Explainability Enhanced Object Detection Transformer with Feature Disentanglement

Wenlong Yu, Ruonan Liu, *Member, IEEE,* Dongyue Chen, Qinghua Hu, *Senior Member, IEEE*

***Abstract*—Explainability is a pivotal factor in determining whether a deep learning model can be authorized in critical applications. To enhance the explainability of models of end-to-end object DEtection with TRansformer (DETR), we introduce a disentanglement method that constrains the feature learning process, following a divide-and-conquer decoupling paradigm, similar to how people understand complex real-world problems. We first demonstrate the entangled property of the features between the extractor and detector and find that the regression function is a key factor contributing to the deterioration of disentangled feature activation. These highly entangled features always activate the local characteristics, making it difficult to cover the semantic information of an object, which also reduces the interpretability of single-backbone object detection models. Thus, an Explainability Enhanced object detection Transformer with feature Disentanglement (DETD) model is proposed, in which the Tensor Singular Value Decomposition (T-SVD) is used to produce feature bases and the Batch averaged Feature Spectral Penalization (BFSP) loss is introduced to constrain the disentanglement of the feature and balance the semantic activation. The proposed method is applied across three prominent backbones, two DETR variants, and a CNN based model. By combining two optimization techniques, extensive experiments on two datasets consistently demonstrate that the DETD model outperforms the counterpart in terms of object detection performance and feature disentanglement. The Grad-CAM visualizations demonstrate the enhancement of feature learning explainability in the disentanglement view.**

***Index Terms*—Deep learning, Explainability, Feature disentanglement, Hybrid Transformer model, Object detection.**

I. Introduction

OBJECT detection is one of the most crucial tasks of AI systems [1]. Over the years, various object detection models based on deep learning have been developed thanks to the great representation ability the Deep Neural Network (DNN) has. Transformer based models are the key members of them. Transformer is expanded into the object detection area with state-of-the-art (SOTA) performance after its birth in the Natural Language Processing (NLP) area [2]. Some excellent object detection models have been produced, such as DETR [3] and its derivatives [4], [5]. DETR views object detection as a direct set prediction problem without antique hand-designed components. It achieves higher performance attributed to the synergy of the local feature extraction ability of Convolutional Neural Network (CNN) and the global feature integration power of Transformer. Unfortunately, the inherent lack of explainability in deep learning causes these models to fail in critical areas, such as autonomous driving, medical diagnosis, and justice [6]–[8]. Naturally, it is necessary to explore the operational mechanism of DNNs so that people can understand and trust these high-performance models.

Explainability is a critical component of trustworthy AI [9]. Various methods have been proposed to make the model more explainable. However, most of them cannot reach an equilibrium between explainability and performance [10]. What we believe is that an effective explainability-enhancing approach should run on the basis of revealing the mechanism of the model, just as scientists analyze a complex task, such as solving a physical equation. Divide-and-conquer [11], [12] is a philosophical methodology to address complex scientific issues including control theory [13], mathematics [14], and law [15]. Specifically, the process involves learning and disentangling, understanding and controlling, and then entangling and applying. This forms a pipeline within this methodology. The complex subject is disentangled into some orthogonal (or even approximate) directions, and each direction can be seen as a simple but specific description of the subject [16], [17]. After the meticulous analysis of each direction, humans can develop a nuanced understanding of this complex subject and finally entangle all directions into a more transparent and explainable model to carry out downstream tasks. Typically, such an interpretable approach does not damage the models but even benefits them because subsequential optimization and control strategies can be applied to these decomposable models. Beyond that, the disentanglement or orthogonalization makes the feature space more parsimonious and structured, which spontaneously boosts the explainability and discriminating power of the model [18].

The deep learning models can also be seen as a kind of complex mapping function. Classical DETR integrates CNN and Transformer, and the interaction between both contributes to DETR's climactic performance. CNN can be viewed as a feature extractor that maps RGB images into a high-dimensional semantical feature space. The Transformer then computes the attention value among all feature tokens to find the object vectors. DETR's backbone can be considered a learning and disentangling module, while the Transformer detector can be viewed as a token entangling mixer.

Literature has shown that the particular disentangled directions of the feature space or parameter space have their own semantic meaning related to the real world, such as

This work was supported in part by the National Natural Scientific Foundation of China (NSFC) under Grant 61925602 and U23B2049. *(Corresponding author: Ruonan Liu).*

Wenlong Yu, Ruonan Liu, Dongyue Chen, and Qinghua Hu are with the College of Intelligence and Computing, Tianjin University, Tianjin 300350, China, and also with the Tianjin Key Lab of Machine Learning, Tianjin University, Tianjin 300350, China (e-mail: wlong_yu@126.com, ruonan.liu@tju.edu.cn, dyue_chen@163.com, huqinghua@tju.edu.cn).

generative models [19]–[21] and discriminative classification models [22], [23]. This indicates that we can enhance the explainability by analyzing these orthogonal directions. In the field of object detection, the design of multiple backbones that directly decouple different semantic information, such as separating classification and regression or dividing features into invariant and specific categories, has led to improved performance and interpretability [21], [24], [25]. These pioneers strengthen our confidence in enhancing the explainability of the object detection Transformer models by introducing the feature disentanglement.

DETR's architecture enables better feature learning. The semantical features extracted by CNN are some specific descriptions of the whole image. However, as for single-backbone object detection models (DETR is a typical example of such models), the single feature extractor needs to simultaneously handle the object classification task and the coordinates regression task. The learned high-level metaphysical semantics are usually indistinguishable as they are entangled with each other. Moreover, learned features often exhibit a tendency to overfit the dataset and become entangled with spurious associations [26]. All of these factors impede the interpretability analysis of object detection models. Therefore, conducting disentanglement analysis on single-backbone object detection models and proposing decoupling methods undoubtedly contributes to enhancing model interpretability.

To formally investigate the operational mechanism of object detection models, this study defines the Batch averaged Feature Inner Product (BFIP) and Batch averaged Feature Cosine Similarity (BFCS) to measure the orthogonality which represents the degree of disentanglement among features. We conduct a detailed analysis of the classification and regression mechanisms from the feature disentanglement perspective, making the operation of the object detection model more transparent. Based on a comparison of two similarity measurement standards, we find that regression leads to feature imbalance, manifesting as a significant disparity in feature activations. Furthermore, BFSP loss is proposed to enhance the feature disentanglement and balance. After that, explainability-enhanced Hybrid Transformer models are built by introducing Tensor Singular Value Decomposition (T-SVD) [27], [28] to the DETR framework. Optimization strategies additionally boost the performance of object detection. Six variants of the model are constructed by associating with classical architectures, including CNN, Transformer, Multi-Layer Perceptron (MLP), Deformable DETR, DINO, and CNN based detection framework. In summary, our contributions to this work are:

- We reveal the difference between classification and regression from the novel view of feature disentanglement. Regression function is found as a factor leading to feature imbalance. This helps explain the mechanism of the object detection.
- BFSP loss is proposed to make features between the extractor and detector more decoupled and balanced. Kinds of disentangling rule-inspired hybrid Transformer models are built based on T-SVD and BFSP.
- Optimization strategies based on the reconstruction of decoupled feature bases and progressive training methods are proposed to upgrade the object detection performance.

Furthermore, extensive experiments on two benchmark datasets show that DETDs outperform their baselines in terms of object detection performance, generalization, and feature disentanglement. Visualizations further validate the enhancing explainability.

II. Related Work

The weak explainability of deep learning models severely limits the applications in some rigorous areas, inspiring many meaningful pieces of research, which can be called XAI [29], [30]. Research on XAI can mainly be divided into active and passive explanation approaches [7], [31]. The difference lies in that the active methods want to directly build transparent models or other forms of models that can output the decision results and the decision-making reasons simultaneously. In contrast, the passive techniques focus on externally giving reasons why an already modeled DNN outputs specific decisions. By introducing causality [26], rule [32], bayesian theory [33], or other deductive evolution methods [34], [35], the results of the active interpretable models [10] can be preliminarily concluded by those mathematical or logical methods. Beyond that, some active methods explain the DNN by adding constrained knowledge [23], prior loss functions [36] or even setting up additional interpretation neural network branches [37]. However, each approach is limited by its own shortcomings, such as the insufficient quantity of knowledge and performance degradation.

In contrast, the passive explanation methods mainly want to decipher the inner mechanism of neural networks. They don't change the original black box substantive characteristics. Zhang et al. [38] and Chen et al. [39] utilize the quantification of information to measure explainability. Feature based visualization methods, such as Saliency [40] and Grad-CAM [41], have been widely used for analyzing the regions that have the most significant impact on the model results. Zhang et al. [42] try to use the Shapley Value of Game Theory to unify these post-hoc passive explanation methods. Moreover, it is worth mentioning that both vector directions in the parameter space [19] and the disentangled features or units extracted by CNN classification task [22], [43] can represent the semantical directions relevant to the real world, which encourages us to study how to enhance the explainability of hybrid object detection Transformer models by disentangling features.

Transformer based models are increasingly dominating computer vision tasks because of their strong ability to extract global information by contextualizing every token [44], [45]. However, there are still not many contributions to explain the reasons for efficiency thoroughly. To some extent, existing explainable methods mainly focus on finding the tokens that significantly influence the model results. LRP [46] and partial LRP [47] compute the layer-wise relevance score propagated from the output to input units to find the essential tokens. Rollout method [48] multiplies all the attention scores recursively. But these methods naively assume that attention can be combined linearly. Chefer et al. [49] combine the attention relevance score with Grad-Cam to visualize the class-specific

vital regions. The explanation of vision Transformer models [45] can directly reshape the class token to the original input image to find the important pixels because the class token mixes up all the classification information. As for DETR [3], we can output the attention related to every decoder query to visualize the output classification and regression results.

Nevertheless, all of them are simple visual explanations. They ignore another explanatory dimension: the feature disentanglement property of the hybrid Transformer model. However, it is difficult to directly match the semantics with the features due to the feature entanglement property in the object detection task. Therefore, it is necessary to change the hybrid object detection Transformer model to be decomposable. This study actively enhances the explainability of hybrid object detection Transformer models from the view of feature disentanglement without performance degradation.

III. Feature Disentanglement Analysis

CNN, MLP, and Transformer are three mainstream architectures in computer vision tasks [50]. All of them can be used as the feature extractor with different behaviors due to the divergent mechanism. In this paper, our primary motivation is that the disentangled feature bases can help explain the principle of the object detection Transformer. In this section, we take CNN based Transformer as an example to analyze the feature disentangling property.

A. End-to-End Object Detection with Transformer

Traditional object detection models, whether one-stage (e.g., YOLO [51]) or two-stage (e.g., Faster R-CNN [52]), often rely on hand-crafted design choices, leading to performance dependencies on empirical settings [53]. In contrast, DETR, as a single-backbone model, achieves remarkable performance [3], [54] without relying on prior knowledge.

CNN and Transformer are fundamental architectures of DETR. Utilizing the receptive field implemented by convolution, CNN can be seen as a local feature extractor. The Transformer is a feature mixer because of the long-term attention mechanism. Specifically, given a batch of images $\mathbf{x} \in \mathbb{R}^{B\times 3\times H\times W}$, CNN extractor $\mathbf{E} = e(\mathrm{x})$ outputs a set of features $\mathcal{F} = \left\{f_1^l, f_2^l, \cdots, f_{C^l}^l\right\}$, where C^l is the channel number in layer l, feature f_i^l is a specific description of the input images. After these non-linear spatial transformation operations, we get a mapping function from the RGB image space to a high-dimensional semantic space spanned by all feature matrices [19], [55]. The feature set $\mathcal{F}$ is sent to the Transformer encoder as serialized visual tokens, which are embedded vectors similar to word tokens in NLP [2]. Traditional one-stage object detection CNN models access the downstream output networks directly [51].

