Transductive Inference with Graphs

Dengyong Zhou and Bernhard Schölkopf
Max Planck Institute for Biological Cybernetics
Spemannstr. 38, 72076 Tuebingen, Germany
{dengyong.zhou, bernhard.schoelkopf}@tuebingen.mpg.de

Abstract

We propose a general regularization framework for transductive inference. The given data are thought of as a graph, where the edges encode the pairwise relationships among data. We develop discrete analysis and geometry on graphs, and then naturally adapt the classical regularization in the continuous case to the graph situation. A new and effective algorithm is derived from this general framework, as well as an approach we developed before.

1 Introduction

Many real-life machine learning problems can be described as follows: given a set of objects, only some of them are labeled; and our task is to predict the labels of remaining unlabeled objects. We can apply any supervised learning algorithm to this problem, i.e., training a classifier with the labeled objects and then use the trained classifier to predict the labels of the unlabeled objects. However, this way is generally not the best. Our task is only to estimate the labels of the given unlabeled data, and not to estimate a classifying function on the continuum of objects containing these unlabeled objects, which is a more complex problem. According to Vapnik's principle, we should directly estimate the labels of the unlabeled objects without depending on the solution of a more complex problem [15]. Such an estimation problem is called transductive inference.

If the objects are totally unrelated to each other, we can not make any prediction statistically better than random guess. Hence we assume that there are pairwise relationships among data and closely related objects are likely to have the same labels. For example, web pages are connected to each other by hyperlinks. A densely connected set of web pages generally have a similar topic. For points distributed in Euclidian space, such as image and text data, the affinity among points may be measured by a function of pairwise Euclidean distances, such as a kernel function, and points on the same cluster or manifold are likely to belong to the same class [4]. A dataset endowed with pairwise relationships can be thought of as a graph, in which the edges encode the relationships among objects. This idea is adopted in many recent works on spectral clustering [11, 10], transductive inference or semi-supervised learning [2, 13, 1, 18, 8, 17], and graph kernels [12]. Most of them are essentially built on the so-called graph Laplacian [5].

Here we propose a general regularization framework on graphs. We first develop the discrete analysis and geometry on graphs,[1] and then construct the regularizer using the discrete

[1]Unless otherwise stated, this material expanded in Section 2 is original. Certain elements of it, to

differential operators. This framework can be considered as the discrete analogue of variational methods on Riemannian manifolds [6, 7] and classical regularization [14, 16] in the continuous case. We follow the notation used in differential topology and geometry, which can be found in any related standard textbook (cf. [9]). The transductive inference algorithm proposed by [17] can be naturally derived from this framework, as well as a new and effective method.

2 Differential Geometry on Graphs

A graph $\Gamma = (V, E)$ consists of a set V of vertices and a set of pairs of vertices $E \subseteq V \times V$ called edges. A graph is undirected if for each edge $(u, v) \in E$ we also have $(v, u) \in E$. Edge e is incident on vertex v if e contains v. A graph is connected if there is a path from every vertex to every other vertex. Here we only consider undirected and connected graphs. Moreover, the graphs have no self-loops or multiple edges. A graph is weighted if it is associated with a function $w : E \to \mathbb{R}_+$ satisfying $w(u, v) = w(v, u)$. The degree function $g : V \to \mathbb{R}_+$ is defined to be

$$g(v) := \sum_{u \sim v} w(u, v), \tag{2.1}$$

where $u \sim v$ denote the set of vertices u connected to v via the edges (u, v). The degree can be regarded as a measure [5]. Let $\mathcal{H}(V)$ denote the Hilbert space of real-valued functions endowed with the usual inner product

$$\langle \varphi, \phi \rangle := \sum_{v} \varphi(v)\phi(v), \tag{2.2}$$

where φ and ϕ denote any two functions in $\mathcal{H}(V)$. Similarly define $\mathcal{H}(E)$. Note that function $\psi \in \mathcal{H}(E)$ need not to be symmetric, i.e., we do not require $\psi(u, v) = \psi(v, u)$.

