The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Metric-Based Auto-Instructor for Learning Mixed Data Representation

Songlei Jian,[†‡] **Liang Hu,**[‡] **Longbing Cao,**[‡] **Kai Lu**[†]
[†]College of Computer, National University of Defense Technology, China
[‡]Advanced Analytics Institute, University of Technology Sydney, Australia
jiansonglei@163.com, rainmilk@gmail.com, longbing.cao@uts.edu.au, kailu@nudt.edu.cn

Abstract

Mixed data with both categorical and continuous features are ubiquitous in real-world applications. Learning a good representation of mixed data is critical yet challenging for further learning tasks. Existing methods for representing mixed data often overlook the heterogeneous coupling relationships between categorical and continuous features as well as the discrimination between objects. To address these issues, we propose an auto-instructive representation learning scheme to enable margin-enhanced distance metric learning for a discrimination-enhanced representation. Accordingly, we design a *metric-based auto-instructor* (MAI) model which consists of two collaborative instructors. Each instructor captures the feature-level couplings in mixed data with fully connected networks, and guides the infinite-margin metric learning for the peer instructor with a contrastive order. By feeding the learned representation into both partition-based and density-based clustering methods, our experiments on eight UCI datasets show highly significant learning performance improvement and much more distinguishable visualization outcomes over the baseline methods.

Introduction

The performance of machine learning methods is heavily dependent on the choice of data representation (Bengio, Courville, and Vincent 2013). Mixed data, which contains both categorical (or discrete) features and continuous (or numerical) features, are ubiquitous in the real world, but only a limited number of representation learning methods for mixed data are available. At the *feature level*, a good representation should capture the heterogeneous coupling relationships (e.g., complex interactions, dependencies) between categorical and continuous features (Cao 2015). Further, at the *object level*, a good representation should express the discrimination and margins between objects to fertilize learning tasks. However, most current methods lying in the feature-level representation learning framework fail to consider object-level discrimination. This paper learns the mixed data representation to capture the feature-level couplings as well as object-level discrimination.

Most of the existing representation methods for mixed data ignore or only partially address the heterogeneous couplings between categorical and continuous features (Cao, Ou, and Yu 2012; Cao 2015). For example, *k-prototype* (Huang 1997) quantifies the distance between mixed types of objects with the summation of Euclidean distance for continuous parts and Hamming distance for categorical parts. This method treats features independently and ignores the couplings between features. Other methods discretize continuous data into categorical ones and then calculate the interactions in terms of categorical ways. For example, *mADD* (Ahmad and Dey 2007) models the interactions between continuous and categorical features by incorporating weights on distance which are calculated based on equal-length discretization. *SpectralCAT* (David and Averbuch 2012) and *CoupledMC* (Wang et al. 2015) both conduct *k*-means clustering on continuous features and use a validity index to choose clustering label as new continuous features. These methods calculate the couplings based on discretized continuous data which may fail to capture the distribution of the continuous data and lead to information loss.

Several model-based methods try to capture the heterogeneous couplings in a transformed data space. For example, *EGMCM* (Rajan and Bhattacharya 2016) learns the dependency between features with Gaussian mixture copulas based on the rank transformation of data during clustering, which may not only result in loss of information but also fail to capture the discriminative information between objects. In recent years, deep neural networks have enabled representation learning (Hinton and Salakhutdinov 2006; Vincent et al. 2010; Bengio, Courville, and Vincent 2013). For example, *autoencoder* (Vincent et al. 2010) is a typical neural model with full connections between features and hidden units. This network structure somehow captures the latent couplings between features and generates a distributed representation in the hidden layer. However, these methods focus on feature-level representation for each object without enhancing the discrimination between objects.

By constructing two compatibly encoded feature spaces to capture feature-level complex couplings in mixed data by neural models, we propose an auto-instructive representation learning scheme to enhance object-level discrimination in representation. Specifically, this auto-instructive representation learning scheme consists of two collaborative instructors. One instructor derives a contrastive order over a triplet by measuring their relative similarity (Frome et al. 2007). Then, the contrastive order serves as the supervision criteria to enable the metric learning component of the peer instructor. Alternately, the peer instructor derives orders from its encoding space as the supervision for the instructor. Under this auto-instructive learning process, these two instructors continuously improve the level of consensus between them and reach a stable state finally. As a result, an object discrimination-enhanced representation is learned.

To summarize, the main contributions of this work are:

- A comprehensive representation for mixed data simultaneously learns (1) the couplings between categorical features and continuous features at the *feature level*, and (2) the discrimination between objects at the *object level*.
- An auto-instructive representation learning scheme with two collaborative instructors learns more discriminative representation between objects by learning the margin-enhanced distance metric.
- A *metric-based auto-instructor* (MAI) model built on two compatible encoding feature spaces is devised to capture the feature-level couplings and enhance the object-level discrimination for the representation of mixed data.

Substantial experiments on eight real-world datasets are undertaken to evaluate our approach. The results prove its effectiveness from three aspects: (1) the representation learned by MAI achieves much better performance compared with other methods for clustering; (2) data representation analysis, including internal clustering criteria and data visualization, indicates that MAI produces more distinguishable representation which more clearly reflects the cluster structure in datasets; (3) the convergence test shows that MAI arrives at a stable state in a few iterations.

Related Work

Most methods achieve mixed representation by transformation and then learn feature couplings on the transformed data. Continuous feature discretization is a classic method for transforming mixed data (Dougherty et al. 1995). For example, *spectralCAT* (David and Averbuch 2012) discretizes continuous features in an automatic manner, in which cluster analysis is applied to each continuous feature while calculating the validity index. However, the transformed data is regarded as independent features and fed into clustering models. The couplings between categorical and continuous features are ignored (Cao 2015; Cao, Ou, and Yu 2012). *coupledMC* (Wang et al. 2015) as a state-of-the-art method takes discretization in *spectralCAT* to transform continuous features into categorical ones, and then uses the similarity between categorical values as continuous representation of mixed data and calculates the Pearson correlation between new continuous features. Due to the information loss of discretization, the similarity of categorical values and Pearson correlation cannot capture the real interactions between categorical and continuous features. Other methods also attempt to represent categorical features with numerical vectors, e.g., one-hot encoding, *CDE* (Jian et al. 2017) and *UFT* (Wei, Chow, and Chan 2015), but they cannot model the heterogeneous couplings in mixed data.

