Unsupervised Image Classification

Soumyadeep, Aditya, Tanish, Mihit, Pranav

***Abstract*—Unsupervised image classification has become increasingly important due to the abundance of unlabeled data and the challenges of manual annotation. Traditional clustering methods like KMeans and DBSCAN have been widely used, but their effectiveness is limited by reliance on hand-engineered features. Recent advancements in self-supervised learning (SSL), particularly with models like DINO and Vision Transformers (ViT), have enabled richer feature representations without labeled data. In this paper, we evaluate the performance of KMeans and DBSCAN on SSL features extracted using DINO and propose a novel method that integrates image captioning via BLIP and subject identification to generate CLIP embeddings. These embeddings are used as classification weights, combining visual and textual information to enhance unsupervised image classification. The introduction of SCAN, a state-of-the-art method, has advanced unsupervised classification by incorporating uncertainty in clustering, leading to more accurate and robust models. Our approach builds on SCAN's success by integrating multimodal features, leveraging both visual and textual data, to push the boundaries of unsupervised learning. We aim to contribute to the development of more scalable and efficient solutions for real-world applications. The code files and processed datasets for this work are available at:** https://github.com/soumyadeep-p/STAMPClustering-UIC/**.**

I. Introduction

A. Background

Image classification is a fundamental task in computer vision that involves assigning a label or category to an image based on its visual content. It serves as the basis for a wide range of applications, including object detection, facial recognition, medical imaging, and autonomous driving. In image classification, a model is trained to analyze the pixel data of an image and predict the class it belongs to from a predefined set of categories. This task is crucial for enabling machines to interpret and understand visual data, bridging the gap between raw pixel-level information and higher-level semantic understanding.

B. Motivation

Deep learning has excelled in computer vision tasks like image classification, but its success depends on large labeled datasets and high computational costs. Creating such datasets is expensive, driving researchers toward training methods that don't rely on labeled data. Self-supervised learning has emerged as a key approach, paving the way for unsupervised techniques like image clustering, which can classify images without the need for manual annotations.

C. Target Problem

Unsupervised image clustering aims to organize a collection of unlabeled images into distinct groups or clusters based on their intrinsic similarities or features. Each cluster should ideally represent a coherent and meaningful image category with common characteristics.

II. Literature Review

A. SCAN

Unsupervised learning has emerged as a critical field in computer vision, driven by the need to develop methods that effectively leverage unlabeled data. Among these, SCAN (Semantic Clustering by Adopting Nearest Neighbors) has set a significant milestone by introducing a novel approach to cluster images based on semantic similarity. SCAN, proposed by Van Gansbeke et al. (2020), addresses the challenges associated with clustering in high-dimensional feature spaces, particularly in the context of unsupervised learning.

SCAN builds upon the success of self-supervised learning methods, such as SimCLR and MoCo, which generate meaningful feature representations from unlabeled data. However, unlike these methods that primarily focus on feature extraction, SCAN emphasizes clustering these representations into semantically meaningful groups. This is achieved through a two-stage pipeline. In the first stage, SCAN utilizes a pre-trained self-supervised model to extract feature embeddings from images. In the second stage, it introduces a clustering approach guided by a nearest neighbor-based consistency loss, ensuring that semantically similar samples are grouped together.

A distinguishing feature of SCAN is its reliance on the concept of mutual nearest neighbors. Instead of directly assigning cluster labels based on proximity in the feature space, SCAN identifies samples that share a mutual nearest neighbor relationship, as these pairs are more likely to belong to the same semantic cluster. This innovative approach reduces the risk of noisy cluster assignments and enhances robustness, particularly in datasets with high intra-class variability.

Furthermore, SCAN incorporates a self-labeling step to iteratively refine the cluster assignments. During this step, confident predictions are used to pseudo-label the data, which is then used to retrain the clustering network. This self-labeling mechanism ensures continuous improvement in clustering accuracy and reduces reliance on manual annotations. SCAN's pipeline is both elegant and efficient, requiring minimal human intervention while delivering state-of-the-art performance on benchmark datasets such as CIFAR-10, STL-10, and ImageNet.

The results achieved by SCAN highlight its strength in unsupervised image classification. On CIFAR-10, SCAN achieves near-supervised performance, clustering images into meaningful groups that align well with ground truth labels. Similarly, its performance on STL-10 and ImageNet demonstrates its scalability and adaptability across datasets of varying complexity.

SCAN has also inspired subsequent research in unsupervised learning, particularly in leveraging mutual nearest neighbor relationships for robust clustering. Methods such as PiCIE and IIC have drawn on SCAN's ideas to refine their clustering frameworks, further validating the impact of SCAN on the field.

Despite its success, SCAN is not without limitations. Its reliance on high-quality feature representations means its performance heavily depends on the quality of the pre-trained self-supervised model. Additionally, the mutual nearest neighbor strategy may struggle in datasets with extreme class imbalances or overlapping classes. Nonetheless, SCAN represents a significant step forward in bridging the gap between self-supervised representation learning and unsupervised clustering, and its contributions continue to shape the development of modern unsupervised learning methods.

