DeViL: Decoding Vision features into Language

Meghal Dani*[1], Isabel Rio-Torto*[3,4], Stephan Alaniz[1], and Zeynep Akata[1,2]

[1] University of Tübingen
[2] MPI for Intelligent Systems
[3] Faculdade de Ciências da Universidade do Porto
[4] INESC TEC
* Equal contribution
{meghal.dani, stephan.alaniz, zeynep.akata}@uni-tuebingen.de
{isabel.riotorto}@inesctec.pt

Abstract. Post-hoc explanation methods have often been criticised for abstracting away the decision-making process of deep neural networks. In this work, we would like to provide natural language descriptions for what different layers of a vision backbone have learned. Our *DeViL* method generates textual descriptions of visual features at different layers of the network as well as highlights the attribution locations of learned concepts. We train a transformer network to translate individual image features of any vision layer into a prompt that a separate off-the-shelf language model decodes into natural language. By employing dropout both per-layer and per-spatial-location, our model can generalize training on image-text pairs to generate localized explanations. As it uses a pre-trained language model, our approach is fast to train and can be applied to any vision backbone. Moreover, *DeViL* can create open-vocabulary attribution maps corresponding to words or phrases even outside the training scope of the vision model. We demonstrate that *DeViL* generates textual descriptions relevant to the image content on CC3M, surpassing previous lightweight captioning models and attribution maps, uncovering the learned concepts of the vision backbone. Further, we analyze fine-grained descriptions of layers as well as specific spatial locations and show that *DeViL* outperforms the current state-of-the-art on the neuron-wise descriptions of the MILANNOTATIONS dataset.

Keywords: Explainable AI · Vision-Language Models · Natural Language Explanations · Open-vocabulary Saliency

1 Introduction

Despite the success of deep vision models, the lack of understanding of how they arrive at their predictions inhibits their widespread adoption in safety-critical fields such as medicine. To better understand arbitrary vision models, post-hoc explanation methods play an important role because they allow inspection a posteriori without any modifications to the architecture or loss function.

One popular strategy to improve the interpretability of vision models are visual attribution maps. Activation-based methods [46,34,36,43,9] take into account the features a vision model produced at a given layer for visualizing the spatial attribution with respect to the network output. However, they are not always reliable [1] and other explanation modalities exist. Natural language explanations (NLEs) have been proposed in the context of vision-language models to extend the language output for a given task with a fitting explanation [33], such as for a VQA task. A less explored direction is directly explaining internal features or neurons of neural networks through natural language [15]. Such explanations would allow a wider reach of application, especially where users are not expected to have expert knowledge and can more easily interpret textual explanations than saliency maps.

In this work, we propose *DeViL* (Decoding Vision features into Language) that can explain features of a vision model through natural language. By employing a general purpose language model, we inherit rich generalization properties, allowing us to additionally perform open-vocabulary saliency analysis. *DeViL* combines providing textual explanations of these visual features with highlighting visual attributions spatially to extract diverse explanations that complement each other. Thus, enabling non-experts to comprehend the network's decision-making process, diagnose potential issues in the model's performance, and enhance user trust and accountability.

With *DeViL*, our key contributions are: i) generating spatial and layer-wise natural language explanations for any off-the-shelf vision model, ii) open-vocabulary saliency attributions coherent with textual explanations, and iii) showing how dropout can be used to generalize from image-text pairs on a global level to fine-grained network inspections. To the best of our knowledge, this is the first work to combine these capabilities into a single model while requiring short training times and using abundantly available captioning data instead of explanation specific datasets.

2 Related Work

Inherently interpretable models. Apart from post-hoc explanation methods, inherently interpretable models try to modify the network architecture or loss function to make the model more interpretable. For instance, induced alignment pressure on the weights during optimisation, either in a linear [10,7,5,6] or non-linear manner [22,49], has been shown to produce more interpretable models. While these methods produce more interpretable models, they require adaptation and re-training of existing models and cannot always recover the original task performance. With the increasing size of models, this is not always a scalable solution. Post-hoc methods trade-off the guaranteed faithfulness of the explanation with broader applicability.

Post-hoc saliency. Saliency attribution is most commonly produced through perturbations [24,29], gradients [38], activations [18,43,46] or their combination [34,9]. Class Activation Mapping (CAM) [46] and its successors, such

as Grad-CAM [34] and Grad-CAM++ [9], are one of the most popular attribution methods that make use of network activations, similar to our approach, but additionally use the gradients of the target class. We set ourselves apart from these methods by introducing open-vocabulary saliency maps and also providing directly related natural language explanations.

Natural Language Explanations. Another explanation alternative are NLEs, which provide textual descriptions accompanying the prediction. The approaches are usually divided into predict-explain and explain-predict paradigms: in the first, a language model is finetuned with features and predictions from a pre-trained vision model [17,26], while in the latter a single model is jointly trained to explain and predict [25,33,30]. While these provide a single NLE for the whole image, Bau et al. [4] attempts to label individual highest activated neurons in a network by correlating them with pixel-wise annotations. Building upon that, Hernandez et al. [15] collect the MILANNOTATIONS datasets where humans labeled sets of images that showed high activations for individual neurons of different network architectures. By training a vision-language model on this data, individual neurons of different layers can be described in natural language. In contrast, *DeViL* can generate both types of NLEs (global and local), without requiring a dataset of human annotated descriptions per neuron or layer. Nonetheless, we show that when our model is trained on MILANNOTATIONS, we obtain a higher correspondence with human descriptions.

