Temporality Spatialization: A Scalable and Faithful Time-Travelling Visualization for Deep Classifier Training

Xianglin Yang[1,2], **Yun Lin**[1,2*], **Ruofan Liu**[2] and **Jin Song Dong**[2]
[1]Department of Computer Science, Shanghai Jiao Tong University, China
[2]School of Computing, National University of Singapore, Singapore
xianglin@u.nus.edu, {dcsliny, dcslirf, dcsdjs}@nus.edu.sg

Abstract

Time-travelling visualization answers *how the predictions of a deep classifier are formed during the training*. It visualizes in two or three dimensional space how the classification boundaries and sample embeddings are evolved during training.

In this work, we propose TimeVis, a novel time-travelling visualization solution for deep classifiers. Comparing to the state-of-the-art solution DeepVisualInsight (DVI), TimeVis can significantly (1) reduce visualization errors for rendering samples' travel across different training epochs, and (2) improve the visualization efficiency. To this end, we design a technique called *temporality spatialization*, which unifies the spatial relation (e.g., neighbouring samples in single epoch) and temporal relation (e.g., one identical sample in neighbouring training epochs) into one high-dimensional topological complex. Such spatio-temporal complex can be used to efficiently train one visualization model to accurately project and inverse-project any high and low dimensional data across epochs. Our extensive experiment shows that, in comparison to DVI, TimeVis not only is more accurate to preserve the visualized time-travelling semantics, but also 15X faster in visualization efficiency, achieving a new state-of-the-art in time-travelling visualization.

1 Introduction

Time-travelling visualization, solution to address *how the deep model predictions are formed during the training*, are emerging as a new branch of explainable AI [Yang *et al.*, 2022c]. Such visualization demonstrates how the high-dimensional classification landscape is learned during training, which can facilitate training-based causal inference on embedding generation [Mallick *et al.*, 2019] and active learning algorithms [Sener and Savarese, 2017].

Any time-travelling visualization solutions for deep classifiers are required to fulfill three spatial properties and one temporal property so as to faithfully reflect the dynamics of high-dimensional classification landscape [Yang *et al.*, 2022c]. Briefly, given a set of high-dimensional representations trained in epoch e_t, the three spatial properties are:

- **Neighbour preserving property:** The neighbours of any high-dimensional representation should be preserved after being projected into a visible low-dimensional space.
- **Boundary distance preserving property:** The relative distance from any high-dimensional representation to its closest classification boundary is preserved after being projected to a visible low-dimensional space.
- **Inverse-projection preserving property:** After projecting a high-dimensional representation $\mathbf{x}$ into low-dimensional space as $\mathbf{y}$, we shall inverse-project $\mathbf{x}$ back to the high-dimensional space as $\mathbf{x}'$ such that $\mathbf{x}$ and $\mathbf{x}'$ are similar.

Besides, the one temporal property requires:

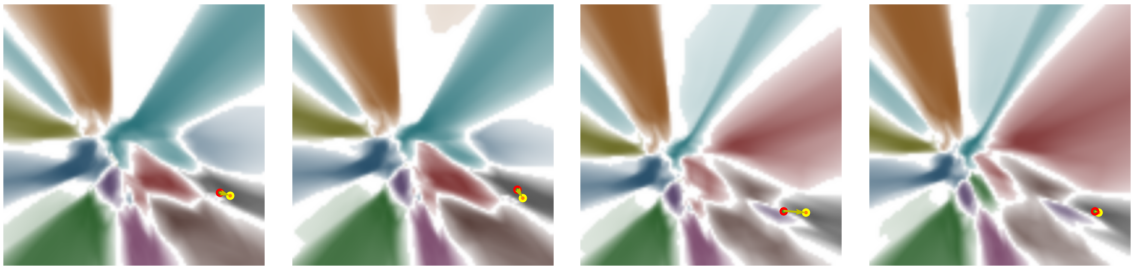
- **Temporal preserving property (continuity):** If the neighbours of a high-dimensional representation $\mathbf{x}^{t+k}$ (from epoch $t+k$) do not change much from its correspondence $\mathbf{x}^t$ (from epoch t), then their low-dimensional projections $\mathbf{y}^{t+k}$ and $\mathbf{y}^t$ shall be close.

These properties ensure that the "animation" of a model's training process can be "played" reflectively (thanks to the spatial properties) and smoothly (thanks to the temporal property) in a visible low-dimensional space.

DeepVisualInsight (DVI), as a state-of-the-art, is further crystallized for fulfilling these properties [Yang *et al.*, 2022c]. Technically, DVI trains visualization models as auto-encoders on several loss functions enforcing the properties. However, the solution has its limit on the visualization scalability and the faithfulness on the travelling semantics of samples across different epochs.

Problem 1: Visualization Scalability. Given that the classification landscape differs one epoch from another, the time-travelling visualization is generally a record-and-replay technique to record and visualize the partially-trained classifier in each epoch. DVI trains one visualization model on each time step. Hence, the visualization training process can be prohibitively costly, especially when the number of training epochs is large. Experiments report that training one visualization model at a normal GPU-workstation takes about 15 minutes [Yang *et al.*, 2022c], which indicates that training the visualization models for 100 epochs takes around 24 hours. Even worse, in order to satisfy the temporal property,

*Corresponding author



(a) epoch 97 (unexpected movement of a point) (b) epoch 98 (unexpected movement of a point) (c) epoch 99 (unexpected movement of a point) (d) epoch 100 (unexpected creation of a green territory)

Figure 1: The erroneous visualization perturbation of DVI while the classification semantics remain the same among the epochs: Each color represents a class, while the white region between the territories represents the classification boundary. Given a sample (i.e., a point), its color represents its label, and its background color represents its prediction. For example, being located in a territory of "bird" (green territory) means a sample is predicted as a bird. We show the erroneous fluctuation trajectory of a static sample. The point with yellow edge indicates the location of current epoch and that of red edge indicates its location of previous epoch.

training the visualization models for the classifier at epoch e_t requires the availability of the visualization model at epoch e_{t-1}, which limits its paralleling potential.