B. *Clustering Algorithms for Unsupervised Learning*

Clustering algorithms group data points based on similarity, aiming to identify natural structures in the data. They can be categorized into partitioning methods, density-based methods, and graph-based methods.

1) ***K-Means Clustering****:* KMeans is a widely used unsupervised clustering algorithm, introduced by MacQueen in 1967. It works by partitioning data into kk clusters to minimize the variance within each cluster. The algorithm iterates by assigning each data point to the nearest centroid and then updating the centroids based on the mean of the assigned points. This process continues until convergence.

While simple, KMeans has seen several enhancements to address its limitations. KMeans++ improves centroid initialization by selecting more spread-out centroids, reducing the risk of poor convergence. Variants like Kernel KMeans extend the algorithm to non-linear data, while Mini-Batch KMeans allows for faster computation on large datasets.

KMeans has applications in areas like image segmentation, document clustering, and customer segmentation due to its computational efficiency. It is also used in anomaly detection, identifying outliers based on their distance from the centroids. However, KMeans assumes spherical clusters of similar sizes, making it unsuitable for datasets with irregular or imbalanced clusters. It also requires the number of clusters (kk) to be predefined, which can be challenging without prior knowledge of the data.

2) ***DBSCAN Clustering****:* DBSCAN (Density-Based Spatial Clustering of Applications with Noise), introduced by Ester et al. in 1996, is a popular density-based clustering algorithm that groups points based on their density in the feature space. Unlike KMeans, DBSCAN does not require the number of clusters to be pre-defined. It identifies clusters as regions of high point density separated by regions of low density, with two main parameters: epsilon (), the radius of neighborhood, and MinPts, the minimum number of points required to form a cluster.

DBSCAN's strength lies in its ability to discover clusters of arbitrary shapes, making it particularly effective in datasets with noise or irregular cluster structures. It can also identify outliers, labeling points that do not fit into any cluster as noise. The algorithm has applications in various domains, including image segmentation, anomaly detection, and spatial data analysis, where the assumption of spherical clusters in KMeans does not hold.

However, DBSCAN also has limitations. It struggles with varying densities within the dataset, as a single and MinPts value may not work well for all clusters. Additionally, the algorithm's performance is sensitive to the choice of parameters, and selecting optimal values for and MinPts can be challenging, especially in high-dimensional spaces. Despite these challenges, DBSCAN remains a valuable tool for clustering and continues to be widely used in unsupervised learning tasks due to its ability to handle noise and discover clusters of arbitrary shapes.

C. *Vision Language Models*

1) ***BLIP****:* BLIP (Bootstrapping Language-Image Pretraining)[5] is a powerful vision-language model designed to enhance image understanding by leveraging large-scale image-text data. It builds upon the success of previous multimodal models by using a bootstrapping mechanism to iteratively refine the image-text alignment. The architecture of BLIP is composed of two primary components: a vision backbone for image feature extraction and a language model for text generation and understanding.

BLIP employs a vision backbone, typically a Vision Transformer (ViT), to extract visual features from images. The ViT processes images in a patch-based manner, dividing the image into fixed-size patches, with each patch treated as a token in the Transformer. The extracted visual features are then passed to a cross-modal Transformer, which merges both image and text information. The text component of the model uses a pre-trained Transformer, such as GPT or BERT, to process textual inputs.

The key innovation of BLIP lies in its bootstrapping pre-training strategy. This mechanism helps the model generate captions for images and refine its understanding of the relationships between text and visual features. Initially, BLIP is trained on a large corpus of image-text pairs, where it learns to predict textual information (e.g., captions or object descriptions) based on the image input. Instead of relying purely on labeled data, BLIP iteratively refines its predictions by generating its own text from images and using this generated text to improve its image understanding. This bootstrapping approach enables BLIP to effectively learn cross-modal alignment, improving performance on tasks such as image captioning and visual question answering (VQA). Unlike traditional methods that rely heavily on manually annotated datasets, BLIP's self-supervised pretraining helps it perform well even with limited labeled data.

BLIP has been evaluated on various vision-language tasks, including image captioning, VQA, and image-text retrieval. For image captioning, BLIP generates descriptive captions by encoding the image with the vision backbone and conditioning the language model on the image features. For VQA, BLIP leverages the learned image-text alignment to answer questions

based on the visual content of the image. In image-text retrieval tasks, the model ranks images and text according to their relevance to each other.

One of the main advantages of BLIP is its ability to generate high-quality, semantically accurate captions and provide robust answers to visual questions, outperforming previous models on many benchmarks. The bootstrapping mechanism reduces reliance on large labeled datasets, making BLIP more efficient for tasks with limited annotations. However, like other vision-language models, BLIP faces challenges in handling complex multimodal queries that require a deep understanding of fine-grained visual details or abstract relationships between objects in images.

In conclusion, BLIP's bootstrapping approach represents a significant advancement in vision-language pretraining. It enables more efficient and scalable learning from large image-text datasets and demonstrates the potential of combining large-scale pretraining with iterative refinement to improve cross-modal understanding across a variety of vision-language tasks.