Captioning Models. Image captioning models [27,31,40,48,16], on the other hand, use large language models (LLMs) to generate free-form captions for the input image. State-of-the-art in captioning is obtained by large-scale pre-training and subsequently finetuning on target datasets [48,16]. On the other hand, lightweight models such as ClipCap [27] perform competitively by relying on pre-trained vision and language models requiring significantly fewer GPU hours to train. *DeViL* is similar to ClipCap in that both models use a translation network between image features and the language model prompt. While ClipCap is capable of generating high-quality captions for images, it cannot be directly used to interpret a model's internal representations. In contrast, by extracting features from multiple layers of the vision model and incorporating dropout, *DeViL* generalizes to more fine-grained descriptions and improves all captioning metrics on CC3M [35].

3 *DeViL* Model

Given a pre-trained vision model g, our goal is to decode the features of several of its L layers into natural language describing what these features encode. These descriptions are specific to the network's activations corresponding to an input image. Let $g_l(x)$ denote the feature map of the lth-layer of g for input image x with spatial size $H_l \times W_l$. We propose *DeViL*, a network that is trained to decode an individual feature vector $g_l(x)_{i,j}$ of an arbitrary spatial location i, j into a natural language description. Since *DeViL* is trained on top of features from a frozen vision model g, we view it as a post-hoc explanation method,

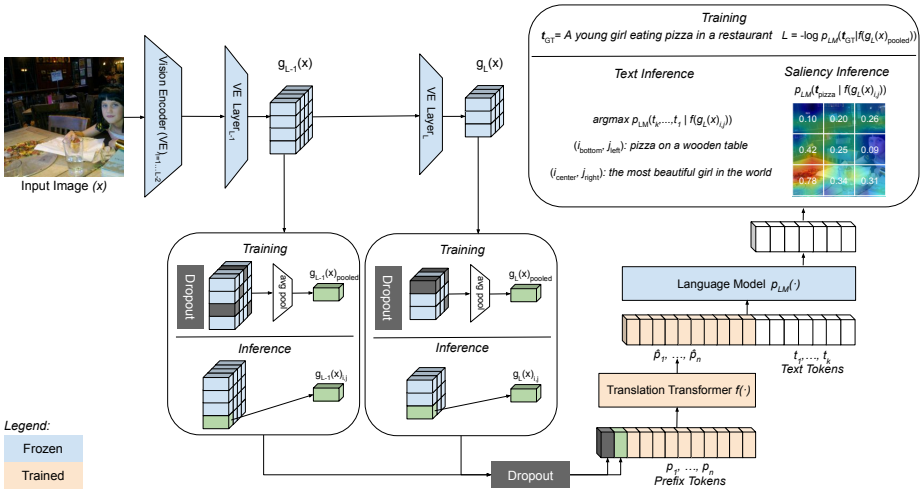

Fig. 1: *DeViL* architecture diagram: Given an input image x and a pre-trained vision encoder with multiple layers (L), we concatenate the average pooling embedding ($g_L(x)_{pooled}$) to learnable prefix tokens (size=10; as in [27]). A translation transformer f then projects prefix tokens $\hat{p}_1, \ldots, \hat{p}_n$ used to condition the language model p_{LM} to get the captions for the entire layer. During inference, we can select individual feature $g_L(x)_{i,j}$ for any arbitrary location i, j of the layer and generate its respective NLE, together with open-vocabulary saliency.

that is able to reason about which concepts have been learned by the vision model retrospectively. Figure 1 depicts an overview of the network architecture of *DeViL*, which we describe in more detail in the following sub-sections.

3.1 Translating vision features into language

Our *DeViL* method is designed to be a lightweight model such that it can be trained and applied to existing vision backbones without requiring a long training time. Instead of training a language model, we make use of a pre-trained language model with frozen weights to generate the natural language descriptions for vision features. Specifically, the language model $p_{\text{LM}}(t_k|t_1, \ldots, t_{k-1})$ is able to predict the next token t from its tokenizer's vocabulary given a sequence of previous tokens as context. As shown in recent works [48,27], the text generation of language models can be guided by conditioning the model on learned prefix tokens instead of finetuning its weights.

We follow this approach and train only a translation network f that takes image features $g_l(x)_{i,j}$ and produces n prefix tokens $\hat{\boldsymbol{p}} = \{\hat{p}_i\}_{i=1}^{n}$ that are used to condition the language model $p_{\text{LM}}(t_k|\hat{p}_1, \ldots, \hat{p}_n, t_1, \ldots, t_{k-1})$ for generating a description. The translation network uses a standard Transformer architecture. As input, we concatenate the sequence of image features $g_l(x)_{i,j}$ with a set of

trained parameters for the initial prefix tokens $\boldsymbol{p}$. From the output, we only keep the transformed prefix tokens $\hat{\boldsymbol{p}}$ to pass on to the language model as prompt in a causal language generation setting.

