Uncovering the Inner Workings of STEGO for Safe Unsupervised Semantic Segmentation

Alexander Koenig Maximilian Schambach Johannes Otterbach

Merantix Momentum

{firstname.lastname}@merantix.com

Abstract

Self-supervised pre-training strategies have recently shown impressive results for training general-purpose feature extraction backbones in computer vision. In combination with the Vision Transformer architecture, the DINO self-distillation technique has interesting emerging properties, such as unsupervised clustering in the latent space and semantic correspondences of the produced features without using explicit human-annotated labels. The STEGO method for unsupervised semantic segmentation contrastively distills feature correspondences of a DINO-pre-trained Vision Transformer and recently set a new state of the art. However, the detailed workings of STEGO have yet to be disentangled, preventing its usage in safety-critical applications.

This paper provides a deeper understanding of the STEGO architecture and training strategy by conducting studies that uncover the working mechanisms behind STEGO, reproduce and extend its experimental validation, and investigate the ability of STEGO to transfer to different datasets. Results demonstrate that the STEGO architecture can be interpreted as a semantics-preserving dimensionality reduction technique.

1. Introduction

Semantic segmentation is the task of assigning pixel-wise class labels to an image. Its applications range from scene understanding for autonomous navigation [19, 43], environmental monitoring [2,4], to medical imaging [32,34] to name a few. Current state-of-the-art deep learning-based semantic segmentation approaches are trained in a supervised fashion and thus require a large amount of training data, *i.e.*, explicit pixel-wise target labels, also known as segmentation maps. However, freely available open-source datasets with dense segmentation labels for industrial applications are scarce and costly to obtain as opposed to per-image class labels such as those used in popular image datasets like ImageNet [33], CIFAR [25], or MNIST [15].

To alleviate this shortcoming, researchers recently turned to self-supervised learning methods to retrieve semantic information from images without using human-annotated labels. The field recently gained traction with self-supervised representation learning algorithms such as SimCLR [10], BYOL [18], and SwAV [8], with which feature extraction backbones can be learned without the explicit use of target labels. The image representations generated by these backbones can then be used for various downstream tasks. Among those algorithms, the DINO pre-training strategy [9] is particularly notable. Combined with the recent Visual Transformer (ViT) architecture [16], the DINO pre-training objective produces semantically consistent patch-wise image embeddings that can be effectively used for various downstream tasks such as image classification [9], part co-segmentation [3], video instance segmentation [9], or neural rendering [39]. In the remainder, we refer to a ViT trained using the DINO pre-training strategy as the DINO backbone, or simply, DINO. These feature extraction backbones also facilitate self-supervised semantic segmentation. For instance, the STEGO architecture and training strategy [20] builds on DINO and post-processes its expressive features through a self-supervised correlation loss, which distills feature correspondences. Despite being compute-efficient by freezing the DINO backbone, the STEGO approach demonstrates impressive capabilities on unsupervised semantic segmentation benchmarks.

Given the promises of these methods, it is essential to understand their transferability to real-world settings before deployment to ensure reproducibility, robustness, and safety. However, looking closely at STEGO, we struggled to reproduce some of the results reported in the original publication [20]. Additionally, self-supervised learning mechanisms are intricate and still poorly understood. In particular, experimental evaluation has yet to clearly separate contributions in ablation studies and benchmark performance across multiple datasets to ensure robust performance in safety-critical applications. Concerning self-supervised segmentation using STEGO, we identified some

missing benchmark results and ablation studies investigating the contribution of this new approach in more detail.

In this paper, we provide a deeper understanding of the STEGO architecture and training strategy to ensure its well-informed usage and motivate further development. Particularly, our contributions are as follows.

1. We reproduce and extend STEGO's experimental validation, demonstrating a stronger baseline performance of the DINO backbone than reported in the STEGO paper [20] and reporting missing evaluation metrics for a more comprehensive evaluation.

2. We disentangle STEGO's working mechanisms and attribute its superior unsupervised performance to both dimensionality reduction of the backbone's embeddings and a non-linear projection that yields more semantically clustered representations.

3. We compare STEGO with well-established dimensionality reduction techniques across different dimensions and datasets, revealing STEGO's better suitability for unsupervised clustering and providing insights into its sensitivity to the target dimension.

2. Background

2.1. Related Works

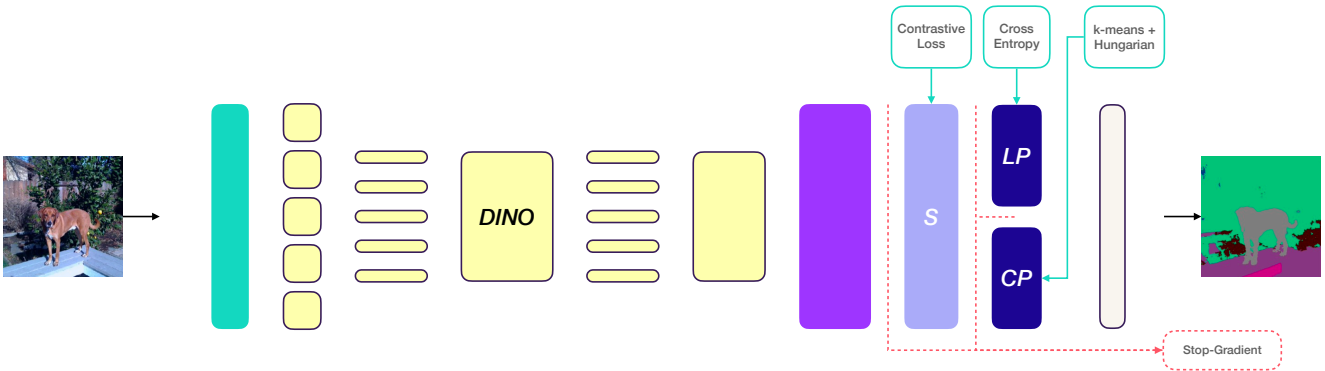
Methods addressing label scarcity in deep semantic segmentation can broadly be taxonomized into *weakly supervised*, and *unsupervised* methods [36]. Weakly supervised algorithms for semantic segmentation often rely on coarse supervision, *e.g.*, by training a segmentation model only through image-level annotations [1, 42] or bounding box labels [30]. Other weakly supervised approaches use incompletely labeled segmentation datasets, also known as semi-supervision [27].

