EXPLORING HYPERGRAPH REPRESENTATION ON FACE ANTI-SPOOFING BEYOND 2D ATTACKS

Gusi Te, Wei Hu[*]*, Zongming Guo*

Wangxuan Institute of Computer Technology, Peking University, Beijing

{tegusi, forhuwei, guozongming}@pku.edu.cn

ABSTRACT

Face anti-spoofing plays a crucial role in protecting face recognition systems from various attacks. Previous model-based and deep learning approaches achieve satisfactory performance for 2D face spoofs, but remain limited for more advanced 3D attacks such as vivid masks. In this paper, we address 3D face anti-spoofing via the proposed Hypergraph Convolutional Neural Networks (HGCNN). Firstly, we construct a computation-efficient and posture-invariant face representation with only a few key points on hypergraphs. The hypergraph representation is then fed into the designed HGCNN with hypergraph convolution for feature extraction, while the depth auxiliary is also exploited for 3D mask anti-spoofing. Further, we build a 3D face attack database with color, depth and infrared light information to validate the proposed paradigm and overcome the deficiency of 3D face anti-spoofing data. Experiments show that our method achieves the state-of-the-art performance over widely used 3D databases as well as the proposed one under various tests.

Index Terms— 3D face anti-spoofing, hypergraph representation, hypergraph convolutional neural network

1. INTRODUCTION

Face recognition has been widely applied to a variety of areas, including access control systems, online payment and user authentication. Nevertheless, vulnerability exists in a large amount of systems that they sometimes fail to recognize fake faces, which may be used by attackers to hack the systems. This is also referred to as *face spoofing* [3], an attempt to deceive recognition systems with photos (print attack), videos (replay attack) or 3D masks.

Various methods have been proposed to address these attacks, *i.e.*, *face anti-spoofing*. Previous model-based approaches make use of RGB images or sequences as the input and feed hand-crafted features such as LBP features into classifiers, which lack 3D information. With the development of deep learning, Convolutional Neural Network (CNN) becomes a powerful tool for feature extraction, and is thus leveraged for face anti-spoofing [14]. Several approaches have achieved promising performance via CNN and other techniques such as optical flow analysis [3]. Depth is a kind of remarkable cue for recognizing 2D attacks such as print and replay attacks. Atoum *et al.* propose to estimate the depth of a face image via a fully convolutional network, which is then utilized for anti-spoofing [2]. However, both methods leverage depth for detecting 2D attacks, while 3D mask attacks remain to be addressed [6]. 3D face anti-spoofing is challenging because 1) the color and texture of 3D masks are often so vivid that it is difficult to detect from RGB information; 2) the 3D structure is reasonable under non-front face postures, which is nontrivial to detect from depth information.

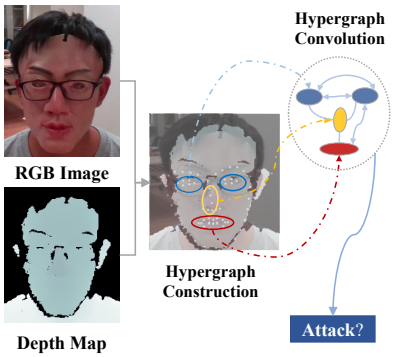

Fig. 1. Illustration of the proposed HGCNN architecture for 3D face anti-spoofing, which takes RGB and depth images as the input and generates classification scores as the output.

To this end, we propose to exploit hypergraph representation of 3D faces with RGB-D information to detect 3D mask attacks via Hypergraph Convolution Neural Networks (HGCNN). As real-time anti-spoofing systems require algorithms that are of *low complexity* and adapt to *various Human postures*, we propose *compact and posture-invariant hypergraph representation of faces*. Specifically, we propose compact face representation by only key points, including landmarks and a few interpolated points, which leads to remarkable reduction of both time and space complexity while re-

Corresponding author: Wei Hu (forhuwei@pku.edu.cn). This work was supported by National Natural Science Foundation of China [61972009], Beijing Natural Science Foundation [4194080], and National Natural Science Foundation of China under contract No. U1636206.