Other methods devise a specific distance or similarity for mixed data in clustering tasks. *K-prototype* (Huang 1997) is an extension to k-means for clustering mixed data by calculating the Euclidean distance for continuous features and the Hamming distance for categorical features and then uses the weighted summation as the final distance for mixed data. Many other methods (Ahmad and Dey 2007; Chen and He 2016; Ji et al. 2012; Jia and Cheung 2017) handle mixed data by proposing different distance measures for continuous or categorical features. Most of them consider the interactions between categorical and continuous features w.r.t. the co-occurrences between discretized continuous and categorical features.

Autoencoder has shown its superiority in producing semantically meaningful and well-separated representations on image and text datasets (Hinton and Salakhutdinov 2006; Baldi 2012). However, *autoencoder* mainly captures the feature-level couplings through minimizing reconstruction error whereas it does not explicitly enhance the discrimination between objects.

Distance metric learning is another way to learn the discrimination information between objects by learning a distance metric for the input space of data from a given collection of pairs of similar/dissimilar points that preserves the distance relation in the training data (Yang and Jin 2006). However, most metric learning methods (Frome et al. 2007; Chechik et al. 2010; Weinberger and Saul 2009) need class labels to guide the learning process and cannot be applied to mixed data directly. Unsupervised distance metric learning, also called manifold learning, is essential to learn an underlying low-dimensional manifold, e.g., *principle component analysis*, *ISOMAP* (Tenenbaum, De Silva, and Langford 2000) and *local linear embedding* (Saul and Roweis 2003). However, they all focus on continuous data and cannot be applied to mixed data or learn the heterogeneous couplings between categorical and continuous data.

Problem Statement

The usual way to represent data is based on an information table $\mathcal{D} = (\mathcal{X}, \mathcal{F})$. Let $\mathcal{X} = \{x_1, x_2, ..., x_N\}$ be a set of data objects with size N, described by a set of d_c categorical features and d_n continuous features, i.e., $\mathcal{F} = \mathcal{F}^c \cup \mathcal{F}^n$ where $\mathcal{F}^c = \{f_1^c, ..., f_{d_c}^c\}$ and $\mathcal{F}^n = \{f_1^n, ..., f_{d_n}^n\}$. Each categorical feature f_i^c has a value domain $\mathcal{V}_i = \{v_1^i, v_2^i, ...\}$. The value domains of different categorical features are distinct, i.e., $\mathcal{V}_i \cap \mathcal{V}_j = \emptyset, \forall i \neq j$. The whole value set of categorical features is the union of all the value domains: $\mathcal{V} = \cup \mathcal{V}_i$, and the size of $\mathcal{V}$ is denoted as L. For each continuous feature f_i^n, the continuous value in object x is denoted by a_i^x.

Given a reference object x and two comparative objects x_i, x_j, we denote them as a triplet $\langle x, x_i, x_j \rangle$. $\langle \mathbf{h}^p, \mathbf{h}_i^p, \mathbf{h}_j^p \rangle$ and $\langle \mathbf{h}^c, \mathbf{h}_i^c, \mathbf{h}_j^c \rangle$ are corresponding representation vectors to learn in two encoding spaces. $\delta_{\mathbf{h}}(\mathbf{h}_i, \mathbf{h}_j)$ signifies the order of the local distance pairs $d(\mathbf{h}, \mathbf{h}_i)$ and $d(\mathbf{h}, \mathbf{h}_j)$ where $d(\cdot, \cdot)$ denotes a distance function. The contrastive orders $\delta_{\mathbf{h}}^p$ and $\delta_{\mathbf{h}}^c$ induced from two spaces are used as the supervision for distance metric learning in peer instructors. After learning, $[\mathbf{h}^p, \mathbf{h}^c]$ is selected as the final representation for each object.

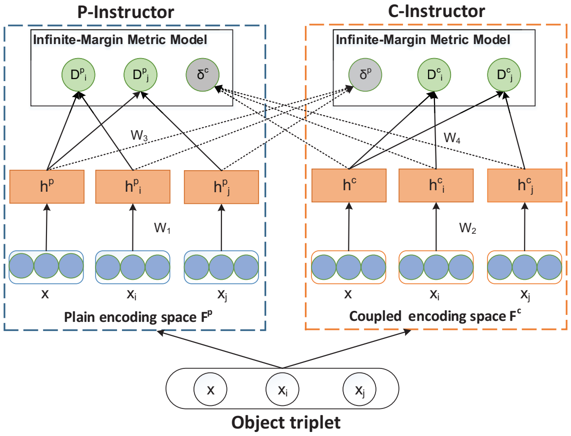


Figure 1: The MAI architecture: metric-based auto-instructor for mixed data representation.

Representation Learning with MAI

To implement the coupling and discrimination-enhanced representation learning, we propose the metric-based auto-instructor (MAI) model which consists of two collaborative instructors. The architecture of MAI, as demonstrated in Figure 1, is a coupled three-layer neural model consisting of P-Instructor and C-Instructor. D_i^p denotes the distance between representation vectors h^p and h_i^p. The first layer inputs the two compatible feature vectors encoded from the raw mixed data, namely the plain encoding space and the coupled encoding space. The second layer is the representation layer which learns the coupling relationships between the features in the input layer and update the representation according to the errors backpropogated from the above layer for infinite-margin optimization. The third layer is the auto-instructive metric learning layer for learning the enhanced discrimination between objects. In the following part, we detail each layer and derive the parameter learning algorithm.

Encoding Layer

In this layer, each object is encoded into two collaborative vectors from the original information table, namely a plain encoding vector and a coupled encoding vector. These two encoding spaces are based on different assumptions and describe the same object from different perspectives.

In the plain encoding space, each feature is treated equally and independently. The plain encoding vector does not differentiate categorical features from continuous features. However, in the coupled encoding space, continuous features are considered to be highly correlated with categorical features. The joint density of mixed-type features including one continuous feature and one categorical feature is estimated to help construct the coupled encoding space.