On the other hand, unsupervised semantic segmentation algorithms use no explicit labels. A core idea behind unsupervised semantic segmentation is clustering dense visual descriptors of an image into perceptual groups. Some algorithms frame this task as a graph partitioning problem, either on the input image [37] or, more recently, on deep ViT-features [29]. Furthermore, standard k-means-style clustering [3], or matrix factorization [12] of features from pre-trained backbones were successfully used to detect related ontologies for unsupervised semantic segmentation. Other methods use self-supervision to learn visual descriptors for unsupervised segmentation. For instance, PiCIE [11] clusters pixel-level features by optimizing a self-supervised loss that enforces invariance to photometric and equivariance to geometric transformations. Self-supervision is also often used for feature distillation of pre-trained backbones before clustering. For example, the MaskContrast [40] algorithm uses a pre-trained network to extract a binary mask of an image's main object, also referred to as saliency estimation, contrastively trains a separate network that maximizes agreement of the pixel embeddings with the extracted object mask, and finally clusters the pixel embeddings using k-means. MaskDistill [41] uses object masks from clustered ViT-features as pseudo-ground-truth labels to train a second network on high-confidence segmentation masks. While previous feature distillation approaches introduce a proxy network, the STEGO [20] algorithm directly utilizes a ViT backbone, distills the produced latent features, and clusters them into distinct ontologies – promising a simple and efficient training strategy.

2.2. STEGO Architecture

The architecture of the STEGO pipeline is given in Figure 1. The approach uses an ImageNet-pre-trained DINO backbone [9] for feature extraction. Before being fed into the ViT, an input image $\mathbf{x}$ is patched into 8×8 image patches which are subsequently mapped into a latent representation using a simple linear layer. A learnable positional encoding is then added to these embedded patches, also called tokens. Multiple Transformer layers then process these tokens. Finally, the ViT's output tokens are reshaped into a D_{ViT}-dimensional image-like feature map $\mathbf{f}_{\text{ViT}}$. The STEGO architecture adds three main modules on top of DINO. First, an optional bilinear upsampling layer is added to upsample the feature map by a factor of 8 to regain the original input image resolution. Then, an unsupervised segmentation head projects the DINO features from D_{ViT} to D_{STEGO} dimensions. Here, the dimensionality D_{ViT} of the ViT features depends on the choice of the backbone ($D_{\text{ViT}} = 384$ for ViT-Small, 768 for ViT-Base) and the target dimension D_{STEGO} is a hyper-parameter. Lastly, a cluster or linear probe is trained with few labels to assign each pixel a probability of each of the N_{C} semantic target classes. The probes evaluate the feature quality of the segmentation head's output. A stop-gradient operation ensures that the simultaneous optimizations of both probes and the segmentation head do not influence each other and that supervised label information propagates neither into the segmentation head nor the backbone. Finally, the output is optionally refined using a Conditional Random Field (CRF) [24].

As shown in Figure 2, the STEGO segmentation head consists of several linear layers and a non-linear ReLU [17] activation function. The segmentation head processes each feature separately by applying a 1×1 convolution to the feature map. That is, neighboring features do not influence each other in this head. Moreover, the module performs a projection of the D_{ViT}-dimensional input vector into a D_{STEGO}-dimensional output embedding space. The authors propose to reduce the channel dimension in the segmentation head, *i.e.*, $D_{\text{STEGO}} < D_{\text{ViT}}$, and specify $D_{\text{STEGO}} = 70$.



Operation	Read Image	Patch Image	Embed Tokens	...	Output Tokens	Reshape Tokens	Upsample Tokens	Segment. Head	Linear Probe / Cluster Probe	CRF Refine
Output	$\mathbf{x}$					$\mathbf{f}_{\text{ViT}}$		$\mathbf{f}_{\text{STEGO}}$		
# Outputs	1	40^2	40^2		40^2	1	1	1	1	1
Channels	3	3	D_{ViT}		D_{ViT}	D_{ViT}	D_{ViT}	D_{STEGO}	N_c	N_c
Height	320	8	1		1	40	320	320	320	320
Width	320	8	1		1	40	320	320	320	320

Figure 1. Architecture of the STEGO validation pipeline. The top of the figure shows abstractions of the operations and their training losses. The table indicates each operation's outputs and their dimensions. $\mathbf{x}$ is the input image, *DINO* is the frozen ViT backbone, $\mathbf{f}_{\text{ViT}}$ is an image-like feature map (*i.e.*, reshaped ViT tokens), S is the segmentation head, and the Linear Probe *(LP)* and Cluster Probe *(CP)* evaluate the segmentation head output $\mathbf{f}_{\text{STEGO}}$ and map to N_c class probabilities. The stop-gradient module ensures that the optimizations of S, *LP*, and *CP* do not influence each other and that label information can neither propagate into S nor *DINO*. We display the validation pipeline, which uses 320×320 images. The training pipeline processes 224×224 images, does not upsample the ViT feature map $\mathbf{f}_{\text{ViT}}$, and does not use CRF by default.

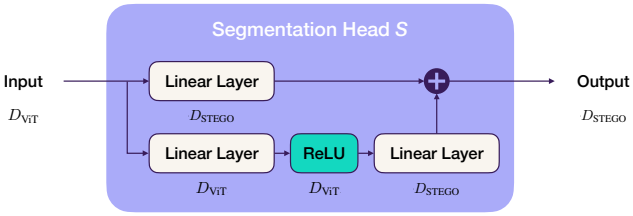


Figure 2. Architecture of the non-linear segmentation head S. The module projects the dimension of a vector from D_{ViT} to D_{STEGO}.

