19th International Conference on Information Fusion
Heidelberg, Germany - July 5-8, 2016

GPLVM and Lava Floor Distance for Label-Deficient Semi-Supervised Learning: Case Study

Rastin Rastgoufard **AbdulRahman Alsamman**
Department of Electrical Engineering
University of New Orleans
New Orleans, LA 70148, U.S.A.
{rrastgou, aalsamma}@uno.edu

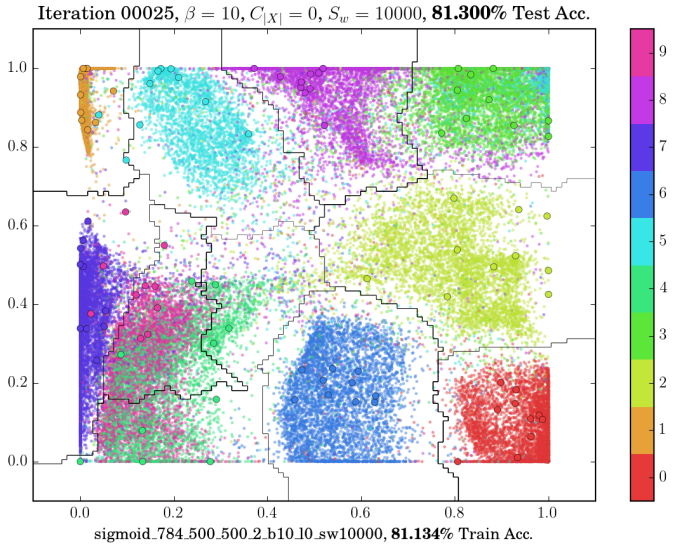
***Abstract*—Label-deficient semi-supervised learning is a challenging setting in which there is an abundance of unlabeled data but a dearth of labeled data. We propose a method for applying Gaussian process latent variable models (GPLVM) in a label-deficient setting, a method in which the discriminative GPLVM objective function trains a back-constraining neural network followed by a transformation into a semi-supervised classifier using the lava floor distance. We provide a case study that visually details the necessities and drawbacks of all of the components of the method, including the behavior of the standard GPLVM as a method of dimension reduction, the effect of adding small amounts of labeled data via discriminative GPLVM, and the characteristics of back-constrained GPLVM using neural networks built with various nonlinearities as a way of handling large data sizes. The method achieves 81% accuracy on unseen testing data compared to a standard support vector classifier trained on the same labeled points which achieves 70% accuracy.**

***Index Terms*—semi-supervised, label-deficient, GPLVM, distance, visualization**

I. Introduction

Label-deficient semi-supervised learning is a difficult variation of semi-supervised learning in which the amount of labeled data is minuscule, thus modeling many real world problems that involve large quantities of inexpensive (and unlabeled) sensor data and very small amounts of expensive human-assigned labels. This case study uses the MNIST data set but has only 100 labeled points, ten examples of each digit, and 49,900 unlabeled points available for training. Several methods have shown fantastic results on this challenging problem [1]–[4], but we examine Gaussian process latent variable models (GPLVM) in this case study that forces all results through a bottleneck of two dimensions for the sake of visualization. Figure 1 shows an example visualization of the bottleneck space and defines the meanings of the colors, circles, dots, and lines in all visualizations.

We propose a two-stage method for applying Gaussian process latent variable models [5] (GPLVM) in this label-deficient setting. We first train a back-constrained GPLVM [6] using an objective function that includes discriminative GPLVM [7], and then we use the novel lava floor distance to extend the system to classification. The lava floor distance operates on the cluster assumption [8] that points of the same class ought to form clusters separated by regions of low density, and it leverages the capabilities of GPLVM to perform reductions down to two-dimensional spaces where the distance

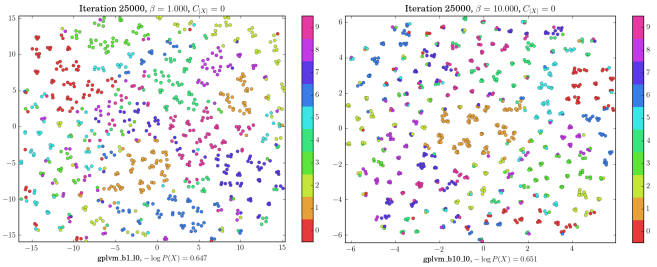


(a)

(b)

Fig. 1: (a) An example of the visualization of the two-dimensional bottleneck layer imposed onto every network. Each color corresponds to a different digit/class according to the colorbar to the right-hand side of the plot, the data points are plotted as dots colored according to their true classes, and the 100 labeled points, ten from each class and shown in (b), are plotted as larger circles with black outlines. The lines that separate clusters are the decision boundaries of a classifier trained in this bottleneck space, and the emboldened percentage in the title shows the accuracy of that classifier on unseen testing data. The 100 labeled points in (b) are constant throughout all experiments. No effort was made to select typical points or to remove outliers, and a support vector classifier trained just on those points in the full-dimensional image space achieves 70% accuracy on unseen testing data.

