Nearly Optimal Semi-Supervised Learning on Subgraphs

Gayatree Ganu
Rutgers University
gganu@cs.rutgers.edu

Branislav Kveton
Technicolor Labs, Palo Alto
branislav.kveton@technicolor.com

ABSTRACT

The harmonic solution (HS) on a graph is one of the most popular approaches to semi-supervised learning. This is the first paper that studies how to identify highly confident HS predictions on a graph based on the HS on its subgraph. The premise of our method is that the subgraph is much smaller than the graph and therefore the most confident predictions can be identified much faster than computing the HS on the graph. We introduce a class of subgraphs that allow for good approximations, prove bounds on the difference in the HS on the graph and its subgraph, and propose an efficient approach to building the subgraphs. Our solution is evaluated in the domains of handwritten digit recognition, and topic discovery in restaurant and hotel reviews. In all cases, we show that only a small portion of the graph is sufficient to identify highly confident predictions.

1. INTRODUCTION

Semi-supervised learning is a field of machine learning that studies learning from both labeled and unlabeled examples. In practice, large amounts of data are available but only a tiny fraction of these data is labeled. Such problems can be often cast as semi-supervised learning problems and various methods for solving these problems exist [17]. The focus of our work is graph-based learning [18], and in particular the harmonic solution (HS) on graphs, which serves as a basis for many semi-supervised learning algorithms [2, 17].

In this work, we show how the harmonic solution on a graph can be approximated by the HS on its subgraph, with provable guarantees on the quality of the approximation. The premise of our paper is that the subgraph is much smaller than the graph and thus the HS on the subgraph can be computed more efficiently. We state conditions under which the subgraph yields a good approximation, prove bounds on the quality of the approximation, and show how to build the graph efficiently. Our method is evaluated on handwritten digit recognition and two bigger problems, topic discovery in restaurant and hotel reviews. Our experimental results support our claims that only a small portion of the graph is usually needed to identify high confidence predictions on the entire graph.

This paper addresses an important problem. In practice, data are large-scale and often only a small portion of labels can be inferred with high confidence. The low confidence predictions are typically of little utility because they are noisy. In this work, we show how to identify high confidence predictions without the overhead of modeling the low confident ones. This is the first such result for graph-based semi-supervised learning.

Our algorithm for building subgraphs can be viewed as a form of self-training (Section 2.2). Similarly to self-training, the algorithm is iterative and easy to implement. Unlike self-training, we provide guarantees on the quality of the solution. In other words, we show how to do self-training in a proper way.

The paper is organized as follows. First, we review related work and introduce basic concepts, such as semi-supervised learning on graphs (Section 2). Second, we motivate our approach, analyze the error in the estimate of the harmonic solution on the subgraph, and propose a method for building the subgraph (Section 3). Third, we evaluate the method on three datasets, two of which are large-scale (Sections 4 and 5). Finally, we conclude and discuss future work.

We adopt the following notation. The symbols $\mathbf{x}_i$ and y_i refer to the i-th example and its label, respectively. All examples belong to one of c classes $y \in \{1, \ldots, c\}$. The examples are divided into the labeled and unlabeled sets, l and u, and we only know labels in the labeled set l. The cardinality of these sets are n_l and n_u. The total number of the examples is $n = n_l + n_u$.

2. RELATED WORK

Semi-supervised learning can be formulated as label propagation on a graph, where the vertices are the examples $\mathbf{x}_i$ [18]. The labels can be computed by solving a system of linear questions:

$$F_u = -(L_{uu})^{-1} L_{ul} F_l, \tag{1}$$

where $F \in \{f_i[k]\}^{n \times c}$ is a matrix of probabilities that the vertex i belongs to the class k; F_l and F_u are the rows of F corresponding to the labeled and unlabeled vertices, respectively; $L = D - W$ is the *Laplacian* of the data adjacency graph W; and D is a diagonal matrix whose i-th diagonal entry is computed as $d_i = \sum_j w_{ij}$. We refer to the i-th row of F as $\mathbf{f}_i = (f_i[1], \ldots, f_i[c])$ and to the *most confident prediction* in the row as $\|\mathbf{f}_i\|_\infty = \max_k |f_i[k]|$. We adopt the convention that vertices are indexed by i and j, and labels by k.

The solution F_u is known as the *harmonic solution (HS)* because it satisfies the *harmonic property* $f_i[k] = \frac{1}{d_i} \sum_{j \sim i} w_{ij} f_j[k]$. It can be rewritten as:

$$\begin{aligned} F_u &= (I - P_{uu})^{-1} P_{ul} F_l \\ &= (I + P_{uu} + P_{uu}^2 + \ldots) P_{ul} F_l, \end{aligned} \tag{2}$$

where $P = D^{-1} W$ is a transition matrix of a random walk on W. As a result, $f_i[k]$ can be viewed as the probability that the random

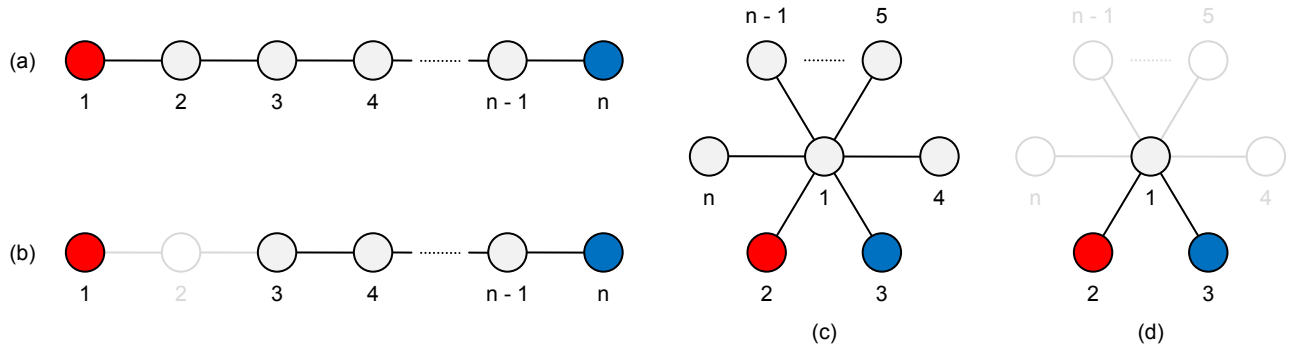

Figure 1: a. A *line graph* of n vertices. b. A subgraph of the line graph on vertices $e = \{1, 2, 3, \ldots, n\}$. c. A *star graph* of n vertices. d. A subgraph of the star graph on vertices $e = \{1, 2, 3\}$.

walk that starts from the vertex i is absorbed at vertices with labels k [18]. Our work relies heavily on this interpretation.

2.1 Large-scale semi-supervised learning

In general, the space and time complexity of computing the harmonic solution (Equation 1) are $\theta(n^2)$ and $\theta(n^3)$, respectively. So it is challenging to compute the exact solution when the number of vertices n exceeds several thousand. The computation can be sped up significantly by taking into account the structure of the problem. For instance, when the graph W is $O(n)$ sparse, the time complexity of computing the HS is $\theta(n^2)$ [17]. Moreover, when the system of linear equations (Equation 1) is properly preconditioned [12], it can be solved approximately in nearly linear time in n. Data dependent kernels for semi-supervised max-margin learning can be also built in nearly linear time [9].