2.3. STEGO Training Strategy

Hamilton *et al.*'s [20] contrastive framework is based on ViT token pairs as opposed to image pairs which are often used in the context of contrastive self-supervised training strategies in computer vision [9, 10]. Hamilton *et al.* define token pairs using the cosine similarity of two ViT tokens: A pair is considered positive when the cosine similarity of two tokens exceeds a certain threshold; otherwise, they are considered negative. Their *contrastive correlation loss* is used to train the segmentation head and push the features of positive token pairs towards alignment, corresponding to a cosine similarity of 1, while negative pairs are pushed towards a cosine similarity of –1.

The loss is implemented as follows. First, they assemble a pair of images $(\mathbf{x}, \mathbf{x}')$ and obtain the corresponding DINO feature maps $(\mathbf{f}_{\text{ViT}}, \mathbf{f}'_{\text{ViT}})$ as well as segmentation head outputs $(\mathbf{f}_{\text{STEGO}}, \mathbf{f}'_{\text{STEGO}})$. Subsequently, they calculate the cosine similarities between all token pairs $\mathbf{f}_{\text{ViT}}$ and $\mathbf{f}'_{\text{ViT}}$ to obtain a feature correspondence tensor $\mathbf{C}_{\text{ViT}}$. Each entry $\mathbf{C}_{\text{ViT},hwij} \in [-1, 1]$ in the feature correspondence tensor is the cosine similarity between a token in $\mathbf{f}_{\text{ViT}}$ and a token in $\mathbf{f}'_{\text{ViT}}$ where (h, w) and (i, j) represent the spatial locations of the tokens in the two feature maps, respectively. Analogously, they calculate the correspondence tensor $\mathbf{C}_{\text{STEGO}}$ of the segmentation head outputs $(\mathbf{f}_{\text{STEGO}}, \mathbf{f}'_{\text{STEGO}})$. Their contrastive correlation loss

$$L_{\text{corr}}(\mathbf{x}, \mathbf{x}', b) = -\sum_{hwij} (\mathbf{C}_{\text{ViT},hwij} - b)\mathbf{C}_{\text{STEGO},hwij} \quad (1)$$

pushes an entry in the STEGO feature correspondence tensor $\mathbf{C}_{\text{STEGO}}$ towards 1 if the respective entry in $\mathbf{C}_{\text{ViT}}$ exceeds the threshold b in the case of a positive token pair and towards –1 otherwise. The parameter b can be interpreted as a form of negative pressure. The paper mentions several strategies to stabilize the loss and prevent collapse, such as spatial centering of the correspondence tensor $\mathbf{C}_{\text{ViT}}$ and zero-clamping of the correspondence tensor $\mathbf{C}_{\text{STEGO}}$.

The choice of image pairs $(\mathbf{x}, \mathbf{x}')$ contributes towards the amount of positive and negative learning signal from the loss function. In total, the loss uses three image pairs: the image with itself $(\mathbf{x}, \mathbf{x})$, the image with one of its 7 nearest neighbors $(\mathbf{x}, \mathbf{x}_{\text{knn}})$, and the image with a randomly chosen one $(\mathbf{x}, \mathbf{x}_{\text{rand}})$. While the first two image pairs provide a

positive training signal, since many token pairs have similar features, the latter one contributes mostly negative signals because most token pairs will likely be dissimilar, assuming the dataset is large and diverse. Finally, the overall loss

$$L = \sum_{i \in \{\text{self,knn,rand}\}} \lambda_i L_{\text{corr}}(\mathbf{x}, \mathbf{x}_i, b_i) \quad (2)$$

is calculated as the weighted sum of the correlation losses of the three individual pairs. Overall, the STEGO loss introduces six hyperparameters, λ_{self}, λ_{knn}, λ_{rand}, b_{self}, b_{knn}, and b_{rand}, which require careful tuning.

3. Reestablishing Baselines

Hamilton *et al*. show that STEGO significantly outperforms other unsupervised segmentation algorithms such as PiCIE [11] on the Cocostuff-27 [7] dataset. Notably, they demonstrate that STEGO outperforms the DINO baseline, where segmentation is performed on the DINO features directly. However, our preliminary experiments had indicated that the performance of the DINO backbone alone is already highly competitive, *i.e*. the DINO backbone produces highly semantic features – an observation in line with the original presentation [9]. This discrepancy motivated us to conduct a controlled reproducibility study of the result stated in the original STEGO paper [20].

3.1. Experimental Setup

Evaluation, datasets, metrics. Recent works on self-supervised learning evaluate the expressivity and generalizability of features on several downstream tasks. Common evaluation protocols in computer vision involve training linear or k-NN classifiers using the ImageNet benchmark [9, 10] or semantic segmentation challenges [18]. We adhere to the protocol in the original presentation to evaluate the features produced by the STEGO segmentation head.

Hamilton *et al*. [20] evaluate the segmentation head features in two styles. First, they propose an *unsupervised cluster probe*, *i.e*. a cosine similarity-based k-means algorithm [28] that detects N_{C} clusters in the segmentation head's representation space. The identified clusters are mapped to human-interpretable labels using a linear sum assignment solved via the Hungarian algorithm [26] on the ground-truth labels. Second, they assess the general feature quality using a *supervised linear probe*, *i.e*. a linear layer processing the extracted fixed segmentation head outputs, which is trained via a standard cross-entropy loss with labels from the ground-truth segmentation map.

To stay consistent with the original work, we use the Cocostuff-27 [7], Cityscapes [13], and Potsdam-3 [22] datasets for all of our experiments. We report the validation mean Intersection over Union (mIoU) and accuracy as the primary evaluation metrics to be directly comparable to the original presentation. We follow the exact evaluation procedure from the STEGO repository by reporting results on the validation set and cherry-picking models performing best on the validation unsupervised mIoU benchmark.

