PACT: Pruning and Clustering-Based Token Reduction for Faster Visual Language Models

Mohamed Dhouib
LIX, École Polytechnique, IP Paris, France
mohamed.dhouib@polytechnique.edu

Davide Buscaldi
LIPN, Université Sorbonne Paris Nord, France
davide.buscaldi@lipn.univ-paris13.fr

Sonia Vanier
LIX, École Polytechnique, IP Paris, France
sonia.vanier@polytechnique.edu

Aymen Shabou
DataLab Groupe, Crédit Agricole S.A, France
aymen.shabou@credit-agricole-sa.fr

Abstract

*Visual Language Models require substantial computational resources for inference due to the additional input tokens needed to represent visual information. However, these visual tokens often contain redundant and unimportant information, resulting in an unnecessarily high number of tokens. To address this, we introduce **PACT**, a method that reduces inference time and memory usage by pruning irrelevant tokens and merging visually redundant ones at an early layer of the language model. Our approach uses a novel importance metric to identify unimportant tokens without relying on attention scores, making it compatible with FlashAttention. We also propose a novel clustering algorithm, called Distance Bounded Density Peak Clustering, which efficiently clusters visual tokens while constraining the distances between elements within a cluster by a predefined threshold. We demonstrate the effectiveness of **PACT** through extensive experiments.*

1. Introduction

Extending Large language models to modalities other than text [11, 18, 19, 55, 56] has seen success in recent years across various domains, especially in the visual domain with models like LLaVA [31] and Qwen-VL [4]. State-of-the-art Visual Language Models generally consist of three main components: a vision encoder, a connector, and a language model. The vision encoder converts input images into visual tokens, which are passed through the connector and then fed to the language model along with the input text. While this architecture has shown impressive performance across different tasks, it suffers from high computational cost due to the large number of visual tokens. In this paper, we introduce two complementary methods to optimize Visual Language Models by reducing inference time and memory requirements: a pruning module and a clustering algorithm. These methods can be used independently or combined, forming the **PACT** approach for greater effectiveness. Notably, our pruning and clustering modules, as well as **PACT**, are applied at inference time and thus require no additional training. The pruning module identifies unimportant visual tokens based on a novel importance metric that evaluates each token's relevance without relying on attention scores. This makes it compatible with FlashAttention [12], as FlashAttention does not support the calculation of attention scores. The second module introduces a novel clustering algorithm, **Distance Bounded Density Peak Clustering (DBDPC)**, which clusters visual tokens while ensuring that the distances between elements within a cluster are constrained by a predefined threshold. By combining these two methods, we develop **PACT**. First, the pruning module eliminates unimportant tokens, then the **DBDPC** algorithm clusters the remaining ones. Tokens that were initially pruned but are sufficiently close to the constructed clusters are reincorporated, ensuring that valuable information from the pruned tokens is recovered. Finally, the tokens within each cluster are merged into a single representative token, reducing the total token count.

By combining both pruning and clustering, **PACT** achieves an effective visual token reduction, addressing both irrelevant and redundant tokens. When applied to LLaVA-OneVision-7B, **PACT** achieves a 50% visual token reduction with negligible performance loss. Moreover, **PACT** exhibits significantly less performance degradation at higher reduction ratios compared to previous methods, achieving 71.3% visual token reduction ratio with only 1.4% performance drop, whereas previous state-of-the-art methods show at best a 4.4% performance drop at an equal reduction ratio. Our contributions are as follows:

- We propose a novel visual token pruning metric that does not rely on attention scores, ensuring compatibility with FlashAttention, and empirically validate its effectiveness.

- We introduce a new clustering algorithm aimed at reducing visual redundancy and show its superiority over other clustering algorithms for visual token reduction.
- We show that combining pruning with clustering-based merging surpasses either technique alone for visual token reduction. By integrating our pruning and clustering algorithms, we propose a novel approach, **PACT**, and demonstrate that it outperforms previous and concurrent works [3, 6, 9, 30, 44]. The codebase used to obtain the results in this study is available at https://github.com/orailix/PACT/tree/main.

2. Related work

2.1. Visual language models

Since the introduction of BLIP-2 [28], the use of a visual encoder followed by a connector that feeds visual vectors to the language model has become the standard architecture for Visual Language Models (VLMs) [7, 17, 50]. Recent models [10, 27, 49] have enhanced VLM architecture with high-resolution handling, which is necessary for document understanding tasks [13, 23]. LLaVA-OneVision [27] divides images into 384×384 crops, encodes each part with SigLIP [54], and uses bilinear interpolation to reduce token count up to 8,748 tokens. InternVL2 [10] splits images into 448×448 tiles, processing up to 40 tiles per image with InternViT [10], and applies pixel shuffle to reduce the number of visual tokens, producing up to 10,240 tokens. Qwen-VL2 [49] uses 2D Rotary Positional Embeddings for dynamic resolution support and merges adjacent tokens via an MLP layer, yet still requires over 10,000 tokens for high resolution images. While these models apply token reduction by merging adjacent tokens to preserve structure, they do not address token irrelevance or redundancy, limiting efficiency.

2.2. Visual token reduction

Reducing the number of visual tokens in Vision Transformers (ViT) has been a key focus of the research community for several years. EViT [29] identifies and merges irrelevant tokens by relying on the attention scores between the class token (`[CLS]`) and visual tokens. ToME [6] proposed a simple yet effective approach that iteratively merges similar tokens throughout the ViT layers. Building on these ideas, recent efforts have extended visual token reduction techniques to VLMs. LaVIT [21] used the Gumbel-Softmax [20] to train a mask that selects tokens for retention, merging discarded tokens into retained ones via additional attention layers. LLaVA-PruMerge [44] accelerates LLAVA 1.5 [31] by leveraging the attention scores between the `[CLS]` token and visual tokens in the last layer of the ViT encoder to decide which tokens to retain, while HiRED [3] refines this approach by allocating token budgets based on attention from earlier layers. However, both these methods are only applicable to architectures where a ViT is used and a `[CLS]` token is added to the input visual sequence, making them incompatible with the majority state-of-the-art VLMs, which do not use a `[CLS]` token. Moreover, both methods attribute scores to tokens at the output of the visual encoder, but recent VLMs merge adjacent visual tokens before passing them to the language model. It is unclear how to attribute pre-merging scores to the resulting tokens, making LLaVA-PruMerge and HiRED inapplicable. We note that LLaVA-PruMerge mitigates information loss by merging pruned tokens with retained ones. However, it does not merge similar retained tokens; therefore, it does not address visual redundancy, a typical limitation of pruning-based methods. TRIM [46] prunes tokens based on similarity with pooled text from CLIP [42]. However, as TRIM relies on textual information for pruning, it is less suitable for multi-turn conversations where, in practice, visual tokens would be pruned solely based on the text information available during the image's forward pass, potentially losing crucial information required to answer subsequent prompts. FastV [9] evaluates token importance via average attention scores, which is not compatible with FlashAttention, adding computational overhead for recent VLMs. VTW [30] removes tokens in deeper layers. While this method shows promising results, its reduction of computational costs is limited as visual tokens are only withdrawn in later layers. These previous methods address only one of two issues: the presence of unimportant tokens or visual redundancy. In this work, we introduce **PACT**, a novel approach that tackles both issues simultaneously by pruning irrelevant tokens and merging visually redundant ones.

3. Method

In this section, we present **PACT**, a method that aims to reduce VLMs inference time and memory usage by pruning unimportant tokens and merging visually redundant ones at an early layer L of the language model. **PACT** consists of three steps: First, unimportant tokens are identified. Next, the remaining tokens are clustered. Finally, tokens in each cluster, along with sufficiently close tokens that were initially discarded, are merged. **PACT** operates within a selected layer L of the language model and is applicable in scenarios where visual tokens are fed into the language model, regardless of the architecture of the visual encoder or connector. The three-step process of PACT is illustrated in Figure 1. We denote the hidden states at layer L by $\mathbf{H} \in \mathbb{R}^{n\times d}$, where n is the number of visual tokens and d is the dimensionality of the hidden states. We denote by $\mathbf{K}, \mathbf{Q} \in \mathbb{R}^{n\times n_h\times d_h}$ the key and query matrices for the visual tokens at layer L, where n_h represents the number of attention heads and d_h is the dimensionality of each atten-

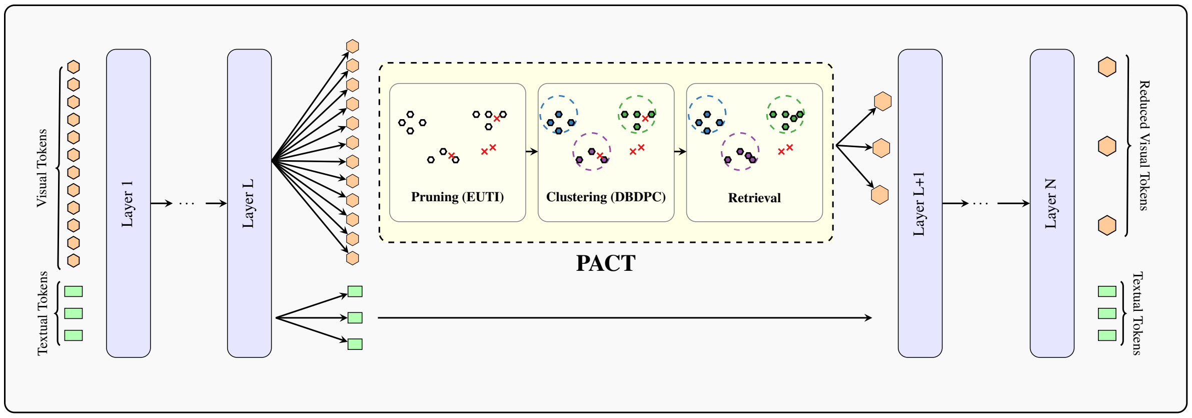


Figure 1. **Simplified illustration of PACT.** This figure illustrates the three-step process of **PACT**: (1) First, **EUTI** is used to prune visual tokens deemed unimportant; (2) Then, **DBDPC** is applied to cluster the remaining tokens, ensuring that the distance between each token and its corresponding cluster center is smaller than the cutoff distance; (3) Finally, initially pruned tokens that are close to cluster centers are reintegrated, and the elements within each cluster are merged to form the reduced set of visual tokens.

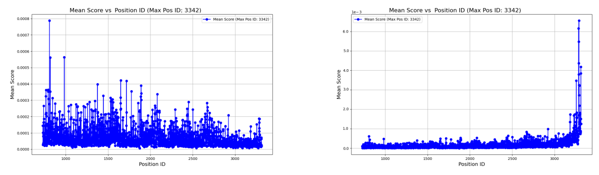
Algorithm 1 EUTI

```
Input: Hidden states $\mathbf{H} \in \mathbb{R}^{n \times d}$; key and query matrices
    $\mathbf{K}, \mathbf{Q} \in \mathbb{R}^{n \times n_h \times d_h}$; pruning percentage $\lambda \in [0, 1]$
Output: Sets of important and unimportant visual tokens
  Step 1: Calculate the global query vector
  $\mathbf{Q}_{\text{global}} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{Q}_i$
  Step 2: Compute the importance score for each visual token
  for all $i = 1, \ldots, n$ do
      $s_i = \frac{1}{n_h} \sum_{j=1}^{n_h} Softmax(\mathbf{k}_i^{(j)} \cdot \mathbf{Q}_{\text{global}}^{(j)}) \cdot \|\mathbf{h}_i\|_2$
  end for
  Step 3: Define sets of important and unimportant tokens
  $S_{\text{important}} = \{i \mid s_i \geq \text{Percentile}(s, \lambda)\}$
  $S_{\text{unimportant}} = \{i \mid s_i < \text{Percentile}(s, \lambda)\}$
  Return $S_{\text{important}}, \; S_{\text{unimportant}}$
```

tion head. For simplicity, we omit the layer index in the notation. We denote the position index of a token by a subscript, while the attention head is indicated by a superscript. For instance, $\mathbf{k}_i^{(j)}$ represents the key vector corresponding to the i-th visual token and the j-th attention head.

3.1. Unimportant tokens identification

A straightforward approach to identifying unimportant tokens at a certain layer L of the used language model is to define the importance of each token as the total attention score that a given token receives from all other tokens [9]. However, this method has three main drawbacks. First, current VLMs utilize FlashAttention [12], which does not support outputting attention scores. Secondly, attention scores are computed with masking, which introduces biases. Tokens at the end of a sequence tend to receive lower average attention scores since fewer tokens attend to them. Calculating the average attention score for each token based solely on the tokens that attend to it can mitigate this masking effect but introduces a new bias: end-of-sequence tokens may exhibit higher scores as they receive attention mainly from nearby tokens. This leads to either earlier or later tokens being pruned more frequently, as shown in Fig. 2. Such positional bias should be avoided, as pruning should depend solely on the information that visual tokens hold, not their position. Finally, relying only on keys and queries at a single layer to determine an importance metric may fail to fully capture the significance of visual tokens across all layers of the language model, mainly because each self-attention layer focuses on different aspects of the visual tokens. To address this, we propose an importance metric that incorporates the accumulated in-



(a) Average attention scores as a function of Position IDs.

(b) Average attention scores relative to non-masked tokens as a function of Position IDs.