*2) **CLIP**:* CLIP (Contrastive Language-Image Pretraining)[4] is a groundbreaking multimodal model designed to learn visual and textual representations by aligning images and their corresponding textual descriptions in a shared embedding space. It achieves this by training a vision model and a language model simultaneously using large-scale datasets of image-text pairs. The core idea behind CLIP is to leverage contrastive learning, where the model learns to associate images with their corresponding textual descriptions while distinguishing them from unrelated text-image pairs.

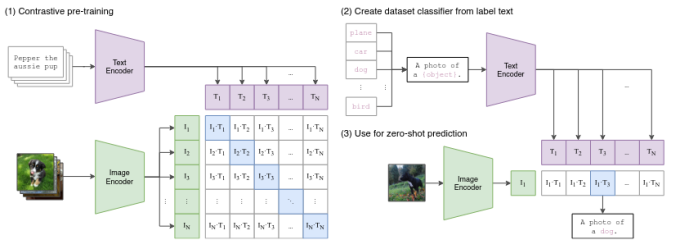


Fig. 1: CLIP Overview

CLIP employs two separate encoders, one for images and one for text. The image encoder typically uses a Vision Transformer (ViT) or ResNet architecture to process the image, while the text encoder uses a Transformer-based architecture similar to models like GPT or BERT to process textual data. Both encoders project their respective inputs into a common embedding space. The model is trained with a contrastive loss, which encourages the embeddings of matched image-text pairs to be closer together in this space, while embeddings of mismatched pairs are pushed apart. This approach allows CLIP to learn visual concepts based solely on the associated text, without requiring labeled data for specific visual tasks.

The contrastive learning setup in CLIP allows it to excel in zero-shot learning scenarios. Once trained, CLIP can be applied to a wide variety of downstream tasks, such as image classification, object detection, and image retrieval, without needing task-specific fine-tuning. This makes CLIP highly versatile and effective in transferring learned knowledge from pretraining to diverse tasks. The model performs exceptionally well in zero-shot classification, where it directly maps images to class names described in natural language, demonstrating its strong ability to generalize across a range of categories and datasets.

CLIP's ability to generalize to new tasks without retraining has made it a valuable tool for various applications, including content moderation, accessibility, and multimodal search. The ability to leverage large-scale image-text data without manual annotation enables CLIP to perform well across diverse domains, even when there is limited labeled data available for the specific task at hand. However, despite its impressive performance, CLIP is not without its limitations. Its reliance on large-scale training data means that the model's performance can be sensitive to biases present in the dataset, and it may not always handle out-of-distribution data effectively.

In conclusion, CLIP represents a major advancement in the field of vision-language models, providing a unified framework that effectively learns multimodal representations. Its use of contrastive learning for aligning images and text allows it to perform exceptionally well on a range of tasks with minimal task-specific fine-tuning, making it a powerful tool for applications requiring cross-modal understanding.

III. Proposed Work

A. *Clustering on DINO Features*

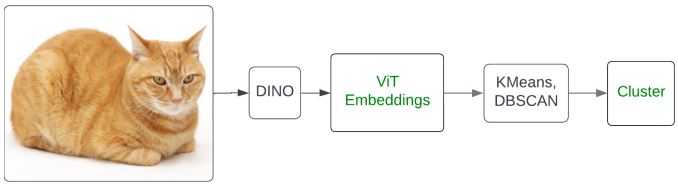


Fig. 2: Clustering on DINO Features

In this work, we leverage the DINO pre-trained Vision Transformer (ViT) model from Facebook to generate feature vectors for the images. Specifically, we use the ViT-S16 architecture, which has been trained on large-scale datasets for self-supervised learning, making it ideal for extracting rich feature representations without requiring labeled data. After passing an image through the DINO model, we take the classification head (or the final output vector) which provides a 384-dimensional feature vector for each image. This feature vector encapsulates high-level semantic information about the image, making it suitable for clustering. We then use KMeans and DBSCAN clustering algorithms to group the images based on these DINO-generated features, establishing a baseline for comparison with our more advanced methods.

