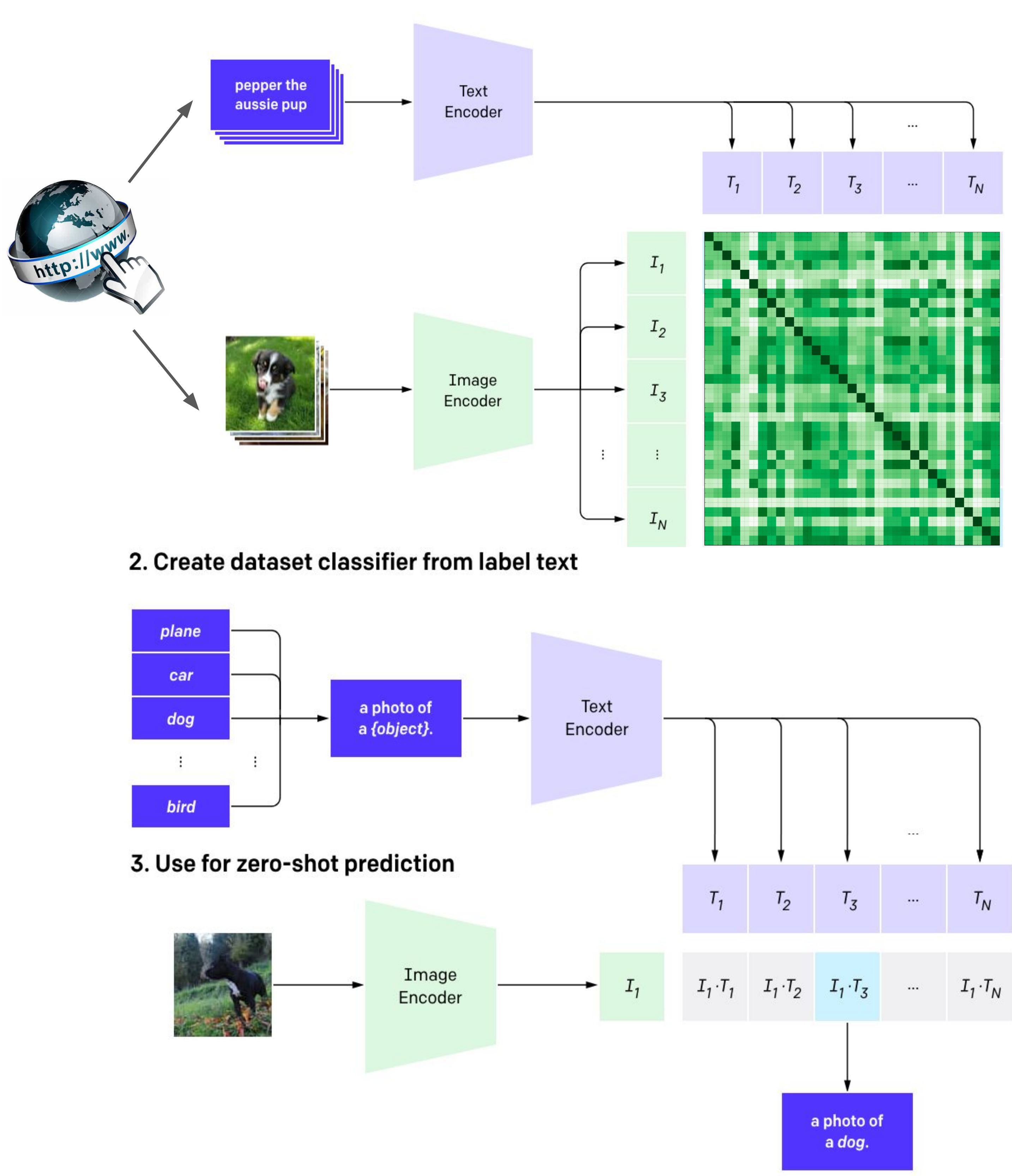
DIME-FM : DIstilling Multimodal and Efficient Foundation Models

Ximeng Sun[1], Pengchuan Zhang[2], Peizhao Zhang[2], Hardik Shah[2], Kate Saenko[1,2], Xide Xia[2]
[1] Boston University, [2] Meta AI