3.2 Learning to decode individual vision features

Obtaining fine-grained data on textual descriptions for neurons or activations of the neural networks as done in [15] is costly at scale. Hence, we resort to more commonly available text-image pairs in the form of a captioning dataset to achieve the same goal. Since these image-text pairs relate the global image content to a single sentence, we describe in the following how we can train on these global relations, but evaluate our model on more fine-grained feature locations across the layers of the vision model.

Given a vision model g, we first specify the layers we would like to decode. For every forward pass of g, we extract the features $g_l(x)$ for every layer l in our set of explained layers. At training time, we perform 2D global average pooling over the spatial locations at each feature layer to obtain a single feature vector per layer $g_l(x)_{\text{pooled}}$. After applying a linear projection onto the same dimensionality and adding a positional embedding, the feature vectors are passed to the translation network f as a sequence. Depending on how many layers are being explained, the sequence length for f varies, i.e., it increases by one for each layer that is explained. At inference time, instead of pooling the visual features, we can select a specific location we would like to explain and only pass that particular feature vector $g_l(x)_{i,j}$ to f.

To train our translation network, we optimize the standard causal language modelling task of predicting the next token given previous tokens. In our context, the language model is conditioned on the image features through the translation network. Thus, our loss is $\mathcal{L} = \log p_{\text{LM}}(\boldsymbol{t}|f(g(x))) = \sum_k \log p_{\text{LM}}(t_k|\hat{\boldsymbol{p}}, \{t_i\}_{i=1}^{k-1})$ with trainable parameters only in the translation network f. Once trained, *DeViL* can decode vision features into natural language by conditioning the language model on a vision feature vector and generating a sentence. In practice, we greedily choose the most probable next word with $\arg\max$.

3.3 Generalization through dropout

The global average pooling at test time is required because we do not have more fine-grained data to train on. However, it creates a discrepancy between training and inference time. To overcome this issue, we introduce two dropout operations. Firstly, we randomly dropout spatial locations i, j of $g_l(x)$ before applying average pooling to obtain $g_l(x)_{\text{pooled}}$. As a result, our translation network observes a larger space of image features that better cover the full distribution of a vision layer's features. Secondly, we randomly subselect the layers from which the translation network f obtains the pooled features. This way, the translation network sometimes receives features only from individual layers during training time. This is crucial because, at inference time, we would like to decode a specific

feature of an individual vision layer so this dropout ensures this input configuration is seen during training. Moreover, it allows to train a single network per vision model instead of one per layer.

In contrast to dropout as proposed by [37], we do not remove features element-wise but always remove a full vector to simulate a spatial location or full layer being missing from the input. Hence, we also do not require to perform any rescaling of the features but rather mask the complete vectors from subsequent operations.

3.4 Open-vocabulary saliency with *DeViL*

DeViL can be used to obtain the probability of a given word or a phrase conditioned on vision features at layer l and location i, j by evaluating $p_{\text{LM}}(\boldsymbol{t}_{\text{query}}|f(g_l(x)_{i,j}))$. When passing overall features from a layer of interest, we obtain a likelihood of the query for every spatial location that we can visualize as a saliency map. By using a general purpose language model, there are no constraints on the query such that we can obtain saliency maps on concepts that lie outside the original training scope of the vision model. This is useful for seeing whether the vision features encode information about related concepts due to their correlation with training data or obtained as a side-effect.

4 Experiments

We evaluate *DeViL* on both its natural language and saliency generation capabilities. *DeViL* is trained on the CC3M [35] dataset. While our goal is to generalize to more fine-grained descriptions of vision features, *DeViL* can still be used as a captioning model. Thus, we start by evaluating image captioning on CC3M, before discussing explanations of vision features obtained through *DeViL*. For fine-grained analysis, we report both qualitative as well as quantitative results on fine-grained neuron descriptions by training and evaluating on the MILAN-NOTATIONS dataset [15]. Lastly, we evaluate both NLEs and saliency obtained through *DeViL* across different layers to show its generalization capabilities. As we focus on explaining the vision backbone instead of producing captions, all images used for qualitative analysis come from sources outside CC3M, such as ImageNet [11], Places365 [47], and COCO [21]. Details about the *DeViL* architecture and training details can be found in the supplementary.

4.1 Evaluating feature descriptions through image captioning

Dataset. We use the official train-validation split of CC3M [35], consisting of 3M image-caption pairs collected from the web. We chose this dataset because it is sufficiently large to cover a large variety in both the vision and language modalities. Its successor CC12M [8] is less focused on high conceptual relevance, and more noisy in caption quality, making CC3M more suitable for tasks that require strong semantic alignment between text and image.