2.1 Boundary Operator

We define the *boundary* operator

$$d : \mathcal{H}(V) \to \mathcal{H}(E)$$

to be

$$(d\varphi)(u, v) := \sqrt{\frac{w(u, v)}{g(u)}}\varphi(u) - \sqrt{\frac{w(u, v)}{g(v)}}\varphi(v), \text{ for all } (u, v) \in E. \tag{2.3}$$

Clearly,

$$(d\varphi)(u, v) = -(d\varphi)(v, u), \tag{2.4}$$

i.e., $d\varphi$ is skew-symmetric. We define the adjoint

$$d^* : \mathcal{H}(E) \to \mathcal{H}(V)$$

of d by

$$\langle d\varphi, \psi \rangle = \langle \varphi, d^*\psi \rangle, \text{ for all } \varphi \in \mathcal{H}(V), \psi \in \mathcal{H}(E). \tag{2.5}$$

We call d^* the *co-boundary* operator. Note that the inner products in the left and right side of (2.5) are respectively in the space $\mathcal{H}(E)$ and $\mathcal{H}(V)$. We can show that d^* is given by (see Appendix A)

$$(d^*\psi)(v) = \sum_{u \sim v} \sqrt{\frac{w(u, v)}{g(v)}} \Big(\psi(v, u) - \psi(u, v) \Big). \tag{2.6}$$

be mentioned below, are related to spectral graph theory literature [5], and some parts of our theory have previously been developed for the special case of lattices in the image processing community [3].

The boundary and co-boundary operators are respectively the discrete analogues of the gradient and divergence operators in continuous case [9].

The *edge derivative*

$$\left.\frac{\partial}{\partial e}\right|_v : \mathcal{H}(V) \to \mathbb{R}$$

along edge $e = (v, u)$ at vertex v is defined by

$$\left.\frac{\partial \varphi}{\partial e}\right|_v := (d\varphi)(v, u). \tag{2.7}$$

Define the *local variation* of φ at v to be

$$\left\|\nabla_v \varphi\right\| := \left[\sum_{e \vdash v} \left(\left.\frac{\partial \varphi}{\partial e}\right|_v\right)^2\right]^{1/2} \tag{2.8}$$

where $e \vdash v$ denotes the set of edges incident on v. Let $\mathcal{S}$ denote a functional on $\mathcal{H}(V)$, for any $p \in [1, \infty)$, which is defined to be

$$\mathcal{S}_p(\varphi) := \frac{1}{p} \sum_v \left\|\nabla_v \varphi\right\|^p. \tag{2.9}$$

The functional $\mathcal{S}_p(\varphi)$ can be thought of as the measure of the *smoothness* of φ.

2.2 Laplace Operator

By analogy with the Laplace-Beltrami operator on forms on Riemannian manifolds [9], we define the *graph Laplacian*

$$\Delta : \mathcal{H}(V) \to \mathcal{H}(V)$$

by

$$\Delta := \frac{1}{2} d^* d. \tag{2.10}$$

Clearly, Δ is a linear operator because both d^* and d are linear. Furthermore, Δ is self-adjoint:

$$\langle d^* d\varphi, \phi \rangle = \langle d\varphi, d\phi \rangle = \langle \varphi, d^* d\phi \rangle. \tag{2.11}$$

An important consequence of (2.11) is

$$\langle \Delta\varphi, \varphi \rangle = \frac{1}{2} \langle d\varphi, d\varphi \rangle = \mathcal{S}_2(\varphi), \tag{2.12}$$

which implies that Δ is positive semi-definite. It follows from (2.12) that

$$\Delta\varphi = \frac{\partial \mathcal{S}_2(\varphi)}{\partial \varphi}. \tag{2.13}$$

Substituting (2.3) and (2.6) into (2.10), we have

$$(\Delta\varphi)(v) = \varphi(v) - \sum_{u \sim v} \frac{w(u, v)}{\sqrt{g(u) g(v)}} \varphi(u). \tag{2.14}$$

In spectral graph theory [5], (2.14) is directly used as the definition of the graph Laplacian. Our derivation of it, however, is new.

We can also define the graph Laplacian with the notion of edge derivative as

$$(\Delta\varphi)(v) := \frac{1}{2} \sum_{e \vdash v} \frac{1}{\sqrt{g}} \left(\frac{\partial}{\partial e} \sqrt{g} \frac{\partial \varphi}{\partial e}\right)\Bigg|_v. \tag{2.15}$$

This is basically the discrete analogue of another definition of the Laplace-Beltrami operator based on the gradient [9]. The equivalence between (2.15) and (2.10) is shown in Appendix B. See [9] for the corresponding equivalence in the continuous case.

