Locally Linear Factorization Machines

Chenghao Liu[1], **Teng Zhang**[1], **Peilin Zhao**[2], **Jun Zhou**[2], **Jianling Sun**[1]
[1]School of Computer Science and Technology, Zhejiang University, China
[2]Artificial Intelligence Department, Ant Financial Services Group, China
{twinsken, faramita, sunjl}@zju.edu.cn, {peilin.zpl, jun.zhoujun}@antfin.com

Abstract

Factorization Machines (FMs) are a widely used method for efficiently using high-order feature interactions in classification and regression tasks. Unfortunately, despite increasing interests in FMs, existing work only considers high order information of the input features which limits their capacities in non-linear problems and fails to capture the underlying structures of more complex data. In this work, we present a novel Locally Linear Factorization Machines (LLFM) which overcomes this limitation by exploring local coding technique. Unlike existing local coding classifiers that involve a phase of unsupervised anchor point learning and predefined local coding scheme which is suboptimal as the class label information is not exploited in discovering the encoding and thus can result in a suboptimal encoding for prediction, we formulate a joint optimization over the anchor points, local coding coordinates and FMs variables to minimize classification or regression risk. Empirically, we demonstrate that our approach achieves much better predictive accuracy than other competitive methods which employ LLFM with unsupervised anchor point learning and predefined local coding scheme.

1 Introduction

Interactions between features play an important role in many classification and regression tasks. One of the widely used approach to leverage such interactions is the polynomial kernel[Friedman *et al.*, 2001] which implicitly maps the data via the kernel trick. However, the cost of storing and evaluating the model is expensive especially for large datasets. This is sometimes called the curse of kernelization [Wang *et al.*, 2010]. To address this issue, Factorization machines (FMs)[Rendle, 2010; 2012] have been proposed to model the high order nested interactions with factorized interaction parameters. The model estimation can be computed in linear time and they only depend on a linear number of parameters. This allows direct optimization and storage of model parameters without the need of storing any training data.

Despite their great success, FMs only consider the high order information of the input features which limits their capacities in non-linear problems and fails to capture the underlying structures of more complex data. Specifically, taking classification task into consideration, not all problems are approximately linearly separable after quadratic mapping. In most cases, real data naturally groups into clusters and lies on nearly disjoint lower dimensional manifolds and thus the original FMs are inapplicable. One solution to address this limitation is the locally linear classifiers [Ladicky and Torr, 2011; Yu *et al.*, 2009; Mao *et al.*, 2015] which leverage the manifold geometric structure to learn a nonlinear function which can be effectively approximated by a linear function with an coding under appropriate localization conditions. Since a nonlinear manifold behaves linearly in the local neighborhood, data on the manifold can be encoded locally in a local coordinate system established by a set of anchor points. Each data point can then be approximated with a linear combination of surrounding anchor points, and the weights are local coding coordinates which can be used for subsequent model training.

Although local coding methods provide a powerful tool for approximating data on the nonlinear manifold, the model performance of locally linear classifiers depends heavily on the quality of the local coding coordinates and the anchor points. The existing locally linear classifiers learn the local coding coordinates, the anchor points and classifiers in two separate steps. They first compute the anchor points with some unsupervised learning method that does not take class label information into account and encode the training data with a predefined local coding scheme, and then feed the results of encoding to the downstream supervised classifier training process. One major issue with this decoupled approach is that it only leverages the anchor points and local coding coordinates to improve classifier training task, but not reverse. This two-step procedure is rather suboptimal as the class label information is not used in discovering the anchor points and local coding coordinates, which is clearly not an optimal encoding for classification task as the two methods are not tightly coupled to fully exploit their potential.

In this paper, we present a novel Locally Linear Factorization Machines (LLFM) which is capable of learning complex non-linear data by exploring local coding technique. Unlike existing local coding classifiers that involve a two-step procedure, a joint optimization is formulated over the anchor points, local coding coordinates and FMs variables to minimize classification or regression risk simultaneously. Specifically, our learning method is capable of refining local coding scheme which adaptively choose the number of nearest anchor points and the corresponding approximation weight according to the current data point $\mathbf{x}$ and the anchor points. With these local coding coordinates, we adopt stochastic gradient descent to efficiently learn both the anchor points and the FMs model simultaneously. Experimental results on benchmark datasets show that our proposed LLFM approach with joint optimization (LLFM-JO) outperforms state-of-the-art methods with predefined fixed local coding scheme (LLFM-DO) or unsupervised anchor point learning(LLFM-APL).

The rest of this paper is organized as follows. We first review related work about FMs and local coding method, followed by introducing the proposed Locally Linear Factorization Machines model. Then we present our joint optimization method with respect to the local coding coordinates, anchor points and FMs parameters. Finally, we discuss empirical results and conclude this work.

2 Related Work

2.1 Factorization Machines

A standard 2-order FMs model takes the form:

$$f^{FM}(\mathbf{x}) = \sum_{j=1}^{p} w_j x_j + \sum_{j=1}^{p} \sum_{j'=j+1}^{p} x_j x_{j'} \sum_{f=1}^{k} v_{j,f} v_{j',f},$$

where p is the dimensionality of feature vector $\mathbf{x} \in \mathbb{R}^p$, $k \ll p$ is a hyper-parameter that denotes the dimensionality of latent factors, and $w_j, v_{j,f}$ are the model parameters to be estimated, i.e., $\Theta = \{w_1, \ldots, w_p, v_{1,1}, \ldots, v_{p,k}\} = \{\mathbf{w} \in \mathbb{R}^p, \mathbf{V} \in \mathbb{R}^{p\times k}\}$. It is equivalent to the following simple equation:

$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + \sum_{j=1}^{p} \sum_{j'=j+1}^{p} (\mathbf{V}\mathbf{V}^\top)_{jj'} x_j x_{j'}.$$

