A. Proofs for Section 3.2

In this Appendix, we prove the propositions in Section 3.2. The point is to compare various potential loss functions for neural transformation learning in their ability to produce useful transformations for self-supervised anomaly detection.

Requirements 1 and 2 formalize what we consider useful transformations. The learned transformations should produce diverse views the share semantic information with the original sample.

We give two example edge-cases that violate the requirements, the 'constant' edge-case in which the transformed views do no longer depend on the original sample (this violates the semantic requirement) and the 'identity' edge-case in which all transformations reproduce the original sample perfectly but violate the diversity requirement.

We compare what happens to various losses for self-supervised anomaly detection under these edge cases. Specifically, we compare the loss of our method $\mathcal{L}$ (Equation (2)) to the transformation prediction loss $\mathcal{L}_P$ (Equation (4)) of Wang et al. (2019b), and the SimCLR loss $\mathcal{L}_C$ (Equation (5), Chen et al. (2020)), which has been used for anomaly detection in Sohn et al. (2021) and Tack et al. (2020). In the original sources these losses have been used with fixed transformations (typically image transformations like rotations, cropping, blurring, etc.). Here we consider the same losses but with the learnable transformations parameterized as defined in Section 3.1. All notation has been defined in Section 3.1.

A.1. Proof of Proposition 1

We first investigate whether we can optimize the transformation prediction loss Equation (4) with respect to the transformation parameters θ_k and the parameters of f_ϕ and obtain learned transformations that fulfill Requirements 1 and 2.

The proposition below states that a constant edge-case can achieve a global minimum of Equation (4), which means that minimizing it can produce transformations the violate Requirement 1.

Proposition 1. *The 'constant' edge-case $f_\phi(T_k(x)) = Cc_k$, where c_k is a one-hot vector encoding the k^{th} position (i.e. $c_{kk} = 1$) tends towards the minimum of $\mathcal{L}_P$ (Equation (4)) as the constant C goes to infinity.*

Proof. As a negative log probability $\mathcal{L}_P \geq 0$ is lower bounded by 0. We want to show that with $f_\phi(T_k(x)) = Cc_k$, (where c_k is a one hot vector and C is a constant,) $\mathcal{L}$ goes to 0 as C goes to infinity. Plugging $f_\phi(T_k(x)) = Cc_k$ into $\mathcal{L}_p$ and taking the limit yields

$$\begin{aligned}\lim_{C\to\infty} \mathcal{L}_P &= \lim_{C\to\infty} \mathbb{E}_{x\sim\mathcal{D}}[-\sum_{k=1}^{K} \log \frac{\exp C}{\exp C + K - 1}] = \lim_{C\to\infty} -K \log \frac{\exp C}{\exp C + K - 1} \\ &= \lim_{C\to\infty} -KC + K\log(\exp C + K - 1) = 0\end{aligned}$$

□

A.2. Proof of Proposition 2

Next, we ask what would happen if we optimized the SimCLR loss $\mathcal{L}_C$ (Equation (5)) with respect to the transformation parameters and the encoder.

The result is, that if we allowed the encoder f_ϕ to be as flexible as necessary to achieve a global minimum of $\mathcal{L}_C$, then we can derive another minimum of $\mathcal{L}_C$ that relies only on identity transformations, thereby obtaining a solution to the minimization problem that violates the diversity requirement.

Proposition 2. *The 'identity' edge-case $T_k(x) = x$ with adequate encoder f_ϕ is a minimizer of $\mathcal{L}_C$ (Equation (5)).*

Proof. $\mathcal{L}_c(\mathcal{M})$ can be separated as the alignment term and the uniformity term.

$$\mathcal{L}_c(\mathcal{M}) = \underbrace{\sum_{i=1}^{N}\left[-\log h(x_1^{(i)}, x_2^{(i)}) - \log h(x_2^{(i)}, x_1^{(i)})\right]}_{\mathcal{L}_{\text{alignment}}} \tag{6}$$
$$+ \underbrace{\sum_{i=1}^{N}\left[\log\left[\sum_{j=1}^{N} h(x_1^{(i)}, x_2^{(j)}) + \sum_{j=1}^{N} \mathbb{1}_{[j\neq i]} h(x_1^{(i)}, x_1^{(j)})\right] + \log\left[\sum_{j=1}^{N} h(x_2^{(i)}, x_1^{(j)}) + \sum_{j=1}^{N} \mathbb{1}_{[j\neq i]} h(x_2^{(i)}, x_2^{(j)})\right]\right]}_{\mathcal{L}_{\text{uniformity}}}.$$