Transformer modules then construct three description matrices $\mathbf{Q}, \mathbf{K}$, and $\mathbf{V}$, representing Query, Key, and Value, respectively, by applying linear layers to the input tokens. Subsequently, the self-attention in encoder and cross-attention in decoder are calculated as $\mathbf{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathrm{softmax}\left(\frac{\mathbf{Q}\cdot\mathbf{K}^\top}{\sqrt{d_k}}\right)\cdot\mathbf{V}$ with position embedding added to represent the original positions. After the multi-layer encoders and decoders, the outputs of the Transformer are sent to double-head Fully Connected layers (FC) $m(\cdot)$ to get an object box set and a classification set as shown in Fig.1. The model can simultaneously handle regression and classification problems by jointly training CNN and Transformer. Formally, the model can be abstracted as

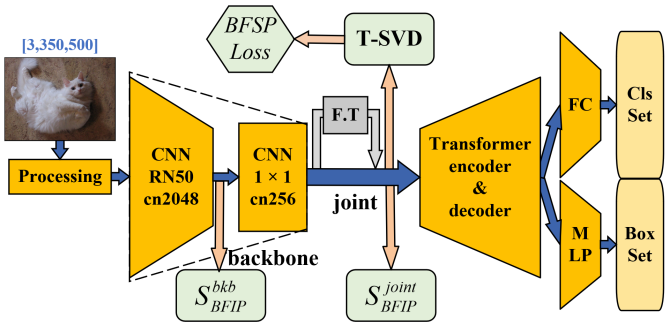


Fig. 1: Illustration of DETR and the implemented computations. CNN module is composed of a ResNet-50 [56] backbone and a 1×1 convolution layer. We extract features from the position after backbone for computing $\mathcal{S}_{BFIP}^{bkb}$ and joint before Transformer for $\mathcal{S}_{BFIP}^{joint}$. The alternative module in the grey box(F.T) transforms the input format from original point-wise to feature-wise.

$$\mathrm{Y} = m(t(f(l(e(\mathbf{x}))))), \tag{1}$$

where $l(\cdot)$ is the mapping function from feature matrices $\mathcal{F}$ to visual tokens for matching the input need of Transformer, $t(\cdot)$ is the Transformer model applied in DETR, and $f(\cdot)$ represents the function that this paper propose (no operation for DETR), respectively. By concatenating the flattened feature row vectors $\mathbf{v}_i^l \in \mathbb{R}^{1\times d^l}$, where $d^l = H^l \times W^l$ denotes the dimension of the feature vector, we get a feature aggregating matrix $\mathbf{F}_l = \left[\mathbf{v}_1^l; \ldots; \mathbf{v}_{C^l}^l\right] \in \mathbb{R}^{C^l\times d^l}$. There are two types of $l(\cdot)$:

Feature-wise: The first type considers each feature as a visual token that directly inputs the Transformer. The token dimension is the length of the flattened feature matrix. Based on our previous analysis, each feature in this input format corresponds to a real-world semantic attribute.

Point-wise: $l(\cdot)$ computes the transpose of the feature matrix $\mathbf{F}_l^\top \in \mathbb{R}^{d^l\times C^l}$. It considers each feature map point, which is the mapping embedding of a corresponding pixel patch, as the visual semantic token. The token dimension is the channel number.

DETR outputs the predicted results when inputting the learned tokens and a set of learned queries to the Transformer decoder. To simultaneously accomplish both object classification and coordinate regression tasks, the total loss function of DETR is formulated as

$$\mathcal{L}_{DETR} = \sum_{i=1}^{N_{dc}} (\lambda_{cls}\mathcal{L}_{cls}^i + \lambda_{box}\mathcal{L}_{box}^i + \lambda_{giou}\mathcal{L}_{giou}^i + \lambda_{card}\mathcal{L}_{card}^i), \tag{2}$$

where i is the i_{th} layer of N_{dc} Transformer decoder layers, $\mathcal{L}_{cls}^i$, $\mathcal{L}_{box}^i$, $\mathcal{L}_{giou}^i$, and $\mathcal{L}_{card}^i$ are object classification loss,

coordinate regression loss, giou loss, and the auxiliary loss, respectively. λ represents the corresponding coefficient. DETR finally sums up all losses.

B. Measurement of Feature Disentanglement

In this paper, we reference the framework of human understanding a specific complex mathematical and physical problem by decoupling it into some straightforward and essential directions. It can be known that CNN realizes the function of extracting image information to certain features. Transformer entangles them to the final output. However, Learned features may contain spurious correlations [26] that can harm the robustness and decoupling of object detection models. Analyzing and improving disentanglement, which reduces these correlations, can unveil the operational mechanisms of object detection models, ultimately enhancing their interpretability and trustworthiness.

In this paper, the Batch averaged Feature Inner Product (BFIP) and Batch averaged Feature Cosine Similarity (BFCS) based on matrix inner product [14] are utilized as measures of the degree of disentanglement among features, which can also be regarded as metrics for feature similarity and orthogonality. BFIP and BFCS of layer l, denoted by $\mathcal{S}^l_{BFIP}$ and $\mathcal{S}^l_{BFCS}$, are formulated as

$$\mathcal{S}^l_{BFIP} = \frac{1}{B}\sum_{b=0}^{B-1}\sum_{i=0}^{C^l-1}\sum_{\substack{j=0\\ j\neq i}}^{C^l-1}\left\langle f^l_i, f^l_j\right\rangle, \tag{3}$$

$$\mathcal{S}^l_{BFCS} = \frac{1}{B}\frac{1}{C^l(C^l-1))}\sum_{b=0}^{B-1}\sum_{i=0}^{C^l-1}\sum_{\substack{j=0\\ j\neq i}}^{C^l-1}\frac{\left\langle f^l_i, f^l_j\right\rangle}{\|f^l_i\|_2\|f^l_j\|_2}, \tag{4}$$

where $f^l_i \in \mathbb{R}^{H^l\times W^l}$ is the i_{th} feature of layer l, B is the batch size, H^l and W^l are the feature dimension, $<\cdot>$ is the inner product operator, $\|\cdot\|_2$ is the normalization operator. To maximize the utilization of GPUs, we upgrade the equations to tensor inner product formats:

$$\mathbf{M} = \mathbf{F}_l \cdot {\mathbf{F}_l}^\top, \tag{5}$$

$$\mathcal{S}^l_{BFIP} = \frac{1}{B}\left(\sum \mathrm{m}_{i,j} - \sum|diag(\mathbf{M})|\right), \tag{6}$$

$$\mathbf{N} = \frac{\mathbf{F}_l}{\|\mathbf{F}_l\|_2}\cdot\frac{{\mathbf{F}_l}^\top}{\|{\mathbf{F}_l}^\top\|_2}, \tag{7}$$

$$\mathcal{S}^l_{BFCS} = \frac{1}{B}\frac{1}{C^l(C^l-1))}\left(\sum \mathrm{n}_{i,j} - \sum|diag(\mathbf{N})|\right), \tag{8}$$

where $\mathbf{F}_l \in \mathbb{R}^{B\times C^l\times d^l}$ is a feature-wise tensor considering batch size, ${\mathbf{F}_l}^\top \in \mathbb{R}^{B\times d^l\times C^l}$ is the transpose of F^l, $\mathrm{m}_{i,j}$, $\mathrm{n}_{i,j}$ are the element of tensor $\mathbf{M}$ and $\mathbf{N}$, $|diag(\mathbf{M})|$ is the absolute value of all pivot diagonal elements of $\mathbf{M}$.

When analyzing DETR's architecture (Fig. 1), the CNN handles feature decoupling and extraction, while the Transformer entangles features for object detection. The joint position acts as a bridge connecting these two functions. Therefore, analyzing decoupling characteristics at this position is crucial. Additionally, features at the backbone's output position (abbreviated as bkb) can supplement the analysis of the decoupling traits of the classification-pretrained backbone at the joint position.

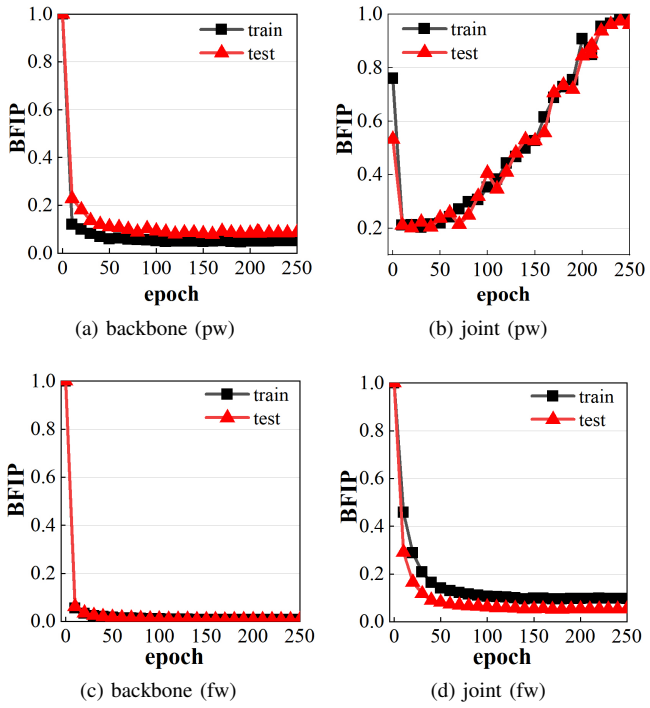
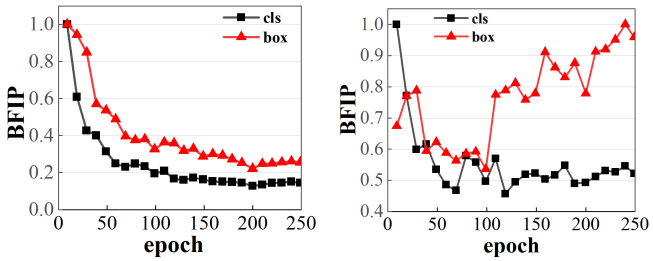


Fig. 2: BFIP analysis of original DETR. All values are max-normalized, with the largest being 1. (a) and (c) represent the $\mathcal{S}_{BFIP}$ whose feature matrices are tokens from the position of backbone output (bkb) and (b) and (d) represent that value of the joint position, respectively. (a) and (b) show $\mathcal{S}_{BFIP}$ in point-wise input format (abbreviated as pw) while (c) and (d) are in feature-wise (abbreviated as fw) input format.

Firstly, we use Equ. 6 to examine the inner product of features. Fig. 2 indicates that the feature-wise input format performs better than the point-wise input format in terms of feature disentanglement. However, it's important to note that we achieve better object detection performance in the point-wise input format, as shown in Table V. Additionally, prior research has shown that establishing a correspondence between decoupled features and semantic attributes is challenging in unsupervised settings. Hence, this paper follows the practice of using a point-wise format commonly adopted by discriminative computer vision models.

As shown in Fig.2, the BFIP value of the backbone position monotonically decreases and converges to a lower limit as training progresses. In contrast, the BFIP value gradually increases in the joint position, indicating that features are becoming more similar and more coupled. It's worth noting that there is only one CNN layer (with a kernel size of 1x1) between the backbone and the joint position. What factors are responsible for this CNN layer having such a significant impact that it alters the trend of the BFIP value?