The gradient can be derived

$$\frac{\partial f^{FM}}{\partial \theta} = \begin{cases} x_j & \theta \text{ is } w_j \\ x_j \sum_{i\neq j} v_{i,f} x_i & \theta \text{ is } v_{j,f} \end{cases} \qquad (1)$$

The main advantage of FMs compared to the polynomial kernel in SVM [Vapnik, 2013] is the pairwise feature interaction weight matrix $\mathbf{Z} = \mathbf{V}\mathbf{V}^\top \in \mathbb{S}^{p\times p}$, where the number of parameters to estimate is reduced from p^2 to kp by utilizing the factorized form. In addition, this factorization form helps to drop the prediction cost to linear runtime especially under sparsity condition. Given a training set $[\mathbf{x}_1, \ldots, \mathbf{x}_n]^\top \in \mathbb{R}^{n\times p}$ and corresponding targets $p[y_1, \ldots, y_n]^\top \in \mathbb{R}^n$, model parameters Θ can be learned by using the principle of empirical risk minimization and solving the following non-convex problem

$$\min_{\mathbf{w}\in\mathbb{R}^p, \mathbf{V}\in\mathbb{R}^{p\times k}} \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f(\mathbf{x}_i)) + \frac{\beta}{2} R(\Theta), \qquad (2)$$

where $R(\Theta)$ is the regularization term and ℓ is a convex loss function incurred. Although the objective is a non-convex problem, this optimization problem can be efficiently solved by many off-the-shelf approaches [Rendle, 2012].

In recent years, some developments in FMs have been proposed to efficiently optimize FMs model. [Blondel *et al.*, 2015] gives a convex formulation of FMs based on the nuclear norm and proposed a two-block coordinate descent algorithm. [CHIN *et al.*,] proposes an alternating minimization algorithm based on Newton method. [Lin and Ye, 2016] designs a construction of an estimation sequence endowed with a CI-RIP condition and proposesp an efficient single-pass alternating updating framework for generalized FMs. [Blondel *et al.*, 2016b] discusses the relationship between high order FMs and ANOVA kernel. [Blondel *et al.*, 2016a] proposes linear time dynamic programming algorithms for evaluating the ANOVA kernel and the algorithm for training arbitrary-order FMs.

2.2 Locally Linear Coding

Local coding methods offer a powerful tool for approximating data on the nolinear manifold. All these methods employ a set of anchor points to encode data as a linear combination of surrounding anchor points, so as to minimize the approximation error. Specifically, let $\mathbf{Z} = \{\mathbf{z}_i\}_{i=1}^m$ denote the set of m anchor points, any point $\mathbf{x}$ is then approximated as $\mathbf{x} \approx \sum_{i=1}^m \gamma_{\mathbf{x},\mathbf{z}_i} \mathbf{z}_i$, where $\gamma_{\mathbf{x},\mathbf{z}_i}$ is the local coding coordinates, depicting the degree of membership of $\mathbf{x}$ to the ith anchor point $\mathbf{z}_i$, constrained by $\sum_{i=1}^m \gamma_{\mathbf{x},\mathbf{z}_i} = 1$. Different encoding schemes have been proposed in literature, [Liu *et al.*, 2011; Van Gemert *et al.*, 2008] proposes local soft-assignment coding, which is defined as:

$$\gamma_{\mathbf{x},\mathbf{z}_i} = \begin{cases} \frac{exp(-cd(\mathbf{x},\mathbf{z}_i))}{\sum_{j\in N_k(\mathbf{x})} exp(-cd(\mathbf{x},\mathbf{z}_j))} & j \in N_k(\mathbf{x}) \\ 0 & otherwise \end{cases} \qquad (3)$$

where $d(\cdot,\cdot)$ is the distance function, and $N_k(\mathbf{x})$ denotes the set of indices of k-nearest anchor points to $\mathbf{x}$. Notice here only the k-nearest anchor points are considered for encoding data point $\mathbf{x}$ with k nonzero weights, and the remaining weights for anchor points are all set to zero. Other local coding methods include local coordinate coding [Yu *et al.*, 2009], inverse Euclidian distance based weighting [Van Gemert *et al.*, 2008; Ladicky and Torr, 2011], etc.

A number of locally linear classifiers have been proposed based on local coding methods. [Ladicky and Torr, 2011] calculates the local coordinates with fixed and predefined local coding scheme, and them treats the local coordinates as weights for assigning training data into different local regions. Separate model are trained for each local region and combined to form a local linear classifier. [Gu and Han, 2013] adopts K-means to partition the data into clusters and then trains a linear SVM for each cluster. Meanwhile, it requires each cluster's model to align with a global model, which can be treated as a type of regularization. Unlike these two-step methods, a joint optimization is formulated over the anchor points, local coding coordinates and FMs variables in our work.

3 Locally Linear Factorization Machines

FMs haven been found successful in many prediction tasks, including classification, regression and ranking. However, they only consider the second order information of the input features which limits their capacity in non-linear problems and fails to capture the underlying structures of complex data.

One intuitive idea for addressing this limitation is to leverage the manifold geometric structure to learn a nonlinear function which can be effectively approximated by a linear function with an coding under appropriate localization conditions. In another word, we assume that in a sufficiently small region the decision boundary is approximately linear and each data point $\mathbf{x}$ can then be approximated with a linear combination of surrounding anchor points, which are usually called local codings scheme. To encode this local linearity with FMs, the model parameters Θ should vary according to the location of the point $\mathbf{x}$ in the feature space as:

$$f^{LLFM}(\mathbf{x}) = \mathbf{w}(\mathbf{x})^\top \mathbf{x} + \sum_{j=1}^{p} \sum_{j'=j+1}^{p} (\mathbf{V}(\mathbf{x})\mathbf{V}(\mathbf{x})^\top)_{jj'} x_j x_{j'}. \tag{4}$$