Parameter	Configuration			
	Actual			Reported
Dataset	Cocostuff	Cityscapes	Potsdam	*
Train steps	7000	7000	5000	-
Batch size	32	32	16	32
Crop type	5-crop	5-crop	No crop	5-crop
Backbone	ViT-B	ViT-B	ViT-S	ViT-B
0-clamp	-	-	True	True
Pointwise	True	False	True	True
D_{STEGO}	90	100	70	70
λ_{rand}	0.15	0.91	0.63	*
λ_{knn}	1.00	0.58	0.25	*
λ_{self}	0.10	1.00	0.67	*
b_{rand}	1.00	0.31	0.76	*
b_{knn}	0.20	0.18	0.02	*
b_{self}	0.12	0.46	0.08	*

Table 1. Training configuration extracted from the pre-trained models of STEGO and configuration reported in the paper [20]. The first 5 parameters are generic training parameters that also apply to our "non-STEGO experiments" (*i.e*., our baseline experiments from Section 3 and the experiments in Section 4). The remaining parameters are STEGO-specific training parameters. We show only parameters that are different between the datasets or deviate from the original paper's information. For the remaining parameters, we refer readers to our supplementary material. Note that "-" means the information was unavailable in the code or the paper, and "*" means multiple results were reported.

Model configuration. Tab. 1's middle column shows the actual training configuration we retrieved from each dataset's pre-trained STEGO model shipped with the paper's source code[1]. The table's right-most column mentions the parameters given in their report. Notably, some information in the *actual configuration* differs from the data reported in the original publication.

The data in Tab. 1 offers other interesting insights. The STEGO embedding dimension D_{STEGO} is different for all datasets, and the dimension reported in the paper only matches the Potsdam dataset. The training configuration for the Potsdam dataset differs from the other ones, *i.e*. using a smaller backbone, fewer training steps, no five-cropping of training images, and a smaller batch size. Furthermore, the loss hyperparameters are remarkably different for each dataset. We also report the hyperparameters for the Potsdam dataset here, which are not given in the original paper [20].

[1] github.com/mhamilton723/STEGO

Method	Cocostuff				Cityscapes				Potsdam			
	Unsupervised Cluster probe		Supervised Linear probe		Unsupervised Cluster probe		Supervised Linear probe		Unsupervised Cluster probe		Supervised Linear probe	
	Acc	mIoU	Acc	mIoU	Acc	mIoU	Acc	mIoU	Acc	mIoU	Acc	mIoU
STEGO (theirs)	56.9	28.2	76.1	41.0	73.2	21.0	-	-	77.0	-	-	-
STEGO (ours)	✓56.9	✓28.2	✓76.1	✓41.1	✓73.2	✓21.0	89.6	28.0	✓77.0	62.6	85.9	74.8
DINO (theirs)	30.5	9.6	66.8	29.4	-	-	-	-	-	-	-	-
DINO (ours)	†42.4	†13.0	†75.8	‡44.4	52.6	15.2	‡91.3	‡34.9	71.3	54.3	84.5	72.8

Table 2. Validation results of reproducibility study showing accuracy ("Acc") and mIoU in % for the different evaluation styles and datasets. Values reported in the original paper [20] are marked "theirs", and missing ones are denoted with "-". Our results evaluating the pre-trained models are marked "STEGO (ours)". "DINO (ours)" is a re-training of the probes directly on the DINO backbone. Successful reproduction is denoted with ✓ and a stronger DINO performance than reported in [20] is indicated with † and ‡. All evaluations post-process with CRF.

We use the "actual configuration" retrieved from the pre-trained models without performing additional hyperparameter optimizations to remain comparable with Hamilton *et al.*'s [20] results. Our source code is publicly available[2] and only applies minimal changes to the original code, *e.g.*, for more extensive logging, which are clearly highlighted.

3.2. Results and Discussion

Tab. 2 displays the results of our reproducibility study across all three benchmarking datasets. First, we show that using the pre-trained STEGO models provided by the authors, we are able to reproduce the results from the paper (highlighted with ✓), which validates our evaluation pipeline. Moreover, we paint a more comprehensive picture by reporting missing metrics for the Cityscapes and Potsdam datasets.

Second, we directly evaluate the features produced by the DINO backbone by re-training the cluster and linear probe on the backbone while fixing all other training settings. We find that the DINO baseline results are consistently better than those provided in the paper [20] (highlighted with †). This is surprising as we use the evaluation pipeline provided by the authors and do not introduce changes that could influence the training performance.

Notably, for linear probe-style evaluation, the DINO baseline performs approximately on par with the STEGO approach, sometimes even outperforming STEGO (highlighted with ‡). This result means that the features produced by DINO and the STEGO segmentation head exhibit similar linear separability. DINO demonstrates a remarkable ability to provide semantically meaningful features and generalize to different datasets despite being pre-trained solely on ImageNet [33], while Hamilton *et al.* [20] fine-tuned the segmentation head on the respective datasets. DINO's performance is particularly noteworthy, given the significant domain shifts between ImageNet and Potsdam's aerial imagery and ego-camera footage from Cityscapes.

However, as shown in Tab. 2, the STEGO embeddings are better suited for the unsupervised cluster probe task than the raw DINO features. Hamilton *et al.* argue that "*[d]ue to the feature distillation process, STEGO's segmentation features tend to form clear clusters*" (page 6, [20]). While more "*distinct clusters*" (page 1, [20]) could explain STEGO's superior k-means clustering performance, there might be additional possible explanations.

First, STEGO can adapt to a domain shift to the new data distribution since it includes a trainable segmentation head. It needs to be clarified what the contribution of the STEGO loss for adapting to new datasets really is, as opposed to the DINO pre-training objective, which could also be used to fine-tune the backbone. For instance, can one fine-tune the ViT backbone – either entirely, using adapters [21], or additional linear heads – using the DINO loss on a new dataset and obtain a performance competitive with STEGO? As Caron *et al.* [9] show, DINO provides semantically meaningful representations. The DINO loss is better understood due to its pervasion in the community and involves less hyperparameter tuning.

