Structure-Encoding Auxiliary Tasks for Improved Visual Representation in Vision-and-Language Navigation

Chia-Wen Kuo
Georgia Tech
albert.cwkuo@gatech.edu

Chih-Yao Ma
Georgia Tech
cyma@gatech.edu

Judy Hoffman
Georgia Tech
judy@gatech.edu

Zsolt Kira
Georgia Tech
zkira@gatech.edu

Abstract

In Vision-and-Language Navigation (VLN), researchers typically take an image encoder pre-trained on ImageNet without fine-tuning on the environments that the agent will be trained or tested on. However, the distribution shift between the training images from ImageNet and the views in the navigation environments may render the ImageNet pre-trained image encoder suboptimal. Therefore, in this paper, we design a set of structure-encoding auxiliary tasks (SEA) that leverage the data in the navigation environments to pre-train and improve the image encoder. Specifically, we design and customize (1) 3D jigsaw, (2) traversability prediction, and (3) instance classification to pre-train the image encoder. Through rigorous ablations, our SEA pre-trained features are shown to better encode structural information of the scenes, which ImageNet pre-trained features fail to properly encode but is crucial for the target navigation task. The SEA pre-trained features can be easily plugged into existing VLN agents without any tuning. For example, on Test-Unseen environments, the VLN agents combined with our SEA pre-trained features achieve absolute success rate improvement of 12% for Speaker-Follower [14], 5% for Env-Dropout [37], and 4% for AuxRN [50].

1. Introduction

In Vision-and-Language Navigation (VLN) [5], an agent navigates in a complicated environment to a target location by following human instructions. In this task, the agent needs to interpret human instructions, encode visual input, and then infer appropriate actions according to the joint textual and visual information. Remarkable progress has been made since the proposition of the Room-to-Room (R2R) dataset by Anderson et al. [5], including generating more training data [14, 22, 37], learning a better joint visual and textual representation [43, 21, 17, 29], improving the agent's internal state representation for the *policy* network (as opposed to visual encoder) by auxiliary tasks [27, 50, 44], and so on.

However, most of the existing works ignore the importance of the underlying visual representation by simply taking an image encoder (CNN model that encodes an image x into a feature $f_x \in R^d$) pre-trained on ImageNet to encode the views in the navigation environments (*e.g.,* Matterport3D [7]). Because of the data distribution shift between ImageNet and the navigation environments, as well as the difference between the pre-training task (image classification) and the target task (VLN), the ImageNet pre-trained image encoder may not be able to encode information crucial for the VLN task. One naïve way to mitigate this negative effect is to fine-tune the image encoder on the target environments and task. However, in the navigation environments, image labels such as semantic segmentation masks, object bounding boxes, or object and scene classes may not be available for fine-tuning the image encoder. Furthermore, it is computationally prohibitive[1] to fine-tune the image encoder jointly with the agent on the target VLN task.

To improve the image encoder without the need for manually annotated labels in the target environments and without fine-tuning with the VLN agent jointly, we pre-train the image encoder on proposed *structure-encoding auxiliary tasks (SEA)* with data available in the navigation environments shown in Figure 1. Specifically, we collect RGB images from different views of the environments, a view's neighboring views, and traversable directions within a view. After that, we pre-train the image encoder via the proposed auxiliary tasks on the gathered data. We then pre-compute the features for each view of the training environments using the frozen, pre-trained image encoder, and train the navigation agent following the classical VLN methods with our pre-computed features.

Combined with our SEA pre-trained features, VLN methods achieve absolute success rate improvement of 12%

[1] To train an agent with panoramic action space, in each iteration, we take 64 trajectories, each trajectory contains 5 steps on average, each step contains 36 views, and each view is a 640 × 480 high-resolution image. These sum up to 10k+ forward passes of high-resolution images through the image encoder in just one training iteration.

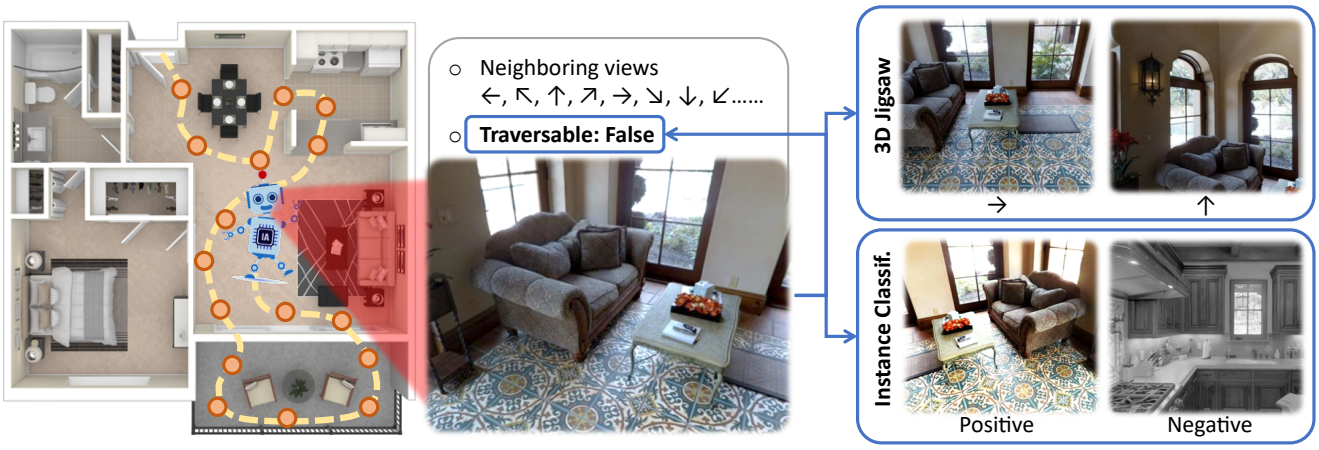


Figure 1: We propose three auxiliary tasks: (1) 3D Jigsaw, (2) Traversability Prediction, and (3) Instance Classification, to improve the visual representation for the downstream VLN task. These auxiliary tasks train only on data available in the navigation environments such as RGB image views, a view's neighboring views, and traversable directions within a view.