Plain Encoding Space The plain encoding space consists of transformed categorical features and continuous features which keep the most complete information. We use one-hot encoding to transform the categorical features into $|\mathcal{V}|$ binary features. Each binary feature has a single 1 corresponding to one categorical value, and all the rest of the entries are 0s. The continuous features are kept with the original values and then concatenated with transformed categorical features to obtain the plain encoding vector. Accordingly, the dimensionality of the plain encoding vector $\mathbf{f}^p \in \mathbb{R}^{d_n+|\mathcal{V}|}$ for each object.

Coupled Encoding Space The coupled encoding space is derived from the coupled encoding matrices of all objects. In a coupled encoding matrix, rows represent the continuous features and columns represent the categorical values. Each entry is filled according to the density estimation of mixed-type pair features, i.e., the interaction between one continuous feature and one categorical feature. For a mixed-type pair of features $\langle f_i^n, f_j^c \rangle$, we treat them as two variables $\langle A_i, V_j \rangle$ and use product kernel (Scott 2015) to estimate its density. Hence, the joint density of value a_i^x from variable A_i and v_j from feature V_j is defined as:

$$p(a_i^x, v_j) = \frac{1}{N} \sum_{k=1}^{N} \{L_\lambda(v_j^k, v_j) W(\frac{a_i^k - a_i^x}{h_i})\} \tag{1}$$

where $W((a_i^k - a_i^x)/h_i)$ is a continuous data kernel function and h_i is the bandwidth for the continuous variable. In this paper, we use the Gaussian kernel as the continuous data kernel function, denoted as $K_h(z) = \frac{1}{\sqrt{2\pi}h} \exp(-\frac{z^2}{2})$ where the bandwidth is chosen according to the standard nonparametric statistics (Li and Racine 2008). $L(v_j^k, v_j)$ is a categorical data kernel function. Here we follow the settings in (Li and Racine 2008) by choosing the Aitchison and Aitken's kernel function (Aitchison and Aitken 1976):

$$L_\lambda(v_j^k, v_j) = \begin{cases} 1, & \text{if } v_j^k = v_j \\ \lambda, & \text{otherwise} \end{cases} \tag{2}$$

where $\lambda \ll 1$ is the bandwidth for the categorical variable.

After obtaining the density of any two pairs of categorical variables and continuous variables, the coupled encoding matrix $\mathbf{M}_x$ for object x is defined as follows.

$$\mathbf{M}_x = \begin{bmatrix} r(f_1^n, v_1) & \dots & r(f_1^n, v_L) \\ \vdots & \ddots & \vdots \\ r(f_{d_n}^n, v_1) & \dots & r(f_{d_n}^n, v_L) \end{bmatrix} \tag{3}$$

where $r(f_i^n, v_j)$ is defined by the density of the mixed-type variables, which encodes the product of interaction between the continuous value and smoothed categorical value.

$$r(f_i^n, v_j) = \begin{cases} a_i^x, & \text{if } p(a_i^x, v_j) \geq \tau \\ \lambda a_i^x, & \text{otherwise} \end{cases} \tag{4}$$

where $p(a_i^x, v_j)$ is the joint density of mixed-type features estimated according to Eq. (1) and τ is the threshold. A small value of τ around 0.02 is preferred in our experiments.

As a result, the heterogeneous couplings between categorical and continuous features are captured by $\mathbf{M}_x$. Since each object is represented by a coupled encoding matrix $\mathbf{M}_x$, we vectorize $\mathbf{M}_x$ as the coupled encoding features, $\mathbf{f}^c \in \mathbb{R}^{d_n \times |\mathcal{V}|}$.

Representation Layer

The features from plain encoding space in the encoding layer are uncorrelated and independent. To capture the couplings between features, we link this encoding vector $\mathbf{f}^p$ to the K-length representation vector $\mathbf{h}^p$ in terms of a fully connected network

$$\mathbf{h}^p = \sigma(\mathbf{f}^p \cdot \mathbf{W}_1) \tag{5}$$

where the weight matrix $\mathbf{W}_1 \in \mathbb{R}^{K\times(d_n+|\mathcal{V}|)}$ encodes the interaction strength over features, and the logistic function $\sigma(z) = 1/(1+e^{-z})$.

Similarly, we map the coupled encoding vector $\mathbf{f}^c$ to the J-length representation vector $\mathbf{h}^c$ in terms of another fully connected network with the interaction weight matrix $\mathbf{W}_2 \in \mathbb{R}^{J\times(d_n\times|\mathcal{V}|)}$ and

$$\mathbf{h}^c = \sigma(\mathbf{f}^c \cdot \mathbf{W}_2). \tag{6}$$

To this point, we have employed two fully connected networks to represent the interactions between all features from the input layers by the representation vectors $\mathbf{h}^p$ and $\mathbf{h}^c$ respectively. In this stage, this representation mainly captures the feature-level couplings like autoencoder. To enhance the discrimination between objects in terms of $\mathbf{h}^p$ and $\mathbf{h}^c$ representation, we put a metric learning layer on the representation layer to backpropagate the errors caused in infinite-margin optimization.

Auto-instructive Metric Learning Layer

For P-Instructor, the distance metric D^p between two objects x and x_i is defined by their representation $\mathbf{h}^p$ and $\mathbf{h}_i^p$:

$$D^p(\mathbf{h}^p, \mathbf{h}_i^p) = (\mathbf{h}^p - \mathbf{h}_i^p)\mathbf{W}_3(\mathbf{h}^p - \mathbf{h}_i^p)^\top \tag{7}$$

where $\mathbf{W}_3 \in \mathbb{R}^{K\times K}$ is a symmetric positive semi-definite matrix and can be decomposed as $\mathbf{W}_3 = \mathbf{M}_1\mathbf{M}_1^\top$, $\mathbf{M}_1 \in \mathbb{R}^{K\times e}$ and $e \geq rank(\mathbf{W}_3)$. Then, Eq. 7 can be rewritten as:

$$D^p(\mathbf{h}^p, \mathbf{h}_i^p) = (\mathbf{h}^p\mathbf{M}_1 - \mathbf{h}_i^p\mathbf{M}_1)(\mathbf{h}^p\mathbf{M}_1 - \mathbf{h}_i^p\mathbf{M}_1)^\top \tag{8}$$

For C-Instructor, the distance metric D^c between two objects x and x_i is defined similarly:

$$D^c(\mathbf{h}^c, \mathbf{h}_i^c) = (\mathbf{h}^c - \mathbf{h}_i^c)\mathbf{W}_4(\mathbf{h}^c - \mathbf{h}_i^c)^\top \tag{9}$$

where $\mathbf{W}_4 \in \mathbb{R}^{J\times J}$ is a symmetric positive semi-definite matrix and $\mathbf{W}_4 = \mathbf{M}_2\mathbf{M}_2^\top$.