Second, STEGO's dimensionality reduction from D_{ViT} to D_{STEGO} (a factor of ≈ 8 for Cocostuff and Cityscapes) could in itself be a reason for a higher unsupervised clustering performance. The k-means algorithm used for clustering minimizes the sum of squared distances between each data point and its cluster centroid. Due to the curse of dimensionality, distances between data points in higher dimensions become exponentially larger and more uniform. This observation also holds for the cosine distance-based variant of k-means applied in the STEGO paper. Less significant distances make it harder to identify clusters in higher dimensional spaces. Hence, it could be that STEGO preserves the semantics of DINO features while projecting them into a lower-dimensional space where k-means per-

[2] github.com/merantix-momentum/stego-studies

forms better. Again, the question remains whether a similar result could be obtained by training a feature downsampling layer or segmentation head using the DINO strategy.

The STEGO paper [20] does not explore these issues experimentally. Particularly, the research community has yet to disentangle the mechanisms behind STEGO's superior performance in the unsupervised clustering case with experimental results. In the remainder of this paper, we take a closer look at STEGO and test whether the method can be interpreted as a dimensionality-reduction technique that preserves DINO's semantic feature correspondences.

4. Disentangling STEGO's Working Principles

We work towards disentangling the functional mechanisms behind STEGO by re-training the segmentation head with different output dimensions D_{STEGO}. Consequently, we draw conclusions from STEGO's performance on the linear and cluster probe downstream tasks. These ablations also give practitioners an intuition for how sensitive STEGO is towards its output dimension D_{STEGO} and serve as another reproducibility experiment. Previously, we argued that STEGO may be interpretable as a semantics-preserving dimensionality reduction technique. Hence, we compare STEGO with two standard methods that can reduce the dimensionality of the high-dimensional ViT features.

4.1. Dimensionality Reduction Baselines

Principal component analysis. Principal component analysis (PCA) [31] is a well-established unsupervised dimensionality reduction technique. First, PCA identifies a set of orthogonal vectors, known as principal components, that capture the most variance in the dataset. Secondly, the algorithm uses the top-k principal components to linearly project the dataset into a lower-dimensional space.

Figure 3 shows the cumulative explained variance of the principal components of the DINO embeddings for all three investigated datasets. For Cityscapes, PCA can explain roughly 75 % of the dataset's variance using only roughly 33 % of the number of components. For the other datasets, more components are required to explain the same variance. The steep cumulative explained variance curve, especially for Cityscapes, indicates great potential for dimensionality reduction techniques operating on the ViT features.

Random Projection. Another compute-efficient dimensionality reduction technique commonly used in the machine learning context is Random Projection (RP) [5, 14], which linearly transforms high-dimensional feature vectors into a randomly chosen subspace. We use Gaussian RP, which initializes the projection matrix with orthogonal column vectors drawn from a normal distribution. The algorithm approximately retains pair-wise distances with high probability [23], which makes RP well-suited for clustering with k-means [6].

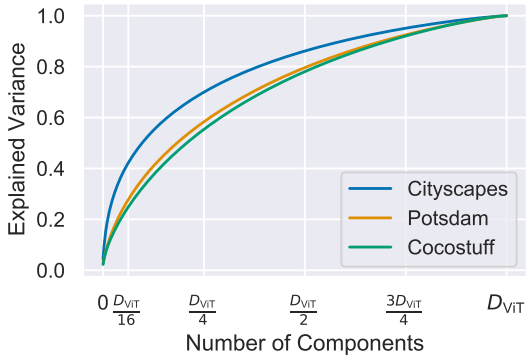


Figure 3. Cumulative explained variance of the principal components of DINO features for all datasets. We fit PCA on at most 5000 randomly selected training images, *i.e.*, $\approx$ 3M ViT-tokens, to keep the memory consumption tractable. The x-axis is normalized as different embedding dimensions are used for the three datasets.

4.2. Results and Discussion

Figure 4 shows our results, which analyze the embedding dimension hyperparameter of various unsupervised semantic segmentation techniques. As previously noted, we use three methods to reduce the dimensionality of the DINO features: the segmentation head of STEGO, PCA, and RP. In the latter two cases, we swap out STEGO's segmentation head and train the linear and cluster probes on the PCA- or RP-transformed output directly while keeping the rest of the training and validation pipeline fixed. We observe several interesting results.

For linear probing in Figure 4 (a–c), STEGO can reduce the dimensionality with little loss in performance up to approximately D_{STEGO}. This means that STEGO is an efficient non-linear dimensionality reduction technique. However, through an ablation across multiple dimensions, we confirm our experimental result from Tab. 2, indicating that STEGO provides little, if any, performance benefit over the DINO baseline for linear probe-style evaluation. By analyzing STEGO's performance at the unreduced DINO dimension, *i.e.*, $D_{\text{STEGO}} = D_{\text{ViT}}$, we can isolate the segmentation head's dimensionality reduction from its non-linear projection. Notably, features produced by the STEGO segmentation head at the unreduced dimension are less or equally linearly separable than DINO features. Hence, the segmentation head does not project DINO's features in a useful way for linear probing. However, it should be noted that the hyperparameters of the STEGO training strategy are not fine-tuned and thus are strictly only compatible with the original output dimension D_{STEGO}. Hence, the reduced performance at the full dimension could also be due to hyperparameter instability. Finally, STEGO performs better or at least on par with the linear dimensionality reduction techniques on the linear probing task.