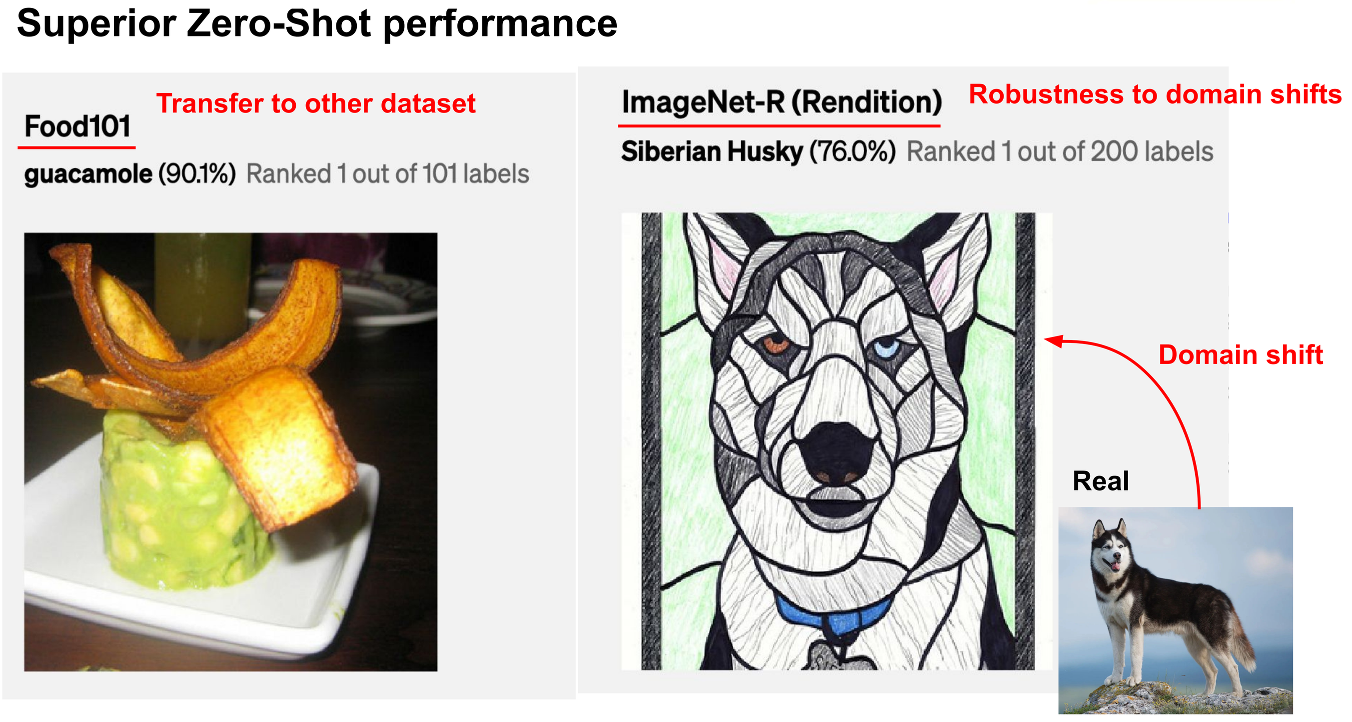
Meta AI · BOSTON UNIVERSITY · ICCV23 PARIS

Background and Motivation

Contrastive Pretraining for Vision-Language Foundation models (such as CLIP)