Method	Vision Backbone	LM	B4 ↑	M ↑	RL ↑	C ↑	S ↑	#Params (M) ↓
DeViL	IN-ResNet50	OPT	5.851	9.572	23.92	65.73	15.23	88
	CLIP-ResNet50	OPT	6.770	10.68	25.90	78.41	17.38	88
	CLIP-ViT	OPT	**7.506**	**11.22**	**26.82**	**86.29**	**18.37**	88
	CLIP-ViT	GPT2	6.349	10.55	25.70	76.55	17.81	40
ClipCap [27]	CLIP-ViT	GPT2	-	-	25.12	71.82	16.07	43
VLP [48]			-	-	24.35	77.57	16.59	115
LEMON [16]			**10.1**	**12.1**	-	**104.4**	**19.0**	196.7

Table 1: Image captioning results on CC3M [35] with different pre-trained vision backbones and language models, and a comparison with state-of-the-art captioning models, either fully [48,16] or partially-finetuned [27]. We report standard captioning metrics, where higher is better. IN-ResNet50: ImageNet [11] pre-trained ResNet50 [14]. CLIP-ResNet50/ViT: ResNet50/ViT versions of the CLIP [31] vision encoder trained on the CLIP dataset.

Baselines. Although our goal is to translate latent representations of pre-trained vision models into language, *DeViL* can still be used to obtain full image descriptions. Hence, we evaluate *DeViL* generated sentences with standard captioning metrics and compare against captioning methods [27,48,16]. ClipCap [27] is a lightweight model that combines the CLIP vision model with a pre-trained language model. Similar to our approach, ClipCap only trains a translation network to keep the training cost low. Both UnifiedVLP [48] and LEMON [16] are large-scale models pre-trained on several datasets not limited to captioning and subsequently finetuned. Thus, they surpass lightweight models such as ClipCap and *DeViL*, but require a lot of resources to train.

Results. We present our results on common language-based metrics: BLEU@4 [28], METEOR [12], ROUGE-L [20], CIDEr [42], and SPICE [3]. We consider ResNet50 [14] trained on ImageNet and CLIP [31] in both its ResNet50 and ViT variants as our vision backbones.

We report image captioning results in Table 1. Between vision backbones, we observe that CLIP-ViT performs better than its ResNet50 counterpart and ResNet50 trained on ImageNet, which is not surprising given CLIP's contrastive vision-language pre-training. Since the vision encoder of CLIP has in the past depicted strong zero-shot capabilities on a variety of tasks, we would expect it also to have a large coverage of visual concepts when we explain their features.

When using the CLIP-ViT backbone, we further ablate the relevance of the pre-trained language model. We make use of OPT-125M [44] and GPT2 [32] in our pipeline. With CLIP-ViT-B-32 as the vision backbone and OPT as the language model, we obtain our best scores surpassing ClipCap on all by a big margin, e.g. a CIDEr of 86.29 vs. 71.82. Even when using GPT2 and the same translation network architecture as ClipCap, we perform better across the board while using fewer parameters (40M vs. 43M). This is due to our model changes in using multiple layers of the vision backbone and the introduction of both feature-level and layer-level dropouts. Compared to large-scale captioning models that

Layer	Token Dropout	Feature Dropout	B4 ↑	M ↑	RL ↑	C ↑	S ↑
			0.3038	3.435	11.76	4.729	1.962
single	✓		6.442	10.34	25.14	73.73	16.86
	✓	✓	6.623	10.49	25.50	75.88	17.11
			7.101	11.16	26.62	82.59	**18.54**
all	✓		7.211	11.09	26.70	83.32	18.48
	✓	✓	**7.506**	**11.22**	**26.82**	**86.29**	18.37

Table 2: Ablating our dropout when evaluated using vision features from all layers or only the very last layer (single) of CLIP-ViT.

require more resources, we still perform better than UnifiedVLP on all metrics despite it requiring 1200h of training on a V100 GPU. In comparison, ClipCap reports 72h of training on GTX1080, while *DeViL* requires 24h of training on an A100. LEMON [16] still surpasses our lightweight model as the state-of-the-art model on captioning. Required training resources for LEMON are not reported.

Ablations. We ablate the proposed token and feature dropouts in Table 2 in terms of captioning metrics for the CLIP-ViT model. This table shows 3 different models trained with incremental combinations of both dropouts. While each *DeViL* model is trained to explain multiple vision backbone layers, we evaluate them in two scenarios: using vision features from all layers of the vision backbone it was trained with (all) and when using only the last layer (single). This ablation study confirms that dropout is essential for producing image-relevant captions for individual layers at inference time. Since the model without token dropout has never seen a single layer being passed to the translation network, it performs poorly in this out-of-distribution scenario making CIDEr drop from 75.88 to 4.729. Feature dropout is less essential, but further improves captioning scores and improves generalization. When we compare results on using all layers rather than just using one, we see an improvement in all scores, e.g. CIDEr increases from 75.88 to 86.29. This suggests that complementary information is encoded in the different layers, making it reasonable to assume we can obtain layer-specific explanations even when training on caption data.

Overall, these results show that with our dropout methods, we can train on several layers at once and perform an evaluation on individual layers by themselves, while also avoiding the need to train one model for each layer we might want to explain later on.