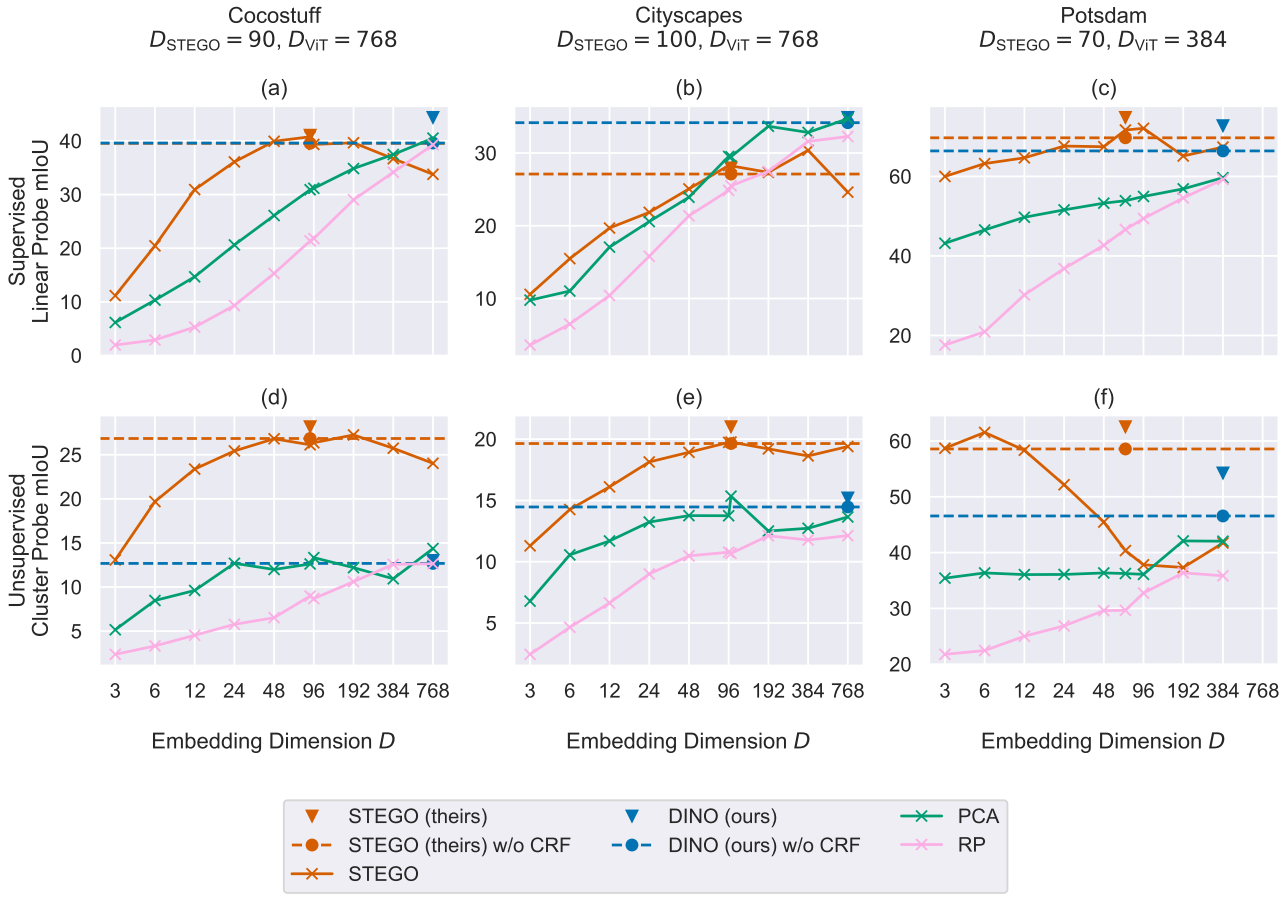


Figure 4. Validation mIoU of different dimensionality reduction techniques with cluster- and linear probe-style evaluation across different embedding dimensions. The ▼ data points are the results from Tab. 2. The • data points show the same results without the CRF post-processing. All remaining data is without CRF to paint an undistorted picture of the effectiveness of the underlying segmentation pipeline. The × data points show the results of models trained with different embedding dimensions and dimensionality reduction algorithms.

Figure 4 (d, e) shows that STEGO consistently outperforms PCA, RP, and plain DINO across all embedding dimensions in the unsupervised clustering task. This demonstrates that STEGO can robustly project ViT features into a wide range of target dimensions – even into much lower ones than suggested in the original paper [20]. Hence, the segmentation head preserves semantic information for the clustering task while simultaneously reducing the number of embedding dimensions. However, STEGO's superior performance for unsupervised clustering cannot only be attributed to its dimensionality-reducing aspect since we also see a performance boost in Figure 4 (d, e) at the original dimension $D_{\text{STEGO}} = D_{\text{ViT}} = 768$. This observation could be evidence of Hamilton *et al.*'s argument of more compact and distinct clusters in features produced by STEGO that could facilitate better k-means convergence. However, STEGO's improvement over DINO may also be due to the adaptation of the segmentation head to the new training data distribution. Training the segmentation head or Transformer adapter layers [21] using a simpler loss (*e.g.*, the DINO loss) could potentially achieve comparable performance. Nevertheless, we conclude for the unsupervised downstream task: Besides efficiently reducing the dimension, STEGO also projects DINO's output features to representations that are more suited for k-means clustering. In contrast, we did not observe a benefit of STEGO for linear probing over the DINO baseline at an unreduced dimension.

Figure 4 (d, e) also demonstrates that PCA approximately retains the unsupervised performance of DINO up to an embedding dimension of $D = 48$ – a significant reduction by a factor of 16 compared to the original dimension $D_{\text{ViT}} = 768$ of the DINO baseline. Simultaneously, in Figure 4 (a–c), the performance of the PCA-transformed features on the linear probing downstream task approximately exponentially decreases with lower embedding dimensions (note that the x-axis is in log-scale). From Figure 3, we know that the first 48 components account for roughly 40 % of variance explained for Cityscapes and

25 % for Cocostuff. Therefore, lower performance of the PCA-projected features on the linear probing task is expected since the transformed features contain much less information. Surprisingly, the unsupervised clustering performance does not worsen until $D = 48$ for Cocostuff and Cityscapes, even though significantly less information is available. Hence, another factor is at play, improving performance for lower embedding dimensions. This factor likely is related to our previous argument that the k-means algorithm converges better in lower dimensions.

