Unsupervised K-Nearest Neighbor Regression

Oliver Kramer
Department for Computer Science
Carl von Ossietzky Universität Oldenburg
26129 Oldenburg, Germany
okramer@icsi.berkeley.edu

Abstract—In many scientific disciplines structures in high-dimensional data have to be found, e.g., in stellar spectra, in genome data, or for face recognition tasks. In this work we present a novel approach to non-linear dimensionality reduction. It is based on fitting K-nearest neighbor regression to the unsupervised regression framework for learning of low-dimensional manifolds. Similar to related approaches that are mostly based on kernel methods, unsupervised K-nearest neighbor (UKNN) regression optimizes latent variables w.r.t. the data space reconstruction error employing the K-nearest neighbor heuristic. The problem of optimizing latent neighborhoods is difficult to solve, but the UKNN formulation allows an efficient strategy of iteratively embedding latent points to fixed neighborhood topologies. The approaches will be tested experimentally.

I. INTRODUCTION

Dimensionality reduction and manifold learning have an important part to play in the understanding of data. Simple approaches are required that can easily be implemented, interpreted, and adapted to huge data sets. In this work we introduce a simple approach called unsupervised K-nearest neighbor regression. Meinicke [9] proposed a general unsupervised regression framework for learning of low-dimensional manifolds. The idea is to reverse the regression formulation such that low-dimensional data samples in latent space optimally reconstruct high-dimensional output data. We take this framework as basis for an iterative approach that fits KNN to this unsupervised setting in a combinatorial variant. The manifold problem we consider is to learn a mapping $\phi: x \rightarrow \mathbb{R}^d$ that a data set consisting of observed pairs $(x, y)$ in $\mathbb{R}^d \times \mathbb{R}^q$ with $d > q$. The problem is a hard optimization problem as the latent variables $x$ are unknown.

In Section II we will review related dimensionality reduction approaches, and repeat KNN regression. Section III presents the concept of UKNN regression and, two iterative strategies that are based on fixed latent space topologies. Conclusions are drawn in Section IV.

II. RELATED WORK

Many dimension reduction methods have been proposed, a very famous one is principal component analysis (PCA), which assumes linearity of the manifold [1]. An extension for learning of non-linear manifolds is kernel PCA [13] that projects the data into a Hilbert space. Further famous approaches for manifold learning are Isomap by Tenenbaum, Silva, and Langford [16], local linear embedding (LLE) by Roweis and Saul [12], and principal curves by Hastie and Stuetzle [4].

A. Unsupervised Regression

The work on unsupervised regression for dimensionality reduction starts with Meinicke [9], who introduced the corresponding algorithmic framework for the first time. In this line of research early work concentrated on non-parametric kernel density regression, i.e., the counterpart of the Nadaraya-Watson estimator [10] denoted as unsupervised kernel regression (UKR). Klanke and Ritter [7] introduced an optimization scheme based on LLE, PCA, and leave-one-out cross-validation (LOO-CV) for UKR. Carreira-Perpiñán and Lu [2] argue that training of non-parametric unsupervised regression approaches is quite expensive, i.e., $O(N^3)$ in time, and $O(N^2)$ in memory. Parametric methods can accelerate learning, e.g., unsupervised regression based on radial basis function networks (RBFs) [14], Gaussian processes [8], and neural networks [15].

B. KNN Regression

In the following, we give a short introduction to K-nearest neighbor regression that is basis of the UKNN approach. The problem in regression is to predict output values $y \in \mathbb{R}^d$ to given input values $x \in \mathbb{R}^q$ based on sets of $N$ input-output examples $\{(x_1, y_1), \ldots, (x_N, y_N)\}$. The goal is to learn a function $f: \mathbb{R}^q \rightarrow \mathbb{R}$ known as regression function. We assume that a data set consisting of observed pairs $(x_i, y_i) \in \mathbb{R}^d \times \mathbb{R}^q$ is given. For a novel data sample $x'$, KNN regression computes the mean of the function values of its K-nearest neighbors:

$$f_{knn}(x') = \frac{1}{K} \sum_{i \in N_k(x')} y_i$$

(1)

with set $N_k(x')$ containing the indices of the $K$-nearest neighbors of $x'$. The idea of KNN is based on the assumption of locality of data space: In local neighborhoods of $x$ data samples are expected to have similar output values $y$ (or class labels) to $f(x)$. Consequently, for an unknown $x'$ the label must be similar to the labels of the closest data samples, which is modeled by the average of the output value of the $K$ nearest samples. KNN has been proven well in various applications, e.g., in detection of quasars in interstellar data sets [3]. A survey of KNN variants has recently been given by Bhatia and Vandana [1].
III. UNSUPERVISED KNN REGRESSION

In this section we introduce an iterative strategy for UKNN regression based on minimization of the data space reconstruction error (DSRE) \[9\].

