TABLE V
AMVTEC ADにおける画像レベルでの異常検出結果（AUROC%を使用）。 AUROC%.

| Model | GANomaly [20] | ITAE [11] | Patch SVDD [4] | SPADE [5] (WR50) | MahalanobisAD [23] (EfficientNet-B4) | PaDiM-WR50-Rd550 | PaDiM EfficientNet-B5 |
|---|---|---|---|---|---|---|---|
| all textures classes | - | - | 94.6 | - | 97.2 | 98.8 | **99.0** |
| all objects classes | - | - | 90.9 | - | 94.8 | 93.6 | **97.2** |
| all classes | 76.2 | 83.9 | 92.1 | 85.5 | 95.8 | 95.3 | **97.9** |

最先端の手法（SPADE ［5］、VAE）を、第IV-A節で説明されるMVTec ADの改変版であるRd-MVTec ADに適用しました。この実験の結果は表VIに示されています。各テスト構成において、MVTec ADに対してランダムシードを使用してデータ前処理を5回実行し、5つの異なるデータセットバージョン（Rd-MVTec AD）を取得します。その後、得られた結果を平均化し、表VIに報告します。提示された結果によると、PaDiM-WR50Rd550は、PROスコアとAUROCの両方で、テクスチャとオブジェクトクラスにおいて他のモデルよりも優れた性能を示しています。さらに、SPADE ［5］とVAEのRd-MVTec ADにおける性能は、通常のMVTec ADでの結果と比較して、PaDiM-WR50-Rd550の性能よりも大幅に低下しています（表IIIを参照）。AUROC結果は、PaDiM-WR50-Rd550で5.3ポイント減少したのに対し、VAEとSPADEではそれぞれ12.2ポイントと8.8ポイントの減少でした。したがって、当手法は既存のテスト済み手法に比べて非一致画像に対してより頑健であると考えられます。

TABLE VI
非整列R-MVTEC ADにおける異常局所化結果。結果はタプル（AUROC%、PRスコア%）として表示されます。

| Model | VAE (R18) | SPADE (WR50) | PaDiM-WR50-Rd550 |
|---|---|---|---|
| all texture classes | (54.7, 23.1) | (84.6, 75.6) | (**92.4**, **77.9**) |
| all object classes | (65.8, 30.2) | (88.2, 65.8) | (**92.1**, **70.8**) |
| all classes | (62.1, 27.8) | (87.2, 69.0) | (**92.2**, **73.1**) |

### D. Scalability gain

時間複雑度。PaDiMでは、ガウスパラメータを全トレーニングデータセットを使用して推定するため、トレーニングの時間複雑度はデータセットのサイズに比例して増加します。しかし、深層神経ネットワークのトレーニングを必要とする方法とは異なり、PaDiMは事前学習済みのCNNを使用するため、複雑な深層学習トレーニングが不要です。したがって、MVTec ADのような小規模データセットでのトレーニングは非常に高速かつ簡単です。最も複雑なモデルであるPaDiM-WR50-Rd550の場合、CPU（Intel CPU 6154 3GHz 72th）でのシリアル実装によるトレーニングは、MVTec ADクラスでは平均150秒、STC動画シーンでは平均1500秒かかります。

TABLE VII
MVT EC AD上でCPU Intel I 7-4710HQ @ 2.50GHzを使用した場合の異常検出の平均推論時間（秒単位）。

| Model | SPADE (WR50) | VAE (R18) | PaDiM R18-Rd100 | PaDiM-WR50-Rd550 |
|---|---|---|---|---|
| Inference time (sec.) | 7.10 | 0.21 | 0.23 | 0.95 |