A few methods can approximate the HS in $\theta(n)$ time [13, 6, 15]. Fergus *et al.* [6] cast this problem as regression on basis functions, which are eigenfunctions over features. Valko *et al.* [15] choose k representative vertices, compute the HS on these vertices, and then propagate the solution to the rest of the graph. The space and time complexity of this method are $\theta(k^2)$ and $\theta(kn + k^3)$, respectively. Our solution is later compared to this approach.

Our work is orthogonal to the existing work on large-scale semi-supervised learning on graphs. In particular, we study the problem where only a small portion of the graph, typically in the vicinity of labeled vertices, is sufficient to identify most confident predictions and show how this subgraph can be learned efficiently. Our method can be easily combined with existing approximations, such as data quantization [15]. In this case, we would learn an approximation to the subgraph.

2.2 Self-training

Self-training [16] is one of the oldest and most popular methods for semi-supervised learning. In self-training, a classifier is initially trained on a few labeled examples. Then it is used to predict labels of unlabeled examples, the most confident predictions are added to the training set, the classifier is retrained, and this is repeated until a stopping criterion is met. Self-training is very common in natural language processing, and was applied to various problems, such as named-entity [5, 10] and relation [3, 1, 11] extraction.

The disadvantage of self-training is that it is subject to local optima and does not provide guarantees on the quality of the approximation [18]. Our algorithm for learning ε-subgraphs (Algorithm 1) can be viewed as a type of self-training. Similarly to self-training, the method is iterative and easy to implement. Unlike self-training, we provide guarantees on the quality of the solution.

3. INFERENCE ON A SUBGRAPH

This section is organized as follows. First, we introduce our setting and demonstrate subgraph inference on two problems. Second, we identify a class of subgraphs called ε-subgraphs, and bound the quality of the HS approximations for these graphs. Finally, we propose a technique for constructing ε-subgraphs and discuss how to apply it in practice.

3.1 Subgraphs

We want to efficiently approximate the harmonic solution on the graph by the HS on its subgraph.

DEFINITION 1. *A* subgraph $W[e]$ *of a graph* W *induced by vertices* e *is a graph over* e *where two vertices are adjacent if and only if they are adjacent in* W*. The subgraph* $W[e]$ *is given by a* $n \times n$ *adjacency matrix:*

$$(W[e])_{ij} = \begin{cases} W_{ij} & (i \in e) \wedge (j \in e) \\ 0 & \text{otherwise.} \end{cases}$$

In other words, the edge W_{ij} *appears in the subgraph* $W[e]$ *only if both* i *and* j *are subgraph vertices* e*.*

We refer to the harmonic solutions on W and its subgraph $W[e]$ as F^* and F^e, respectively. The respective solutions at the vertex i are denoted by $\mathbf{f}_i^*$ and $\mathbf{f}_i^e$. If the vertex $i \notin e$ is not in the subgraph, we assume that $\mathbf{f}_i^e = \mathbf{0}$.

Our goal is to learn a subgraph such that the difference between the harmonic solutions $\mathbf{f}_i^*$ and $\mathbf{f}_i^e$ is bounded at vertices $i \in e$. The difference is measured by the *max-norm distance*:

$$\|\mathbf{f}_i^e - \mathbf{f}_i^*\|_\infty = \max_k |f_i^e[k] - f_i^*[k]| \,. \tag{3}$$

We opt for this distance because it allows us to identify highly confident predictions on W. In particular, note that when $\|\mathbf{f}_i^e - \mathbf{f}_i^*\|_\infty$ is small and the confidence $f_i^e[k]$ on the subgraph is high, then the corresponding confidence $f_i^*[k]$ on W is bounded from below as:

$$\begin{aligned} f_i^*[k] &= f_i^e[k] - (f_i^e[k] - f_i^*[k]) \\ &\geq f_i^e[k] - \max_k |f_i^e[k] - f_i^*[k]| \\ &= f_i^e[k] - \|\mathbf{f}_i^e - \mathbf{f}_i^*\|_\infty \,, \end{aligned} \tag{4}$$

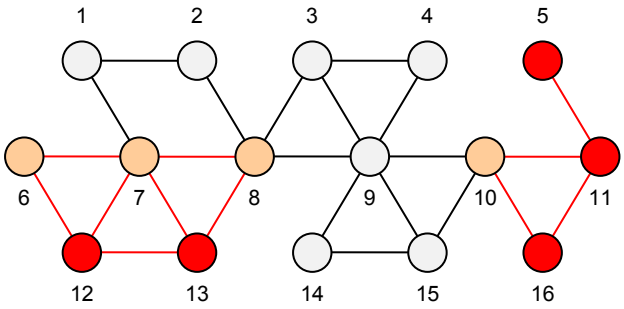

Figure 2: An ε-subgraph over nine vertices. The dark and light red vertices are the core and surface vertices of the ε-subgraph, respectively. The subgraph edges are shown in red color.

and is also high. In the rest of the section, we discuss two examples of inference on subgraphs.

The first example is a *line graph* of n vertices (Figure 1a):

$$W_{ij} = \mathbb{1}\{|j-i| = 1\} \quad \forall\, i, j \in \{1, \ldots, n\}\,. \tag{5}$$

In this graph, all consecutive vertices, i and $i+1$, are adjacent. The vertices 1 and n are labeled as classes 1 and 2, respectively. Let the subgraph $W[e]$ be induced by $n-1$ vertices $e = \{1, 3, 4, \ldots, n\}$ (Figure 1b), all but the vertex 2. Moreover, let the size of the graph n increase. Then note that $\mathbf{f}_3^* \to (1, 0)$ because the distance of the vertex 3 to labeled vertices 1 and n remains constant and increases linearly with n, respectively. On the other hand, $\mathbf{f}_3^e = (0, 1)$ for all n because the labeled vertex 1 cannot be reached from the vertex 3. So the distance between the harmonic solutions $\|\mathbf{f}_3^e - \mathbf{f}_3^*\|_\infty \to 1$. In other words, the HS on subgraphs can be inaccurate, even when the subgraph covers a large portion of the entire graph W.

The second example is a *star graph* of n vertices (Figure 1c):

$$W_{ij} = \mathbb{1}\{(i = 1) \vee (j = 1)\} \quad \forall\, i, j \in \{1, \ldots, n\}\,. \tag{6}$$

In this graph, all vertices are adjacent to a single central vertex. The vertices 2 and 3 are labeled as classes 1 and 2, respectively. Let the subgraph $W[e]$ be induced by 3 vertices $e = \{1, 2, 3\}$ (Figure 1d). Moreover, let the size of the graph n increase. Note that regardless of n, $\mathbf{f}_1^* = \mathbf{f}_1^e = (\frac{1}{2}, \frac{1}{2})$. So the distance between the two solutions $\|\mathbf{f}_1^e - \mathbf{f}_1^*\|_\infty$ is zero for all n. In other words, the HS on subgraphs can be highly accurate, even when the subgraph covers only a tiny portion of the entire graph W.