978-0-9964527-4-8



(a) $\beta = 1$ (b) $\beta = 10$

(c) $\beta = 100$ (d) $\beta = 1000$

Fig. 2: Varying the length scale parameter β in the standard GPLVM objective L_G, Equation 4.

between two locations is the shortest path weighted by density and can be computed efficiently using Djikstra's algorithm. The lava floor distance is similar to the image forest transform [9] except that it is applied to a density histogram instead of an image.

Two related approaches in this label-deficient setting are EmbedNN [1] and the manifold tangent classifier [2]. EmbedNN, which augments a neural network classifier with an unsupervised learning algorithm, is similar to our combination of GPLVM with neural networks. The manifold tangent classifier, which uses tangent information first learned from a contractive autoencoder in a subsequent classifier, is similar to our two-stage process, first extracting density information in the form of the lava floor distance and then using it in a classifier. Our method differs from [2] in the usage of GPLVM instead of contractive autoencoders and lava floor distance instead of manifold tangents, and it differs from [1] in that it trains the network using a single objective function for the purpose of dimension reduction with classification at a later stage.

The rest of this paper describes the formulation of our two-stage method, including the necessities and drawbacks of the individual components, in section II, presents the experimental results of the case study in section III, and provides discussion in section IV. The back-constraining neural network is trained using the objective function of Equation 15, and its extension to classification is through the lava floor distance described in subsection II-B. In section III, we compare the classification performance of our proposed lava floor distance against a standard support vector classifier (SVC) trained on the same GPLVM-trained networks, showing that the lava floor distance improves classification accuracy and successfully utilizes information from the unlabeled portion of the data. Finally, section V summarizes by providing concluding remarks.

II. Formulation

We propose a two-stage method for applying GPLVM in a label-deficient setting. The first stage trains a neural network using combined GPLVM, discriminative GPLVM, and back-constrained GPLVM objectives and is described in subsection II-A. The second stage extends GPLVM to classification using the lava floor distance and is described in subsection II-B.

A. GPLVM and Objective Functions

Data points are stored in a matrix Y with N rows, one row per data point, and D columns, one column per dimension. Corresponding to every high dimensional point is a point in a reduced-dimension latent space. Let the collection of corresponding latent points be X with N rows, one row per data point, and $P = 2$ columns, one column per dimension of the latent space. Determination of the matrix X is the goal of GPLVM.

The GPLVM likelihood function evaluates the likelihood of the observed data Y given a corresponding X and a chosen covariance matrix K.

$$P(Y|X,K) = \frac{1}{(2\pi)^{DN/2}|K|^{D/2}} \exp\left(-\frac{1}{2}\text{tr}(K^{-1}YY^T)\right) \tag{1}$$

The exponentiated quadratic covariance K is a function of X and has a single parameter β. The elements of K are K_{ij}, corresponding to x_i and x_j, the ith and jth rows of X.

$$K_{ij} = k(x_i, x_j) = \exp\left(-\frac{|x_i - x_j|^2}{\beta}\right) \tag{2}$$

The matrix X is found by minimizing $-\log P(Y)$, the GPLVM negative log-likelihood with implied conditioning on X and β inside of the matrix K.

$$-\log P(Y) = \frac{DN}{2}\ln(2\pi) + \frac{D}{2}\ln|K| + \frac{1}{2}\text{tr}(K^{-1}YY^T) \tag{3}$$

The first two terms are safe to ignore, leaving the GPLVM objective function L_G.

$$L_G = \text{tr}(K^{-1}YY^T) \tag{4}$$

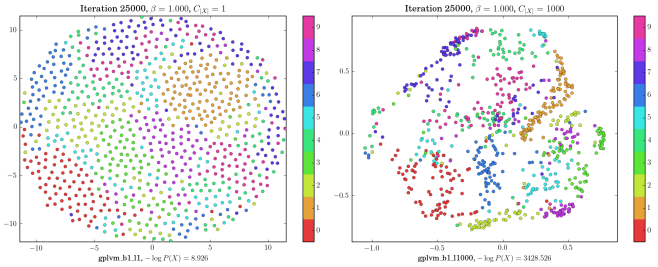
Figure 2 shows the result of minimizing L_G with varying values of β using the first $N = 1000$ images from the MNIST data set as Y. The most noticeable outcome is that high dimensional data points are shy or anti-social. In order to force more interaction between points, we add a norm penalty term, $L_{|X|}$, onto the objective function.

$$L_{|X|} = \sum_n x_n x_n^T \tag{5}$$

As before, x_n is the nth row of X. The combined objective is $L_{G,X}$.

$$L_{G,X} = L_G + C_{|X|}L_{|X|} \tag{6}$$

The value of $C_{|X|}$ controls the strength of the penalty, and Figure 3 shows the effect of its increase when $\beta = 1$.