4.2 MILAN: explaining local visual features

Since *DeViL* was designed first and foremost for localized feature inspection in natural language, we strive to compare our method more directly on an explainability task, and especially on one such task at the local feature level. Thus, we also train *DeViL* on MILANNOTATIONS [15], a dataset of over 50k human descriptions of sets of image regions obtained from the top neuron activating

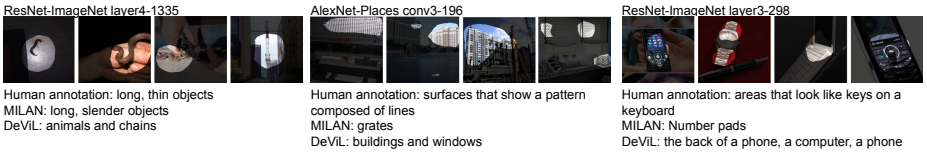

Fig. 2: Qualitative results on MILANNOTATIONS [15] and comparison with the MILAN [15] model.

images of several known models like ResNet [14] or AlexNet [19]. For each base model, the authors collect descriptions for the top 15 activated images of each neuron in the network. Each image is masked by its corresponding activation mask, highlighting only the regions for which the corresponding neuron fired.

We compare with ClipCap and the MILAN model [15] trained on MILANNOTATIONS [15]. Although our model was designed to work with layer-wise feature maps for a single image and not at the neuron level, we adapt *DeViL* by pooling over the 15 masked images given for each neuron.

Method	B4 ↑	M ↑	RL ↑	C ↑	S ↑	BS ↑
MILAN [15]	-	-	-	-	-	0.362
ClipCap [27]	3.99	9.62	27.0	25.1	10.8	0.381
DeViL	**6.28**	**11.3**	**30.6**	**33.7**	**13.3**	**0.382**

Table 3: Evaluating MILAN, ClipCap and *DeViL* on MILANNOTATIONS.

We report the NLP metrics including BERTScore [45] results in Table 3. The results are averaged over several generalization experiments proposed by [15]. We report a complete comparison in terms of BERTScore of all 13 experiments in the supplementary.

The average BERTScore across scenarios is 0.362, 0.381 and 0.382 for MILAN, ClipCap and *DeViL*, respectively. The margins between all models are small as BERTScore is very sensitive to small differences that can still indicate a reliable ranking [43]. Considering all other language metrics, the difference between ClipCap and *DeViL* is more pronounced and *DeViL* outperforms ClipCap consistently. A qualitative comparison with MILAN [15] can be seen in Figure 2. We refer the reader to examples like "animals and chains" and "building and windows". Both quantitative and qualitative results validate *DeViL's* generalization ability and its primary intended goal: to faithfully decode localized vision features into natural language.

4.3 Diverse layer-wise explanations of vision features

Deep neural networks learn to extract meaningful patterns from input data by progressively building up a hierarchy of features. The lower layers tend to detect simple patterns like edges and curves, while the higher layers learn to recognize more complex and abstract concepts [39,43,2,14]. We verify a similar trend in the descriptions generated by *DeViL*. We generate descriptions for each spatial location of the feature map at layer l and produce saliency maps to measure the spatial support of the main subject in the sentence generated at a single location.

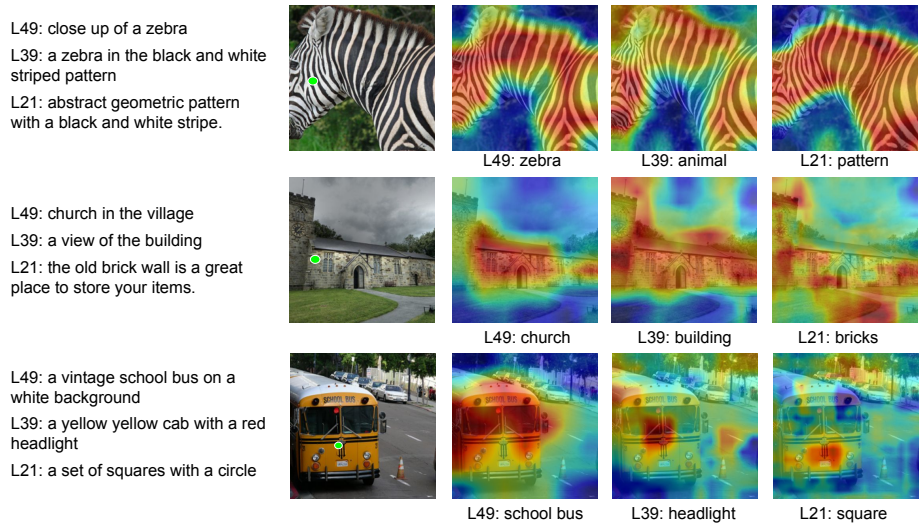

Fig. 3: Generated descriptions at different layers of the CLIP-ResNet50 backbone (L49, L39 & L21) for the location marked with a green dot and corresponding saliency maps for the relevant words in them.

Our model assigns a probability score to a textual query being generated based on its relevance to the visual features. This score can then be visualized as a heatmap, with higher scores corresponding to areas of the image where the vision model encoded the textual concept.