3.2 ε-subgraphs

Our examples illustrate that the HS on a subgraph can be both a good and bad approximation of that on the entire graph. The quality of the approximation depends on how the subgraph is constructed. In this work, we analyze a special family of subgraphs that provide guarantees on the quality of the approximation.

DEFINITION 2. *An* ε-subgraph $W[e]$ *induced by vertices* e *is a subgraph that has two additional properties. First, the set* e *can be divided into* core*:*

$$e^+ = \left\{i \in e : \|\mathbf{f}_i^e\|_\infty > \varepsilon\right\}$$

surface*:*

$$e^- = \left\{i \in e : \|\mathbf{f}_i^e\|_\infty \leq \varepsilon\right\}$$

vertices such that all neighbors of the core vertices e^+ *in* W *are in* e*. Second, all labeled vertices* l *are in* e*.*

Informally, we refer to the vertices whose most probable label is predicted with at least ε probability, $\|\mathbf{f}_i\|_\infty > \varepsilon$, and no more than ε probability, $\|\mathbf{f}_i\|_\infty \leq \varepsilon$, as *high* and *low confidence* predictions, respectively. The core e^+ and surface e^- vertices in the ε-subgraph are examples of high and low confidence predictions, respectively.

The ε-subgraph is a graph where high confidence predictions are separated from the vertices $u \setminus e$ that are not in the subgraph by the low confidence ones (Figure 2). Due to this structure, we can make two claims. First, if a vertex is not predicted with high confidence on the subgraph, it cannot be predicted with high confidence on the graph. Second, if a prediction on the subgraph is highly confident, it is also confident on the graph.

In the rest of this section, we prove these claims. First, we show that if a vertex is not a core vertex, it cannot be predicted with high confidence on W.

PROPOSITION 1. *Let* $i \in u \setminus e^+$*. Then* $\|\mathbf{f}_i^*\|_\infty \leq \varepsilon$*.*

Proof: Our proof is based on the random-walk interpretation of the HS. In particular, $f_i^*[k]$ is the probability that random walks on W that start in the vertex i are absorbed at vertices with labels k. Note that $W[e]$ is an ε-subgraph. Therefore, $l \subseteq e^+$ and all neighbors of e^+ are in e. So all random walks on W from $i \in u \setminus e^+$ must visit at least one surface vertex before being absorbed. Let j be the first such vertex after no vertex from $u \setminus e$ is visited. Then $f_i^*[k]$ can be rewritten as:

$$f_i^*[k] = \sum_{j \in e^-} \phi_{ij}^k f_j^e[k], \tag{7}$$

where ϕ_{ij}^k is the fraction of the aforementioned random walks that correspond to the vertex j and $f_j^e[k]$ is the absorption probability at this vertex. Note that $f_j^e[k]$ is bounded from above by ε. Therefore, it follows:

$$\|\mathbf{f}_i^*\|_\infty \leq \max_k |f_i^e[k]| \leq \max_k \Big| \underbrace{\sum_{j \in e^-} \phi_{ij}^k}_{1} f_j^e[k] \Big| \leq \varepsilon. \tag{8}$$

This concludes our proof. ■

Proposition 1 says that if a vertex is not a core vertex, it can only be predicted with low confidence on W. Another way of interpreting the result is that only the core vertices e^+ can be ever predicted with high confidence. In the following claim, we bound the difference between the harmonic solutions on W and its subgraph $W[e]$ on the core vertices e^+.

PROPOSITION 2. *Let* $i \in e^+$*. Then:*

$$\|\mathbf{f}_i^e - \mathbf{f}_i^*\|_\infty \leq \frac{1 - \|\mathbf{f}_i^e\|_\infty}{1 - \varepsilon}.$$

Proof: Our proof consists of two main steps. First, we bound from below the fraction of random walks that visit only the core vertices e^+. Since all neighbors of the vertices e^+ are in $W[e]$, these walks do not change between W and $W[e]$. Based on this fact, we prove an upper bound on the difference in the harmonic solutions.

Let p_{ik}^+ and p_{ik}^- be probabilities that random walks on $W[e]$ that start in the vertex i, and visit only the core vertices and at least one surface vertex, respectively, are absorbed at vertices with labels k. Then the probability that the vertex i has label k can be written as:

$$f_i^e[k] = \alpha p_{ik}^+ + (1 - \alpha) p_{ik}^-, \tag{9}$$

where α is the fraction of the walks that visit only the core vertices e^+. The fraction α can be bounded from below as:

$$\alpha = \frac{f_i^e[k] - p_{ik}^-}{p_{ik}^+ - p_{ik}^-} \geq \frac{f_i^e[k] - p_{ik}^-}{1 - p_{ik}^-} \geq \frac{f_i^e[k] - \varepsilon}{1 - \varepsilon}. \quad (10)$$

The last inequality is due to the fact that $\frac{f_i^e[k] - p_{ik}^-}{1 - p_{ik}^-}$ decreases when p_{ik}^- increases and $f_i^e[k] \leq 1$. We bound p_{ik}^- from above by ε. This upper bound follows from the observation that the probability p_{ik}^- can be rewritten as:

$$p_{ik}^- = \sum_{j \in e^-} \phi_{ij}^k f_j^e[k], \quad (11)$$

where ϕ_{ij}^k is the fraction of random walks from the vertex i where the first visited surface vertex is j and $f_j^e[k]$ is the absorption probability at the vertex j. Since $W[e]$ is an ε-subgraph, the probability $f_j^e[k]$ can be bounded from above by ε. So any convex combination of $f_j^e[k]$, such as p_{ik}^-, is bounded by ε.

The lower bound on the fraction α (Equation 10) holds for all k. So we bound α from below by the largest one with respect to k:

$$\alpha \geq \max_k \left\{ \frac{f_i^e[k] - \varepsilon}{1 - \varepsilon} \right\} = \frac{\|\mathbf{f}_i^e\|_\infty - \varepsilon}{1 - \varepsilon}. \quad (12)$$

Since the fraction of random walks that visit only the core vertices e^+ is bounded from below, and these walks are the same on W and $W[e]$, the difference in the harmonic solutions on W and $W[e]$ can be bounded from above as:

$$|f_i^e[k] - f_i^*[k]| \leq 1 - \frac{\|\mathbf{f}_i\|_\infty - \varepsilon}{1 - \varepsilon} = \frac{1 - \|\mathbf{f}_i\|_\infty}{1 - \varepsilon}. \quad (13)$$

This concludes our proof. ■

Proposition 2 says that highly-confident predictions on the subgraph $W[e]$ are good approximations of those on the graph W. For instance, when $f_i^e[k] = 0.9$ and $\varepsilon = 0.5$, the distance $\|\mathbf{f}_i^e - \mathbf{f}_i^*\|_\infty$ cannot be larger than 0.2, no matter how much W and $W[e]$ differ. Our upper bound on $\|\mathbf{f}_i^e - \mathbf{f}_i^*\|_\infty$ can be applied to Equation 4 and yields the following lower bound.