for Speaker-Follower [14], 5% for Env-Dropout [37], and 4% for AuxRN [50] on Test-Unseen under the single-run setting (*i.e.,* without pre-exploration or beam search). To understand how the agents benefit from our SEA pre-trained features, we conduct thorough ablation studies on what information is encoded by and how the agent's navigation performance is affected by each auxiliary task. Compared with ImageNet pre-trained features, our SEA pre-trained features better encode structural information of the scenes, which are crucial for the target navigation tasks. The source code and collected data for pre-training the image encoder as well as the pre-trained SEA features will be released to facilitate future research in VLN. Our key contributions are that we:

- Design and customize a set of auxiliary tasks to improve the visual representation by training on images and metadata easily attainable in the navigation environments.
- Achieve significant performance improvement on the unseen environments when combining our SEA pre-trained features with VLN methods including Speaker-Follower [14], Env-Dropout [37], and AuxRN [50].
- Conduct thorough ablation studies to understand how an agent benefits the proposed auxiliary tasks and SEA pre-trained features.

2. Related Works

2.1. Auxiliary Tasks for Training an Agent

In vision and language navigation, due to the limited amount of training data, researchers have proposed auxiliary tasks to regularize the agent model and to provide additional training signals. Despite their success in the VLN task, this line of works focuses on refining the agent's internal state representation for the policy network rather than the visual representation, and simply encode each view with a frozen, ImageNet [35] pre-trained image encoder. Ma et al. [27] proposed a progress monitoring module to improve the grounding between visual and textual information. Huang et al. [21] also aimed at improving the grounding between visual and textual information by a cross-modal alignment loss to classify aligned instruction-path pairs. To further improve the agent's state representation, they propose a coherence loss to predict future k steps. Another way of improving the cross-modal grounding is to pre-train the model on paired data of image and text [17, 29]. To improve the model's generalizability, Wang et al. [44] proposed an adversarial training strategy to remove scene-specific information from the agent's state representation. Zhu et al. [50] achieved significant improvement over the previous state of the art with four auxiliary tasks including speaker model, progress monitor, orientation prediction, and trajectory-instruction matching.

In reinforcement learning (RL), it has been shown that jointly training the agent with auxiliary tasks improves state representations and greatly expedites training. Some common auxiliary tasks in RL include: (1) future prediction [26, 15, 46, 13, 32], which predicts an agent's future state conditioned on its current state and the actions taken, (2) inverse dynamic [15, 46, 32], which predicts the actions taken between two states, and (3) contrastive learning [3, 36], which applies contrastive learning to refine the state representation. Despite differences between RL works and VLN, AuxRN [50] (which we further improve upon) incorporate an auxiliary task of agent's orientation prediction in VLN, which is similar to the inverse dynamic auxiliary task in RL. We also draw inspiration from the RL line of works to propose our auxiliary tasks. For example, the concept of traversability prediction is similar to Chaplot et al. [8], which target building a topological map of

the environment for image-goal navigation. Different from these lines of work, our proposed auxiliary tasks focuses on the improvement of visual representation, which is later used by a VLN agent for navigation and state representation. Our proposed auxiliary tasks are effective and improve upon AuxRN, which introduces several auxiliary tasks for training the VLN policy, by large margins.

2.2. Self-Supervised Learning in Computer Vision

Self-supervised learning has achieved great success in learning good visual representations without labels for data. The learned representations generalize well to a wide variety of downstream tasks such as image classification, object detection, scene classification, and so on. To train the model without labels, self-supervised learning methods define auxiliary tasks by visual clues that are inherent to the data. Such visual clues include: spatial information from images [12, 30], spatial-temporal information from video or motion [42, 1, 33], image colors [34, 48, 24, 25], etc. Recently, contrastive learning [31, 39, 10, 18, 16, 6] has achieved comparable performance with its supervised counterpart. The goal of contrastive learning is to learn a visual representation that is invariant to a set of image augmentations [40, 10, 49, 51, 45] by identifying an image's augmented copy from a pool of other images.

Inspired by the success of self-supervised learning in computer vision, in this work we design three auxiliary tasks to improve the image encoder without the need for data labels such as trajectory-instruction pairs, object labels, etc. Furthermore, since the agent can move in the interactive environments, we take advantage of this when designing the auxiliary tasks. For example, different from other jigsaw-like self-supervised tasks that generate jigsaw puzzles by cropping images [12, 30] or by consecutive frames in video clips [2], our proposed 3D jigsaw actively samples neighboring views, introducing more natural variations from the change of viewpoints.

3. Method

In existing VLN methods, the ImageNet pre-trained features may be suboptimal due to the data distribution shift between ImageNet and the navigation environments, and the difference between the pre-training classification task and the target VLN task. As explained in Section 1, the naïve solution of fine-tuning the image encoder in the target environments is inapplicable due to the lack of labeled images. Furthermore, it is also computationally prohibitive to jointly train the image encoder with the agent on the target VLN task. Therefore, we seek to design auxiliary tasks that can improve the image encoder but rely only on data available in the navigation environments.