According to [Yu *et al.*, 2009; Ladicky and Torr, 2011], smoothness and constrained curvature of the decision boundary implies that the function $\mathbf{w}(\mathbf{x})$ and $\mathbf{V}(\mathbf{x})$ are Lipschitz in the feature space $\mathbf{x}$. Thus, for a local coding scheme defined by anchor points, we can approximate the weight function $\mathbf{w}(\mathbf{x}), \mathbf{V}(\mathbf{x})$ of FMs using local coding as:

$$\mathbf{w}(\mathbf{x}) \approx \sum_{i=1}^{m} \gamma_{\mathbf{x},\mathbf{z}_i} \mathbf{w}_{\mathbf{z}_i}, \mathbf{V}(\mathbf{x})\mathbf{V}(\mathbf{x})^\top \approx \sum_{i=1}^{m} \gamma_{\mathbf{x},\mathbf{z}_i} (\mathbf{V}_{\mathbf{z}_i} \mathbf{V}_{\mathbf{z}_i}^\top).$$

Substituting these equations into the prediction function $f^{LLFM}(\mathbf{x})$, we obtain:

$$\begin{aligned}
& f^{LLFM}_{\gamma_{\mathbf{x},\mathbf{z}},\mathbf{Z},\Theta_{LLFM}}(\mathbf{x}) \\
=& \sum_{i=1}^{m} \gamma_{\mathbf{x},\mathbf{z}_i} \mathbf{w}_{\mathbf{z}_i}^\top \mathbf{x} + \sum_{i=1}^{m} \gamma_{\mathbf{x},\mathbf{z}_i} \sum_{j=1}^{p} \sum_{j'=j+1}^{p} (\mathbf{V}_{\mathbf{z}_i} \mathbf{V}_{\mathbf{z}_i}^\top)_{jj'} x_j x_{j'} \\
=& \sum_{i=1}^{m} \gamma_{\mathbf{x},\mathbf{z}_i} \left(\mathbf{w}_{\mathbf{z}_i}^\top \mathbf{x} + \sum_{j=1}^{p} \sum_{j'=j+1}^{p} (\mathbf{V}_{\mathbf{z}_i} \mathbf{V}_{\mathbf{z}_i}^\top)_{jj'} x_j x_{j'} \right) \\
=& \sum_{i=1}^{m} \gamma_{\mathbf{x},\mathbf{z}_i} f^{FM}_{\mathbf{z}_i}(\mathbf{x}) \qquad (5)
\end{aligned}$$

where $\Theta_{LLFM} = \{\Theta_{\mathbf{z}_i}\}_{i=1}^m = \{\mathbf{w}_{\mathbf{z}_i}, \mathbf{V}_{\mathbf{z}_i}\}_{i=1}^m$ are the model parameters corresponding to the anchor point $\mathbf{z}_i$. This transformation can be seen as a finite kernel transforming a p-dimensional problem into a mp-dimensional one. It can also be interpreted as defining a locally linear Factorization Machines as the weighted average of m separate FMs with respect to each anchor point, where the weights are determined by the local coding coordinates. Let $\gamma_{\mathbf{x},\mathbf{z}} = [\gamma_{\mathbf{x},\mathbf{z}_1}, \cdots, \gamma_{\mathbf{x},\mathbf{z}_m}]^\top$ and $\mathbf{f}_{\mathbf{Z}}^{FM} = [f_{\mathbf{z}_1}^{FM}(\mathbf{x}), \cdots, f_{\mathbf{z}_m}^{FM}(\mathbf{x})]^\top$ be m dimensional vectors by stacking the m FMs models. The prediction function in Equation (5) can be written as $f^{LLFM}_{\gamma_{\mathbf{x},\mathbf{z}},\mathbf{Z},\Theta_{LLFM}}(\mathbf{x}) = \gamma_{\mathbf{x},\mathbf{z}}^\top \mathbf{f}_{\mathbf{Z}}^{FM}$.

4 Joint Optimization Method for Locally Linear Factorization Machines

To evaluate $f^{LLFM}_{\gamma_{\mathbf{x},\mathbf{z}},\mathbf{Z},\Theta_{LLFM}}(\mathbf{x})$ for each data point $\mathbf{x}$, we need to calculate the corresponding local coding coordinates $\gamma_{\mathbf{x},\mathbf{z}}$, that is further depend on the anchor points $\mathbf{z}$ being used and the local coding scheme, which means the prediction function $f^{LLFM}_{\gamma_{\mathbf{x},\mathbf{z}},\mathbf{Z},\Theta_{LLFM}}(\mathbf{x})$ depends on the model parameters Θ_{LLFM}, the anchor point variable $\mathbf{z}$ and local coding coordinates $\gamma_{\mathbf{x},\mathbf{z}}$. This leads to a natural two-step approach taken by existing methods [Ladicky and Torr, 2011; Yu *et al.*, 2009; Gu and Han, 2013], which first estimate the anchor points for each data point by adopting K-means clustering and evaluate the local coding coordinates with a predefined scheme by utilizing exponential decay scheme or inversely-proportional decay scheme, and then feeds the anchor points and the local coding coordinates to the downstream supervised model training. This two-step learning procedure is inconsistent with the objective function and rather suboptimal as the prediction information is not used in discovering the anchor points and the local coding scheme. This motivates a joint optimization method for LLFMs. Using a similar formulation to Equation (2), we define the LLFM optimization problem as follows:

$$\min_{\gamma_{\mathbf{x},\mathbf{z}},\mathbf{Z},\Theta_{LLFM}} \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f^{LLFM}_{\gamma_{\mathbf{x}_i,\mathbf{z}},\mathbf{Z},\Theta_{LLFM}}(\mathbf{x}_i)) + \frac{\beta}{2} R(\Theta_{LLFM}). \tag{6}$$

Note that here the local coding coordinates $\gamma_{\mathbf{x},\mathbf{z}}$ and anchor points $\mathbf{z}$ are treated as the variables to be optimized. Previous approaches [Mao *et al.*, 2015] select local coding coordinates without supervised information, which is not guaranteed to retain the discriminative information for prediction. Consequently, the selected anchor points and the local coding coordinates may not be optimal for the model being trained. On the contrary, the use of embedded optimization for the local coding scheme, the anchor points and the model parameters in Equation (6) is crucial to the success of our approach. The objective function in Equation (6) is a non-convex optimization problem when considering variables $\gamma_{\mathbf{x},\mathbf{z}}, \mathbf{Z}, \Theta_{LLFM}$ together. Therefore, we iteratively optimize $\gamma_{\mathbf{x},\mathbf{z}}, \mathbf{Z}, \Theta_{LLFM}$ until convergence to obtain a local minimum.