PROPOSITION 3. *Let $i \in e^+$. Then:*

$$f_i^*[k] \geq f_i^e[k] - \frac{1 - \|\mathbf{f}_i^e\|_\infty}{1 - \varepsilon}.$$

3.3 Normalized Laplacian

The Laplacian L in the harmonic solution is often substituted for the *normalized Laplacian* $L_n = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$. In this section, we show how to generalize our results to the normalized Laplacian L_n. The main difficulty is that the HS on L_n does not have a clear random-walk interpretation. To obtain it, we rewrite the HS as:

$$\begin{aligned} F_u &= (I - D_{uu}^{-\frac{1}{2}} W_{uu} D_{uu}^{-\frac{1}{2}})^{-1} D_{uu}^{-\frac{1}{2}} W_{ul} D_{ll}^{-\frac{1}{2}} F_l \\ &= (D_{uu}^{\frac{1}{2}} (I - D_{uu}^{-1} W_{uu}) D_{uu}^{-\frac{1}{2}})^{-1} D_{uu}^{-\frac{1}{2}} W_{ul} D_{ll}^{-\frac{1}{2}} F_l \\ &= D_{uu}^{\frac{1}{2}} (I - D_{uu}^{-1} W_{uu})^{-1} D_{uu}^{-1} W_{ul} D_{ll}^{-\frac{1}{2}} F_l \\ &= D_{uu}^{\frac{1}{2}} (I - P_{uu})^{-1} P_{ul} D_{ll}^{-\frac{1}{2}} F_l. \end{aligned} \quad (14)$$

Finally, we multiply both sides of Equation 14 by $D_{uu}^{-\frac{1}{2}}$ and get:

$$\begin{aligned} [D_{uu}^{-\frac{1}{2}} F_u] &= (I - P_{uu})^{-1} P_{ul} [D_{ll}^{-\frac{1}{2}} F_l] \\ &= (I + P_{uu} + P_{uu}^2 + \dots) P_{ul} [D_{ll}^{-\frac{1}{2}} F_l]. \end{aligned} \quad (15)$$

Algorithm 1 Incremental subgraph learning.

Input:
Graph W
Confidence level $\varepsilon \in [0, 1]$

$e^{(1)} \leftarrow l \cup \{i \in u \; \exists j \in l : w_{ij} > 0\}$
$t \leftarrow 1$
repeat
  $L \leftarrow n \times n$ Laplacian of $W[e^{(t)}]$
  infer $F_u \leftarrow -(L_{uu})^{-1} L_{ul} F_l$
  $e^+ \leftarrow \left\{ i \in e^{(t)} : \|\mathbf{f}_i\|_\infty > \varepsilon \right\}$
  $e^{(t+1)} \leftarrow e^{(t)} \cup \left\{ i \in u : \exists j \in e^+ \; (w_{ij} > 0) \right\}$
  $t \leftarrow t + 1$
until $((|e^{(t)}| = |e^{(t-1)}|) \vee (|e^{(t)}| = n))$

Output:
ε-subgraph $W[e^{(t)}]$

Equation 15 has the same form as Equation 2. As a result, we may conclude that the HS on the normalized Laplacian L_n has indeed a random walk interpretation, where the HS F_u is additionally scaled by $D_{uu}^{-\frac{1}{2}}$.

Let the self-similarity w_{ii} of all vertices i be 1. Then $d_i \geq 1$ for all vertices i. Moreover, note that $||\mathbf{f}_i||_1 = 1$ at all labeled vertices because $\mathbf{f}_i$ is a distribution over labels. As a result, $||d_i^{-\frac{1}{2}} \mathbf{f}_i||_1 \leq 1$ at these vertices. From Equation 15, it follows that $||d_i^{-\frac{1}{2}} \mathbf{f}_i||_1 \leq 1$ for all i. So $d_i^{-\frac{1}{2}} \mathbf{f}_i$ can be loosely interpreted as a distribution.

Based on our discussion, the HS on the normalized Laplacian L_n has a random-walk interpretation (Equation 15), and $||d_i^{-\frac{1}{2}} \mathbf{f}_i||_\infty \leq ||d_i^{-\frac{1}{2}} \mathbf{f}_i||_1 \leq 1$ at all vertices i. Therefore, we can follow the same reasoning as in Section 3.2 and generalize our results as follows.

PROPOSITION 4. *Let F^* and F^e be harmonic solutions on the normalized Laplacians of W and its ε-subgraph $W[e]$. Let the* core *and* surface *vertices be defined as:*

$$e_n^+ = \left\{ i \in e : d_i^{-\frac{1}{2}} \|\mathbf{f}_i^e\|_\infty > \varepsilon \right\}$$

and

$$e_n^- = \left\{ i \in e : d_i^{-\frac{1}{2}} \|\mathbf{f}_i^e\|_\infty \leq \varepsilon \right\},$$

respectively. Then:

$$d_i^{-\frac{1}{2}} \|\mathbf{f}_i^*\|_\infty \leq \varepsilon$$

for all $i \in u \setminus e_n^+$. Moreover:

$$d_i^{-\frac{1}{2}} \|\mathbf{f}_i^e - \mathbf{f}_i^*\|_\infty \leq \frac{1 - d_i^{-\frac{1}{2}} \|\mathbf{f}_i^e\|_\infty}{1 - \varepsilon}$$

for all $i \in e_n^+$.

3.4 Algorithm

In Section 3.2, we showed how to bound the difference between the harmonic solution on the graph W and its ε-subgraph $W[e]$. In this section, we propose an algorithm for building ε-subgraphs.

The algorithm proceeds in iterations. First, the subgraph vertices $e^{(t)}$ are initialized to labeled vertices and their neighbors. Second, we compute the harmonic solution on the subgraph $W[e^{(t)}]$. Third, we find vertices e^{+} whose labels are inferred with sufficiently high confidence and add their neighbors to the subgraph vertices $e^{(t+1)}$. Finally, we compute the harmonic solution on the graph $W[e^{(t+1)}]$ and repeat all steps until the set $e^{(t)}$ stops growing. Our method is outlined in Algorithm 1.

Algorithm 1 can terminate for one of two reasons. First, $|e^{(t)}| = |e^{(t-1)}|$. In this case, all neighbors of highly confident predictions e^{+} are in $e^{(t)}$, and so $W[e^{(t)}]$ is an ε-subgraph. Second, $|e^{(t)}| = n$. In this case, $W[e^{(t)}]$ is the entire graph, which is an ε-subgraph for all ε. So Algorithm 1 always outputs an ε-subgraph. Note that the set $e^{(t)}$ increases monotonically, $e^{(t)} \subseteq e^{(t+1)}$. As a consequence, Algorithm 1 always terminates, for one of the above reasons.

The *confidence level* ε is the only parameter of Algorithm 1. The parameter ε controls the size of the resulting ε-subgraph $W[e]$. The smaller the value of ε, the larger the subgraph. We study this trend in Section 4.3.