3.1. Problem Setting

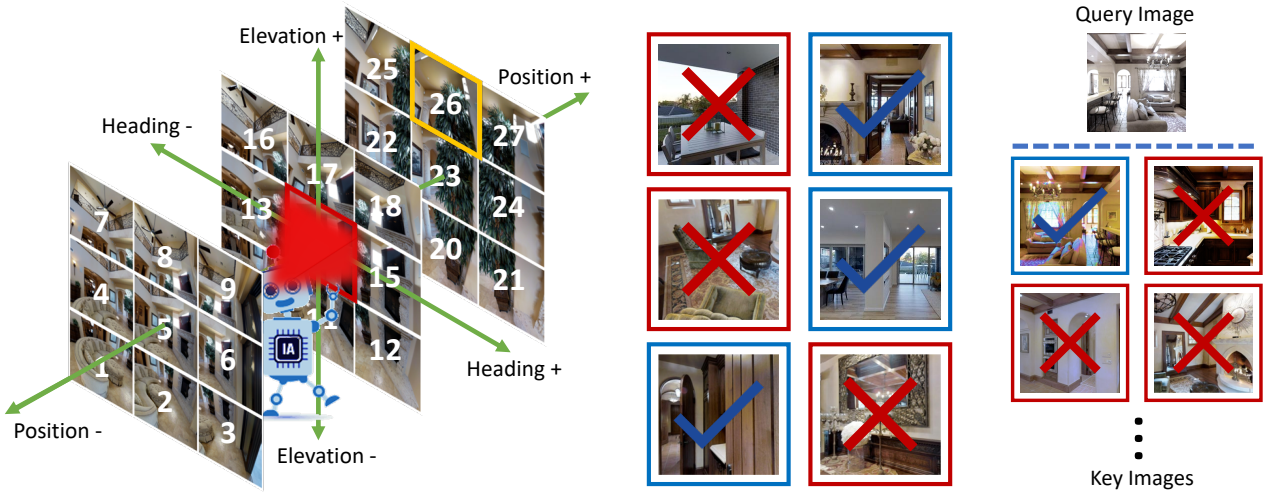
In VLN, a navigation agent is given training data in the form of trajectory-instruction pairs in different indoor environments. In the training environments, the agent is also allowed to access data such as RGB image views, a view's neighboring views, and traversable directions contained in a view. In this paper, for fair comparison with other works, the proposed auxiliary tasks are trained with only these information to make sure that the performance gain is *not* coming from additional training signals (*e.g.,* semantic segmentation map, room type).

3.2. Auxiliary Tasks

With the data collected from the environments, we aim to design auxiliary tasks that help the image encoder encode visual information that is crucial for the target VLN task. To find out what are important features, we start by observing the following instruction example: *"**Exit** the **screening room**, make a **right**, go **straight into** the **room** with the **globe** and stop."* As highlighted above, to correctly follow the instruction, the agent needs to encode the following information from its image encoder: (1) structural information of the scene (exit, right, straight into), and (2) discriminative information for scenes and objects (screening room, room, globe) in the visual representation. Therefore, we design three auxiliary tasks shown in Figure 2: (1) *3D jigsaw*, (2) *traversability prediction*, and (3) *instance classification* to encode these crucial information for VLN.

3.2.1 3D Jigsaw

In order to follow the instructions correctly to reach the target location, the agent has to interpret instructions such as "turn **left**", "turn **right** when you see the sofa on your **left**", "stop **in front of** the TV" from the visual representation. Therefore, we propose the auxiliary task of 3D jigsaw to encode structural information of the scene by predicting the relative poses (position, heading, and elevation) of two views. As shown in Figure 2a, given an *anchor view* x_a in the red box, a *query view* x_q in the yellow box is sampled from the neighboring views around the anchor view. Neighboring views are views within the $[-1, 0, +1]$ range of discretized headings, elevations, and positions, forming a 3D jigsaw with 27 views ($3 \times 3 \times 3$). The label of the neighboring views (jigsaw labels) can be uniquely determined by their relative poses to the anchor view (the "numbers" overlaid on the neighboring views in Figure 2a). If the sampled anchor view is looking up, neighboring views of {7-9, 16-18, 25-27} in Figure 2a would be unavailable due to the way views are discretized (similarly for the case of looking down.) On the other hand, if the sampled anchor view does not contain any traversable directions, neighboring views of {19-27} would be unavailable because the agent cannot go



(a) 3D jigsaw (b) Traversability prediction (c) Instance Classification

Figure 2: (Best viewed on the computer, in color and zoomed in.) We design three auxiliary tasks to encode structural information of the scenes, as well as the discriminative feature for object and scene classification crucial for the VLN task. ***(a)*** The auxiliary task of 3D jigsaw is to predict the relative pose between an anchor view (red box) and a query view (yellow). The query view is sampled from the anchor view's neighboring views along the elevation, heading, and position dimensions. ***(b)*** The auxiliary task of traversability prediction is to predict whether a view contains any traversable direction. The images in the blue box are labeled as *True* (contain traversable directions), and the images in the red box are labeled as *False* (do not contain traversable directions.) ***(c)*** The auxiliary task of instance classification is to identify a view's augmented copy from a pool of other image views. In this example, the view in the blue box is the corresponding augmented copy (positive pair), while the views in the red box are other image views (negative pairs).

one step forward. Nevertheless, the auxiliary task and jigsaw labels can still be constructed in a similar way. Those unavailable neighboring views are simply removed.

Conditioned on the anchor view, the 3D jigsaw task is formulated as a 27-class classification problem. The prediction p_{jig} is computed by:

$$p_{jig} = \text{softmax}(\phi_{jig}([f_{enc}(x_a), f_{enc}(x_q)])), \quad (1)$$

where f_{enc} is the image encoder shared with other auxiliary tasks, ϕ_{jig} is a multi-layer perceptron specific for 3D jigsaw, and $[\cdot,\cdot]$ is a concatenation operation along the feature dimension. The loss is simply a cross-entropy loss:

$$\mathcal{L}_{jig} = -\frac{1}{N}\sum_{i}^{N} y_{i,jig} \log p_{i,jig}, \quad (2)$$

where $y_{i,jig}$ and $p_{i,jig}$ are the jigsaw label and prediction for the i-th training example, respectively, and the loss is averaged over a mini-batch of N examples.

