Learn to Propagate Reliably on Noisy Affinity Graphs (Supplementary Materials)

Lei Yang[1[0000−0002−0571−5924]], Qingqiu Huang[1[0000−0002−6467−1634]], Huaiyi Huang[1[0000−0003−1548−2498]], Linning Xu[2[0000−0003−1026−2410]], and Dahua Lin[1[0000−0002−8865−7896]]

[1] The Chinese University of Hong Kong
[2] The Chinese University of Hong Kong, Shenzhen
{yl016,hq016,hh016,dhlin}@ie.cuhk.edu.hk,linningxu@link.cuhk.edu.cn

1 Prove properties of multi-view confidence.

Let $\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_n$ be the n predictions obtained from n *views* for a vertex v, where $\mathbf{p}_i = (p_{i1}, p_{i2}, ..., p_{im})$, m is the number of classes, and $\sum_{j=1}^{m} p_{ij} = 1$, $p_{ij} \in [0, 1]$. The final prediction of v is defined as

$$\mathbf{p} = (\overline{p}_1, \overline{p}_2, ..., \overline{p}_m), \tag{1}$$

where $\overline{p}_j = \frac{1}{n}\sum_{i=1}^{n} p_{ij}$, $\sum_{j=1}^{m} \overline{p}_j = 1$. Recall the confidence of v is defined as

$$c = \max_j \overline{p}_j, \tag{2}$$

Our hypothesis is that c takes a high value only when predictions are consistent and all with low entropy.

First, we prove that a high value of c implies predictions are consistent. Consider n prediction values on dimension j, *i.e.*, $\{p_{1j}, p_{2j}, ..., p_{nj}\}$. We use variance σ_j^2 to describe the consistency of predictions. The smaller σ_j^2, the higher consistency of predictions, and vice versa. σ_j^2 can be represented as

$$\sigma_j^2 = \frac{1}{n}\sum_{i=1}^{n} p_{ij}^2 - (\frac{1}{n}\sum_{i=1}^{n} p_{ij})^2 = \frac{1}{n}\sum_{i=1}^{n} p_{ij}^2 - \overline{p}_j^2 \tag{3}$$

From Eq. 3 and $p_{ij} \in [0, 1]$, we have,

$$\begin{aligned} \overline{p}_j &= \sqrt{\frac{1}{n}\sum_{i=1}^{n} p_{ij}^2 - \sigma_j^2} \leq \sqrt{1 - \sigma_j^2} \\ \overline{p}_j &= \frac{1}{n}\sum_{i=1}^{n} p_{ij} \geq \frac{1}{n}\sum_{i=1}^{n} p_{ij}^2 = \overline{p}_j^2 + \sigma_j^2 \geq \sigma_j^2 \end{aligned} \tag{4}$$

Thus, c is bounded by

$$\sigma_k^2 \leq c \leq \sqrt{1-\sigma_k^2}, \tag{5}$$

where $k = \text{argmax}_j\, \overline{p}_j$. Hence, the bound of c depends on σ_k. When σ_k is 0, the lower bound of c is 0 and the upper bound of c is 1. As σ_k increases, the lower bound of c increases and the upper bound of c decreases. Therefore, one necessary condition for c to take a high value is the small variance of the prediction values, which indicates the high consistency of predictions.

Second, we prove that a high value of c implies predictions have low entropy. Since we have proved that a high value of c implies predictions are consistent, we only need to consider the entropy of final prediction (Eq. 1). The normalized entropy E of $\mathbf{p}$ can be written as

$$E = -\sum_j \overline{p}_j \log_m \overline{p}_j = -c \log_m c + E', \tag{6}$$

where E' includes $m-1$ values except the maximum one. To normalize the maximum value of entropy to 1, we use m as the base of log. For simplicity, we do not mention it in the proof below. By Jensen's inequality, we have

$$\begin{aligned} E' &= -\sum_{j \neq k} \overline{p}_j \log \overline{p}_j \\ &\leq -(\sum_{j \neq k} \overline{p}_j) \log \frac{\sum_{j \neq k} \overline{p}_j}{m-1} = -(1-c) \log \frac{1-c}{m-1} \end{aligned} \tag{7}$$