Figure 2. **Illustration of the bias induced by the use of the average attention scores across visual tokens as a pruning metric.** In (a), averaging attention over all tokens favors earlier tokens, leading to pruning later tokens more frequently. In (b), averaging only over attending tokens reverses the bias, leading to earlier tokens being pruned more often.

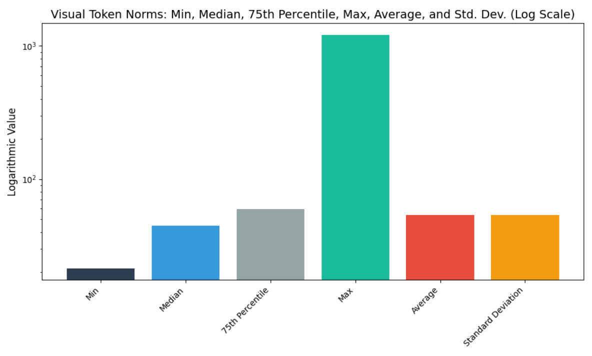


Figure 3. **Illustration of visual token norm statistics at the fourth layer of LLaVA-OneVision-7B.**

formation from the hidden states and the layer-specific information from the keys and queries at an early layer L. We refer to this method as **E**fficient **U**nimportant **T**okens **I**dentification (**EUTI**). We speculate that the norm of hidden states can provide critical information about the importance of each visual token, as they reflect how much information a particular token carries through the network. Figure 3 presents statistics on the hidden state norms of visual tokens at the fourth layer of LLaVA-OneVision-7B, indicating a high variance. This variance suggests that certain visual tokens accumulate more information through residual connections and may therefore be more important for subsequent calculations. To leverage information from both hidden state norms and the key and query vectors, we first compute a global query vector $\mathbf{Q}_{\text{global}}$ as the average of all query vectors across visual tokens:

$$\mathbf{Q}_{\text{global}} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{Q}_i \tag{1}$$

This vector represents the overall query information requested by visual tokens at layer L across all attention heads. The importance score for each visual token is then computed by first taking the dot product between its key and the global query for each attention head. A softmax is applied across visual tokens within each attention head, followed by averaging across attention heads. The final score is obtained by scaling the result with the hidden state norm:

$$s_i = \frac{1}{n_h} \sum_{j=1}^{n_h} \text{Softmax}\left(\mathbf{k}_i^{(j)} \cdot \mathbf{Q}_{\text{global}}^{(j)}\right) \cdot \|\mathbf{h}_i\|_2 \tag{2}$$

Then, we divide the visual tokens into important and unimportant tokens, using a parameter $\lambda \in [0,1]$ to control the percentage of tokens deemed unimportant. The two sets are defined as follows:

$$S_{\text{important}} = \{i \mid s_i \geq \text{Percentile}(s, \lambda)\} \tag{3}$$

$$S_{\text{unimportant}} = \{i \mid s_i < \text{Percentile}(s, \lambda)\} \tag{4}$$

Unimportant tokens can be pruned, or the resulting sets can be combined with a clustering algorithm to further reduce the number of visual tokens, as we will show in the next section. The full **EUTI** algorithm is illustrated in Algorithm 1. We note that in the case where Rotary Embeddings are used [47], we use the keys and queries before their application to avoid any positional bias.

3.2. Clustering-based merging of visual tokens

Distance Bounded Density Peak Clustering Relying solely on the importance scores presented above to prune unimportant tokens can lead to a significant reduction in visual tokens, retaining only important ones. However, redundant information may still be present across retained visual tokens. Therefore, we propose merging the redundant visual tokens using a clustering algorithm. We desire our clustering algorithm to have the following characteristics:

(a) Low computational time.
(b) Avoid assigning points that are far from each other, in terms of feature similarity, into the same cluster.

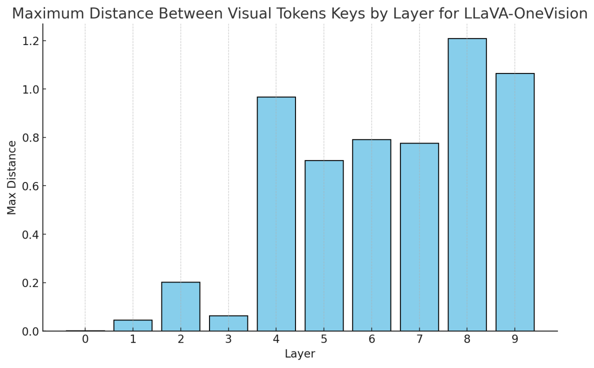


Figure 4. **Illustration of the maximum distance between the keys of visual tokens for the first 10 layers of LLaVA-OneVision-7B before the application of rotary embeddings.**

Table 1. **Throughput ratio, reduction ratio, and GPU memory usage for PACT, FastV, VTW, and ToME applied to LLaVA-OneVision-7B. Results are reported at a 98.6% Approach-to-Reference Metric Ratio.**

	No reduction	PACT (ours)	FastV	VTW	ToME
Reduction Ratio	0%	**71.3%**	50%	25%	40%
LLM Throughput Ratio	100%	**225%**	165%	160%	137%
GPU Maximum Memory Consumption (GB)	27.4	**19.05**	30.4	19.2	21.4

Condition (b) ensures that outliers are not assigned to distant cluster centers, as we speculate that these outliers contain important information and should only be merged with nearby outliers or remain as single points in separate clusters. Condition (b) also guarantees that points in each cluster will be relatively close to each other, which minimizes information loss when assigning a single vector as their representative. The Density Peaks Clustering (DPC) algorithm [5] is appealing in this context because it satisfies condition (a), unlike iterative clustering algorithms like k-means [2]. However, DPC does not satisfy condition (b) as it can form large clusters where boundary points may be distant from each other. The same issue arises with other algorithms such as DBSCAN [14]. Therefore, we propose a new clustering algorithm, which we call **D**istance **B**ounded **D**ensity **P**eaks **C**lustering (**DBDPC**).

DBDPC takes as input a set of vectors $\{\mathbf{u}_i \in \mathbb{R}^{d_1}\}_{i=1}^{q}$, where $q, d_1 \in \mathbb{N}^+$, and outputs a set of clusters. Our algorithm's output depends on two parameters, the cutoff distance $d_c \in \mathbb{R}^+$ and a normalization factor $d_n \in \mathbb{R}^+$, as well as a distance function $d : \mathbb{R}^{d_1} \times \mathbb{R}^{d_1} \to \mathbb{R}^+$. We define the distance between two vectors $\mathbf{u}_i$ and $\mathbf{u}_j$ as:

$$d_{ij} = d(\mathbf{u}_i, \mathbf{u}_j) = 1 - \frac{\mathbf{u}_i \cdot \mathbf{u}_j}{\|\mathbf{u}_i\|_2 \|\mathbf{u}_j\|_2} \tag{5}$$

Then the local density ρ_i is calculated as:

$$\rho_i = \sum_j e^{-d_{ij}/d_n} \tag{6}$$

We process the $\mathbf{u}_i$ vectors from highest to lowest ρ values and designate a vector as a cluster center if its minimum distance from already selected centers is greater than d_c. Each vector $\mathbf{u}_i$ is then assigned to the cluster of the closest center. Our algorithm guarantees that the distance from each vector to its cluster center is less than d_c, thereby satisfying condition (b) stated above. The full **DBDPC** algorithm is detailed in Algorithm 2. The center identification process in **DBDPC** ensures that inter-cluster distances are upper-bounded by $2d_c \times (2 - d_c)$ while distances between cluster centers are lower-bounded by d_c, which we formally prove in Appendix B. We note that several parts of our algorithm are presented as for-loops for clarity. However, all computations are parallelizable on GPU, as there are no dependencies between the elements of each loop, except for the part where we select cluster centers. For this part, we use a recursive algorithm that efficiently identifies an initial set of centers and discarded vectors, thereby reducing the number of vectors to be processed. We explain this in detail in Appendix D. For a comparison between **DBDPC** and DPC, as well as a qualitative comparison with other clustering algorithms, refer to Appendix C.

Which vectors should be used for distance calculation? As previously discussed, the **DBDPC** algorithm operates on a set of vectors that are used for distance calculation. To achieve effective clustering, the dot product between these vectors needs to accurately reflect the similarity between the corresponding visual tokens. Fortunately, transformers address this issue through the QKV self-attention mechanism.

Algorithm 2 DBDPC

```
Input: Cutoff distance $d_c \in \mathbb{R}^+$, normalization factor
  $d_n \in \mathbb{R}^+$, set of vectors $\{\mathbf{u}_i \in \mathbb{R}^{d_1}\}_{i=1}^{q}$
Output: Cluster center indices $C_{\text{centers}}$, element indices in
  each cluster $C_{\text{elements}}$
  for all pairs $(\mathbf{u}_i, \mathbf{u}_j)$ do
    $d_{ij} = 1 - \frac{\mathbf{u}_i \cdot \mathbf{u}_j}{\|\mathbf{u}_i\|_2 \|\mathbf{u}_j\|_2}$
  end for
  for all vectors $\mathbf{u}_i$ do
    $\rho_i = \sum_{j=1}^{q} e^{-d_{ij}/d_n}$
  end for
  Sort vectors by $\rho_i$ in descending order, obtaining indices
  $[i_1, i_2, \ldots, i_q]$
  Initialize $C_{\text{centers}} = \{i_1\}$, $C_{\text{elements}} = \{i_1 : \emptyset\}$
  for all indices $i_k$ in sorted order do
    if $\min_{s \in C_{\text{centers}}} d_{i_k s} > d_c$ then
      $C_{\text{centers}} = C_{\text{centers}} \cup \{i_k\}$
      $C_{\text{elements}}[i_k] = \emptyset$
    end if
  end for
  for all indices $i$ do
    $s_i = argmin_{s \in C_{\text{centers}}} d_{is}$
    $C_{\text{elements}}[s_i] = C_{\text{elements}}[s_i] \cup \{i\}$
  end for
  Return $C_{\text{centers}}, C_{\text{elements}}$
```

Specifically, the key vectors K provide a meaningful representation of each token, tailored for dot product similarity. Therefore, we will use the key vectors in the **DBDPC** algorithm. Formally, we have:

$$C_{\text{centers}}, C_{\text{elements}} = \text{DBDPC}(K') \tag{7}$$

where $K' = \{\mathbf{u}_i \in K \mid i \in S_{\text{important}}\}$ is the subset of keys consisting of elements with indices in $S_{\text{important}}$.

What about unimportant tokens near cluster centers? Tokens initially deemed unimportant but close enough to cluster centers have a high probability of being mislabeled. We add these tokens to the corresponding cluster to limit information loss. Formally, we define a threshold based on a coefficient α, where any token $\mathbf{u}_i$, initially excluded, is added to the cluster of the closest center $s \in C_{\text{centers}}$ if its distance to the center satisfies $d_{is} < \alpha \cdot d_c$. Specifically, the new cluster elements set $C^{(s)}_{\text{elements}}$ is updated as follows:

$$S^{(s)}_{\text{added}} = \{i \in S_{\text{unimportant}} \mid s = argmin_{s' \in C_{\text{centers}}} d_{is'} \text{ and } d_{is} < \alpha \cdot d_c\} \tag{8}$$

$$C^{(s)}_{\text{elements}} \leftarrow C^{(s)}_{\text{elements}} \cup S^{(s)}_{\text{added}} \tag{9}$$

Merging Finally, the hidden states corresponding to the elements in each cluster are merged. Formally, the merged

Algorithm 3 PACT

```
Input: Hidden states $\mathbf{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_n] \in \mathbb{R}^{n \times d}$; key
    and query matrices $\mathbf{K}, \mathbf{Q} \in \mathbb{R}^{n \times n_h \times d_h}$; position IDs
    $\mathbf{P} = [p_1, \ldots, p_n]$; pruning percentage $\lambda \in [0, 1]$; cutoff
    distance $d_c > 0$; tolerance coefficient $\alpha > 0$
Output: Merged hidden states $\mathbf{H}'$; new position IDs $\mathbf{P}'$
  Step 1: Identify important and unimportant tokens
  $S_{\text{important}},\ S_{\text{unimportant}} \leftarrow \text{EUTI}(\mathbf{H},\ \mathbf{K},\ \mathbf{Q},\ p)$
  Step 2: Cluster important tokens with DBDPC
  $\mathbf{K}' \leftarrow \{\mathbf{k}_i \in \mathbf{K} \mid i \in S_{\text{important}}\}$
  $C_{\text{centers}},\ C_{\text{elements}} \leftarrow \text{DBDPC}(\mathbf{K}',\ d_c)$
  Step 3: Assign unimportant tokens to sufficiently
  close clusters.
  for all $i \in S_{\text{unimportant}}$ do
      $s_i \leftarrow argmin_s d_{is}$
      if $d_{is_i} < \alpha.d_c$ then
          $C^{(s_i)}_{\text{elements}} \leftarrow C^{(s_i)}_{\text{elements}} \cup \{i\}$
      end if
  end for
  Step 4: Merge hidden states and assign position IDs
  for all $s \in C_{\text{centers}}$ do
      $\mathbf{h}'_s \leftarrow \frac{1}{|C^{(s)}_{\text{elements}}|} \sum_{i \in C^{(s)}_{\text{elements}}} \mathbf{h}_i$
      $p'_s \leftarrow p_s$
  end for
  Return $\mathbf{H}'$, $\mathbf{P}'$
```

hidden states are computed as:

$$\mathbf{H}' = \left\{ \frac{1}{|C^{(j)}_{\text{elements}}|} \sum_{i \in C^{(j)}_{\text{elements}}} \mathbf{h}_i \;\middle|\; C^{(j)}_{\text{elements}} \in C_{\text{elements}} \right\} \quad (10)$$

Defining the position IDs Accurately assigning position IDs to each vector in the new hidden states $\mathbf{H}'$ is crucial, especially for models using Rotary embeddings, as these IDs determine the input image structure or the temporal dependencies of the input video. In order to achieve a low statistical discrepancy compared to regular inference, we assign the position ID for each vector from H' as its corresponding cluster center. The full **PACT** pipeline is shown in Algorithm 3. When Rotary Embeddings are used, **DBDPC** uses the keys after these embeddings are applied, whereas **EUTI** uses the keys and queries before applying these embeddings. For clarity, we omit this detail in Algorithm 3. We also note that both **DBDPC** and **EUTI**, as well as **PACT**, do not use textual tokens. Therefore, visual token reduction is performed independently of the textual context, making our method well-suited for multi-turn conversations.