3.2.2 Traversability Prediction

In order to encode the layout (structure) and navigation information of the scene and environments, we propose an auxiliary task of traversability prediction shown in Figure 2b. The image encoder classifies a given image view as *true* when the view contains traversable directions, otherwise classifies it as *false*. Following the practice in Matterport3D (MP3D) simulator [5, 7], a traversable direction is contained within the current view if a discretized traversable location is within the horizontal field of the current view and within 5 meters Euclidean distance of the current location. This information is acquired by building and parsing the navigation graph of the environments and is provided in the MP3D simulator as well as many other VLN simulators and datasets [19, 38, 41, 9].

The traversability prediction task is formulated as a binary classification problem. The prediction p_{nav} is computed as:

$$p_{trav} = \sigma(\phi_{trav}(f_{enc}(x))), \quad (3)$$

where f_{enc} is the image encoder shared with other auxiliary tasks, ϕ_{trav} is a multi-layer perceptron specific for traversability prediction, and σ is the sigmoid activation function. The loss is simply a binary cross-entropy loss:

$$\mathcal{L}_{trav} = -\frac{1}{N}\sum_{i}^{N} y_{i,trav} \log p_{i,trav} + (1 - y_{i,trav}) \log(1 - p_{i,trav}), \quad (4)$$

where $y_{i,trav}$ and $p_{i,trav}$ are the traversability label and prediction for the i-th training example, respectively, and the loss is averaged over a mini-batch of N examples.

3.2.3 Instance Classification

To follow instructions correctly, the agent has to encode scene information such as kitchen, bedroom, bathroom, and so on, as well as object information such as chair, sofa, TV, etc. In computer vision, instance classification [18, 10] has achieved remarkable progress in representation learning. It has been shown that the representation learned by instance classification transfers well to many downstream tasks, such as object classification, object detection, scene classification. Therefore, we apply instance classification as an auxiliary task in the navigation environments to encode discriminative information for objects and scenes.

As shown in Figure 2c, given an image view x, we generate a query image x_q and a key image x_k by applying image augmentations, such as color jittering, lighting adjustment, affine transform, etc, on the image view x. Given the query image x_q, instance classification task is to identify the corresponding key image x_k (positive sample) from a pool of other image views (negative samples). Similar to MoCo [18], we use a memory bank to increase the number of negative samples by storing the encoded features of training samples from previous mini-batches. We also use the current image encoder to encode x_q and the moving-averaged image encoder to encode x_k. The instance prediction p_{ins} can be computed as:

$$p_{ins} = \frac{\exp(\phi_{ins}(f_{enc}(x_q)) \cdot \hat{\phi}_{ins}(\hat{f}_{enc}(x_k)/\tau))}{\sum_i \exp(\phi_{ins}(f_{enc}(x_q)) \cdot m_i/\tau)}, \quad (5)$$

where f_{enc} is the current image encoder shared with other auxiliary tasks, ϕ_{ins} is a multi-layer perceptron specific for instance classification, $\hat{f}_{enc}$ and $\hat{\phi}_{ins}$ are their moving-averaged version, m_i is the i-th entry (negative samples) in the memory bank, and τ is a scaling factor (temperature). The loss is simply a cross-entropy loss:

$$\mathcal{L}_{ins} = -\frac{1}{N}\sum_{i}^{N} y_{i,ins} \log p_{i,ins}, \quad (6)$$

where $y_{i,ins}$ and $p_{i,ins}$ are the label for positive samples and instance prediction for the i-th training example, respectively, and the loss is averaged over a mini-batch of N examples.

To learn a good visual representation by instance classification, image augmentations play a crucial role [10, 11]. Good image augmentations depend on the downstream task, as well as the form of data on which instance classification is applied [40]. It has been shown that color jittering, Gaussian blur, horizontal flip, and resize crop are particularly useful for learning on datasets with object-centric images such as ImageNet [10]. In navigation environments, however, image views contain multiple objects in a scene. Aggressive resize crop (scale ranged between $[0.2, 1.0]$) may remove important information and lead to ambiguous situations as illustrated in Figure 3. Hence, we use a weak resize crop (scale ranged between $[0.8, 1.0]$) with an affine transform in place of the aggressive resize crop.

Figure 3: Applying aggressive resize crop augmentation that is effective on object-centric images may remove important visual clues in scene images with multiple objects. In this example, it is ambiguous to classify the image in blue with the image in yellow as a positive pair.

3.3. Training Procedure

In this section, we explain how to train the image encoder efficiently and how to train the VLN agent with our SEA pre-trained features.

3.3.1 Image Encoder

We propose a training procedure to reuse the data in a mini-batch across three auxiliary tasks. Without data reuse, we need two training samples for 3D jigsaw, one for traversability, and one for instance classification, which sum up to four training samples. This will be expensive in terms of computation and memory usage especially for loading and training on the high-resolution images in VLN.

Given the i-th image x_i in a mini-batch, in 3D jigsaw we use x_i as the anchor view and sample a query view $x_{i,q}$ from x_i's neighboring views. x_i is reused for traversability prediction. In instance classification, we again reuse x_i as the query image view, and its augmented copy as the key image view $x_{i,k}$. Two images are sampled in total. To further reduce computation, only x_i is fed into the current model for backpropagation. All the other image views including $x_{i,q}$ from 3D jigsaw and $x_{i,k}$ from instance classification are fed into the moving-averaged image encoder inspired by MoCo [18]. In this way, we can drastically save computation since images used for different auxiliary tasks are shared. Furthermore, the features computed by the moving-averaged image encoder reduce memory usage by not constructing the computation graph and reduce computation by not performing backpropagation.

