Appendix

詳細は以下を参照

バックボーン。バックボーンとして、ImageNetで事前訓練されたEfficientNet-B4 [32][1]を使用します。EfficientNet-B4 [32]の層1から層5までの特徴量は、それぞれ24、32、56、160、448のチャンネル数を有します。ここで「層」とは、同じ特徴量サイズを有するステージの組み合わせを指します。5 つの特徴は、同じサイズにリサイズされ、連結されて 720 チャネルの特徴マップを形成します。MVTec-AD [4] では、画像サイズと特徴サイズは、それぞれ 512 × 512 および 32×32 に設定されています。したがって、32×32×720 の形状の特徴マップが得られます。CIFAR-10 [18] の場合、画像サイズは 32 × 32 と非常に小さいです。そのため、特徴マップのサイズは比較的大きく設定され（出力ストライドは 4）、8 × 8 × 720 の特徴マップが得られます。

トランスフォーマー。まず、特徴マップに1×1の畳み込みを適用し、チャネル数を720から256に削減します。その後、特徴マップを分離して特徴トークンに分割します。MVTec-AD [4]とCIFAR-10 [18]では、それぞれチャネル数256で1024と64の特徴トークンがあります。位置埋め込みは、入力特徴トークンと同じサイズの学習済み埋め込みです。

トランスフォーマーエンコーダーは、4 層の [33] の標準アーキテクチャに従っています。各層は、マルチヘッド自己注意層、フィードフォワード層、および層正規化によるショートカット接続で構成されています。注意のヘッド数は 8 です。フィードフォワード層のアーキテクチャは、次のとおりです。

Layer	Input	FC1	Relu	FC2
Output Size	256	1024	1024	256

さらに、ポジション情報をより多く保持するため、各自己アテンション層にポジションエンベディングが追加されており、最初の層だけでなくすべての層に適用されています。
トランスフォーマーデコーダーも 4 つのデコーダー層で構成されています。各層は、自己注意部分と相互注意部分の 2 つの部分で構成されています。自己注意部分には、マルチヘッド自己注意層と、レイヤー正規化によるショートカット接続が含まれます。相互注意部分は、マルチヘッド相互注意層、フィードフォワード層、およびレイヤー正規化によるショートカット接続で構成されています。両方のアテンション層のヘッド数は8に設定されています。フィードフォワード層のアーキテクチャはトランスフォーマーエンコーダーと同じです。また、すべてのアテンション層にポジションエンベディングが追加されています。クエリエンベディングは、入力特徴トークンと同じサイズの学習済みエンベディングです。

トランスフォーマーの出力は入力と同じサイズです（MVTec-ADでは1024 × 256、CIFAR-10では64 × 256）。次に、チャネルを256から720に増やすため、1 × 1の畳み込みが適用されます。reshape後、再構築された特徴マップ（MVTec-ADでは32 × 32 × 720、CIFAR-10では8 × 8 × 720）が得られます。

[1] We use the EfficientNet-B4 checkpoint in https://github.com/lukemelas/EfficientNet-PyTorch/releases/download/1.0/efficientnet-b4-6ed6700e.pth

Appendix

A More Details

Backbone. We use the ImageNet pre-trained EfficientNet-B4 [32] [1] as the backbone. The features from layer1 to layer5 of EfficientNet-B4 [32] have the channel of 24, 32, 56, 160, 448, respectively. Here we define "layer" as the combination of stages that have the same size of features. The 5 features are resized to the same size and concatenated together to form a 720-channel feature map. For MVTec-AD [4], the image size and the feature size are set as 512×512 and 32×32, respectively. Therefore, a feature map with the shape of $32 \times 32 \times 720$ is obtained. For CIFAR-10 [18], the image size is 32×32, which is quite small. Thus the size of the feature map is set relatively large (with the output stride of 4), so an $8 \times 8 \times 720$ feature map is obtained.

Transformer. A 1×1 convolution is applied firstly to the feature map to reduce the channel from 720 to 256. Then the feature map is split to separate feature tokens. For MVTec-AD [4] and CIFAR-10 [18], there are 1024 and 64 feature tokens with the channel of 256, respectively. The position embedding is a learned embedding with the same size as the input feature tokens.

The transformer encoder follows the standard architecture in [33] with 4 layers. Each layer consists of a multi-head self-attention layer, a feed forward layer, and a shortcut connection with layer normalization. The head number in attention is 8. The architecture of the feed forward layer is shown as follows.

Layer	Input	FC1	Relu	FC2
Output Size	256	1024	1024	256

Besides, the position embedding is added in each self-attention layer rather than only in the first layer to keep more position information.

The transformer decoder also has 4 decoder layers. Each layer is composed of 2 parts, the self-attention part and the cross-attention part. The self-attention part includes a multi-head self-attention layer and a shortcut connection with layer normalization. The cross-attention part consists of a multi-head cross-attention layer, a feed forward layer, and a shortcut connection with layer normalization. The head number in both attention layers is set as 8. The architecture of the feed forward layer is the same as that in the transformer encoder. Also, the position embedding is added in all attention layers. The query embedding is a learned embedding with the same size as the input feature tokens.

The outputs of the transformer have the same size as the inputs (1024×256 for MVTec-AD, 64×256 for CIFAR-10). Then a 1×1 convolution is applied to increase the channel from 256 to 720. After reshape, we obtain the reconstructed feature map ($32 \times 32 \times 720$ for MVTec-AD, $8 \times 8 \times 720$ for CIFAR-10).

[1] We use the EfficientNet-B4 checkpoint in https://github.com/lukemelas/EfficientNet-PyTorch/releases/download/1.0/efficientnet-b4-6ed6700e.pth