From (6)(7), we have

$$E \leq -c \log c - (1-c) \log \frac{1-c}{m-1}, \tag{8}$$

Let $f(c) = -c \log c - (1-c) \log \frac{1-c}{m-1}$. Taking the derivative of $f(c)$, we have

$$\frac{\partial f(c)}{\partial c} = \log \frac{1-c}{c(m-1)} \leq 0, \tag{9}$$

where $\frac{1}{m} \leq c \leq 1$, as c is the maximum value. Since Eq. 9 is consistently below 0, it illustrates that $f(c)$ monotonically decreases when $c \in [\frac{1}{m}, 1]$. When E takes a small value, it implies that the lower bound of $f(c)$ decreases and thus the upper bound of c increases; When E takes a high value, it implies that the lower bound of $f(c)$ increases and thus the upper bound of c decreases. Therefore, another necessary condition for c to take a high value is the small lower bound of $f(c)$, which indicates $\mathbf{p}$ has small entropy.

From previous discussions, we prove that c takes a high value only when predictions are consistent and all with low entropy.

Table 1: Performance comparison of transductive methods on noisy affinity graphs in ImageNet. We randomly select classes and initial seeds for 5 times and report the average results of 5 runs

	ImageNet			
Noise ratio ρ	0%	10%	30%	50%
LP [11]	77.74±2.65	70.51±2.27	59.47±1.8	51.43±1.48
GCN [4]	83.17±3.65	75.37±3.12	66.28±2.87	64.09±2.69
GAT [6]	83.93±3.36	75.99±2.56	66.3±3.12	63.34±2.06
GraphSAGE [2]	82.42±1.07	73.42±2.88	63.84±3.42	59.12±3.45
GraphSAGE† [2]	81.39±3.37	73.53±2.85	63.42±2.82	58.99±3.88
FastGCN [1]	81.34±3.84	74.08±3.24	63.79±3.24	58.81±2.76
SGC [7]	84.78±3.35	76.71±3.0	67.97±2.51	65.63±2.58
Ours	**85.16±3.24**	**76.96±2.85**	**69.28±2.45**	**68.25±1.89**

2 Experimental results with standard deviation

As introduced in Sec. 4.1 in the main text, we repeat our experiments by 5 runs. For each run, we first randomly select 10 classes from $\mathcal{D}_{all}$ and randomly split 1% as the labeled initial seeds. Then, we randomly select ρ percent of images not belong to the 10 classes as the noise. As shown in the two tables below, we observe that the randomly selected 10 classes result in a large standard deviation among different runs, but the proposed method consistently outperforms previous approaches and thus achieving higher mean accuracy.

Table 2: Performance comparison of transductive methods on noisy affinity graphs in Ms-Celeb-1M. We randomly select classes and initial seeds for 5 times and report the average results of 5 runs

	Ms-Celeb-1M (1%)			
Noise ratio ρ	0%	10%	30%	50%
LP [11]	95.13±1.29	89.01±1.33	88.31±1.17	87.19±0.99
GCN [4]	99.6±0.11	99.6±0.38	96.37±0.42	96.3±0.36
GAT [6]	99.59±0.04	96.48±0.51	94.24±0.68	94.01±0.73
GraphSAGE [2]	99.57±0.1	95.68±0.28	92.21±0.58	91.06±0.48
GraphSAGE† [2]	99.59±0.08	95.62±0.37	92.38±0.4	91.19±0.46
FastGCN [1]	99.62±0.07	95.6±0.35	92.08±0.52	90.83±0.81
SGC [7]	99.63±0.07	97.43±0.35	96.71±0.32	96.5±0.31
Ours	**99.66±0.05**	**97.59±0.34**	**96.93±0.25**	**96.81±0.24**