Finally, the image encoder is optimized by the sum of losses from the three auxiliary tasks:

$$\mathcal{L} = \lambda_{jig}\mathcal{L}_{jig} + \lambda_{trav}\mathcal{L}_{trav} + \lambda_{ins}\mathcal{L}_{ins} \quad (7)$$

We empirically set $\lambda_{jig} = \lambda_{trav} = \lambda_{ins} = 1$ without further hyper-parameter tuning.

3.3.2 Agent

After pre-training the image encoder with the proposed auxiliary tasks, we pre-compute the features for each discretized view in the training environments following the convention in [5] and all other VLN works. The VLN agent then uses our pre-computed features of each view in place of the ImageNet pre-trained features for training. By decoupling the training of the image encoder from the VLN agent, other VLN methods can benefit from our improved visual representation with minimal modification.

4. Experiments

4.1. Dataset

In this paper, we propose and validate our method on the Matterport3D (MP3D) simulator [5, 7] and Room-to-Room (R2R) dataset [5] , but the method is applicable in general navigation settings [19, 38, 41, 9] where neighboring image views and traversability information are available.

Dataset for pre-training the image encoder. To pre-train the image encoder, we collect data available in the environments of the MP3D simulator such as RGB image views, a view's neighboring views, traversable directions within a view, etc. Following Anderson et al. [5], at each location, the views are discretized at 30° interval in the range of $[0^\circ, 330^\circ]$ for heading, and $[-30^\circ, 30^\circ]$ for elevation, resulting in 36 views at each location. Following the environment splits in [5], the pre-training dataset is composed of around 275k discretized image views in the Train environments, 34k in the Val-Unseen, and 71k in the Test-Unseen. The image encoder is only pre-trained on data from the Train environments.

Dataset for training the VLN agent. We use the Room-to-Room (R2R) dataset [5] which contains 7,189 training data in the form of human instruction and trajectory pairs. Each trajectory is paired with three instructions. The whole dataset is divided into four sets: Train, Val-Seen, Val-Unseen, and Test-Unseen. The Val-Seen environments are the same as the Train environments but with different navigation instructions. On the other hand, the Val-Unseen and Test-Unseen environments are different with the Train environments and also with different navigation instructions.

4.2. Evaluation

The effectiveness of the pre-trained features is evaluated by the performance of the agent on the target VLN task. Since the training of the image encoder and the agent are decoupled, the performance improvement of the agent can be solely attributed to the improvement of the image representation. The agent is evaluated on both seen (Val-Seen), and unseen (Val-Unseen and Test-Unseen) environments. Even though benchmarking on seen environments (Val-Seen) has been conducted, the primary goal of the VLN agents is to learn to generalize well on unseen environments (Val-Unseen and Test-Unseen). Following [5, 4] and other VLN methods, the agent is evaluated with the following metrics: (1) **TL**: average trajectory length, (2) **NE**: navigation error defined as the average shortest path distance between the agent's final location and the target location, (3) **SR**: success rate defined as the percentage of agent's final location within three meters from the target location, and (4) **SPL**: SR weighted by path length that penalize SR by TL.

4.3. Main Results

We first show that our pre-trained features are superior to the ImageNet pre-trained features, and can boost navigation agents' performance by simply training with our SEA pre-trained features in place of the ImageNet pre-trained features. The agent is evaluated under the single-run setting, where only data from the training environments are available for training both the agent and the image encoder. No extra information is included in comparison with other VLN methods since the image encoder is also pre-trained only on data from the training environments. The single-run setting tests the generalization performance of both the agent and the visual representation to new held-out environments. For the VLN agents, we select Speaker-Follower [14], Env-Dropout [37], and AuxRN [50], and replace the ImageNet pre-trained features with our SEA pre-trained features. We use the released code from these VLN methods and train the agent with our SEA pre-trained features without any hyper-parameter tuning for the agent.

In Table 1, with our SEA pre-trained features, all three agents achieve consistent improvement in Val-Unseen and Test-Unseen. Notably, in Test-Unseen, the most important part of the evaluation since it tests *generalization* performance to new held-out environments, our SEA pre-trained features achieve 12% absolute improvement in both SR and SPL for Speaker-Follower, and 4% for the already strong AuxRN agent. The improvement we obtain can be solely attributed to our pre-trained features, not to the improvement of the agents as we didn't tune the agent at all. We anticipate higher performance is possible with tuning. These results also highlight the importance of visual representation that has long been ignored in the VLN task. Furthermore, since the improvement of the visual representation is orthogonal to the improvement of the agent, other VLN agents and follow-on works can also benefit from and build on our SEA pre-trained features. We will release the pre-

