows any token in the middle receives information from both sides, GPT uses unidirectional encoding where input texts are processed in one direction. Therefore, it seems more reasonable to use [eos] token embedding placed at the end of the sentence for GPT. However, as shown in the t-SNE figure (Fig. 7), the resulting embedding space is highly non-separable among different categories, and thus cannot provide useful information to help the vision model. How to effectively use the [eos] embeddings in unidirectional language models requires further exploration in the future.

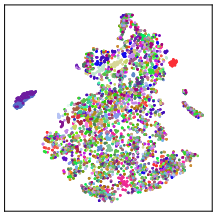


Figure 7: T-SNE of [eos] token embeddings of GPT2-L.

References

[1] Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. *IJCV*, 129(6):1789–1819, 2021. 3

[2] Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Well-read students learn better: On the importance of pre-training compact models. *CoRR, abs/1908.08962*, 2019. 3