4.1 Local Coding Coordinates Optimization Method

To start off, we first present our optimization method for the local coding coordinates $\gamma_{\mathbf{x},\mathbf{z}}$. Take the weight function $\mathbf{w}(\mathbf{x})$ into consideration, recall we seek to find the best local approximation in a sense of minimizing the distance between this approximation and the ground truth. Assume that for any data point $\mathbf{x}$, the ground truth holds that $\mathbf{w}_{\mathbf{x}} = \mathbf{w}(\mathbf{x}) + \epsilon_{\mathbf{x}}$, where $\mathbf{w}(\cdot)$ is a Lipschitz continuous function that for any $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^p$ it holds that $|\mathbf{w}(\mathbf{x}_1) - \mathbf{w}(\mathbf{x}_2)| \le L \cdot d(\mathbf{x}_1, \mathbf{x}_2)$ for some predefined distance function $d(\cdot,\cdot)$ and $\epsilon_{\mathbf{x}}$ is a noise term that $\mathbb{E}[\epsilon_{\mathbf{x}}|\mathbf{x}] = 0$ and $|\epsilon_{\mathbf{x}}| \le b$ for some given $b > 0$. Our task is to estimate $\mathbf{w}(\mathbf{x})$, where we restrict the estimator $\hat{\mathbf{w}}(\mathbf{x})$ to be of the form $\hat{\mathbf{w}}(\mathbf{x}) = \sum_{i=1}^m \gamma_{\mathbf{x},\mathbf{z}_i} \mathbf{w}_{\mathbf{z}_i}$. That is, the estimator is a weighted average of the anchor points. Formally, the objective is to minimize the absolute distance between our approximation and the ground truth $\mathbf{w}(\mathbf{x})$, we need to solve

$$\min_{\gamma_{\mathbf{x},\mathbf{z}}} |\sum_{i=1}^{m} \gamma_{\mathbf{x},\mathbf{z}_i} \mathbf{w}_{\mathbf{z}_i} - \mathbf{w}(\mathbf{x})| s.t. \sum_{i=1}^{m} \gamma_{\mathbf{x},\mathbf{z}_i} = 1; \gamma_{\mathbf{x},\mathbf{z}_i} \ge 0, \forall i.$$

Decomposing the above objective into a sum of bias and variance terms, we can transforms it into

$$\begin{aligned}
& |\sum_{i=1}^{m} \gamma_{\mathbf{x},\mathbf{z}_i} \mathbf{w}_{\mathbf{z}_i} - \mathbf{w}(\mathbf{x})| \\
=& |\sum_{i=1}^{m} \gamma_{\mathbf{x},\mathbf{z}_i} \epsilon_{\mathbf{z}_i} + \sum_{i=1}^{m} \gamma_{\mathbf{x},\mathbf{z}_i} (\mathbf{w}(\mathbf{z}_i) - \mathbf{w}(\mathbf{x}))|
\end{aligned}$$

$$\le |\sum_{i=1}^{m} \gamma_{\mathbf{x},\mathbf{z}_i} \epsilon_{\mathbf{z}_i}| + |\sum_{i=1}^{m} \gamma_{\mathbf{x},\mathbf{z}_i} (\mathbf{w}(\mathbf{z}_i) - \mathbf{w}(\mathbf{x}))|$$

$$\le |\sum_{i=1}^{m} \gamma_{\mathbf{x},\mathbf{z}_i} \epsilon_{\mathbf{z}_i}| + L \sum_{i=1}^{m} \gamma_{\mathbf{x},\mathbf{z}_i} d(\mathbf{z}_i, \mathbf{x}) \tag{7}$$

By Hoeffding's inequality it follows that $|\sum_{i=1}^{m} \gamma_{\mathbf{x},\mathbf{z}_i} \epsilon_{\mathbf{z}_i}| \le C\|\gamma_{\mathbf{x},\mathbf{z}}\|_2$ for $C = b\sqrt{2\log(\frac{2}{\delta})}$, w.p. at least $1-\delta$. Inequality (7) yield a upper bound guarantee for solving the original objective with high probability, we can formulate the new problem as the following optimization:

$$\min_{\gamma_{\mathbf{x},\mathbf{z}}} C\|\gamma_{\mathbf{x},\mathbf{z}}\|_2 + \gamma_{\mathbf{x},\mathbf{z}}^{\top}\mathbf{u} \quad s.t. \sum_{i=1}^{m} \gamma_{\mathbf{x},\mathbf{z}_i} = 1; \gamma_{\mathbf{x},\mathbf{z}_i} \ge 0, \forall i. \tag{8}$$

where $\mathbf{u} = \{Ld(\mathbf{z}_1, \mathbf{x}), \cdots, Ld(\mathbf{z}_m, \mathbf{x})\}$. Its Lagrangian is:

$$\mathcal{L}(\gamma_{\mathbf{x},\mathbf{z}}, \theta, \lambda)$$

$$= p\|\gamma_{\mathbf{x},\mathbf{z}}\|_2 + \gamma_{\mathbf{x},\mathbf{z}}^{\top}\mathbf{u} + \lambda(1 - \sum_{i=1}^{m} \gamma_{\mathbf{x},\mathbf{z}_i}) - \sum_{i=1}^{m} \theta_i \gamma_{\mathbf{x},\mathbf{z}_i}$$

where $\lambda \in \mathbb{R}$ and $\theta_1, \cdots, \theta_m \ge 0$ are the Lagrange multipliers. This optimization problem has a convex objective function and feasible affine constraints. Thus, satisfying the KKT conditions is a necessary and sufficient condition for finding the problem's optimum. Setting the partial derivative of $\mathcal{L}(\gamma_{\mathbf{x},\mathbf{z}}, \theta, \lambda)$ with respect to $\gamma_{\mathbf{x},\mathbf{z}}$ to zero gives:

$$\frac{\gamma_{\mathbf{x},\mathbf{z}_i}}{\|\gamma_{\mathbf{x},\mathbf{z}}\|_2} = \lambda - \mathbf{u}_i + \theta_i. \tag{9}$$