Interestingly, all STEGO results in Figure 4 follow a similar trend, where the cluster and linear performance peaks at a dimension $D_{\text{STEGO}} \ll D_{\text{ViT}}$. Furthermore, Figure 4 (d, e) shows a similar trend for the PCA cluster performance, which also has a slight upwards trend at approximately D_{STEGO}. These results suggest some optimal configuration for the k-means algorithm, where both effects – lower dimensions mean less information content, but a better k-means convergence – balance out, again indicating that the dimensionality reduction is a critical contribution to STEGO's performance in the unsupervised clustering case. Another explanation for the peak STEGO performance could be that the hyperparameters, which Hamilton *et al*. [20] identified, work best for D_{STEGO} and that hyperparameter tuning influences the results at other dimensions.

As shown in Figure 4 (a, b, d, e), we are able to reproduce the originally reported performance of STEGO without CRF for Cocostuff and Cityscapes. However, the Potsdam results in Figure 4 (f) behave quite differently than those on the more diverse and larger datasets. While in our previous analysis, we found that all STEGO cluster probe results show a conceptually similar trend (*i.e*., performance peaks at a dimension $D \ll D_{\text{ViT}}$ and they robustly outperform the baselines), the entire Potsdam curve is shifted to the left and does not meet the expected performance of the pre-trained STEGO model. This vastly different behavior is likely due to two reasons. First, the Potsdam aerial imagery dataset is out of distribution input for the ImageNet-pre-trained DINO backbone, likely leading DINO to produce lower-quality embeddings. Second, the segmentation head is trained on significantly fewer data for Potsdam than Cityscapes and Cocostuff, making it more challenging for the module to compensate for lower-quality DINO features. Note that Potsdam is inherently smaller than Cityscapes and Cocostuff containing only roughly 4.5k training images as opposed to 15k and 490k pre-processed training images for Cityscapes and Cocostuff, respectively. Nevertheless, it is impressive that STEGO achieves over 60% mIoU using only 6-dimensional embeddings in Figure 4 (f). A likely reason for much better relative performance at lower dimensions is the considerably easier segmentation task having only 3 classes for Potsdam, as opposed to 27 for Cocostuff and Cityscapes. Separating these three classes using the cluster probe in very high dimensions is challenging, again supporting our argument that STEGO is an efficient dimensionality reduction technique.

Finally, due to computational efficiency, we cache the precomputed DINO features using the Squirrel library [38] for PCA and RP. Therefore, we apply a slightly different shuffling than the one used in the case of STEGO. The shuffling styles can influence the performance of the mini-batch k-means variant and the linear probe results, which might explain why the PCA and RP results in Figure 4 sometimes do not match the DINO performance at D_{ViT} exactly. Note that for all STEGO results shown in Figure 4, we run the original code with the unaltered shuffling strategy.

Minor results in Figure 4 include that all other dimensionality reduction techniques in the linear and cluster probe downstream task outperform RP. This is somewhat surprising as RP approximately preserves pair-wise distances. A possible explanation might be the use of a cosine similarity-based k-means clustering. Investigating different unsupervised clusterings, such as a standard Euclidian distance-based k-means or more advanced clustering approaches, is an interesting direction for future investigations. However, fitting RP is exceptionally fast and compute-efficient. Finally, CRF improves performance by a few percent, as indicated in the STEGO paper [20] and other works in the field.

5. Conclusion

This work provided a deeper understanding and evaluation of the recent STEGO method for unsupervised semantic segmentation. We uncovered the working mechanisms behind the architecture's segmentation head, highlighted a remarkable performance of the DINO backbone, and found that STEGO compares favorably against traditional dimensionality reduction techniques. A limitation of our analysis is that we did not optimize STEGO's loss parameters for each embedding dimension.

Future research should detail the relationship between the loss parameters and task performance. Additionally, it is essential to investigate whether self-supervised fine-tuning of the DINO backbone on new datasets can yield comparable results to STEGO. Lastly, it is interesting to explore other clustering strategies, such as over-clustering of the features with k-means as proposed in MaskDistill [41], or using different clustering algorithms that perform well in high-dimensional spaces such as h-NNE [35] for which we show first results in the supplementary material.

Acknowledgements. This work was supported by the Federal Ministry for Economic Affairs and Climate Action (BMWK) on the basis of a decision by the German Bundestag as part of the safe.trAIn project under grant no. 19I21039G.

Contributions. All authors designed the study. A.K. implemented and carried out the experiments. A.K. and M.S. wrote the manuscript. M.S. and J.O. supervised the work.

References

[1] Jiwoon Ahn and Suha Kwak. Learning pixel-level semantic affinity with image-level supervision for weakly supervised semantic segmentation. In *Conference on Computer Vision and Pattern Recognition*, 2018. 2

[2] Ahmad Alzu'bi and Lujain Alsmadi. Monitoring deforestation in Jordan using deep semantic segmentation with satellite imagery. *Ecological Informatics*, 2022. 1

[3] Shir Amir, Yossi Gandelsman, Shai Bagon, and Tali Dekel. Deep ViT features as dense visual descriptors. *European Conference on Computer Vision Workshop*, 2022. 1, 2

[4] Tanmay Anand, Soumendu Sinha, Murari Mandal, Vinay Chamola, and Fei Richard Yu. AgriSegNet: Deep aerial semantic segmentation framework for IoT-assisted precision agriculture. *IEEE Sensors Journal*, 2021. 1

[5] Ella Bingham and Heikki Mannila. Random projection in dimensionality reduction: Applications to image and text data. In *International Conference on Knowledge Discovery and Data Mining*, 2001. 6

