APPENDIX

TRAINING DETAILS

We report the architecture and hyper-parameters used in the experiments in Section VI for each of the data sets. We used for most of the data intensive experiments Ubuntu 16.04 on a machine with 2x Intel Xeon Gold 6126 CPU @2.6GHz/3.7GHz 12C/24T each, 2x512GB SSD for system (RAID0 = 512GB usable), and 4xNvidia GTX 1080Ti 11GB GPUs.

a) Active learning training: After each batch of queries the network is trained for a certain number of epochs, starting from the previous configuration. The reported accuracies are averaged over 5 runs. All the experiments were performed using PyTorch [18].

b) Checkerboard: We consider a pool data set of $|X_{\text{pool}}| = 2000$ labeled points drawn from the checkerboard distribution. We start with a training set X_ℓ^0 composed of 4 points randomly drawn from the pool set for each of the classes (so that $\left|X_\ell^0\right| = 8$) and train a feed-forward neural network. The accuracy and its variance are measured on a separate test data set of $N_{\text{test}} = 200$ points drawn from the checkerboard distribution in Fig. 2. All the results are averaged over 5 runs.

We trained a fully-connected network with 2 hidden layers of width 30 each. We optimized using SGD with batch size 1, learning rate 0.001 and momentum 0.9. We ran 100 epochs after each query. The experiments were run on a pool set of size 2000. The accuracy was evaluated on a separate test data set sampled from the same distribution (200 points).

c) MNIST: The fully-connected model we used had 2 hidden layers of width 100 and 50 respectively. The convolutional model used was composed by a convolutional layer with 16 channels and a kernel of size 5×5, a MaxPool layer with a kernel of size 2 and padding 2, and 2 hidden fully-connected layers of width 20 each. For both models, we optimized using Adam [12] with batch size 8 and learning rate 0.001. For both models, a BatchNorm [10] layer was added before each hidden fully-connected layer. We ran 100 epochs after each query. The experiments were run on a pool set formed by a (balanced) randomly selected subset of the training data set of size 10000. The accuracy was evaluated on the test data set (10000 points).

d) CIFAR10 : The network used was a VGG-16 architecture pre-trained on ImageNet. We took the convolutional part of such network and added 2 fully-connected layers of width 512 and 20 respectively. A dropout layer was added after each of these fully-connected layers. Only the fully-connected layers were trained. We optimized using Adam [12] with batch size 100. After each query the learning rate was initialized to 0.0003 and decayed by 0.5 every 30 epochs. We ran 100 epochs after each query. The full training data set (50000 points) was used as pool data set for the experiments. The accuracy was evaluated on the test data set (10000 points).

e) SVHN : The network used was a VGG-16 architecture. We optimized using SGD with batch size 50, learning rate 0.005, momentum 0.9 and weight-decay 0.0005. We ran 50 epochs after each query. The experiments were run on a pool set formed by a (balanced) randomly selected subset of the training data set of size 20000. The accuracy was evaluated on the test data set (26032 points).

f) Openml155: For the OpenML, we use a two-layer Perceptron with ReLU activations (MLP) as in [11]. The embedding dimensionality of the MLP is 1024, as more capacity helps the model fit training data. We fit models using cross-entropy loss and the Adam variant of SGD. We use a learning rate of 0.0001.

g) DDAL: The following parameters were used for the diffusion algorithm in the experiments presented in Section VI. For the experiment with the checkerboard data set (experiment in Figure 2), we used $T = 4$, $K = 10$ and $P = 1$. For the experiments with the MNIST data set (experiment in Figure 4), we used $T = 5$, $K = 10$ and $P = 1$. For the experiment with the CIFAR10 data set (experiment in Figure 4), we used $T = 4$, $K = 20$ and $P = 10$.

h) Bayesian criterion: In order to perform the active learning queries using the Bayesian criterion, we added a dropout layer after each hidden fully connected layer to the models with no dropout layers.