Let $\gamma_{\mathbf{x},\mathbf{z}}^{\star}$ be the optimal solution. According to the KKT conditions, if $\gamma_{\mathbf{x},\mathbf{z}_i}^{\star} > 0$ it follows that $\theta_i = 0$. Otherwise, $\gamma_{\mathbf{x},\mathbf{z}_i}^{\star} = 0$ and $\lambda \le \mathbf{u}_i$. Substituting it into the equality constraint $\sum_i \gamma_{\mathbf{x},\mathbf{z}_i}^{\star} = 1$, for any $\gamma_{\mathbf{x},\mathbf{z}_i}^{\star} > 0$, we have

$$\gamma_{\mathbf{x},\mathbf{z}_j}^{\star} = \frac{\lambda - \mathbf{u}_j}{\sum_{\gamma_{\mathbf{x},\mathbf{z}_i}^{\star} > 0}(\lambda - \mathbf{u}_i)} = \frac{\lambda - Ld(\mathbf{z}_j, \mathbf{x})}{\sum_{\gamma_{\mathbf{x},\mathbf{z}_i}^{\star} > 0}(\lambda - Ld(\mathbf{z}_i, \mathbf{x}))} \tag{10}$$

It demonstrates that the optimal weight $\gamma_{\mathbf{x},\mathbf{z}_i}^{\star}$ is proportional to $-d(\mathbf{z}_i, \mathbf{x})$, whose weight decay is quite slow compared to the popular exponential decay scheme or inversely-proportional decay scheme that used in [Mao *et al.*, 2015; Gu and Han, 2013; Ladicky and Torr, 2011]. It also shows that parameter λ has a cutoff effect that only nearest anchor points that $\lambda - Ld(\mathbf{z}_i, \mathbf{x}) > 0$ are considered for encoding data point $\mathbf{x}$, the weights for the remaining anchor points are all set to zero. This is consistent with the previous predefined local coding scheme. Note that the objective (8) is a convex optimization problem, which can be efficiently solved using off-the-shelf toolbox. Here we follow the method in [Anava and Levy, 2016]. The key idea is to greedily add neighbors according to their distance from $\mathbf{x}$ until a stopping condition is achieved. Our algorithm is presented in Algorithm 1.

Denote by k the number of nonzero weights which correspond to the k smallest value of $\mathbf{u}$. Squaring and summing Equation (9) over all the nonzero elements of $\gamma_{\mathbf{x},\mathbf{z}}^{\star}$, we have

$$1 = \sum_{\gamma_{\mathbf{x},\mathbf{z}_i}^{\star} > 0} \frac{\gamma_{\mathbf{x},\mathbf{z}_i}^{\star}}{\|\gamma_{\mathbf{x},\mathbf{z}}^{\star}\|_2} = \sum_{\gamma_{\mathbf{x},\mathbf{z}_i}^{\star} > 0} (\lambda - \mathbf{u}_i)^2 \tag{11}$$

Algorithm 1 Local Coding Coordinates (LLC) Optimization Algorithm

```
Input: data point $\mathbf{x}$ and anchor points $\mathbf{Z} = \{\mathbf{z}_1, \cdots, \mathbf{z}_m\}$
Initialization: $\lambda_0 = \mathbf{u}_1 + 1$, $k = 0$ and compute the vector of ascending ordered distance $\mathbf{u} \in \mathbb{R}^m$
while $\lambda_k > \mathbf{u}_{k+1}$ and $k \le n - 1$ do
    Update $k \leftarrow k + 1$
    Compute $\lambda_k$ based on (12)
end while
Output: The number of nearest anchor points $k$, compute the local coding coordinates $\gamma_{\mathbf{x},\mathbf{z}}$ based on (10)
```

which is equivalent to $k\lambda^2 - 2\lambda\sum_{i=1}^{k}\mathbf{u}_i + (\sum_{i=1}^{k}\mathbf{u}_i^2 - 1) = 0$. Solving this quadratic equation with respect to λ and ignoring the solution that violate $\gamma_{\mathbf{x},\mathbf{z}_i}^{\star} \ge 0$, we get

$$\lambda = \frac{1}{k}(\sum_{i=1}^{k}\mathbf{u}_i + \sqrt{k + (\sum_{i=1}^{k}\mathbf{u}_i)^2 - k\sum_{i=1}^{k}\mathbf{u}_i^2}) \tag{12}$$

4.2 Anchor Points and FMs Optimization Method

For the anchor points and FMs optimization, we apply the SGD method to the objective in (6). Specifically, at each iteration, we randomly sample a data point $\mathbf{x}$ and its corresponding target y, then we update the anchor points $\mathbf{Z}$ and FMs parameters Θ_{LLFM}. Since the data point $\mathbf{x}$ is approximate as a linear combination of its k-nearest anchor points, only the k-nearest anchor points need to be optimized. For updating anchor points $\mathbf{z}$, we take the partial derivative of the objective (6) with respect to $\mathbf{z}$ while fixing Θ_{LLFM}. The derivative $\frac{\partial\gamma_{\mathbf{x},\mathbf{z}}^{\top}}{\partial\mathbf{z}_i}$ is a $(p + p^2) \times m$ matrix, among which only k columns are non zero. The ith column is computed as:

$$\frac{L\mathbf{s}(\lambda - Ld(\mathbf{z}_i, \mathbf{x}) - \sum_{\gamma_{\mathbf{x},\mathbf{z}_j} > 0}(\lambda - Ld(\mathbf{z}_j, \mathbf{x})))}{(\sum_{\gamma_{\mathbf{x},\mathbf{z}_j} > 0}(\lambda - Ld(\mathbf{z}_j, \mathbf{x})))^2} \tag{13}$$

where $\mathbf{s} = \frac{\partial d(\mathbf{z}_i, \mathbf{x})}{\partial\mathbf{z}_i}$. The other nonzero columns except the ith column are computed as

$$-\frac{L\mathbf{s}}{(\sum_{\gamma_{\mathbf{x},\mathbf{z}_j} > 0}(\lambda - Ld(\mathbf{z}_j, \mathbf{x})))^2} \tag{14}$$

where $\mathbf{z}_j$ also belongs to the k-nearest neighbors of $\mathbf{x}$ and it is not equal to $\mathbf{z}_i$. Then the ith anchor point $\mathbf{z}_i$ is updated as:

$$\mathbf{z}_i \leftarrow \mathbf{z}_i + \frac{\rho_{\mathbf{z}}}{t + t_0} \frac{\partial\gamma_{\mathbf{x},\mathbf{z}}^{\top}}{\partial\mathbf{z}_i} f_{\mathbf{Z}}^{FM} \frac{\partial\ell(y, f_{\gamma_{\mathbf{x},\mathbf{z}},\mathbf{Z},\Theta_{LLFM}}^{LLFM}(\mathbf{x}))}{\partial f_{\gamma_{\mathbf{x},\mathbf{z}},\mathbf{Z},\Theta_{LLFM}}^{LLFM}(\mathbf{x})} \tag{15}$$

where $\rho_{\mathbf{z}}$ is the learning rate, t denotes the current iteration number and t_0 is a positive constant [Bordes *et al.*, 2009]. The optimal learning rate is denoted as $\frac{\rho}{t+t_0}$ [Shalev-Shwartz *et al.*, 2007]. Note that our method works well with different loss function and can be applied to classification task or regression task, which only needs to minorly modify the last term about the derivative of the loss function in update rule (15). The FMs variables Θ_{LLFM} can also be updated by utilizing SGD method (more details can be found in [Rendle, 2012; 2010]). Specifically, the update rules for FMs parameter $\theta_{\mathbf{z}_i}$ with respect to anchor point $\mathbf{z}_i$ is:

$$\theta_{\mathbf{z}_i} \leftarrow \theta_{\mathbf{z}_i} + \frac{\rho_\theta}{t+t_0}\Big(\frac{\partial f^{FM}_{\mathbf{z}_i}}{\partial \theta_{\mathbf{z}_i}} \gamma_{\mathbf{x},\mathbf{z}_i} \frac{\partial \ell(y, f^{LLFM}_{\gamma_{\mathbf{x},\mathbf{z}},\mathbf{Z},\Theta_{LLFM}}(\mathbf{x}))}{\partial f^{LLFM}_{\gamma_{\mathbf{x},\mathbf{z}},\mathbf{Z},\Theta_{LLFM}}(\mathbf{x})} + \frac{\partial R(\theta_{\mathbf{z}_i})}{\partial \theta_{\mathbf{z}_i}}\Big) \tag{16}$$

where $\frac{\partial f^{FM}_{\mathbf{z}_i}}{\partial \theta_{\mathbf{z}_i}}$ can be found in Equation (1). Algorithm 2 summarizes the proposed Locally linear Factorization Mahicnes with pJoint Optimization (LLFM-JO) method.

Algorithm 2 Locally Linear Factorization Mahicnes Joint Optimization Algorithm (LLFM-JO)

```
Input: Training data$\{(x_n, y_n\}_{n=1}^N$, the number of anchor points $m$ and parameters $\rho, \beta, L$
while no convergence do
    Sample a data point $\mathbf{x}$ randomly
    Compute the local coordinate $\gamma_{\mathbf{x},\mathbf{z}}$ according to Algorithm 1
    Compute the loss $\ell(y, f^{LLFM}_{\gamma_{\mathbf{x},\mathbf{z}},\mathbf{Z},\Theta_{LLFM}}(\mathbf{x}))$
    for each nearest anchor point $i$ with respect to data point $\mathbf{x}$ do
        Update the $i$th nearest anchor point of $\mathbf{x}$ via (15)
        Update the FMs model parameters with respect to this anchor point via (16)
    end for
end while
Output: Compute the local coding coordinates $\gamma_{\mathbf{x},\mathbf{z}_i}$ based on (10)
```

5 Experiments

In this section, we empirically investigate whether our proposed LLFM-JO method can achieve better performance compared to other state-of-the-art methods which employ LLFM method with unsupervised anchor point learning (LLFM-APL) and predefined local coding scheme (LLFM-DO) on benchmark datasets. Furthermore we examine the efficacy and efficiency of joint optimization.

5.1 Experimental Testbeds and Setup

We conduct our experiments on six public datasets. Table 1 gives a brief summary of these datasets. All the datasets are normalized to have zero mean and unit variance in each dimension. To make fair comparison, all the algorithms are conducted over 5 experimental runs of different random permutations. We apply logistic loss [Rendle, 2012] for training and evaluate the performance of our proposed methods for classification task by measuring accuracy and logistic loss. We adopt squared Euclidean distance function in local soft-assignment coding and our local coding coordinates optimization method. For parameter settings, we perform grid search to choose the best parameters for each algorithm on the training set.