Given a reference object x and two comparative objects x_i, x_j, we can easily obtain the metric pairs $\langle D^p(\mathbf{h}^p, \mathbf{h}_i^p)$, $D^p(\mathbf{h}^p, \mathbf{h}_j^p)\rangle$ and $\langle D^c(\mathbf{h}^c, \mathbf{h}_i^c)$, $D^c(\mathbf{h}^c, \mathbf{h}_j^c)\rangle$ according to the above definition. In normal metric learning methods (Frome et al. 2007), it needs to provide the order of metric pairs $D(\mathbf{h}, \mathbf{h}_i)$ and $D(\mathbf{h}, \mathbf{h}_j)$. However, we do not have class labels to define this order in an unsupervised learning problem. To address the unsupervised challenge, we design an auto-instructive process to get a contrastive order of distance pairs from the peer instructor. Formally, we define a binary function $\delta_\mathbf{h}$ to denote the order of distance pairs over a triplet of representations.

$$\delta_\mathbf{h}(\mathbf{h}_i, \mathbf{h}_j) = \begin{cases} 1, \text{if } d(\mathbf{h}, \mathbf{h}_i) > d(\mathbf{h}, \mathbf{h}_j) \\ 0, \text{otherwise.} \end{cases} \tag{10}$$

where d is a local distance function, e.g., Euclidean distance, cosine dissimilarity. Here we choose $d(\mathbf{h}, \mathbf{h}_i) = \|\mathbf{h} - \mathbf{h}_i\|_2$.

Given a triplet $\langle x, x_i, x_j\rangle$, C-Instructor induces an order $\delta_\mathbf{h}^c(\mathbf{h}_i, \mathbf{h}_j)$ to work with the distance metric learning process in P-Instructor. Accordingly, the log probability of $D_i^p > D_j^p$ conditional on $\delta_{\mathbf{h}^c}^c$ is:

$$\begin{aligned} &\log P_{\Theta^p}(D_i^p > D_j^p|\delta_{\mathbf{h}^c}^c) = \delta_{\mathbf{h}^c}^c(\mathbf{h}_i^c, \mathbf{h}_j^c)\log\sigma(D_i^p - D_j^p) \\ &+ (1 - \delta_{\mathbf{h}^c}^c(\mathbf{h}_i^c, \mathbf{h}_j^c))\log(1 - \sigma(D_i^p - D_j^p)) \end{aligned} \tag{11}$$

where D_i^p is the simplified notation of $D^p(\mathbf{h}^p, \mathbf{h}_i^p)$, and $\Theta^p = \{\mathbf{W}_1, \mathbf{M}_1\}$ defines the model parameter set in the P-Instructor.

Similarly, the log conditional probability of $D_i^c > D_j^c$ in C-Instructor is:

$$\begin{aligned} &\log P_{\Theta^c}(D_i^c > D_j^c|\delta_{\mathbf{h}^p}^p) = \delta_{\mathbf{h}^p}^p(\mathbf{h}_i^p, \mathbf{h}_j^p)\log\sigma(D_i^c - D_j^c) \\ &+ (1 - \delta_{\mathbf{h}^p}^p(\mathbf{h}_i^p, \mathbf{h}_j^p))\log(1 - \sigma(D_i^c - D_j^c)) \end{aligned} \tag{12}$$

where $\Theta^c = \{\mathbf{W}_2, \mathbf{M}_2\}$.

As a result, we can write a pair of coupled loss functions w.r.t. P-Instructor and C-Instruct over all possible triplets:

$$\begin{cases} L_{\Theta^p} = -\displaystyle\sum_{\langle x,x_i,x_j\rangle} \log P_{\Theta^p}(D_i^p > D_j^p|\delta_{\mathbf{h}^c}^c) \\ L_{\Theta^c} = -\displaystyle\sum_{\langle x,x_i,x_j\rangle} \log P_{\Theta^c}(D_i^c > D_j^c|\delta_{\mathbf{h}^p}^p) \end{cases} \tag{13}$$

Specially, if we regard $D^p(\mathbf{h}^p, \mathbf{h}_i^p)$ as an energy function, and MAI can be viewed as an energy-based model (LeCun et al. 2006). Without loss of generality, we have the following log loss when $\delta_{\mathbf{h}^c}(\mathbf{h}_i^c, \mathbf{h}_j^c) = 1$:

$$\begin{aligned} L &= -\log P_{\Theta^p}(D_i^p > D_j^p) = -\log\sigma(D_i^p - D_j^p) \\ &= \log(1 + e^{(D_j^p - D_i^p)}) \end{aligned} \tag{14}$$

This form is a common variation of the hinge loss, which can be seen as a "soft" version of the hinge loss with an *infinite margin* (LeCun et al. 2006). That is, we set up an infinite-margin metric learning objective, which backpropagates the error to the representation layer to enlarge the margins between the representation of objects.

Parameter Learning

In the previous section, we have set up a pair of coupled loss functions as Eq. (13). As both loss functions are differentiable, a gradient descent-based algorithm can be derived for parameter estimation.