Proportional attention Merging tokens reduces their influence in the attention mechanism and can therefore deteriorate performance if many important tokens are merged together. To mitigate this, we employ proportional attention.

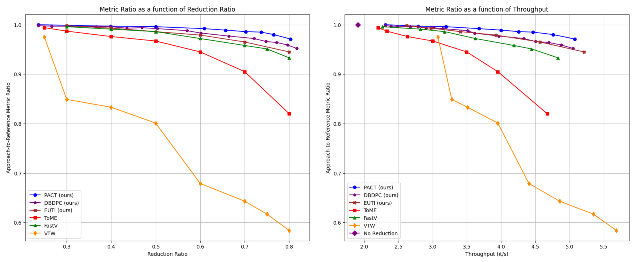


Figure 5. **Comparison between PACT, DBDPC, and EUTI against other visual token reduction methods across various reduction ratios applied on LLaVA-OneVision-7B.**

Let K, Q, and V denote the keys, queries, and values at a layer L', where $L' \geq L$. For each attention head j, the attention scores are calculated as follows:

$$A^{(j)} = \text{softmax}\left(\frac{Q^{(j)} K^{(j)\top}}{\sqrt{d_{l'}}} + \log \mathbf{W} + \mathbf{B} \right) \quad (11)$$

where $d_{l'}$ is the dimensionality of the query for each attention head. Here, $\mathbf{W}$ is a matrix representing the weight of each token, and $\mathbf{B}$ is the attention mask. Specifically, for visual tokens, w_{i_0,i_1} represents the size of the cluster corresponding to token i_1, for any value of i_0. For each textual token at position t, $w_{i_0,t} = 1$, as they remain unmerged, retaining a weight of one. By scaling the attention scores based on $\mathbf{W}$, the model effectively treats each visual token as if it represents multiple tokens. We note that when using proportional attention, we use PyTorch's scaled dot-product attention, which produces similar results to the official FlashAttention implementation while supporting custom masks.

Selecting the layer L for token reduction: To ensure maximum computational gain, we must choose an early layer L for visual token reduction. However, we also require that the keys at the selected layer are not too similar, allowing for effective clustering and pruning. Thus, we select the earliest layer where the maximum distance between keys is sufficiently high. Figure 4 shows that in the initial layers of LLaVA-OneVision-7B, the keys corresponding to visual tokens are quite similar, indicating a lack of distinctive features necessary for effective pruning and clustering.

4. Experiments

4.1. Evaluation datasets

We evaluate the effectiveness of **PACT** using diverse benchmarks, similar to those used for LLaVA-OneVision-7B, covering single-image, multi-image, and video tasks. We use AI2D [22], TextVQA [45], ChartQA [37], DocVQA [38], and InfographicVQA [39] to assess **PACT**'s ability to reduce visual tokens while maintaining performance

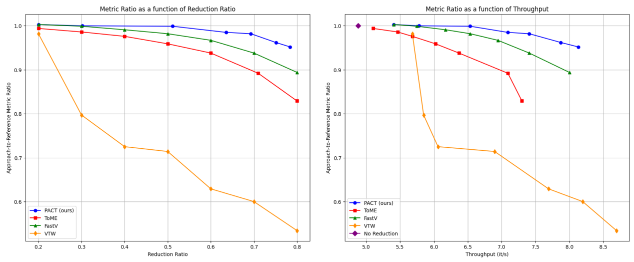


Figure 6. **Comparison between PACT and other visual token reduction methods across various reduction ratios applied on Qwen2-VL-7B-Instruct.**

in text-rich documents. To test reasoning across multiple disciplines, we use MME [15], MMBench [32], MMVet [51], MathVerse [57], MathVista [34], MMMU [53], MMStar [8], and ScienceQA [33]. Additionally, Vibe-Eval [40], MM-LiveBench [26], and LLaVA-Bench-Wilder [25] evaluate its robustness in real-world scenarios and visual chat contexts. We use LLaVA-Interleave Bench [25] and MuirBench [48] to examine **PACT**'s efficiency in token reduction while preserving inter-image reasoning. To assess performance in video comprehension tasks, we use ActivityNet-QA [52], MLVU [58], VideoMME [16], EgoSchema [36], and PerceptionTest [41]. Finally, Video-ChatGPT [35] evaluates the method's effectiveness in dialogue-based video interaction.

4.2. Evaluation setup

In our comparison, we include approaches where the reduction is applied at a single layer, similar to **PACT**, such as FastV and clustering-based visual token reduction. For these approaches, we refer to the reduction ratio as the relative reduction in the number of visual tokens, defined as $1 - \frac{\text{number of visual tokens after reduction}}{\text{number of visual tokens before reduction}}$. For all these approaches, we use the same value of L and vary hyperparameters to test across different reduction ratios. For methods that use progressive token reduction, like ToME [6], or apply reduction after the visual encoder, as PruMerge and HiReD, or when the reduction ratio cannot be controlled at a fixed layer, such as VTW, we adjust the parameters of these approaches to achieve the same average number of visual tokens across all layers as the one-layer reduction methods for a given reduction ratio. When evaluating clustering algorithms for visual token reduction, we apply proportional attention, as it consistently improves performance across all clustering algorithms, especially at high reduction ratios. Additionally, it is crucial to correctly assign position IDs to the resulting reduced set of visual tokens. Details on the assignment strategy are presented in Appendix E. When reporting processing time or throughput, we take into account the total time required by both the language model and the reduction algorithm per input element. In the next section, we base our comparison on a metric called the Approach-to-Reference Metric Ratio, defined as the average of the ratio of the metric of the tested approach to the metric obtained without visual token reduction across all test datasets. Formally we have *Approach-to-Reference Metric Ratio* $= \frac{1}{N}\sum_{i=1}^{N}\frac{\text{Metric}_{\text{with reduction}}(i)}{\text{Metric}_{\text{no reduction}}(i)}$ where N is the total number of test datasets. This metric indicates how much of the original model capacity is retained. It is important to note that when using ToME for visual token reduction, a reduction ratio greater than 50% can't be achieved if the number of visual tokens is reduced by a fixed amount in each layer, as suggested in [6]. Instead, we use a scheduler to achieve higher reduction ratios, which we explain in Appendix F. More details on the hyperparameters used for evaluating **PACT** are provided in Appendix G. We follow the same dataset splits and metrics used for evaluating LLaVA-OneVision wherever feasible. More details are provided in Appendix H. Note that all experiments were conducted on a single A100 GPU.

4.3. Results

We compare **PACT** with FastV [9], VTW [30], ToME [6], PruMerge [44] and HiRED [3] on LLaVA-OneVision-7B, InternVL2-8B, Qwen2-VL-7B-Instruct and LLaVA-1.6-Mistral-7B. Since HiRED and PruMerge are only applicable to LLaVA-1.6, we exclude them from other comparisons. As shown in figures 5, 6, 7, and 8 **PACT** con-

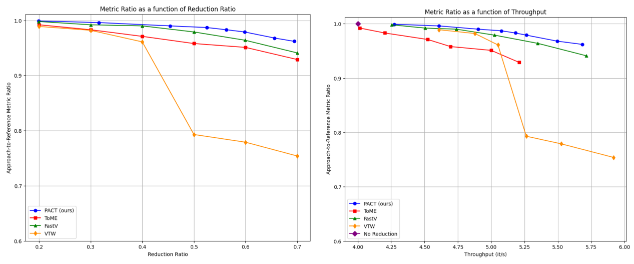


Figure 7. **Comparison between PACT and other visual token reduction methods across various reduction ratios applied on InternVL2-8B.**

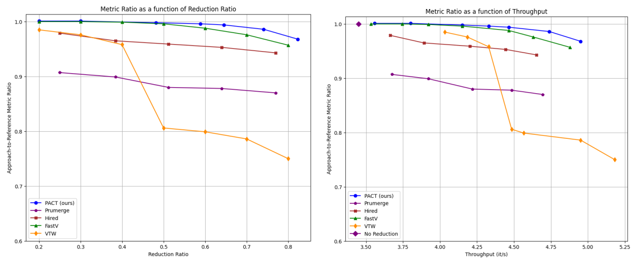


Figure 8. **Comparison between PACT and other visual token reduction methods across various reduction ratios applied on LLaVA-1.6-Mistral-7B.**

Table 2. **Comparison of PACT with FastV, VTW, and ToME on LLaVA-OneVision-7B. Algo. Time** refers to the average time the algorithm takes per input element, measured in seconds. **Proc. Time** refers to the average time taken by both the language model and the reduction algorithm per input element. **Red. Ratio** stands for average Reduction Ratio. The Algo. Time for VTW is nearly zero, and thus omitted. The different visual token reduction methods are evaluated at the same reduction ratio as **PACT**.

Dataset	No reduction		PACT (Ours)				FastV			VTW		ToME		
	Metric	Proc. Time	Metric	Red. Ratio	Proc. Time	Algo. Time	Metric	Proc. Time	Algo. Time	Metric	Proc. Time	Metric	Proc. Time	Algo. Time
VideoMME	58.5	0.792	**57.6**	65.6%	0.369	0.021	57.0	0.371	0.040	46.9	0.296	57.0	0.417	0.091
MME	1579	0.554	1564.0	70.2%	0.243	0.017	**1576.0**	0.244	0.016	842.0	0.231	1556.9	0.317	0.084
DocVQA	87.2	1.088	**84.4**	67.9%	0.519	0.026	84.3	0.524	0.051	10.5	0.449	61.9	0.576	0.099
MLVU	65.2	0.795	**64.7**	66.4%	0.361	0.022	62.9	0.369	0.040	54.4	0.312	63.4	0.417	0.092
LLaVA-Interleave	64.1	0.249	**64.0**	69.7%	0.133	0.010	58.9	0.139	0.007	32.4	0.123	50.3	0.192	0.068
ChartQA	79.9	0.671	76.5	68.5%	0.341	0.019	**77.0**	0.342	0.016	16.6	0.307	63.4	0.402	0.082
MMBench	80.6	0.249	**80.3**	69.3%	0.135	0.010	79.0	0.140	0.005	52.4	0.125	79.7	0.193	0.066
MuirBench	42.0	0.384	**43.1**	67.8%	0.178	0.013	40.4	0.178	0.009	34.9	0.162	40.5	0.233	0.072
ScienceQA	95.9	0.238	**93.8**	69.6%	0.133	0.010	91.6	0.137	0.006	80.0	0.124	**93.8**	0.190	0.066
MMMU	49.2	0.139	**48.9**	70.4%	0.104	0.007	**48.9**	0.106	0.003	43.5	0.093	48.6	0.124	0.062
AI2D	81.5	0.382	**81.0**	69.8%	0.186	0.013	79.4	0.191	0.014	69.7	0.177	79.7	0.244	0.073
InfographicVQA	66.0	0.895	**61.9**	64.7%	0.481	0.023	58.6	0.483	0.040	24.5	0.408	48.3	0.607	0.130
MMStar	62.0	0.297	**60.1**	69.7%	0.147	0.011	58.6	0.152	0.007	37.2	0.165	**60.1**	0.229	0.069
ActivityNetQA	54.5	0.921	**55.1**	70.0%	0.419	0.029	53.7	0.425	0.042	36.6	0.394	54.1	0.513	0.203
MM-LiveBench	73.1	4.434	**71.7**	67.5%	3.212	0.047	64.4	3.221	0.044	41.0	3.080	64.2	3.607	0.102
LLaVA-Wilder	71.0	10.10	**71.5**	70.0%	8.262	0.035	71.0	8.263	0.025	48.8	7.515	68.0	7.926	0.085
MathVerse	16.8	0.831	16.6	74.2%	0.361	0.021	16.1	0.382	0.036	**17.6**	0.301	16.5	0.559	0.150
MathVista	63.3	0.440	**62.0**	70.7%	0.271	0.015	59.5	0.272	0.016	38.5	0.260	55.0	0.338	0.071
MMVet	58.0	4.602	**58.4**	70.4%	3.793	0.035	51.7	3.795	0.036	15.7	3.652	47.2	4.115	0.212
Vibe-Eval	41.6	5.153	**39.1**	71.1%	3.709	0.032	38.2	3.714	0.047	12.3	3.550	31.2	4.317	0.095
VideoChatGPT	3.25	2.972	**3.25**	67.2%	1.863	0.029	3.22	1.866	0.040	1.92	1.320	3.19	1.975	0.205
EgoSchema	60.1	0.811	**60.1**	66.6%	0.351	0.021	58.7	0.353	0.044	44.8	0.297	59.8	0.391	0.091
PerceptionTest	52.1	0.801	**52.3**	66.9%	0.353	0.023	51.7	0.357	0.040	45.0	0.296	51.1	0.393	0.090
TextVQA	75.8	0.690	75.0	67.2%	0.332	0.023	**75.5**	0.336	0.029	11.6	0.287	62.5	0.392	0.087