Let $W[e^{(T)}]$ be the ε-subgraph generated by Algorithm 1. Then the space complexity of Algorithm 1 is $O(|e^{(T)}|^2)$, the space taken by $W[e^{(T)}]$. The time complexity is $O(T|e^{(T)}|^3 + n|e^{(T)}|)$, where T is the number of iterations, $O(|e^{(T)}|^3)$ is the cost of computing the HS in each iteration, and $n|e^{(T)}|$ is an upper bound on the total number of tests for neighbors of the vertices e^{+} when the set $e^{(t+1)}$ is generated. As a result, when $|e^{(T)}| \ll n$, the time complexity of our method is linear in n.

The above analysis makes no assumptions on $W[e^{(T)}]$. Suppose that $W[e^{(t)}]$ is $O(|e^{(t)}|)$ sparse for all t. Then the space and time complexity of our method are only $O(|e^{(T)}|)$ and $O(T|e^{(T)}|^2)$, respectively. Sparsity is common in practice.

3.5 Practical issues

The worst-case number of iterations in Algorithm 1 can be large. Consider the line graph example in Section 3.1 (Figure 1a). In this example, the labeled vertices are $n-1$ hops apart, and our method converges in $O(n)$ iterations irrespective of the confidence level ε, when the ε-subgraph covers the entire graph W.

Obviously, this behavior is undesirable. To avoid such cases, we suggest regularizing the harmonic solution as [8]:

$$F_u = -(L_{uu} + \gamma I_u)^{-1} L_{ul} F_l, \tag{16}$$

where I_u is a $n_u \times n_u$ identity matrix and γ is a tunable parameter. The regularizer γI_u can be viewed as a special *sink* vertex. At each step, the random walk on the graph W is absorbed in the sink with probability $\frac{\gamma}{d_i+\gamma}$. So the confidence $f_i[k]$ of labels decreases with the distance from labeled vertices l. In particular, note that:

$$f_i[k] \leq \left(1 - \frac{\gamma}{d_i + \gamma}\right)^{\tau_i}, \tag{17}$$

where τ_i is the number of hops between the vertex i and the closest labeled example. The above claim holds for any graph.

The parameter γ can be used to control the number of vertices in the subgraph. In particular, note that Algorithm 1 can add a vertex only if its neighbor is among the vertices e^{+}, which are defined as $e^{+} = \left\{ i \in e^{(t)} : \|\mathbf{f}_i\|_\infty > \varepsilon \right\}$. When:

$$\gamma = \left(\exp\left[-\frac{\log \varepsilon}{\kappa}\right] - 1\right) \max_i d_i, \tag{18}$$

$\|\mathbf{f}_i\|_\infty \leq \varepsilon$ for all vertices that are at least κ hops from the closest labeled vertex. This can be easily verified by substituting the above

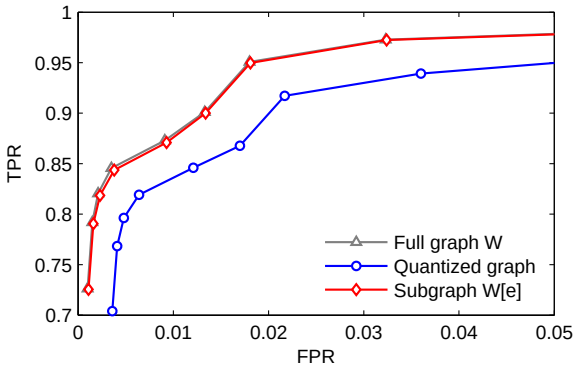

Figure 3: The TPR and FPR of three harmonic solutions on the digit recognition dataset.

γ to Equation 17. As a result, Algorithm 1 never adds a vertex that is more than k hops from the closest labeled vertex. In Section 4.3, we study how the performance of Algorithm 1 changes with γ.

Finally, note that Algorithm 1 requires the graph W to be sparse. Therefore, we consider similarity functions of the form:

$$w_{ij} = \begin{cases} \exp\left[-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|_2^2}{2\sigma^2}\right] & \|\mathbf{x}_i - \mathbf{x}_j\|_2^2 \leq \phi \\ 0 & \text{otherwise,} \end{cases} \tag{19}$$

where the parameter ϕ controls the sparsity of W.

4. DIGIT RECOGNITION

We now evaluate our approach on the digit recognition problem. The dataset and experimental setup are described in detail in Section 4.1. The digit recognition dataset is small and therefore we can build the entire data adjacency graph W. In addition, all data points are labeled. Therefore, we can easily evaluate the accuracy of our approach and compare it to baselines (Section 4.2). We also study the sensitivity of our method to the setting of its parameters (Section 4.3).

4.1 Experimental setup

The performance of our solution is evaluated on the problem of handwritten digit recognition [7]. The digit recognition dataset comprises of 5620 examples, digits between 0 and 9. Each digit is described by 64 features, which are discretized on an 8×8 grid.

The data adjacency graph W is constructed as described in Section 3.5. The heat parameter and sparsity threshold are set as $\sigma = 0.1\sqrt{K}$ and $\phi = 2\sigma$, where $K = 64$ is the number of features. In summary, the similarity of examples decreases exponentially with distance and is zero when the examples are more distant than 2σ. This results in a sparse graph W.

We cast digit recognition as a set of binary classification problems, one for each pair of the digits. In each problem, we label 10 randomly selected examples for each digit in the pair. The labels of other examples are inferred on the ε-subgraph $W[e]$. We regularize the Laplacian as described in Section 3.5 and set the regularization parameter γ to 0.1. Our results are averaged over all classification problems.

4.2 Comparison to baselines

The digit recognition dataset is small and therefore we can build the data adjacency graph W on this dataset. The accuracy of predictions on the ε-subgraph is expected to be suboptimal when com-

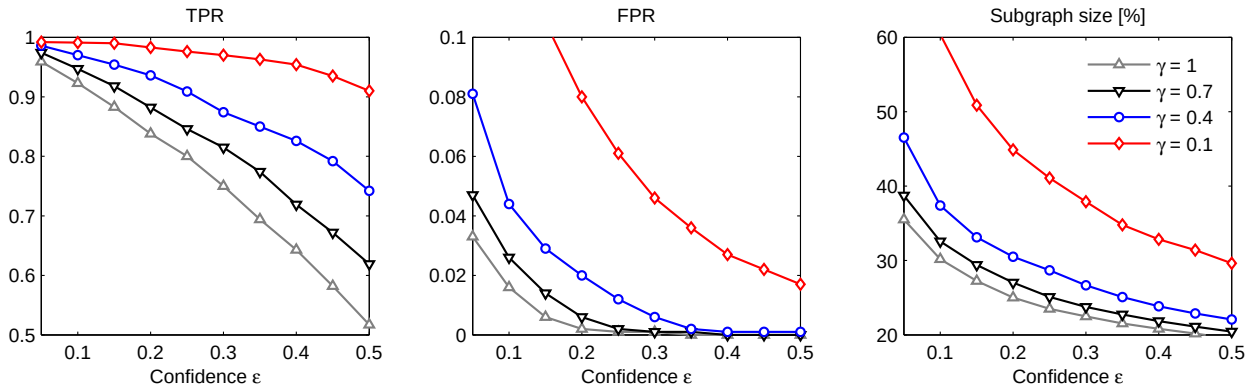

Figure 4: From left to right, we report the TPR and FPR of the HS on ε-subgraphs, and the size of the subgraphs $100\frac{|e|}{n}$, as functions of the confidence threshold ε. The subgraphs are built for four different values of γ.

pared to those on W. Perhaps surprisingly, we show that our predictions are nearly optimal. We also compare our solution to graph quantization (Section 2.1). In graph quantization, the graph W is approximated by a smaller graph on k representative examples. In our experiments, $k = |e|$ and therefore the quantized graph has the same size as the corresponding ε-subgraph. In other words, the HS on the quantized graph is of the same time and space complexity as the final iteration of Algorithm 1.