The backbone used in the model is pre-trained on the Im-



(a) backbone (b) joint

Fig. 3: BFIP analysis of original DETR working on separately classification (cls) and regression (box) mode. (a) and (b) represent the $\mathcal{S}_{BFIP}$ whose feature matrices are tokens from the position of backbone output (bkb) and joint, respectively. All values are max-normalized, with the largest being 1.

ageNet dataset for a classification task and was subsequently fine-tuned with a small learning rate in our object detection training schedule. This results in the pre-trained backbone retaining a strong classification inductive bias that is resistant to change through fine-tuning. Under such an explanation, we primarily infer that it is the regression function of object detection that causes the opposite directions observed in Fig.2. To be more specific and precise, we successively disable the regression and classification functions to reveal their own role in DETR. The BFIP values are acquired accordingly as shown in Fig.3. When we train only the classification function (i.e., set λ_{box}, λ_{giou}, and λ_{card} to be 0 in Equ.2, and also dismiss regression from DETR's query matching function), we find that both $\mathcal{S}_{BFIP}^{bkb}$ and $\mathcal{S}_{BFIP}^{joint}$ decreases. Contrary to this consistency, $\mathcal{S}_{BFIP}^{bkb}$ and $\mathcal{S}_{BFIP}^{joint}$ exhibit opposite trends at the case of individual training regression function (i.e., segregate all settings related to classification). Under these circumstances, our initial assumption has been validated. It is the regression function that causes the opposite trend.

To further analyze the impact of classification and regression on features of backbone and joint position, BFCS (Equ.8) is utilized to compute the similarity, as shown in Table I. It can be observed that the degree of disentanglement of the joint position is considerably smaller than that of the backbone position. When only enabling the classification function, we find that $\mathcal{S}_{BFCS}^{bkb}$ slumps while $\mathcal{S}_{BFCS}^{joint}$ has a slight increase. The same result is obtained when individually training the regression function. The entanglement between classification and regression significantly exacerbates the similarity computed from the backbone position. This explains why double-backbone object detection models [25] behave feature activation results when decoupling classification and regression, whereas single-backbones do not have this effect.

Comparing Equ.6 with Equ.8, the difference between BFIP and BFCS lies in the feature norm regularization. Given the scenario where $\mathcal{S}_{BFIP}^{joint}$ increases while $\mathcal{S}_{BFCS}^{joint}$ remains nearly unchanged, we deduce that the absolute values of features, forced by regression loss, are becoming larger, while those forced by classification loss remain stable. Considering that only a limited number of kernels are activated and a small

TABLE I: BFCS and BFSR results on VOC dataset.

Method	Loss	BFSR	Position	BFCS
DETR	cls + box	352	bkb	0.68
			joint	0.25
	cls	306	bkb	0.24
			joint	0.26
	box	403	bkb	0.48
			joint	0.26
DETR + BFSP	cls + box	280	bkb	0.44
			joint	0.22
	cls	328	bkb	0.16
			joint	0.26
	box	271	bkb	0.26
			joint	0.23

number of features correspond to high-level semantics [19], [57], features should exhibit characteristics of discreteness and sparsity. As a consequence, the regression function causes a small number of features at the joint position to be activated to a much greater extent than other features, resulting in a deteriorating imbalance among features. This imbalance makes the activation area more centralized and reduces the richness of object semantic information. Therefore, equilibrating the feature imbalance induced by the regression loss and enhancing the feature disentanglement is crucial for optimizing DETR series single-backbone one-stage object detection models.

C. Batch Averaged Feature Spectral Penalization

Mathematically, singular values represent most characteristics of a tensor. Larger singular values have a greater impact on a tensor when the ratio between them is larger. Batch feature averaged singular radio (BFSR) can represent the feature distribution. It is formulated as

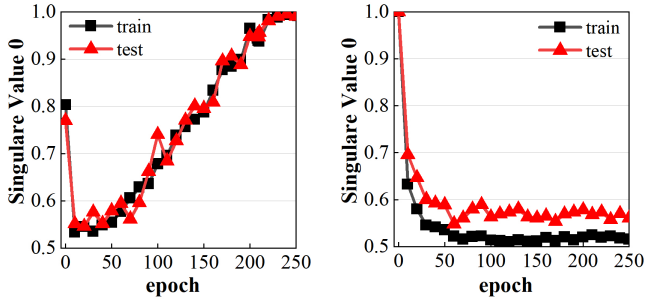
$$\mathcal{S}_{BFSR} = \frac{\sum_{b=0}^{B-1}\sum_{i=0}^{C^l-1}\sigma_0}{\sum_{b=0}^{B-1}\sum_{i=0}^{C^l-1}\sigma_{-1}}, \tag{9}$$

where σ_0 and σ_{-1} represent the largest and smallest singular value. Table I shows $\mathcal{S}_{BFSR}$ of the features from the joint position. The regression loss significantly increases the $\mathcal{S}_{BFSR}$, whereas the classification loss decreases it. The larger the $\mathcal{S}_{BFSR}$ is, the more imbalanced the features become.

Literature has shown that forcing the distribution of singular values to be balanced can achieve the goal of reducing feature correlation and improving the representation and generalization of visual recognition tasks [58], [59]. We analyze the largest singular value and decompose the extracted feature tensor based on T-SVD [27], [28], [60]. As shown in Fig.4a, the largest singular value follows the same trend as the value of $\mathcal{S}_{BFIP}$ shown in Fig. 2b and Fig. 3b. Features extracted from the input batch are first stacked as a 4-D tensor $\mathbf{F}_{\mathrm{T}} = [\mathcal{F}_1, \mathcal{F}_2, \cdots, \mathcal{F}_B] \in \mathbb{R}^{B\times C^l\times H^l\times W^l}$. We then use T-SVD to decompose the feature tensor:

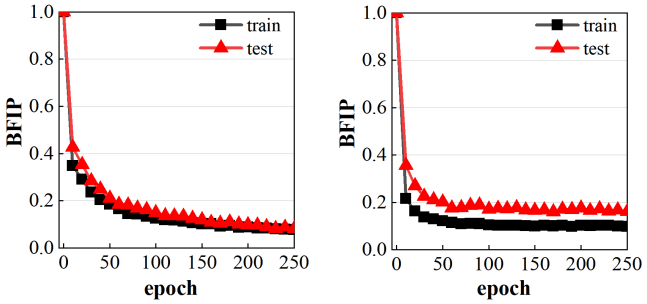
$$\mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top = T\text{-}SVD(\mathbf{F}_{\mathrm{T}}), \tag{10}$$

where $\mathbf{\Sigma} \in \mathbb{R}^{B\times C^l\times L^l\times L^l}$ is the obtained singular values, L^l is the min value between W^l and H^l, $\mathbf{U}$ and $\mathbf{V}^\top$ are



(a) DETR (b) DETR + BFSP

Fig. 4: Singular value analysis. All values are max-normalized, with the largest being 1. (a) represents the largest singular value decomposed by T-SVD over features of the joint position(sv0) of the DETR model. (b) represents the value decomposed from the model with the BFSP loss.



(a) backbone (b) joint

Fig. 5: BFIP analysis of DETD. All values are max-normalized, with the largest value set to 1. (a) and (b) represent the BFIP values computed using feature matrices extracted from the backbone output position and the joint position.

left and right orthogonal tensors whose columns are singular vectors, respectively. After that, the largest singular values of $\mathbf{\Sigma}$ are collected to compute the Batch Feature averaged Spectral Penalization loss (BFSP):

$$\mathcal{L}_{\text{bfsp}} = \frac{1}{B}\sum_{b=0}^{B-1}\sum_{c=0}^{C^l-1}\sigma_{b,c}^2, \tag{11}$$

where $\sigma_{b,c}$ is the largest singular value belonging to the feature extracted from channel c and image b of batch B.

The BFSP loss is inspired by BSP loss [58] and the differences are clear. First of all, BFSP is used as a feature disentanglement and balance intensifier. Our motivation is to enhance feature disentanglement and balance the feature distribution for single-backbone object detection models. The analysis of the feature decoupling properties in classification and regression has led us to use BFSP to optimize object detection models. BSP is used in a domain-adaptive classification scenario. Beyond that, the defined BFSP loss targets the optimization and decomposition of the set of all features within a batch. This is in comparison with BSP, which takes a single feature from the final CNN output layer as input rather than the middle layer of the neural network. Furthermore, BSP does not take into account the correlation between features.

After deploying the $\mathcal{L}_{\text{bfsp}}$, the trends of the $\mathcal{S}_{BFIP}$ and the largest singular value have completely shifted downward, as demonstrated in Fig.5 and 4b. This indicates an enhancement in disentanglement ability, as depicted in Table I.

IV. Approach

After demonstrating the disentanglement property and proposing the method, we further formally build the Explainability Enhanced object detection Transformer with feature Disentanglement (DETD) model, as shown in Fig.6.

A. Disentanglement and Reconstruction

In order to take full advantage of the decoupling property and give a careful but uninjurious dissection of DETD, we disentangle the features to a set of bases first and reconstruct them as a kind of optimized tokens.

Disentanglement: Traditional disentangling representation learning methods mainly extract the decomposed components by the same number of Networks. But these methods can hardly guarantee the orthogonality of every base and the integrity of information [24]. T-SVD is a better method to realize the decomposition without adding parameters. Concretely, H^l and W^l are set to be equal for computing simple and practical. After computing Equ.(10), we extend the dimension of those two orthogonal tensors $\mathbf{U}$ and $\mathbf{V}$, and a diagonal tensor $\mathbf{\Sigma}$ to $B \times C^l \times H^l \times H^l \times H^l$. Inner product among them and an orthogonal identity base is then computed as

$$\mathbf{B} = \mathbf{U}\mathbf{\Sigma}\mathbf{E_b}\mathbf{V}^\top, \tag{12}$$

where $\mathbf{E}_b \in \mathbb{R}^{B\times C^l\times H^l\times H^l\times H^l}$ is a 5-D tensor. It is repeatedly constructed $B \times C^l$ times by duplicating a cubic identity tensor whose elements are all zero except the one in the principal diagonal. $\mathbf{B} \in \mathbb{R}^{B\times C^l\times H^l\times H^l\times H^l}$ represents the total feature bases. The feature bases decomposed by every certain feature are orthogonal because of the orthogonal identity bases $\mathbf{E}_b$. Section V-B also shows that the feature bases decomposed from different features are orthogonal.