First of all, the gradient of the loss function from the P-Instructor w.r.t. the model parameters Θ^p is given as follows:

$$\frac{\partial L}{\partial \Theta^p} = -\sum_{\langle x,x_i,x_j\rangle} \frac{\partial}{\partial \Theta^p}\log P_{\Theta^p}(D_i^p > D_j^p|\delta_{\mathbf{h}^c}^c) \tag{15}$$

where $\Theta^p = \{\mathbf{W}_1, \mathbf{M}_1\}$. Then, the gradients w.r.t. $\mathbf{M}_1$ and $\mathbf{W}_1$ for each triplet $\langle x, x_i, x_j\rangle$ are given below:

$$\begin{aligned} &\frac{\partial}{\partial \mathbf{M}_1}\log P_{\Theta^p}(D_i^p > D_j^p|\delta_{\mathbf{h}^c}^c) \\ &= 2[\delta_{\mathbf{h}^c}^c(1 - \sigma(D_i^p - D_j^p)) - (1 - \delta_{\mathbf{h}^c}^c)\sigma(D_i^p - D_j^p)]H \end{aligned} \tag{16}$$

where
$H = [(\mathbf{h}^p - \mathbf{h}_i^p)^\top(\mathbf{h}^p - \mathbf{h}_i^p) - (\mathbf{h}^p - \mathbf{h}_j^p)^\top(\mathbf{h}^p - \mathbf{h}_j^p)]\mathbf{M}_1.$

$$\frac{\partial}{\partial \mathbf{W}_1} \log P_{\Theta^p}(D_i^p > D_j^p | \delta^c_{\mathbf{h}^c}) = 2[\delta^c_{\mathbf{h}^c}(1 - \sigma(D_i^p - D_j^p)) - (1 - \delta^c_{\mathbf{h}^c})\sigma(D_i^p - D_j^p))(R_i - R_j)] \quad (17)$$

where R_i and R_j are given below:

$$\begin{aligned} R_i =& 2\mathbf{f}^\top((\mathbf{h}^p - \mathbf{h}_i^p) \cdot \mathbf{W}_3 \odot ((\mathbf{h}^p \odot (1 - \mathbf{h}^p))) \\ &- 2\mathbf{f}_i^\top((\mathbf{h}^p - \mathbf{h}_i^p) \cdot \mathbf{W}_3 \odot ((\mathbf{h}_i^p \odot (1 - \mathbf{h}_i^p))) \end{aligned} \quad (18)$$

$$\begin{aligned} R_j =& 2\mathbf{f}^\top((\mathbf{h}^p - \mathbf{h}_j^p) \cdot \mathbf{W}_3 \odot ((\mathbf{h}^p \odot (1 - \mathbf{h}^p))) \\ &- 2\mathbf{f}_j^\top((\mathbf{h}^p - \mathbf{h}_j^p) \cdot \mathbf{W}_3 \odot ((\mathbf{h}_j^p \odot (1 - \mathbf{h}_j^p))) \end{aligned} \quad (19)$$

where $\odot$ denotes the element-wise product.

Similarly, the gradients of the loss function from the C-Instructor have the same form. Accordingly, we can obtain the gradient-based updated equation for each parameter. The complete process of MAI is briefly demonstrated in Algorithm 1, where Γ is a function to assign the adaptive learning rate, e.g., AdaGrad, RMSProp, Adam (Ruder 2016). For constructing a mini-batch of triplets, we first randomly select N_b different reference objects, and then sample a pair of comparative objects for each reference object.

When the model has been trained, given a plain encoding vector $\mathbf{f}^p$ and a coupled encoding vector $\mathbf{f}^c$, we can immediately get the new representation $[\mathbf{h}^p, \mathbf{h}^c]$ (cf. Eq. 5, 6) by concatenating $\mathbf{h}^p$ and $\mathbf{h}^c$. Additionally, the distance between objects x and x_i can be measured by the average distance metric $D(x, x_i) = [D^p(\mathbf{h}^p, \mathbf{h}_i^p) + D^c(\mathbf{h}^c, \mathbf{h}_i^c)]/2$ (cf Eq. 7, 9). This distance can be directly used for clustering tasks. In fact, $\mathbf{h}_D^p = \mathbf{h}^p\mathbf{M}_1$ (cf. Eq. 8) can be regarded as a metric-based representation, and then $[\mathbf{h}_D^p, \mathbf{h}_D^c]$ is the concatenated representation over two encoding spaces.

Experiments

Considering the computational power of GPUs on matrix operation, we use a GPU-based adaptive stochastic gradient descent (SGD) optimizer over mini-batches to speed up the training, thus MAI can be applied on large data. We use Adam in our implementation[1].

Comparison Methods

We compare two baseline methods and two state-of-the-art methods with our models.

- Plain encoding: plain encoding vectors ignore the coupling between features, and the distance over each feature between two objects are calculated independently.
- Coupled encoding: coupled encoding vectors highlight the coupling relationships between categorical and continuous features.
- CoupledMC: a state-of-the-art method for feature representation of mixed data.

Algorithm 1 A SGD-based learning algorithm for MAI

```
Input: $\mathcal{X}$: data objects, $MaxIter$: maximum iterations,
    $nBatch$: number of minibatches, $N_b$: batchSize
Output: $\Theta^p, \Theta^c$ - the parameters
Construct the plain encoding $\mathbf{F}^p$ and the coupled encoding $\mathbf{F}^c$
Initialize $\Theta^p, \Theta^c$
while $iter \leq MaxIter$ do
    for $q = 1 : nBatch$ do
        $\mathcal{M}_{triplet} : \{\langle \mathbf{x}, \mathbf{x_i}, \mathbf{x_j} \rangle\}^{N_b} \leftarrow getMinibatch()$
        Calculate $\{\langle \mathbf{h}^p, \mathbf{h}_i^p, \mathbf{h}_j^p \rangle\}^{N_b}$ of $\mathcal{M}_{triplet}$ (cf. Eq. 5)
        Calculate $\{\langle \mathbf{h}^c, \mathbf{h}_i^c, \mathbf{h}_j^c \rangle\}^{N_b}$ of $\mathcal{M}_{triplet}$ (cf. Eq. 6)
        $\delta^c \leftarrow$ order for each $\langle \mathbf{h}^c, \mathbf{h}_i^c, \mathbf{h}_j^c \rangle$ (cf. Eq. 10)
        $\delta^p \leftarrow$ order for each $\langle \mathbf{h}^p, \mathbf{h}_i^p, \mathbf{h}_j^p \rangle$ (cf. Eq. 10)
        $\nabla L_{\Theta^p}(\mathcal{M}_{triplet}) \leftarrow$ the gradient w.r.t. $\{\mathbf{W}_1, \mathbf{M}_1\}$ (cf. Eq. 16, 17)
        $\nabla L_{\Theta^c}(\mathcal{M}_{triplet}) \leftarrow$ the gradient w.r.t. $\{\mathbf{W}_2, \mathbf{M}_2\}$
        $\Theta^p \leftarrow \Theta^p - \Gamma(\nabla L_{\Theta^p}(\mathcal{M}_{triplet}))$
        $\Theta^c \leftarrow \Theta^c - \Gamma(\nabla L_{\Theta^c}(\mathcal{M}_{triplet}))$
    end for
end while
```

