Experimental study of Manifold learning and tangent propagation

Temirlan Ashimov

Department of Mathematics
Nazarbayev University

May 6, 2021

Introduction to Manifold

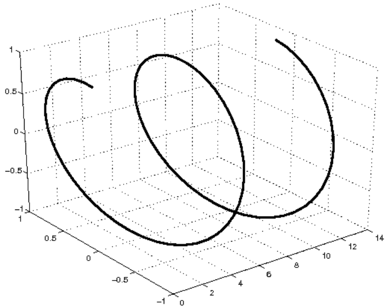
In mathematical literature, one can find many non-equivalent definitions of manifold, that reflect the needs of algebraic geometry, differential geometry, or topology. The following definition is general enough to serve our purposes.

Introduction to Manifold

Definition: A manifold is a topological space that locally resembles Euclidean space near each point. More precisely, each point of an n-dimensional manifold has a neighborhood that is homeomorphic to the Euclidean space of dimension of n. In this more precise terminology, a manifold is referred to as an n-manifold.

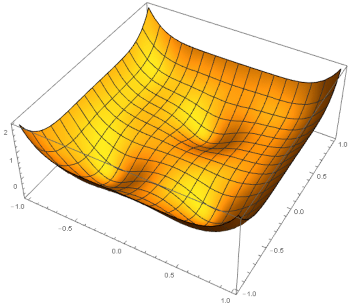
Introduction to Manifold

Intuitively, a manifold is a generalization of curves and surfaces of arbitrary dimensions.



Example 1: *one*-dimensional manifolds are also called curves. Range of injective smooth mapping $\mathcal{M} : \Omega \to \mathbb{R}^3$: i.e. $\mathcal{M}(\Omega) = \{\mathcal{M}(x) | x \in \Omega\}$ (where $\Omega \subseteq \mathbb{R}$ is an open set).

Introduction to Manifold



Example 2: *two*-dimensional manifolds are also called surfaces. Range of injective smooth mapping $\mathcal{M} : \Omega \to \mathbb{R}^3$: i.e. $\mathcal{M}(\Omega) = \{\mathcal{M}(x) | x \in \Omega\}$ (where $\Omega \subseteq \mathbb{R}^2$ is an open set).

Manifold Learning

The "curse of dimensionality" is an obstacle in analysis of data. In order to deal with this kind of problem several **Dimensionality Reduction** (DR) methods exist. The key idea of these methods is that data points are sampled from a neighborhood of low-dimensional manifold, which is embedded in high-dimensional space.

Manifold learning

For instance: given data points

$$\mathbf{X} = \{X_1, X_2, ..., X_n\} \subset \mathbb{R}^m.$$

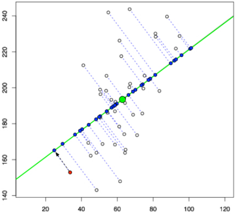
The task is to find an embedding mapping f (preferably linear or "almost" linear)

$$f(\mathbf{X}) = \mathbf{Y} = \{Y_1, Y_2, ..., Y_n\} \subset \mathbb{R}^k$$

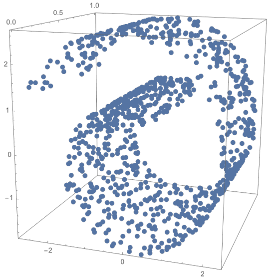
of m-dimensional $\mathbf{X}$ to k-dimensional $\mathbf{Y}$ where $m >> k$ such that some information is maintained.

Manifold Learning

There are two distinct approaches in the DR methods: *linear* and *nonlinear*. **Manifold Learning** is an approach to nonlinear DR.



(a) Linear manifold.



(b) Nonlinear manifold.

Autoencoders

Auto-encoder (AE) is a particular type of Artificial Neural Networks. It learns an *encoder* function f which maps x from input m-dimensional space to a lower k-dimensional space, in combination with *decoder* function g that maps inversely to the original m-dimensional space.