In Figure 3, we compare the TPR and FPR of the harmonic solutions on ε-subgraphs, the entire graph W, and the quantized graph. In ε-subgraphs, we only predict the labels of the core vertices e^+. To make a fair comparison, the two baselines also predict only top $\left|e^+\right|$ most confident labels according to the HS. We vary the confidence level ε to get subgraphs of various sizes, and get a point on the ROC curve for each ε.

Figure 3 shows that our method makes almost as accurate predictions as the HS on the graph W. In fact, our results are so good that the corresponding ROC curves are hard to distinguish. Moreover, note that our method performs significantly better than graph quantization. This shows that smart allocation of vertices, by building ε-subgraphs, leads to better results than covering the graph by randomly chosen representative examples.

4.3 Sensitivity to parameter settings

In the second experiment, we study the sensitivity of the HS on the ε-subgraph to its two parameters, the confidence level ε (Section 3.4) and the regularization parameter γ (Section 3.5). We plot ROC curves for several values of γ and vary ε to get points on each curve. Our results are summarized in Figure 4.

In most cases, the TPR and FPR of the harmonic solution on ε-subgraphs are pretty high and relatively low, respectively. When ε is close to zero, the subgraphs cover a large portion of the graph and the TPR is at the maximum. As ε increases, the FPR and TPR decrease. Finally, when $\varepsilon = 0.5$, all subgraphs cover less than 30% of the entire graph and we make only confident predictions. Note that the minimal subgraph to predict correctly all instances of the two digits in each problem is about 20% of W.

The regularization parameter γ controls the size of ε-subgraphs (Section 3.5). As γ increases, fewer vertices are added to the subgraphs since the confidence of predictions decreases faster with the number of hops from labeled vertices. Therefore, the TPR and FPR decrease. Finally, note that ε-subgraph learning changes smoothly with γ and ε, and it is not sensitive to small perturbations of these parameters.

5. TOPIC DISCOVERY

In this section, we tackle a large real-world problem of semantic inference or topic discovery over graphs generated from free-form text. We analyze two large datasets of restaurant and hotel reviews, and infer multi-class membership probabilities over nouns and contextual descriptors in the corpora. It is computationally expensive to build a full graph over these datasets. Hence, our method is very useful in finding a relevant subgraph for the task at hand. In Section 5.1, we define the construction of the graph W that is approximated by our method. In Section 5.2, we introduce the domains of restaurant and hotel reviews, and describe our datasets. Our results are presented in Section 5.3 and we compare these with baseline methods in Section 5.4. Finally, we study the computational complexity of our algorithm in Section 5.5.

5.1 Data adjacency graph

We build a semantically coherent graph W over the text by utilizing the contextual descriptors around the words in the text. While semantically dissimilar words are often used in the same sentence, the descriptive context around the words creates a strong topical link. For instance, in our restaurant reviews the words "food" and "service" which belong to obviously different restaurant topics co-occur almost ten times as often as the words "food" and "chicken". However, we never expect to see the phrase "service is delicious", and we could use the contextual descriptor "___ is delicious" to link words under the food topic.

We build the subgraph over the textual data using Algorithm 1. Our graph comprises of two types of vertices - words and descriptors. The contextual descriptors consist of 1 to 5 words appearing before, after, or both before and after the words in review sentences. A five word window is sufficient to capture most commonly used phrases. There are millions of context descriptors and we consider only those which occur at least 20 times. In addition, we prune the list of descriptors to remove those with only stop words; a descriptor like "the ___" is not very informative. Secondly, our graph comprises of words that fit the descriptors. We restrict this step to finding nouns as in a sentence as the semantic meaning is often carried in the nouns. Therefore, we build a bipartite data adjacency graph W where the two parts correspond to words and their contextual descriptors.

The words and descriptors that co-occur in a sentence are linked by edges weighted by the point-wise mutual information (PMI) score [14, 4]. The edge weight between word i and context de-

	Restaurants	Hotels
Reviews	37224	137234
Sentences	344217	971739
Businesses	2122	3370
Users	18743	No unique user identifiers available
Distinct Nouns	8482	11212
Distinct Words	12080	19045

Table 1: Description of two large reviews datasets.

scriptor j is:

$$w_{ij} = \log\left(\max\left\{\frac{P(i \cap j)}{P(i)P(j)} - \phi, 1\right\}\right), \qquad (20)$$

where $P(i \cap j)$ is the probability that word i and context j appear together, $P(i)$ is the probability of the word i, and $P(j)$ is the probability of the context j. The value of $P(i)$ is estimated directly from data. Instead of computing $P(i \cap j)$ and $P(j)$, which would require normalization over all contexts, we estimate the ratio $\frac{P(i \cap j)}{P(j)}$ as the fraction of contexts j with the word i. The threshold ϕ controls the sparsity of the graph.

5.2 Datasets

We obtained two large user reviews datasets from popular online reviewing websites: the restaurant reviews dataset was mined from Yelp (http://www.yelp.com) and the hotel reviews dataset was mined from TripAdvisor (http://www.tripadvisor.com). Both these datasets have very different properties as described below and summarized in Table 1. Yet, our methods are easily applicable to these large and diverse datasets and our algorithm finds very precise semantic clusters as shown in Section 5.3.

The restaurant reviews corpus has 37k reviews with an average length of 9.2 sentences. The 344k sentences were used to compute the PMI for constructing the graph. The vocabulary in the restaurant reviews corpus is very diverse and contains several proper nouns like menu items and server names. We used the openNLP toolkit (http://opennlp.sourceforge.net/) for sentence delimiting and part-of-speech tagging to detect the nouns in the data. We ignore infrequently occurring misspellings and idiosyncratic word formulations, and retain the nouns that occur at least 10 times in the corpus. The restaurant reviews dataset contains 8482 distinct nouns. We defined five semantic categories over the text: food, price, service, ambience, social intent (describing the purpose of the visit). We used only a handful of labeled seed words for each class. The choice of the seed words was based on the frequencies of these words in the corpus as well as their generally applicable meaning to a broad set of words. The seed words used for inference are shown in the top portion of Table 2.

The hotel reviews dataset is much larger with 137k reviews on hotels in Europe, as shown in Table 1. Yet, the average number of sentences in a review is only 7.1 sentences and despite four times as many reviews as the restaurants corpus, the number of distinct nouns is only 11k. In the hotel reviews dataset, reviewers rate six different aspects of the hotel: Cleanliness, Spaciousness, Service, Location, Value and Sleep Quality. Assuming that these six semantic classes are important to users, we adhered to the same in our experiments. The labeled seed words used for each class are shown in Table 4.