Qualitative. Figure 3 corresponds to the output generated by the CLIP-ResNet50 model for layers 21, 39, and 49. The green dot in the image shows the location for which the description has been generated. The generated descriptions showcase the aforementioned hierarchy, with lower layers mentioning lower level features like colors (e.g. "red headlight") and shapes (e.g. "squares with a circle") and higher layers mentioning objects (e.g. progression from "building" to "church"). The saliency maps also validate the spatial location of these words, e.g. higher saliency scores for "zebra" at L49, "animal" at L39, and "pattern" at L21.

From simple to rich language descriptions. To quantify how well *DeViL* captures the differences of what specific vision layers encode, we analyze the generated text descriptions across layers. Specifically, we obtain a single text description per layer by averaging the vision features for each layer and decode them individually with *DeViL*. In Table 4, we take a look at the properties of the language generated by *DeViL* on CC3M. As measured by the CIDEr score, we observe that the text similarity to human captions increases with later layers as more semantically meaningful embeddings are produced by the vision model. We also see that the language shifts from using many adjectives in earlier layers to more verbs and a more comprehensive vocabulary in higher layers. This

indicates a shift from describing simple concepts with a limited set of attributes to richer descriptions that also describe relations and actions between objects in the image. In Figure 4, we show the saliency for the same word across different layers of CLIP-ResNet50. The top layer encodes "Zebra", but not "Pattern", and conversely for a lower layer. These insights can help us better understand intermediate features of a pre-trained model and assist in choosing the right features for a downstream task.

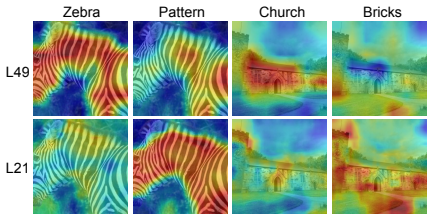

Fig. 4: Semantics are encoded in different layers, CLIP-ResNet50 model.

Model	Layer	Adj (%)	Verbs (%)	#Uniq. Words	CIDEr ↑
CLIP-ViT	9	6.37	6.30	4790	61.6
CLIP-ViT	6	7.60	5.78	4115	44.1
CLIP-ViT	3	8.04	5.24	2554	20.2
CLIP-ResNet50	49	6.30	6.66	5228	71.2
CLIP-ResNet50	39	6.69	6.43	4395	54.1
CLIP-ResNet50	21	6.89	5.90	3374	34.6

Table 4: Per-layer statistics of adjectives, verbs, # of unique words.

Quantifying spatial and layer coherence. We want to further analyze how well the spatially localized descriptions generated by *DeViL* capture the content of the underlying image patches. Intuitively, earlier layers encode local information of a smaller underlying patch, while later layers can encode more global information. To test this, we create center crops of sizes 32x32, 64x64, and 128x128 along the full image size of 224x224 and compare their similarity to *DeViL* descriptions of the vision feature closest to the center in each layer's feature map. We use CLIP-ResNet50 to embed the cropped images as well as *DeViL* descriptions and compare their similarities. Figure 6 plots which crop size is most similar with the generated text. We observe that smaller patches have much higher similarity with earlier layer descriptions, and conversely bigger patches with later layers. Descriptions of L21 best match 64x64 patches and this distribution shift progressively to L49 fitting best to the full image content. These results validate that *DeViL* exposes what the vision model encodes both in lower layers describing the content of local patches and global concepts in higher layers.

4.4 Inspecting different vision models through saliency

Since *DeViL* incorporates an LLM, it has the ability to produce saliency maps for any word, not being limited by the closed-set of classes of a given dataset. Figure 5 presents a comparison of different vision backbones in terms of saliency maps for different words. The different models can distinguish "cat" and "dog", but also identify both as "animal". Interestingly, since CLIP-based models are trained with not only images but also text, they can identify the written word "pizza" on the bottom-most image, while an ImageNet trained ResNet50 cannot.

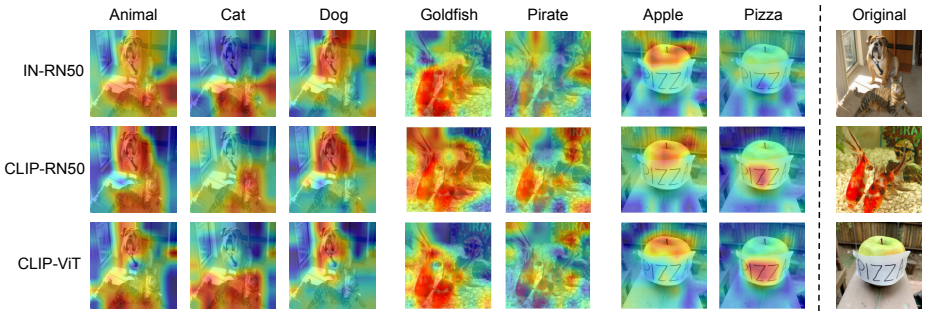

Fig. 5: Open-vocabulary saliency maps for different backbones with the final layer. For CLIP-ViT we use the penultimate layer.