(a) $C_{|X|} = 1$ (b) $C_{|X|} = 1000$

Fig. 3: Varying the norm penalty $C_{|X|}$ in $L_{G,X}$, Equation 6.

The discriminative GPLVM variant adds label information to the standard GPLVM. DGPLVM originally [7] adds an objective to keep points within the same class together, S_w, and another objective to create space between classes, S_b. We use the knowledge that our data set is anti-social, so S_b can be ignored. The data set needs to be separated into two pieces, Y_u and Y_l, such that the labeled points are in Y_l and the remaining points are in Y_u. The corresponding latent locations are similarly separated into X_u and X_l. Each $x \in X_l$ belongs to the class $c(x)$, which is known beforehand, and the mean of class $c(x)$ is $\bar{x}_{c(x)}$. The discriminative objective, L_D, is applicable only to the labeled latent locations X_l.

$$L_D = \sum_{x \in X_l} (x - \bar{x}_{c(x)})(x - \bar{x}_{c(x)})^T \tag{7}$$

Combining the discriminative objective with $L_{G,X}$ gives $L_{G,X,D}$.

$$L_{G,X,D} = L_G + C_{|X|} L_{|X|} + S_w L_D \tag{8}$$

Figure 4 shows the result of applying the discriminative objective to 100 labeled points, and a quick comparison to Figure 2 and Figure 3 reveals that not only are our points anti-social, but the unlabeled are afraid of the labeled! Fortunately, Figure 4 (c) shows a reasonable arrangement of points once a significant norm penalty is imposed.

Back-constraining neural networks allow GPLVM to scale to $N > 1000$ data points. The back-constraining function f has parameters θ such that the latent locations of the data points are $f(Y|\theta)$ instead of X, thereby changing the goal of the optimization to finding the parameters θ. The new objective function, L_{BC}, is the same as that of GPLVM

$$L_{BC} = \text{tr}(K^{-1} Y Y^T) \tag{9}$$

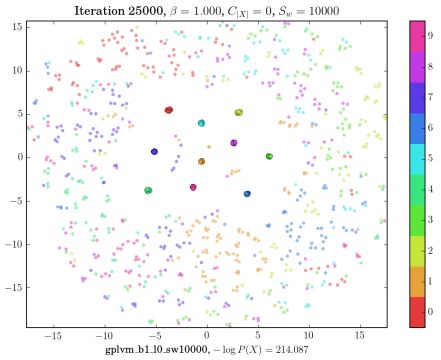
except that the entries of the covariance matrix K are computed using $f(Y)$ instead of X.

$$K_{ij} = k(f(y_i), f(y_j)) = \exp\left(-\frac{|f(y_i) - f(y_j)|^2}{\beta}\right) \tag{10}$$

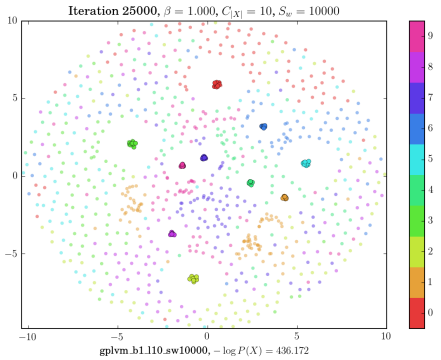
The matrix K has elements K_{ij}, and y_i, y_j are the ith and jth rows of Y.

The function f is a neural network constructed by a stack of layers with a chosen nonlinearity. Each layer i has an input X_i, an output Y_i, weights W_i, biases b_i, and nonlinearity s_i.

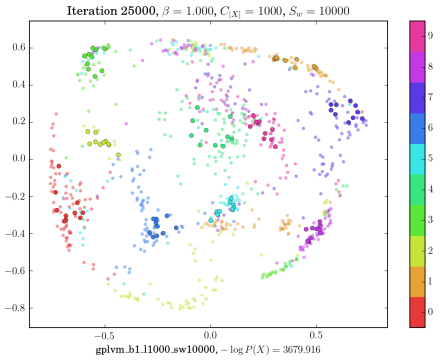
$$Y_i = s_i(X_i W_i + b_i) \tag{11}$$



(a) $C_{|X|} = 0$



(b) $C_{|X|} = 10$



(c) $C_{|X|} = 1000$

Fig. 4: Unlabeled data points are afraid of labeled points until they are forced to play nice together. All three images come from Equation 8 with $S_w = 10000$ but with varying $C_{|X|}$.