Labeled seed words				
food	**price**	**service**	**ambience**	**social intent**
food	price	service	ambience	boyfriend
dessert	cost	staff	ambiance	date
appetizer	costs	waiter	atmosphere	birthday
appetizers	value	waiters	decor	lunch
Top-10 discovered words by ε-subgraph inference				
food	**price**	**service**	**ambience**	**social intent**
starters	pricier	illusionist	downside	bday
apps	steamed	bruno	setting	graduation
howard	pricing	swan	general	lady
starter	tho	particular	presentation	husband
flatbreads	diego	server	sink	goodbye
error	source	proprietor	comstock	wife
don	crap	servers	vibe	farewell
sides	theres	waitress	impression	bachelorette
app	attitude	banter	uses	mom
desert	interpretation	tenders	octopus	sally

Table 2: Labeled seed words and top 10 words discovered by the HS on the ε-subgraph in the restaurant reviews domain.

5.3 ε-subgraph inference evaluation

We computed the HS on the ε-subgraph on the restaurant reviews dataset, and learned class labels for words and descriptors. The algorithm is parameterized as $\gamma = 1$, $\varepsilon = 0.3$, and $\phi = 128$. These parameters were chosen such that we explore only a small portion of the graph. Our algorithm quickly converges in 7 iterations and finds semantic confidence scores over 11% nouns. Table 2 shows the top 10 words with the highest class membership probability returned by the HS on the ε-subgraph. We observe that our method assigns high confidence scores to several synonyms of the seed words like *server, waitress, proprietor* for the service class and our method also captures common abbreviations and domain specific usage of words like *apps* and *starters* for the food class.

We do not have ground truth on the semantic meaning on words. Therefore, we manually evaluated the lists of top-K words with the highest confidence scores for belonging to each semantic group. We evaluated the performance of ε-subgraph inference using the precision@K metric for each semantic class. A high precision value indicates that a large number of the top-K words returned by the algorithm are labeled with the correct semantic class. Three judges evaluated the quality of the word classification (with inter-annotator kappa = 76%) and we used the majority vote for assessing the results. Table 3 shows the precision of the returned results for the five semantic classes at $K = \{5, 10, 20\}$. We see that at $K = 10$, we have a very high precision of over 90% for service and social intent and over 60% for food and ambiance. Our performance on the price class is poor because users rarely write about the price and rely on the metadata price classification of the restaurant.

Using the same parameter settings on the hotel reviews dataset, we run 5 iterations and explore a subgraph over 20% nouns and 14k descriptors, again only a small fraction of the complete graph over the text. Table 4 shows the top 10 words with the highest class membership probability returned by our ε-subgraph inference algorithm. Next, we evaluate the labels learned for the six semantic categories in our corpus. The precision@K for the ε-subgraph inference on the hotel reviews dataset is shown in Table 5. We have a high precision (above 70%) for all categories in this dataset with perfect precision for the service class. These results are slightly better than the ones on the restaurant reviews dataset. We believe that the improvement in precision is due to the better defined and

Precision @	Semantic Class	ε-Subgraph Inference	Quantized Subgraph	Self Training
5	**food**	**0.8**	0.6	0.2
	price	0.4	0.4	**0.6**
	service	**0.8**	**0.8**	0.6
	ambience	**0.8**	**0.8**	0.6
	social intent	**1**	**1**	0.8
	Average	**0.76**	0.72	0.56
10	**food**	**0.7**	0.6	0.5
	price	0.2	0.4	**0.5**
	service	**0.9**	0.6	0.5
	ambience	0.6	0.6	**0.7**
	social intent	**1**	0.8	0.8
	Average	**0.68**	0.6	0.6
20	**food**	0.65	**0.75**	0.3
	price	**0.35**	0.3	**0.35**
	service	**0.9**	0.55	0.6
	ambience	0.55	0.55	**0.8**
	social intent	**1**	0.55	0.75
	Average	**0.69**	0.54	0.56

Table 3: Precision at top K semantic labels learned in the restaurant reviews domain.

Labeled seed words					
cleanliness	**service**	**spaciousness**	**location**	**value**	**sleep quality**
cleanliness	service	size	location	price	sleep
dirt	staff	closet	area	cost	bed
mould	receptionist	bathroom	place	amount	sheet
smell	personel	space	neighborhood	rate	noise
Top-10 discovered words by ε-subgraph inference					
cleanliness	**service**	**spaciousness**	**location**	**value**	**sleep quality**
accomodation	folks	sup	neighbourhood	package	noises
accommodation	clerks	wardrobe	someplace	airfare	pram
oder	attendants	vale	neighborood	deal	mattress
mold	workers	storage	schillerstrasse	tariff	sleeping
wonder	receptionists	fringe	recomend	sum	crash
odor	gals	warning	palce	alot	coins
mildew	personel	cupboard	hell	vs	jams
show	staffers	drawer	neigbourhood	lot	noice
hygiene	julie	counter	cul	charge	terror
stains	benedetta	shelf	intending	evaluation	pensionato

Table 4: Labeled seed words and top 10 words discovered by the HS on the ε-subgraph in the hotel reviews domain.

distinct classes in the hotels domain derived directly from TripAdvisor.

5.4 Comparison to baselines

We now compare the precision of our method with two baseline methods for semantic class labeling. First, we build a quantized subgraph (Section 2.1) by a random selection of vertices. We build the similarity graph using Equation 20, and compute the HS on this subgraph. In essence, inference on this quantized graph differs from our method only in the selection of the vertices to build the subgraph. For a fair comparison with our ε-subgraph inference method, we used the same number of noun and descriptor vertices as the subgraph found by our technique. Table 3 shows the precision@K for this quantized subgraph on the restaurant reviews dataset. As expected, we see an overall lower precision for the labels learned over the quantized subgraph in comparison with our method. At $K = 20$, while ε-subgraph inference generates high precision labels (> 0.9) for the service and social intent classes, quantization generates significantly lower precision (0.55).

As a second baseline, we adapt the self-training method from [10] called Espresso. As described in Section 2.2, the principle idea is that at each iteration Espresso finds new vertices in the data graph and deterministically assigns class labels to a few. The reliability metric described in [11] is used by Espresso to label vertices. This greedy method differs from our algorithm only in that it makes hard class decisions based on the reliability metric and there is no random walk inference computation. The construction of the similarity graph is identical to our implementation. As shown in Table 3, the Espresso self training algorithm provides less accurate class labels on the restaurant reviews dataset. Moreover, the computational complexity of such an algorithm is very high. Across all semantic classes, this self-training algorithm explored as many as 25% and 43% nouns in the restaurant and hotel reviews datasets only in the fourth iteration. Eventually, Espresso evaluated over 94% nouns. Hence, self-training methods achieve low precision and low efficiency in comparison to our ε-subgraph inference method. For all $K = 5, 10, 20$, the average precision across all five semantic classes is highest when using our ε-subgraph inference. At $K = 20$, we see a 28% and 23% improvement averaged across all classes by our method over quantization and self training respectively.