A. Variants of multi-class extension

In Section IV, we described a possible formulation to extend the diffusion-based active learning criterion to the multi-class setting. We propose in the following other possible formulations.

a) One-vs-all approach: In the one-vs-all setting described in Section IV, the batch is queried according to

$$\hat{X} = \arg\min_{i \in X_u}^{B} q\left(\chi_{:,i}^{(T)}\right)$$

for some chosen function q which measures a notion of uncertainty at point x_i, given the matrix $\chi^{(T)}$. In Section IV, we chose q to be

$$q\left(\chi_{:,i}^{(T)}\right) = \min_{c \in [C]} \left|\chi_{c,i}^{(T)}\right|, \text{ or } q\left(\chi_{:,i}^{(T)}\right) = \left\|\chi_{:,i}^{(T)}\right\|_p$$

for some $p \in [1, \infty]$.

b) Multivariate diffusion approach: Moving from the one-vs-all approach, we can performs the query as follows, using the property that M is a stochastic matrix. For each data point x_i, we propagate a probability vector $\chi_i^{(t)} \in \Delta_C$. This vector can be initialized as

$$\chi_{i,c}^{(0)} = \begin{cases} 1 & \text{if } i \in X_\ell \text{ and } c = y_i \\ 0 & \text{if } i \in X_\ell \text{ and } c \neq y_i \\ \frac{1}{C} & \text{otherwise} \end{cases}$$

We can therefore diffuse the matrix aggregating the signal for all the points, $\chi^{(t)} \in \mathbb{R}^{N \times C}$, and diffuse it as in the binary case:

$$\chi^{(t)} = M\chi^{(t-1)}, \qquad \chi^{(t)}|_{X_\ell} = \chi^{(0)}|_{X_\ell}$$

Since M is stochastic, it holds that $\chi^{(t)} \in \Delta_C$ at each iteration t. Therefore we can interpret each vector $\chi_i^{(t)}$ as a probability vector of the data point x_i belonging to different classes,

obtained by the diffusion above. It therefore makes sense to choose the points to query as

$$\hat{X} = \arg\min_{k \in X_u}^{B} q\left(\chi_i^{(T)}\right)$$

where $q : \Delta_C \to \mathbb{R}$ is some measure of uncertainty. Possible choices include:

- Uncertainty: $q(p) = p_{c^*}$, where $c^* = \arg\max_c p_c$;
- Margin: $q(p) = p_{c^*} - p_{c_2^*}$, where c^* is defined as above and $c_2^* = \arg\max_{c \in [C] \setminus \{c^*\}} p_c$;
- Negative entropy: $q(p) = \sum_{c \in [C]} p_c \log p_c$.

B. PROOF OF LEMMA 1

Proposition 1. *A general solution to the Jacobi iteration (**??**) after t iterations and with initial conditions specified by $\chi^{(0)}$ is given by*

$$\chi^{(t)} = (M)^t \chi^{(0)} = c_1 \lambda_1^t \phi_1 + ... + c_n \lambda_n^t \phi_n, \qquad (10)$$

where $\chi^{(t)}_{i:x_i \in X_l} = y_i$, for each $t \in \{0, ..., T\}$, and $c_1, ..., c_n$ are coefficients that are prescribed by the initial condition $\chi^{(0)} = c_1\phi_1 + ... + c_n\phi_n$, where $\lambda_1, ..., \lambda_n$ and $\phi_1, ..., \phi_n$ are the eigenvalues and eigenvectors of M.

Via a simple algebraic manipulation those eigenvectors are also shown to be the eigenvectors of the graph's Laplacian:

Proposition 2. *$\lambda_1, ..., \lambda_n$, $\phi_1, ..., \phi_n$ are also solutions to the system:*

$$L\phi = (1 - \lambda)D\phi, \qquad (11)$$

where $L = D - W$.

The eigenvector ϕ_2 is a solution to the relaxed Minimal Normalized Cut problem (*MinNCut*) [5] in a graph $G(V, E)$ constructed from data representation in the hidden layer f_θ^n:

$$MinNCut = \arg\min_{h : h^T D \mathbf{1} = 0} \frac{h^T L h}{h^T D h}. \qquad (12)$$

We follow on the proposition 1 eq. (10). We first note that $\chi^{(0)}$ has the sign of its non-zero coordinates as the subset of the coordinates of ϕ_2 (up to a sign permutation), and after each iteration they are restarted. Since ϕ_1 has constant sign (and magnitude) restarting with opposite signs causes c_1 to be suppressed to zero already at $t = 0$. Since $\phi_2 \perp \phi_i$ for $i = 3, ..., N$, there have to be coordinates $j_1, ..., j_k$; $j_k \in \{1, ..., N - 1\}$, for which $sign(\phi_i(x_{j_m})) \neq sign(\phi_2(x_{j_m}))$ $1 \leq m \leq k$. We consider two cases:

1) $\chi^{(0)}$ contains such non-zero coordinates. For all eigenvectors $\phi_{i_1}, ..., \phi_{i_l}$ with such sign inequality we have that $c_2 > c_{i_l}$ for $i_l \in \{3, ..., N\}$, and since $\lambda_2 \geq \lambda_{i_l}$ we have that ϕ_2 is dominant. The restarting of χ prevents it from converging to 0 in case $c_2\lambda_2 < 1$, and we conclude our proof. For all other eigen-components (for which $\chi^{(0)}$ has 0 coordinates where $sign(\phi_i(x_{j_m})) \neq sign(\phi_2(x_{j_m}))$ $1 \leq m \leq k$) the following case also holds.
2) All coordinates $j_1, ..., j_k$ in $\chi^{(0)}$ are zero: Since for a well-separated GMM the kernel K is a 2-block stochastic matrix, point x_{j_m} will be transduced only from points x_l for which $y_{j_m} = y_l$. For every such point x_{j_m} there exist an iteration t' such that $\chi^{(t')}(x_{j_m})$ becomes non-zero and attains the sign of y_l. At this t' all eigen-components ϕ_i, $i = 3, ..., N$ in the iterant $\chi^{(t')}$, which have coordinates such that $sign(\phi_i(x_{j_m})) \neq sign(\chi^{(t')}(x_{j_m}))$ $1 \leq m \leq k$, will be reduced. Once all such points are visited the dominant component will be ϕ_2, and the remaining components will converge to zero.

C. PROOF OF PROPOSITION 1

The eigenvectors of the kernel M form a complete basis and therefore any vector in $\mathbb{C}^N$ can be represented as their linear combination. What is left to prove is that any initial conditions can be matched. To this end we need to show that

$$Sc = \chi^{(0)} \qquad (13)$$

can be solved, where S is the matrix whose columns are the eigenvectors of B_J (the iteration matrix introduced in Section III for the iteration in eq. (**??**)). Since the eigenvectors are linearly independent S is non-singular and (10) always admits a solution.

D. PROOF OF PROPOSITION 2

Simple algebraic operations yield the result:

$$D^{-1}W\phi = \lambda I\phi \Leftrightarrow \phi K\phi = \lambda I\phi \Leftrightarrow W\phi = \lambda D\phi \Leftrightarrow$$
$$(D - L)\phi = \lambda D\phi \Leftrightarrow L\phi = D\phi - \lambda D\phi \Leftrightarrow L\phi = (1 - \lambda)D\phi$$

E. PROOF OF THEOREM 1

We note that eq. (8) is derived from several steps and after reducing constant factors. We elaborate these steps below:

1) The first term of (8) depends on the balancedness β, and accounts for the number of queries required to achieve a representative for every class. The ratio in (8) is a direct result of Lemma 4 in [7].
2) Since at every diffusion step K nodes are transduced with a label, K^T nodes will have a soft label after diffusing from a labelled node (assuming a connected graph). On the other hand, the approximate number of labelled nodes needed to cover the graph with soft labels can be derived in expectation from the ratio $\frac{N}{K^T}$ for connected graphs. When $T = O(\log_K N)$ all $O(N)$ unlabelled nodes are reachable in expectation in T iterations, and $O(1)$ labelled nodes are needed. Exploration involves $O(\frac{N}{K^T})$ queries to guarantee all nodes have non-zero labels via diffusion.
3) The search process is dictated by the minimal value of the diffusion soft label (criterion (6)) obtained after each propagation of the labels from a pair of nodes v_i v_j of opposite labels, to their neighbors along the shortest path between them. This propagation repeats until two nodes of opposite sign are discovered. Since the absolute value among those two nodes is expected to be minimal, our query criterion selects one of those nodes to be queried, and the process repeats until his opposite label neighbor is queried as well. For a regular graph grid in which all the transition probabilities are equal, the queried node will reside at the midpoint between v_i and v_j, thus resulting in $O(\log M)$ queries, where M is the length of the shortest path between them.