3 More Details and Analysis

Details about experimental settings. All algorithms use the same affinity graph constructed as follows. We regard each sample as a vertex and connect it

with its K nearest samples. The edge weight $e_{i,j}$ is the cosine similarity between v_i and v_j if $(i,j) \in \mathcal{E}$, otherwise it is zero. $\mathcal{G}$ can be alternatively represented as a sparse affinity matrix $\mathbf{A}$, where the space complexity is $O(NK)$. The experiments show that the results are not sensitive to K when varying it from 30 to 80 on two datasets. Therefore, we choose $K = 30$ to reduce memory overhead. For GCN, we use a 2-layer GCN with 256 hidden units at each layer. For GraphSAGE, the sample size of GraphSAGE is set to 15 for both two layers and the batch size is set to 32. For FastGCN, we sample $6,000$ one-hop neighbors and $1,000$ two-hop neighbors.

Details about efficiency of path extraction. We refer to *visited times* of a vertex as the number of patches it belongs to. We conduct experiments on ImageNet with $10K$ vertices with $c_\tau = 0.9, \Delta c_\tau = 500$ and $s = 3000$. When propagating 100 iterations, the average visited times of vertices are about 6. Most samples are visited 2 times and only a very few samples are visited more than 10 times. As the number of visited times increases, the ratio of incorrect samples over correct ones increases, indicating that regions with low confidence will be visited more times than those with high confidence.

Besides, at the end of the propagation, the high-confidence set contains about 45% of all the samples, while only 0.1% samples has been selected as the start vertex. Since our patch extraction considers expected confidence gain, it will ignore the start vertices which reside in the local patch that already has very high confidence, which shows the effectiveness of our approach.

Details about active learning experiments. To test the effectiveness of indictors in active learning, we use three different indicators to select the same number of unlabeled samples for annotation. All methods annotate as much unlabeled data as the initial labeled data. The unlabeled *in-class* samples are annotated to their ground-truth classes and the unlabeled *out-of-class* samples are annotated to -1. For a fair comparison, we train a standard GCN [4] using the previous initial labeled samples in conjunction with the annotated samples, where the outliers are excluded in the training. Note that if applying the proposed propagation algorithm, it can also exploit the annotated out-of-class samples for confidence estimation.

The experimental setting is the same as the other ablation studies in the main text, where the labeled ratio is 1% and the noise ratio is 50%. The number of annotated samples is also 1%. As shown in Table. 4(a) in the main text, *Baseline* indicates the accuracy of GCN before annotation. *Random* only brings a slight performance gain as it may randomly select a lot of easy samples for annotation. *GCN* improves accuracy through annotating some low confident samples, which are hard for GCN to recognize in previous training. Our estimated confidence addresses the noise issue on the unlabeled data, leading to larger performance gain by annotating the same number of unlabeled data.

Except the out-of-sample noise in the real-world, the emergence of adversarial samples become a new kind of possible noise source [5]. How to extend our methods to boarder noise scenario remains to be studied in the future.

Details about inductive learning experiments. We show that the proposed method can be applied to inductive learning by providing the "pseudo labels". We compare with two methods for generating "pseudo labels" and apply them to train face recognition model in a supervised manner. CDP [10] is a recent face clustering method and does not require the labeled seeds. GCN [4] leverages the initial labeled seeds and propagate the labels to the rest unlabeled samples. Baseline refers to the result before training with pseudo labels.

For the experiment, we randomly select $1K$ person with $120K$ images from Ms-Celeb-1M and randomly samples 1% as the labeled data. To simulate the noisy setting, we then randomly sample $60K$ face images outside the selected $1K$ person. We use the same pretrained face model from [9] and regard its performance as the *Baseline*. As shown in Table. 4(b) in the main text, without confidence design, GCN only performs slightly better than the unsupervised single version of CDP. Our method surpasses the previous two methods by providing more accurate pseudo labels. As the face recognition model is well learned, it is significant to improve $\sim$2% on the challenging MegaFace [3], using only $180K$ noisy unlabeled data. In real practice, instead of discarding the identified noise, we can further boost the performance by performing clustering algorithms [9, 8] on the identified our-of-class samples, which may discover some new classes for training.

