

*Department of Computer & Information Science*

*Technical Reports (CIS)*

---

*University of Pennsylvania*

*Year 2009*

---

Online Learning a Binary Labeling of a  
Graph

Mickey Brautbar  
University of Pennsylvania

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-09-02

This paper is posted at ScholarlyCommons.  
[http://repository.upenn.edu/cis\\_reports/896](http://repository.upenn.edu/cis_reports/896)

---

# Online Learning a Binary Labeling of a Graph

---

Mickey brautbar

BRAUTBAR@SEAS.UPENN.EDU

Department of Computer and Information Science, University of Pennsylvania, PA 19104 USA

**Keywords:** graph labeling, prediction on graphs, online learning

## Abstract

We investigate the problem of online learning a binary labeling of the vertices for a given graph. We design an algorithm, *Majority*, to solve the problem and show its optimality on clique graphs. For general graphs we derive a relevant mistake bound that relates the algorithm's performance to the cut size (the number of edges between vertices with opposite labeling) and the maximum independent set in the graph. We next introduce a novel complexity measure of the true labeling - the *frontier* and relate the number of mistakes incurred by *Majority* to this measure. This allows us to show, in contrast to previous known approaches, that our algorithm works well even when the cut size is bigger than the number of vertices. A detailed comparison with previous results is given.

## 1. Introduction

Consider the following natural scenario: we are given a graph that connects a set of entities. Edges in the graph correspond to direct influence/relationship between entities. The graph is known and is given to us in advance. Each entity holds a private binary value, its labeling. In each round we are presented with a previously unlabeled vertex in the network and are asked to predict its labeling; if the labeling of the vertices in the graph are chosen completely at random then we will inevitably be mistaken at least half the time. However, if there is some structure to the labeling of the vertices, then we would like to be able to predict well. Apart from the field of Machine Learning, the online labeling problem naturally arises in several related fields. (Herbster & Pontil, 2006) describe an in-

triguing example, showing the importance of the graph labeling problem in the context of Web Advertisement Systems. In the field of Social Networks, an important task is to predict whether a person will vote for a certain candidate, given the voting results of his friends. This can be done using a graph labeling algorithm.

We introduce the *Majority* algorithm in order to efficiently solve the labeling problem. We then derive a mistake bound that relates the algorithm's performance to the cut-size measure. In contrast to previous results (Pelckmans & Suykens, 2008; Herbster et al., 2008; Herbster & Pontil, 2006), our algorithm is proven to perform well even when the cut size (the number of edges with opposite labeling) is bigger than the number of vertices. In order to show this we define a novel complexity measure, the *frontier* measure, and bind the performance of our algorithm to it.

*Majority* algorithm is run-time efficient, as it runs in linear time in the number of vertices and edges. This stands in contrast to graph Laplacian based methods which run in cubic time in the number of vertices. A detailed comparison to previous results is given.

## 2. The 'Majority' algorithm

We are given an undirected, non weighted graph  $G = (V, E)$ . A natural strategy for predicting the label of a new vertex is to use the majority vote of its neighbors which were labeled so far.

Some notation is in place. Denote by  $m_{majority}$  the number of mistakes incurred by *Majority* on a given prediction sequence. Denote the size of the largest independent set in  $G$  by  $\alpha$ . Define the cut size induced by  $y^*$  (number of edges with opposite labeling) to be:

$$\text{CutSize}(y^*) = |\{e = (u, v) : y^*(u) = '+', y^*(v) = '-'\}|$$

**Theorem 2.1 (upper bound)** *Assume the given graph  $G$  is a clique. Let  $s$  be a sequence of vertices, and let  $n_1$  be the number of vertices labeled with one*