Reconstruction: In this article, we take the reconstruction method as one optimization strategy of the feature bases. According to Equ. (10) and (12), we get a set of singular values from the b_{th} image c_{th} feature $\mathbf{\Sigma}_{b,c} = \{\sigma_1, \sigma_2 \ldots, \sigma_{H^l}\}$ with $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_{H^l}$ and a set of feature bases $\mathbf{B}_{b,c} = \{b_1, \ldots, b_{H^l}\}$. Considering that the bases with larger singular values have more implications on the total tensor and the tensor can be reconstructed by summing up different bases [60], we leverage these properties as rules to optimize the feature tensor:

$$\mathbf{F_{rec}} = \mathbf{U}\mathbf{\Sigma_{rec}}\mathbf{V}^\top = \sum_{i=s}^{o}\sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^\top = \sum_{i=s}^{o} b_i, \tag{13}$$

where $\boldsymbol{u}_i$ and $\boldsymbol{v}_i$ represent the eigenvectors of $\mathbf{U}$ and $\mathbf{V}$, s and o are two hyperparameters for defining the start and end positions in the singular value set $\mathbf{\Sigma}_{b,c}$ or feature base set

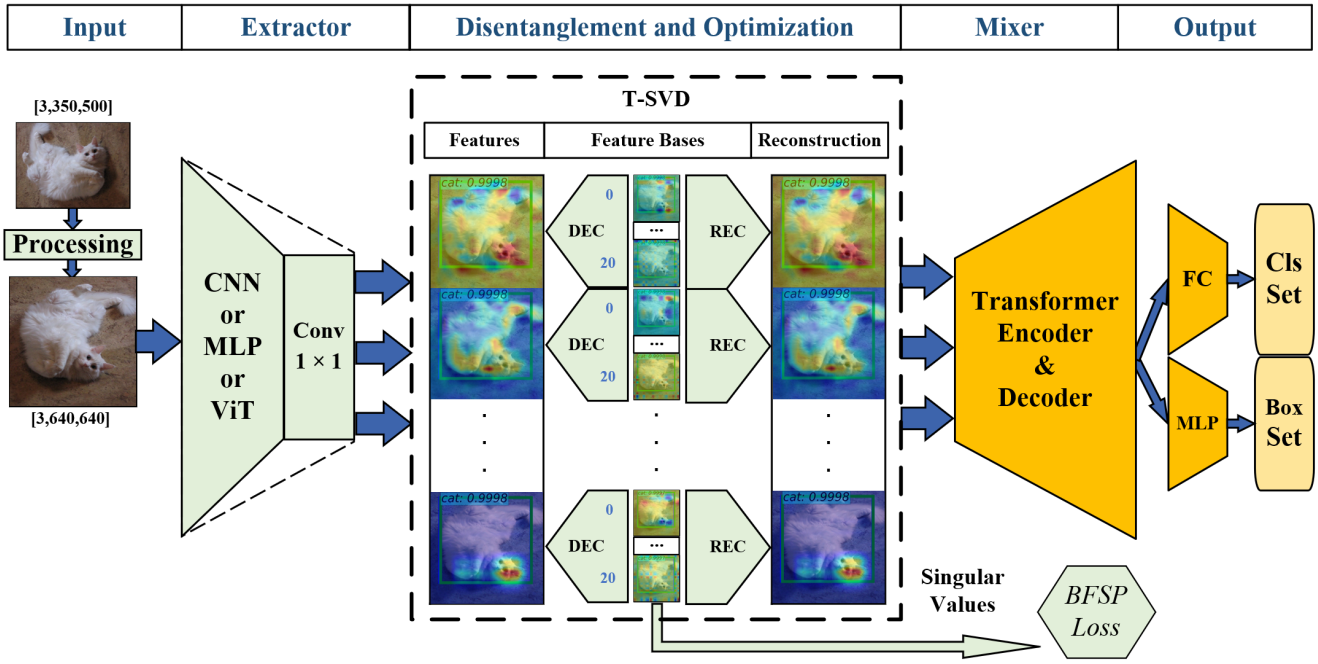


Fig. 6: Illustration of DETD Model. All modules in the color orange are DETR modules; in the color green are the modified DETD parts. A 1×1 convolution layer is added to the extractor module to adjust the input dimension. T-SVD decomposes feature tensor. The produced feature bases are shown in sequence according to the order of eigenvalues. The singular values for computing $\mathcal{L}_{BFSP}$ are taken from the T-SVD module.

$\mathbf{B}_{b,c}$. This method is used without adding further complex computation since we have already decomposed the features for computing BFSP.

B. Model with Disentanglement

Concretely, a given image in random size is firstly resized to square 640×640 (will be discussed in section V-C). We then extract a set of backbone features in size of $2048 \times 20 \times 20$ after the 32 times shrunk. A convolution layer with kernel size one is utilized to adjust the channel number of the features to 256. Similar to literature [59], we add 1e-3 to the feature matrix to enhance the stability of feature decomposition. T-SVD module is applied on the feature tensor taken from the joint position to get the disentangled feature bases $\mathbf{B}$. The singular values are used for computing BFSP loss to realize the disentanglement goal. After the reconstruction module Equ. (13), the generated features $\mathbf{F_{rec}}$ carry varying levels of information derived from the original features according to the configuration of hyperparameter s and o. Finally, the total loss of DETD can be formulated as

$$\mathcal{L}_{DETD} = \mathcal{L}_{DETR} + \lambda_{BFSP}\mathcal{L}_{BFSP}, \tag{14}$$

where λ_{BFSP} is the corresponding coefficient.

C. DETD Model Variants

In order to give more convincing proof to show that our feature disentanglement method is indeed efficient no matter in what kinds of single-backbone object detection Transformer architectures, we extend our method to five variants: CNN backbone based, ViT backbone based, MLP backbone based, Deformable DETR based, and DINO based, respectively. The last two variants are abbreviated as DFDETD and DIDETD. In addition, we also deploy the proposed method on the Borderdet model (abbreviated as BDDETD) to validate our method's performance on the traditional pure CNN single-backbone one-stage object detection model.

ViT Based Transformer: Vision Transformer (ViT) [45] is one of the most popular Transformer frameworks in computer vision. It achieved SOTA performance in the classification task. It has been pre-trained on the ImageNet dataset in the conditions of patch size set to 16, embedding dimension set to 384, and the pre-training image size set to 224. We adopt the pre-trained ViT offered by timm python package. However, the difference from conventional ViT approaches lies in that we delete the class token of the ViT output to make the dimension in tune with that of the mixer. Regarding the bridge between the ViT extractor and DETD mixer, the output of the final ViT block is reshaped to a feature tensor by two CNN layers. The first CNN layer, whose kernel size is three, mainly adjusts the feature size to 20×20 to reduce the computing complexity of T-SVD. By the way, we can also apply other parameterless modules to replace this CNN layer to build a pure Transformer based DETD, such as Global Average Pooling (GAP), with just a slight performance perturbation. The latter CNN is the same as the treatment of the ResNet backbone discussed above. Actually, the convolution layer with kernel size one can be seen as a linear transformation layer.

MLP based Transformer: MLP is the most classical architecture of deep learning. Referencing the framework of

Transformer, MLP-mixer [50] makes the MLP architecture get back to the peak in classification tasks. However, We have to choose AS-MLP version [61] as the MLP based feature extractor rather than that famous noumenon because it can not adaptively adjust the input image size to meet the hybrid object detection Transformer's requirement. AS-MLP is an evolution version that is able to capture the local dependencies and be applied to the downstream object detection tasks by axially shifting channels of the feature map. It is also pre-trained on the ImageNet dataset in the conditions of patch size set to 4, embedding dimension set to 96, shifting size set to 5, and the pre-training image size set to 224.

Deformable DETR and DINO Variants: In this part, we mainly want to show that our method can be efficiently applied to other end-to-end hybrid Transformer object detectors. Deformable DETR [4] and DINO [62] are two advanced versions of DETR. They significantly upgrade the architecture of DETR and drastically reduce the training time by extracting multi-scare feature maps. In this study, we conduct experiments on them with the model configuration of $L = 3$. It extracts the output feature maps of the last two stages. The third feature map, the lowest resolution, is obtained via a 3×3 stride 2 convolution on the final ResNet stage. Accordingly, we add three CNN layers to parallelly adjust the feature size and the dimension to match the Transformer. We sum up the bases proportionally because of their multi-scale approach.

CNN based detection model Borderdet: Borderdet model [63] is a classic in the realm of CNN based object detection models. What sets it apart from the previous models is that we didn't add extra convolutional layers. Instead, we directly apply the BFSP and optimization strategies between the feature extractor and object detector.

V. Experiments

We train and evaluate the DETD models on two object detection datasets. The code of DETD will be public at https://github.com/YuWLong666/DETD.

A. Experiment Setup

We employ PASCAL VOC 2007 + 2012 training and validation splits for training and test 2007 split for evaluation, which results in about 15K and 5k images, respectively [64]. It contains 20 classes of images and bounding box annotations. We also perform experiments on MS COCO 2017 [65] dataset containing 118k training images and 5k validation images in 80 classes. The performance metrics include mAP, AP50, AP75, and other integral metrics over multiple thresholds.

We follow DETR [3] to set the basic training parameters of DETD. DFDETD, DIDETD, and BDDETD are follow their counterparts, which means we do not adjust the parameters harshly but get a more excellent result. We train and evaluate on 2080Ti GPUs with the batch size set to 4 (2 for ResNet-152, ViT-Base, and ASMLP-Base backbones) for VOC dataset. As for COCO dataset, we use 3090 GPUs with the batch size set to 4. The length of the DETD Transformer decoder query is set to 25 when trained on VOC dataset and is changed to 100 when it turns to COCO due to the increase of categories. It is set to 75 when used to train DFDETD on VOC dataset since that number in traditional Deformable DETR is three times that of DETR. It is set to 400 when training DIDETD on VOC dataset and 600 for COCO.

According to Equ.(10), Equ. (13), and Fig.6, we decompose the original features to feature bases and optimize them by summing up the first o bases. Based on this paradigm, four types of DETD models have been trained and evaluated. An initial DETD model with both BFSP loss and reconstruction procedure is established to realize the whole idea. It is trained from scratch for 250 epochs. After that, a DETR model is trained for 200 epochs to achieve a pre-trained checkpoint in no BFSP and no T-SVD mode. We then use this checkpoint to fine-tune the DETD model with BFSP loss and reconstruction. We drop the final two-branch FC networks and the mapping layers between the extractor and Transformer to retrain them in the fine-tuning process. This cross-trained DETD model (abbreviated as DETD-cross) is trained for 150 epochs to ensure the training converges. The learning rate schedule maintains 0.0001 for the first 90 epochs and drops by ten times afterward. Finally, DETD without BFSP and DETD-cross without BFSP are trained in the same setting to evaluate the optimization step. All these DETD models and DETR baseline derive their branches that are constructed by CNN, MLP, Transformer, Deformable, DINO, and Borderdet respectively. λ_{BFSP} is set to 0.001 when using BFSP loss.

B. Feature Disentanglement Results

Equ.(12) has shown that the bases decoupled from a specific feature are orthogonal. We further compute the BFIP value among all feature bases. Fig.7a, 7c, and 7e show the BFIP values of DETDs whose feature extractor are ResNet-50, ViT-Small, and ASMLP-Tiny, respectively. When considering the BFCS values, shown in Table II, it becomes apparent that Transformer and MLP-based object detection models outperform the CNN-based model from the perspective of feature disentanglement. What needs to be emphasized is that we do not force features produced by DNN to be strictly orthogonal. Feature disentanglement reduces redundant correlations, enhancing the possibility of achieving semantic decoupling and alignment. The proposed DETD models with BFSP loss can improve the orthogonality of all feature bases as they converge to smaller values approaching zero during the training process.

As shown in Fig.7b, 7d, and 7f, the BFSP loss reduces the largest singular value by order of magnitude, with the ratio between larger and smaller also decreasing. To intuitively visualize the feature distribution, we compile statistics on the features extracted from the joint position, as shown in Fig.8. Specifically, we use Global Average Pooling (GAP) to compute the activation value of every feature. Subsequently, a histogram is created to represent the activation degree of each feature. It can be observed that most features are in a state of inhibition, indicating that an equal number of CNN kernels are suppressed. The BFSP loss has a significant positive impact on the similarity of the regression function due to the balancing effect among features, as shown in Fig. 8b. It is noteworthy that the BFSP loss encourages more CNN kernels to activate

TABLE II: Results of object detection performance(AP, %) of different DETR and DETD models on VOC dataset. BFCS represents the BFCS value of the joint position. The item of Epochs records the number of total training epochs. The GPU Hours are tested on these conditions: single 2080Ti GPU, batch size 1, and epoch 1. Fix means the model takes fix-sized 640 $\times$ 640 images as input. 256 and 400 represent the Transformer's embedding dimension. T, S, and B represent Tiny, Small, and Base versions of the according backbones, respectively.