これらのトレーニング手順は、GPUハードウェアを使用してフォワードパスと共分散推定を実行することでさらに高速化可能です。一方、セクションIV-Bで説明された手順に従い、MVTec ADで各クラスごとに10,000枚のイメージを使用してVAEをトレーニングする場合、1つのGPU NVIDIA P5000を使用すると、各クラスあたり2時間40分かかります。一方、SPADE ［5］は学習パラメーターが存在しないため、トレーニングが不要です。それでも、テスト前に通常のトレーニング画像の埋め込みベクトルをすべて計算してメモリに格納します。これらのベクトルはK-NNアルゴリズムの入力となり、表VIIに示すようにSPADEの推論が非常に遅くなります。表VIIでは、主流のCPU（Intel i7-4710HQ CPU @ 2.50GHz）を使用したシリアル実装でモデル推論時間を測定しています。MVTec ADにおいて、SPADEの推論時間は、同様のバックボーンを持つ当社のPaDiMモデルに比べて約7倍遅いです。これは計算コストの高いNN検索が原因です。当社のVAE実装（再構築ベースのモデルと類似）は最も高速なモデルですが、シンプルなモデルPaDiM-R18-Rd100の推論時間は同じオーダーです。同様の複雑さを持つにもかかわらず、PaDiMはVAE手法を大幅に上回っています（セクションV-B参照）。

メモリ複雑度。SPADE ［5］やPatch SVDD ［4］とは異なり、当モデルの空間複雑度はデータセットのトレーニングサイズに依存せず、画像解像度のみに依存します。PaDiMはメモリに事前学習済みのCNNと各パッチに関連するガウスパラメータのみを保持します。表VIIIでは、パラメータをfloat32でエンコードした場合のSPADE、当VAE実装、PaDiMのメモリ要件を示しています。同等のバックボーンを使用した場合、SPADEはMVTec ADにおいてPaDiMよりもメモリ消費量が少ない。しかし、STCのような大規模なデータセットでSPADEを使用すると、そのメモリ消費量は扱いにくくなるのに対し、PaDiM-WR50-Rd550は7倍少ないメモリを必要とする。PaDiMの空間複雑度は、セクションIV-Bで説明したように、後者のデータセットで入力画像の解像度がより高いことから、MVTec ADからSTCへ移行する際に増加します。最後に、当社のフレームワークPaDiMの利点の一つは、ユーザーが推論時間要件、リソース制限、または期待される性能に合わせて、バックボーンと埋め込みサイズを選択することで、方法を容易に適応できる点です。

## VI. Conclusion

私たちは、分布モデリングに基づく異常検出と局所化のためのフレームワークPaDiMを、1クラス学習設定において提案しました。これはMVTec ADとSTCデータセットで最先端の性能を達成しています。さらに、評価プロトコルを非一致データに拡張し、初めて

TABLE V
ANOMALY DETECTION RESULTS (AT THE IMAGE LEVEL) ON THE MVTEC AD USING AUROC%.

| Model | GANomaly [20] | ITAE [11] | Patch SVDD [4] | SPADE [5] (WR50) | MahalanobisAD [23] (EfficientNet-B4) | PaDiM-WR50-Rd550 | PaDiM EfficientNet-B5 |
|---|---|---|---|---|---|---|---|
| all textures classes | - | - | 94.6 | - | 97.2 | 98.8 | **99.0** |
| all objects classes | - | - | 90.9 | - | 94.8 | 93.6 | **97.2** |
| all classes | 76.2 | 83.9 | 92.1 | 85.5 | 95.8 | 95.3 | **97.9** |

state-of-the-art methods (SPADE [5], VAE) on a modified version of the MVTec AD, Rd-MVTec AD, described in Section IV-A. Results of this experiment are displayed in Table VI. For each test configuration we run 5 times data preprocessing on the MVTec AD with random seeds to obtain 5 different versions of the dataset, denoted as Rd-MVTec AD. Then, we average the obtained results and report them in Table VI. According to the presented results, PaDiM-WR50-Rd550 outperforms the other models on both texture and object classes in the PRO-score and the AUROC. Besides, the SPADE [5] and VAE performances on the Rd-MVTec AD decrease more than the performance of PaDiM-WR50-Rd550 when comparing to the results obtained on the normal MVTec AD (refer to Table III). The AUROC results decrease by 5.3p.p for PaDiM-WR50-Rd550 against 12.2p.p and 8.8p.p decline for VAE and SPADE respectively. Thus, we can conclude that our method seems to be more robust to non-aligned images than the other existing and tested works.