A. Concept

An UKNN regression manifold is defined by variables \(x \in X \subset \mathbb{R}^d\) with unsupervised formulation of an UKNN regression manifold

\[ f_{uknn}(x; X) = \frac{1}{K} \sum_{i \in N_K(x; X)} y_i. \tag{2} \]

Matrix \(X\) contains the latent points \(x\) that define the manifold, i.e., the low-dimensional representation of data \(Y\). Parameter \(x\) is the location where the function is evaluated. An optimal UKNN regression manifold minimizes the DSRE

\[ R(X) = \frac{1}{N} \| Y - f_{uknn}(x; X) \|^2_F, \tag{3} \]

with Frobenius norm

\[ \| A \|^2_F = \sum_{i=1}^d \sum_{j=1}^N |a_{ij}|^2. \tag{4} \]

In other words: an optimal UKNN manifold consists of low-dimensional points \(X\) that minimize the reconstruction of the data points \(Y\) w.r.t. the KNN regression method. Regularization in UKNN regression is not as important as regularization in other methods that fit into the unsupervised regression framework. For example, in UKR regularization means penalizing extension in latent space with \(R(X)_p = R(X) + \lambda \| X \|\), and weight \(\lambda\) \[9\]. In KNN regression moving the low-dimensional data samples infinitely apart from each other does not have the same effect as long as we can still determine the K-nearest neighbors. But for practical purposes (limitation of size of numbers) it might be reasonable to restrict continuous KNN latents spaces as well, e.g., to \(x \in [0, 1]^d\). In the following section fixed latent space topologies are used that do not require further regularization.

B. Iterative Strategy 1

For KNN not the absolute positions of data samples in latent space are relevant, but the relative positions that define the neighborhood relations. This perspective reduces the problem to a combinatorial search for neighborhoods \(N_K(x_i, X)\) with \(i = 1, \ldots, N\) that can be solved by testing all combinations of \(K\)-element subsets of \(N\) elements, i.e., all \(\binom{N}{K}\) combinations. The problem is very difficult to solve, in particular for high dimensions. In the following, we introduce a combinatorial approach to UKNN, and introduce two iterative local strategies.

The idea of our first iterative strategy (UKNN 1) is to iteratively assign the data samples to a position in an existing latent space topology that leads to the lowest DSRE. We assume fixed neighborhood topologies with equidistant positions in latent space, and therefore restrict the optimization problem of Equation 2 to a search in a subset of latent space.

As a simple variant we consider the linear case of the latent variables arranged equidistantly on a line \(x \in \mathbb{R}\). In this simplified case only the order of the elements is important. The first iterative strategy works as follows:

1) Choose one element \(y \in Y\) i.i.d.,
2) test all \(N + 1\) intermediate positions of the \(N\) embedded elements in latent space,
3) choose the latent position that achieves the lowest DSRE, and embed \(y\),
4) remove \(y\) from \(Y\), and repeat from step 1 until all elements have been embedded.

Figure 1 illustrates the \(N + 1\) possible embeddings of a data sample into an existing order of points in latent space (yellow circles). For example, the position of element \(x_3\) results in a lower DSRE with \(K = 2\) than the position of \(x_5\), as the mean of the two nearest neighbors of \(x_3\) is closer to \(y\) than the mean of the two nearest neighbors of \(x_5\).

The complexity of UKNN 1 can be described as follows. Each DSRE evaluation takes \(Kd\) computations. We assume that the \(K\) nearest neighbors are saved in a list during the embedding for each latent point \(x\), so that the search for indices \(N_K(x, X)\) takes \(O(1)\) time. The DSRE has to be computed for \(N + 1\) positions, which takes \((N + 1) \cdot Kd\) steps, i.e., \(O(N)\) time.

C. Iterative Strategy 2

The iterative approach introduced in the last section tests all intermediate positions of previously embedded latent points. We propose a second iterative variant (UKNN 2) that only tests the neighbored intermediate positions in latent space of the nearest embedded point \(y^* \in Y\) in data space. The second iterative strategy works as follows:

1) Choose one element \(y \in Y\) i.i.d.,
2) look for the nearest \(y^* \in Y\) that has already been embedded (w.r.t. distance measure like Euclidean distance),
3) choose the latent position that achieves the lowest DSRE, and embed \( y \),
4) remove \( y \) from \( Y \), and repeat from step 1 until all elements have been embedded.