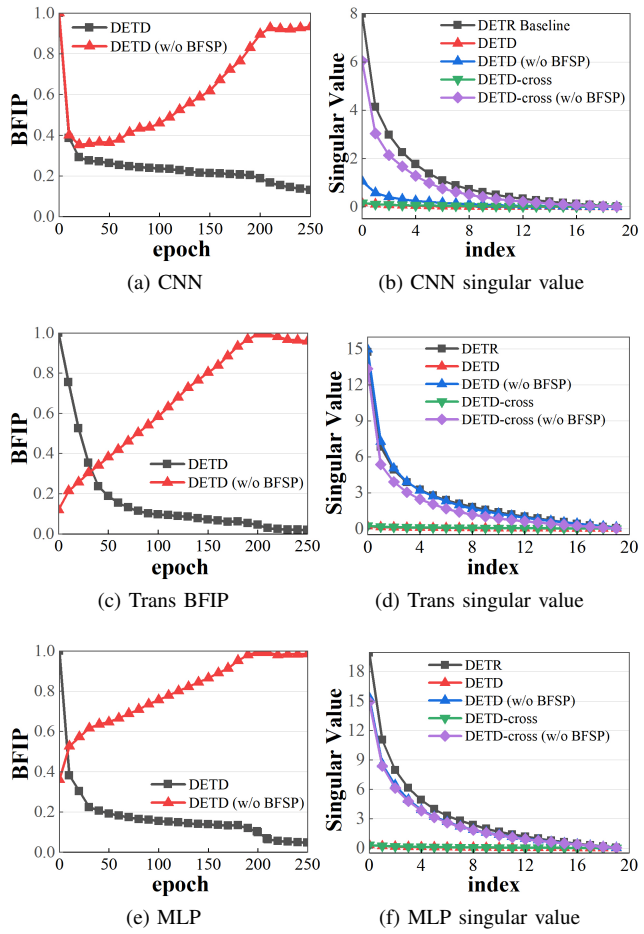
Extractors		Models	mAP	AP_{50}	AP_{75}	BFCS	Epochs	Pretrain Epochs	Params	GPU Hours
CNN	ResNet-50	DETR [5]	54.10	78.00	58.30	–	–	–	41.3M	1.06
		DETR (Fix 256)	53.76	77.52	57.69	0.25	250	–	41.9M	1.12
		DETR (Fix 400)	54.17	77.67	58.19	0.25	250	–	55.5M	1.00
		DETR (Fix 400)	54.26	77.72	58.29	0.27	350	–	55.5M	1.00
		DETR (+ BFSP)	53.28	78.04	57.03	**0.22**	250	–	41.9M	1.03
		DETD	53.65	77.70	57.62	**0.22**	250	–	41.9M	1.03
		DETD (w/o BFSP)	54.35	77.92	58.12	0.25	250	–	41.9M	1.05
		DETD-cross	54.67	77.72	58.62	**0.22**	150	200	41.9M	1.03
		DETD-cross (w/o BFSP)	**55.14**	**78.05**	**59.22**	0.23	150	200	41.9M	1.05
	ResNet-101	DETR	54.60	78.49	58.42	0.25	250	–	60.2M	1.58
		DETD	54.76	78.68	59.57	**0.15**	250	–	60.2M	1.61
		DETD (w/o BFSP)	54.73	78.67	59.09	0.32	250	–	60.2M	1.60
		DETD-cross	55.51	**78.95**	60.43	0.22	150	200	60.2M	1.61
		DETD-cross (w/o BFSP)	**55.66**	78.75	**60.51**	0.23	150	200	60.2M	1.60
	ResNet-152	DETR	56.14	79.79	60.79	0.26	250	–	75.8M	1.70
		DETD	56.14	**80.00**	60.32	**0.25**	250	–	75.8M	1.73
		DETD (w/o BFSP)	56.04	79.72	60.91	0.26	250	–	75.8M	1.72
		DETD-cross	56.59	79.65	60.91	**0.25**	150	200	75.8M	1.73
		DETD-cross (w/o BFSP)	**56.69**	79.72	**61.47**	0.26	150	200	75.8M	1.72
Transformer	ViT-S	DETR	54.30	79.16	57.30	0.38	250	–	41.6M	1.12
		DETD	55.22	79.11	58.88	**0.19**	250	–	41.7M	1.20
		DETD (w/o BFSP)	54.85	**79.49**	58.18	0.28	250	–	41.7M	1.16
		DETD-cross	55.27	78.30	58.94	0.30	150	200	41.7M	1.20
		DETD-cross (w/o BFSP)	**55.34**	78.67	**59.36**	0.40	150	200	41.7M	1.16
	ViT-B	DETR	56.87	80.10	61.18	0.34	250	–	108.1M	2.26
		DETD	56.93	79.96	60.52	**0.29**	250	–	108.1M	2.30
		DETD (w/o BFSP)	57.23	**80.74**	61.60	0.31	250	–	108.1M	2.29
		DETD-cross	57.05	79.53	60.76	0.30	150	200	108.1M	2.30
		DETD-cross (w/o BFSP)	**57.45**	79.25	**61.61**	0.34	150	200	108.1M	2.29
MLP	ASMLP-T	DETR	58.96	81.79	63.20	0.23	250	–	45.2M	1.42
		DETD	59.07	82.25	63.73	**0.14**	250	–	45.2M	1.40
		DETD (w/o BFSP)	59.19	**82.82**	63.37	0.22	250	–	45.2M	1.37
		DETD-cross	59.49	81.74	64.03	**0.14**	150	200	45.2M	1.40
		DETD-cross (w/o BFSP)	**60.10**	82.40	**65.15**	0.23	150	200	45.2M	1.37
	ASMLP-S	DETR	59.99	82.98	65.20	0.26	250	–	66.5M	1.88
		DETD	60.49	83.32	65.81	**0.24**	250	–	66.5M	1.91
		DETD (w/o BFSP)	60.07	83.17	65.62	0.25	250	–	66.5M	1.90
		DETD-cross	**60.94**	**83.51**	**66.60**	**0.24**	150	200	66.5M	1.91
		DETD-cross (w/o BFSP)	60.65	83.14	66.22	0.25	150	200	66.5M	1.91
	ASMLP-B	DETR	61.44	83.81	66.77	0.34	250	–	104.5M	2.51
		DETD	61.26	83.28	66.92	**0.31**	250	–	104.5M	2.55
		DETD (w/o BFSP)	61.84	83.75	67.49	**0.31**	250	–	104.5M	2.54
		DETD-cross	**62.57**	**83.88**	67.87	**0.31**	150	200	104.5M	2.55
		DETD-cross (w/o BFSP)	62.28	83.67	**68.03**	0.32	150	200	104.5M	2.54

and play essential roles in the entire model. As more features are activated, the similarity between features decreases. The BFSP loss effectively compels different features to pay more detailed attention to the task.

C. Optimization Strategies Analysis

Deep learning model training involves learning correlations between data [26], while BFSP has the opposite effect. Therefore, intuitively, BFSP potentially reduces object detection performance. We conduct an experiment where only BFSP is deployed without any optimization measures, as shown in Fig. 9. It can be observed that the introduction of BFSP leads to a performance decrease. As the BFSP coefficient increases, this phenomenon becomes more pronounced. We explain this performance degradation as being caused by the early introduction of BFSP into the model training. The BFSP loss maintains the same strength throughout the entire training. The learned features are strongly decoupled, but the learning of indispensable correlations is insufficient.

The optimization strategies in this paper mainly consist of two parts. The first is the reconstruction process after T-SVD decomposition based on Equ.13. Considering that the tensor

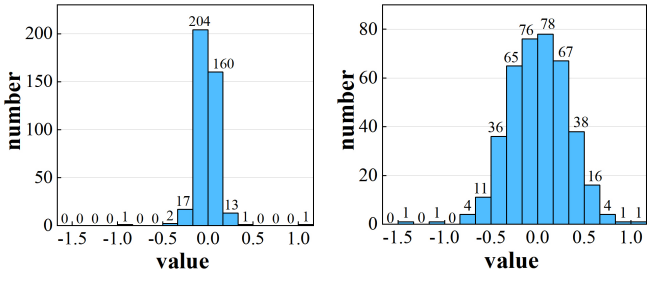


(a) CNN (b) CNN singular value

(c) Trans BFIP (d) Trans singular value

(e) MLP (f) MLP singular value

Fig. 7: Feature Disentanglement analysis. We compute the BFIP values of DETD and DETD without BFSP loss, as shown in (a), (c), and(e). (b), (d), and (f) show the singular values of different models. BFIP values are max-normalized, with the largest being 1. Different rows represent CNN, Transformer, and MLP based hybrid models, respectively.



(a) DETR (b) DETD

Fig. 8: Histograms of features in the joint position. Values are max-normalized, with the largest being 1.

can be reconstructed by summing up different bases [60]. we leverage these properties as rules to boost the object detection task. This technique can implicitly mitigate the impact of the BFSP loss, thereby improving object detection performance. For example, we only utilize the first 8 bases (i.e., o=8), effectively setting the singular values of the remaining 12 bases

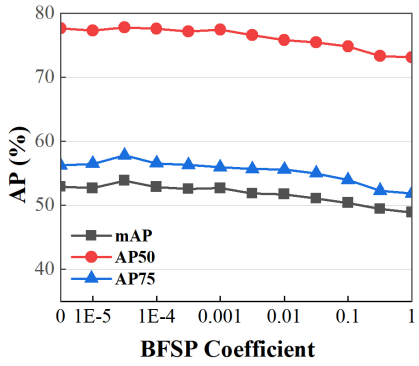


Fig. 9: BFSP coefficient analysis on VOC dataset. The evaluated model is ResNet-50 DETR.

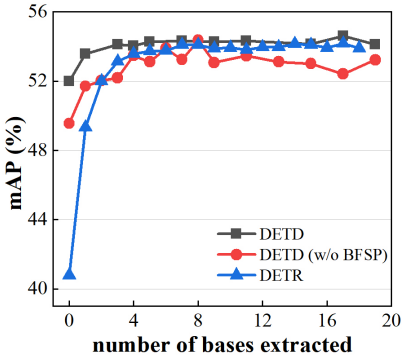


Fig. 10: The mAP values(AP, %) of different DETD models and different bases. We reconstruct features by Equ. (13) with setting $s = 0$. The x-axis represents the o value.

to zero, thereby increasing the ratio of large feature values.

We conduct experiments regarding the number of feature bases extracted, as shown in Fig.10. It shows that the performance grows to a convergence point as the number o increases. As for different feature extractors, we achieve better performance when o is set to 40% to 70% of all feature bases. Experimental results, Table II, confirm that this approach can indeed enhance model performance. Furthermore, as depicted in the figures, the introduction of BFSP loss makes the distribution of feature bases more uniform. The uniformity of DETD surpasses that of models without BFSP.