The dimensions of W_i and b_i are chosen to match the desired input and output shapes of that layer, and the layers are stacked such that the output of one layer, Y_i, is the input to the next layer, X_{i+1}. In this case study, the nonlinearity s_i is one of three element-by-element functions.

$$\text{softsign}: \quad s(x) = \frac{x}{1 + |x|} \tag{12}$$

$$\text{nonplus}: \quad s(x) = \frac{1}{1 + e^{-x}} + \frac{x}{20} - \frac{1}{2} \tag{13}$$

$$\text{sigmoid}: \quad s(x) = \frac{1}{1 + e^{-x}} \tag{14}$$

The input to the first layer of the stack always has $784 = 28 \times 28$ dimensions, the output always has two dimensions,

and there are zero to five hidden layers of 500 units in between. All layers contribute W_i and b_i to the set of parameters θ.

Combining L_{BC} with the previous norm-penalty and discriminative objectives gives $L_{BC,X,D}$.

$$L_{BC,X,D} = L_{BC} + C_{|X|}L_{|X|} + S_w L_D \tag{15}$$

Compared to Equation 8, L_{BC} is a drop-in replacement of L_G with two exceptions: the optimization is with respect to θ as opposed to X, and the network f can be trained using stochastic gradient descent on a large data set by subdividing the data into minibatches.

B. Lava Floor Distance

Networks trained with Equation 15 perform dimension reduction and are influenced by label information, but they are not classifiers. A way to convert them is to train a standard support vector classifier in the reduced-dimension space. SVCs learned in this fashion perform reasonably well, as can be seen in Table I and Figure 6 (a) - (c), but they ignore knowledge gained from unlabeled data. Figure 5 (a) shows that the decision boundaries of a standard SVC often cross high density regions, thus violating the cluster assumption.

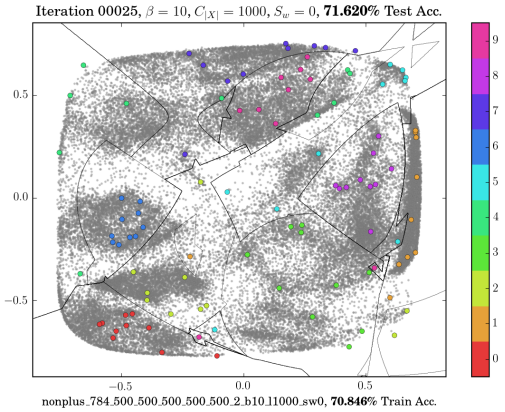
The lava floor distance captures information about the distribution $P(f(Y))$ and makes it available for use in a classifier. It is a distance d between any location (r, c) and any other location (r', c') in the two-dimensional space. The distance is computed according to the following six steps.

1) Divide the two-dimensional space into a grid of R rows and C columns.
2) Count the number of points $f(Y)$ that fall into each cell to create a histogram.
3) Convolve a Gaussian kernel of desired σ with the $R \times C$ histogram. The result is a kernel density estimate $P(r, c)$ that can be evaluated at any location (r, c).
4) Create G, a square $R \times C$ by $R \times C$ sparse graph.
5) For every cell (r, c) in the space, add an edge between it and every neighboring cell (r_n, c_n). The weight (or cost) of the edge is L_{LF}.

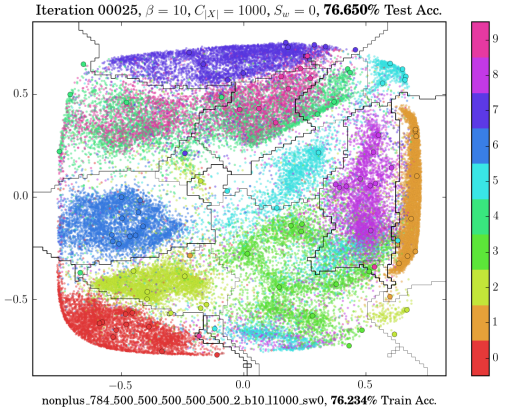
$$L_{LF}(r, c, r_n, c_n|P) = \left(1 - \frac{P(r_n, c_n)}{\max(P)}\right)^8 \tag{16}$$

6) The distance $d(r, c, r', c'|P)$ is the length of the shortest path through G between (r, c) and (r', c'). Shortest paths are computed using Dijkstra's algorithm.

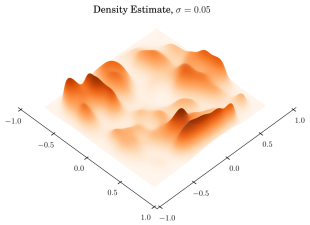
Two points y and y' in the high-dimensional space, reduced to $f(y)$ and $f(y')$ in the two-dimensional space, are considered as belonging to cells (r, c) and (r', c'), and the distance between them is the lava floor distance $d(r, c, r', c'|P)$. The distance d contains the lengths of the shortest paths, and Figure 5 (d) shows visualizations of the shortest paths between all 100 labeled points and a randomly chosen location inside of the blue cluster of sixes. Notice that the paths tend to favor traveling through regions of high density and only cross low-density regions when necessary. Figure 5 (b) shows the decision boundaries found by using the lava floor distance in a k-nearest neighbor classifier (LFKNN). The classification boundaries run along low-density regions whenever possible.