sistently outperforms other methods at both equal reduction ratios and equal throughput across all four models. VTW experiences a significant performance drop for reduction ratios above 40%, indicating that removing all visual tokens is only effective when done in later layers. FastV and ToME struggle at high reduction ratios, while PruMerge and HiRED exhibit degradation even at low reduction ratios. Meanwhile, **PACT** maintains acceptable performance even at high reduction ratios. Table 2 and Table 3 shows that **PACT** outperforms other approaches on most of the test datasets when applied on LLaVA-OneVision-7B and Qwen2-VL-7B-Instruct. The same conclusion applies to other models, with detailed results provided in Appendix I. In Tab. 1, we report the reduction ratio, throughput, and maximum GPU memory consumption of the different approaches at an equal Approach-to-Reference Metric Ratio of 98.6% on LLaVA-OneVision-7B. **PACT** significantly outperforms the other methods, achieving a reduction ratio of 71.3%, a GPU memory reduction of 31%, and a 225% speedup in the language model's inference time. The per-dataset results used to compute these metrics are shown in Tab. 5. Tab. 1 also indicates that when using FastV, the maximum GPU memory consumption is relatively high due to the costly computation of attention scores. We further compare **DBDPC** against agglomerative clustering [1], k-means [2], Density Peaks Clustering (DPC) [5], and DBSCAN [14], with results presented in Fig. 9. The graphs reveal that **DBDPC** consistently outperforms other clustering algorithms for visual token reduction, exhibiting less performance degradation at equal reduction ratios and demonstrating improved computational efficiency, leading to better throughput. These results validate our hypothesis that, for an effective visual token reduction, it is necessary to ensure that the distances between elements within each cluster do not exceed a predefined threshold. Fig. 5 also shows that **EUTI** consistently outperforms FastV at equal reduction ratios and is less costly, as it does not require the computation of attention scores. In addition, unlike FastV, **EUTI** does not introduce a GPU memory overhead[1]. We provide additional numerical results in Appendix I.

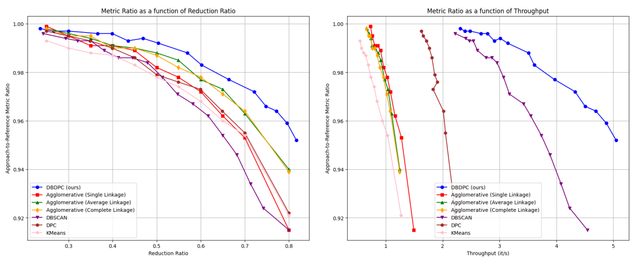


Figure 9. **Comparison of DBDPC and other clustering algorithms for visual token reduction at different reduction ratios on LLaVA-OneVision-7B.**

4.4. Ablation study

Fig. 5 shows that **PACT** consistently outperforms both **DBDPC** and **EUTI** across various reduction ratios. This confirms that combining clustering and pruning techniques

[1]EUTI achieves roughly the same memory reduction as PACT.

Table 3. **Comparison of PACT with FastV, VTW, and ToME applied on Qwen2-VL-7B-Instruct across Various Datasets.**

Dataset	No Reduction		PACT (Ours)			FastV		VTW		ToME	
	Metric	Proc. Time	Metric	Red. Ratio	Proc. Time	Metric	Proc. Time	Metric	Proc. Time	Metric	Proc. Time
MME	1654.5	0.238	**1666.5**	86.3%	0.110	1500.0	0.111	709.24	0.120	1610.9	0.140
DocVQA	93.9	0.516	**90.5**	77.5%	0.294	86.6	0.298	8.5	0.249	42.9	0.350
TextVQA	81.8	0.155	**80.4**	67.5%	0.132	79.9	0.135	13.2	0.118	66.2	0.151
InfographicVQA	74.6	0.478	**70.6**	69.7%	0.278	63.3	0.273	21.5	0.225	43.9	0.299
ChartQA	80.8	0.145	**76.0**	61.1%	0.135	69.2	0.134	12.9	0.123	55.1	0.155
MMBench	77.6	0.074	**77.1**	51.5%	0.077	**77.1**	0.074	76.9	0.073	75.9	0.080
MuirBench	40.7	0.159	**41.2**	76.9%	0.113	40.4	0.112	37.9	0.111	75.8	0.125
MMMU	51.4	0.109	**51.2**	72.6%	0.093	49.3	0.092	45.4	0.088	48.9	0.105
AI2D	79.9	0.105	**78.4**	64.2%	0.096	76.2	0.097	69.0	0.087	76.4	0.115
MMStar	56.0	0.072	**54.8**	61.3%	0.072	51.5	0.067	40.3	0.065	53.8	0.077
EgoSchema	62.1	0.360	**61.6**	60.0%	0.207	60.2	0.212	46.3	0.190	61.2	0.230
MathVerse	25.3	0.620	**24.5**	82.2%	0.393	23.7	0.396	13.9	0.296	18.1	0.651
MathVista	59.2	0.249	**57.7**	73.3%	0.195	56.4	0.194	36.8	0.165	53.5	0.275
MMVet	24.9	4.700	**25.1**	80.3%	3.820	22.3	3.830	2.7	3.650	16.7	4.780
Vibe-Eval	47.5	3.200	**46.1**	85.0%	2.310	44.3	2.375	13.1	1.993	29.6	3.620
LLaVA-Interleave	35.9	0.120	**35.5**	73.7%	0.100	34.7	0.101	33.2	0.096	35.3	0.125
MM-LiveBench	72.6	3.970	**70.7**	77.1%	3.040	63.0	3.120	39.7	2.970	57.6	4.450

yields better performance than using each approach independently, as the combined method addresses both visual tokens irrelevance and redundancy. We ablate several components of the **DBDPC** algorithm and present the results in Fig. 10. First, we ablate token merging by selecting the center of each cluster as the representative token instead of merging tokens within each cluster. We also ablate the use of proportional attention. Additionally, we ablate the assignment of position IDs to the reduced set of tokens and experiment with two alternatives: using the mean of position IDs of all elements in each cluster and assigning position IDs sequentially after reordering the reduced set according to the mean of position IDs. Finally, we ablate the use of key vectors in the clustering process and instead use hidden states. Our results show that each ablated component contributes positively to the performance of the **DBDPC** algorithm. Notably, correctly assigning position IDs to the reduced set is crucial, as these position IDs reflect the structure of input images and the temporal order of input videos. Additionally, proportional attention proves effective at higher reduction ratios, while token merging enhances performance once the reduction ratio exceeds 50%. The figure also confirms that keys are better suited for cosine similarity-based distance calculations, as they are naturally used in dot products within the attention mechanism. We perform two separate ablations on Eq. (2) of the **EUTI** algorithm. The first ablation removes the use of hidden state norms, while the second ablates the use of the global query, which corresponds to using only the hidden state norms. The results in Fig. 10 show that combining both the global query-based score and the norm of hidden states consistently leads to better results than using either metric alone, suggesting that they provide complementary information about the importance of each visual token. Finally, we ablate the pruned token recovery module in **PACT** by setting α to zero, with results presented in Fig. 11. The plot shows that reintegrating visual tokens initially deemed unimportant but close enough to a cluster center consistently enhances performance across different reduction ratios, supporting our hypothesis that these tokens were likely mislabeled by the **EUTI** module. Figure 11 also shows the effect of the choice of the reduction layer on **PACT**'s performance, demonstrating the effectiveness of our reduction layer identification approach. We provide additional numerical results in Appendix J.

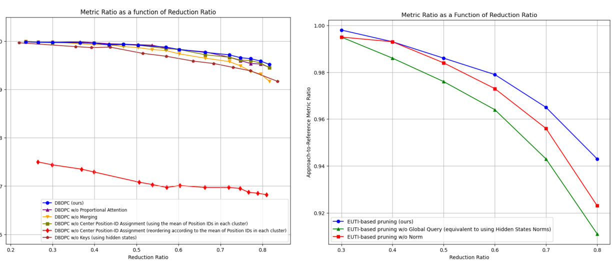


Figure 10. **Ablation study of DBDPC and EUTI on LLaVA-OneVision-7B.**

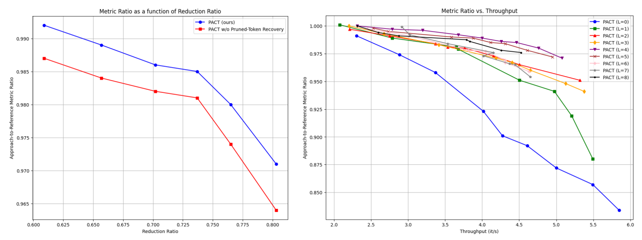


Figure 11. **Ablation study of PACT on LLaVA-OneVision-7B.**

5. Conclusion

In this work, we presented **PACT**, a method that addresses both visual token irrelevance and redundancy. **PACT** is a plug-and-play solution that does not require additional training. It does not rely on textual tokens for visual token reduction, making it well-suited for multi-turn conversations. Additionally, it operates independently of the visual encoder and connector architecture, making it broadly applicable across various Visual Language Models. Our results confirm that the number of visual tokens in Visual Language Models is unnecessarily large and provide valuable insights for effective token reduction. This opens the door for future work in designing more efficient connectors and architectures for VLMs.

6. Acknowledgments

This work received financial support from Crédit Agricole S.A. through the research chair with Ecole Polytechnique on Trustworthy and Responsible AI. This work was granted access to the HPC resources of IDRIS under the allocation 2024-AD011014793R1 made by GENCI.

References

[1] Marcel R Ackermann, Johannes Blömer, Daniel Kuntze, and Christian Sohler. Analysis of agglomerative clustering. *Algorithmica*, 69:184–215, 2014. 8, 3

[2] Mohiuddin Ahmed, Raihan Seraj, and Syed Mohammed Shamsul Islam. The k-means algorithm: A comprehensive survey and performance evaluation. *Electronics*, 9(8):1295, 2020. 5, 8, 3

[3] Kazi Hasan Ibn Arif, JinYi Yoon, Dimitrios S Nikolopoulos, Hans Vandierendonck, Deepu John, and Bo Ji. Hired: Attention-guided token dropping for efficient inference of high-resolution vision-language models in resource-constrained environments. *arXiv preprint arXiv:2408.10945*, 2024. 2, 7

[4] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A frontier large vision-language model with versatile abilities. *arXiv preprint arXiv:2308.12966*, 2023. 1

[5] Panthadeep Bhattacharjee and Pinaki Mitra. A survey of density based clustering algorithms. *Frontiers of Computer Science*, 15:1–27, 2021. 5, 8, 3

[6] Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your vit but faster. *arXiv preprint arXiv:2210.09461*, 2022. 2, 7, 4

[7] Keqin Chen, Zhao Zhang, Weili Zeng, Richong Zhang, Feng Zhu, and Rui Zhao. Shikra: Unleashing multimodal llm's referential dialogue magic. *arXiv preprint arXiv:2306.15195*, 2023. 2

[8] Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, et al. Are we on the right way for evaluating large vision-language models? *arXiv preprint arXiv:2403.20330*, 2024. 7

[9] Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models. *arXiv preprint arXiv:2403.06764*, 2024. 2, 3, 7

[10] Zhe Chen, Weiyun Wang, Hao Tian, Shenglong Ye, Zhangwei Gao, Erfei Cui, Wenwen Tong, Kongzhi Hu, Jiapeng Luo, Zheng Ma, et al. How far are we to gpt-4v? closing the gap to commercial multimodal models with open-source suites. *arXiv preprint arXiv:2404.16821*, 2024. 2

[11] Yunfei Chu, Jin Xu, Xiaohuan Zhou, Qian Yang, Shiliang Zhang, Zhijie Yan, Chang Zhou, and Jingren Zhou. Qwen-audio: Advancing universal audio understanding via unified large-scale audio-language models. *arXiv preprint arXiv:2311.07919*, 2023. 1

[12] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness, 2022. 1, 3

[13] Mohamed Dhouib, Ghassen Bettaieb, and Aymen Shabou. Docparser: End-to-end ocr-free information extraction from visually rich documents. In *International Conference on Document Analysis and Recognition*, pages 155–172. Springer, 2023. 2

[14] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, pages 226–231, 1996. 5, 8, 3

[15] Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, et al. Mme: A comprehensive evaluation benchmark for multimodal large language models. *arXiv preprint arXiv:2306.13394*, 2023. 7

[16] Chaoyou Fu, Yuhan Dai, Yondong Luo, Lei Li, Shuhuai Ren, Renrui Zhang, Zihan Wang, Chenyu Zhou, Yunhang Shen, Mengdan Zhang, et al. Video-mme: The first-ever comprehensive evaluation benchmark of multi-modal llms in video analysis. *arXiv preprint arXiv:2405.21075*, 2024. 7