We now compare the inference of the alternate baseline methods on the hotel reviews dataset. The performance of the HS on the ε-subgraph is significantly better than the baselines of quantization and the self-training. Averaging across the six semantic classes at $K = 20$, we see a large improvement of 29% and 45% of our method over quantization and self training respectively. In the future, we wish to evalaute these alternate approaches with a much larger set of hand-labelled assessments.

5.5 Computational complexity

We now assess the gain in performance by using our method. We present results in the restaurant reviews domain; the hotels domain had similar gains. On the restaurant reviews dataset our algorithm runs for 7 iterations and finds semantic confidence scores over 11% nouns in the corpus. To evaluate the reduction in computational cost, we generated all context descriptors in the corpus using the neighborhood window around all occurrences of words and found that our corpus contains 41k frequent descriptors (occurring at least 20 times). In comparison, our algorithm generates a subgraph comprising less than 5k descriptors. Thus, we explore only a small fraction of the complete graph that is most semantically meaningful.

The computation time of the topic discovery experiments is dominated by queries to the database that store the sentences from the reviews. At every iteration, we expand highly confident vertices e^+. This involves retrieving sentences from the database containing the vertices and finding new neighboring vertices in the text. The set e^+ comprises of only 0.7% nodes in the restaurant reviews domain, resulting in significant improvement in computation time. In addition, the ε-subgraph $W[e^{(T)}]$ comprises of only 11% nodes and the space required $O(|e^{(T)}|^2)$ is two orders of magnitude smaller than the full graph built on all vertices (Section 3.4). However, our algorithm requires finding the HS for each iteration. Yet, our experiments require very few iterations in general, 7 and 5 iterations over the restaurant and hotel domains respectively. Therefore, the dominating time is the database queries for finding neighbors of vertices and generating the adjacency matrix. Overall our algorithm achieves significant performance gain over a one-shot inference on

Precision @	Semantic Class	ε-Subgraph Inference	Quantized Subgraph	Self Training
5	**cleanliness**	**0.8**	0.6	**0.8**
	service	**1**	**1**	0.8
	spaciousness	**0.8**	0.6	0.2
	location	0.8	**1**	0.6
	value	**1**	0.8	0.6
	sleep quality	**1**	0	0.6
	Average	**0.9**	0.8	0.6
10	**cleanliness**	**0.8**	0.7	0.7
	service	**1**	**1**	0.7
	spaciousness	**0.8**	**0.8**	0.5
	location	**0.8**	**0.8**	0.5
	value	**0.9**	0.6	0.6
	sleep quality	0.7	**0.9**	0.4
	Average	**0.83**	0.8	0.57
20	**cleanliness**	**0.8**	0.65	0.7
	service	**1**	0.75	0.75
	spaciousness	**0.7**	0.65	0.5
	location	**0.8**	0.6	0.65
	value	**0.85**	0.55	0.5
	sleep quality	**0.7**	0.6	0.25
	Average	**0.81**	0.63	0.56

Table 5: Precision at top K semantic labels learned in the hotel reviews domain.

the full graph.

6. CONCLUSIONS

The harmonic solution on a graph is a popular approach to semi-supervised learning. Unfortunately, the method does not scale well with the size of training data n because its space and time complexity are $\theta(n^2)$ and $\theta(n^3)$, respectively. In this paper, we study a new approach to approximating the harmonic solution. In particular, we show how highly confident HS predictions on a graph can be identified based on a subgraph. The premise of our technique is that the subgraph is much smaller than the graph and hence we can identify highly confident predictions at a lower cost than computing the HS on the entire graph. We identify a class of subgraphs that allow for good approximations, prove bounds on the quality of these approximations, and propose an algorithm for building the subgraphs. Our method is evaluated on several problems, two of which are large-scale, and we demonstrate its utility and scalability. In our future work, we plan to advance both the theoretical and practical aspects of our method.

For instance, we derived an upper bound on the difference in the HS on the graph W and its subgraph $W[e]$. However, it is not clear what is the minimum number of vertices that are necessary to build an ε-subgraph for a set of labeled vertices l and confidence level ε. We would like to prove that the ε-subgraph that is built greedily is not much larger than the minimal subgraph for a given ε.

Our current solution does not scale to subgraphs with more than a few thousand vertices. In future, we plan to quantize our approximations and learn subgraphs whose space complexity is bounded.

7. REFERENCES

[1] E. Agichtein and L. Gravano. Snowball: Extracting relations from large plain-text collections. In *Proceedings of the fifth ACM conference on Digital libraries*, pages 85–94, 2000.

[2] M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7:2399–2434, 2006.

[3] S. Brin. Extracting patterns and relations from the world wide web. In *WebDB Workshop at 6th International Conference on Extending Database Technology, EDBTï£¡98*, pages 172–183, 1998.

[4] K. W. Church and P. Hanks. Word association norms, mutual information, and lexicography. *Computational Linguistics*, 16:22–29, 1990.

[5] O. Etzioni, M. Cafarella, D. Downey, S. Kok, A.-M. Popescu, T. Shaked, S. Soderland, D. S. Weld, and A. Yates. Web-scale information extraction in knowitall: (preliminary results). In *WWW*, pages 100–110, 2004.

[6] R. Fergus, Y. Weiss, and A. Torralba. Semi-supervised learning in gigantic image collections. In *Advances in Neural Information Processing Systems 22*, pages 522–530, 2009.

[7] M. D. Garris, J. L. Blue, G. T. Candela, G. T. C, D. L. Dimmick, J. Geist, P. J. Grother, S. A. Janet, and C. L. Wilson. Nist form-based handprint recognition system. 1994.

[8] B. Kveton, M. Valko, A. Rahimi, and L. Huang. Semi-supervised learning with max-margin graph cuts. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, pages 421–428, 2010.

[9] G. Lever, T. Diethe, and J. Shawe-Taylor. Data dependent kernels in nearly-linear time. In *Proceedings of the 15th International Conference on Artificial Intelligence and Statistics*, pages 685–693, 2012.

[10] P. Pantel, E. Crestan, A. Borkovsky, A.-M. Popescu, and V. Vyas. Web-scale distributional similarity and entity set expansion. In *EMNLP*, pages 938–947, 2009.

[11] P. Pantel and M. Pennacchiotti. Espresso: leveraging generic patterns for automatically harvesting semantic relations. In *ACL*, pages 113–120, 2006.

[12] D. Spielman and S.-H. Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing*, pages 81–90, 2004.

[13] A. Talwalkar, S. Kumar, and H. Rowley. Large-scale manifold learning. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2008.

[14] P. Turney. Mining the web for synonyms: PMI-IR versus LSA on TOEFL, 2001.

[15] M. Valko, B. Kveton, L. Huang, and D. Ting. Online semi-supervised learning on quantized graphs. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence*, 2010.

[16] D. Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pages 189–196, 1995.

[17] X. Zhu. Semi-supervised learning literature survey. Technical Report 1530, University of Wisconsin-Madison, 2008.

[18] X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using Gaussian fields and harmonic functions. In *Proceedings of the 20th International Conference on Machine Learning*, pages 912–919, 2003.