Problem 2: Faithfulness of Travelling Semantics (Visualization Error). When we project the learned representations of a sample $X = \{\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^w\}$ ($\mathbf{x}^i$ indicates the sample representation at i-th epoch) to low-dimensional space in chronological order, their projections $Y = \{\mathbf{y}^1, \mathbf{y}^2, \ldots, \mathbf{y}^w\}$ form a travelling trajectory. When we observe the movement of $\mathbf{y}^i$ to $\mathbf{y}^{i+1}$ on the two-dimensional canvas, it can be caused by (1) the actual movement from $\mathbf{x}^i$ to $\mathbf{x}^{i+1}$ in the high-dimensional space and (2) the visualization perturbation or errors. We call the movement caused by the latter as the *unfaithful* movement of a sample. The visualization model is trained as $V = \langle \phi, \psi \rangle$ where ϕ projects high-dimensional points to low-dimensional space and ψ inverse-projects low-dimensional points back to high-dimensional space. DVI trains the visualization model $V = \langle \phi, \psi \rangle$ at epoch $i + 1$ by retraining from the visualization model at epoch i. The randomness-inducing retraining process can inevitably incur unintentional fluctuation to visualize the classification landscape. In addition, the temporal loss designed in DVI can *freeze* the minority of dynamic samples while the majority samples do not move in the high-dimensional space. As a result, the visualized travelling semantics cannot well faithfully reflect the dynamics in high-dimensional space.

In this work, we propose TimeVis to uniformly address the above problems by a technique called *temporality spatialization*. In comparison to DVI which trains a visualization model for each partially trained classifier, we train a visualization model to capture the representation embeddings of *all* the classifiers. Specifically, our approach "spatializes" the temporal relations of a sample and its correspondences in other epochs. In TimeVis, we regard every representation has both *spatial* neighbours and *temporal* neighbours. Technically, we (1) collect all appeared representation during the training, (2) construct a *spatio-temporal complex* to capture both spatial and temporal relation of the representations in the same high-dimensional space, and (3) learn the dimensional projection and inverse-projection based on such complex. To further improve the visualization scalability, we propose a complex-reduction technique for the spatio-temporal complex, which can successfully compress the size of the complex without compromising much visualization accuracy.

Our experiments on a variety of datasets show that, in comparison to DVI, TimeVis can (1) significantly improve the training scalability (15X faster), (2) largely improve travelling faithfulness, and (3) achieve comparable performance on the other spatial properties.

In summary, this work makes the following contributions:

- We identify two critical problems limiting the practical applicability of the emerging time-travelling visualization.
- We propose *temporality spatialization* to significantly improve the visualization scalability and visualization faithfulness of travelling semantics, achieving a new state-of-the-art in time-travelling visualization.
- We build a tool TimeVis based on our technique to support its practical use, which is publicly available at [Yang *et al.*, 2022b] and https://github.com/xianglinyang/TimeVis.

2 Motivation

Figure 1 shows the time-travelling visualization results of DVI at late training stage when training ResNet18 on the CIFAR10 dataset. At late stage, the semantics of the classifier converge. Generally, the loss, the training accuracy, and the testing accuracy do not change. Individually, the representations of almost all the training samples preserve their k nearest neighbours in those epochs.

However, DVI has two visualization errors which can cloud the understanding of the classification landscape. First, a green territory (i.e., the bird class) is created in the middle, locating near to the red territory (i.e., the cat class) from epoch 99 to epoch 100. However, the bird samples falling in this new green territory are no closer to the cat-image samples in the representation space. Second, we further observe some erroneous fluctuation of samples on the canvas. The sample moves back-and-forth during the four epochs as shown in Figure 1. On the contrary, both its representation and neighbours remain the same across the epochs in the high-dimensional

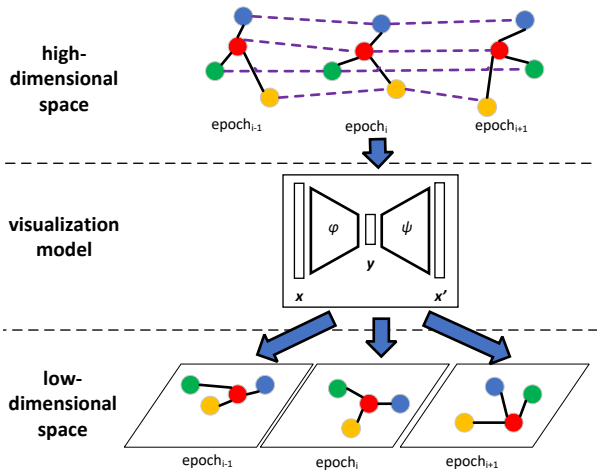


Figure 2: Overview: the solid lines represent spatial neighbour relations, the dashed lines represent temporal neighbour relations. Each color represents the point identity across epochs. A unified visualization model can project any samples to the low-dimensional space.

space. The noisy visualization perturbation can misguide the user to doubt the training stableness, and misunderstand what exactly happen in the high-dimensional space. Due to space limit, we also show a "frozen" sample on the canvas by DVI on [Yang *et al.*, 2022b], where the sample is very dynamic in the high-dimensional space.