5.2 Performance Comparison

In our experiments, we compare the following methods:

- **FM**: Factorization Machines with stochastic gradient descent optimization method, which is the baseline method we introduced in Section 2.1.
- **LLFM-DO**: Locally Linear Factorization Machines with Decoupled Optimization method. We first compute the anchor points by K-means clustering and encode the training data with local soft-assignment coding, and then estimate the LLFM model parameters. This method is a baseline to validate the efficacy of joint optimization over the anchor points, local coding coordinates and model parameters simultaneously.
- **LLFM-APL**: Locally Linear Factorization Machines with Anchor Point Learning method. We jointly estimate both the anchor points and FMs model parameters as described in Section 4.2 but using the fixed local soft-assignment coding scheme as Equation (3). This method is a baseline to validate the efficacy of local coding coordinates optimization method in algorithm 1.
- **LLFM-JO**: The proposed LLFM method with Joint Optimization method in algorithm 2.

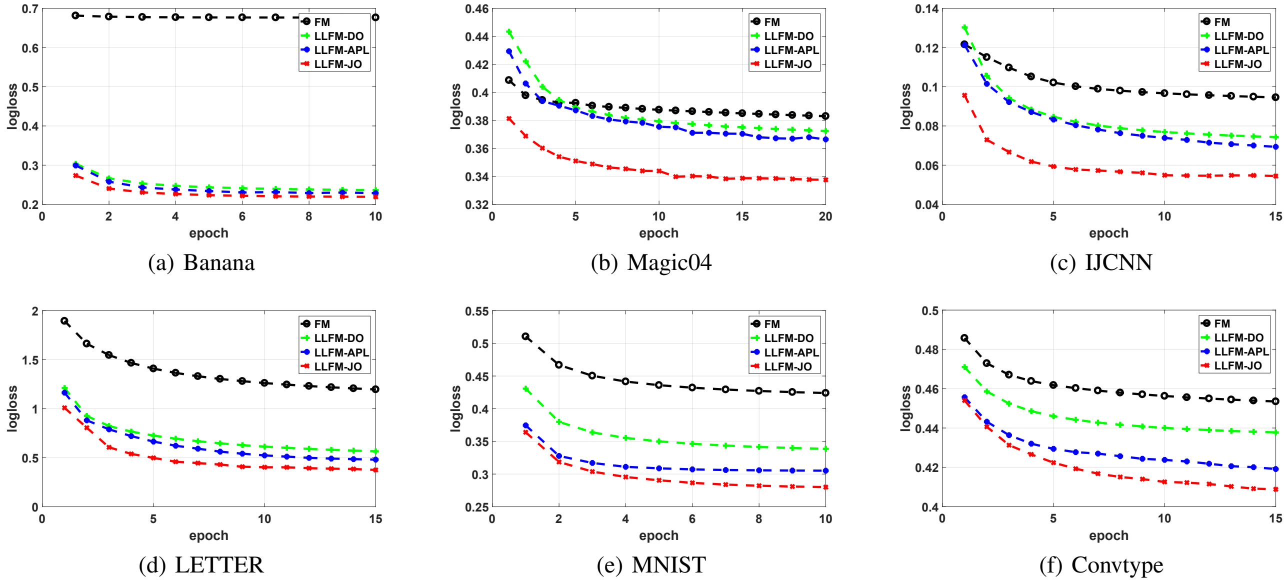
Dataset	#Training	#Test	#class
Banana	3533	1767	2
Magic04	12680	6340	2
IJCNN	49990	91701	2
LETTER	15000	5000	26
MNIST	60000	10000	10
Covtype	387342	193670	2

Table 1: Summary of datasets used in our experiments.

5.3 Experimental Results

The detailed comparison results are shown in Table 2 and Figure 1. We can observe that:

- All of LLFM methods that employ local linear coding techniques achieve better performance in terms of training loss, test loss and test accuracy compared with original FM method. This validates the efficacy of leveraging local coding techniques to improve the performance of FMs. Since local linear coding techniques could leverage the manifold geometric structure to learn a much more complex nonlinear model.
- It is not surprising that the performance of LLFM-JO, LLFM-APL is much better than LLFM-DO. It reveals the importance of jointly optimizing over the anchor points and the FMs model. It is even more critical on LETTER and MNIST datasets since the dimension of input features is much bigger and the LLFM-DO may be more risky and unreliable in finding the right direction for update at each iteration. Moreover, it can be clearly seen that the performance of LLFM-JO outperforms LLFM-APL especially on LETTER dataset, this clearly demonstrates the power of the local coding coordinate optimization method in algorithm 1.
- As evidenced by Table 2, LLFM-JO takes less test time compared with LLFM-DO and LLFM-APL. The reason is that test time scales linearly with the parameter



(a) Banana

(b) Magic04

(c) IJCNN

(d) LETTER

(e) MNIST

(f) Convtype

Figure 1: Epoch-wise demonstration of different algorithms with logloss on test data

Banana	Train loss	Test loss	Acc(%)	Time
FM	0.6810	0.6770	53.53	1×
LLFM-DO	0.2614	0.2371	89.80	5.84×
LLFM-APL	0.2573	0.2337	89.87	5.89×
LLFM-JO	**0.2396**	**0.2201**	**90.49**	7.00×

Magic04	Train loss	Test loss	Acc(%)	Test time
FM	0.3783	0.3804	83.98	1×
LLFM-DO	0.3661	0.3715	84.22	4.92×
LLFM-APL	0.3582	0.3670	84.44	5.14×
LLFM-JO	**0.3220**	**0.3364**	**85.95**	6.30×

IJCNN	Train loss	Test loss	Acc(%)	test time
FM	0.0949	0.0928	96.70	1×
LLFM-DO	0.0815	0.0760	97.58	4.97×
LLFM-APL	0.0786	0.0682	97.92	5.16×
LLFM-JO	**0.0564**	**0.0511**	**98.55**	4.87×

LETTER	Train loss	Test loss	Acc(%)	test time
FM	1.3899	1.2166	83.19	1×
LLFM	0.6966	0.5753	92.57	5.01×
LLFM-APL	0.6315	0.4819	93.70	4.92×
LLFM-JO	**0.5129**	**0.3958**	**94.56**	4.33×