Similarly, the word "pirate", written on a stone, is also identified by CLIP-based models for the aquarium image. These failure cases of the CLIP model have been discovered independently (i.e. typography attack [13]) and the explanations of our *DeViL* model expose these as well, highlighting its faithfulness to the vision model. Obtaining saliency maps for open-vocabulary words that did not appear in a model's training data such as the supercategory "animal" or the class "pirate" is a novel contribution of our method. It also allows a more direct comparison with models of different pre-training tasks such as CLIP, which is not possible with methods that rely on the model's output to be the same for direct comparisons. More qualitative results can be found in the supplementary material.

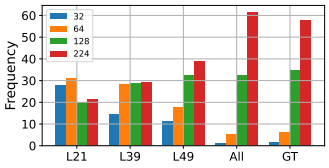

Fig. 6: Frequency of most similar patch size for a given *DeViL* description (L21, L39, L49, All) and human caption (GT) using CLIP-ResNet50.

Layer	Metric	Grad-CAM [34]	*DeViL* (CC3M)
L49	Del.	0.342	0.466
	Ins.	0.763	0.697
L39	Del.	0.465	0.494
	Ins.	0.677	0.661
L21	Del.	0.405	0.484
	Ins.	0.593	0.635

Table 5: Deletion (↓) and insertion (↑) comparison between Grad-CAM [34] and *DeViL* ResNet50 at different layers (L49, L39, and L21).

We also compare our saliency maps quantitatively with Grad-CAM [34], which proposes interpreting a neural network via the gradients for a target class flowing through a given layer. Results are presented in Table 5 for 3 layers of the ResNet50 model (L49, L39, and L11). We report commonly used deletion (Del.) and insertion (Ins.) metrics [29]. The Del.(Ins.) metric measures the drop(rise) in the probability of a class as relevant pixels given by the saliency map are gradually removed(inserted) from the image. Thus, lower(higher) is better for Del.(Ins.). Although we are unable to outperform Grad-CAM, the latter uses

gradients, which provide more information than just activations alone. Furthermore, Grad-CAM is specialized in the network's output classes, while *DeViL* can perform open-vocabulary saliency. An interesting future research direction is to improve and align open-vocabulary saliency with existing approaches.

5 Conclusion

The ability to interpret deep learning models' predictions is crucial for their effective deployment in real-world applications. The visualizations of intermediate feature maps have been shown to be a promising approach for improving model interpretability, but understanding these feature maps often requires further analysis. Our proposed *DeViL* approach for explaining feature maps through natural language is unique as it generates textual descriptions for individual features, and can also produce open-vocabulary saliency attributions. We evaluate the efficacy of our model's language generations on CC3M and MILANNOTATIONS, outperforming competing models, and show extensive qualitative results validating that our open-vocabulary saliency exposes which concepts are understood by each of the model's layers.

6 Acknowledgments

This work was supported by the ERC (853489-DEXIM), DFG (2064/1 – project number 390727645), BMBF (Tübingen AI Center, FKZ: 01IS18039A), FCT (under PhD grant 2020.07034.BD), Carl Zeiss Stiftung and Else Kröner-Fresenius-Stiftung (EKFS). The authors thank the IMPRS-IS for supporting Meghal Dani.

References

1. Adebayo, J., Gilmer, J., Muelly, M., Goodfellow, I.J., Hardt, M., Kim, B.: Sanity checks for saliency maps. In: NeurIPS (2018)
2. Agrawal, P., Carreira, J., Malik, J.: Learning to see by moving. In: ICCV (2015)
3. Anderson, P., Fernando, B., Johnson, M., Gould, S.: Spice: Semantic propositional image caption evaluation. In: ECCV (2016)
4. Bau, D., Zhou, B., Khosla, A., Oliva, A., Torralba, A.: Network dissection: Quantifying interpretability of deep visual representations. In: CVPR (2017)
5. Bohle, M., Fritz, M., Schiele, B.: Convolutional dynamic alignment networks for interpretable classifications. In: CVPR (2021)
6. Böhle, M., Fritz, M., Schiele, B.: B-cos networks: alignment is all we need for interpretability. In: CVPR (2022)
7. Brendel, W., Bethge, M.: Approximating cnns with bag-of-local-features models works surprisingly well on imagenet. ICLR (2019)
8. Changpinyo, S., Sharma, P., Ding, N., Soricut, R.: Conceptual 12m: Pushing web-scale image-text pre-training to recognize long-tail visual concepts. In: CVPR (2021)