978-1-7281-1331-9/20/$31.00

taining most facial information. Because the key points tend to reside on irregular grids, we propose to represent them on a hypergraph, which models *high-order relationship* of samples via hyperedges, each of which links multiple samples so as to *capture the high-level feature of a local facial region*. Also, as hypergraphs describe the relative relationship instead of the absolute Euclidean distance among samples, the hypergraph representation is *invariant to postures*, *i.e.*, the relationship remains the same no matter how one moves the face. Hence, hypergraph-based facial representation is more robust to face motions or emotions compared with traditional image representation.

Thereafter, we take the hypergraph representation of faces as the input, and design a framework of HGCNN based on hypergraph convolution. As an extension to Graph Convolutional Neural Networks (GCNN) [4, 10], we take both data features and the aforementioned hypergraph instead of a simple graph as the input, and extract higher-order features via hypergraph graph convolution similar to [8]. Further, we exploit the depth auxiliary for 3D mask anti-spoofing, where depth maps share the same hypergraph representation as the RGB cue and then go through hypergraph convolution for feature learning. The extracted depth features are then concatenated with textural features as the final node features for classification.

To validate the proposed paradigm, we further build a 3D face attack database containing color, depth and Infrared ray (IR) data acquired from Intel RealSense SR300, which embodies more subjects and variations than prior 3D face databases. Extensive experiments demonstrate the superiority and robustness of our method on existing 3D databases and the proposed one.

2. RELATED WORK

Previous face anti-spoofing methods can be classified into spatial methods, temporal methods and fusion methods.

Spatial methods. Texture is a good hint for discriminating between real faces and fake ones, since print or replay attacks exhibit different textural characteristics from real faces. Li *et al.* are the first to take frequency distribution into consideration [12]. Other hand-crafted features are introduced to tackle face anti-spoofing, such as LBP [12] and HoG [11]. CNN-based anti-spoofing methods regard the problem as binary classification and extract features from texture images by traditional networks such as VGG [16].

Temporal methods. Several methods explore the potential of liveness detection from temporal sequences, including head movements and facial expressions. Pan *et al.* propose a straight-forward method that utilizes eye-blinking to detect whether the facial motion is authentic [17]. Besides, optical flow is introduced to analyze tiny expressions, which is essential to extract rigid movements of masks [3]. Edmunds *et al.* extract low-level motion features such as eye gazing and head pose from video clips to integrate high-level features [5].

Fusion methods. Leveraging on existing methods, some approaches combine texture and temporal cues, which achieve significant performance. Asim *et al.* propose a CNN-based spatial-temporal feature extraction framework for classification [1]. In [7], Feng *et al.* propose a multi-cue integration framework, including image quality, optical flow map, followed by a neural network classifying integrated features. Furthermore, Liu *et al.* exploit remote photoplethysmography (rPPG), a kind of signal reflecting facial liveness, and propose a neural network combining CNN and RNN to generate rPPG signals for liveness detection [14] [13].

3. BACKGROUND IN HYPERGRAPHS

A hypergraph $\mathcal{G} = \{\mathcal{V}; \mathcal{E}; \mathbf{W}\}$ consists of a vertex set $\mathcal{V}$, a hyperedge set $\mathcal{E}$ where each hyperedge e_i is assigned a weight $w(e_i)$, and a diagonal matrix of the hyperedge weights $\mathbf{W}$. Further, $\mathcal{G}$ can be represented by a $|\mathcal{V}| \times |\mathcal{E}|$ matrix $\mathbf{H}$, with entries $h(v; e) = 1$ if a vertex $v \in e$ and 0 otherwise, which is referred to as the incidence matrix of $\mathcal{G}$. Based on $\mathbf{H}$, the degree of each vertex $v \in \mathcal{V}$ is $d(v) = \sum_{e2E} w(e)\mathbf{H}(v; e)$, whereas the degree of each hyperedge $e \in \mathcal{E}$ is $(e) = \sum_{v2V} \mathbf{H}(v; e)$. For k-uniform hypergraphs considered in our context, the degrees of all the hyperedges are the same, *i.e.*, $(e_i) = k; \forall e_i \in \mathcal{E}$. We then let $\mathbf{D}_v$ and $\mathbf{D}_e$ denote the diagonal matrices containing the vertex and hyperedge degrees, respectively.