B. STAMP Clustering

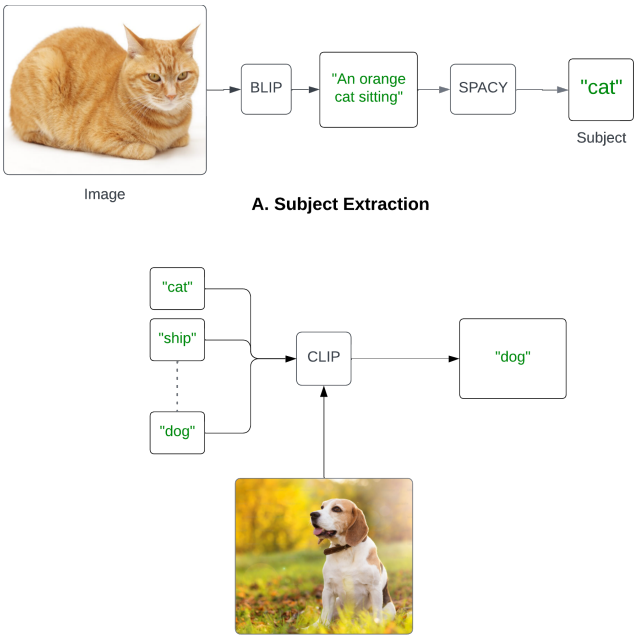


Fig. 3: STAMPClustering Overview

For the STAMP Clustering method, we first pass each image through the BLIP (Bootstrapping Language-Image Pre-training) model, which generates descriptive captions for the image. These captions provide a textual description that reflects the content and context of the image. We then use the SpaCy Natural Language Processing (NLP) model to analyze the captions and extract the most frequently occurring subjects, which we treat as pseudo-labels representing the image classes. These subjects are used to initialize the STAMPCluster module. The STAMPCluster module then generates CLIP embeddings for each subject, which act as the classification weights. During inference, when a new image is inputted, we calculate the cosine similarity between the CLIP-encoded image features and the subject embeddings to assign a predicted class to the image, similar to how CLIP works for zero-shot classification tasks.

1) ***Intuition behind the Architecture****:* The key intuition behind this approach lies in the capabilities of the BLIP model, which has been extensively trained on large-scale datasets. This extensive training allows BLIP to generate captions that capture semantic meaning, providing rich context about the image content. These captions, enriched with semantic information, are further processed to extract meaningful subjects, which act as proxy labels for the images. Using CLIP to generate embeddings for these subjects enables us to transform the textual descriptions into vector space representations, aligning them with the visual features of the images. The use of cosine similarity between the image features and subject embeddings allows for a natural, effective classification scheme that combines the strengths of both vision and language models.

2) ***Image Captioning****:* In our approach, image captioning is performed using the BLIP model, a state-of-the-art language-vision pre-trained model. The BLIP model generates captions for input images by encoding visual features and then decoding them into natural language. Let the image be denoted as I. We generate the caption for I as C using the BLIP model, where the caption

$$C = \text{BLIP}(I)$$

. The BLIP model is designed to produce meaningful textual descriptions based on the content of the image, providing a bridge between the visual features and the language space.

3) ***Subject Extraction****:* To classify images, we first extract subjects from the generated captions. This is done by utilizing SpaCy's dependency parsing to identify the main noun subjects in each caption. Let the caption be denoted as C, and we process it through SpaCy's NLP model to extract the subject S. For a caption $C =$ "A man is skiing", the extracted subject $S =$ "man", which is identified as the noun subject ($nsubj$) in the sentence. This subject extraction step helps us associate each image with its corresponding class.

$$S = ExtractSubject(C)$$

.

4) ***Subject Filtering****:* After extracting the subjects from captions, we filter out any irrelevant or ambiguous subjects. For example, common subjects like "person" or "animal" might occur frequently and need to be clustered into more specific categories. We define a threshold for frequency and consider only those subjects that appear frequently across the dataset. Let S_{freq} denote the frequency of a subject S in the dataset. If $S_{freq} >$ threshold, then S is considered a valid subject label for the image.

5) ***Text Embeddings****:* Once subjects are extracted and filtered, we use the CLIP model to generate embeddings for these subjects. CLIP (Contrastive Language-Image Pretraining) is capable of mapping both text and image features into a shared embedding space. Let S be the extracted subject, and we encode it using the CLIP text encoder to obtain the embedding

$$\mathbf{e}_S = \text{CLIP\_text\_encode}(S)$$

. Similarly, we encode the image I using the CLIP image encoder to obtain the image embedding

$$\mathbf{e}_I = \text{CLIP\_image\_encode}(I)$$

. These embeddings are then used for clustering and classification tasks.

6) ***Inference****:* During inference, the image I is encoded into its feature vector $\mathbf{e}_I$ using the CLIP image encoder. Similarly, the subject embedding $\mathbf{e}_S$ for each subject is precomputed. To classify the image, we calculate the cosine similarity between the image embedding and each subject embedding, and the image is assigned the subject with the highest similarity. The cosine similarity $\cos(\theta)$ between two vectors $\mathbf{e}_I$ and $\mathbf{e}_S$ is computed as:

$$\cos(\theta) = \frac{\mathbf{e}_I \cdot \mathbf{e}_S}{\|\mathbf{e}_I\|\|\mathbf{e}_S\|}$$

Where $\mathbf{e}_I$ and $\mathbf{e}_S$ are the normalized feature vectors for the image and the subject, respectively. The subject with the highest cosine similarity is chosen as the predicted class for the image.

IV. Experimental and Results

A. Datasets

*1) **CIFAR-10**:* CIFAR-10 is a widely used dataset for image classification, consisting of 60,000 32x32 RGB images divided into 50,000 training and 10,000 testing samples. The dataset contains ten classes: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck, with 6,000 images per class, making it well-balanced. CIFAR-10 was introduced by Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton and has become a standard benchmark for supervised and unsupervised learning methods. Its small image size and diverse set of categories make it ideal for testing algorithms on multi-class classification tasks.