Autoencoders

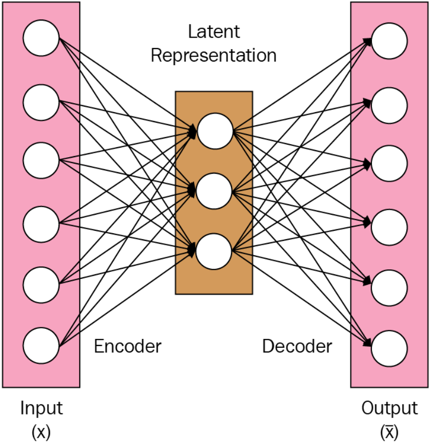


图 1: An illustration of auto-encoder's architecture.

Autoencoders

In fact, the AE is a "correcting" function ϕ that is composition of encoder f and decoder g mappings

$$f : \mathbb{R}^m \to \mathbb{R}^k$$
$$g : \mathbb{R}^k \to \mathbb{R}^m$$

where the difference between original data points (x_i) and "corrected" data points $(\phi(x_i))$ is being minimized:

$$\phi^* \leftarrow \arg\min_{\phi:\mathbb{R}^m \to \mathcal{M}^*} \frac{1}{N} \sum_{i=1}^{N} \|x_i - \phi(x_i)\|^2,$$

where $\mathcal{M}^*$ is the hidden manifold onto which data points are projected $(\mathcal{M}^* \subseteq \mathbb{R}^m)$.

Tangent Bundle

The *tangent bundle* of a manifold $\mathcal{M}$ is a collection of tangent planes at all points on the manifold $\mathcal{M}$. Every tangent plane has its own Euclidean coordinate system or *chart*

Tangent Bundle

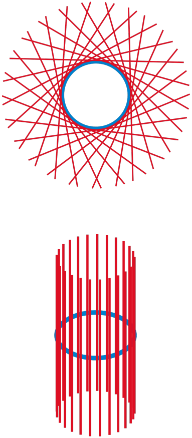


图 2: Tangent bundle of a manifold which in circular form.

Tangent Bundle

The chart on x is defined by the Singular Value Decomposition of the Jacobian for the encoder function f, where f is smooth ($J^T(x) = U(x)S(x)V^T(x)$, where $U(x)$, $V(x)$ are orthogonal matrices and $S(x)$ is a diagonal matrix). The tangent plane $\mathcal{H}_x$ at x is obtained by the span of vectors $\mathcal{B}_x$ such that:

$$\mathcal{B}_x = \{U_k(x)|S_{kk}(x) > \epsilon\} \text{ and } \mathcal{H}_x = \{x + v|v \in span(\mathcal{B}_x)\},$$

where U_k is the k-column of the matrix $U(x)$.

Tangent Propagation

The **Tangent Propagation** is a regularization technique, which encourages the model to be robust with respect to small changes of input data points. In order to achieve the robustness, we have added penalty term to the cost function:

$$\Omega(x) = \sum_{u \in \mathcal{B}_x} \|\frac{\partial o}{\partial x} u\|^2,$$

where o is the output of neural networks.

Manifold Tangent Classifier

- In previous section, auto-encoder was introduced as a manifold learning technique. In fact, the auto-encoder should be a smooth "correcting" function ϕ.
- **Manifold Tangent Classifier** (MTC) algorithm includes second desirable property of "correcting" function ϕ - smoothness of the hidden manifold $\mathcal{M}$. The smoothness is implemented by additional penalty term:

$$\gamma \mathbb{E}_{\epsilon \sim \mathcal{N}(0,\sigma^2 \mathbb{I})} \| \frac{\partial f}{\partial x}(x_i) - \frac{\partial f}{\partial x}(x_i + \epsilon) \|^2.$$