TABLE VI
ANOMALY LOCALIZATION RESULTS ON THE NON-ALIGNED RD-MVTEC AD. RESULTS ARE DISPLAYED AS TUPLES (AUROC%, PRO-SCORE%)

| Model | VAE (R18) | SPADE (WR50) | PaDiM-WR50-Rd550 |
|---|---|---|---|
| all texture classes | (54.7, 23.1) | (84.6, 75.6) | (**92.4**, **77.9**) |
| all object classes | (65.8, 30.2) | (88.2, 65.8) | (**92.1**, **70.8**) |
| all classes | (62.1, 27.8) | (87.2, 69.0) | (**92.2**, **73.1**) |

### D. Scalability gain

**Time complexity**. In PaDiM, the training time complexity scales linearly with the dataset size because the Gaussian parameters are estimated using the entire training dataset. However, contrary to the methods that require to train deep neural networks, PaDiM uses a pretrained CNN, and, thus, no deep learning training is required which is often a complex procedure. Hence, it is very fast and easy to train it on small datasets like MVTec AD. For our most complex model PaDiM-WR50-Rd550, the training on a CPU (Intel CPU 6154 3GHz 72th) with a serial implementation takes on average 150 seconds on the MVTec AD classes and 1500 seconds on average on the STC video scenes. These training procedures could be further accelerated using GPU hardware for the forward pass and the covariance estimation. In contrast, training the VAE with 10 000 images per class on the MVTec AD following the procedure described in Section IV-B takes 2h40 per class using one GPU NVIDIA P5000. Conversely, SPADE [5] requires no training as there are no parameters to learn. Still, it computes and stores in the memory before testing all the embedding vectors of the normal training images. Those vectors are the inputs of a K-NN algorithm which makes SPADE's inference very slow as shown in Table VII.

TABLE VII
AVERAGE INFERENCE TIME OF ANOMALY LOCALIZATION IN SECONDS ON THE MVTEC AD WITH A CPU INTEL I7-4710HQ @ 2.50GHZ.

| Model | SPADE (WR50) | VAE (R18) | PaDiM R18-Rd100 | PaDiM-WR50-Rd550 |
|---|---|---|---|---|
| Inference time (sec.) | 7.10 | 0.21 | 0.23 | 0.95 |

In Table VII, we measure the model inference time using a mainstream CPU (Intel i7-4710HQ CPU @ 2.50GHz) with a serial implementation. On the MVTec AD, the inference time of SPADE is around seven times slower than our PaDiM model with equivalent backbone because of the computationally expensive NN search. Our VAE implementation, which is similar to most reconstruction-based models, is the fastest model but our simple model PaDiM-R18-Rd100 has the same order of magnitude for the inference time. While having similar complexity, PaDiM largely outperfoms the VAE methods (see Section V-B).

**Memory complexity**. Unlike SPADE [5] and Patch SVDD [4], the space complexity of our model is independent of the dataset training size and depends only on the image resolution. PaDiM keeps in the memory only the pretrained CNN and the Gaussian parameters associated with each patch. In Table VIII we show the memory requirement of SPADE, our VAE implementation, and PaDiM, assuming that parameters are encoded in float32. Using equivalent backbone, SPADE has a lower memory consumption than PaDiM on the MVTec AD. However, when using SPADE on a larger dataset like the STC, its memory consumption becomes intractable, whereas PaDiM-WR50-Rd550 requires seven times less memory. The PaDiM space complexity increases from the MVTec AD to the STC only because the input image resolution is higher in the latter dataset as described in Section IV-B. Finally, one of the advantages of our framework PaDiM is that the user can easily adapt the method by choosing the backbone and the embedding size to fit its inference time requirements, resource limits, or expected performance.

## VI. CONCLUSION

We have presented a framework called PaDiM for anomaly detection and localization in one-class learning setting which is based on distribution modeling. It achieves state-of-the-art performance on MVTec AD and STC datasets. Moreover, we extend the evaluation protocol to non-aligned data and the first