Since after exploration with $T = O(\log_K N)$ the maximal shortest distance between any two nodes v_i, v_j, of opposite label is at most $2K^T$ hops, we have that at most $\log 2K^T = 1 + T \log K = 1 + \log_K N \cdot \log K$ queries are required until the first two connected cut nodes are recovered. Since in a connected grid each node has $K = 2^d$ neighbors. We obtain at most $1 + d \log_K N$ queries are sufficient.

4) The next propagation will reach another cut node in a constant number of propagations, specifically for any grid in just 2 hops. Since this is a constant the following queries scale as the size of the cut - $|\partial C|$.

F. PROOF OF THEOREM 2

We start with auxiliary results (proved below) that show that the diffusion iteration converges to a minimal cut solution in the graph $G(V, E, W)$ derived from the top hidden layer, under the separation requirement. As a result, the decision boundary is recovered accurately because the GMM cluster assignments correspond to the ground truth labels.

We start with observing the general form of the solution of the iteration (**??**) as in Proposition 1:

$$\chi^{(t+1)} = M^{t+1}\chi^{(0)} = c_1\lambda_1^t\phi_1 + ... + c_n\lambda_n^t\phi_n, \tag{14}$$

where $\chi^{(t)}_{l:x_i \in X_l} = y_i$, for each $t \in \{0, ..., T\}$, and $c_1, ..., c_n$ are coefficients that are prescribed by the initial condition $\chi^{(0)}$: $\chi^{(0)} = c_1\phi_1 + ... + c_n\phi_n$, and $\lambda_1, ..., \lambda_n$ and $\phi_1, ..., \phi_n$ are the eigenvalues and eigenvectors of M.

Let $\lambda_1, ..., \lambda_n$ and $\phi_1, ..., \phi_n$ be the set of eigenvalues and eigenvectors of M ordered in the decreasing order of the eigenvalues. Using Proposition 2 and Lemma 1 we show the equivalence of the iteration solution as $t \to \infty$ to the Minimal Normalized Cut solution (*MinNCut*) [5].

As achieved by Lemma 1, the construction of the decision boundary can be done by following the convergence to the minimal cut solution. We therefore show that the sample complexity is dominated by the initial class sampling and the sampling in the exploration stage, which allows the labels to propagate through the whole graph for a given T.

1) At the initial stage data is queried until at least one sample of each label exists. Assuming that β is the measure of balancedness (as definition 2), the first component to consider is the complexity required to discover all classes, following on Lemma 4 in [7].
2) Next, we are looking to sample a set of labelled nodes such that when diffused T times all unlabelled nodes will have a non-zero soft label. As discussed above in the proof of Theorem 1, the expected number of queries is $O(\frac{N}{KT})$ to guarantee coverage by diffusion for almost all nodes in T diffusion steps. On the other hand, setting $T = \log_K N$, renders this complexity as constant $O(1)$, independent of N, which can be neglected.
3) Once all such points are queried the minimal cut corresponding to the Gaussian clusters is obtained accurately according to Lemma 1, for sufficiently large T via the diffusion process. The optimality of the spectral cut solution here is supported by Theorem 1.1 in [16], which we give below for completion:

   Theorem 3. *For N data points generated from a Gaussian Mixture model, if*

   - *number of clusters is finite*
   - *the sizes of clusters are in the same order*
   - *the minimum distance among centers goes to infinity*
   - *the dimension d is at most in the same order of N*

   then with high probability, spectral clustering achieves the optimal clustering rate, which is

$$l(\hat{\chi}, \chi^*) = N \exp\Big(- (1 - o(1))\frac{\triangle^2}{8}\Big) \tag{15}$$

   where $l(\cdot, \cdot)$ is the Humming loss function, and $\triangle$ is the distance between the centroids.

   The theorem provides the final step, showing that ϕ_2 is an optimal solution for finding the decision boundary between the Guassians in the GMM.

We note that the assumptions of Theorem 3, in particular, the high separation between the class-clusters (here modeled by the GMM) are attained due to the highly trained network at the refinement stage. In this stage the penultimate layer is well trained and present a structured graph where members of different class are separated.