MNIST	Train loss	Test loss	Acc(%)	test time
FM	0.4733	0.4247	95.61	1×
LLFM	0.3051	0.3331	95.40	5.01×
LLFM-APL	0.2586	0.3080	95.86	5.27×
LLFM-JO	**0.2558**	**0.2822**	**96.26**	1.74×

Covtype	Train loss	Test loss	Acc(%)	Test time
FM	0.4713	0.4557	78.42	1×
LLFM-DO	0.4509	0.4374	79.70	6.83×
LLFM-APL	0.4312	0.4178	80.34	6.87×
LLFM-JO	**0.4270**	**0.4130**	**80.78**	4.99×

Table 2: Comparison of different algorithms in terms of training loss, test loss, classification accuracy and test time (normalized to test time of FM)

k, LLFM model need to compute k FM model with respect to k nearest anchor points for prediction. However, LLFM-DO and LLFM-APL are restricted to choose one value of k during the locally linear coding procedure, LLFM-JO method could choose k adaptively, which is in line with our theoretical findings.

- To further examine the effectiveness of joint optimization, we record the objective function value as well as the logistic loss on the test data for epoch. Figure 1 shows the epoch-wise results of different algorithm. It can be seen clearly that the test log loss of all different algorithms are monotonically decreasing over the epochs and LLFM-JO outperforms the other algorithms significantly.

6 Conclusion and Future Work

In this work, we present a novel Locally Linear Factorization Machines (LLFM) model that exploring local coding technique. Unlike existing previous methods that learn the anchor points and local coding scheme separately before model training process we formulate a joint optimization over anchor point, local coding coordinate and FMs variables to minimize classification or regression risk. Our encouraging results show that LLFM-JO achieves better predictive accuracy than other competitive methods which employ LLFM with unsupervised anchor point learning and predefined local coding scheme. A shortcoming of current method is the high computation cost of model training due to the search and update of the local coding coordinates, it will be interesting to develop efficient local coding coordinates optimization approach. Future work includes combining LLFM model with learning to rank technique [Cao *et al.*, 2007] and AUC maximization[Zhao *et al.*, 2011].

Acknowledgments

This work was supported by State Street Zhejiang University Technology Centers.

References

[Anava and Levy, 2016] Oren Anava and Kfir Levy. k*-nearest neighbors: From global to local. In *Advances in Neural Information Processing Systems*, pages 4916–4924, 2016.

[Blondel *et al.*, 2015] Mathieu Blondel, Akinori Fujino, and Naonori Ueda. Convex factorization machines. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 19–35. Springer, 2015.

[Blondel *et al.*, 2016a] Mathieu Blondel, Akinori Fujino, Naonori Ueda, and Masakazu Ishihata. Higher-order factorization machines. In *Advances in Neural Information Processing Systems*, pages 3351–3359, 2016.

[Blondel *et al.*, 2016b] Mathieu Blondel, Masakazu Ishihata, Akinori Fujino, and Naonori Ueda. Polynomial networks and factorization machines: New insights and efficient training algorithms. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 850–858, 2016.

[Bordes *et al.*, 2009] Antoine Bordes, Léon Bottou, and Patrick Gallinari. Sgd-qn: Careful quasi-newton stochastic gradient descent. *Journal of Machine Learning Research*, 10(Jul):1737–1754, 2009.

[Cao *et al.*, 2007] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th international conference on Machine learning*, pages 129–136. ACM, 2007.

[CHIN *et al.*,] WEI-SHENG CHIN, BO-WEN YUAN, MENG-YUAN YANG, and CHIH-JEN LIN. An efficient alternating newton method for learning factorization machines.

[Friedman *et al.*, 2001] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics Springer, Berlin, 2001.

[Gu and Han, 2013] Quanquan Gu and Jiawei Han. Clustered support vector machines. In *AISTATS*, pages 307–315, 2013.

[Ladicky and Torr, 2011] Lubor Ladicky and Philip Torr. Locally linear support vector machines. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 985–992, 2011.

[Lin and Ye, 2016] Ming Lin and Jieping Ye. A non-convex one-pass framework for generalized factorization machine and rank-one matrix sensing. In *Advances in Neural Information Processing Systems*, pages 1633–1641, 2016.

[Liu *et al.*, 2011] Lingqiao Liu, Lei Wang, and Xinwang Liu. In defense of soft-assignment coding. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2486–2493. IEEE, 2011.

[Mao *et al.*, 2015] Xue Mao, Zhouyu Fu, Ou Wu, and Weiming Hu. Optimizing locally linear classifiers with supervised anchor point learning. In *IJCAI*, pages 3699–3706, 2015.

[Rendle, 2010] Steffen Rendle. Factorization machines. In *2010 IEEE International Conference on Data Mining*, pages 995–1000. IEEE, 2010.

[Rendle, 2012] Steffen Rendle. Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(3):57, 2012.

[Shalev-Shwartz *et al.*, 2007] Shai Shalev-Shwartz, Yoram Singer, and Nathan Srebro. Pegasos: Primal estimated subgradient solver for svm. In *Proceedings of the 24th international conference on Machine learning*, pages 807–814. ACM, 2007.

[Van Gemert *et al.*, 2008] Jan C Van Gemert, Jan-Mark Geusebroek, Cor J Veenman, and Arnold WM Smeulders. Kernel codebooks for scene categorization. In *European conference on computer vision*, pages 696–709. Springer, 2008.

[Vapnik, 2013] Vladimir Vapnik. *The nature of statistical learning theory*. Springer Science & Business Media, 2013.

[Wang *et al.*, 2010] Zhuang Wang, Koby Crammer, and Slobodan Vucetic. Multi-class pegasos on a budget. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 1143–1150, 2010.

[Yu *et al.*, 2009] Kai Yu, Tong Zhang, and Yihong Gong. Nonlinear learning using local coordinate coding. In *Advances in neural information processing systems*, pages 2223–2231, 2009.

[Zhao *et al.*, 2011] Peilin Zhao, Rong Jin, Tianbao Yang, and Steven C Hoi. Online auc maximization. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 233–240, 2011.