The influence of graph patches with different scales. The scale of graph patches affects the efficiency and accuracy of propagation. In our approach, the scale of the graph patch is *dynamic*. Small patches would be extracted at the beginning, since most of the samples are unlabeled and the expected confidence gain is easy to achieve. As the propagation proceeds, the number of confident vertices increases while the average expected confidence gain decreases, the algorithm encourages more aggressive updates over larger patches. Although we

Table 3: Influence of maximum patch size

Maximum patch size (s)	Accuracy	Memory	Runtime
100	64.63	6M	34s
500	65.41	28M	111s
1000	65.64	56M	158s
3000	65.83	168M	316s
5000	65.81	280M	564s
10000	65.92	561M	1077s

cannot directly control the scale of each graph patch in our approach, we can change the maximum patch size, which would also show the influence of patch size. We set $\Delta c_\tau = 500$ and vary maximum patch size from 100 to 10000. As shown in the table above, when the maximum size of graph patch is too small ($s = 100$), the graph convolutional networks can only leverage a limited number of neighbors for GCN prediction, resulting in the inferior performance. The accuracy increases with the increase of s and it gradually saturates as s increases

beyond 3000. When the maximum size of graph patch is close to the entire graph size ($s = 10000$), it introduces a large amount of computational cost and memory overhead but receives slightly performance gain.

Table 4: Design choices to include confidence into Eq.1

Design	Accuracy	Memory	Runtime
(1)	61.5	168M	316s
(2)	65.9	205M	386s
Ours	65.8	168M	316s

Design choices to include confidence into Eq.1. We investigate two design choices to include confidence into Eq.1. in the main text. First, we can apply the suggested Hadamard product. As the initial confidence of unlabeled data is small, their node features after multiplication is negligible in GCN's prediction. Ignoring neighbor information may potentially impair the initial predication and the propagation later on. Second, we can concatenate the confidence distribution $\mathbf{p}$ to vertex feature $\mathbf{x}$. Although it does not suffer from initial confidence, the vertex features would be a $N \times (D + M)$ matrix, where M is the number of classes. When M is large, it introduces large computational cost and memory demand.

References

1. Chen, J., Ma, T., Xiao, C.: Fastgcn: fast learning with graph convolutional networks via importance sampling. arXiv preprint arXiv:1801.10247 (2018)
2. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: Advances in Neural Information Processing Systems. pp. 1024–1034 (2017)
3. Kemelmacher-Shlizerman, I., Seitz, S.M., Miller, D., Brossard, E.: The megaface benchmark: 1 million faces for recognition at scale. In: CVPR (2016)
4. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
5. Qiu, H., Xiao, C., Yang, L., Yan, X., Lee, H., Li, B.: Semanticadv: Generating adversarial examples via attribute-conditional image editing. arXiv preprint arXiv:1906.07927 (2019)
6. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
7. Wu, F., Zhang, T., Souza Jr, A.H.d., Fifty, C., Yu, T., Weinberger, K.Q.: Simplifying graph convolutional networks. arXiv preprint arXiv:1902.07153 (2019)
8. Yang, L., Chen, D., Zhan, X., Zhao, R., Loy, C.C., Lin, D.: Learning to cluster faces via confidence and connectivity estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2020)
9. Yang, L., Zhan, X., Chen, D., Yan, J., Loy, C.C., Lin, D.: Learning to cluster faces on an affinity graph. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2298–2306 (2019)

10. Zhan, X., Liu, Z., Yan, J., Lin, D., Loy, C.C.: Consensus-driven propagation in massive unlabeled data for face recognition. In: ECCV (2018)
11. Zhu, X., Ghahramani, Z.: Learning from labeled and unlabeled data with label propagation. Tech. rep., Citeseer (2002)