Manifold Tangent Classifier

Thus, the *cost* function of **MTC**'s auto-encoder (contractiveness is added as well) is defined as:

$$\phi^* \leftarrow \arg\min_{\phi:\mathbb{R}^m \to \mathcal{M}^*} \frac{1}{N}\sum_{i=1}^{N} \|x_i - \phi(x_i)\|^2 + \lambda \|\frac{\partial f}{\partial x}(x_i)\|^2 + \gamma \mathbb{E}_{\epsilon \sim \mathcal{N}(0,\sigma^2 \mathbb{I})} \|\frac{\partial f}{\partial x}(x_i) - \frac{\partial f}{\partial x}(x_i + \epsilon)\|^2.$$

Alternating Scheme

- The goal of **Alternating scheme** algorithm is reduction of dimension of data points from m-dimensional space into d-dimensional manifold, where $d < k$ of k-dimensional *code space*.
- Revised minimization objective:

$$\phi^* \leftarrow \arg\min_{\phi:\mathbb{R}^m\rightarrow\mathbb{R}^m, \forall x: rank(\frac{\partial\phi}{\partial x}(x))\leq d} \frac{1}{N}\sum_{i=1}^{N}\|x_i - \phi(x_i)\|^2,$$

where $\phi^* : \mathbb{R}^m \rightarrow \mathbb{R}^m$ is a smooth function whose Jacobian's rank is, ideally less than d, or equal to d at all data points.

Alternating Scheme

- The *cost* function of **Alternating Scheme** algorithm is defined as:

$$F(\theta, \langle B_j \rangle)_{j=1}^{M} = \sum_{i=1}^{N} \|x_i - \phi_\theta(x_i)\|^2 + \gamma \mathbb{E}_{\epsilon \sim \mathcal{N}(0,\sigma^2 I_n)} \|\frac{\partial \phi_\theta}{\partial x}(x_i + \epsilon) - \frac{\partial \phi_\theta}{\partial x}(x_i)\|^2 + \lambda \sum_{j=1}^{M} \|\frac{\partial \phi}{\partial x}(x_{i_j}) - B_{x_{i_j}}\|_F^2,$$

where the *cost* function should be minimized at the same time over θ and matrices $B_{x_{i_j}}$ such that $rank(B_{x_{i_j}}) \leq d$, $j = \overline{1, M}$.

Alternating Scheme

- For fixed matrices $B_{x_{i_j}}$, minimization over θ can be achieved by any gradient descent technique. For fixed parameter θ, minimization over $\{B_{x_{i_j}}\}_{j=1}^{M}$ is equivalent to setting

$$B_{x_{i_j}} \leftarrow U^j_{1:n,1:d}\Sigma^j_{1:d,1:d}(V^j_{1:n,1:d})^T$$

where $\frac{\partial\phi_\theta}{\partial x}(x_{i_j}) = U^j\Sigma^j(V^j)^T$ is a singular value decomposition of $\frac{\partial\phi_\theta}{\partial x}(x_{i_j})$.

Alternating Scheme

Algorithm 1: The alternating algorithm. Hyper-parameters: $m, \lambda, \gamma, \sigma, \alpha$