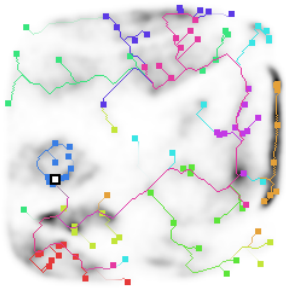
(a) SVC Boundaries



(b) LFKNN Boundaries



(c) Density Estimate



(d) Shortest Paths

Fig. 5: Various views of the lava floor distance given a fixed $f(Y)$. (a) Decision boundaries obtained using a standard support vector classifier utilizing the (colored) labeled data while ignoring information about (gray) unlabeled data. (b) Decision boundaries using lava floor k-nearest neighbors (LFKNN) utilizing both labeled and unlabeled data are a full five percent better than (a). (c) Kernel density estimate of unlabeled points from (a) and (b). (d) Shortest paths between all (colored squares) labeled points to a specific (black-outlined square) location while taking into account low-density penalties. The lava floor name comes from the game many of us played as children where we hopped from chair to chair while pretending the ground is covered in lava. Moving atop a piece of furniture is free, but touching the ground is costly.

TABLE I: Classification accuracies on unseen testing data of BC-GPLVM with and without label information across varying length scales β, network sizes, and nonlinearities. Each cell has two accuracies side-by-side: the number to the left shows the result of ignoring label information by setting $S_w = 0$, and the number to the right forces points of the same class together by setting $S_w = 10000$. Each $\circ$ is a hidden layer of 500 units. All accuracies less than 70% are grayed out, and accuracies greater than 75% are bolded.

NN Size	$\beta = 1$	$\beta = 10$	$\beta = 100$	$\beta = 1000$
784,2	48.34 46.22	33.08 44.39	37.56 45.10	28.96 36.51
784,∘,2	60.98 62.17	61.81 59.18	40.57 41.92	31.76 38.84
784,∘,∘,2	59.27 65.58	62.36 70.86	44.07 42.88	36.07 31.17
784,∘,∘,∘,2	69.09 64.31	73.21 70.22	37.05 42.24	24.93 34.56
784,∘,∘,∘,∘,2	71.54 47.56	65.57 55.95	32.72 37.59	31.04 45.14
784,∘,∘,∘,∘,∘,2	66.86 48.87	70.02 43.44	39.60 49.48	22.01 28.57

(a) softsign with SVC, $C_{|X|} = 0$

NN Size	$\beta = 1$	$\beta = 10$	$\beta = 100$	$\beta = 1000$
784,2	43.85 45.82	39.35 45.71	38.17 41.67	18.25 29.39
784,∘,2	57.17 60.16	56.22 67.49	34.67 43.71	20.90 28.74
784,∘,∘,2	61.60 63.96	58.68 64.65	43.46 49.64	22.93 27.12
784,∘,∘,∘,2	60.37 67.48	66.32 **76.67**	35.83 46.37	22.10 31.47
784,∘,∘,∘,∘,2	62.63 66.70	72.92 **77.33**	39.38 52.98	25.13 47.89
784,∘,∘,∘,∘,∘,2	62.61 73.18	71.62 75.14	32.28 40.96	26.11 34.98

(b) nonplus with SVC, $C_{|X|} = 1000$

NN Size	$\beta = 1$	$\beta = 10$	$\beta = 100$	$\beta = 1000$
784,2	44.75 46.28	46.66 47.61	44.78 39.20	21.67 31.14
784,∘,2	59.15 63.80	62.59 75.38	65.23 65.29	21.95 40.17
784,∘,∘,2	67.84 70.28	75.09 **79.34**	66.33 72.49	35.34 44.05
784,∘,∘,∘,2	68.26 72.89	75.36 **76.64**	65.73 68.25	22.51 48.00
784,∘,∘,∘,∘,2	68.86 75.04	20.29 74.20	10.32 63.14	15.87 13.56
784,∘,∘,∘,∘,∘,2	10.09 24.33	10.09 11.17	18.23 10.09	10.09 16.94

(c) sigmoid with SVC, $C_{|X|} = 0$

NN Size	$\beta = 1$	$\beta = 10$	$\beta = 100$	$\beta = 1000$
784,2	46.01 44.35	48.39 47.49	45.23 37.29	20.87 34.19
784,∘,2	63.67 64.71	62.78 74.85	63.87 65.73	19.78 40.87
784,∘,∘,2	66.80 66.25	71.81 **81.30**	64.35 74.58	25.60 43.00
784,∘,∘,∘,2	68.99 75.06	**78.20 80.18**	65.95 71.38	19.81 42.62
784,∘,∘,∘,∘,2	65.97 75.44	14.42 **80.20**	10.28 63.05	11.76 12.06
784,∘,∘,∘,∘,∘,2	10.28 22.72	10.28 12.99	17.51 10.28	10.28 10.16

(d) sigmoid with LFKNN, $C_{|X|} = 0$

III. Experiment and Results

The objective functions are minimized using gradient descent. Optimization of the objective functions L_G, $L_{G,X}$, and $L_{G,X,D}$ first begins with random initializations for the elements of X, and the combined objective $L_{BC,X,D}$ begins with a random initialization of the parameters θ. Theano [10] is capable of symbolically obtaining the gradients of an objective function with respect to X or θ. Note that the pairwise differences needed in the covariance matrix K can be obtained symbolically using Theano's broadcasting rules (derived from NumPy's broadcasting rules) while avoiding loops.