2.3 Curvature Operator

By analogy with the curvature of a surface which is measured by the change in the unit normal [9], we define the *graph curvature*

$$\kappa : \mathcal{H}(V) \to \mathcal{H}(V)$$

by

$$\kappa\varphi := d^*\left(\frac{d\varphi}{\|\nabla\varphi\|}\right). \tag{2.16}$$

Substituting (2.3) and (2.6) into (2.16), we have

$$(\kappa\varphi)(v) = \sum_{u\sim v} \frac{w(u,v)}{\sqrt{g(v)}}\left(\frac{1}{\|\nabla_v\varphi\|} + \frac{1}{\|\nabla_u\varphi\|}\right)\left(\frac{\varphi(v)}{\sqrt{g(v)}} - \frac{\varphi(u)}{\sqrt{g(u)}}\right). \tag{2.17}$$

Unlike the graph Laplacian given by Equality (2.10), the graph curvature is a non-linear operator.

Similar to Equality (2.15), we also have an equivalent definition of the graph curvature based on the gradient:

$$(\kappa\varphi)(v) := \sum_{e\vdash v} \frac{1}{\sqrt{g}}\left(\frac{\partial}{\partial e}\frac{\sqrt{g}}{\|\nabla\varphi\|}\frac{\partial\varphi}{\partial e}\right)\Bigg|_v. \tag{2.18}$$

Moreover, as in Equality (2.13), an elegant property of the graph curvature is

$$\kappa\varphi = \frac{\partial\mathcal{S}_1(\varphi)}{\partial\varphi}. \tag{2.19}$$

This can be shown by the following:

$$\begin{aligned}
\frac{\partial\mathcal{S}_1(\varphi)}{\partial\varphi}\Bigg|_v &= \sum_{u\sim v}\left[\frac{w(u,v)}{\|\nabla_v\varphi\|}\left(\frac{\varphi(v)}{g(v)} - \frac{\varphi(u)}{\sqrt{g(u)g(v)}}\right) + \frac{w(u,v)}{\|\nabla_u\varphi\|}\left(\frac{\varphi(v)}{g(v)} - \frac{\varphi(u)}{\sqrt{g(u)g(v)}}\right)\right] \\
&= \sum_{u\sim v} w(u,v)\left(\frac{1}{\|\nabla_v\varphi\|} + \frac{1}{\|\nabla_u\varphi\|}\right)\left(\frac{\varphi(v)}{g(v)} - \frac{\varphi(u)}{\sqrt{g(u)g(v)}}\right) \\
&= \sum_{u\sim v} \frac{w(u,v)}{\sqrt{g(v)}}\left(\frac{1}{\|\nabla_v\varphi\|} + \frac{1}{\|\nabla_u\varphi\|}\right)\left(\frac{\varphi(v)}{\sqrt{g(v)}} - \frac{\varphi(u)}{\sqrt{g(u)}}\right).
\end{aligned}$$

Comparing the last equality with (2.17), we complete the proof.

3 Regularization on Graphs

Given a graph $\Gamma = (V, E)$, let y denote a function in $\mathcal{H}(V)$, defined by $y(v) = 1$ or -1 if v is labeled as positive or negative and 0 otherwise. Our goal is to search for another function f in $\mathcal{H}(V)$, which is not only *smooth* enough on Γ but also *close* enough to the given function y. Then each object v is classified as $\operatorname{sign} f(v)$. This idea is formalized via the following optimization problem:

$$\operatorname*{argmin}_{f\in\mathcal{H}(V)}\{\mathcal{S}_p(f) + \frac{\mu}{2}\|f - y\|^2\}. \tag{3.1}$$

The first term in (3.1) is the *smoothness term* or *regularizer*, which requires f not to change too much between closely related objects. The second term is the *fitting term*, which says that f should not be far away from y. The trade-off between these two competing terms is captured by a positive parameter μ. Clearly, the smaller the parameter μ, the smoother the function f.

3.1 2-Smoothness

In the case of $p = 2$, the optimization problem (3.1) is

$$\operatorname*{argmin}_{f\in\mathcal{H}(V)} \left\{ \frac{1}{2} \sum_v \left\|\nabla_v f\right\|^2 + \frac{\mu}{2}\|f - y\|^2 \right\}. \tag{3.2}$$

By Equality (2.13), we have

Theorem 1. *The solution of (3.2) satisfies*

$$\Delta f + \mu(f - y) = 0. \tag{3.3}$$

Equality (3.3) can be thought of as a second-order differential equation on graphs. Since Δ is a linear operator, this is a linear equation and there is a simple closed form solution,

$$f = \mu(\Delta + \mu I)^{-1} y, \tag{3.4}$$

where I denotes the identity operator. It is not hard to see that (3.4) is in fact the algorithm proposed by [17].