Figure 2 (right) illustrates the embedding of a 2-dimensional point \( y \) (yellow) left or right of the nearest point \( y^* \) in data space. The position with the lowest DSRE is chosen. In comparison to UKNN 1, \( N \) distance comparisons in data space have to be computed, but only 2 positions have to be tested w.r.t. the data space reconstruction error. UKNN 2 computes

\[
\text{a satisfying topological sorting. Parts (c) and (d) show the embeddings of UKNN 1 and UKNN 2 of the spiral, which also lead to reasonable results.}
\]

2) 3D-S: In the following, we will test UKNN regression on a 3-dimensional \( S \) data set (3D-S). The variant without a hole consists of 500 data points, the variant with a hole in the middle consists of 400 points. Figure 4 (a) shows the order of elements of the 3D-S data set without a hole at the

\[
\text{the nearest embedded point } y^* \text{ for each data point, which takes } Nd \text{ steps. Only for the two neighbors the DSRE has to be computed, resulting in an overall number of } Nd + 2Kd \text{ steps, i.e., it takes } O(N) \text{ time. Because of the multiplicative constants, UKNN 2 is faster in practice. For example, for } N = 1,000, K = 10, \text{ and } d = 100, \text{ UKNN 1 takes } 1,001,000 \text{ steps, while UKNN 2 takes 102,000 steps. Testing all combinations takes } \binom{1000}{10} \text{ steps, which is not computable in reasonable time. The following experimental section will answer the question, if this speedup of UKNN 2 has to be paid with worse DSREs.}
\]

D. Experiments

This section shows the behavior of the iterative strategies on three test problems. We will compare the DSRE of both strategies to the initial DSRE at the end of this section.

1) 2D-S and 2D-Spiral: First, we compare UKNN 1 and UKNN 2 on a simple 2-dimensional data set, i.e., the 2-dimensional noisy \( S \) with \( N = 200 \) (2D-S), and the 2-dimensional spiral with \( N = 100 \). Figure 3 shows the experimental results. Similar colors correspond to neighboring latent points. The following experiments employ \( K = 5 \) nearest neighbors. Part (a) shows an UKNN 1 embedding of the 2D-S data set. Part (b) shows the embedding of the same data set with UKNN 2. The colors of both embeddings show

\[
\text{Fig. 3. Upper part: (a) UKNN 1, and (b) UKNN 2 embedding with } K = 5 \text{ on 2D-S. Lower part: UKNN 1 (c), and UKNN 2 embedding (d) with } K = 5 \text{ of the spiral data set.}
\]

2) 3D-S: In the following, we will test UKNN regression on a 3-dimensional \( S \) data set (3D-S). The variant without a hole consists of 500 data points, the variant with a hole in the middle consists of 400 points. Figure 4 (a) shows the order of elements of the 3D-S data set without a hole at the

\[
\text{Fig. 4. Results of UKNN on 3D-S: (a) the unsorted } S \text{ at the beginning, (b) the embedded } S \text{ with UKNN 1 and } K = 10, \text{ (c) the embedded } S \text{ with UKNN 2 and } K = 10, \text{ and (d) a variant of } S \text{ with a hole embedded with UKNN 2.}
\]
beginning. The corresponding embedding with UKNN 1 and $K = 10$ is shown in Part (b) of the figure. Again, similar colors correspond to neighbored points in latent space. Part (c) of Figure 4 shows the UKNN 2 embedding achieving similar results. Also on the UKNN embedding of the $S$ data set with hole, see the lower right Part (d) of the figure, a reasonable neighbored assignments can be observed. Quantitative results for the DSRE are reported in Table I.

3) USPS Digits: Last, we experimentally test UKNN regression on test problems from the USPS digits data set [5]. For this sake we take 100 data samples of 256-dimensional (16 x 16 pixels) pictures of handwritten digits of 2’s and 5’s. We embed a one-dimensional manifold, and show the high-

dimensional data that is assigned to every 14th embedded latent point. Similar digits are neighbored in latent space.

Fig. 5. UKNN 2 embeddings of USPPS digits: (a) 2’s, and (b) 5’s. Digits are shown that are assigned to every 14th embedded latent point. Similar digits are neighbored in latent space.

4) DSRE Comparison: Last, we compare the DSRE achieved by both strategies, compared to the initial DSRE on all test problems. For the USPS digits data set we choose the number 7. Table I shows the experimental results of three settings for the neighborhood size $K$. The lowest DSRE on each problem is highlighted with bold figures. After application of the iterative strategies the DSRE is significantly lower than initially. Increasing $K$ results in higher DSREs. UKNN 1 achieves lower DSREs than UKNN 2, with exception of 2D-

TABLE I

<table>
<thead>
<tr>
<th></th>
<th>2D-S</th>
<th>3D-S</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>init</td>
<td>201.6</td>
<td>290.0</td>
</tr>
<tr>
<td>UKNN 1</td>
<td>19.6</td>
<td>27.1</td>
</tr>
<tr>
<td>UKNN 2</td>
<td>29.2</td>
<td>70.1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>3D-S hole</th>
<th>digits (7)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>init</td>
<td>577.0</td>
<td>727.6</td>
</tr>
<tr>
<td>UKNN 1</td>
<td>80.7</td>
<td>108.1</td>
</tr>
<tr>
<td>UKNN 2</td>
<td>101.8</td>
<td>204.4</td>
</tr>
</tbody>
</table>

REFERENCES