*2) **MNIST**:* MNIST is a benchmark dataset designed for handwritten digit recognition tasks. It consists of 70,000 28x28 grayscale images, with 60,000 used for training and 10,000 for testing. The dataset includes ten classes corresponding to the digits 0 through 9, with approximately 7,000 images per class, ensuring a balanced distribution. Introduced by Yann LeCun, Corinna Cortes, and Christopher J.C. Burges, MNIST has served as a foundational dataset for evaluating machine learning models due to its simplicity and relevance in optical character recognition tasks.

*3) **STL-10**:* STL-10 is a challenging dataset designed for unsupervised and semi-supervised learning, derived from ImageNet. It comprises 130,000 images, including 5,000 labeled training samples, 8,000 labeled testing samples, and 100,000 unlabeled images. The images are 96x96 in resolution, significantly larger than CIFAR-10, and are categorized into ten classes: airplane, bird, car, cat, deer, dog, horse, monkey, ship, and truck. Created by Adam Coates, Andrew Ng, and Honglak Lee, STL-10 is particularly suited for exploring self-supervised learning techniques due to its large unlabeled set and high-quality images. The combination of labeled and unlabeled data makes it an excellent dataset for evaluating advanced learning methods.

B. Training Details

The training process was organized into two distinct stages: dataset processing and clustering. During the processing stage, we employed the DINO pre-trained model to extract feature vectors for all samples, which were then used to generate new training, validation, and testing splits. Simultaneously, we used BLIP to generate subject labels for each sample in the datasets, enriching the dataset with meaningful textual annotations. The clustering stage leveraged these processed datasets, enabling efficient and accurate unsupervised learning. We utilized the elbow method to determine the optimal number of clusters for KMeans, ensuring a balance between within-cluster variance and computational efficiency. For DBSCAN, hyperparameter tuning was performed to identify the best values for eps and $min_samples$, optimizing the model's ability to discover clusters of varying densities. In our proposed approach, we experimented with different token lengths and thresholds for the number of samples associated with each subject to achieve the best classification results.

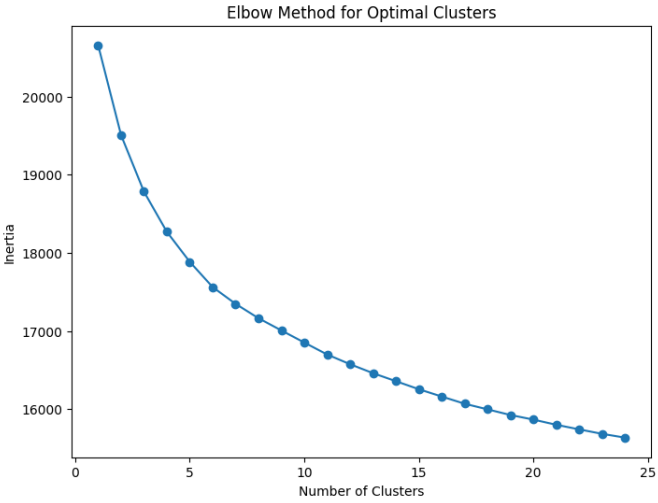


Fig. 4: KMeans using Elbow Method

All training and experimentation were conducted on an NVIDIA P100 GPU, ensuring the computational resources necessary for handling large-scale datasets and complex multimodal models.

C. Evaluation Metrics

To assess the performance of our unsupervised classification methods, we employ accuracy (ACC), normalized mutual information (NMI), and adjusted Rand index (ARI). These metrics offer a comprehensive evaluation of clustering and classification effectiveness. Using these criteria, we benchmark our proposed model against various unsupervised clustering methods and SCAN, the current state-of-the-art approach.

*1) **Clustering Accuracy (ACC)**:* Accuracy measures the proportion of correctly classified samples and is commonly used to evaluate the performance of classification tasks. For unsupervised classification, the predicted clusters are first aligned with the ground truth labels using the Hungarian algorithm to maximize correspondence. Formally, accuracy is defined as:

$$\text{ACC} = \frac{\sum_{i=1}^{n} 1(y_i = \text{map}(\hat{y}_i))}{n},$$

where y_i and $\hat{y}_i$ represent the true and predicted labels, respectively, map denotes the optimal alignment function, and n is the total number of samples. Higher accuracy indicates better alignment between clusters and true labels.

*2) **Normalized Mutual Information (NMI)**:* NMI quantifies the amount of shared information between the predicted and ground truth clusters while normalizing for their individual entropy. It ranges from 0 (no mutual information) to 1 (perfect correlation). Formally, NMI is computed as:

$$\text{NMI}(Y, \hat{Y}) = \frac{2 \cdot I(Y; \hat{Y})}{H(Y) + H(\hat{Y})},$$

where $I(Y;\hat{Y})$ is the mutual information between the true labels Y and predicted labels $\hat{Y}$, and $H(Y)$ and $H(\hat{Y})$ are their respective entropies. NMI is invariant to label permutations and is widely used in clustering evaluations.

*3) **Adjusted Rand Index (ARI)**:* ARI measures the similarity between the ground truth and predicted clustering by considering all pairwise combinations of samples. It accounts for chance agreements, making it more robust than the standard Rand index. ARI is defined as:

$$\text{ARI} = \frac{\text{Index} - \text{Expected Index}}{\text{Max Index} - \text{Expected Index}},$$

where the Index measures the agreement between pairs of samples, the Expected Index accounts for random labeling, and the Max Index normalizes the score. ARI ranges from -1 to 1, where values closer to 1 indicate stronger agreement.