A sufficient condition of $\min(\mathcal{L}_c(\mathcal{M}))$ is both $\mathcal{L}_{\text{alignment}}$ and $\mathcal{L}_{\text{uniformity}}$ are minimized.

$$\min(\mathcal{L}_c(\mathcal{M})) \geq \min(\mathcal{L}_{\text{alignment}}) + \min(\mathcal{L}_{\text{uniformity}})\,. \tag{7}$$

Given an adequate encoder f_ϕ^*, that is flexible enough to minimize both $\mathcal{L}_{\text{alignment}}$ and $\mathcal{L}_{\text{uniformity}}$ for all transformation pairs T_1 and T_2, we will show we can construct another solution to the minimization problem that relies only on identity transformations.

The alignment term is only minimized for all T_1, T_2, if $f_\phi^*(T_1(x^{(i)})) = f_\phi^*(T_2(x^{(i)}))$ for all $x^{(i)} \sim \mathcal{M}$. So we know for f_ϕ^* that

$$f_\phi^* = \arg\min_{f_\phi} \mathcal{L}_{\text{alignment}} \quad \iff \quad \text{sim}(f_\phi^*(T_1(x^{(i)})), f_\phi^*(T_2(x^{(i)}))) = 1 \quad \forall\, x^{(i)} \sim \mathcal{M} \tag{8}$$
$$\iff \quad f_\phi^*(T_1(x^{(i)})) = f_\phi^*(T_2(x^{(i)})) \quad \forall\, x^{(i)} \sim \mathcal{M}. \tag{9}$$

Define $\tilde{f}_\phi = f_\phi^* \circ T_1$. Since $f_\phi^*(T_1(x^{(i)})) = f_\phi^*(T_2(x^{(i)}))$,

$$\tilde{f}_\phi(\mathbb{I}(x^{(i)})) = f_\phi^*(T_1(x^{(i)})) = f_\phi^*(T_2(x^{(i)})) \quad \forall\, x^{(i)} \sim \mathcal{M}. \tag{10}$$

Using only the identity transformation $\mathbb{I}(x) = x$ for T_1 and T_2, and $\tilde{f}_\phi$ as the encoder in $\mathcal{L}_C$ yields the same minimal loss as under T_1, T_2 and f_ϕ^*. □

A.3. Proof of Proposition 3

Finally, we investigate the effect of the edge-cases from Propositions 1 and 2 on our objective Equation (2).

Proposition 3. *The edge-cases of Propositions 1 and 2 do not minimize $\mathcal{L}$ (DCL, Equation (2)).*

We divide the proposition and its proof into two parts.

Proposition 3, Part 1. *The 'constant' edge-case $f_\phi(T_k(x)) = Cc_k$, where c_k is a one-hot vector encoding the k^{th} position (i.e. $c_{kk} = 1$) does not minimize $\mathcal{L}$ (DCL, Equation (2)) for any C, also not as C tends to infinity.*

Proof. We prove $\nabla_\theta \mathcal{L}(x) \neq 0$ in the 'constant' edge-case. For simplicity, we define $z_k := f_\phi(T_k(x))$, and $z := f_\phi(x)$. The gradient of $\mathcal{L}(x)$ with respect to the parameters is