3 Problem Definition

Given a dataset $\mathcal{D} = \{\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_n\}$ in d dimensional space (n is the dataset size), a set of classes $\mathcal{C} = \{c_1, c_2, ..., c_m\}$ (m is the number of class size), we have a sequence of w deep classifiers $f^1(\cdot), f^2(\cdot), \ldots, f^w(\cdot)$ taken in chronological order, where each $f^t : \mathbb{R}^d \to \mathbb{R}^m$ is a learned classifier at the training epoch e_t. We assume that each deep classifier $f^t(\cdot)$ can be decomposed into a feature function $g^t : \mathbb{R}^d \to \mathbb{R}^r$ and a prediction function $h^t : \mathbb{R}^r \to \mathbb{R}^m$ (i.e., $f^t = h^t \circ g^t$) with r being the dimension of the representation space. Thus, we can derive a set of sample representations $X^t = \{\mathbf{x}_1^t, \mathbf{x}_2^t, \ldots, \mathbf{x}_n^t\}$ for each epoch t, where $\mathbf{x}_i^t = g^t(\mathbf{s}_i) \in \mathbb{R}^r$. Note that we use superscript and subscript to denote different time steps and different samples respectively. The learned representations of a sample $\mathbf{s}_i$ in different time steps are denoted as $X_i = \{\mathbf{x}_i^1, \mathbf{x}_i^2, \ldots, \mathbf{x}_i^w\}$.

We aim for one visualization model $V = \langle \phi, \psi \rangle$ where $\phi : \mathbb{R}^r \to \mathbb{R}^l$ and $\psi : \mathbb{R}^l \to \mathbb{R}^r$ ($l \in \{2, 3\}$ is the dimension of the visible low-dimensional space). Given a visualization solution $V = \langle \phi, \psi \rangle$, we (1) visualize each representation $\mathbf{x}_i^t$ as $\mathbf{y}_i^t = \phi(\mathbf{x}_i^t)$ in a visible low-dimensional space; and (2) paint any point $\mathbf{y} \in \mathbb{R}^l$ in canvas by $c = \arg\max_{c \in \mathcal{C}} h^t(\psi(\mathbf{y}))$ to form classification landscape. Moreover, unconfident area (i.e., decision boundaries) is colored as white. In addition, we require $V = \langle \phi, \psi \rangle$ to satisfy three spatial properties (i.e., neighbour preserving property, boundary distance preserving property, and inverse-project preserving property) and one temporal property [Yang *et al.*, 2022c].

4 Methodology

Overview Figure 2 shows an overview of TimeVis solution. In high-dimension space, we consider a superset of representations $\mathbf{X} = X^1 \cup X^2 \cup ... \cup X^w$. For each sample $x_i^t \in X$, we define its k spatial neighbours and k temporal neighbours. An example of temporal neighbour of x_i^t can be x_i^{t+1} or x_i^{t-1}, i.e., the identical self in the next or previous epoch. By this means, the temporal information is "spatialized". In contrast to spatial neighbours, temporal neighbours can capture the dynamics of a sample in high-dimensional space across epochs. Moreover, the temporal and spatial neighbours are now comparable, which allows us to further capture the relationship of different samples under different epochs.

Based on the defined neighbouring relation, we build a unified *spatio-temporal complex* for $\mathbf{X}$. Naturally, we only need one visualization model $V = \langle \phi, \psi \rangle$ to project between $\mathbf{x} \in \mathbf{X} \subset \mathbb{R}^r$ and $\mathbf{y} \in \mathbf{Y} \subset \mathbb{R}^l$. Note that the same location $\mathbf{y}$ in low-dimensional space may have different color (i.e., prediction) in different epoch t according to $c = \arg\max_{c \in \mathcal{C}} h^t(\psi(\mathbf{y}))$. In this work, we use the auto-encoder architecture to implement V. Since the temporal relation has been spatialized, the auto-encoder just needs to be trained based loss functions corresponding to the three spatial properties (see Appendix A [Yang *et al.*, 2022a]).

4.1 Spatio-temporal Complex

Spatial Neighbours Given X^t and a distance metric defined on X^t, $d(\cdot) : \mathbb{R}^r \times \mathbb{R}^r \to \mathbb{R}_{\geq 0}$ (e.g., Euclidean distance), we define x_i^t's k spatial neighbours $N_k^S(\mathbf{x}_i) = \operatorname{argmin}_{\mathcal{J} \subset X^t \setminus \{x_i^t\}, |\mathcal{J}|=k} \sum_{x_j^t \in \mathcal{J}} d(\mathbf{x}_j^t, \mathbf{x}_i^t)$ as its k spatial neighbours.

Temporal Neighbours Given a superset $\mathbf{X} = X_1 \cup X_2 \cup \cdots \cup X_n$ and a distance metric defined on $\mathbf{X}$, $d(\cdot)$. For a given $\mathbf{x}_i^t \in X_i$, we denote its k nearest temporal neighbours as $N_k^T(\mathbf{x}_i^t) = \operatorname{argmin}_{\mathcal{J} \subset X_i \setminus \{x_i^t\}, |\mathcal{J}|=k} \sum_{x_i^{t'} \in \mathcal{J}} d(\mathbf{x}_i^{t'}, \mathbf{x}_i^t)$.