These metrics together provide a holistic view of the clustering and classification quality, ensuring that both alignment with true labels and robustness to label permutations are considered.

D. *Results*

Method	Dataset	ACC	NMI	ARI
KMeans [1]	CIFAR-10	61.0	53.0	41.0
DBSCAN [2]	CIFAR-10	10.2	0.1	1.2
SCAN [3]	CIFAR-10	**88.3**	**79.7**	**77.2**
Our Model	CIFAR-10	76.6	67.6	65.5
KMeans [1]	STL-10	95.1	90.7	89.4
DBSCAN [2]	STL-10	36.3	54.5	30.4
SCAN [3]	STL-10	80.9	69.8	64.6
Our Model	STL-10	**95.7**	**91.8**	**91.8**
KMeans [1]	MNIST	34.4	**32.9**	**18.3**
DBSCAN [2]	MNIST	11.3	0.2	0.0
Our Model	MNIST	**36.7**	28.6	18.0

TABLE I: State of the Art Comparison

V. Conclusion

In this work, we explored unsupervised image classification by leveraging advanced self-supervised learning features and multimodal representations. Through extensive experiments, we evaluated the performance of traditional clustering techniques such as KMeans and DBSCAN on DINO pre-trained Vision Transformer features, demonstrating their effectiveness in identifying meaningful clusters in image data. Additionally, we proposed a novel approach that integrates BLIP-based image captioning and subject identification with CLIP embeddings to create an innovative classification framework.

Our results highlight the importance of combining visual and textual modalities to enhance the robustness and accuracy of unsupervised learning methods. By systematically tuning hyperparameters for clustering algorithms and introducing an adaptable framework in our proposed model, we showed the potential for improved performance compared to existing approaches, including SCAN.

This work underscores the significance of leveraging multimodal information in unsupervised classification tasks and sets a foundation for future research in combining vision and language models to address complex challenges in unsupervised learning. Further exploration into scaling this approach to larger datasets and more diverse domains remains a promising direction for continued innovation.

References

[1] MacQueen, J. (1967). Some Methods for Classification and Analysis of Multivariate Observations. *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, 1, 281-297.

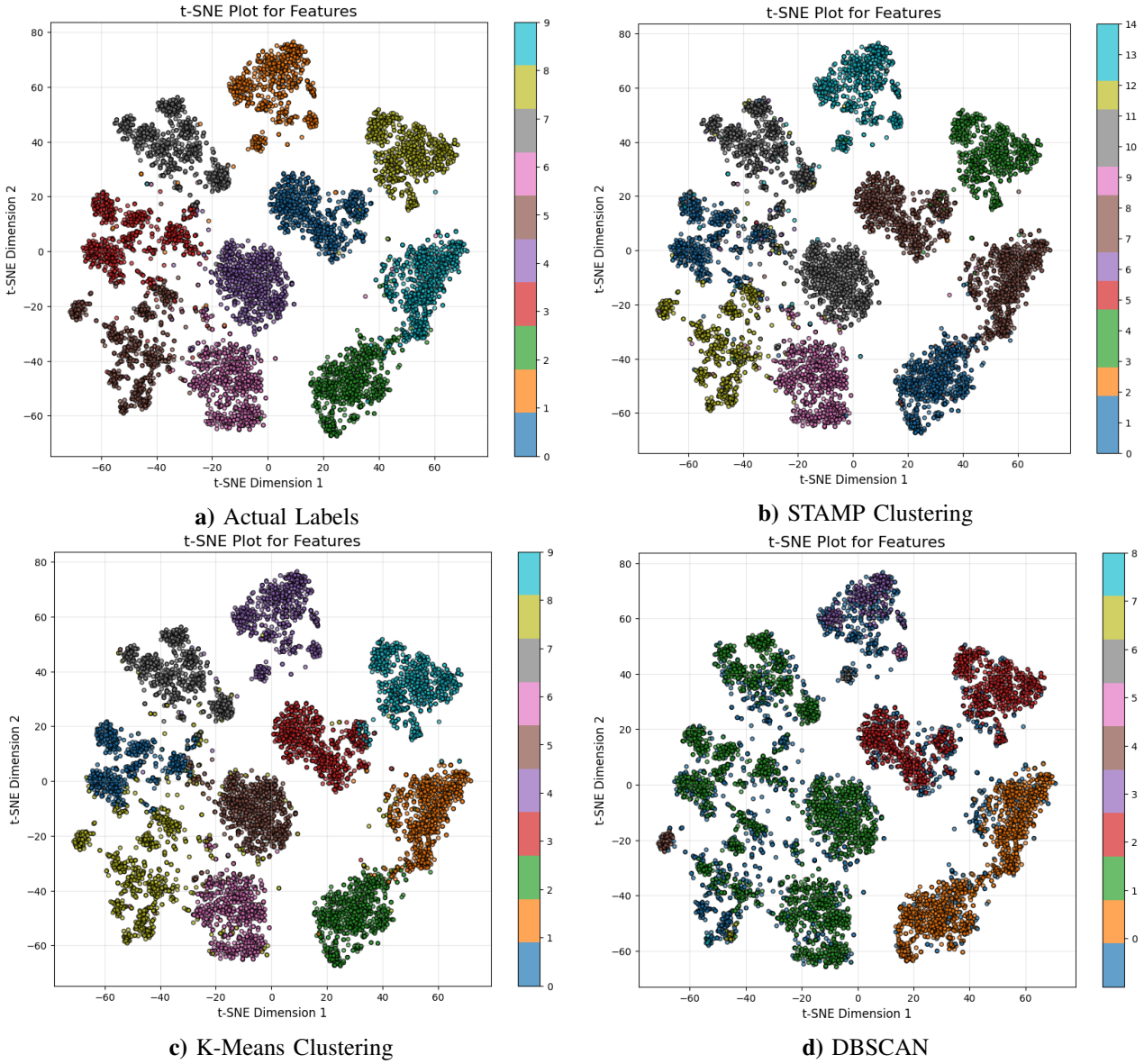
[2] Ester, M., Kriegel, H.-P., Sander, J., & Xu, X. (1996). A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96)*, 226-231.