9. Chattopadhay, A., Sarkar, A., Howlader, P., Balasubramanian, V.N.: Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. In: WACV (2018)
10. Chen, C., Li, O., Tao, D., Barnett, A., Rudin, C., Su, J.K.: This looks like that: deep learning for interpretable image recognition. NeurIPS (2019)
11. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: CVPR (2009)
12. Denkowski, M., Lavie, A.: Meteor universal: Language specific translation evaluation for any target language. In: WMT (2014)
13. Goh, G., †, N.C., †, C.V., Carter, S., Petrov, M., Schubert, L., Radford, A., Olah, C.: Multimodal neurons in artificial neural networks. Distill (2021)
14. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
15. Hernandez, E., Schwettmann, S., Bau, D., Bagashvili, T., Torralba, A., Andreas, J.: Natural language descriptions of deep visual features. In: ICLR (2022)
16. Hu, X., Gan, Z., Wang, J., Yang, Z., Liu, Z., Lu, Y., Wang, L.: Scaling up vision-language pre-training for image captioning. In: CVPR (2022)
17. Kayser, M., Camburu, O.M., Salewski, L., Emde, C., Do, V., Akata, Z., Lukasiewicz, T.: e-vil: A dataset and benchmark for natural language explanations in vision-language tasks. In: ICCV (2021)
18. Kim, B., Wattenberg, M., Gilmer, J., Cai, C.J., Wexler, J., Viégas, F.B., Sayres, R.: Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV). In: ICML (2018)
19. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NeurIPS (2012)
20. Lin, C.Y.: Rouge: A package for automatic evaluation of summaries. In: WAS (2004)
21. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: ECCV (2014)
22. Liu, W., Liu, Z., Yu, Z., Dai, B., Lin, R., Wang, Y., Rehg, J.M., Song, L.: Decoupled networks. In: CVPR (2018)
23. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: ICLR (2019)
24. Lundberg, S.M., Lee, S.: A unified approach to interpreting model predictions. In: Guyon, I., von Luxburg, U., Bengio, S., Wallach, H.M., Fergus, R., Vishwanathan, S.V.N., Garnett, R. (eds.) NeurIPS (2017)
25. Majumder, B.P., Camburu, O., Lukasiewicz, T., Mcauley, J.: Knowledge-grounded self-rationalization via extractive and natural language explanations. In: ICML (2022)
26. Marasović, A., Bhagavatula, C., sung Park, J., Le Bras, R., Smith, N.A., Choi, Y.: Natural language rationales with full-stack visual reasoning: From pixels to semantic frames to commonsense graphs. In: EMNLP (2020)
27. Mokady, R., Hertz, A., Bermano, A.H.: Clipcap: Clip prefix for image captioning. arXiv preprint arXiv:2111.09734 (2021)
28. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: Bleu: a method for automatic evaluation of machine translation. In: ACL (2002)
29. Petsiuk, V., Das, A., Saenko, K.: RISE: randomized input sampling for explanation of black-box models. In: BMVC (2018)
30. Plüster, B., Ambsdorf, J., Braach, L., Lee, J.H., Wermter, S.: Harnessing the power of multi-task pretraining for ground-truth level natural language explanations. arXiv preprint arXiv:2212.04231 (2022)

31. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: ICML (2021)
32. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language models are unsupervised multitask learners (2019)
33. Sammani, F., Mukherjee, T., Deligiannis, N.: NLX-GPT: A model for natural language explanations in vision and vision-language tasks. In: CVPR (2022)
34. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-cam: Visual explanations from deep networks via gradient-based localization. In: ICCV (2017)
35. Sharma, P., Ding, N., Goodman, S., Soricut, R.: Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In: ACL (2018)
36. Smilkov, D., Thorat, N., Kim, B., Viégas, F., Wattenberg, M.: Smoothgrad: removing noise by adding noise (2017)
37. Srivastava, N., Hinton, G.E., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. JMLR **15** (2014)
38. Sundararajan, M., Taly, A., Yan, Q.: Axiomatic attribution for deep networks. In: ICML (2017)
39. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: CVPR (2015)
40. Tewel, Y., Shalev, Y., Schwartz, I., Wolf, L.: Zerocap: Zero-shot image-to-text generation for visual-semantic arithmetic. In: CVPR (2022)
41. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: NeurIPS (2017)
42. Vedantam, R., Lawrence Zitnick, C., Parikh, D.: Cider: Consensus-based image description evaluation. In: CVPR (2015)
43. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: ECCV (2014)
44. Zhang, S., Roller, S., Goyal, N., Artetxe, M., Chen, M., Chen, S., Dewan, C., Diab, M., Li, X., Lin, X.V., et al.: Opt: Open pre-trained transformer language models. arXiv preprint arXiv:2205.01068 (2022)
45. Zhang, T., Kishore, V., Wu, F., Weinberger, K.Q., Artzi, Y.: Bertscore: Evaluating text generation with bert. In: ICLR (2020)
46. Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., Torralba, A.: Learning deep features for discriminative localization. In: CVPR (2016)
47. Zhou, B., Lapedriza, A., Khosla, A., Oliva, A., Torralba, A.: Places: A 10 million image database for scene recognition. IEEE T-PAMI (2017)
48. Zhou, L., Palangi, H., Zhang, L., Hu, H., Corso, J., Gao, J.: Unified vision-language pre-training for image captioning and vqa. In: AAAI (2020)
49. Zoumpourlis, G., Doumanoglou, A., Vretos, N., Daras, P.: Non-linear convolution filters for cnn-based learning. In: ICCV (2017)