**Algorithm 1** Majority

**Input:** a previously unlabeled vertex  $v$

- 1: Let  $S \subseteq V$  be the set containing all neighbors of  $v$  that had been labeled in previous iterations.
- 2: **if**  $|S| = 0$  **then**
- 3:   predict '+'
- 4: **else**
- 5:   predict  $\hat{y}(v)$  to be the **most** popular label type between the vertices in  $S$ , according to  $y^*$ , and '+' in a case of tie.
- 6: **end if**

labeling type by  $y^*$ , where  $n_1 \leq n - n_1$ . Then,

$$m_{Majority} \leq 2n_1 + 1$$

**Proof:**

Assume for contradiction that  $m_{Majority} \geq 2n_1 + 2$ . Assume w.l.o.g. that there are less vertices labeled as '+' than '-' by  $y^*$ . Denote the number of vertices labeled as '+' by  $y^*$  by  $n_1$ . Notice that there are two types of mistakes. The first type, which we call type I is when *Majority* predicts '-' but the real labeling is '+'. The second type of mistakes, type II, is when *Majority* predicts '+' but the real labeling is '-'. First we observe that the number of type I mistakes is at most  $n_1$ , since otherwise we have more than  $n_1$  vertices labeled as '+' by  $y^*$ . Therefore, the number of type II mistakes is at least  $n_1 + 2$ . Take the last vertex  $v$  in the prediction sequence that the algorithm incurred a type II mistake on. When the algorithm was requested to label  $v$  he had already made a mistake labeling at least  $n_1 + 1$  vertices that were labeled as '-' by  $y^*$ . Moreover, since *Majority* predicted  $v$  to have a '+' label we must have seen up to that moment at least as many vertices labeled as '+' as vertices labeled as '-'. From this fact and the previous one, we must have seen at least  $n_1 + 1$  vertices labeled as '+' by  $y^*$  immediately before labeling  $v$ , a contradiction.

Interestingly, this bound is optimal under a mild assumption on the size of  $n$ , even if we may device an algorithm that receive, in advance, the size of  $n_1$ .

**Theorem 2.2 (lower bound)** *Assume the given graph  $G$  is a clique. Fix the size of  $n_1$ . Let  $A$  be any deterministic algorithm for the graph labeling problem. Then, assuming that  $n \geq 4n_1 + 1$ , one may construct a sequence of vertices and a labeling for it  $y^*$ , where  $n_1$  is the number of vertices labeled with the less frequent label type by  $y^*$ , such that,*

$$m_A \geq 2n_1 + 1$$

**Proof:**

Nature will choose any  $2n_1 + 1$  vertices, call this set  $S$ . Then, on each round, Nature will choose a new vertex from  $S$  and label it by the opposite type of the labeling the deterministic algorithm  $A$  would label that new vertex. After Nature had asked  $A$  to label all vertices in  $S$ , Nature counts and check what is the label less common in the sequence so far. Assume without loss of generality that this label type is '+', and denote the number of times it appeared so far by  $t$ . Note that  $t \leq n_1$ . Then, for the next  $n_1 - t$  rounds, Nature will choose any previously unlabeled vertex and label it by '+'. From that time on (from round  $3n_1 - t + 2$ ), Nature will choose any previously unlabeled vertex and label it by '-'. By doing this Nature labels exactly  $n_1$  of the vertices (the less common ones) with one label type and the rest with the other type.

We now turn to analyze the performance on general graphs.

**Theorem 2.3** *For any graph  $G$ , any sequence of prediction requests  $s$ , and any true labeling  $y^*$ ,*

$$m_{majority} \leq \text{CutSize}(y^*) + \alpha$$

**Proof:**

Define the sets  $R$  and  $T$  as follows:

$R = \{v \in V \text{ s.t. when } Majority \text{ was asked to label } v, \text{ none of } v\text{'s neighbors had been labeled before.}\}$

$T = \{v \in V \text{ s.t. when } Majority \text{ was asked to label } v, \text{ at least one of } v\text{'s neighbors had been labeled before.}\}$

Note that by definition,  $R \cap T = \emptyset$  and that  $R \cup T = V$ . will shall first prove two auxiliary lemmas.