Method	Val-Seen				Val-Unseen				Test-Unseen			
	TL	NE↓	SR↑	SPL↑	TL	NE↓	SR↑	SPL↑	TL	NE↓	SR↑	SPL↑
RCM [43]	10.65	3.53	0.67	-	11.46	6.09	0.43	-	11.97	6.12	0.43	0.38
Self-Monitoring [27]	-	**3.22**	0.67	0.58	-	5.52	0.45	0.32	18.04	5.67	0.48	0.35
Regretful Agent [28]	-	3.23	0.69	0.63	-	5.32	0.50	0.41	13.69	5.69	0.48	0.40
PREVALENT [17]	10.32	3.67	0.69	0.65	10.19	4.71	**0.58**	**0.53**	10.51	5.30	0.54	0.51
Relationship Graph [20]	10.13	3.47	0.67	0.65	9.99	4.73	0.57	**0.53**	10.29	4.75	0.55	0.52
Speaker-Follower [14]	-	3.36	0.66	-	-	6.62	0.35	-	14.82	6.62	0.35	0.28
Env-Dropout [37]	11.00	3.99	0.62	0.59	10.70	5.22	0.52	0.48	11.66	5.23	0.51	0.47
AuxRN [50]	-	3.33	**0.70**	**0.67**	-	5.28	0.55	0.50	-	5.15	0.55	0.51
Speaker-Follower + SEA (ours)	12.80	3.68	0.64	0.56	13.61	5.16	0.51 (+16%)	0.42	14.07	5.42	0.47 (+12%)	0.40 (+12%)
Env-Dropout + SEA (ours)	10.31	3.44	0.69	0.66	9.88	4.76	0.56 (+4%)	0.52 (+4%)	10.18	4.89	0.56 (+5%)	0.53 (+6%)
AuxRN + SEA (ours)	10.28	3.43	0.68	0.65	9.80	**4.55**	0.57 (+2%)	**0.53** (+3%)	10.31	**4.71**	**0.59** (+4%)	**0.55** (+4%)

Table 1: Comparison to other classical VLN methods under the single-run setting, where the image encoder and the agent have no access to unseen environments (Val-Unseen and Test-Unseen) during training. The Speaker-Follower [14], Env-Dropout [37], and AuxRN [50] methods combined with our SEA features achieve significant performance improvement on both Val-Unseen and Test-Unseen sets.

	3D Jigsaw	Traversability	Instance Classif.
Initial accuracy	5.19	64.12	0.62
Final accuracy	50.83	89.85	99.86

Table 2: The classification accuracy (in percentage) of each auxiliary task at the beginning and the end of training.

training dataset, source code, and SEA pre-trained features to facilitate future research in VLN.

4.4. Analysis

Does the image encoder indeed learn to perform well on the auxiliary tasks? Since the agent's improvement is coming from the improved visual representation, which is coming from the training on the three proposed auxiliary tasks, we first verify that the proposed auxiliary tasks are learnable and the image encoder indeed learns to perform well on the tasks. We report the performance (accuracy in percentage) on a held-out validation set at the beginning of training and at the end of training for each auxiliary task. Note that the image views in the held-out validation set are collected from Val-Unseen environments different from the training environments.

The results are shown in Table 2. The image encoder indeed learns to do well on all the auxiliary tasks. The accuracy numbers should not be compared across different auxiliary tasks as they vary in difficulty both because of the number of "categories" but also due to intrinsic difficulty (*e.g.,* jigsaw is known to be harder [12].)

What information is encoded by training on the auxiliary tasks? Now that we know the learned image encoder does well on the auxiliary tasks, we further analyze what information is encoded in the features. Therefore, we conduct ablation studies on the pre-trained image encoder. Specifically, we first train the image encoder with different combinations of auxiliary tasks. We then append a light-weight head to the image encoder and fine-tune only the head (with the image encoder frozen) to downstream tasks: (1) semantic segmentation, (2) normal estimation, (3) multi-label object classification, and (4) scene classification. The training data are taken from a subset of the Taskonomy [47] dataset. Semantic segmentation and normal estimation require structural information of the scene, while multi-label object classification and scene classification require discriminative information of objects and classes.

The results are shown in Table 3. We first compare the full model (#2) and the ImageNet features. The full model performs significantly better in semantic segmentation and normal estimation while maintaining slightly better or comparable performance in multi-label object classification and scene classification. This explains why the agent trained with our SEA pre-trained features performs much better in the target VLN task: *our SEA pre-trained features successfully encode more structural information of the scenes*, which are crucial for performing a navigation task in addition to the discriminative information of objects and scenes. For example, agents frequently make decisions based on instructions like: "turn right when you see the sofa on your left," which requires the understanding of the structure of the scene to successfully follow.

Next, we observe that instance classification (#5,7,8) is the most effective auxiliary task across all downstream tasks. Even though 3D jigsaw and traversability do not perform well by themselves (#3,4), when combined with instance classification (#7,8 compared with #5), they are beneficial for encoding structural information of the scenes as they provide substantial gains in semantic segmentation and normal estimation. Furthermore, 3D jigsaw also provides

	Conditions			Downstream Tasks			
	3D Jigsaw	Traversability Pred.	Instance Classif.	Semantic Segmentation (mAP) ↑	Normal Estimation (RMSE) ↓	Object Classif. (mAP) ↑	Scene Classif. (accuracy) ↑
ImageNet	-	-	-	29.40	0.585	36.63	**71.48**
#2 (all)	✓	✓	✓	**40.27**	**0.523**	**36.86**	69.88
#3	✓			30.32 (-25%)	0.557 (+7%)	27.07 (-27%)	61.42 (-12%)
#4		✓		23.69 (-41%)	0.568 (+8%)	27.32 (-26%)	58.17 (-17%)
#5			✓	35.76 (-11%)	0.545 (+4%)	33.72 (-9%)	69.64 (-0%)
#6	✓	✓		34.12 (-15%)	0.546 (+4%)	29.2 (-21%)	63.67 (-9%)
#7	✓		✓	37.46 (-7%)	0.533 (+2%)	34.21 (-7%)	70.12 (0%)
#8		✓	✓	37.69 (-6%)	0.540 (-3%)	32.56 (-12%)	68.99 (-1%)

Table 3: The analysis of what information is encoded by which auxiliary task. The number in the parenthesis at row #3 - #8 represents the relative difference with respect to the full model with all auxiliary tasks combined (row #2).