- Autoencoder (Hinton and Salakhutdinov 2006): uses neural networks to learn the representation in an unsupervised way. We use plain encoding vectors as the input of the autoencoder.
- MAI-F: our model uses $[\mathbf{h}^p, \mathbf{h}^c]$ as the representation.
- MAI-D: our model uses $[\mathbf{h}_D^p, \mathbf{h}_D^c]$ as the representation.

The length of the representation layer in MAI and the hidden layer of Autoencoder is set to 200. The maximum number of iterations is 30 and the batchsize is 200 in MAI. The dimensionalities of $\mathbf{M}_1$ and $\mathbf{M}_2$ are set to 60 in MAI.

Evaluation Methods and Datasets

We apply the representation methods to the typical partition-based clustering *k*-means and density-based clustering DBSCAN (Ester et al. 1996), to test the quality of representation. To evaluate the clustering performance, the Adjusted Mutual Information (AMI) (Vinh, Epps, and Bailey 2010) is used, whose higher values indicate better clustering. The Adjusted Rand Index (ARI) is another measure to evaluate the clustering performance (Rajan and Bhattacharya 2016; Ni et al. 2017), which is based on ground-truth labels similar to AMI, hence we only report the AMI results. The AMI results of *k*-means on each dataset are the average over 20 validations of clustering with distinct starting points due to the instability of *k*-means clustering. In *k*-means clustering, we fix the cluster number to the number of classes in Table 1. For DBSCAN clustering, the Euclidean distance is used to find the neighbors.

We use eight real-world UCI datasets (Lichman 2013) from different domains for the experiments: Echocardiogram (Echo), Hepatitis, Auto MPG Dataset (MPG), Statlog Heart (Heart), Australian Credit Approval (ACA), Credit Approval (CRX), Contraceptive Method Choice (CMC), and Census Income (Income). The statistics of the datasets

[1] The MATLAB implementation of Algorithm 1 is available at https://github.com/jiansonglei/MAI

Table 1: Statistics of UCI datasets

Datasets	$\lvert\mathcal{X}\rvert$	$\lvert\mathcal{F}^c\rvert$	$\lvert\mathcal{F}^n\rvert$	$\lvert Class\rvert$
Echo	132	2	8	3
Hepatitis	155	13	6	2
MPG	398	2	5	6
Heart	270	8	5	2
ACA	690	8	6	2
CRX	690	9	6	2
CMC	1473	7	2	3
Income	32561	8	6	2

are shown in Table 1, including the number of objects, the number of categorical features, the number of continuous features, and the number of ground-truth classes. We substitute the missing values with the mode value for categorical features and the mean value for continuous features.

K-means Clustering Results

The AMI results of MAI-F and MAI-D, compared with plain encoding, coupled encoding, coupledMC and autoencoder on *k*-means clustering, are shown in Table 2. MAI-F and MAI-D are in the first two positions on six datasets. On average, MAI-F demonstrates approximately 33%, 59%, 56%, and 27% improvement over plain encoding, coupled encoding, coupledMC and autoencoder respectively. MAI-D demonstrate an approximate 34%, 61%, 57%, and 28% improvement over plain encoding, coupled encoding, coupledMC and autoencoder respectively.

For most datasets, the MAI representation achieves the best performance since it better captures the complex coupling in mixed data and enhances the discrimination between objects as well. Plain encoding cannot differentiate categorical and continuous features or capture the heterogeneity between them. Coupled encoding only highlights the couplings between categorical features and continuous features. CoupledMC is rule-based so that it cannot learn from data and the discretization also brings bias to the representation. Autoencoder cannot learn from other discriminative information except the reconstruction error.

DBSCAN Clustering Results

The AMI results of MAI-F compared with plain encoding, coupled encoding, coupledMC and autoencoder on DBSCAN clustering are shown in Table 3[2]. MAI-F achieves the best performance on seven datasets, and on average MAI-F performs more than twice better than the other comparison methods in terms of AMI.

Since DBSCAN clustering detects clusters in an automatic way, the cluster numbers of different representations are varied. MAI uses an infinite-margin metric model to marginalize the distance between objects, therefore the clusters are more visible in MAI-F which helps the clustering algorithm to find the proper clusters. Taking the largest dataset *Income* as an example, DBSCAN produces 493 clusters and 291 clusters with plain encoding and representation learned by the autoencoder respectively. The huge number of clusters has less semantic meaning in real applications. Due to the discrimination-enhanced representation learned by MAI, the cluster number detected by MAI-enabled DBSCAN is closer to the ground-truth number of classes.

Data Representation Analysis

To evaluate the separability of the representations, we calculate the Calinski-Harabasz (CH) index based on the ground-truth class label (Caliński and Harabasz 1974) .

$$CH = \frac{\sum_i n_i d^2(c_i, c)/(NC-1)}{\sum_i \sum_{x \in C_i} d^2(x, c_i)/(N-NC)} \tag{20}$$

where C_i is the i-th cluster; n_i is the number of objects in C_i; c_i is the center of C_i; $d(x, y)$ is the distance between x and y. A larger *CH* indicates better clustering results.

The *CH* index is a classical internal clustering validation measure which evaluates the cluster validity based on the average between- and within-cluster sum of squares. Instead of calculating *CH* on the clustering results, we conduct *CH* on the ground-truth clustering label to reflect the separability of different representation methods without introducing bias from the clustering process. The *CH* results of representations from plain encoding, coupled encoding, coupledMC, autoencoder and MAI are shown in Table 4. It shows that the *CH* index of MAI-F is the largest compared with the other methods, which indicates MAI produces more separable representation and reflects the cluster structure in data.