The second part is the explicit training schedule of BFSP loss. DETD-cross and DETD(w/o BFSP)-cross models adopt a stepwise paradigm for setting the coefficient of BFSP λ_{BFSP}, starting with an initial value of 0 and transitioning to 0.001 after a certain number of epochs. This method is essentially a trade-off between decoupling and object detection performance. From Table II, the maximum decoupling occurs when BFSP loss is introduced at the beginning of training (i.e., DETD), which reduces model performance. Introducing BFSP in the later stages of training (i.e., DETD-cross) can enhance decoupling characteristics while improving performance.

In this paper, we primarily adopt a fixed size of 640 as the model's input. Firstly, Original DETR uses randomly sized

TABLE III: Object detection performance(AP, %) of DINO models in various input formats. The GPU Hours are acquired in these conditions: single 3090 GPU, batch size 1, epoch 1. The evaluated dataset is COCO.

Input Type	mAP	$\mathbf{AP}_{50}$	$\mathbf{AP}_{75}$	$\mathbf{AP}_{\mathrm{S}}$	$\mathbf{AP}_{\mathrm{M}}$	$\mathbf{AP}_{\mathrm{L}}$	DIDETD GPU Hours
DINO (Fix 640)	40.18	59.23	42.42	18.33	44.22	61.09	**8.16**
DINO (Fix 768)	42.45	61.68	45.31	20.68	46.54	61.98	8.56
DINO (Fix 1024)	44.98	64.02	48.33	24.50	49.43	**62.20**	8.79
DINO (Fix 1280)	**46.15**	**64.76**	**49.66**	**27.50**	**50.30**	61.83	25.43
DINO (Original)	44.57	63.87	47.48	24.56	49.14	61.55	18.22

TABLE IV: Object detection performance(AP, %) results of DINO and Deformable models on VOC dataset.

Methods	Input Type	Models	mAP	$\mathbf{AP}_{50}$	$\mathbf{AP}_{75}$	Epochs	Pretrain Epochs
Deformable	Square 640	DFDETR (Fix 640)	56.22	80.52	61.62	60	–
		DFDETD	56.43	80.59	61.81	60	–
		DFDETD (w/o BFSP)	**56.61**	**80.79**	**62.41**	60	–
		DFDETD-cross	56.60	79.98	61.40	40	20
		DFDETD-cross (w/o BFSP)	56.58	79.78	61.75	40	20
DINO	Square 640	DINO (Fix 640)	62.23	82.39	67.31	15	–
		DIDETD	62.21	82.32	67.68	15	–
		DIDETD (w/o BFSP)	62.34	82.47	67.65	15	–
		DIDETD -cross	**62.67**	**82.85**	**68.04**	10	10
		DIDETD -cross (w/o BFSP)	62.55	82.80	68.03	10	10
	Original	DINO (Original)	63.34	82.88	69.19	15	–
		DIDETD	63.33	82.69	69.12	15	–
		DIDETD (w/o BFSP)	63.64	82.92	69.36	15	–
		DIDETD -cross	**64.23**	83.29	**70.35**	10	10
		DIDETD -cross (w/o BFSP)	64.21	**83.47**	70.31	10	10

image matrices, with a maximum input length of 1333, as input. This significantly reduces the speed of GPU SVD decomposition. In such a predicament, a small fixed input size is a good option to choose. Secondly, as shown in Table II, when using a fixed square input size of 640, DETR achieves the same performance as with random inputs on VOC dataset.

We also conduct experiments on the COCO dataset using the DINO model [62] to evaluate whether the input type influences performance, especially for small objects. As shown in Table III, when the input size reaches 1280, the GPU decomposition speed significantly decreases, adding 16.7 GPU hours compared to an input size of 1024, and even slower than the original random inputs. As long as the input does not exceed 1024, the time taken for GPU SVD decomposition can be negligible. Random inputs are also nearly 10 GPU hours slower than fixed 1024 inputs. Small objects (i.e., $\mathbf{AP}_{\mathrm{S}}$) also increase as the model's input size becomes bigger. The impact of fixed input size on small objects is greater than the impact on other sizes. It provides an advantage for larger objects (i.e., $\mathbf{AP}_{\mathrm{L}}$). We find that when using a fixed 1024 input, the model's performance is the same as with the original random inputs regardless of object size. When the input is fixed at 1280, the model even outperforms the original random inputs by 1.58% in terms of mAP and 2.94% in terms of $\mathbf{AP}_{\mathrm{S}}$.

D. Object Detection Performance

Experiments on VOC dataset: Our method is applied to three mainstream backbone-based models, including CNN based (i.e., ResNet-50, 101, and 152), Transformer based (i.e., ViT-Small and Base), and MLP based (i.e., ASMLP-Tiny, Small, and Base) models. The model size has reached a maximum of 108M parameters. We obtain the expected results across all DETD models, as shown in Table II.

DETD models achieve comparable object detection performance to DETR, while the similarity metric BFCS outperforms DETR. All DETD (w/o BFSP) models, DETD-cross models, and DETD-cross(w/o BFSP) models perform better object detection performance than their baselines. DETD-cross(w/o BFSP) models are the best, surpassing their baselines by an average of 1% in terms of mAP. This is mainly attributed to the optimization strategies we specifically proposed. Taking ResNet-50 DETD as an example, the AP performance drops by about 0.5% when BFSP loss is enabled (i.e., 53.28%). However, the performance increases to 54.67% when both optimization metrics are utilized. At these points, the BFCS value remains at its lowest (i.e., 0.22). When only the two optimization strategies are used without introducing the BFSP loss (i.e., DETD (w/o BFSP) and DETD-cross (w/o BFSP)), the model's detection performance reaches its peak, surpassing baseline by 1%. In addition, the convergence epochs of cross-trained DETDs are also significantly reduced thanks to the pre-training process. As for different extractors, we get the same conclusion. ASMLP based models achieve the highest performance, demonstrating the superiority of its framework (62.28%).

When comparing feature similarity (i.e., BFCS in Table II), across all types of backbones, the proposed method indeed enhances the model's decoupling ability. ASMLP-T based DETD achieves the lowest feature similarity 0.14 with the help of BFSP loss. Additionally, we find that as the depth of the

TABLE V: Results of Object Detection(AP, %) in feature-wise input format.

Method	mAP	AP50	AP75
DETR (feature-wise)	42.71	69.05	44.71
DETR (feature-wise + reconstruction)	43.04	68.44	44.69

TABLE VI: Object detection performance(AP, %) and BFCS of Borderdet models on VOC dataset. The pretrain epochs for BDDETD-c and BDDETD-c(w/o BFSP) is 16000 iterations.

Models	**mAP**	$\mathbf{AP}_{50}$	$\mathbf{AP}_{75}$	**BFCS**
Borderdet	56.28	79.31	61.82	0.22
BDDETD	56.29	78.85	60.86	**0.14**
BDDETD (w/o BFSP)	**57.71**	**80.39**	**62.52**	0.21
BDDETD-c	56.32	79.47	61.36	0.20
BDDETD-c (w/o BFSP)	57.32	80.07	62.33	0.21

model backbone increases, all three major frameworks show the characteristic of increasing feature similarity. We infer that, as the model's learning and representation capabilities improve, the model's fitting ability to the same dataset also increases, leading to a stronger inductive bias that can hardly be affected by BFSP loss.

As shown in Table IV, DFDETD and DIDETD preserve their original excellent characteristics in terms of performance and speed. The performance of DIDETD in its original input format also surpasses that of the square 640 format. DINO achieves the best performance, reaching 64.23%. Similarly, all DETD models show improvements compared with their baselines. The largest improvement occurs when the input mode is random size input, with an increase of 0.89%.

The results conducted in the condition of feature-wise input format are shown in Table V. It severely degrades the object detection performance by up to 10% in terms of mAP. This can be explained by Fig. 11.

To further substantiate the model's generalization capabilities, we initially conduct an evaluation on a CNN based single-backbone object detection model, Borderdet [63]. The results are shown in Table VI. We find that the model's performance improves, with the maximum increase being 1.04% in terms of mAP. The feature similarity decreases by 36% when the BFSP loss is taken into account. This is encouraging and suggests that our method is applicable to this class of models.

Experiments on COCO dataset: We also conduct experiments on another extensive dataset, COCO, as shown in Table VII. We reach the same conclusion as on the VOC dataset.

When evaluating DINO on the COCO dataset using three different input formats, as shown in Table VIII, the results are consistent with the experiments on the VOC dataset. The largest improvement occurs when the input mode is random size input, with an increase of up to 0.9% in terms of AP.

These results demonstrate the effectiveness and the generalization ability of our method at both model and dataset levels.

E. Other Penalty Variants

BFCS and BFIP are employed in this paper to quantify the decoupling properties between features and can also be independently utilized as penalties. We conduct experiments by solely adding a penalty of BFCS or BFIP. The total loss functions of these two DETD variants are

$$\mathcal{L}_{DETD-BFCS} = \mathcal{L}_{DETR} + \lambda_{BFCS}\mathcal{S}_{BFCS}, \quad (15)$$

$$\mathcal{L}_{DETD-BFIP} = \mathcal{L}_{DETR} + \lambda_{BFIP}\mathcal{S}_{BFIP}, \quad (16)$$

where λ_{BFCS} and λ_{BFIP} are the corresponding coefficients. As illustrated in Fig. 12, from the perspective of object detection performance, the optimal results for both variants (i.e., 52.97% and 53.26%) are lower than DETD (53.65%). Both approaches show effectiveness in inducing feature disentanglement, with the BFCS penalty being the most optimal. When applying the same optimization strategy to both variants (i.e., the cross-operation on DETD-BFCS and DETD-BFIP variants with their according loss coefficients set to the best), their performance (52.92% and 53.22%) is lower than the baseline DETR (54.17%). Our method DETD-cross outperforms both variants by 1.4%. Therefore, we pursue feature disentanglement by penalizing the feature singular values (i.e., BFSP loss in this paper).

TABLE VII: Results of object detection(AP, %) on COCO dataset. The letter c is the abbreviation of cross.

Method	mAP	AP50	AP75	APS	APM	APL
DETR	35.48	55.39	36.74	12.91	38.20	58.30
DETD	34.82	54.98	35.83	11.58	37.32	57.32
DETD (w/o BFSP)	36.19	56.18	37.51	13.50	38.67	58.81
DETD-c	35.93	55.55	37.57	13.18	38.68	58.08
DETD-c(w/o BFSP)	**36.58**	**56.80**	**38.24**	**14.47**	**39.51**	**59.15**

F. Generalization Performance

We use clipart dataset [66] to verify generalization performance in a domain generalization setting. Clipart is a commonly used dataset for validating model generalization [24]. It shares the same image categories and evaluation standards as VOC dataset. Experimental results, shown in Table IX, indicate that BFSP loss can indeed improve the generalization of DETR, with an improvement of 2.2%. When the influence of BFSP is weakened (i.e., DETD-cross), generalization performance also decreases. However, it still remains 1.1% higher than DETR. Additionally, we must recognize an issue that, while BFSP can enhance generalization, the DETR framework itself still lacks sufficient generalizability.

G. Feature Activation Visualization

Grad-Cam [41] visualization is conducted to intuitively demonstrate the effectiveness of our method. The visualizations are shown in the Fig. 11. In the regression visualization (i.e., the bottom row), compared to DETR, DETD's feature activation maps provide a more detailed depiction of the object's boundaries, enhancing the model's saliency detection capabilities and aiding in more accurate determination of the object coordinates. This also confirms that our method activates more features. Each feature represents a semantic pixel patch when the input format is point-wise. More feature activations enhance the model's semantic learning capabilities,

TABLE VIII: Object detection performance(AP, %) results of DINO and DIDETD models on COCO dataset.