3.2 1-Smoothness

In the case of $p = 1$, the optimization problem (3.1) is

$$\operatorname*{argmin}_{f\in\mathcal{H}(V)} \left\{ \sum_v \left\|\nabla_v f\right\| + \frac{\mu}{2}\|f - y\|^2 \right\}. \tag{3.5}$$

By Equality (2.19), we have

Theorem 2. *The solution of (3.5) satisfies*

$$\kappa f + \mu(f - y) = 0. \tag{3.6}$$

As we have mentioned, the curvature κ is a non-linear operator, and we are not aware of any closed form solution for this equation. However, we can construct an iterative algorithm to obtain the solution. Substituting (2.17) into (3.6), we have

$$\sum_{u\sim v} \frac{w(u,v)}{\sqrt{g(v)}} \left(\frac{1}{\|\nabla_u f\|} + \frac{1}{\|\nabla_v f\|} \right) \left(\frac{f(v)}{\sqrt{g(v)}} - \frac{f(u)}{\sqrt{g(u)}} \right) + \mu(f(v) - y(v)) = 0. \tag{3.7}$$

Define the function $m : E \to \mathbb{R}$ by

$$m(u,v) = w(u,v) \left(\frac{1}{\|\nabla_u f\|} + \frac{1}{\|\nabla_v f\|} \right). \tag{3.8}$$

Then

$$\sum_{u\sim v} \frac{m(u,v)}{\sqrt{g(v)}} \left(\frac{f(v)}{\sqrt{g(v)}} - \frac{f(u)}{\sqrt{g(u)}} \right) + \mu(f(v) - y(v)) = 0,$$

which can be transformed into

$$\left(\sum_{u\sim v} \frac{m(u,v)}{g(v)} + \mu \right) f(v) = \sum_{u\sim v} \frac{m(u,v)}{\sqrt{g(u)g(v)}} f(u) + \mu y(v).$$

Define the function $p : E \to \mathbb{R}$ by

$$p(u,v) = \frac{\dfrac{m(u,v)}{\sqrt{g(u)g(v)}}}{\displaystyle\sum_{u\sim v} \frac{m(u,v)}{g(v)} + \mu}, \text{ if } u \neq v; \text{ and } p(v,v) = \frac{\mu}{\displaystyle\sum_{u\sim v} \frac{m(u,v)}{g(v)} + \mu}. \tag{3.9}$$

Then

$$f(v) = \sum_{u \sim v} p(u,v) f(v) + p(v,v) y(v). \tag{3.10}$$

Thus we can consider using the iteration

$$f^{(t+1)}(v) = \sum_{u \sim v} p^{(t)}(u,v) f^{(t)}(v) + p^{(t)}(v,v) y(v), \text{ for all } v \in V \tag{3.11}$$

to obtain the solution of (3.5), in which the coefficients $p^{(t)}$ are updated according to Equality (3.9) and further (3.8). The initial value can be set by $f^{(0)}(v) = y(v)$ for all $v \in V$. In addition, with the progress of the iteration, the function will be very smooth on some vertices, which will lead the local variations at these vertices to be almost zero. Hence, for the good numerical conditioning in Equality (3.8), we regularize the local variation (2.8) as

$$\left\| \nabla_v \varphi \right\| = \left[\sum_{e \vdash v} \left(\left. \frac{\partial \varphi}{\partial e} \right|_v \right)^2 + \epsilon \right]^{1/2}, \tag{3.12}$$

where ϵ is a small positive number. (In our experiment, $\epsilon = 1e-10$.)

Note that the 2-smoothness method also has a corresponding iteration algorithm [17], which can be intuitively understood as information diffusion. The basic difference between these two iteration procedures is that in the 1-smoothness iteration the weight coefficients $p(u,v)$ are also *adaptively* updated at each iteration, in addition to the classifying function being updated. This weight update causes the diffusion inside clusters to be enhanced, and the diffusion across clusters to be reduced.