Given an embedding $x_i^t \in X_i$, we use a temporal sliding window with size W to describe its temporal spectrum. In other words, we find its k temporal neighbours $N_k^T(x_i^t)$ from the set $X_i' = \{max(1, x_i^{t-\frac{W}{2}}), \ldots, x_i^t, \ldots, min(N, x_i^{t+\frac{W}{2}})\}$. Here, W is an even positive integer, the threshold k and W are the user-defined values.

Spatio-temporal Complex Given the universal superset $X = X^1 \cup X^2 \cup \cdots \cup X^T$, we build the spatio-temporal complex as a graph $G_{st} = \langle X, E_s \cup E_t \rangle$ where

- the spatial relation set is $E_s = \{(\mathbf{x}_i^t, \mathbf{x}_j^t) | \mathbf{x}_i^t, \mathbf{x}_j^t \in X^t, (\mathbf{x}_i^t \in N_k^S(\mathbf{x}_j^t) \text{ or } \mathbf{x}_j^t \in N_k^S(\mathbf{x}_i^t))\}$
- the temporal relation set is $E_t = \{(\mathbf{x}_i^{t_l}, \mathbf{x}_i^{t_m}) | \mathbf{x}_i^{t_l}, \mathbf{x}_i^{t_m} \in \mathbf{X}, (\mathbf{x}_i^{t_l} \in N_k^T(\mathbf{x}_i^{t_m}) \text{ or } \mathbf{x}_i^{t_m} \in N_k^T(\mathbf{x}_i^{t_l}))\}$.

For any $e = (\mathbf{x}_i^{t_l}, \mathbf{x}_j^{t_m})$ ($e \in E_s \cup E_t$), we assign its edge weight as the similarity $p_{i^{t_l} j^{t_m}}$ on the manifold $\mathcal{M}$ (See Appendix B for the definition).

Following the training process as in [Yang *et al.*, 2022c] to synthesize boundary points based on the spatial points, train our auto-encoder as the visualization model, we sample the edges in G_{st}, project the edge ends (i.e., data samples) to

Algorithm 1 Spatial-temporal Complex Construction

Input: a sequence of T spatial complex, $\langle X^1, E_{s_1}\rangle, \langle X^2, E_{s_2}\rangle, \ldots, \langle X^T, E_{s_T}\rangle$, window size, W
Output: reduced spatial-temporal complex, G'_{st}
1: **for** each epoch $t \in [1, T]$ **do**
2: $\quad X^{t'}, E'_{s_t} = reduce(X^t, E_{s_t})$
3: **end for**
4: $E'_t \leftarrow \emptyset$
5: **for** each epoch $t \in [1, T]$ **do**
6: $\quad E'_t = E'_t \cup construct\_temporal\_relation(X^{t'}, W)$
7: **end for**
8: **return** $G'_{st} = \langle X^{1'} \cup \cdots \cup X^{T'}, E'_{s_1} \cup \cdots \cup E'_{s_T} \cup E'_t\rangle$

Algorithm 2 Greedy Down-sampling Algorithm

Input: a initial sampling size s, a coverage threshold th_c, a predefined radius r, a representation set X
Output: a subset X'
1: $X' = random\_sample(X, s)$
2: $X_c = coverage(X', X, r)$
3: **while** $\frac{|X_c|}{|X|} < th_c$ **do**
4: $\quad \mathbf{x}' = k\_center\_greedy(X \setminus X_c)$
5: $\quad X' = X' \cup \{\mathbf{x}'\}$
6: $\quad X_c = coverage(X', X, r)$
7: **end while**
8: **return** X'

low-dimensional space, and tune the model regarding the loss functions corresponding to the three spatial properties.

4.2 Complex Reduction

It is prohibitively expensive to train a visualization model on the super large spatio-temporal complex when epoch number w and dataset size n are large. The dataset size n is usually much larger than epoch number w (i.e., $n \gg w$), meaning the training barrier lies in n.

A spatial-temporal complex G_{st} can also be written as $G_{st} = \langle X^1 \cup X^2 \cup \cdots \cup X^w, (E_{s_1} \cup E_{s_2} \cup \cdots \cup E_{s_w}) \cup E_t\rangle$, where each E_{s_t} ($t \in [1, w]$) represents the spatial neighbouring relations in epoch e_t, and E_t represents the temporal neighbouring relations among epochs $[1, w]$. We denote $G_t = \langle X^t, E_{s_t}\rangle$ as a spatial complex at epoch e_t. From the perspective of complex construction, reducing the size of each X^t can efficiently leads to the size reduction of E_{s_t} and E_t. It in turn reduces the training cost. In his work, we simplify the complex construction as Algorithm 1. Namely, we reduce each spatial complex regarding the complex structure and reduction size (line 1-3). Then, we build up temporal relations between reduced spatial complex (line 5-7).

Spatial Complex Reduction Given a spatial complex $G_s = \langle X, E\rangle$, we aim to reduce it to $G'_s = \langle X', E'\rangle$ so that (1) G_s and G'_s can share as much spatial topological structure as possible and (2) $||X| - |X'||$ can be maximized. The former ensures the representativeness of G'_s for G_s, and the latter ensures the efficiency of the follow-up complex construction and model training. In this work, we estimate the topological similarity between G_s and G'_s by their persistent homology. For self-containment, we provide a brief explanation of persistent homology. Readers can refer to [Chazal *et al.*, 2015; Moor *et al.*, 2020] for more details.