**Lemma 2.3.1** *If  $u \in R, v \in R$  then  $(u, v) \notin E$ .*

**Proof:**

Suppose not. Let  $u \in R, v \in R$ , s.t.  $(u, v) \in E$ . Assume w.l.o.g. that *Majority* was asked to label  $v$  first. Then, at the later time when *Majority* was asked to label  $u, v$  was already labeled. Since  $v$  is a neighbor of  $u, u \notin R$ , a contradiction.

**Lemma 2.3.2** *Majority misclassifies at most  $\text{CutSize}(y^*)$  vertices from  $T$ .*

**Proof:**

The proof follows using amortized analysis; define a potential function  $\phi(G, y^*) = \text{CutSize}(y^*)$ . Each time a new vertex is misclassified by *Majority* algorithm, we charge that vertex with the number of edges connecting it to its neighboring vertices who had already been labeled in previous iterations, and had been labeled by  $y^*$  the opposite label of  $v$ . Clearly, since *Majority* misclassified the vertex  $v$ , at least one new edge of the cut induced by  $y^*$  is found. Therefore,

*Majority* misclassifies at most  $\phi(G, y^*)$  vertices of  $T$ .

Back to the proof of the theorem; the number of mistakes *Majority* incurs on vertices in  $R$  is upper bounded by  $\alpha$  since, by the first lemma,  $R$  is an independent set. Using the second lemma, the number of mistakes *Majority* incurs on vertices in  $T$  is upper bounded by  $\text{CutSize}(y^*)$ . Since  $R \cup T = V$ , the total number of mistakes is less than  $\text{CutSize}(y^*) + \alpha$ .

**Example 2.4** *The graph is made of two cliques connected by a single edge. The vertices in the first clique are labeled as '+' while the vertices in the second clique as '-'.  $\alpha$  is therefore two and the cut size induced by  $y^*$  is one, so *Majority* incurs at most three mistakes.*

### 3. Refining the Measure of Complexity

Using the cut-size as a measure of complexity is useful when the cut-size is smaller than the number of vertices. We next suggest a novel complexity measure, one which is always at most the number of vertices.

**Definition 3.1** Let  $\text{Frontier}^+$  be all the vertices in the cut induced by  $y^*$  that are labeled as '+' and are connected to a vertex of the opposite labeling:  $\text{Frontier}^+ = \{v : y^*(v) = '+' \text{ and } \exists u \text{ s.t. } y^*(u) = '-' \text{ and } (u, v) \in E\}$ . Similarly, define,  $\text{Frontier}^- = \{v : y^*(v) = '-' \text{ and } \exists u \text{ s.t. } y^*(u) = '+' \text{ and } (u, v) \in E\}$ .

**Definition 3.2** The *frontier of a labeling*  $y^*$  is defined as  $\text{Frontier} = \text{Frontier}^+ \cup \text{Frontier}^-$ . The *frontier size of a labeling*  $y^*$ ,  $\text{FrontierSize}(y^*)$ , is defined as the size of  $\text{Frontier}(y^*)$ .

We may now strengthen theorem 2.3.

**Theorem 3.3** *For any sequence of prediction requests  $s$ , and any true labeling  $y^*$ ,*

$$m_{\text{majority}} \leq \text{FrontierSize}(y^*) + \alpha$$

**Proof:**

Looking back that the proof of theorem 2.3, we see that the first lemma still holds. We can now define and prove an updated version for the second lemma.

**Lemma 3.3.1** *Majority misclassifies at most  $\text{FrontierSize}(y^*)$  vertices from  $T$ .*

**Proof:**

The proof follows using amortized analysis; define a potential function  $\phi(G, y^*) = \text{FrontierSize}(y^*)$ . Each time a new vertex is misclassified by *Majority* algorithm we charge that vertex with one. Clearly,

since *Majority* misclassified the vertex, at least one new frontier vertex induced by  $y^*$  is found. Therefore, *Majority* misclassifies at most  $\phi(G, y^*)$  vertices of  $T$ .