$$\begin{aligned}
\nabla_\theta \mathcal{L}(x) &= \sum_{k=1}^{K}\left[-\nabla_\theta \text{sim}(z_k, z) + \frac{h(x_k, x)\nabla_\theta \text{sim}(z_k, z) + \sum_{l\neq k} h(x_k, x_l)\nabla_\theta \text{sim}(z_k, z_l)}{h(x_k, x) + \sum_{l\neq k} h(x_k, x_l)}\right] \\
&= \sum_{k=1}^{K}\left[\frac{(h(x_k, x) - h(x_k, x) - \sum_{l\neq k} h(x_k, x_l))\nabla_\theta \text{sim}(z_k, z) + \sum_{l\neq k} h(x_k, x_l)\nabla_\theta \text{sim}(z_k, z_l)}{h(x_k, x) + \sum_{l\neq k} h(x_k, x_l)}\right] \\
&= \sum_{k=1}^{K}\left[\frac{\sum_{l\neq k} h(x_k, x_l)(\nabla_\theta \text{sim}(z_k, z_l) - \nabla_\theta \text{sim}(z_k, z))}{h(x_k, x) + \sum_{l\neq k} h(x_k, x_l)}\right],
\end{aligned} \tag{11}$$

where the score function $h(\cdot)$ (Equation (1)) is larger than zero. We plug $z_k := f_\phi(T_k(x)) = Cc_k$ into Equation (11).

$$\nabla_\theta \mathcal{L}(x) = \sum_{k=1}^{K} \left[\frac{\sum_{l \neq k} (\nabla_\theta \text{sim}(Cc_k, Cc_l) - \nabla_\theta \text{sim}(Cc_k, z))}{h(Cc_k, x) + K - 1} \right] \tag{12}$$

$$\nabla_\theta \mathcal{L}(x) = \sum_{k=1}^{K} \left[\frac{\sum_{l \neq k} (\nabla_\theta \text{sim}(c_k, c_l) - \nabla_\theta \text{sim}(c_k, z))}{h(c_k, x) + K - 1} \right], \tag{13}$$

where the second line is obtained by using the fact that the cosine similarity only depends on the normalized vectors and hence C can be dropped.

As $c_k \perp c_l$, $\nabla_\theta \text{sim}(c_k, c_l) - \nabla_\theta \text{sim}(c_k, z) \leq 0$, and the equality is true only if $c_l = z/|z|$. So, the sufficient and necessary condition of $\nabla_\theta \mathcal{L}(x) = 0$ is $\forall k$, $c_k = z/|z|$. As $c_k \neq c_l$, it contradicts with the sufficient and necessary condition. Therefore, $\nabla_\theta \mathcal{L}(x) \neq 0$ and the 'constant' edge-case cannot be a minimizer of $\mathcal{L}$ □

Proposition 3, Part 2. *The 'identity' edge-case $T_k(x) = x$ does not minimize $\mathcal{L}$ (Equation (2)) for adequate encoder f_ϕ.*

Proof. Plugging $T_k(x) = x$ for all k into Equation (2)

$$\mathcal{L} = \mathbb{E}_{x \sim \mathcal{D}}[-\sum_{k=1}^{K} \log \frac{h(x, x)}{K h(x, x)}] = K \log K. \tag{14}$$

This is K times the cross-entropy of the uniform distribution, meaning that using the identity transformation is equivalent to random guessing for the task of the DCL, which is to predict which sample is the original given a transformed view. When f_ϕ is adequate (i.e. flexible enough) we can do better than random on $\mathcal{L}$. This can be seen in the anomaly scores in Figure 2b which are much smaller than $K \log K$ after training (better than random).

We can also construct examples which achieve better than random performance. For example, for $K = 2$, taking $z_1 \perp z$, $z_2 \perp z$, and $z_1 = -z_2$, does better than random. The loss values (with $\tau = 1$) of the two cases are

- 'Identity' edge-case: $z_1 = z_2 = z$, so $\mathcal{L}(x) = -2 \log(0.5) = 1.386$.
- Counterexample: $z_1 \perp z$, $z_2 \perp z$, and $z_1 = -z_2$, so $\mathcal{L}(x) = -2 \log(1/(1 + \exp(-1))) = 0.627$.