Persistent homology is a concept from topological data analysis (TDA) which describes the topological features on a given point set $\mathbb{X}$ where the distance of any pair of points can be evaluated. In topology, a *simplice* represents a high-dimensional topological structure such as point, triangle, tetrahedrons, etc. A *simplicial complex* $\mathcal{K}$ consists of a set of simplices. Let $\mathcal{K}_\mathcal{X}$ be the universal set consisting of all the simplices on $\mathbb{X}$, a *filtration* is a family of simplicial complexes on $\mathcal{K}_\mathcal{X}$ regarding a distance-based condition $\mathcal{C}$ so that:

1. Given a distance a, $\mathcal{C}$ can be used to slice a subset of simplices $\mathcal{K}_a \subset \mathcal{K}_\mathcal{X}$. For example, line segments (1-simplix in hyperspace) of length $l \leq a$ can be selected to $\mathcal{K}_a$.
2. For any $a \leq b$, $\mathcal{K}_a \subseteq \mathcal{K}_b$.

Specifically, a filtration $\mathcal{F} = \{\mathcal{K}_a \subset \mathcal{K}_\mathcal{X} : a \in \mathbb{R}_{\geq 0}\}$. Given $\mathcal{K}_a \in \mathcal{F}$, we can derive topological features (e.g., holes) based on its simplices. For example, a topological hole can be constructed by connecting a set of simplices where each simplice $x \in \mathcal{K}_a$. Therefore, on $\mathbb{X}$, a topological feature π can appear at distance a_{birth} and disappear at a distance a_{death}, denoted as $life(\pi) = (a_{birth}, a_{death})$. We define *persistent homology* on $\mathbb{X}$ to track the life span of all the topological features on $\mathbb{X}$. *Persistent diagram* is a representation of persistent homology where the life span of topological features are represented as dots/lines in a plane.

Given two compact subsets X and X' from the same metric space $\mathbb{X}$, the bottleneck distance $d_b(\mathcal{D}_X, \mathcal{D}_{X'})$ of their persistent diagrams $\mathcal{D}_X$ and $\mathcal{D}_{X'}$ is proved to satisfy the following constraint [Chazal *et al.*, 2016; Chazal *et al.*, 2014]:

$$d_b(\mathcal{D}_X, \mathcal{D}_{X'}) \leq 2d_H(X, X') \tag{1}$$

The Hausdorff distance d_H between X and X' from the same metric space $(\mathbb{X}, d)$ is:

$$d_H = \max\{\max_{\mathbf{x}\in\mathbf{X}} \min_{\mathbf{x}'\in\mathbf{X}'} d(\mathbf{x}, \mathbf{x}'), \max_{\mathbf{x}'\in\mathbf{X}'} \min_{\mathbf{x}\in\mathbf{X}} d(\mathbf{x}, \mathbf{x}')\} \tag{2}$$

Hence, Hausdorff distance $d_H(X, X')$ serves as a upper bound for $d_b(\mathcal{D}_X, \mathcal{D}_{X'})$. Therefore, we can transform the spatial complex reduction problem to a search problem to find $X' \subset X$ such that (1) $d_H(X, X')$ is minimized and (2) $||X| - |X'||$ is maximized. In this work, we design a greedy down-sampling algorithm similar to the k center problem [Farahani and Hekmatfar, 2009] to address the issue.

Greedy Down-sampling Algorithm Given radius r, we denote $B(\mathbf{x}, r)$ as the ball centered at $\mathbf{x}$ with radius r where $\mathbf{x} \in \mathbb{X}$. We reduce the problem as the search problem for X' where $|X'|$ is minimized so that $\bigcup_{\mathbf{x}'_i \in X'} B(\mathbf{x}'_i, r)$ can cover X, which is known to be an NP-hard. Hence, we design a greedy down-sampling algorithm to achieve a sub-optimal subset X' as Algorithm 2.

In Algorithm 2, we first randomly sample s datapoints from X as the initial X' (line 1). Then, we calculate the coverage set $X_c \subset X$ where each $\mathbf{x} \in X_c$ is covered by $\bigcup_{\mathbf{x}'\in X'} B(\mathbf{x}', r)$. Next, we keep adding into X' new datapoint from $X \setminus X_c$ with k-center-greedy algorithm [Gonzalez, 1985] (i.e., selecting the datapoint with largest coverage), until $\bigcup_{\mathbf{x}'\in X'} B(\mathbf{x}', r)$ can cover X regarding user-defined threshold th_c. The threshold $th_c < 1$ is set because of the

sensitivity of Hausdorff distance to outliers, especially when the input dataset contains noises. Finally, we can achieve a minimized possible X' most topologically similar to X.

Radius Selection Note that, Algorithm 2 requires a predefined radius, which is adaptive according to the distribution of X_t in the high-dimensional space. Intuitively, the radius should be large when the representations are uniformly distributed in the space (with a larger X' for X, e.g., in the early training stage). In contrast, the radius can be smaller when the representations are densely clustered in the space (with a smaller X' for X, e.g., in the late training stage). We use Equation 3 to calculate the radius given the distribution of X_t.

$$r_t = \frac{r_0}{(\frac{c_t}{c_0})^{\alpha} \cdot (\frac{d_t}{d_0})^{\beta}} \tag{3}$$

In Equation 3, we first choose a reference epoch e_0, then calculate the maximum l_2 norm of all representations c_0 as a normalization term, and its intrinsic dimension [Facco *et al.*, 2017] d_0 as a measurement of representation diversity. Then, we adjust the radius under epoch e_t with Equation 3. Here, r_0, α, and β are user-defined thresholds.