The results of optimizing the combined objective function $L_{BC,X,D}$ are shown in Table I and Figure 6 for fixed β, S_w, $C_{|X|}$, network depth, and nonlinearity. Table I shows how frequently unseen testing images are classified correctly, and Figure 6 shows the mapping of all training images along with decision boundaries. The results of SVC and lava floor k-nearest neighbor (LFKNN) are shown separately, but the underlying networks are identical for the same sets of parameters; Figure 6 (f) and (h) are direct counterparts. For reference, a support vector classifier trained on the 100 labeled points in the original image space achieves 70% accuracy on unseen test data.

All of the results coming from $L_{BC,X,D}$ have the norm penalty $C_{|X|}$ set to zero except for the cases in which the nonplus nonlinearity is involved. The softsign and sigmoid nonlinearities are bounded functions, whereas the nonplus nonlinearity is unbounded, and thus softsign and sigmoid do not need a norm penalty term in the objective function. Values of $C_{|X|} < 1000$ consistently led to poor classification results for the nonplus nonlinearity.

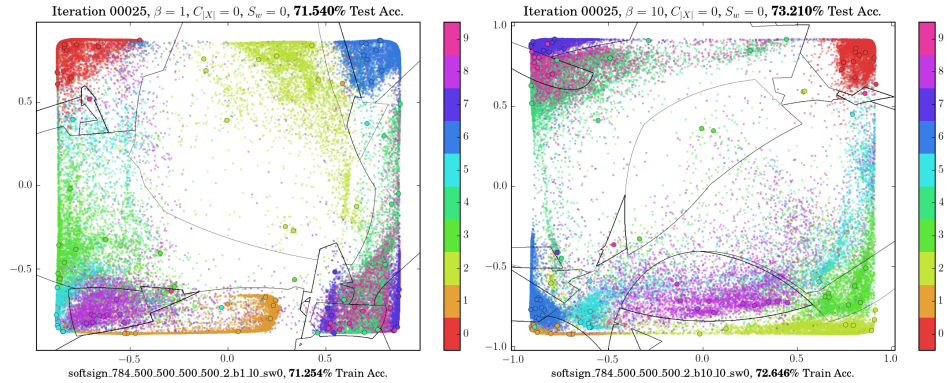
IV. Discussion

The results of the case study highlight five areas of discussion: the length scale β, the formation or lack of clusters caused by varying $C_{|X|}$, the utilization of space by different nonlinearities, the effects of network depth, and the value of the discriminative objective.

The length scale parameter β seems to have very little impact on the standard GPLVM objective function L_G for high dimensional data, as can be seen in Figure 2. The latent positions move far enough apart that, regardless of β, no two points in the latent space have strong interactions with one another. For the norm-penalized and the back-constrained versions of the objective functions, β needs to be selected appropriately in order to obtain the best results, as evidenced by the last two columns of Table I.

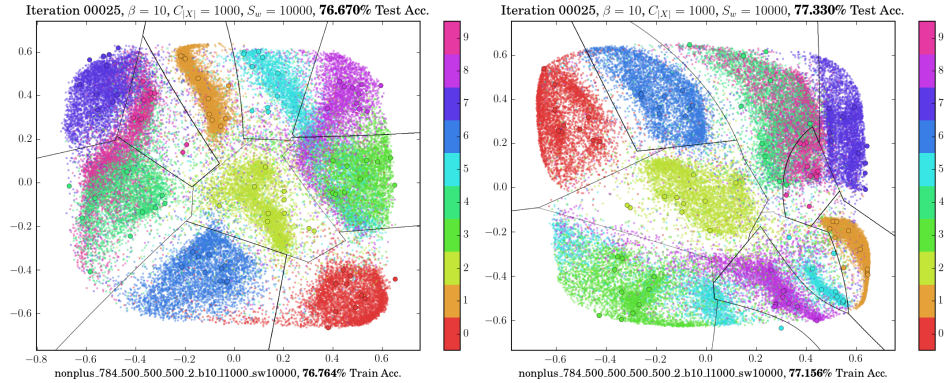
We find it surprising that GPLVM and DGPLVM do not produce "clusters" unless points are forced to intermingle. In the latent space, GPLVM is known to preserve the dissimilarity of data points as opposed to the distances between points [6], but the ballooning of the space is unusual. Thanks to the dissimilarity preservation and despite the ballooning, GPLVM organizes points well, as can be seen best in Figure 3 (a) where even though all of the points are separated, the zeros (colored red) are in the bottom left and all of the ones (colored orange) are near the top right. What surprises us even more is the effect, the fear that unlabeled points have of labeled points, of forcing together known instances of each class by adding the discriminative objective L_D, as seen in Figure 4. This issue appears only in the label-deficient setting and not in the fully-labeled setting in which DGPLVM was originally proposed [7]. Neither the ballooning nor the fear are relevant with large $C_{|X|}$ or with bounded back-constraints.