[3] Geng, X., Li, X., & Hoiem, D. (2021). SCAN: Learning to Classify Images with Uncertainty. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, 4341-4350.

[4] Radford, A., Kim, J. W., Hallacy, C., et al. (2021). Learning Transferable Visual Models From Natural Language Supervision. *Proceedings of the 2021 International Conference on Machine Learning (ICML)*, 8748-8763.

[5] Li, J., Li, Y., & Wang, L. (2022). BLIP: Bootstrapping Language-Image Pretraining. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016-2025.

APPENDIX I - T-SNE PLOTS FOR CLUSTER VISUALIZATION



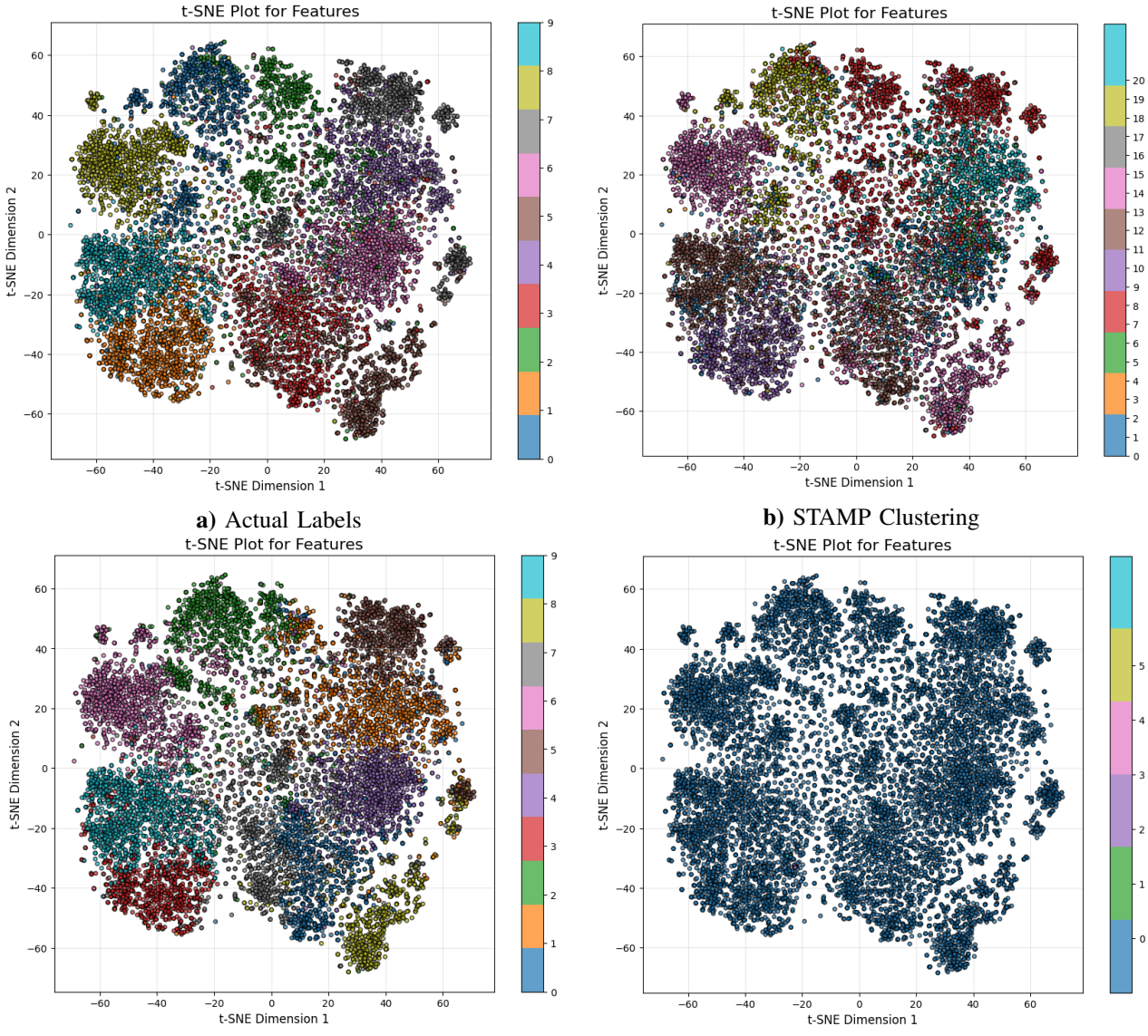
a) Actual Labels