5 Experiment

We evaluate TimeVis with the following research questions. More details are at TimeVis website [Yang *et al.*, 2022b]

- **RQ1**: Whether TimeVis can faithfully reflect the travelling semantics in high-dimensional space?
- **RQ2**: How scalable is TimeVis?
- **RQ3**: While being more temporally reflective and computationally scalable, whether and how TimeVis need to compromise other spatial properties?

Baseline, Datasets, and Subject Model We choose DVI as our baseline as it is the only time-travelling visualization tool designed to support all spatial and temporal properties. To have a fair comparison, we follow the experiment settings of DVI on selecting datasets and subject models. Specifically, we train ResNet18 on three datasets: CIFAR-10, MNIST, Fashion-MNIST. The high-dimensional representations are extracted from the global average pooling layer (GAP). We visualize all the training epochs of the classifiers on CIFAR-10, MNIST and Fashion-MNIST, i.e., 160, 20, and 50.

Visualization Property Measurement We follow the defined spatial and temporal measurements in [Yang *et al.*, 2022c]. Specifically, (1) $nn_{pv}(k)$ for the k spatial neighbour preserving measurement; (2) $boundary_{pv}(k)$ for the k boundary neighbour preserving measurement; (3) rec_{pv} for prediction-preserving measurement after a representation is reconstructed from low-dimensional space; and (4) $temp_{pv}(k)$ for temporal-preserving measurement of two consecutive epochs.

Note that, TimeVis is designed for addressing the accumulated visualization error suffered by DVI, but the temporal-preserving measurement $temp_{pv}(k)$ only evaluates temporal continuity between two consecutive epochs. Therefore, we define two more measurements $temp_g$ and $temp_l$ to evaluate

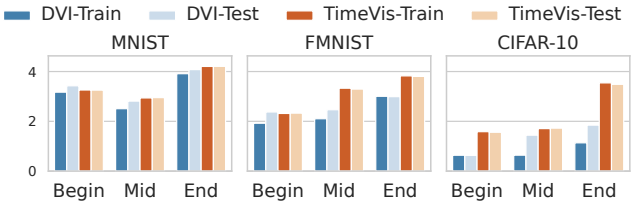


Figure 3: k NN Temporal Neighbour Preserving (k=5)

how the travelling semantics of each high-dimensional representation are preserved in the low-dimensional space across multiple training epochs. First, we evaluate how many k nearest temporal neighbours of a representation $\mathbf{x}_i^t$ can be preserved in the low-dimensional space; Formally, we define $temp_l(i,t,k)$ as:

$$temp_l(i,t,k) = \frac{\sum_{t'=1}^{T} \mathbb{I}(\mathbf{x}_i^{t'} \in N_k^T(\mathbf{x}_i^t) \wedge \mathbf{y}_i^{t'} \in N_k^T(\mathbf{y}_i^t))}{k}$$

where $\mathbb{I}$ is the indicating function evaluated to either 0 or 1. Second, we evaluate the correlation of two rankings for any $\mathbf{x}_i^t$. One ranks its temporal neighbours by distance, denoted as $r(\mathbf{x}_i^t)$; the other ranks its the temporal neighbours of its correspondence $\mathbf{y}_i^t$'in low-dimensional space by distance, denoted as $r(\mathbf{y}_i^t)$. Then the $temp_g$ is defined as:

$$temp_g(i,t) = corr(r(\mathbf{x}_i^t), r(\mathbf{y}_i^t))$$

We use Euclidean distance as $d(\cdot)$ and Kendall's τ as correlation measurement. Higher $temp_g(i,t)$ indicates better preserved semantics, and vice versa.

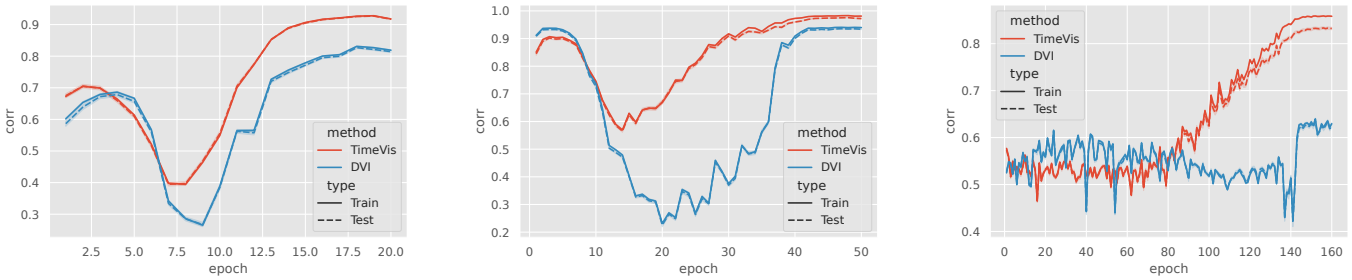
Runtime Configuration We let TimeVis and DVI share the same auto-encoder architecture, i.e., $(r, \frac{r}{2}, \frac{r}{2}, \frac{r}{2}, \frac{r}{2}, 2)$ and $(2, \frac{r}{2}, \frac{r}{2}, \frac{r}{2}, \frac{r}{2}, r)$ given the feature space dimension as r. We choose W as the epoch length. More configuration details can be referred in the Appendix E [Yang *et al.*, 2022a].