The counterexample achieves a lower loss value than the 'identity' edge-case. So the 'identity' edge-case is not the minimum of $\mathcal{L}(x)$. □

B. Implementation details

B.1. Implementations of NeuTraL AD on time series datasets

The networks in the neural transformations used in all experiments consist of a stack of three residual blocks of 1d convolutional layers with instance normalization layers and ReLU activations, as well as one convolutional layer on the top. All convolutional layers are with the kernel size of 3, and the stride of 1. All bias terms are fixed as zero, and the learnable affine parameters of the instance normalization layers are frozen. The dimension of the residual blocks is the double data dimension. The convolutional layer on the top has an output dimension as the data dimension. For the multiplicative parameterization, a sigmoid activation is added to the end.

The encoder used in all experiments consists of residual blocks of 1d convolutional layers with ReLU activations, as well as one 1d convolutional layer on the top of all residual blocks, which is used to reduce the dimensions to the latent dimension 64. No batch normalization layer is applied.

The detailed network structure (from bottom to top) in each time series dataset is:

- SAD: (i) one residual block with the kernel size of 3, the stride of 1, and the output dimension of 32. (ii) four residual blocks with the kernel size of 3, the stride of 2, and the output dimensions of 32, 64, 128, 256. (iii) one 1d convolutional layer with the kernel size of 6, the stride of 1, and the output dimension of 64.

- NATOPS: (i) one residual block with the kernel size of 3, the stride of 1, and the output dimension of 32. (ii) four residual blocks with the kernel size of 3, with the stride of 2, and the output dimensions of 32, 64, 128, 256. (iii) one 1d convolutional layer with the kernel size of 4, the stride of 1, and the output dimension of 64.

- CT: (i) one residual block with the kernel size of 3, the stride of 1, and the output dimension of 32. (ii) six residual blocks with the kernel size of 3, the stride of 2, and the output dimensions of 32, 64, 128, 256, 256, 256. (iii) one 1d convolutional layer with the kernel size of 3, the stride of 1, and the output dimension of 64.

- EPSY: (i) one residual block with the kernel size of 3, the stride of 1, and the output dimension of 32. (ii) six residual blocks with the kernel size of 3, the stride of 2, and the output dimensions of 32, 64, 128, 256, 256, 256. (iii) one 1d convolutional layer with the kernel size of 4, the stride of 1, and the output dimension of 64.

- RS: (i) one residual block with the kernel size of 3, the stride of 1, and the output dimension of 32. (ii) three residual blocks with the kernel size of 3, the stride of 2, and the output dimensions of 32, 64, 128. (iii) one 1d convolutional layer with the kernel size of 4, the stride of 1, and the output dimension of 64.

B.2. Implementations of baselines

- Traditional Anomaly Detection Baselines. OC-SVM, IF, and LOF are taken from scikit-learn library with default parameters.

- Deep Anomaly Detection Baselines. The implementations of Deep SVDD, DROCC, and DAGMM are adopted from the published codes with a similar encoder as NeuTraL AD. DAGMM has a hyperparameter of the number of mixture components. We consider the number of components between 4 and 12 and select the best performing one.

- Self-supervised Anomaly Detection Baselines. The implementation of GOAD is taken from the published code. The results of GOAD depend on the choice of the output dimension r of affine transformations. We consider the reduced dimension $r \in \{2^2, 2^3, ..., 2^6\}$, and select the best performing one. We craft specific time series transformations for the designed classification-based baseline. The hand-crafted transformations are the compositions of flipping along the time axis (true/false), flipping along the channel axis (true/false), and shifting along the time axis by 0.25 of its time length (forward/backward/none). By taking all possible compositions, we obtain a total of $2 * 2 * 3 = 12$ transformations.

- Anomaly Detection Baselines for Time Series. The RNN is parameterized by two layers of recurrent neural networks, e.g. GRU, and a stack of two linear layers with ReLU activation on the top of it which outputs the mean and variance at each time step. The implementation of LSTM-ED is taken from the web.

C. Tabular datasets

The four used tabular datasets are:

- Arrhythmia: A cardiology dataset from the UCI repository contains 274 continuous attributes and 5 categorical attributes. Following the data preparation of previous works, only 274 continuous attributes are considered. The abnormal classes include 3, 4, 5, 7, 8, 9, 14, and 15. The rest classes are considered as normal.