Among the three nonlinearities that we tested, sigmoid consistently achieved the best performance. We think this is because of two reasons. The primary reason is that the exponentiated quadratic covariance function, the Gaussian likelihood, and the sigmoid function all have exponential components, thus allowing data points to move and to be placed freely in the entire usable space. Comparing Figure 6 (b) and (e), we can see clearly that the learned encoder in the softsign case cannot use the center, and thus almost all of the encoded data points are forced to the edges of the space, whereas the encoder in the sigmoid case has no trouble filling the space naturally. The second reason is that the sigmoid nonlinearity, due to the fact that it is a bounded function,



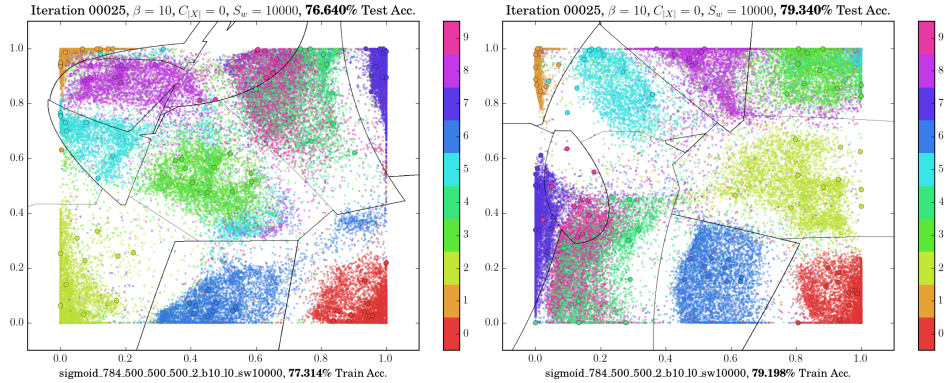
(a) softsign with SVC, 71.54%

(b) softsign with SVC, 73.21%



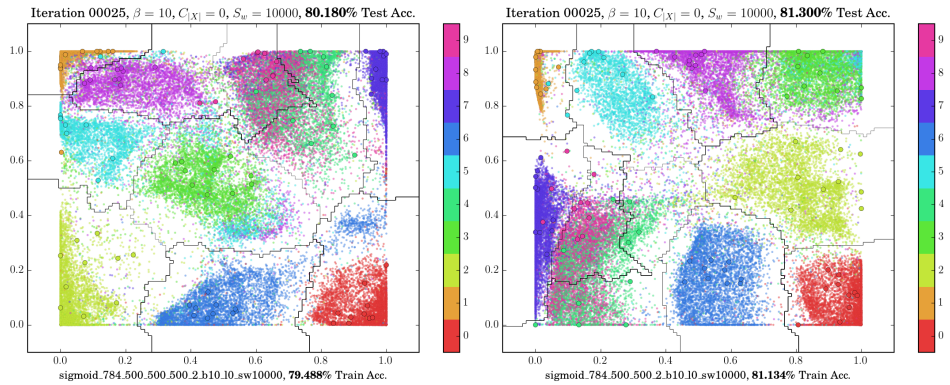
(c) nonplus with SVC, 76.67%

(d) nonplus with SVC, 77.33%



(e) sigmoid with SVC, 76.64%

(f) sigmoid with SVC, 79.34%



(g) sigmoid with LFKNN, 80.18%

(h) sigmoid with LFKNN, 81.30%

Fig. 6: Visualizations of the encodings of all training images in learned spaces along with classification accuracies and boundaries found by SVC and LFKNN. The two best results from each of the four subtables of Table I are shown.

does not need $L_{|X|}$, allowing the learning process to focus more on the remaining sub-objectives. Comparing Figure 6 (d) and (f), we can see that the usage of space in both cases is very similar with the exception of the edges and corners that are less sharp in the case of the nonplus nonlinearity. When the nonplus nonlinearity is forced to have a large norm penalty $C_{|X|}$ in order to have reasonable results, it basically becomes like sigmoid, but ever so slightly worse, as evidenced by Table I. We originally wanted to use rectifying linear units and the softplus nonlinearities [11], but their shapes led to the matrix K becoming singular and thus non-invertible.