[17] Tao Gong, Chengqi Lyu, Shilong Zhang, Yudong Wang, Miao Zheng, Qian Zhao, Kuikun Liu, Wenwei Zhang, Ping Luo, and Kai Chen. Multimodal-gpt: A vision and language model for dialogue with humans. *arXiv preprint arXiv:2305.04790*, 2023. 2

[18] Jiaming Han, Kaixiong Gong, Yiyuan Zhang, Jiaqi Wang, Kaipeng Zhang, Dahua Lin, Yu Qiao, Peng Gao, and Xiangyu Yue. Onellm: One framework to align all modalities with language. *arXiv preprint arXiv:2312.03700*, 2023. 1

[19] Shaohan Huang, Li Dong, Wenhui Wang, Yaru Hao, Saksham Singhal, Shuming Ma, Tengchao Lv, Lei Cui, Owais Khan Mohammed, Qiang Liu, et al. Language is not all you need: Aligning perception with language models. *arXiv preprint arXiv:2302.14045*, 2023. 1

[20] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016. 2

[21] Yang Jin, Kun Xu, Liwei Chen, Chao Liao, Jianchao Tan, Bin Chen, Chenyi Lei, An Liu, Chengru Song, Xiaoqiang Lei, et al. Unified language-vision pretraining with dynamic discrete visual tokenization. *arXiv preprint arXiv:2309.04669*, 2023. 2

[22] Aniruddha Kembhavi, Mike Salvato, Eric Kolve, Minjoon Seo, Hannaneh Hajishirzi, and Ali Farhadi. A diagram is worth a dozen images. In *European conference on computer vision*, pages 235–251. Springer, 2016. 6

[23] Geewook Kim, Teakgyu Hong, Moonbin Yim, JeongYeon Nam, Jinyoung Park, Jinyeong Yim, Wonseok Hwang, Sangdoo Yun, Dongyoon Han, and Seunghyun Park. Ocr-free document understanding transformer. In *Computer Vision – ECCV 2022*, pages 498–517, Cham, 2022. Springer Nature Switzerland. 2

[24] Bohao Li, Rui Wang, Guangzhi Wang, Yuying Ge, Yixiao Ge, and Ying Shan. Seed-bench: Benchmarking multimodal llms with generative comprehension. *arXiv preprint arXiv:2307.16125*, 2023. 4

[25] Bo Li, Kaichen Zhang, Hao Zhang, Dong Guo, Renrui Zhang, Feng Li, Yuanhan Zhang, Ziwei Liu, and Chunyuan Li. Llava-next: Stronger llms supercharge multimodal capabilities in the wild, 2024. 7

[26] Bo Li, Peiyuan Zhang, Kaichen Zhang, Fanyi Pu, Xinrun Du, Yuhao Dong, Haotian Liu, Yuanhan Zhang, Ge Zhang, Chunyuan Li, and Ziwei Liu. Lmms-eval: Accelerating the development of large multimoal models, 2024. 7, 4, 5

[27] Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*, 2024. 2, 4, 5

[28] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597*, 2023. 2

[29] Youwei Liang, Chongjian Ge, Zhan Tong, Yibing Song, Jue Wang, and Pengtao Xie. Not all patches are what you need: Expediting vision transformers via token reorganizations. *arXiv preprint arXiv:2202.07800*, 2022. 2

[30] Zhihang Lin, Mingbao Lin, Luxi Lin, and Rongrong Ji. Boosting multimodal large language models with visual tokens withdrawal for rapid inference. *arXiv preprint arXiv:2405.05803*, 2024. 2, 7

[31] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *arXiv preprint arXiv:2304.08485*, 2023. 1, 2

[32] Yuan Liu, Haodong Duan, Yuanhan Zhang, Songyang Zhang Bo Li, and Wangbo Zhao. Mmbench: Is your multi-modal model an all-around player? *arXiv:2307.06281*, 2023. 7

[33] Pan Lu, Swaroop Mishra, Tony Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering, 2022. 7

[34] Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. In *International Conference on Learning Representations (ICLR)*, 2024. 7

[35] Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad Shahbaz Khan. Video-chatgpt: Towards detailed video understanding via large vision and language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL 2024)*, 2024. 7

[36] Karttikeya Mangalam, Raiymbek Akshulakov, and Jitendra Malik. Egoschema: A diagnostic benchmark for very long-form video language understanding. *Advances in Neural Information Processing Systems*, 36:46212–46244, 2023. 7

[37] Ahmed Masry, Do Xuan Long, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. Chartqa: A benchmark for question answering about charts with visual and logical reasoning. *arXiv preprint arXiv:2203.10244*, 2022. 6

[38] Minesh Mathew, Dimosthenis Karatzas, and CV Jawahar. Docvqa: A dataset for vqa on document images. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 2200–2209, 2021. 6

[39] Minesh Mathew, Viraj Bagal, Rubèn Tito, Dimosthenis Karatzas, Ernest Valveny, and CV Jawahar. Infographicvqa. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1697–1706, 2022. 6

[40] Piotr Padlewski, Max Bain, Matthew Henderson, Zhongkai Zhu, Nishant Relan, Hai Pham, Donovan Ong, Kaloyan Aleksiev, Aitor Ormazabal, Samuel Phua, Ethan Yeo, Eugenie Lamprecht, Qi Liu, Yuqi Wang, Eric Chen, Deyu Fu, Lei Li, Che Zheng, Cyprien de Masson d'Autume, Dani Yogatama, Mikel Artetxe, and Yi Tay. Vibe-eval: A hard evaluation suite for measuring progress of multimodal language models. *arXiv preprint arXiv:2405.02287*, 2024. 7

[41] Viorica Patraucean, Lucas Smaira, Ankush Gupta, Adria Recasens, Larisa Markeeva, Dylan Banarse, Skanda Koppula, Mateusz Malinowski, Yi Yang, Carl Doersch, et al. Perception test: A diagnostic benchmark for multimodal video models. *Advances in Neural Information Processing Systems*, 36, 2024. 7

[42] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021. 2

[43] Erich Schubert. *A Triangle Inequality for Cosine Similarity*, page 32–44. Springer International Publishing, 2021. 1

[44] Yuzhang Shang, Mu Cai, Bingxin Xu, Yong Jae Lee, and Yan Yan. Llava-prumerge: Adaptive token reduction for efficient large multimodal models. *arXiv preprint arXiv:2403.15388*, 2024. 2, 7

[45] Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. Towards vqa models that can read. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8317–8326, 2019. 6

[46] Dingjie Song, Wenjun Wang, Shunian Chen, Xidong Wang, Michael Guan, and Benyou Wang. Less is more: A simple yet effective token reduction method for efficient multimodal llms. *arXiv preprint arXiv:2409.10994*, 2024. 2

[47] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024. 4

[48] Fei Wang, Xingyu Fu, James Y Huang, Zekun Li, Qin Liu, Xiaogeng Liu, Mingyu Derek Ma, Nan Xu, Wenxuan Zhou, Kai Zhang, et al. Muirbench: A comprehensive benchmark for robust multi-image understanding. *arXiv preprint arXiv:2406.09411*, 2024. 7

[49] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution, 2024. 2

[50] Qinghao Ye, Haiyang Xu, Guohai Xu, Jiabo Ye, Ming Yan, Yiyang Zhou, Junyang Wang, Anwen Hu, Pengcheng Shi, Yaya Shi, et al. mplug-owl: Modularization empowers large language models with multimodality. *arXiv preprint arXiv:2304.14178*, 2023. 2

[51] Weihao Yu, Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Zicheng Liu, Xinchao Wang, and Lijuan Wang. Mm-vet: Evaluating large multimodal models for integrated capabilities, 2023. 7

[52] Zhou Yu, Dejing Xu, Jun Yu, Ting Yu, Zhou Zhao, Yueting Zhuang, and Dacheng Tao. Activitynet-qa: A dataset for understanding complex web videos via question answering. In *AAAI*, pages 9127–9134, 2019. 7

[53] Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, Cong Wei, Botao Yu, Ruibin Yuan, Renliang Sun, Ming Yin, Boyuan Zheng, Zhenzhu Yang, Yibo Liu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. In *Proceedings of CVPR*, 2024. 7

[54] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training, 2023. 2

[55] Duzhen Zhang, Yahan Yu, Chenxing Li, Jiahua Dong, Dan Su, Chenhui Chu, and Dong Yu. Mm-llms: Recent advances in multimodal large language models. *arXiv preprint arXiv:2401.13601*, 2024. 1

[56] Hang Zhang, Xin Li, and Lidong Bing. Video-llama: An instruction-tuned audio-visual language model for video understanding. *arXiv preprint arXiv:2306.02858*, 2023. 1

[57] Renrui Zhang, Dongzhi Jiang, Yichi Zhang, Haokun Lin, Ziyu Guo, Pengshuo Qiu, Aojun Zhou, Pan Lu, Kai-Wei Chang, Peng Gao, and Hongsheng Li. Mathverse: Does your multi-modal llm truly see the diagrams in visual math problems?, 2024. 7

[58] Junjie Zhou, Yan Shu, Bo Zhao, Boya Wu, Shitao Xiao, Xi Yang, Yongping Xiong, Bo Zhang, Tiejun Huang, and Zheng Liu. Mlvu: A comprehensive benchmark for multi-task long video understanding. *arXiv preprint arXiv:2406.04264*, 2024. 7

PACT: Pruning and Clustering-Based Token Reduction for Faster Visual Language Models Supplementary Materials

A. On the density peaks clustering algorithm

Density Peak Clustering (DPC) is a clustering algorithm that identifies cluster centers based on local density and the distance to points with higher density, denoted as δ_i. The density, ρ_i, can be measured by counting the number of points within a cutoff distance d_c from $\mathbf{u}_i$, or by using a Gaussian function where nearby points contribute more to the density, $\rho_i = \sum_j \exp\left(-\left(\frac{d_{ij}}{d_c}\right)^2\right)$. Points with high ρ_i and δ_i values are selected as cluster centers. This selection can be done by defining a threshold t and designating points as cluster centers where $\rho_i \cdot \delta_i \geq t \times \max(\rho_i \cdot \delta_i)$, or by selecting a fixed percentage. Other points are then assigned to the cluster of the nearest higher-density point, iterating from the highest to the lowest density. This process can create clusters of varying shapes, where the maximum distance between elements within a cluster can be extremely large. In extreme cases, the two farthest points in the input data can end up in the same cluster.

B. DBDPC Characteristics

This section aims to prove that **DBDPC** guarantees that: Each element's distance to its assigned cluster center is at most d_c and that all cluster centers are at least d_c apart.
Assume, for contradiction, that at least one of the following statements is false:

1. There exists an element i assigned to a cluster such that its distance to the cluster center is greater than d_c, i.e., $d_{is} > d_c$.
2. There exist two cluster centers s_1, s_2 such that their pairwise distance is at most d_c, i.e., $d_{s_1 s_2} \leq d_c$.

Contradiction for Assumption 1 In **DBDPC**, each element i is assigned to its closest cluster center:

$$s_i = \arg\min_{s \in C_{\text{centers}}} d_{is}.$$

If $d_{is} > d_c$ for a given center s, then we have $d_{is'} > d_c$ for all centers. However, in the **DBDPC** selection process, an element is assigned as a cluster center if its minimum distance to already selected centers is over d_c. Thus, i should have been selected as a new cluster center, and its distance to the closest cluster center would be zero, which leads to a contradiction, proving that every element satisfies $d_{is} \leq d_c$.

Contradiction for Assumption 2 Assume, without loss of generality, that s_2 is chosen after s_1. By the center selection criterion, a new center s_2 is added only if:

$$\min_{s \in C_{\text{centers}}} d_{s_2 s} > d_c.$$

If $d_{s_1 s_2} \leq d_c$, then s_2 shouldn't be selected as a cluster center, which leads to a contradiction. Thus, no two centers can be closer than d_c.
Inter-cluster distance upper-bound : Here we will refer to cosine similarity by sim. Let's x and y be two points in the same cluster, and s their cluster center. Since each point $\mathbf{x}$ is within d_c of its cluster center $\mathbf{s}$ and the distance used in the **DBDPC** algorithm is $1 - \text{sim}$, we have $\text{sim}(\mathbf{x}, \mathbf{s}) \geq 1 - d_c$. We have from [43]:

$$\text{sim}(\mathbf{x}, \mathbf{y}) \;\geq\; \text{sim}(\mathbf{x}, \mathbf{s}) \cdot \text{sim}(\mathbf{s}, \mathbf{y}) \;+\; m - 1,$$

$$\text{where } m = \min\Big\{\text{sim}(\mathbf{x}, \mathbf{s})^2,\ \text{sim}(\mathbf{s}, \mathbf{y})^2\Big\}.$$

Using $\text{sim}(\mathbf{x}, \mathbf{s}), \text{sim}(\mathbf{s}, \mathbf{y}) \geq 1 - d_c$ we get

$$\text{sim}(\mathbf{x}, \mathbf{y}) \;\geq\; (1 - d_c)^2 + (1 - d_c)^2 - 1 \;=\; 1 - 2\,d_c\,(2 - d_c).$$

Finally, converting this back to the distance $d(\mathbf{x}, \mathbf{y}) = 1 - \text{sim}(\mathbf{x}, \mathbf{y})$, we obtain:

$$d(\mathbf{x}, \mathbf{y}) \;\leq\; 2\,d_c\,(2 - d_c).$$

Therefore, the intra-cluster distance in the **DBDPC** algorithm is bounded by $2\,d_c\,(2 - d_c)$.