for $j = 1, ..., M$ **do**
  $B_{x_{i_j}} \leftarrow 0$
end
for $t = 1, ..., T$ **do**
  while θ *has not converged* **do**
    Sample $\{y_i\}_{i=1}^{m} \sim \mathrm{P}_{data(x_1,...,x_N)}$
    Sample $\{\epsilon_i\}_{i=1}^{m} \sim \mathcal{N}(0, \sigma^2)$
    Sample $\{z_i\}_{i=1}^{m} \sim \mathrm{P}_{data(x_{i_1},...,x_{i_M})}$
    $L \leftarrow \frac{1}{m}\sum_{i=1}^{m}(y_i - \phi_\theta(y_i))^2 + \frac{\gamma}{m}\sum_{i=1}^{m}\|\frac{\partial\phi_\theta(y_i+\epsilon_i)}{\partial x} - \frac{\partial\phi_\theta(y_i)}{\partial x}\|^2 + \frac{\lambda}{m}\sum\|\frac{\partial\phi_\theta(z_i)}{\partial x} - B_{z_i}\|^2$
    $\theta \leftarrow Optimizer(\nabla_\theta L, \theta, \alpha)$
  end
  $\theta_t \leftarrow \theta$
  for $j = 1, ..., M$ **do**
    $U^j\Sigma^j(V^j)^T \leftarrow SVD(\frac{\partial g_{\theta_t}}{\partial x}(x_{i_j}))$
    $B_{x_{i_j}} \leftarrow U^j_{1:n,1:k}\Sigma^j_{1:k,1:k}(V^j_{1:n,1:k})^T$
  end
end
Output: $f \leftarrow \phi_{\theta_t}$

Experiments

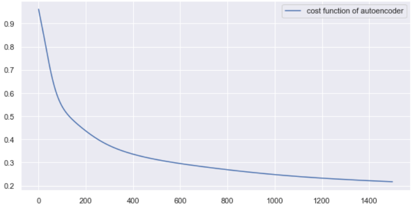
In this work, we implemented Standard Neural Network without regularization, Manifold Tangent Classifier (MTC), and Alternatig Scheme (AS) for an autoencoder (our algorithm). Further task is to compare the accuracy of the algorithm to other models. All experiments were executed on a FIFA 20 dataset using Python 3 programming language.

Experiments

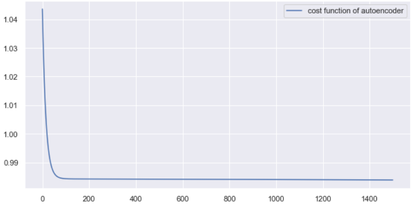
- Data is split into training (0.8), validation (0.1), and test (0.1) sets.
- Hyperparameters of **MTC**'s **AE**1: $m = 40$, $k = 20$, $d = 7$, $\sigma = 0.1$, $\gamma = 1.0$.
- Hyperparameters of **MTC**'s **AE**2: $m = 40$, $k1 = 20$, $k2 = 10$, $d = 7$, $\sigma = 0.1$, $\gamma = 1.0$.
- Hyperparameters of **AS**'s **AE**2: $m = 40$, $k1 = 20$, $k2 = 10$, $d = 7$, $\sigma = 0.1$, $\gamma = 1.0$, $\lambda = 10.0$.
- Optimal hyperparameters of the Regression model were chosen after the application of $Grid - Search$.

Experiments

There are *cost* functions of **MTC**'s autoencoders:



(a) 1-hidden layer.



(b) 2-hidden layers.

Experiments

Results			
Metrics	Linear Regression	Standard Neural Network	1-Layer MTC
R^2	0.34	0.89	0.91

Results		
Metrics	2-Layer MTC	2-Layer AS
R^2	0.91	0.93

Conclusion

Our experiments demonstrate that:

- **MTC**'s autoencoder successfully minimizes the cost function and solves the manifold learning task.
- **MTC** shows better performance than Standard Neural Network (without regularization) does.
- **AS** shows the best performance.

References

1. Rifai, S., Dauphin, Y. N., Vincent, P., Bengio, Y., Muller, X. (2011). The manifold tangent classifier. Advances in neural information processing systems, 24, 2294-2302.
2. Kuleshov, A. P., Bernshtein, A. V., Yanovich, Y. A. (2018). Manifold learning based on kernel density estimation. *Uchenye Zapiski Kazanskogo Universiteta. Seriya Fiziko – Matematicheskie Nauki*, 160(2), 327-338.
3. Bernshtein, A. V. (2018). Manifold learning in statistical tasks. *Uchenye Zapiski Kazanskogo Universiteta. Seriya Fiziko – Matematicheskie Nauki*, 160(2), 229-242.

Thanks!