# Materialistic: Selecting Similar Materials in Images – Supplemental Material

PRAFULL SHARMA, MIT, USA and Adobe Research, USA
JULIEN PHILIP, Adobe Research, UK
MICHAEL GHARBI, Adobe Research, US
BILL FREEMAN, MIT, US
FREDO DURAND, MIT, US
VALENTIN DESCHAINTRE, Adobe Research, UK

**ACM Reference Format:**
Prafull Sharma, Julien Philip, Michael Gharbi, Bill Freeman, Fredo Durand, and Valentin Deschaintre. 2023. Materialistic: Selecting Similar Materials in Images – Supplemental Material. *ACM Trans. Graph.* 42, 4, Article 1 (August 2023), 2 pages. https://doi.org/10.1145/3592390

## 1 MODEL ARCHITECTURE

In this document, we define the details of the model architecture required to implement the proposed model.

### 1.1 Extracting DINO features

We use DINO ViT8 as the backbone visual representation for our model [Caron et al. 2021]. Transformers treat image patches as a bag of words, and therefore an input image is split into $(8, 8)$ patches. This vision transformer comprises 11 self-attention blocks, where each block outputs a 768 dimensional global context token and *i.e.* 768 dimensional local tokens, one for each $(8, 8)$ patch of the image. Note that since each of the local tokens holds information about a single patch, our approach involves operating spatially on this low-resolution representation from multiple DINO blocks. Given an input image, we extract the local tokens and global token from 4 blocks, namely block 2, 5, 8, and 11, following previous work [Ranftl et al. 2021].

### 1.2 Spatial Processing

*1.2.1 Combining global and local tokens.* Upon extracting the local and global tokens we combine them: the local tokens contain information about a patch while the global token stores information about the context of the image. We first combine the information in the global token with each of the local tokens using a linear projection. This is implemented by concatenating the global token from a given block with each local token of the respective block along the channel dimension and passing them through a linear layer. The output is a 768 dimensional token map of shape $(768, \frac{H}{8}, \frac{W}{8})$.

*1.2.2 Processing using convolution.* We aim to create a spatial pyramid by processing each of these token maps using a CNN with a kernel size of 1, stride of 1, and no padding to output a 256 dimensional token map, maintaining the same spatial extent. Note that there is an independent convolution layer for each of the blocks.

We further upsample some of these outputs to generate a spatial pyramid, where the token map from the deepest blocks (block 8 and 11) are maintained at the original resolution, while token maps corresponding to blocks 2 and 5 are bilinearly upsampled by a scale factor of 4 and 2 respectively before being passed in a convolution layer.

*1.2.3 Cross-Similarity Feature Weighting Layer.* For each token map in the spatial pyramid generated from the previous layer, we aim to compute a similarity-weighted embedding map where the similarity is computed with respect to a query embedding computed based on a "user-input" query pixel (randomly selected during training).

This proposed operator can be divided into three steps.

*Computing the query embedding.* Given a token map from the spatial pyramid described earlier and the query pixel, we first apply a LayerNorm to the token map. Then, we compute the patch index where the query pixel belongs. Given that the embedding at this patch index contains information for the entire patch, we further process this embedding by concatenating the local coordinates of the query point in the patch to the embedding and passing it through an MLP with 1 hidden layer. The output embedding is of the same dimension (256) as the input.

*Computing the similarity.* The layer-normalized token map is passed through two independent linear layers to yield Key K and Value V. Using the computed query embedding, we compute the similarity to all embeddings in K, by computing a dot product followed by a sigmoid. This results in a 1-dimensional score between [0, 1] for each of the embeddings in the token map.

*Computing the weighted embeddings.* This similarity score is then multiplied element-wise with V, yielding an embedding map of the same spatial extent as the input token map.

*1.2.4 Fusing embedding maps at multiple scales.* These similarity-weighted embedding maps are then fused using residual convolution layers starting from the lowest resolution map. For every pair of embedding maps $L_{i-1}$ and $L_i$, where i is the block number in [1, 4]. We begin by processing the lowest $i$ block using a CNN with 2 ReLU-convolution blocks. Both convolutions output embedding maps of the same channel and spatial dimension as the input using a kernel size of 3, a stride of 1, and zero-padding of size 1. The output is then bilinearly upsampled by 2x (if needed to match the

Authors' addresses: Prafull Sharma, MIT, USA and Adobe Research, USA, ; Julien Philip, Adobe Research, UK, ; Michael Gharbi, Adobe Research, US, ; Bill Freeman, MIT, US, ; Fredo Durand, MIT, US, ; Valentin Deschaintre, Adobe Research, UK, deschain@adobe.com.

2023. 0730-0301/2023/8-ART1 $15.00
https://doi.org/10.1145/3592390

resolution of $L_i$) and added to $L_i$. The resulting embedding map is passed through a similar 2 ReLU-convolution block, followed by an output convolution with kernel size 1, stride of 1, and no padding.

After fusing all embedding maps of different spatial extents, we use the resulting feature map of shape $(256, H, W)$ to compute a per-pixel score using an MLP. This MLP is comprised of 3 hidden layers, the first one outputting 256 dimensional latents, the second and third one outputting 128 dimensional latents, and the final outputting a single-channel score map of shape $(H, W)$. The resulting score map is passed through a sigmoid to get the final per-pixel similarity scores in $[0, 1]$.

## REFERENCES

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. 2021. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 9650–9660.

René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. 2021. Vision Transformers for Dense Prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 12179–12188.