Results (RQ1): Preserving travelling semantics Figure 3 shows that TimeVis improves the travelling semantics in different stages. The Mann-Whitney significance testing on $temp_l(i,t,k)$ shows that p-value is smaller than 10^{-5}. Figure 4 further shows how temporal correlation $temp_g(i,t)$ of two approaches changes along the training. Overall, TimeVis outperforms DVI, especially in the late training stages.

Solution		CIFAR-10	MNIST	FMNIST
TimeVis	CC	2572.812	315.457	724.044
	training	1462.170	235.887	657.353
	overall	4034.987	551.344	1381.397
DVI	overall	50212.400	9563.142	22985.150

Table 1: Visualization Training Overhead (in seconds), CC stands for complex construction

Results (RQ2): Visualization scalability Table 1 shows that, the visualization cost of DVI is ∼15X than TimeVis. Comparing to training the visualization model for each recorded classifier, TimeVis only needs to train the visualization model for once.



(a) MNIST (b) FMNIST (c) CIFAR-10

Figure 4: Temporal neighbours ranking correlation

Solution	CIFAR-10		MNIST		FMNIST	
	train	test	train	test	train	test
TimeVis	-0.268	-0.267	-0.542	-0.543	-0.162	-0.164
DVI	-0.240	-0.238	-0.470	-0.470	-0.318	-0.316

Table 2: Temporal property

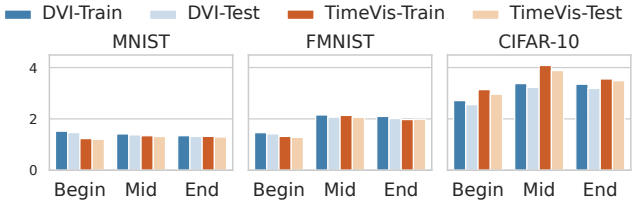


Figure 5: k NN Neighbour Preserving (k=15)

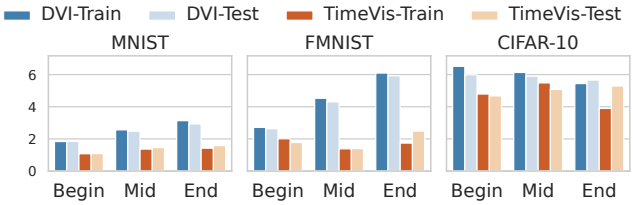


Figure 6: k Boundary Neighbour Preserving (k=15)

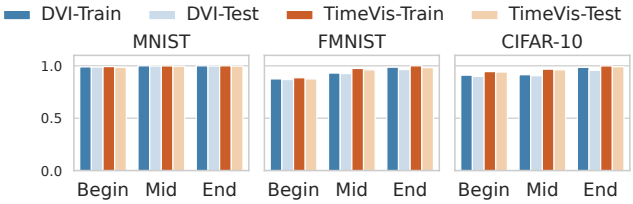


Figure 7: Prediction Preserving Rate

Results (RQ3): Potential compromise Figure 5, Figure 6, Figure 7, and Table 2 compare the performance between TimeVis and DVI. We can see that they are comparable in the measurements of neighbour preserving and prediction preserving rate. However, DVI outperforms TimeVis regarding boundary distance preserving property. We will enhance TimeVis by improving the integration of the boundary points into the spatio-temporal complex in the future work.

6 Related Work

Visual Explanation of Deep Models Many research works have been proposed to provide visual explanations to deep models, including activation maps [Alsallakh *et al.*, 2021; Zeiler and Fergus, 2014] and attribution methods [Adebayo *et al.*, 2018]. Particularly, IntGrad [Sundararajan *et al.*, 2017] and ACE [Chattopadhyay *et al.*, 2019] are the pioneering work in the area. TimeVis focuses on visualizing the training dynamics, which is complementary to those tools to understand the behaviors of deep models.

Dimension Reduction Dimension reduction methods are widely used for visualizing the high-dimensional feature vectors. Existing techniques can be divided into parametric ones (e.g., Topological Autoencoder [Moor *et al.*, 2020], VAE-SNE [Graving and Couzin, 2020], parametric UMAP [Sainburg *et al.*, 2021]) and non-parametric ones such as UMAP [McInnes *et al.*, 2018].

They are the pioneering work to derive time-travelling visualization. Comparing to those techniques on visualizing a "snapshot", time-travelling visualization generates "animation" to understand the model training process.

Time-travelling visualization DVI [Yang *et al.*, 2022c] first proposes three spatial and one temporal properties for any time-traveling visualization methods and trains auto-encoders to fulfill all properties. However, DVI still suffers from large visualization cost and accumulative visualization errors. TimeVis is proposed to addresses the above two problems.

7 Conclusion

We propose TimeVis, a temporality-spatialization based time-travelling visualization approach. TimeVis constructs a unified spatio-temporal complex which captures spatial and temporal neighbours of any high-dimensional samples. TimeVis improves the state-of-the-art solution DVI both in training efficiency and visualization faithfulness. In the future, feedback-based solutions [Lin *et al.*, 2017], trace alignment based solutions [Wang *et al.*, 2019], and input manipulation based solution [Xiao *et al.*, 2021] will be designed to improve tool usability and extensibility;

Acknowledgements

This research is supported in part by the Minister of Education, Singapore (T2EP20120-0019, T1-251RES1901, MOET32020-0004), and A*STAR, CISCO Systems (USA) Pte. Ltd and National University of Singapore under its Cisco-NUS Accelerated Digital Economy Corporate Laboratory (Award I21001E0002)

References

[Adebayo *et al.*, 2018] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. *arXiv preprint arXiv:1810.03292*, 2018.