- Thyroid: A medical dataset from the UCI repository contains attributes related to hyperthyroid diagnosis. Following the data preparation of previous works, only 6 continuous attributes are considered. The hyperfunction class is treated as abnormal, and the rest 2 classes are considered as normal.

- KDDCUP: The KDDCUP99 10 percent dataset from the UCI repository contains 34 continuous attributes and 7 categorical attributes. Following the data preparation of previous works, 7 categorical attributes are represented by one-hot vectors. Eventually, the data has 120 dimensions. The attack samples are considered as normal, and the non-attack samples are considered as abnormal.

- KDDCUP-Rev: It is derived from the KDDCUP99 10 percent dataset. The non-attack samples are considered as normal, and attack samples are considered as abnormal. Following the data preparation of previous works, attack data is sub-sampled to consist of 25% of the number of non-attack samples.

D. Additional qualitative results

D.1. Results for time series data

In Figure 7, we show the learned transformations (parameterized as $T(x) = M(x) \quad x$ and $K = 4$) on spoken Arabic digits. The learned transformations given one normal example are shown in the first row. The learned transformations given one example of each abnormal class are shown in the following rows.

In Figure 8, we show the learned transformations (parameterized as $T(x) = M(x)$ and $K = 4$) on NATOPS. The learned transformations given one normal example are shown in the first row. The learned transformations given one example of each abnormal class are shown in the following rows.

D.2. Results for tabular data

The learned transformations of thyroid, which are visualized in Figure 9, offer us the possible explanations of why a data instance is an anomaly. We illustrate one normal example and three anomalies in the first row. Since the anomaly score Equation (3) is the sum of terms caused by each transformation. Each score shown in the last row has four bars indicating the terms caused by each transformation. The score of the normal example is very low, and its bars are invisible from the plot. The scores of three anomalies are mainly contributed by different terms (colored with orange). The four learned masks are colored blue and listed in four rows. M_4 focuses on checking the value of the fourth attribute and contributes high values to the scores of all listed anomalies. In comparison, M_2 is less useful for anomaly detection. NeuTraL AD is able to learn diverse transformations but is not guaranteed to learn transformations that are useful for anomaly detection, since no label is included in the training.

We project the score terms of test data contributed by T_1, T_3, and T_4 to a simplex to visualize which transformation dominates the anomaly score in Figure 10. From the left subplot, we can see, the scores of normal data (blue) are not dominated by any single transformation, while the scores of anomalies are mainly dominated by T_3 and T_4. In the right subplot, we visualize the magnitudes of scores via transparency. We can see, the score magnitudes of normal data are clearly lower than the score magnitudes of anomalies.