Input Type	Models	mAP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	Epochs	Pretrain Epochs
Square 640	DINO	40.18	59.23	42.42	18.33	44.22	61.09	15	–
	DIDETD	**41.12**	**60.01**	**43.58**	**18.41**	44.89	**62.51**	15	–
	DIDETD (w/o BFSP)	41.05	60.09	43.56	18.40	**45.20**	61.82	15	–
	DIDETD-cross	40.44	59.56	42.70	18.70	44.43	61.58	10	10
	DIDETD-cross (w/o BFSP)	40.38	59.35	42.77	18.37	44.61	60.91	10	10
Square 1024	DINO	44.98	64.02	48.33	24.50	49.43	62.20	15	–
	DIDETD	45.01	63.82	48.34	24.24	49.34	61.87	15	–
	DIDETD (w/o BFSP)	44.81	63.85	47.93	24.12	49.14	62.34	15	–
	DIDETD-cross	**45.42**	64.34	**48.89**	**24.91**	**49.45**	**63.82**	10	10
	DIDETD-cross (w/o BFSP)	45.34	**64.38**	48.60	24.81	49.38	63.59	10	10
Original	DINO	44.57	63.87	47.48	24.56	49.14	61.55	15	–
	DIDETD	44.07	63.09	47.25	24.23	48.72	61.42	15	–
	DIDETD (w/o BFSP)	44.46	63.46	47.72	24.68	48.83	62.02	15	–
	DIDETD-cross	45.21	64.57	48.52	24.52	49.77	63.13	10	10
	DIDETD-cross (w/o BFSP)	**45.47**	**64.76**	**48.85**	**25.30**	**49.85**	**63.40**	10	10

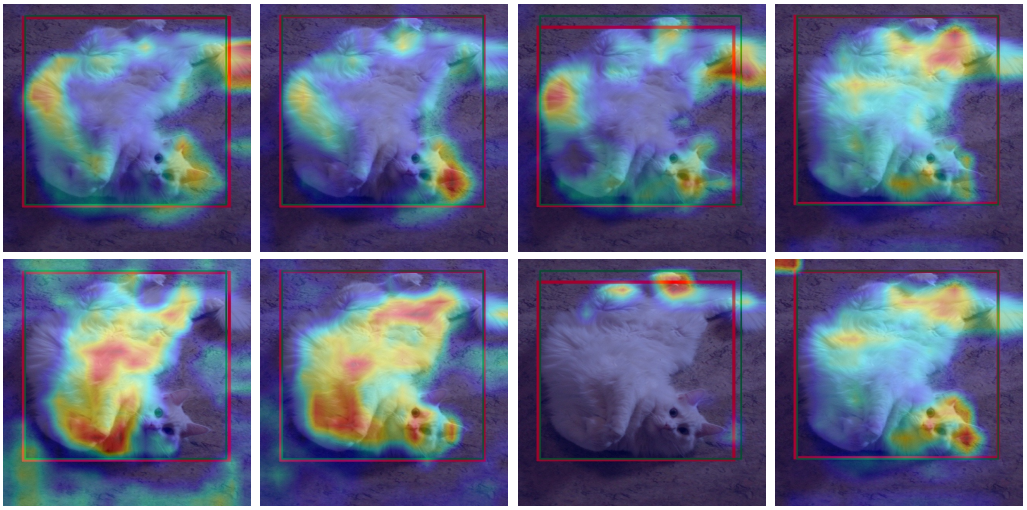
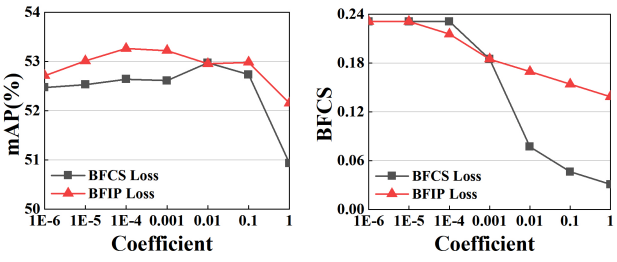


Fig. 11: Grad-Cam analysis of classification and regression functions. The top row is the activation of classification. The bottom row is the activation of regression.

TABLE IX: Generalization results(AP, %). The source domain is VOC dataset. The target domain is Clipart dataset.

Models	mAP	AP_{50}	AP_{75}
DETR	13.15	22.08	13.14
DETD	**15.35**	**24.11**	**16.34**
DETD (w/o BFSP)	14.18	23.26	14.89
DETD-cross	14.24	23.99	14.59
DETD-cross (w/o BFSP)	13.78	22.52	15.10



(a) Performance (b) BFCS value

Fig. 12: The results of the model with only the addition of BFCS and BFIP loss. The X-axis represents the coefficients of the added loss functions. BFCS represents the BFCS value of the joint position.

aligning more closely with human prior knowledge, thereby enhancing explainability.

In an image, a single object often contains multiple semantic pieces of information. For instance, the head of a cat can correspond to one semantic piece of information, while the tail of the cat may correspond to another. When using a single-backbone DETR, coupled features tend to activate in concentrated regions, making it challenging to demonstrate their correspondence with semantic information related to objects, thus reducing interpretability. The proposed method enhances the interpretability of features by producing decoupled features that correspond to activations in more locations associated

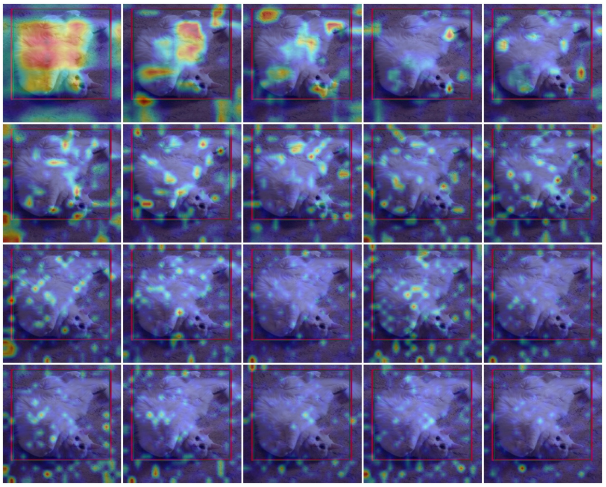


Fig. 13: Feature bases of the DETD model. Feature bases are sorted according to the eigenvalues.

TABLE X: Object detection performance(AP, %) and BFCS value of DETD-cross models trained on various epochs. The learning rate drop operation for each model occurs within the last 20 epochs before training completion.

Total Epochs	Epochs	Pretrain Epochs	mAP	AP50	AP75	BFCS
270	150	120	54.54	77.83	58.83	0.22
310	150	160	54.67	77.54	58.44	0.22
330	150	180	54.57	77.07	58.85	0.22
250	150	100	53.97	77.21	57.91	0.22
250	130	120	54.62	77.32	59.13	0.22
250	100	150	54.44	77.38	59.15	0.22
250	50	200	54.50	77.59	58.44	0.22
290	90	200	54.62	77.67	58.45	0.22
330	130	200	54.72	77.50	59.09	0.22
350	150	200	54.67	77.68	58.62	0.22

with more semantic information. This results in a richer set of semantic information input to the transformer, which, in turn, is conducive to improving object detection performance.

Besides, visualizations also explain that DETR in feature-wise input manner behaves with lower performance due to classification and regression being coupled. This is because this approach fundamentally doesn't treat each pixel patch as a semantic attribute object. Regarding the feature bases shown in Fig.13, we can see that the largest singular value dominates the original feature map. Additionally, we have observed that the smaller singular values tend to focus more on global information, potentially leading to a degradation in performance.

H. Training Epochs Analysis

We conduct experiments to validate the impact of training epochs on object detection performance and BFCS value. Table X presents the object detection performance and BFCS value of the DETD-cross model under various settings of pre-trained epochs and fine-tuned epochs. We first control that the total number of pre-trained and fine-tuned epochs amounts to 250. The optimal performance (54.62%) occurs when the number of pre-trained epochs and fine-tuned epochs is 120 and 130 (abbreviated as 120-130). This is 0.05% lower compared to the 200-150 setting (54.67%). The BFCS value for all DETD-cross configurations is 0.22, confirming the efficacy of the optimization strategy proposed in this paper (i.e., cross) in enhancing object detection performance while maintaining feature disentanglement effects. When the pre-trained epochs and fine-tuned epochs are insufficient, both models experience a decrease in performance, yet still surpass the DETD model. When controlling the pre-trained epochs at 200 varying the number of fine-tuned epochs, and controlling the number of fine-tuned epochs and varying the pre-trained epochs, the model demonstrates similar outcomes. Even with 130 fine-tuned epochs, the model's performance surpasses that of the 200-150 configuration by 0.05%. Therefore, we configure 200-150 as our experimental configuration to ensure robust pre-trained and fine-tuned model convergence.

VI. Conclusion

In this paper, we explore and reveal the intrinsic disentanglement characteristics of the single-backbone object detection models. The defined BFIP and BFCS values demonstrate that features between the extractor and Transformer have the disentanglement property by adding the progressive BFSP loss. The relation between classification and regression grows closer from this aspect. Based on the decomposition method T-SVD and the optimization methods, we build a series of explainability enhanced DETD models based on CNN, Transformer, MLP, Deformable DETR, DINO, and Borderdet, respectively. Experiments demonstrate that our rule-inspired feature disentanglement method not only reduces feature similarity but also leads to improvements in generalization and object detection performance when applied in conjunction with optimization strategies. Furthermore, it improves explainability. There are several problems worthy of future work. First, introduce more optimization methods, such as stable and efficient tensor decomposition methods, to further improve the performance and training efficiency of the model. Second, since feature disentanglement aims to enhance semantical explainability, the correspondence between feature directions and semantics of the real world needs to be clarified, especially for introducing human knowledge. Third, there are still many disentanglement properties that need to be discovered since the BFSP loss changes the singular value distribution of the feature tensor, and it may influence other characteristics of DNNs.

References

[1] Z.-Q. Zhao, P. Zheng, S.-T. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11, pp. 3212–3232, 2019.

[2] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 6000–6010.

[3] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European Conference on Computer Vision*. Springer, 2020, pp. 213–229.

[4] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "Deformable detr: Deformable transformers for end-to-end object detection," *arXiv preprint arXiv:2010.04159*, 2020.

[5] A. Bar, X. Wang, V. Kantorov, C. J. Reed, R. Herzig, G. Chechik, A. Rohrbach, T. Darrell, and A. Globerson, "Detreg: Unsupervised pretraining with region priors for object detection," *arXiv preprint arXiv:2106.04550*, 2021.

[6] S. Atakishiyev, M. Salameh, H. Yao, and R. Goebel, "Explainable artificial intelligence for autonomous driving: A comprehensive overview and field guide for future research directions," 2021.

[7] Y. Zhang, P. Tiňo, A. Leonardis, and K. Tang, "A survey on neural network interpretability," *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2021.

[8] A. Kasirzadeh, "Reasons, values, stakeholders: a philosophical framework for explainable artificial intelligence," *arXiv preprint arXiv:2103.00752*, 2021.

[9] B. Li, P. Qi, B. Liu, S. Di, J. Liu, J. Pei, J. Yi, and B. Zhou, "Trustworthy ai: From principles to practices," 2021.

[10] C. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," *Nature Machine Intelligence*, vol. 1, no. 5, pp. 206–215, 2019.