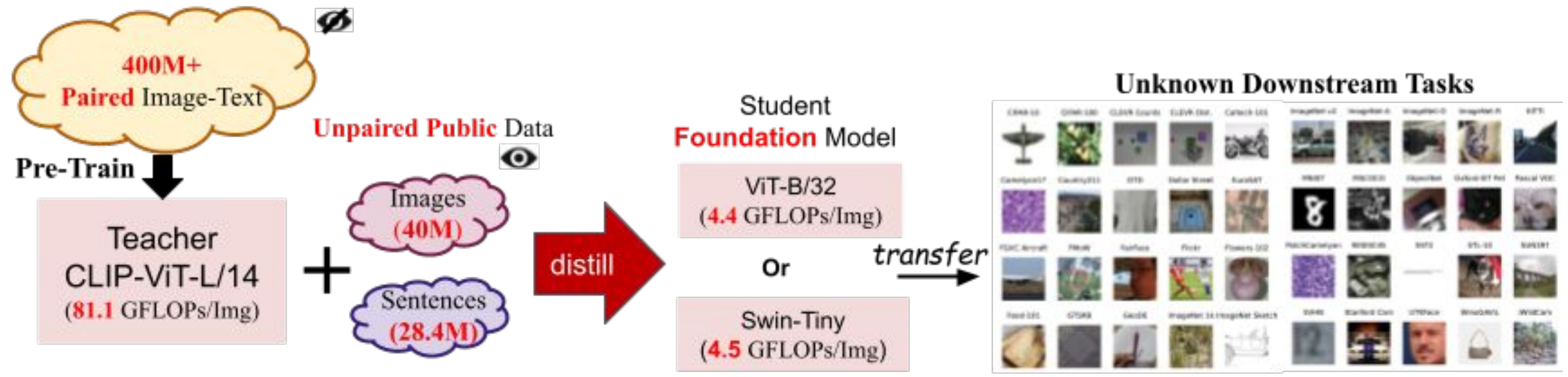
Superior Zero-Shot performance



⚠ **Problem: very expensive to train a CLIP-like model! ! !**

- **Heavy Data Consumption and Data Privacy**
  - 400M pretraining image-text pairs
  - private to OpenAI
- **Expensive Training:**
  - batch-size = 32,768
  - largest ResNet model, RN50x64, took 18 days to train on 592 V100 GPUs
  - largest Vision Transformer took 12 days on 256 V100 GPUs

How to efficiently train a customizable CLIP-like model?



Our Remarkable Performance

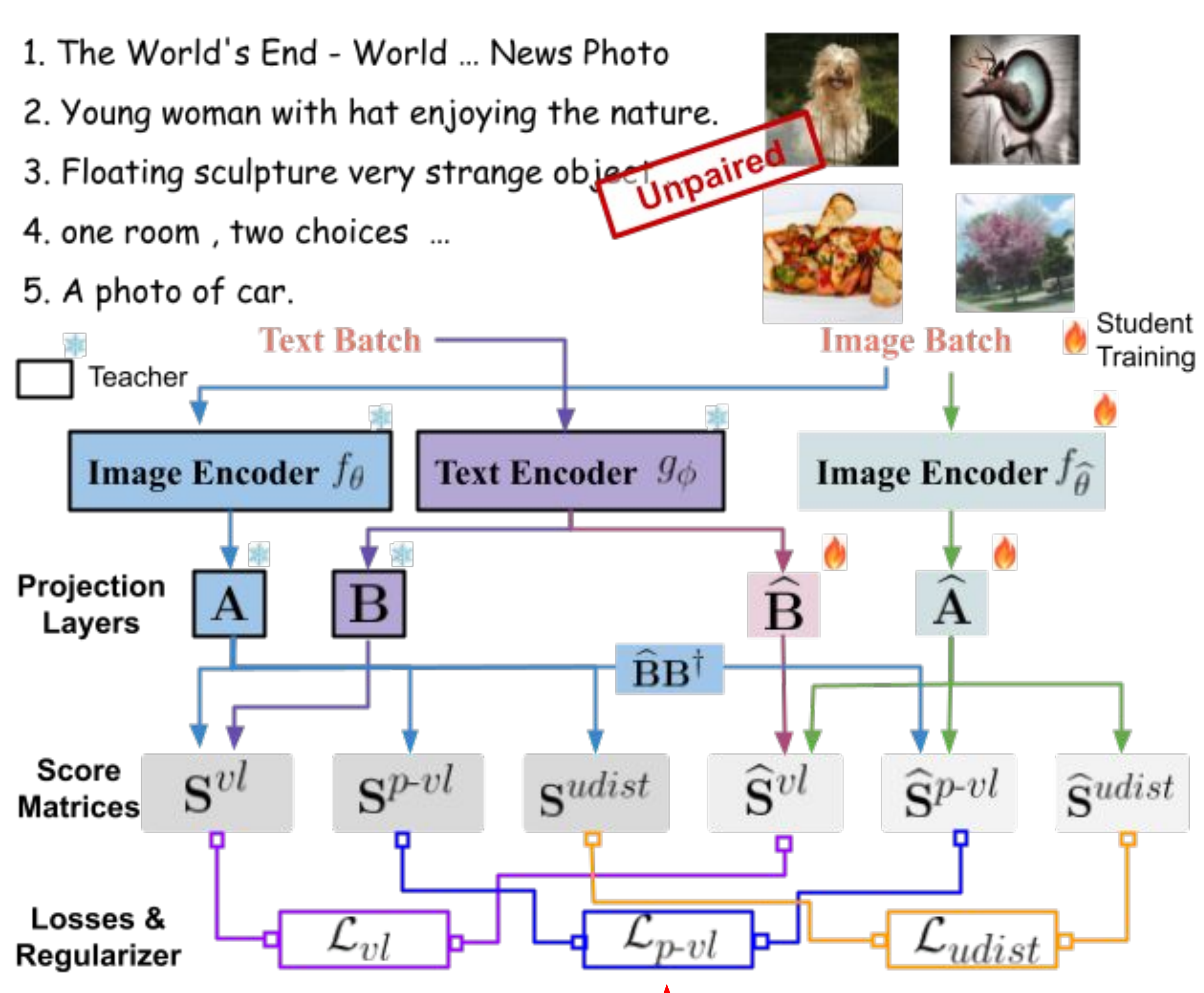
	ZS on IN-1K	ZS on ELEVATER	LP on ELEVATER	Robustness
OpenAI CLIP-VIT-B/32	63.4%	57.2%	78.2%	48.6%
Distill-VIT-B/32 (Constructed NLP Corpus)	66.5%	56.4%	79.2%	50.2%
Distill-VIT-B/32 * (Caption Corpus))	64.8%	55.0%	78.6%	49.4%
Distill-VIT-B/32 * (Task-Aware + Captions))	66.1%	57.7%	79.4%	50.4%