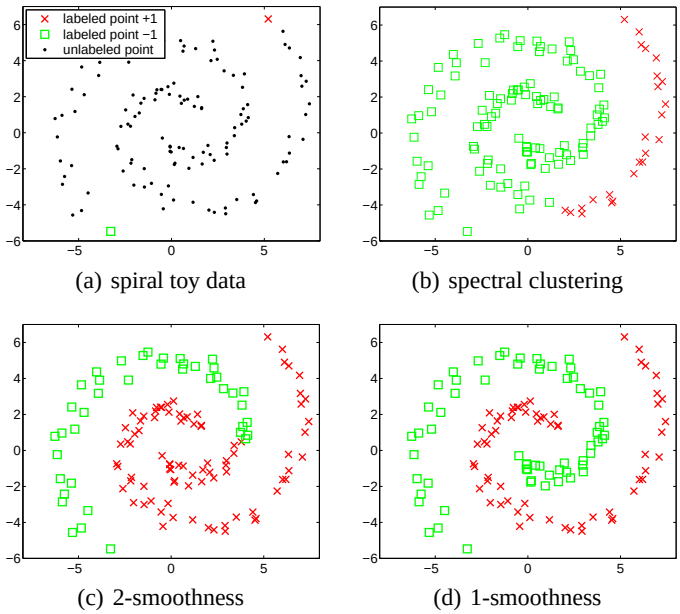
4 Experiment

As we have mentioned before, the 2-smoothness method is that proposed by [17]. Hence, for those interested in a comparison with other methods may refer to [17]. Our interest here is only to explore the differences between the 2-smoothness and 1-smoothness regularizers.

We consider a toy problem shown in Figure 1(a), which consists of two intertwined spirals in $\mathbb{R}^2$. This toy data can be viewed as the elongated version of the two moon pattern discussed in [17]. Let u and v denote any two points in this dataset. The pairwise relationships among the data is defined by $w(u,v) = \exp(-\lambda \|u - v\|)$ if $u \neq v$ and 0 otherwise, where λ is a positive parameter and $\|\cdot\|$ denotes Euclidean norm. Then this dataset can be thought of as a fully connected weighted graph.

This is not a trivial classification problem. The spectral clustering methods [11, 10] can not correctly cluster the toy data into two inherent clusters (Figure 1(b)). The 2-smoothness method also fails to capture the intrinsic structure aggregated by the data (Figure 1(c)). Note that both the 2-smoothness regularizer and the spectral clustering algorithms are built on the graph Laplacian. The classification result from the 1-smoothness method is shown in Figure 1(d). In the toy problem, the 1-smoothness approach is more effective in detecting the edges of different clusters than the 2-smoothness method.

Unfortunately, the convergence of the iteration procedure for the 1-smoothness algorithm is presently too slow to evaluate it on a significant real-world dataset with large amounts of unlabeled data. We are working on speeding up the iteration method. In addition, it may be possible to find other efficient ways to optimize the 1-smoothness cost function instead of this simple iteration.



(a) spiral toy data (b) spectral clustering

(c) 2-smoothness (d) 1-smoothness

Figure 1: Classification on the spiral toy data. Note that only 1-smoothness algorithm can correctly classify the points located at the tightly intertwined center.

5 Discussion

We have introduced a framework comprising certain discrete analogue of differential geometry on graphs. This allowed us to derive the algorithm proposed by [17], which is a powerful transductive inference method, as a regularization approach. Moreover, we proposed another approach, which is based on the (nonlinear) curvature rather than the (linear) Laplacian, and illustrated it using a toy example. Note that this corresponds to a nonlinear regularization, which is related to anisotropic diffusion or curvature flow, as used in the image processing community (cf. [3]), rather than isotropic diffusion or heat flow (cf. [6]), which is known to correspond to Laplacian regularization.

Acknowledgements Thank Mikhail Belkin, Olivier Bousquet, Olivier Chapelle, Bin Yu, and Ingo Steinwart for helpful discussions, and Arthur Gretton and Matthias Hein for their help.

References

[1] M. Belkin, I. Matveeva, and P. Niyogi. Regression and regularization on large graphs. Technical report, University of Chicago, 2003.

[2] A. Blum and S. Chawla. Learning from labeled and unlabeled data using graph mincuts. In *ICML*, 2001.

[3] T. Chan and J. Shen. Variational restoration of non-flat image features: models and algorithms. *SIAM Journal of Applied Mathematics*, 61(4):1338–1361, 2000.

[4] O. Chapelle, J. Weston, and B. Schölkopf. Cluster kernels for semi-supervised learning. In *NIPS*, 2002.