C. A comparison between DBDPC and other clustering algorithms

Comparison between DBDPC and DPC: We note that, aside from using densities, **DBDPC** is fundamentally different from DPC. Please refer to Appendix A for a detailed explanation of the **DPC** algorithm. The center identification process in **DBDPC** results in two main characteristics with formal proof detailed in Appendix B. First, the distance between each element and its cluster center is below d_c, which leads to inter-cluster distances being upper-bounded by $2d_c \times (2 - d_c)$. Additionally, the distance between cluster centers is lower-bounded by d_c. These guarantees do not hold for DPC, leading to two drawbacks. Since inter-cluster distances are not controlled, merging these vectors may result in merging highly dissimilar vectors, leading to information loss. Also, in high-density regions, the distance between cluster centers becomes too small, making DPC ineffective in addressing information redundancy.
A Qualitative comparison Figure 12 presents the clustering results for **DBDPC**, DPC, DBSCAN, and K-Means on a

Algorithm 4 Recursive Center Identification for DBDPC with Iterative Center Identification

```
Input: Cutoff distance $d_c \in \mathbb{R}^+$, set of vectors $\mathbf{U} = \{\mathbf{u}_i \in \mathbb{R}^{d_l}\}_{i=1}^n$, density values $\{\rho_i\}_{i=1}^n$, distance matrix $D = [d_{ij}]$, fallback threshold $T > 0$
Output: Cluster center indices $C_{\text{centers}}$
  Initialize cluster center set $C_{\text{centers}} = \emptyset$
  Set the density of each point :
      $\rho_i = \text{argsort}(\{-\rho_j\}_{j=1}^n)[i]$
  while $\mathbf{U} \neq \emptyset$ do
    Compute $\delta_i$ for all vectors $\mathbf{u}_i \in \mathbf{U}$:
        $\delta_i = \min_{\rho_j > \rho_i} d_{ij}$
    Select cluster candidates:
        $\mathbf{C}_{\text{new}} = \{\mathbf{u}_i \in \mathbf{U} \mid \delta_i > d_c\}$
    $C_{\text{centers}} \leftarrow C_{\text{centers}} \cup \mathbf{C}_{\text{new}}$
    Update remaining vectors:
        $\mathbf{U} \leftarrow \mathbf{U} \setminus \left(\mathbf{C}_{\text{new}} \cup \left\{\mathbf{u}_k \in \mathbf{U} \mid \begin{array}{c} \exists \mathbf{u}_i \in \mathbf{C}_{\text{new}} \\ \text{such that } d_{ik} \leq d_c \end{array}\right\}\right)$
    if $|\mathbf{C}_{\text{new}}| < T$ then
      Order remaining vectors $\mathbf{U}$ by decreasing $\rho_i$:
      $\mathbf{U} \leftarrow \text{Sort}(\mathbf{U}, \text{key} = \rho_i, \text{order} = \text{descending})$
      Call Iterative Center Identification:
      $C_{\text{centers}} \leftarrow \text{IterativeCenterIdentification}(C_{\text{centers}}, \mathbf{U}, d_c)$
      return $C_{\text{centers}}$
    end if
  end while
  return $C_{\text{centers}}$

  Function: Iterative Center Identification
  Inputs: Remaining vectors $\mathbf{U}$ (ordered by $\rho_i$), current cluster center set $C_{\text{centers}}$, cutoff distance $d_c$
  Outputs: Updated cluster center indices $C_{\text{centers}}$
  for all $\mathbf{u}_i \in \mathbf{U}$ do
    if $\min_{\mathbf{u}_s \in C_{\text{centers}}} d_{is} > d_c$ then
      $C_{\text{centers}} \leftarrow C_{\text{centers}} \cup \{\mathbf{u}_i\}$
    end if
  end for
  return $C_{\text{centers}}$
```

predefined set of two-dimensional points. The figure shows that only **DBDPC** and DBSCAN identify isolated points as distinct clusters, a crucial feature for visual token reduction, as these points contain unique and thus potentially valuable information. We note that, for DBSCAN, these isolated points may be identified as noise, depending on the chosen hyperparameters. Moreover, **DBDPC** partitions both the left and right groups of points into the same number of clusters, maintaining consistency despite the higher density on the left side. In contrast, DPC tends to form a greater number of clusters in high-density regions while creating large clusters in low-density areas, whereas DBSCAN follows the opposite pattern, producing large clusters in high-density regions. In the context of visual token reduction, merging points within these large clusters can result in information loss, leading to performance degradation and making DPC and DBSCAN less suitable than **DBDPC** for this task. We note that the results presented in Fig. 12 for DPC and DBSCAN may change when modifying the hyperparameters; however, the characteristics discussed above persist across different hyperparameter choices.

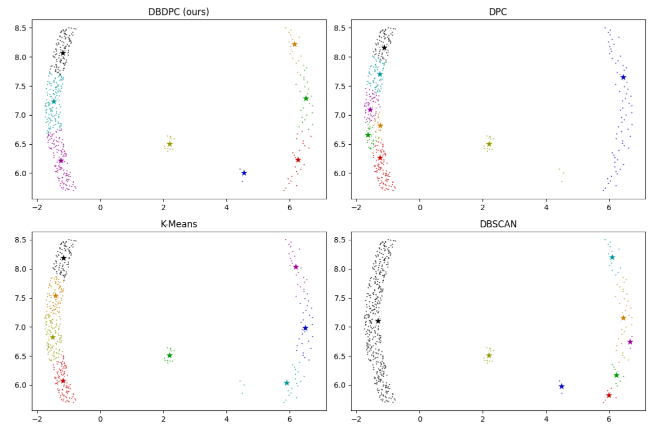


Figure 12. **An illustrative example of the difference in clustering characteristics between DBDPC and other clustering algorithms.** Two-dimensional points and the Euclidean distance were used for illustration purposes.

D. Efficient center identification in DBDPC

D.1. A recursive approach

To enhance the efficiency of the **DBDPC** algorithm, we introduce a recursive center identification method that reduces computational overhead while maintaining clustering accuracy. In the **DBDPC** algorithm, vectors are processed in descending order of their local densities ρ_i, and a vector $\mathbf{u}_i$ is selected as a cluster center if it is farther than the cutoff distance d_c from all previously selected centers. Implementing this as described in the algorithm requires sequentially iterating through all the vectors and checking distances to all previously selected centers, which does not fully leverage GPU parallelization capabilities. In the **DBDPC** algorithm, when two points have the same density, one is treated as if it has a higher density than the other, depending on the order of their processing. To replicate this behavior, we assign the density of each point to its rank as:

$$\rho_i = \text{rank}_i = \text{argsort}(\{-\rho_j\}_{j=1}^{n})[i]$$

Our accelerated method leverages the quantity δ_i, representing the minimum distance from vector $\mathbf{u}_i$ to any higher-density vector:

$$\delta_i = \min_{\rho_j > \rho_i} d_{ij} \tag{12}$$

If $\delta_i > d_c$, then $\mathbf{u}_i$ is selected as a cluster center because it is not within d_c of any higher-density vector, which are the only potential cluster centers that can be selected before d_{ij} in the **DBDPC** algorithm. In addition, any vector within d_c of a cluster center identified using δ_i has a lower density than that center, as cluster centers identified using δ_i are not within d_c of any higher-density vector. In the **DBDPC** algorithm, such a vector would not be chosen as a cluster center because it violates the distance condition relative to already selected centers. By identifying these vectors early, we can exclude them from further consideration as potential centers. We repeat this process recursively: after selecting cluster centers where $\delta_i > d_c$ and excluding vectors within d_c of these centers, we process the remaining vectors. This recursion continues until the number of newly discovered cluster centers becomes small (e.g., less than 10). At that point, we fall back to the **DBDPC** method, processing the remaining vectors iteratively to ensure all potential centers are considered. This recursive approach reduces the number of iterations in the main loop and enhances parallelization, particularly on GPUs, by minimizing sequential computation. By leveraging δ_i and incorporating an early exclusion mechanism, the recursive center identification method reduces computational time while ensuring the same clustering results as the **DBDPC** algorithm. The recursive approach decreases the number of iterations and enhances GPU parallelization by minimizing sequential computation, making the algorithm more efficient for large datasets. The recursive center identification method is presented in Algorithm 4. We note that in practice this recursive approach reduce the computational time of the **DBDPC** algorithm by around 3 times.

D.2. Proof of correctness of the recursive approach

To validate the correctness of the accelerated method, we demonstrate the following key points: selected centers are valid cluster centers, excluded vectors are not cluster centers and identifying remaining cluster centers is equivalent to identifying cluster centers on the reduced set. Proving these points suffices to establish correctness, as the remaining vectors after the recursive steps are treated the same as in the **DBDPC** algorithm.

Selected Centers Are Valid Cluster Centers In the **DBDPC** algorithm, for any vector $\mathbf{u}_i$, only vectors with higher densities are considered for selection as cluster centers before $\mathbf{u}_i$. If $\mathbf{u}_i$ is not within d_c of any higher-density vector (i.e., $\delta_i > d_c$) then the distance of $\mathbf{u}_i$ from any previously selected center cannot exceed the cutoff distance d_c. Consequently, $\mathbf{u}_i$ satisfies the condition for being a cluster center in the **DBDPC** algorithm, as it is farther than d_c from all centers processed earlier.

Excluded Vectors Are Not Cluster Centers Vectors within d_c of a cluster center identified using δ_i have lower densities than that center, as these centers are not within d_c to any higher density point. In the **DBDPC** algorithm, such vectors would not be selected as cluster centers because they are within d_c to an already selected center, violating the distance condition. Therefore, excluding these vectors early does not affect the selection of valid cluster centers.

Identifying Remaining Cluster Centers is Equivalent to Identifying Cluster Centers on the Reduced Set After selecting cluster centers where $\delta_i > d_c$ and excluding vectors within d_c of these centers, we focus on the reduced set of remaining vectors for further processing. The critical observation is that the previously selected cluster centers are not within d_c of any vector in the reduced set. This is ensured by the exclusion step, where all vectors within d_c of these centers have been removed. Consequently, when identifying new cluster centers within the reduced set, we do not need to consider distances to the previously selected centers, as they cannot influence the selection due to their distance. Moreover, the vectors that have been excluded are not potential cluster centers themselves. Meaning that they can not influence the center selection process. This means that any vector satisfying $\delta > d_c$ in the reduced set, is actually not within d_c to any higher density potential cluster center form the initial set, making it a cluster center.

E. On the choice of Positional IDs for clustering algorithms

In our work, we benchmark four clustering algorithms: agglomerative clustering [1], k-means [2], Density Peaks Clustering (DPC) [5], and DBSCAN [14]. For each algorithm, we use the key vectors for clustering, apply a cosine similarity-based distance (as in **DBDPC**), and evaluate two strategies: merging the hidden states within each cluster or selecting the cluster center as a representative token. We report the best-performing approach for each algorithm. Similar to **DBDPC**, we assign the position ID of the cluster center to the resulting vectors. However, apart from DPC, the other clustering algorithms do not explicitly provide a cluster center. For k-means and agglomerative clustering, we select the cluster center as the point closest to the average of all points in the cluster, using keys and cosine similarity. For DBSCAN, we experimented with choosing the point connected to the most other points within the cluster and found this approach to yield slightly better results, aligning better with the principles of DBSCAN. Thus, we adopted this strategy in our tests.

F. More about applying ToME to Visual Language Models

ToMe reduces the number of visual tokens at each layer of the transformer. For a given layer i, the process starts by splitting the tokens into two distinct sets, A and B. Each token in set A is matched with its most similar counterpart in set B, using cosine similarity based on key vectors to determine the closest pairs. The top r_i pairs with the highest similarity are then selected for merging. Connected components from the matched pairs are combined into single vectors, where hidden states are averaged. It is important to note that each connected component contains exactly one element from set B, and when applying ToME to Visual Language Models, this element's position ID is assigned to the merged token. In [6], the number of visual tokens was reduced by a fixed quantity ($r_i = r$). However, this fixed reduction scheme cannot achieve more than a 50% reduction unless no reduction is done at later layers when the number of tokens drops below r, which goes against the gradual reduction strategy proposed in ToMe. To enable higher reduction ratios, we adopt a linearly decreasing scheduler, where the reduction is higher in early layers and decreases in later layers. This approach achieves a smaller average number of visual tokens across the network while still reducing the token count at each layer, allowing us to reach high reduction ratios effectively.