**Lemma 3.3.2** *The frontier size of the true labeling  $y^*$  is at most twice the cut-size induced by  $y^*$*

**Proof:**

Enumerate all cut edges; each new edge in the cut introduces at most two new frontier vertices and, therefore,  $\text{FrontierSize}(y^*) \leq 2 \cdot \text{CutSize}(y^*)$ .

Notice however, that  $\text{FrontierSize}(y^*)$  may be much smaller than  $\text{CutSize}(y^*)$ .

**Example 3.4** *Assume having one big clique  $A$  with  $n - 2\sqrt{n}$  vertices in it, all labeled as '+', a cliques  $B$  with  $\sqrt{n}$  vertices, all labeled '+', and a clique  $C$  with  $\sqrt{n}$  vertices, all labeled '-'. All possible edges between  $B$  and  $C$  are presented. No other edges appear in  $G$ . Clearly,  $|B| \cdot |C| = n$ , and the cut size is exactly  $n$ . However, by theorem 3.3,  $m_{\text{majority}} \leq 2\sqrt{n} + 3$ .*

Using the proofs of theorems 2.3 and 3.3 we conclude:

**Corollary 3.5** *For any true labeling  $y^*$ ,*

$$m_{\text{majority}} \leq \alpha + \min\{\text{CutSize}(y^*), \text{FrontierSize}(y^*)\}$$

*Moreover, if the sequence presented to *Majority* contains at most  $k$  non contiguous nodes, namely, there are at most  $k$  times such that the vertex presented at that time wasn't a neighbor of a previously labeled vertex, then*

$$m_{\text{majority}} \leq k + \min\{\text{CutSize}(y^*), \text{FrontierSize}(y^*)\}.$$

For example, the last bound holds with  $k = 1$  for a meta crawling application, where one moves from one web page to a neighbor webpage and needs to predict some boolean property of the webpages.

## 4. Possible extensions

### 4.1. Evolving graphs

In our framework the graph  $G$  is given in advance. In fact, a closer look at *Majority*'s performance theorems show that our analysis works for dynamic graphs where the only constraint is that when one needs to label a vertex, no new edges are allowed to be added between that vertex and **previously** labeled vertices. It is perfectly fine to have growing edges between a vertex which is not yet labeled and any other vertex.

### 4.2. Multi class labeling

In the multi-class labeling problem, each label belongs to some fixed class of discrete labels of size  $s$ .  $s = 2$

is the binary problem treated beforehand. A closer observation of the analysis of theorem 3.3 shows that it can be easily generalized to give a corresponding mistake bound for the multi-class labeling problem: the mistake bound will consist of  $\alpha$  plus the sum of all frontier vertices, where a frontier vertex is one who has a neighbor vertex of a different label type.

### 4.3. Weighted graphs

When we are given weights on the edges theorem 2.3 continues to hold if we replace *Majority* with a weighted-majority version and define the cut-size as the sum of the weighted edges in the labeling cut.