Dataset	Method	Zero-Shot ELEVATER	Zero-Shot IN-1K	Linear Probing ELEVATER
IN-21K	UniCL	27.2%	28.5%	74.8%
	UniCL*	40.9%	51.4%	75.3%
	Distill-UniCL*	**45.6%**	**59.5%**	**76.2%**
IN-21K + YFCC-14M	UniCL	37.1%	40.5%	**77.1%**
	UniCL*	44.6%	58.7%	75.4%
	Distill-UniCL*	**47.6%**	**60.0%**	76.6%

Question: What logits should we distill in the open-vocabulary image recognition task?

🌟🌟🌟 **We find out the choice of text corpus is critical.**

Key Contribution 1: Three Distillation Losses



We propose the Pseudo Text Corpus and Pseudo VL knowledge distillation loss

Key Contribution 2: Construct the visually-grounded Text Corpus

Algorithm 1: Constructing text corpus $\mathcal{T}$

Input : image embeddings $\mathcal{U}$ as defined in Eq.15. A large text corpus $\mathcal{T}_{large}$.

Output : Selected text corpus $\mathcal{T}$, and $|\mathcal{T}| \approx |\mathcal{U}|$

$\mathcal{U}_{left} \longleftarrow \mathcal{U},\ \mathcal{T}_{avail} \longleftarrow \mathcal{T}_{large},\ \mathcal{T} \longleftarrow \emptyset, U_p = \infty$

while $\mathcal{U}_{left} \neq \emptyset$ and $|\mathcal{U}_{left}|/U_p < 0.95$ **do**

  $U_p = |\mathcal{U}_{left}|,\ Matched = dict()$

  for $\boldsymbol{u} \in \mathcal{U}_{left}$ **:**

    /* find the best text that matches the image */

    $\boldsymbol{t}(\boldsymbol{u}) = \arg\max_{\boldsymbol{t} \in \mathcal{T}_{avail}} s(\boldsymbol{u}, \mathbf{B} \cdot g_\phi(\boldsymbol{t}))$ //

    $Matched[\boldsymbol{u}] = \boldsymbol{t}(\boldsymbol{u})$

  for $\boldsymbol{u}, \boldsymbol{t} \in Matched.items()$ **:**

    /* For all images matching to the same text, pick the first match */

    if $\boldsymbol{t} \in \mathcal{T}_{avail}$ **:**

      $\mathcal{U}_{left} \longleftarrow \mathcal{U} \backslash \{\boldsymbol{u}\}$

      $\mathcal{T}_{avail} \longleftarrow \mathcal{T}_{avail} \backslash \{\boldsymbol{t}\},\ \mathcal{T}.add(\boldsymbol{t})$