To illustrate the separability of MAI-enabled representation, Figure 2 visualizes the representations by different methods on dataset *CRX*. For the visualization, t-SNE (Maaten and Hinton 2008) is applied to the representation of each object. It shows that the representation learned from MAI is the most separable and the cluster structure is clearer than the others. The separation of representation makes clustering easier which is evidenced in Table 2. Since *k*-means clustering is sensitive to initialization, the clustering results always vary each time and the deviation of clustering results is large. However, the deviation of clustering results of MAI is almost 0 for most datasets, which results from the large-margin separation of representation from MAI.

Convergence Test

We apply the representation learned in each iteration into *k*-means clustering and show that the AMI results vary with representations from each iteration. Here, we report the convergence speed of the MAI model on datasets *CRX* and *ACA*. Similar results can be obtained on other datasets. As Figure 3 shows, the clustering performance converges within 15 iterations.

Conclusions

We have proposed a metric-based auto-instructor (MAI) representation learning model which captures both the feature couplings and the discrimination between objects for

[2] We do not display MAI-D in the tables due to space limitation, and MAI-D has close results to MAI-F in our experiments.

Table 2: *K*-means clustering performance w.r.t. AMI ± standard deviation. The top two performers for each are boldfaced.

Datasets	Plain encoding	Coupled encoding	CoupledMC	Autoencoder	MAI-F	MAI-D
Echo	0.1789±0.1033	0.1749±0.0444	0.1237±0.1147	0.2493±0.0207	**0.3246±0.0000**	**0.3304±0.0000**
Hepatitis	0.1453±0.0703	0.1761±0.0292	0.1532±0.0342	0.1689±0.0163	**0.1848±0.0000**	**0.1905±0.0000**
MPG	0.1490±0.0106	0.1477±0.0184	0.1373±0.0347	0.1536±0.0086	**0.1831±0.0232**	**0.1770±0.0000**
Heart	**0.3130±0.0688**	0.1439±0.0642	0.1037±0.1215	**0.3302±0.0042**	0.2632±0.0000	0.2774±0.0000
ACA	0.3204±0.1518	0.3433±0.1726	0.3182±0.0627	0.3477±0.0844	**0.4258±0.0000**	**0.4258±0.0000**
CRX	0.2322±0.1191	0.0836±0.1109	0.2714±0.1361	0.1445±0.1477	**0.4267±0.0000**	**0.4267±0.0000**
CMC	0.0293±0.0052	0.0269±0.0013	**0.0333±0.0070**	0.0292±0.0037	**0.0327±0.0077**	0.0303±0.0081
Income	0.1139±0.0361	**0.1414**±0.0291	0.1258±0.0658	0.1314±0.0000	**0.1325±0.0000**	**0.1325±0.0000**
Average	0.1853±0.0707	0.1547±0.0588	0.1583±0.0722	0.1944±0.0353	**0.2467±0.0064**	**0.2488±0.0010**

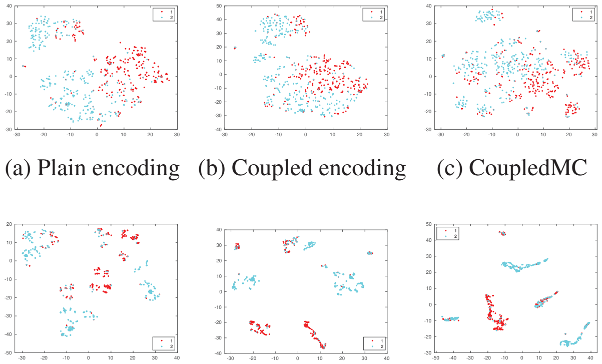
Table 3: DBSCAN clustering performance w.r.t. AMI. Note: $|C|$ refers to the number of detected clusters. AE refers to autoencoder and CMC refers to CoupledMC for short.

Datasets	PF($\|C\|$)	CF($\|C\|$)	CMC($\|C\|$)	AE($\|C\|$)	MAI-F($\|C\|$)
Echo	0.123(5)	0.011(3)	0.067(2)	0.188(7)	**0.392**(3)
Hepatitis	0.019(4)	0.044(2)	0.037(5)	0.016(2)	**0.075**(3)
MPG	0.031(20)	0.037(16)	0.049(13)	0.149(2)	**0.237**(3)
Heart	0.024(4)	0.001(2)	0.003(2)	0.003(2)	**0.130**(3)
ACA	0.003(4)	0.021(7)	0.031(2)	0.087(20)	**0.227**(6)
CRX	0.003(4)	0.018(6)	0.061(2)	0.102(16)	**0.242**(5)
CMC	0.002(21)	0.009(2)	0.115(5)	0.003(13)	**0.043**(2)
Income	**0.157**(493)	0.052(6)	0.052(6)	0.108(291)	0.1304(15)
Average	0.0451	0.0242	0.0519	0.0818	0.1845

Table 4: Calinski-Harabasz index on representation w.r.t. the Euclidean distance for ground-truth labels.

Datasets	PF	CF	CMC	AE	MAI-F
Echo	14.60	7.14	5.12	21.99	**56.81**
Hepatitis	11.76	8.65	15.91	16.05	**44.15**
MPG	19.18	7.34	7.53	41.88	**45.91**
Heart	32.35	16.83	5.64	56.49	**91.85**
ACA	72.90	31.69	16.92	124.37	**288.31**
CRX	67.78	65.94	20.77	106.97	**226.55**
CMC	16.82	12.46	17.21	22.44	**35.35**
Income	1419.90	2029.04	1729.04	3009.80	**5045.45**

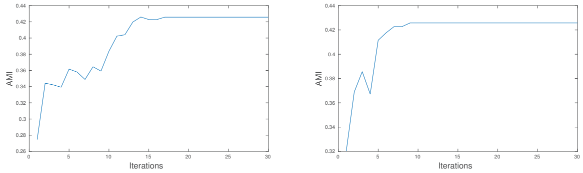
mixed data. By auto-instructive metric learning with infinite-margin objectives induced from two collaborative instructors, MAI builds the consensus of mixed data representation between the plain encoding space and the coupled encoding space. Extensive experiments have demonstrated that MAI outperforms other state-of-the-art representation methods. With data representation analysis, we have shown that the MAI representation makes data objects more distinguishable and discloses the natural cluster information in data. MAI is easily extended to a deep structure like the stacked autoencoder. In fact, MAI is a very general representation learning framework not limited to mixed data, which has the potential to be applied to multimodal learning and domain adaption.