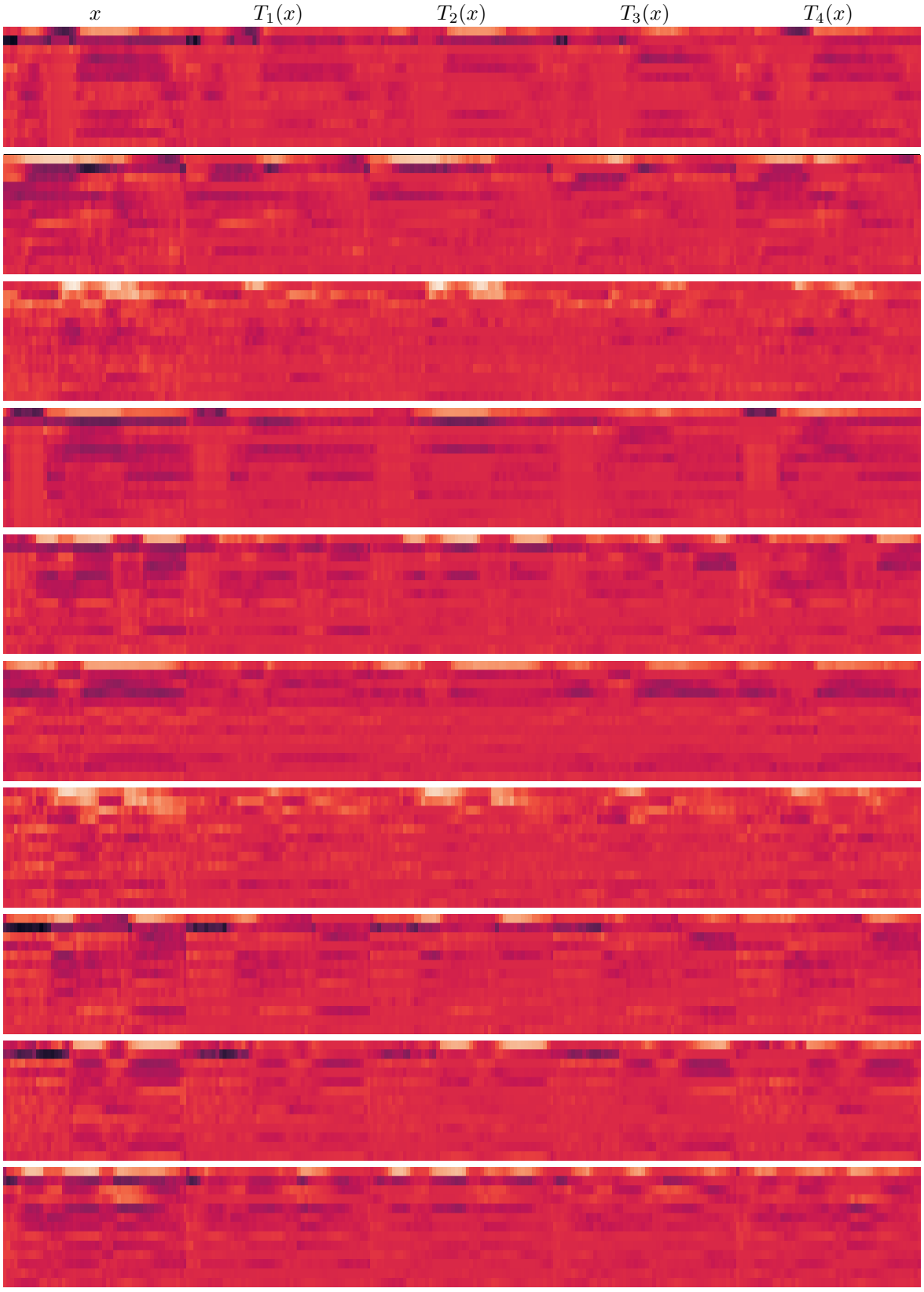

Figure 7. The learned transformations ($T(x) = M(x) \quad x$) on SAD. The first row: the learned transformations of one given example of the normal class. The rest rows: the learned transformations of one given example of each abnormal class.