[6] Christos Boutsidis, Anastasios Zouzias, and Petros Drineas. Random projections for k-means clustering. In *Advances in Neural Information Processing Systems*, 2010. 6

[7] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. COCO-Stuff: Thing and stuff classes in context. In *Conference on Computer Vision and Pattern Recognition*, 2018. 4

[8] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *Advances in Neural Information Processing Systems*, 2020. 1

[9] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *International Conference on Computer Vision*, 2021. 1, 2, 3, 4, 5

[10] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning*, 2020. 1, 3, 4

[11] Jang Hyun Cho, Utkarsh Mall, Kavita Bala, and Bharath Hariharan. PiCIE: Unsupervised semantic segmentation using invariance and equivariance in clustering. In *Conference on Computer Vision and Pattern Recognition*, 2021. 2, 4

[12] Edo Collins, Radhakrishna Achanta, and Sabine Susstrunk. Deep feature factorization for concept discovery. In *European Conference on Computer Vision*, 2018. 2

[13] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The Cityscapes dataset for semantic urban scene understanding. In *Conference on Computer Vision and Pattern Recognition*, 2016. 4

[14] Sanjoy Dasgupta. Experiments with random projection. In *Conference on Uncertainty in Artificial Intelligence*, 2000. 6

[15] Li Deng. The MNIST database of handwritten digit images for machine learning research [Best of the Web]. *IEEE Signal Processing Magazine*, 2012. 1

[16] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16×16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. 1

[17] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *International Conference on Artificial Intelligence and Statistics*, 2011. 2

[18] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent – a new approach to self-supervised learning. In *Advances in Neural Information Processing Systems*, 2020. 1, 4

[19] Tianrui Guan, Divya Kothandaraman, Rohan Chandra, Adarsh Jagan Sathyamoorthy, Kasun Weerakoon, and Dinesh Manocha. GA-Nav: Efficient terrain segmentation for robot navigation in unstructured outdoor environments. *IEEE Robotics and Automation Letters*, 2022. 1

[20] Mark Hamilton, Zhoutong Zhang, Bharath Hariharan, Noah Snavely, and William T. Freeman. Unsupervised semantic segmentation by distilling feature correspondences. In *International Conference on Learning Representations*, 2022. 1, 2, 3, 4, 5, 6, 7, 8

[21] 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*, 2019. 5, 7

[22] Xu Ji, Joao F. Henriques, and Andrea Vedaldi. Invariant information clustering for unsupervised image classification and segmentation. In *International Conference on Computer Vision*, 2019. 4

[23] William Johnson and Joram Lindenstrauss. Extensions of Lipschitz maps into a Hilbert space. *Contemporary Mathematics*, 1984. 6

[24] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected CRFs with Gaussian edge potentials. In *Advances in Neural Information Processing Systems*, 2011. 2

[25] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. *Technical Report, University of Toronto*, 2009. 1

[26] Harold W Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 1955. 4

[27] Xin Lai, Zhuotao Tian, Li Jiang, Shu Liu, Hengshuang Zhao, Liwei Wang, and Jiaya Jia. Semi-supervised semantic segmentation with directional context-aware consistency. In *Conference on Computer Vision and Pattern Recognition*, 2021. 2

[28] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Berkeley Symposium on Mathematical Statistics and Probability*, 1967. 4

[29] Luke Melas-Kyriazi, Christian Rupprecht, Iro Laina, and Andrea Vedaldi. Deep spectral methods: A surprisingly strong baseline for unsupervised semantic segmentation and

localization. In *Conference on Computer Vision and Pattern Recognition*, 2022. 2

[30] Youngmin Oh, Beomjun Kim, and Bumsub Ham. Background-aware pooling and noise-aware loss for weakly-supervised semantic segmentation. In *Conference on Computer Vision and Pattern Recognition*, 2021. 2

[31] Karl Pearson. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 1901. 6

[32] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention*, 2015. 1

[33] 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*, 2015. 1, 5

[34] Monjoy Saha and Chandan Chakraborty. Her2Net: A deep framework for semantic segmentation and classification of cell membranes and nuclei in breast cancer evaluation. *IEEE Transactions on Image Processing*, 2018. 1

[35] M. Saquib Sarfraz, Marios Koulakis, Constantin Seibold, and Rainer Stiefelhagen. Hierarchical nearest neighbor graph embedding for efficient dimensionality reduction. In *Conference on Computer Vision and Pattern Recognition*, 2022. 8

[36] Wei Shen, Zelin Peng, Xuehui Wang, Huayu Wang, Jiazhong Cen, Dongsheng Jiang, Lingxi Xie, Xiaokang Yang, and Q. Tian. A survey on label-efficient deep image segmentation: Bridging the gap between weak supervision and dense prediction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023. 2

[37] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2000. 2

[38] Squirrel Developer Team. Squirrel: A Python library that enables ML teams to share, load, and transform data in a collaborative, flexible, and efficient way. *GitHub. Note: https://github.com/merantix-momentum/squirrel-core*, 2022. 8

[39] Vadim Tschernezki, Iro Laina, Diane Larlus, and Andrea Vedaldi. Neural feature fusion fields: 3D distillation of self-supervised 2D image representations. In *International Conference on 3D Vision*, 2022. 1

[40] Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, and Luc Van Gool. Unsupervised semantic segmentation by contrasting object mask proposals. In *International Conference on Computer Vision*, 2021. 2

[41] Wouter Van Gansbeke, Simon Vandenhende, and Luc Van Gool. Discovering object masks with transformers for unsupervised semantic segmentation. In *arXiv*, 2022. 2, 8

[42] Yude Wang, Jie Zhang, Meina Kan, Shiguang Shan, and Xilin Chen. Self-supervised equivariant attention mechanism for weakly supervised semantic segmentation. In *Conference on Computer Vision and Pattern Recognition*, 2020. 2

[43] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip H.S. Torr, and Li Zhang. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *Conference on Computer Vision and Pattern Recognition*, 2021. 1