(a) Plain encoding (b) Coupled encoding (c) CoupledMC

(d) Autoencoder (e) MAI-F (f) MAI-D

Figure 2: The t-SNE visualization of data representations by different methods on dataset *CRX*.



(a) MAI on *CRX* dataset (b) MAI on *ACA* dataset

Figure 3: Convergence test on datasets *CRX* and *ACA*.

Acknowledgement

The work is partially supported by program for New Century Excellent Talents in University, by National Science Foundation (NSF) China 61402492, 61402486, 61379146, by the National Key Research and Development Program of China (No. 2016YFB0200401), by the China Scholarship Council (CSC Student ID 201603170310).

References

Ahmad, A., and Dey, L. 2007. A k-mean clustering algorithm for mixed numeric and categorical data. *Data & Knowledge Engineering* 63(2):503–527.

Aitchison, J., and Aitken, C. G. 1976. Multivariate binary discrimination by the kernel method. *Biometrika* 63(3):413–420.

Baldi, P. 2012. Autoencoders, unsupervised learning, and deep architectures. In *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, 37–49.

Bengio, Y.; Courville, A.; and Vincent, P. 2013. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35(8):1798–1828.

Caliński, T., and Harabasz, J. 1974. A dendrite method for cluster analysis. *Communications in Statistics-theory and Methods* 3(1):1–27.

Cao, L.; Ou, Y.; and Yu, P. S. 2012. Coupled behavior analysis with applications. *IEEE Transactions on Knowledge and Data Engineering* 24(8):1378–1392.

Cao, L. 2015. Coupling learning of complex interactions. *J. Information Processing and Management* 51(2):167–186.

Chechik, G.; Sharma, V.; Shalit, U.; and Bengio, S. 2010. Large scale online learning of image similarity through ranking. *Journal of Machine Learning Research* 11(Mar):1109–1135.

Chen, J.-Y., and He, H.-H. 2016. A fast density-based data stream clustering algorithm with cluster centers self-determined for mixed data. *Information Sciences* 271–293.

David, G., and Averbuch, A. 2012. Spectralcat: Categorical spectral clustering of numerical and nominal data. *Pattern Recognition* 45(1):416–433.

Dougherty, J.; Kohavi, R.; Sahami, M.; et al. 1995. Supervised and unsupervised discretization of continuous features. In *ICML*, volume 12, 194–202.

Ester, M.; Kriegel, H.-P.; Sander, J.; Xu, X.; et al. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *SIGKDD*, 226–231.

Frome, A.; Singer, Y.; Sha, F.; and Malik, J. 2007. Learning globally-consistent local distance functions for shape-based image retrieval and classification. In *ICCV*, 1–8. IEEE.

Hinton, G. E., and Salakhutdinov, R. R. 2006. Reducing the dimensionality of data with neural networks. *Science* 313(5786):504–507.

Huang, Z. 1997. Clustering large data sets with mixed numeric and categorical values. In *PAKDD*, 21–34. Singapore.

Ji, J.; Pang, W.; Zhou, C.; Han, X.; and Wang, Z. 2012. A fuzzy k-prototype clustering algorithm for mixed numeric and categorical data. *Knowledge-Based Systems* 30:129–135.

Jia, H., and Cheung, Y.-M. 2017. Subspace clustering of categorical and numerical data with an unknown number of clusters. *IEEE Transactions on Neural Networks and Learning Systems*.

Jian, S.; Cao, L.; Pang, G.; Lu, K.; and Gao, H. 2017. Embedding-based representation of categorical data by hierarchical value coupling learning. In *IJCAI*, 1937–1943.

LeCun, Y.; Chopra, S.; Hadsell, R.; Ranzato, M.; and Huang, F. 2006. A tutorial on energy-based learning. *Predicting Structured Data* 1:0.

Li, Q., and Racine, J. S. 2008. Nonparametric estimation of conditional cdf and quantile functions with mixed categorical and continuous data. *Journal of Business & Economic Statistics* 26(4):423–434.

Lichman, M. 2013. UCI machine learning repository.

Maaten, L. v. d., and Hinton, G. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research* 9(Nov):2579–2605.

Ni, X.; Quadrianto, N.; Wang, Y.; and Chen, C. 2017. Composing tree graphical models with persistent homology features for clustering mixed-type data. In *ICML*, 2622–2631.

Rajan, V., and Bhattacharya, S. 2016. Dependency clustering of mixed data with gaussian mixture copulas. In *IJCAI*, 1967–1973.

Ruder, S. 2016. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*.

Saul, L. K., and Roweis, S. T. 2003. Think globally, fit locally: unsupervised learning of low dimensional manifolds. *Journal of Machine Learning Research* 4(Jun):119–155.

Scott, D. W. 2015. *Multivariate density estimation: theory, practice, and visualization*. John Wiley & Sons.

Tenenbaum, J. B.; De Silva, V.; and Langford, J. C. 2000. A global geometric framework for nonlinear dimensionality reduction. *Science* 290(5500):2319–2323.

Vincent, P.; Larochelle, H.; Lajoie, I.; Bengio, Y.; and Manzagol, P.-A. 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research* 11(Dec):3371–3408.

Vinh, N. X.; Epps, J.; and Bailey, J. 2010. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *Journal of Machine Learning Research* 11(Oct):2837–2854.

Wang, C.; Chi, C.-H.; Zhou, W.; and Wong, R. K. 2015. Coupled interdependent attribute analysis on mixed data. In *AAAI*, 1861–1867.

Wei, M.; Chow, T. W.; and Chan, R. H. 2015. Clustering heterogeneous data with k-means by mutual information-based unsupervised feature transformation. *Entropy* 17(3):1535–1548.

Weinberger, K. Q., and Saul, L. K. 2009. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research* 10(Feb):207–244.

Yang, L., and Jin, R. 2006. Distance metric learning: A comprehensive survey. *Michigan State Universiy* 2(2).