G. Implementation details and hyperparameters for PACT

For all experiments on LLaVA-OneVision-7B, we set $d_n = 2$, $\alpha = 1.5$, and $L = 4$. While the optimal values of each parameter may vary depending on the dataset, we aim to evaluate the real-world effectiveness of our approach by using consistent values across all testing datasets. The results in Tab. 2 were obtained using $d_c = 0.21$ and $\lambda = 0.55$, while those in Tab. 1 were obtained using $d_c = 0.17$ and $\alpha = 0.7$. Additionally, to demonstrate the performance of our approach at different reduction ratios, we vary d_c and λ and report the results. The values of the fixed parameters d_n and α were chosen by performing a grid search on SeedBench [24], which is why we do not include SeedBench in the testing datasets. It is important to note that finding the optimal parameters for all testing datasets is not the focus of this study, as this would require extensive testing of different values for d_c, λ, L, α, and d_n on all test sets. Such an approach would not accurately reflect the real-world performance of our method. Instead, we chose to only vary d_c and λ to evaluate the effectiveness of our approach at different reduction ratios. When testing on SeedBench, we found that a pruning ratio higher than 60% harms performance. Therefore, we vary the pruning ratio between 10% and 60% and test across different values of d_c. When testing **PACT** on LLaVA-1.6-Mistral-7B, Qwen2-VL-7B-Instruct and InternVL2-8B. We use the same values of d_n and α as when testing on LLaVA-OneVision-7B. We note that these hyperparameters may not be optimal; however, as we aim to test the generalizability of our approach, we opt to use the same hyperparameters across models. Figure 13, Figure 14 and Figure 15 show the maximum distance between the keys at several layers of the language model for LLaVA-1.6-Mistral-7B, Qwen2-VL-7B-Instruct and InternVL2-8B. Following the same approach for LLaVA-OneVision-7B, we choose $L = 4$ for Qwen2-VL-7B-Instruct and $L = 7$ for InternVL2-8B. We note that the choice of the reduction layer for InternVL2-8B is not as evident as for LLaVA-OneVision-7B and Qwen2-VL-7B-Instruct, as the increase in maximum distance from one layer to the next is sometimes minimal, making it unclear which layer offers the best balance between accuracy and computational efficiency. However, since we do not aim to experimentally determine the optimal reduction layer, we end up choosing $L = 7$, as the maximum distance between keys is increased by an acceptable amount between the seventh and eighth layer. Following the same approach we use $L = 7$ for LLaVA-1.6-Mistral-7B.

H. More about test datasets and used metrics

For evaluating the different approaches, we use LMMs-Eval [26] and aim to follow the same dataset splits and metrics as used in [27]. We detail the used splits and metrics in Tab. 4. Some datasets require evaluation using a GPT model through the OPENAI API or other closed-source models. However, for many datasets the version of the closed-source model used in evaluating LLaVA-OneVision in [27] is no longer available. So we use the latest version of GPT-4 for our assessments at the time of publication (gpt-4o-2024-08-06). We also observed that when calling a closed-source model like GPT-4 via an API, the responses are not fully deterministic, even with a temperature set to zero, introducing some noise into the evaluation metrics. To reduce this noise, we exclude all these datasets when testing across different reduction ratios. On the other hand, for Tab. 1, we exclude MMVet, Vibe-Eval, VideoChatGPT, MM-LiveBench, and LLaVA-Wilder as they have high inference times, which would dominate the throughput calculation.

For certain datasets, such as DocVQA, InfoVQA, and TextVQA, we use the validation split contrary to [27]. This choice allows us to test various reduction ratios and approaches without requiring submission to the test server, which would be impractical for extensive testing. For datasets requiring a test set submission (EgoSchema and PerceptionTest), where either the validation set is typically

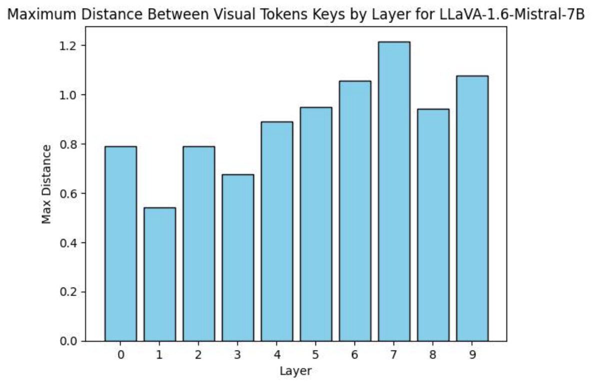


Figure 13. **Illustration of the maximum distance between the keys of visual tokens for the first 10 layers of LLaVA-1.6-Mistral-7B before the application of rotary embeddings.**

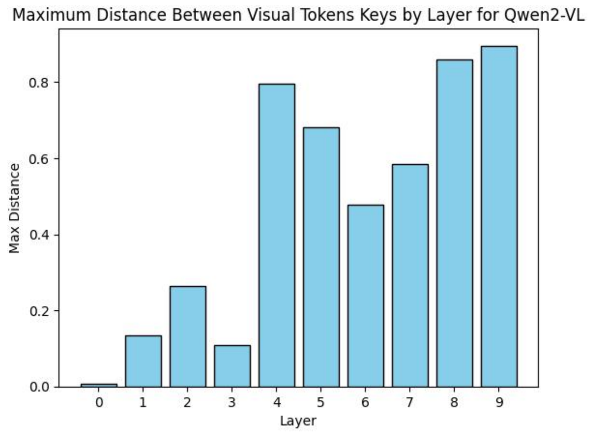


Figure 14. **Illustration of the maximum distance between the keys of visual tokens for the first 10 layers of Qwen2-VL-7B-Instruct before the application of rotary embeddings.**

not used for evaluation or does not exist, we report the submission-based metrics evaluated directly on the test set. As explained above, for some datasets our evaluation setup differs from the one used for evaluating LLaVA-OneVision in [27], which may result in variations in the reported results for this model on certain datasets. This is primarily due to the use of validation splits for DocVQA, InfoVQA, and TextVQA, as well as the reliance on GPT-based metrics for some datasets (a common practice for these benchmarks, making alternative evaluation difficult). Nevertheless, our comparisons remain fair, as the same evaluation procedure is consistently applied across all approaches and reduction ratios.

We note that when using reduction methods, results may include slight variations due to edge cases where distances or importance metrics for different vectors are equal. That's why we report results based on the average of three different runs for each dataset.

Notably, when testing on Qwen2-VL-7B-Instruct without reduction, some datasets encountered GPU out-of-memory errors (MLVU, VideoMME, and ActivityNet Perception) which we excluded from the test set. Additionally, results on ScienceQA were quite low when tested without reduction (0.132), leading to its exclusion from testing as well. We note that, as we use LMM-Eval [26] for evaluation, results differ for some datasets from the officially reported results, as prompts are sometimes not formatted in the same manner. This observation also applies to InternVL2-8B.

I. Additional numerical results

Table 8 and Tab. 9 show a comparison of **DBDPC** and various clustering algorithms for a reduction ratio of approximately 60% on LLaVA-OneVision-7B across multiple datasets. The results demonstrate that **DBDPC** outperforms other clustering algorithms in visual token reduction for the majority of the datasets. Additionally, the tables show that the clustering process for **DBDPC** is significantly faster than that of other clustering algorithms. Table 10 presents a comparison of **EUTI**-based visual token pruning and FastV for a reduction ratio of approximately 60% on LLaVA-OneVision-7B across various datasets. The results indicate that **EUTI** outperforms FastV on most datasets while also being more computationally efficient. Table 15 shows that using keys for distance calculations in DBDPC outperforms hidden states across the majority of the test datasets. Also, we present a comparison between **PACT** and other visual reduction techniques for InternVL2-8B, and LLaVA-1.6-Mistral-7B across different datasets in Tab. 6, and Tab. 7.

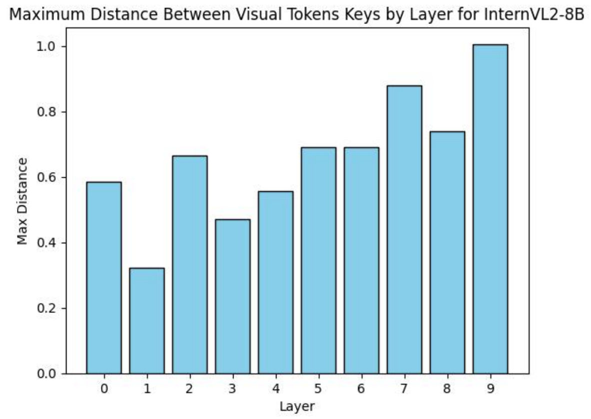


Figure 15. **Illustration of the maximum distance between the keys of visual tokens for the first 10 layers of InternVL2-8B before the application of rotary embeddings.**

J. Ablation study : Additional numerical results

Table 11 shows a comparison between **PACT**, **DBDPC**, and **EUTI** for a reduction ratio of approximately 70%, applied on LLaVA-OneVision-7B. The results demonstrate that **PACT**, which combines both clustering and pruning, outperforms the other two methods that are either clustering-based or pruning-based across various datasets. More importantly, **DBDPC** and **EUTI** exhibit a significant drop in performance on some of the datasets, which is not the case for **PACT**. We note that numerical results for the ablation studies conducted on **DBDPC**, **EUTI**, and **PACT** can be found in Tab. 12, Tab. 13 and Tab. 14.

Table 4. **Dataset Splits, Subsets, and Evaluation Metrics Used in Our Experiments.** Default indicates the use of the standard test split or cases where only one split/subset is available. The evaluation metrics employed are those commonly used for the respective datasets and generally the ones proposed in the official papers. For GPT-based scores (or any model-based scores), this means that a GPT model was used during evaluation, typically to extract answers from the generated output text, which are then matched with the ground truth to calculate accuracy using exact matches. When accuracy is reported, it generally implies that only an exact match is considered a correct answer.

Dataset	**Split**	**Subset**	**Evaluation Metric**
VideoMME	Default	No subtitles	Accuracy
MME	Default	Default	MME Perception Score
DocVQA	Validation	Default	ANLS
MLVU	Default	Default	Accuracy
LLaVA-Interleave	Default	Out-domain	Accuracy
ChartQA	Validation	Default	Relaxed Accuracy
MMBench	Validation	English	GPT-based Score
MuirBench	Default	Default	Accuracy
ScienceQA	Default	Vision only	Accuracy
MMMU	Validation	Default	Accuracy
AI2D	Default	Default	Accuracy
InfographicVQA	Validation	Default	ANLS
MMStar	Default	Default	Accuracy
ActivityNetQA	Default	Default	GPT-based Score
MM-LiveBench	Default	2406	GPT-based Score
LLaVA-Wilder	Default	Small	GPT-based Score
MathVerse	Default	Vision mini	GPT-based Score
MathVista	Default	Testmini	GPT-based Score
MMVet	Default	Default	GPT-based Score
Vibe-Eval	Default	Default	REKA-based Score
VideoChatGPT	Default	Default	GPT-based Score
EgoSchema	Default	Default	Submission
PerceptionTest	Default	Multiple Choice QA	Submission
TextVQA	Validation	Default	Official metric

Table 5. **Performance of PACT on LLaVA-OneVision-7B using $d_c = 0.17$ and $\alpha = 0.7$.**

Dataset	**PACT (Ours)**			
	Metric	**Red. Ratio**	**Proc. Time**	**Algo. Time**
VideoMME	57.7	69.2%	0.321	0.021
MME	1571.0	72.1%	0.226	0.017
DocVQA	85.4	71.1%	0.467	0.026
MLVU	64.8	69.2%	0.322	0.022
LLaVA-Interleave	62.2	72.2%	0.133	0.010
ChartQA	77.3	71.4%	0.309	0.019
MMBench	79.9	72.0%	0.134	0.010
MuirBench	42.4	70.9%	0.175	0.013
ScienceQA	93.5	72.0%	0.130	0.010
MMMU	48.8	72.6%	0.103	0.007
AI2D	81.2	72.5%	0.173	0.013
InfographicVQA	61.5	70.0%	0.403	0.023
MMStar	59.5	72.3%	0.147	0.011
ActivityNetQA	55.1	70.0%	0.409	0.029
MathVerse	17.1	76.0%	0.350	0.021
MathVista	62.1	73.0%	0.260	0.015
EgoSchema	60.0	69.1%	0.320	0.021
PerceptionTest	52.3	70.0%	0.301	0.023
TextVQA	75.5	69.2%	0.320	0.023

Table 6. **Comparison of PACT with FastV, VTW, and ToME applied on InternVL2-8B on Various Datasets.**

Dataset	**No Reduction**		**PACT (Ours)**			**FastV**		**VTW**		**ToME**	
	Metric	**Proc. Time**	**Metric**	**Red. Ratio**	**Proc. Time**	**Metric**	**Proc. Time**	**Metric**	**Proc. Time**	**Metric**	**Proc. Time**
VideoMME	52.2	0.247	**51.1**	68.4%	0.151	**51.1**	0.155	51.0	0.142	50.2	0.190
MME	1621.0	0.171	1591.9	69.9%	0.121	1588.7	0.118	**1627.0**	0.111	1533.3	0.155
MLVU	50.6	0.439	**49.7**	68.8%	0.326	48.8	0.325	49.5	0.333	29.3	0.343
LLaVA-Interleave	40.0	0.390	39.0	71.2%	0.265	**39.7**	0.263	39.6	0.230	36.7	0.316
MMBench	81.9	0.161	**80.4**	70.4%	0.118	80.2	0.116	80.2	0.109	70.8	0.165
MuirBench	35.7	0.432	34.4	70.3%	0.249	**35.6**	0.258	33.7	0.210	32.7	0.296
ScienceQA	97.1	0.165	**97.1**	70.8%	0.118	95.8	0.116	95.7	0.109	89.9	0.151
MMMU	48.5	0.167	**48.0**	70.6%	0.126	47.7	0.126	47.8	0.119	47.5	0.156
AI2D	82.5	0.146	**81.4**	70.7%	0.112	78.5	0.110	79.6	0.105	74.4	0.142
MMStar	59.0	0.179	**56.7**	70.4%	0.186	54.2	0.184	53.4	0.352	55.1	0.156
PerceptionTest	57.7	0.300	**56.8**	66.0%	0.203	56.2	0.213	34.1	0.192	55.2	0.228
EgoSchema	54.0	0.240	**53.7**	67.0%	0.155	53.1	0.163	32.2	0.146	52.9	0.172
ActivityNet	51.7	0.240	**51.3**	66.0%	0.153	51.0	0.161	30.8	0.143	50.4	0.171
MM-LiveBench	68.0	3.075	**67.3**	68.0%	2.140	67.0	2.247	40.4	2.003	66.6	2.354