[5] F. Chung. *Spectral Graph Theory*. Number 92 in CBMS-NSF Regional Conference Series in Mathematics. SIAM, 1997.

[6] J. Eells and J. H. Sampson. Harmonic mappings of Riemannian manifolds. *American Journal of Mathematics*, 86:109–160, 1964.

[7] R. Hardt and F. H. Lin. Mappings minimizing the L^p norm of the gradient. *Communications on Pure and Applied Mathematics*, 40:556–588, 1987.

[8] T. Joachims. Transductive learning via spectral graph partitioning. In *ICML*, 2003.

[9] J. Jost. *Riemannian Geometry and Geometric Analysis*. Springer-Verlag, Berlin-Heidelberg, third edition, 2002.

[10] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *NIPS*, 2001.

[11] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[12] A. Smola and R. I. Kondor. Kernels and regularization on graphs. In *Learning Theory and Kernel Machines*. Springer-Verlag, Berlin-Heidelberg, 2003.

[13] M. Szummer and T. Jaakkola. Partially labeled classification with Markov random walks. In *NIPS*, 2001.

[14] A. N. Tikhonov and V. Y. Arsenin. *Solutions of Ill-posed Problems*. Wiley, NY, 1977.

[15] V. N. Vapnik. *Statistical learning theory*, pages 339–371. Wiley, NY, 1998.

[16] G. Wahba. *Spline Models for Observational Data*. Number 59 in CBMS-NSF Regional Conference Series in Applied Mathematics. SIAM, 1990.

[17] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *NIPS*, 2003.

[18] X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using Gaussian fields and harmonic functions. In *ICML*, 2003.

A Co-boundary Operator

Expand the left side of (2.5):

$$\begin{aligned}
\langle d\varphi, \psi\rangle &= \sum_{(u,v)\in E} d\varphi(u,v)\psi(u,v) = \sum_{(u,v)\in E}\left(\sqrt{\frac{w(u,v)}{g(u)}}\varphi(u) - \sqrt{\frac{w(u,v)}{g(v)}}\varphi(v)\right)\psi(u,v)\\
&= \sum_{(u,v)\in E}\sqrt{\frac{w(u,v)}{g(u)}}\varphi(u)\psi(u,v) - \sum_{(u,v)\in E}\sqrt{\frac{w(u,v)}{g(v)}}\varphi(v)\psi(u,v)\\
&= \sum_r\sum_{v\sim r}\sqrt{\frac{w(r,v)}{g(r)}}\varphi(r)\psi(r,v) - \sum_r\sum_{u\sim r}\sqrt{\frac{w(u,r)}{g(r)}}\varphi(r)\psi(u,r)\\
&= \sum_r \varphi(r)\sum_{v\sim r}\sqrt{\frac{w(r,v)}{g(r)}}\Big(\psi(r,v) - \psi(v,r)\Big).
\end{aligned}$$

The last equality implies Equality (2.6).

B Laplace Operator

Here we show that the definitions (2.10) and (2.15) are equivalent:

$$\begin{aligned}
(\Delta\varphi)(v) &= \frac{1}{2\sqrt{g(v)}}\sum_{e\vdash v}\left[\sqrt{\frac{w(u,v)}{g(u)}}\left(\sqrt{g}\frac{\partial\varphi}{\partial e}\right)\Big|_u - \sqrt{\frac{w(u,v)}{g(v)}}\left(\sqrt{g}\frac{\partial\varphi}{\partial e}\right)\Big|_v\right]\\
&= \frac{1}{2\sqrt{g(v)}}\sum_{e\vdash v}\sqrt{w(u,v)}\left(\frac{\partial\varphi}{\partial e}\Big|_u - \frac{\partial\varphi}{\partial e}\Big|_v\right) = \frac{1}{\sqrt{g(v)}}\sum_{e\vdash v}\sqrt{w(u,v)}\frac{\partial\varphi}{\partial e}\Big|_v\\
&= \frac{1}{\sqrt{g(v)}}\sum_{u\sim v}\left(\frac{w(u,v)}{\sqrt{g(v)}}\varphi(v) - \frac{w(u,v)}{\sqrt{g(u)}}\varphi(u)\right) = \varphi(v) - \sum_{u\sim v}\frac{w(u,v)}{\sqrt{g(u)g(v)}}\varphi(u).
\end{aligned}$$

Comparing the last equality with Equality (2.14), we complete the proof.