	Conditions			Val-Seen				Val-Unseen			
	3D Jigsaw	Traversability Pred.	Instance Classif.	TL	NE↓	SR↑	SPL↑	TL	NE↓	SR↑	SPL↑
#1 (all)	✓	✓	✓	10.31	**3.44**	**0.69**	**0.66**	**9.88**	**4.76**	**0.56**	**0.52**
#2	✓			10.04	4.67	0.57 (-12%)	0.55 (-11%)	9.58	5.22	0.52 (-4%)	0.49 (-3%)
#3		✓		10.15	5.89	0.47 (-22%)	0.44 (-21%)	9.52	5.93	0.47 (-9%)	0.43 (-8%)
#4			✓	10.46	3.74	0.64 (-5%)	0.62 (-4%)	9.93	5.37	0.53 (-3%)	0.49 (-3%)
#5	✓	✓		10.21	4.33	0.62 (-7%)	0.58 (-7%)	9.87	5.08	0.53 (-3%)	0.49 (-3%)
#6	✓		✓	10.41	3.93	0.65 (-4%)	0.62 (-4%)	9.65	4.83	0.55 (-1%)	0.51 (-1%)
#7		✓	✓	10.36	3.82	0.66 (-3%)	0.63 (-3%)	10.21	5.33	0.53 (-3%)	0.49 (-3%)

Table 4: The correlation between agent's navigation performance and the features pre-trained with different sets of auxiliary tasks. The number in the parenthesis at row #2 - #7 represents the absolute difference with respect to the full model (row #1.)

marginal gains in multi-label object classification and scene classification (#7 compared with #5).

How does agent's performance correlate with each auxiliary task? Now that we know which auxiliary task helps encode what kind of information, we would like to assess whether this encoded information is truly beneficial to the agent's final navigation performance. Therefore, we conduct ablation studies under the single run setting with the Env-Drop agent [37]. Specifically, we first train the image encoder with different combinations of auxiliary tasks, use the pre-trained image encoder to generate pre-computed features, and train the agent with the pre-computed features.

The results on Val-Seen and Val-Unseen are shown in Table 4. The agent's performance drops when any of the auxiliary tasks are removed while training the image encoder. Similar to what we've found previously, instance classification (#4,6,7) is the most effective auxiliary task among the three, while 3D jigsaw and traversability are also beneficial as they further improve the performance when combined with instance classification. For example, on top of instance classification, 3D jigsaw helps the agent perform even better on Val-Unseen (#6 compared with #4). Traversability prediction does not help much on Val-Unseen but is beneficial on Val-Seen (#7 compared with #4.) This could be explained by navigation graphs (traversability) of the environments providing a strong prior of the environment layout [23]. Thus, the auxiliary task of traversability prediction learns to encode the prior, adapting to the environments.

5. Conclusion

We propose structure-encoding auxiliary tasks (SEA) to improve the visual representation, long ignored in VLN. Three auxiliary tasks, 3D jigsaw, traversability prediction, and instance classification, are proposed and customized to pre-train the image encoder on data gathered in the navigation environments. 3D jigsaw and instance classification help better encode both structural information of scenes and discriminative information of objects and scenes, while traversability prediction helps better encode structural information and adapt the visual representation to the target navigation environments. The VLN agents combined with our SEA pre-trained features (without tuning) achieve 12% SR improvement for Speaker-Follower, 5% for Env-Dropout, and 4% for AuxRN in test-unseen under the single-run setting. The contributions of proposed auxiliary tasks and SEA pre-trained features are orthogonal to other VLN works, and we will release the collected dataset, source code, and pre-trained features to facilitate further research in visual representations for VLN.

References

[1] Pulkit Agrawal, Joao Carreira, and Jitendra Malik. Learning to see by moving. In *Proceedings of the IEEE international conference on computer vision*, pages 37–45, 2015.

[2] Unaiza Ahsan, Rishi Madhok, and Irfan Essa. Video jigsaw: Unsupervised learning of spatiotemporal context for video action recognition. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 179–189. IEEE, 2019.

[3] Ankesh Anand, Evan Racah, Sherjil Ozair, Yoshua Bengio, Marc-Alexandre Côté, and R Devon Hjelm. Unsupervised state representation learning in atari. In *Advances in Neural Information Processing Systems*, pages 8769–8782, 2019.

[4] Peter Anderson, Angel Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, et al. On evaluation of embodied navigation agents. *arXiv preprint arXiv:1807.06757*, 2018.

[5] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3674–3683, 2018.

[6] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in Neural Information Processing Systems*, 33, 2020.

[7] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. *International Conference on 3D Vision (3DV)*, 2017.

[8] Devendra Singh Chaplot, Ruslan Salakhutdinov, Abhinav Gupta, and Saurabh Gupta. Neural topological slam for visual navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12875–12884, 2020.

[9] Howard Chen, Alane Suhr, Dipendra Misra, Noah Snavely, and Yoav Artzi. Touchdown: Natural language navigation and spatial reasoning in visual street environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12538–12547, 2019.

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

[11] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020.

[12] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE international conference on computer vision*, pages 1422–1430, 2015.

[13] Alexey Dosovitskiy and Vladlen Koltun. Learning to act by predicting the future. In *International Conference on Learning Representations (ICLR)*, 2017.

[14] Daniel Fried, Ronghang Hu, Volkan Cirik, Anna Rohrbach, Jacob Andreas, Louis-Philippe Morency, Taylor Berg-Kirkpatrick, Kate Saenko, Dan Klein, and Trevor Darrell. Speaker-follower models for vision-and-language navigation. In *Advances in Neural Information Processing Systems*, pages 3314–3325, 2018.

[15] Daniel Gordon, Abhishek Kadian, Devi Parikh, Judy Hoffman, and Dhruv Batra. Splitnet: Sim2sim and task2task transfer for embodied visual navigation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1022–1031, 2019.

[16] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in Neural Information Processing Systems*, 33, 2020.