b) STAMP Clustering

c) K-Means Clustering

d) DBSCAN

Fig. 5: t-SNE plots for the STL-10 dataset are generated using the DINO features extracted from the test split of the dataset.



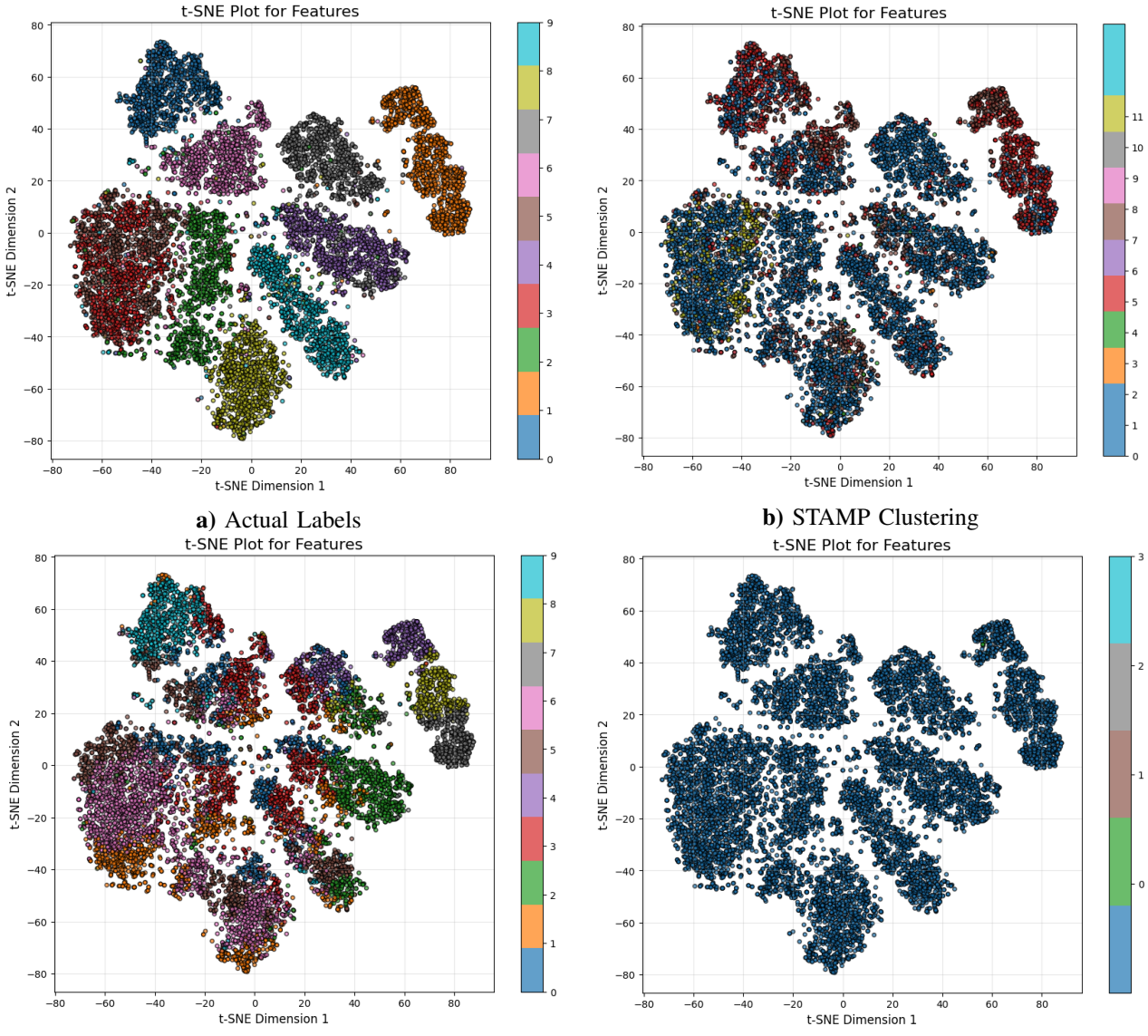
a) Actual Labels

b) STAMP Clustering

c) K-Means Clustering

d) DBSCAN

Fig. 6: t-SNE plots for the CIFAR-10 dataset are generated using the DINO features extracted from the test split of the dataset.



a) Actual Labels

b) STAMP Clustering

c) K-Means Clustering

d) DBSCAN

Fig. 7: t-SNE plots for the MNIST dataset are generated using the DINO features extracted from the test split of the dataset.