[Alsallakh *et al.*, 2021] Bilal Alsallakh, Narine Kokhlikyan, Vivek Miglani, Shubham Muttepawar, Edward Wang, Sara Zhang, David Adkins, and Orion Reblitz-Richardson. Debugging the internals of convolutional networks. In *eXplainable AI approaches for debugging and diagnosis.*, 2021.

[Chattopadhyay *et al.*, 2019] Aditya Chattopadhyay, Piyushi Manupriya, Anirban Sarkar, and Vineeth N Balasubramanian. Neural network attributions: A causal perspective. In *International Conference on Machine Learning*, pages 981–990. PMLR, 2019.

[Chazal *et al.*, 2014] Frédéric Chazal, Vin De Silva, and Steve Oudot. Persistence stability for geometric complexes. *Geometriae Dedicata*, 173(1):193–214, 2014.

[Chazal *et al.*, 2015] Frederic Chazal, Brittany Fasy, Fabrizio Lecci, Bertrand Michel, Alessandro Rinaldo, and Larry Wasserman. Subsampling methods for persistent homology. In Francis Bach and David Blei, editors, *ICML*, volume 37 of *Proceedings of Machine Learning Research*, pages 2143–2151, Lille, France, 07–09 Jul 2015. PMLR.

[Chazal *et al.*, 2016] Frédéric Chazal, Vin De Silva, Marc Glisse, and Steve Oudot. *The structure and stability of persistence modules*. Springer, 2016.

[Facco *et al.*, 2017] Elena Facco, Maria d'Errico, Alex Rodriguez, and Alessandro Laio. Estimating the intrinsic dimension of datasets by a minimal neighborhood information. *Scientific reports*, 7(1):1–8, 2017.

[Farahani and Hekmatfar, 2009] Reza Zanjirani Farahani and Masoud Hekmatfar. *Facility location: concepts, models, algorithms and case studies*. Springer Science & Business Media, 2009.

[Gonzalez, 1985] Teofilo F Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical computer science*, 38:293–306, 1985.

[Graving and Couzin, 2020] Jacob M Graving and Iain D Couzin. Vae-sne: a deep generative model for simultaneous dimensionality reduction and clustering. *BioRxiv*, 2020.

[Lin *et al.*, 2017] Yun Lin, Jun Sun, Yinxing Xue, Yang Liu, and Jinsong Dong. Feedback-based debugging. In *2017 IEEE/ACM 39th International Conference on Software Engineering (ICSE)*, pages 393–403. IEEE, 2017.

[Mallick *et al.*, 2019] Koushik Mallick, Sanghamitra Bandyopadhyay, Subhasis Chakraborty, Rounaq Choudhuri, and Sayan Bose. Topo2vec: A novel node embedding generation based on network topology for link prediction. *IEEE Transactions on Computational Social Systems*, 6(6):1306–1317, 2019.

[McInnes *et al.*, 2018] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[Moor *et al.*, 2020] Michael Moor, Max Horn, Bastian Rieck, and Karsten Borgwardt. Topological autoencoders. In *International conference on machine learning*, pages 7045–7054. PMLR, 2020.

[Sainburg *et al.*, 2021] Tim Sainburg, Leland McInnes, and Timothy Q. Gentner. Parametric UMAP Embeddings for Representation and Semisupervised Learning. *Neural Computation*, 33(11):2881–2907, 10 2021.

[Sener and Savarese, 2017] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*, 2017.

[Sundararajan *et al.*, 2017] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*, pages 3319–3328. PMLR, 2017.

[Wang *et al.*, 2019] Haijun Wang, Yun Lin, Zijiang Yang, Jun Sun, Yang Liu, Jin Song Dong, Qinghua Zheng, and Ting Liu. Explaining regressions via alignment slicing and mending. *IEEE Transactions on Software Engineering*, 2019.

[Xiao *et al.*, 2021] Yan Xiao, Ivan Beschastnikh, David S Rosenblum, Changsheng Sun, Sebastian Elbaum, Yun Lin, and Jin Song Dong. Self-checking deep neural networks in deployment. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*, pages 372–384. IEEE, 2021.

[Yang *et al.*, 2022a] Xianglin Yang, Yun Lin, Ruofan Liu, and Jin Song Dong. Appendix. https://sites.google.com/view/timevis/home, 2022. Accessed: 2022-01-15.

[Yang *et al.*, 2022b] Xianglin Yang, Yun Lin, Ruofan Liu, and Jin Song Dong. Website of timevis. https://sites.google.com/view/timevis/home, 2022. Accessed: 2022-01-15.

[Yang *et al.*, 2022c] Xianglin Yang, Yun Lin, Ruofan Liu, Zhenfeng He, Chao Wang, Jin Song Dong, and Hong Mei. Deepvisualinsight: Time-travelling visualization for spatio-temporal causality of deep classification training. In *The Thirty-Sixth AAAI Conference on Artificial Intelligence (AAAI)*, 2022.

[Zeiler and Fergus, 2014] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.