[17] Weituo Hao, Chunyuan Li, Xiujun Li, Lawrence Carin, and Jianfeng Gao. Towards learning a generic agent for vision-and-language navigation via pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13137–13146, 2020.

[18] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020.

[19] Karl Moritz Hermann, Mateusz Malinowski, Piotr Mirowski, Andras Banki-Horvath, Keith Anderson, and Raia Hadsell. Learning to follow directions in street view. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 11773–11781, 2020.

[20] Yicong Hong, Cristian Rodriguez-Opazo, Yuankai Qi, Qi Wu, and Stephen Gould. Language and visual entity relationship graph for agent navigation. *Advances in Neural Information Processing Systems*, 33, 2020.

[21] Haoshuo Huang, Vihan Jain, Harsh Mehta, Alexander Ku, Gabriel Magalhaes, Jason Baldridge, and Eugene Ie. Transferable representation learning in vision-and-language navigation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7404–7413, 2019.

[22] Vihan Jain, Gabriel Magalhaes, Alexander Ku, Ashish Vaswani, Eugene Ie, and Jason Baldridge. Stay on the Path: Instruction Fidelity in Vision-and-Language Navigation. In *Proc. of ACL*, 2019.

[23] Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *European Conference on Computer Vision*, pages 104–120. Springer, 2020.

[24] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Learning representations for automatic colorization. In *European conference on computer vision*, pages 577–593. Springer, 2016.

[25] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Colorization as a proxy task for visual

understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6874–6883, 2017.

[26] Kuang-Huei Lee, Ian Fischer, Anthony Liu, Yijie Guo, Honglak Lee, John Canny, and Sergio Guadarrama. Predictive information accelerates learning in rl. *Advances in Neural Information Processing Systems*, 33, 2020.

[27] Chih-Yao Ma, Jiasen Lu, Zuxuan Wu, Ghassan AlRegib, Zsolt Kira, Richard Socher, and Caiming Xiong. Self-monitoring navigation agent via auxiliary progress estimation. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

[28] Chih-Yao Ma, Zuxuan Wu, Ghassan AlRegib, Caiming Xiong, and Zsolt Kira. The regretful agent: Heuristic-aided navigation through progress estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6732–6740, 2019.

[29] Arjun Majumdar, Ayush Shrivastava, Stefan Lee, Peter Anderson, Devi Parikh, and Dhruv Batra. Improving vision-and-language navigation with image-text pairs from the web. In *European Conference on Computer Vision*, pages 259–274. Springer, 2020.

[30] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision*, pages 69–84. Springer, 2016.

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

[32] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 16–17, 2017.

[33] Deepak Pathak, Ross Girshick, Piotr Dollár, Trevor Darrell, and Bharath Hariharan. Learning features by watching objects move. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2701–2710, 2017.

[34] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2536–2544, 2016.

[35] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[36] Aravind Srinivas, Michael Laskin, and Pieter Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. *arXiv preprint arXiv:2004.04136*, 2020.

[37] Hao Tan, Licheng Yu, and Mohit Bansal. Learning to navigate unseen environments: Back translation with environmental dropout. In *NAACL-HLT 2019: Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 2610–2621, 2019.

[38] Jesse Thomason, Michael Murray, Maya Cakmak, and Luke Zettlemoyer. Vision-and-dialog navigation. In *Conference on Robot Learning*, pages 394–406. PMLR, 2020.

[39] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *European Conference on Computer Vision*, pages 776–794. Springer, 2020.

[40] Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. What makes for good views for contrastive learning? *arXiv preprint arXiv:2005.10243*, 2020.

[41] Arun Balajee Vasudevan, Dengxin Dai, and Luc Van Gool. Talk2nav: Long-range vision-and-language navigation with dual attention and spatial memory. *International Journal of Computer Vision*, 129(1):246–266, 2021.

[42] Xiaolong Wang and Abhinav Gupta. Unsupervised learning of visual representations using videos. In *Proceedings of the IEEE international conference on computer vision*, pages 2794–2802, 2015.

[43] Xin Wang, Qiuyuan Huang, Asli Celikyilmaz, Jianfeng Gao, Dinghan Shen, Yuan-Fang Wang, William Yang Wang, and Lei Zhang. Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6629–6638, 2019.

[44] Xin Wang, Vihan Jain, Eugene Ie, William Yang Wang, Zornitsa Kozareva, and Sujith Ravi. Environment-agnostic multitask learning for natural language grounded navigation. In *European Conference on Computer Vision*. Springer, 2020.

[45] Tete Xiao, Xiaolong Wang, Alexei A Efros, and Trevor Darrell. What should not be contrastive in contrastive learning. *arXiv preprint arXiv:2008.05659*, 2020.

[46] Joel Ye, Dhruv Batra, Erik Wijmans, and Abhishek Das. Auxiliary tasks speed up learning pointgoal navigation. *arXiv preprint arXiv:2007.04561*, 2020.

[47] Amir R. Zamir, Alexander Sax, William B. Shen, Leonidas J. Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2018.

[48] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European conference on computer vision*, pages 649–666. Springer, 2016.

[49] Nanxuan Zhao, Zhirong Wu, Rynson WH Lau, and Stephen Lin. What makes instance discrimination good for transfer learning? *arXiv preprint arXiv:2006.06606*, 2020.

[50] Fengda Zhu, Yi Zhu, Xiaojun Chang, and Xiaodan Liang. Vision-language navigation with self-supervised auxiliary reasoning tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10012–10022, 2020.

[51] Barret Zoph, Golnaz Ghiasi, Tsung-Yi Lin, Yin Cui, Hanxiao Liu, Ekin Dogus Cubuk, and Quoc Le. Rethinking pre-training and self-training. *Advances in Neural Information Processing Systems*, 33, 2020.