Table 7. **Comparison of PACT with FastV, Prumerge, and Hired applied on LLaVA-1.6-Mistral-7B across multiple datasets.**

Dataset	**No Reduction**		**PACT (Ours)**			**FastV**		**Prumerge**		**Hired**	
	Metric	**Proc. Time**	**Metric**	**Red. Ratio**	**Proc. Time**	**Metric**	**Proc. Time**	**Metric**	**Proc. Time**	**Metric**	**Proc. Time**
MME	1500.0	0.237	**1507.1**	70.3%	0.159	1503.9	0.158	1485.4	0.166	1497.0	0.168
DocVQA	70.0	0.363	**67.1**	67.1%	0.284	64.5	0.281	48.8	0.293	65.8	0.295
ChartQA	52.9	0.332	**49.3**	70.1%	0.259	48.9	0.261	36.0	0.264	46.1	0.266
MMBench	68.2	0.226	**68.0**	71.9%	0.155	67.9	0.154	66.2	0.160	67.6	0.164
ScienceQA	73.0	0.197	72.7	71.5%	0.144	**73.2**	0.145	71.7	0.148	72.9	0.149
MMMU	34.2	0.239	**34.9**	71.5%	0.171	34.7	0.169	33.9	0.180	33.9	0.180
AI2D	67.5	0.233	**67.5**	70.9%	0.160	67.0	0.158	64.5	0.165	65.9	0.166
InfographicVQA	36.9	0.294	**35.6**	66.2%	0.226	33.4	0.229	31.9	0.236	31.6	0.236
MMStar	36.2	0.375	**36.7**	71.9%	0.350	36.6	0.400	35.1	0.345	35.9	0.345

Table 8. **Comparison of DBDPC and Agglomerative Clustering Methods for a Reduction Ratio of approximately 60% on LLaVA-OneVision-7B.**

Dataset	DBDPC (ours)			Agg. (Single Linkage)			Agg. (Average Linkage)			Agg. (Complete Linkage)		
	Metric	Proc. Time	Algo. Time	Metric	Proc. Time	Algo. Time	Metric	Proc. Time	Algo. Time	Metric	Proc. Time	Algo. Time
VideoMME	57.4	0.389	0.040	57.6	1.504	1.148	57.0	1.657	1.316	57.9	1.690	1.350
MME	1563.8	0.255	0.028	1554.1	0.994	0.738	1559.2	1.123	0.868	1563.0	1.151	0.897
DocVQA	84.7	0.530	0.044	83.6	1.899	1.379	84.4	2.185	1.662	84.3	2.308	1.777
MLVU	64.2	0.384	0.039	64.0	1.574	1.229	65.2	1.675	1.329	64.8	1.700	1.355
LLaVA-Interleave	62.1	0.151	0.016	62.0	0.425	0.277	61.5	0.446	0.298	61.4	0.446	0.298
ChartQA	76.0	0.366	0.031	74.5	1.151	0.798	75.8	1.253	0.910	75.8	1.277	0.930
MMBench	80.1	0.151	0.016	79.5	0.427	0.277	79.7	0.437	0.291	79.8	0.449	0.299
MuirBench	43.2	0.215	0.023	41.4	0.667	0.474	42.0	0.727	0.534	42.0	0.738	0.544
ScienceQA	94.7	0.147	0.015	94.8	0.394	0.250	94.7	0.416	0.271	94.7	0.413	0.269
MMMU	48.3	0.110	0.009	48.4	0.218	0.110	49.3	0.232	0.121	48.2	0.225	0.117
AI2D	80.7	0.202	0.022	80.8	0.667	0.472	80.6	0.748	0.551	80.1	0.753	0.557
InfographicVQA	61.6	0.528	0.046	57.1	1.608	1.181	59.8	1.818	1.394	59.8	1.870	1.436
MMStar	60.5	0.167	0.018	60.2	0.507	0.344	59.8	0.556	0.390	60.5	0.560	0.395

Table 9. **Comparison of DBDPC, DBSCAN, DPC, and KMeans Clustering Methods for a Reduction Ratio of approximately 60% on LLaVA-OneVision-7B.**

Dataset	DBDPC (ours)			DBSCAN			DPC			KMeans		
	Metric	Proc. Time	Algo. Time	Metric	Proc. Time	Algo. Time	Metric	Proc. Time	Algo. Time	Metric	Proc. Time	Algo. Time
VideoMME	57.4	0.389	0.040	57.4	0.394	0.046	56.9	0.729	0.392	57.3	1.725	1.383
MME	1563.8	0.255	0.028	1560.3	0.274	0.036	1549.9	0.637	0.380	1549.9	1.254	0.999
DocVQA	84.7	0.530	0.044	84.2	0.533	0.044	83.0	0.950	0.442	79.6	2.059	1.544
MLVU	64.2	0.384	0.039	64.2	0.391	0.048	64.2	0.727	0.382	64.6	1.725	1.377
LLaVA-Interleave	62.1	0.151	0.016	60.4	0.159	0.026	63.9	0.258	0.121	62.3	0.711	0.566
ChartQA	76.0	0.366	0.031	75.2	0.369	0.034	75.2	0.758	0.415	74.2	1.399	1.059
MMBench	80.1	0.151	0.016	78.1	0.153	0.020	79.5	0.326	0.179	79.9	0.702	0.552
MuirBench	43.2	0.215	0.023	42.4	0.219	0.028	42.0	0.466	0.273	42.9	0.955	0.763
ScienceQA	94.7	0.147	0.015	91.2	0.150	0.024	94.3	0.251	0.117	93.4	0.661	0.518
MMMU	48.3	0.110	0.009	47.8	0.130	0.030	48.3	0.187	0.078	48.2	0.500	0.391
AI2D	80.7	0.202	0.022	79.2	0.202	0.022	80.3	0.455	0.264	81.1	1.062	0.860
InfographicVQA	61.6	0.528	0.046	54.0	0.531	0.052	56.6	0.975	0.547	57.8	1.780	1.357
MMStar	60.5	0.167	0.018	56.6	0.179	0.028	60.6	0.376	0.213	60.2	0.828	0.661

Table 10. **Comparison of EUTI-based visual tokens pruning and FastV for a Reduction Ratio of approximately 60% on LLaVA-OneVision-7B.**

Dataset	EUTI (Ours)			FastV		
	Metric	Proc. Time	Algo. Time	Metric	Proc. Time	Algo. Time
VideoMME	58.4	0.351	0.005	57.6	0.381	0.040
MME	1560.0	0.256	0.004	1570.7	0.283	0.025
DocVQA	86.5	0.521	0.005	85.3	0.559	0.032
MLVU	64.3	0.355	0.004	63.1	0.391	0.040
LLaVA-Interleave	58.9	0.140	0.003	59.7	0.152	0.007
ChartQA	78.6	0.344	0.004	78.0	0.363	0.016
MMBench	80.2	0.142	0.003	79.2	0.151	0.005
MuirBench	40.0	0.191	0.003	40.8	0.204	0.009
ScienceQA	93.6	0.137	0.003	92.3	0.149	0.006
MMMU	48.8	0.101	0.002	47.3	0.110	0.003
AI2D	81.1	0.191	0.003	80.3	0.202	0.009
InfographicVQA	63.0	0.425	0.005	60.3	0.473	0.040
MMStar	59.6	0.159	0.003	59.6	0.170	0.007

Table 11. **Comparison of PACT with Standalone Methods: EUTI-based Visual Token Pruning and DBDPC Clustering Algorithm for a Reduction Ratio of approximately 70%, applied on LLaVA-OneVision-7B.**

Dataset	**PACT**			**DBDPC**			**EUTI**		
	Metric	**Proc. Time**	**Algo. Time**	**Metric**	**Proc. Time**	**Algo. Time**	**Metric**	**Proc. Time**	**Algo. Time**
VideoMME	57.5	0.321	0.021	57.3	0.342	0.040	58.4	0.305	0.005
MME	1558.7	0.226	0.017	1543.7	0.243	0.028	1595.9	0.213	0.004
DocVQA	84.3	0.467	0.026	82.5	0.500	0.044	85.3	0.456	0.005
MLVU	64.6	0.322	0.022	63.9	0.358	0.039	64.4	0.291	0.004
LLaVA-Interleave	63.9	0.133	0.010	62.6	0.149	0.016	57.1	0.127	0.003
ChartQA	77.2	0.311	0.019	75.1	0.333	0.031	78.2	0.292	0.004
MMBench	80.2	0.134	0.010	79.7	0.147	0.016	79.6	0.128	0.003
MuirBench	42.8	0.175	0.013	43.2	0.211	0.023	39.9	0.164	0.003
ScienceQA	93.6	0.130	0.010	93.8	0.142	0.015	92.2	0.123	0.003
MMMU	48.9	0.103	0.007	47.2	0.109	0.009	48.9	0.096	0.002
AI2D	80.6	0.173	0.013	80.5	0.191	0.022	79.9	0.164	0.003
InfographicVQA	61.9	0.403	0.023	58.8	0.465	0.046	60.4	0.360	0.005
MMStar	59.5	0.147	0.011	59.5	0.163	0.018	59.2	0.140	0.003

Table 12. **Ablation Studies on DBDPC-based visual token reduction for a Reduction Ratio of approximately 60% on LLaVA-OneVision-7B.** We report only the metrics, as processing time is similar across different approaches. When ablating the Center Position-IDs assignment, we reorder the hidden states based on the mean of the Position-IDs of the elements in each cluster and then assign position IDs sequentially.

	DBDPC	**w/o Center Position-IDs assignment**	**w/o Proportional Attention**	**w/o Merging**
VideoMME	57.4	58.0	57.9	57.5
MME	1563.8	1539.3	1523.8	1476.9
DocVQA	84.7	28.2	84.2	83.1
MLVU	64.2	61.2	63.9	63.5
LLaVA-Interleave	62.1	69.6	63.2	63.6
ChartQA	76.0	24.8	76.0	74.4
MMBench	80.1	76.1	80.1	79.6
MuirBench	43.2	26.5	43.2	44.0
ScienceQA	94.7	67.4	94.2	93.6
MMMU	48.3	34.5	47.6	48.2
AI2D	80.7	43.0	80.4	79.9
InfographicVQA	61.6	17.8	59.8	58.7
MMStar	60.5	58.9	59.6	59.1

Table 13. **Ablation Studies on the EUTI-based Visual Token Pruning for a Reduction Ratio of approximately 70%, applied on LLaVA-OneVision-7B.** We report only the metrics, as processing time is similar across different approaches.

Dataset	EUTI	EUTI w/o Norm	Norm (EUTI w/o Global Query)
VideoMME	58.4	57.6	56.6
MME	1595.9	1573.4	1576.5
DocVQA	85.3	85.1	79.7
MLVU	64.3	63.0	63.1
LLaVA-Interleave	57.1	57.9	52.9
ChartQA	78.2	76.4	76.7
MMBench	79.6	79.4	79.4
MuirBench	40.0	40.5	39.6
ScienceQA	92.2	91.8	93.5
MMMU	48.9	49.3	49.2
AI2D	79.9	79.9	79.7
InfographicVQA	60.4	60.1	49.3
MMStar	59.2	57.4	59.2

Table 14. **Ablation Study on Pruned Tokens Recovery for a Reduction Ratio of approximately 70%.** We remove the token recovery step, which is equivalent to Setting α to Zero. We report only the metrics, as processing time is similar across both approaches.

Dataset	PACT	PACT w/o Pruned-Token Recovery
VideoMME	57.6	57.4
MME	1556.7	1576.3
DocVQA	84.3	84.3
MLVU	64.6	64.2
LLaVA-Interleave	63.9	59.6
ChartQA	76.4	76.4
MMBench	79.9	79.8
MuirBench	42.8	42.2
ScienceQA	93.3	93.6
MMMU	48.5	48.5
AI2D	80.6	80.6
InfographicVQA	61.9	61.3
MMStar	75.1	74.9

Table 15. **Ablation Study on Keys Utilization in DBDPC for a Reduction Ratio of approximately 60%.** Metrics are reported, as processing time is similar across both configurations.

Dataset	DBDPC	DBDPC w/o Keys
VideoMME	57.40	57.22
MME	1563.80	1526.18
DocVQA	84.70	80.50
MLVU	64.20	64.60
LLaVA-Interleave	62.10	60.80
ChartQA	76.00	68.80
MMBench	80.10	79.21
MuirBench	43.20	41.40
ScienceQA	94.70	91.90
MMMU	48.30	47.90
AI2D	80.70	79.10
InfographicVQA	61.6	56.70
MMStar	60.50	58.40