[11] M. Hosseini and V. Swaminathan, "Adaptive 360 vr video streaming: Divide and conquer," *2016 IEEE International Symposium on Multimedia (ISM)*, pp. 107–110, 2016. [Online]. Available: https://api.semanticscholar.org/CorpusID:9311481

[12] Y. Mei, M. N. Omidvar, X. Li, and X. Yao, "A competitive divide-and-conquer algorithm for unconstrained large-scale black-box optimization," *ACM Transactions on Mathematical Software (TOMS)*, vol. 42, pp. 1 – 24, 2016. [Online]. Available: https://api.semanticscholar.org/CorpusID:4865734

[13] J. Zabczyk, *Mathematical control theory*. Springer, 2020.

[14] R. A. Horn and C. R. Johnson, *Matrix analysis*. Cambridge university press, 2012.

[15] E. A. Posner, K. E. Spier, and A. Vermeule, "Divide and conquer," *Journal of Legal Analysis*, vol. 2, no. 2, pp. 417–471, 2010.

[16] W. ZaiDao, W. Jia-Rui, W. X. Xu, and P. Quan, "A review of disentangled representation learning," *Acta Automatica Sinica*, vol. 48, no. 2, p. 24, 2022.

[17] J. Zaidi, J. Boilard, G. Gagnon, and M.-A. Carbonneau, "Measuring disentanglement: A review of metrics," *arXiv preprint arXiv:2012.09276*, 2020.

[18] Y. Ma, D. Y. Tsao, and H. yeung Shum, "On the principles of parsimony and self-consistency for the emergence of intelligence," *ArXiv*, vol. abs/2207.04630, 2022.

[19] Y. Shen and B. Zhou, "Closed-form factorization of latent semantics in gans," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 1532–1540.

[20] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4401–4410.

[21] R. She, P. Fan, X.-Y. Liu, and X. Wang, "Interpretable generative adversarial networks with exponential function," *IEEE Transactions on Signal Processing*, vol. 69, pp. 3854–3867, 2021.

[22] B. Zhou, D. Bau, A. Oliva, and A. Torralba, "Interpreting deep visual representations via network dissection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, pp. 2131–2145, 2019.

[23] H. Liu, R. Wang, S. Shan, and X. Chen, "What is tabby? interpretable model decisions by learning attribute-based classification criteria," *IEEE transactions on pattern analysis and machine intelligence*, 2019.

[24] A. Wu, R. Liu, Y. Han, L. Zhu, and Y. Yang, "Vector-decomposed disentanglement for domain-invariant object detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 9342–9351.

[25] H. Wang, Q. Wang, H. Zhang, Q. Hu, and W. Zuo, "Crabnet: Fully task-specific feature learning for one-stage object detection," *IEEE Transactions on Image Processing*, vol. 31, pp. 2962–2974, 2022.

[26] B. Schölkopf, F. Locatello, S. Bauer, N. R. Ke, N. Kalchbrenner, A. Goyal, and Y. Bengio, "Toward causal representation learning," *Proceedings of the IEEE*, vol. 109, no. 5, pp. 612–634, 2021.

[27] G. H. Golub and C. Reinsch, "Singular value decomposition and least squares solutions," in *Linear algebra*. Springer, 1971, pp. 134–151.

[28] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Rev.*, vol. 51, pp. 455–500, 2009.

[29] D. Gunning and D. Aha, "Darpa's explainable artificial intelligence (xai) program," *AI Magazine*, vol. 40, no. 2, pp. 44–58, Jun. 2019.

[30] C. Molnar, *Interpretable machine learning*. Lulu. com, 2020.

[31] F.-L. Fan, J. Xiong, M. Li, and G. Wang, "On interpretability of artificial neural networks: A survey," *IEEE Transactions on Radiation and Plasma Medical Sciences*, 2021.

[32] D. Nauck and R. Kruse, "Obtaining interpretable fuzzy classification rules from medical data," *Artificial intelligence in medicine*, vol. 16, no. 2, pp. 149–169, 1999.

[33] H. Sun, "An accurate and interpretable bayesian classification model for prediction of herg liability," *ChemMedChem: Chemistry Enabling Drug Discovery*, vol. 1, no. 3, pp. 315–322, 2006.

[34] W. Ke, C. Wu, X. Fu, C. Gao, and Y. Song, "Interpretable test case recommendation based on knowledge graph," in *2020 IEEE 20th International Conference on Software Quality, Reliability and Security (QRS)*. IEEE, 2020, pp. 489–496.

[35] U. Shalit, F. D. Johansson, and D. A. Sontag, "Estimating individual treatment effect: generalization bounds and algorithms," in *ICML*, 2017.

[36] X. Shao, A. Skryagin, W. Stammer, P. Schramowski, and K. Kersting, "Right for better reasons: Training differentiable models by constraining their influence function," in *Proceedings of Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI)*, 2021.

[37] R. Zellers, Y. Bisk, A. Farhadi, and Y. Choi, "From recognition to cognition: Visual commonsense reasoning," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6713–6724, 2019.

[38] X. Cheng, Z. Rao, Y. Chen, and Q. Zhang, "Explaining knowledge distillation by quantifying the knowledge," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 12 925–12 935.

[39] J. Chen, L. Song, M. Wainwright, and M. Jordan, "Learning to explain: An information-theoretic perspective on model interpretation," in *International Conference on Machine Learning(ICML)*. PMLR, 2018, pp. 883–892.

[40] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," in *In Workshop at International Conference on Learning Representations*. Citeseer, 2014.

[41] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 618–626.

[42] J. Ren, D. Zhang, Y. Wang, L. Chen, Z. Zhou, Y. Chen, X. Cheng, X. Wang, M. Zhou, J. Shi, and Q. Zhang, "A unified game-theoretic interpretation of adversarial robustness," *ArXiv*, vol. abs/2111.03536, 2021.

[43] D. Wang, X. Cui, X. Chen, R. K. Ward, and Z. J. Wang, "Interpreting bottom-up decision-making of cnns via hierarchical inference," *IEEE Transactions on Image Processing*, vol. 30, pp. 6701–6714, 2021.

[44] K. Han, Y. Wang, H. Chen, X. Chen, J. Guo, Z. Liu, Y. Tang, A. Xiao, C. Xu, Y. Xu *et al.*, "A survey on visual transformer," *arXiv preprint arXiv:2012.12556*, 2020.

[45] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," *ArXiv*, vol. abs/2010.11929, 2021.

[46] A. Binder, G. Montavon, S. Lapuschkin, K.-R. Müller, and W. Samek, "Layer-wise relevance propagation for neural networks with local renormalization layers," in *ICANN*, 2016.

[47] E. Voita, D. Talbot, F. Moiseev, R. Sennrich, and I. Titov, "Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned," *arXiv preprint arXiv:1905.09418*, 2019.

[48] S. Abnar and W. Zuidema, "Quantifying attention flow in transformers," *arXiv preprint arXiv:2005.00928*, 2020.

[49] H. Chefer, S. Gur, and L. Wolf, "Transformer interpretability beyond attention visualization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 782–791.

[50] I. O. Tolstikhin, N. Houlsby, A. Kolesnikov, L. Beyer, X. Zhai, T. Unterthiner, J. Yung, D. Keysers, J. Uszkoreit, M. Lucic, and A. Dosovitskiy, "Mlp-mixer: An all-mlp architecture for vision," in *NeurIPS*, 2021.

[51] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[52] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.

[53] S. Zhang, C. Chi, Y. Yao, Z. Lei, and S. Z. Li, "Bridging the gap between anchor-based and anchor-free detection via adaptive training sample

selection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 9759–9768.
[54] P. Sun, Y. Jiang, E. Xie, W. Shao, Z. Yuan, C. Wang, and P. Luo, "What makes for end-to-end object detection?" in *International Conference on Machine Learning(ICML)*. PMLR, 2021, pp. 9934–9944.
[55] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.
[56] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
[57] M.-M. Cheng, P.-T. Jiang, L.-H. Han, L. Wang, and P. Torr, "Deeply explain cnn via hierarchical decomposition," *International Journal of Computer Vision*, vol. 131, no. 5, pp. 1091–1105, 2023.
[58] X. Chen, S. Wang, M. Long, and J. Wang, "Transferability vs. discriminability: Batch spectral penalization for adversarial domain adaptation," in *International Conference on Machine Learning(ICML)*, 2019, pp. 1081–1090.
[59] Q. Wang, J. Xie, W. Zuo, L. Zhang, and P. Li, "Deep cnns meet global covariance pooling: Better representation and generalization," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 8, pp. 2582–2597, 2020.
[60] J. Lu, "Matrix decomposition and applications," *arXiv preprint arXiv:2201.00145*, 2022.
[61] D. Lian, Z. Yu, X. Sun, and S. Gao, "As-mlp: An axial shifted mlp architecture for vision," *ArXiv*, vol. abs/2107.08391, 2021.
[62] H. Zhang, F. Li, S. Liu, L. Zhang, H. Su, J. Zhu, L. M. Ni, and H.-Y. Shum, "Dino: Detr with improved denoising anchor boxes for end-to-end object detection," *arXiv preprint arXiv:2203.03605*, 2022.
[63] H. Qiu, Y. Ma, Z. Li, S. Liu, and J. Sun, "Borderdet: Border feature for dense object detection," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*. Springer, 2020, pp. 549–564.
[64] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International journal of computer vision*, vol. 88, no. 2, pp. 303–338, 2010.
[65] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.
[66] N. Inoue, R. Furuta, T. Yamasaki, and K. Aizawa, "Cross-domain weakly-supervised object detection through progressive domain adaptation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 5001–5009.

Wenlong Yu received the B.S. degree in energy and power engineering, and the M.S. degree in mechanical and control engineering from Shandong University, Jinan, China, in 2017 and 2020, respectively. He is currently working toward the Ph.D. degree in computer science and techniques at the College of Intelligence and Computing, Tianjin University, Tianjin, China. His research interests include interpretability of AI, computer vision, and intelligent unmanned systems.

Ruonan Liu (M'19) received the B.S., M.S. and PhD degrees from Xi'an Jiaotong University, Xi'an, China, in 2013, 2015 and 2019, respectively. She was a postdoctoral researcher with the School of Computer Science, Carnegie Mellon University in 2019. She now is an associate professor in the College of Intelligence and Computing, Tianjin University, Tianjin, China. Her research interests include artificial intelligence and machine vision systems.

Dongyue Chen received the B.S. and M.S. degrees from Southwest Jiaotong University, Chengdu, China, and Ph.D. degrees from Tianjin University, Tianjin, China, in 2015, 2018, and 2023, respectively. She is currently a postdoctoral researcher with the College of Intelligence and Computing, Tianjin University, Tianjin, China. Her research interests include artificial intelligence, incremental learning, trustworthy deep learning, and fault diagnosis of mechanical systems.

Qinghua Hu (SM'13) received the B.S., M.S., and Ph.D. degrees from the Harbin Institute of Technology, Harbin, China, in 1999, 2002, and 2008, respectively. After that, he joined the Department of Computing at the Hong Kong Polytechnical University as a postdoctoral fellow. He is currently a Chair Professor at the College of Intelligence and Computing, Tianjin University, Tianjin, China. His research interest is focused on uncertainty modeling, multi-modality learning, and incremental learning, funded by the National Natural Science Foundation of China and The National Key Research and Development Program of China. He has published more than 300 peer-reviewed papers in IEEE TKDE, IEEE TPAMI, IEEE TNNLS, etc. He is an Associate Editor of the IEEE Transactions on Fuzzy Systems, ACTA AUTOMATICA SINICA, and ACTA ELECTRONICA SINICA.