The best classification results, according to Table I, come from networks with two to four hidden layers. Networks with fewer layers are not powerful enough to freely control positions in the latent space according to GPLVM's guidance. Networks with too many layers suffer the traditional problems associated with deep networks, mainly that deep networks tend to be degenerate when randomly initialized, and they are not necessarily capable of recovering from all initializations. We are unsure if deeper networks will yield better results for the MNIST dataset, but we see that even shallow networks can produce good results.

The inclusion of the discriminative objective L_D improves classification performance. In each cell of Table I, the accuracies to the right, corresponding to the discriminative case, are larger than the accuracies to the left. The softsign results are an exception but largely irrelevant because softsign's performance is very poor overall in this setting. The LFKNN classifier outperforms SVC in the best cases, cases in which the classes are well separated and the discriminative objective guided all instances of the same class to be very near each other. In those few cases, using unlabeled points' locations to inform the classification boundaries' locations works well. We are unsure if the labeled points are typical or outliers of their respective classes. Typical data points would more naturally fall into their classes' clusters even without the addition of L_D, but outliers would force an informative warping of the space with the addition of L_D.

V. Conclusion

GPLVM can be used in a label-deficient setting by following the proposed two-stage method. We have shown how to train neural networks using GPLVM objectives, and the proposed lava floor distance successfully makes the learned knowledge available for classification. The case study reveals the behaviors of the individual components of the two-stage method, and it also provides results that highlight the behaviors of the entire method. Visualizations made possible by bottlenecking all networks through a two-dimensional space provide valuable insights.

For the standard GPLVM objective L_G, Figure 2 shows that high-dimensional data points are anti-social in the sense that none of them rest close together or are clustered in the reduced-dimension space regardless of the covariance's length scale parameter β. The norm-penalizing objective $L_{|X|}$ forces points together, with Figure 3 showing that large penalties lead to the formation of clusters.

The objective L_D captures label information to help guide the dimension reduction, but Figure 4 shows that even though the labeled points in each class form tight clusters, the unlabeled points stay far away from the labeled points. Using a strong weighting on $L_{|X|}$ forces all points to interact and leads to the expected formation of clusters.

The back-constrained objective L_{BC} extends the capabilities of GPLVM to handle large data sets. Based on Table I and Figure 6, the sigmoid nonlinearity is particularly well-matched to GPLVM and the exponentiated quadratic covariance function. By comparison, the softsign nonlinearity is not able to fully utilize the reduced dimension space, leading to many points being degenerately forced to the edges. Deep networks are not required in order to get good results; the best classification accuracies come from networks with as few as two hidden layers.

The lava floor distance, coupled with simple k-nearest neighbor classifiers, successfully extends networks trained with the GPLVM objectives to label-deficient settings. Comparing Table I (c) and (d) shows that the best LFKNN test-data accuracies are consistently better than the best SVC accuracies. In both cases, the discriminative objective L_D improves performance. The proposed method outperforms the baseline standard support vector classifier trained only on the 100 labeled points in the full-dimension image space.

References

[1] J. Weston, F. Ratle, H. Mobahi, and R. Collobert, "Deep learning via semi-supervised embedding," in *Neural Networks: Tricks of the Trade*. Springer, 2012, pp. 639–655.

[2] S. Rifai, Y. N. Dauphin, P. Vincent, Y. Bengio, and X. Muller, "The manifold tangent classifier," in *Advances in Neural Information Processing Systems*, 2011, pp. 2294–2302.

[3] N. Pitelis, C. Russell, and L. Agapito, "Semi-supervised learning using an unsupervised atlas," in *Machine Learning and Knowledge Discovery in Databases*. Springer, 2014, pp. 565–580.

[4] D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling, "Semi-supervised learning with deep generative models," in *Advances in Neural Information Processing Systems*, 2014, pp. 3581–3589.

[5] N. D. Lawrence, "Gaussian process latent variable models for visualisation of high dimensional data," *Advances in neural information processing systems*, vol. 16, no. 3, pp. 329–336, 2004.

[6] N. D. Lawrence and J. Quiñonero-Candela, "Local distance preservation in the gp-lvm through back constraints," in *Proceedings of the 23rd international conference on Machine learning*. ACM, 2006, pp. 513–520.

[7] R. Urtasun and T. Darrell, "Discriminative gaussian process latent variable model for classification," in *Proceedings of the 24th international conference on Machine learning*. ACM, 2007, pp. 927–934.

[8] O. Chapelle and A. Zien, "Semi-supervised classification by low density separation," in *Proceedings of the tenth international workshop on artificial intelligence and statistics*, vol. 1, 2005, pp. 57–64.

[9] A. X. Falcão, J. Stolfi, and R. de Alencar Lotufo, "The image foresting transform: Theory, algorithms, and applications," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 26, no. 1, pp. 19–29, 2004.

[10] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, "Theano: a CPU and GPU math expression compiler," in *Proceedings of the Python for Scientific Computing Conference (SciPy)*, June 2010, oral Presentation.

[11] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 2010, pp. 807–814.