## 5. Comparison to previous results

1. In their seminal paper, (Pelckmans & Suykens, 2008) present the *Graphtron* algorithm which predicts the labeling of a vertex by the majority vote between neighboring vertices of  $v$  that had been **misclassified** before. *Majority* may be viewed as a more aggressive version of the *Graphtron* where one uses all neighboring vertices that had been previously labeled and not only the misclassified ones for the prediction task. Pelckmans and Suykens prove that the set  $M$  of vertices, where *Graphtron* predicts incorrectly, satisfies  $\sum_{v \in M} d_{M,v} \leq 4 \cdot \text{cut}(y^*)$ , where  $d_{M,v}$  is the number of vertices adjacent to  $v$  that reside in  $M$ . Their proof is based on the graph Laplacian properties and its relation with the labeling cut induced by  $y^*$ . We now give a purely combinatorial proof to show that *Majority* fulfills a corresponding mistake bound. Let  $E$  be the set of vertices where the *Majority* algorithm predicts incorrectly. Then,  $\sum_{v \in E} d_{E,v} \leq 4 \cdot \text{cut}(y^*)$ . In order to show this let  $v_i$  be the vertex that was needed to be labeled at time  $i$ . Define the vertex order, by which they were labeled, by  $O$ . Let  $n_v^{O,G}$  the number of edges in  $G$  connecting  $v$  to previously labeled vertices. Similarly, let  $n_v^{O,E}$  the number of edges connecting  $v$  to previously labeled vertices that reside in  $E$ . Then,  $\sum_{v \in E} d_{E,v} = 2 \sum_{v \in E} n_v^{O,E} \leq 2 \sum_{v \in E} n_v^{O,G} = 4 \sum_{v \in E} \frac{n_v^{O,G}}{2} \leq 4 \text{cut}(y^*)$ . The first equality is true since the sum of degrees equals twice the number of edges, and the last inequality is true since each time a mistake occurs at least half of the edges connecting a vertex to previously labeled vertices are cut edges.
2. (Herbster et al., 2005; Herbster & Pontil, 2006; Herbster, 2008) describe a series of modifications to the Perceptron Algorithm, all based on a kernel

defined by the pseudo-inverse of the graph Laplacian. In (Herbster et al., 2005), a mistake bound of the form  $m \leq \text{CutSize}(y^*) \cdot \text{Diameter}(G) \cdot \left(\frac{n}{\min n^+, n^-}\right)$ , is given, where  $\text{Diameter}(G)$  is the graph diameter, and  $n^+, n^-$ , are the number of vertices labeled as '+' and '-', respectively. In (Herbster & Pontil, 2006), this bound is improved to give  $m \leq 4 \text{CutSize}(y^*) \cdot (R(G) + 1)$ , where  $R(G)$  is the resistance diameter of  $G$  (see (Herbster & Pontil, 2006) for formal definition). In (Herbster, 2008) cluster structure in a graph is exploited. In that paper, they define the *covering number*  $N(G, \rho)$ , the minimum number of balls of diameter  $\rho$  that contain together all the vertices of  $G$ , under the *resistance distance* (see (Herbster, 2008) for definitions). The mistake bound proven therein is  $m \leq \min_{\rho > 0} N(G, \rho) + 4 \text{CutSize}(y^*) \rho + 1$ . Notice that the first two bounds mentioned are vacuous when the cut size induced by  $y^*$  is bigger than  $n$ . The *Majority* algorithm would provide better bounds in this case. Moreover, comparing to the last bound, if  $\alpha$  is not too big then the bound made by *Majority*,  $m \leq \alpha + \text{FrontierSize}(y^*)$ , would be better. Consider the following example: we have one clique with  $2\sqrt{n}$  vertices, half labeled as '+' and half as '-'. That clique is connected through a single edge to another clique, defined on the rest of the vertices, with all its vertices labeled as '+'. Clearly,  $\text{CutSize}(y^*) = n$ , rendering the bounds in (Herbster et al., 2005), (Herbster & Pontil, 2006) vacuous. The bound of (Herbster, 2008) can be shown to give  $8\sqrt{n} + 3$ . However, the bound given by *Majority* would be  $2\sqrt{n} + 3$ , since  $\alpha = 2$  and  $\text{FrontierSize}(y^*) = 2\sqrt{n} + 1$ .

## References

- Herbster, M. (2008). Exploiting cluster-structure to predict the labeling of a graph. *ALT '08*.
- Herbster, M., Lever, G., & Pontil, M. (2008). Online prediction on large diameter graphs. In *Nips '06*.
- Herbster, M., & Pontil, M. (2006). Prediction on a graph with a perceptron. *NIPS* (pp. 577–584).
- Herbster, M., Pontil, M., & Wainer, L. (2005). Online learning over graphs. *ICML '05* (pp. 305–312). Bonn, Germany.
- Pelckmans, K., & Suykens, J. A. K. (2008). An online algorithm for learning a labeling of a graph. *MLG 08', Helsinki, Finland*.