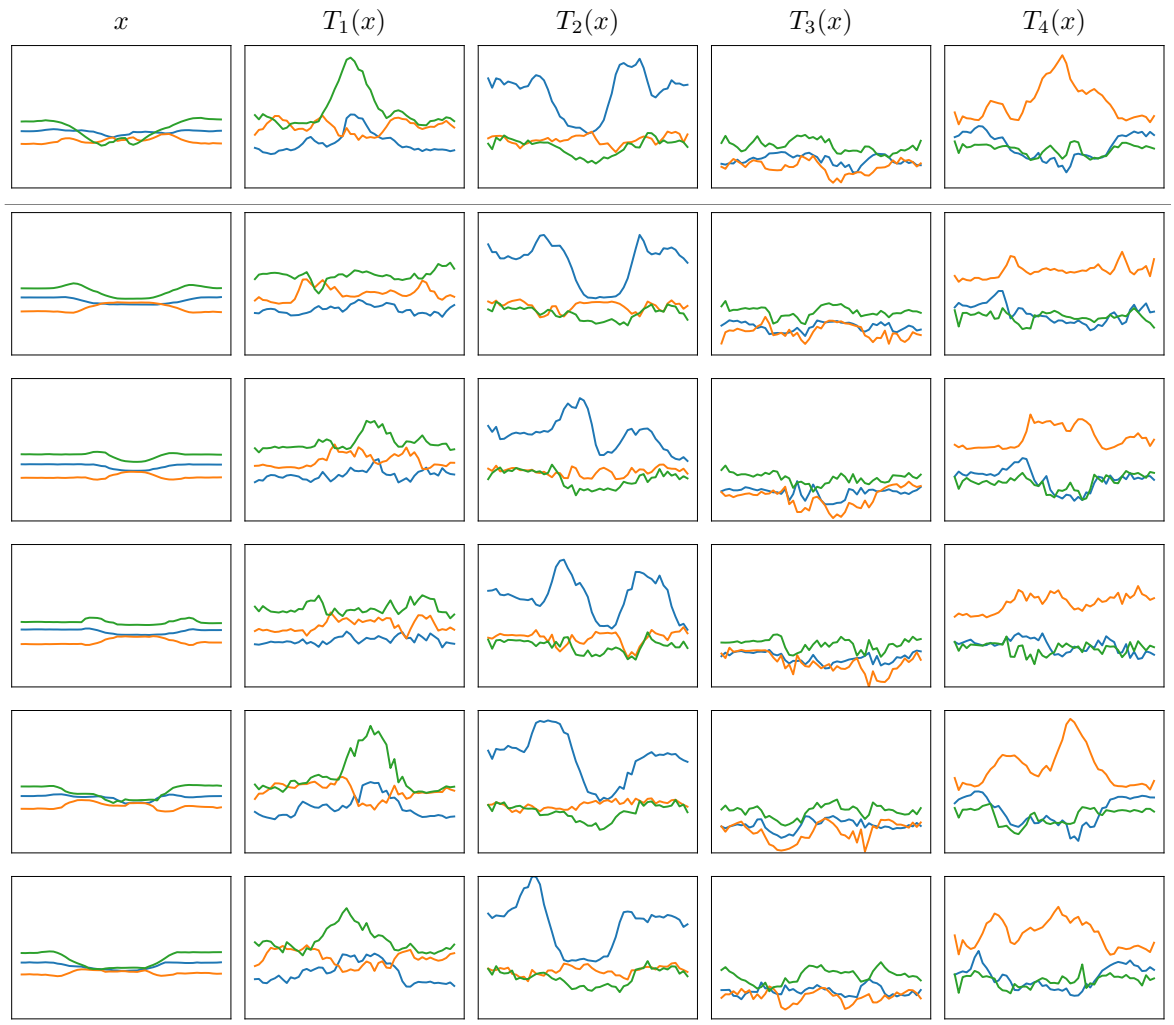

Figure 8. The learned transformations ($T(x) = M(x)$) on NATOPS. The first row: the learned transformations of one given example of the normal class. The rest rows: the learned transformations of one given example of each abnormal class.

Figure 9. Visualizations of thyroid. The first row: one normal example and three abnormal examples. The second to the fourth rows: the learned four masks of them. The fifth row: the scores contributed by each transformation of them, where the highest term is colored by orange. The plots on each row are under the same scale.

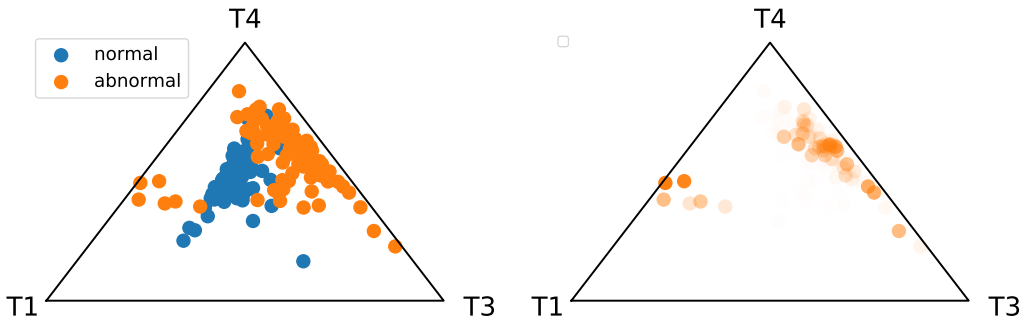

Figure 10. Visualizations of thyroid. We project the scores contributed by T_1, T_3, and T_4 to a simplex. The subplot on the left visualizes which transformation dominates the score. The subplot on the right visualizes the scores via transparency.