In the context of graph convolutional neural networks, we employ the normalized Laplacian in [18] because of its normalization property to avoid numerical instabilities, which is defined as

$$\mathcal{L} = \mathbf{I} - \mathbf{D}_v^{\frac{1}{2}} \mathbf{H} \mathbf{W} \mathbf{D}_e^{1} \mathbf{H}^{>} \mathbf{D}_v^{\frac{1}{2}}: \quad (1)$$

4. APPROACH

4.1. HGCNN Architecture

Fig. 2 illustrates the pipeline of the proposed framework. The input consists of RGB and depth images. Firstly, we extract landmarks from the RGB image and then augment the extracted landmarks for the purpose of denser point sets, which conduces to learning local features, as demonstrated in Fig. 3. Secondly, we construct a k-uniform hypergraph over the landmarks according to Euclidean distance of point pairs and compute the hypergraph Laplacian, which is shared by the input depth map. Then we feed the RGB and depth features of the landmarks into separate branches of hypergraph convolution, along with the computed hypergraph Laplacian for learning high-order textural and depth features respectively. Finally, we concatenate the extracted features and employ Multi-Layer Perceptron (MLP) to acquire the output classification scores.

4.2. Hypergraph Representation

Different from existing methods, our network takes graphs rather than images as the input, which means the hypergraph

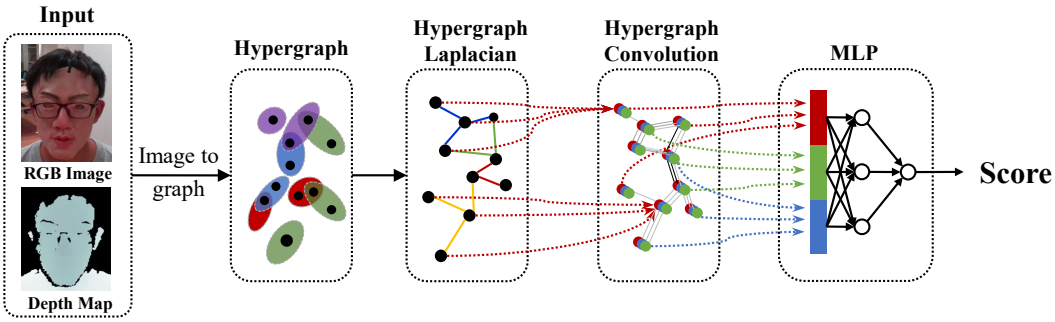

Fig. 2. The framework of our proposed HGCNN given a pair of RGB and depth images for 3D face anti-spoofing.

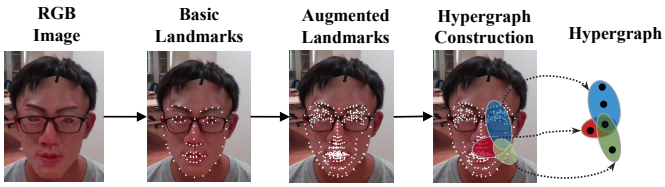

Fig. 3. Landmark-based hypergraph construction from a given RGB image.

structure is the key to the subsequent neural network. In graph-based image representation, a pixel is often treated as a vertex in the graph, and similar pixels are connected via edges. Due to the enormous amount of pixels in an image, it is extremely time-consuming to take every pixel as the input. Instead, we propose to construct a hypergraph based on limited number of facial landmarks, as illustrated in Fig. 2.

Landmark detection and augmentation. There are plenty of methods for face landmark extraction. In order to extract landmarks rapidly and robustly, we employ 68 landmarks for representation. If the landmarks are out of box or not detected, the current frame is neglected. Since the number of landmarks is not enough for feature extraction from neural networks, point augmentation is necessary. We thus propose to augment points by interpolation of the detected landmarks.

We firstly calculate k-nearest-neighbors of each landmark, and add the midpoint of each neighboring pair to the point set as augmentation. The distance metric between a pair of points $\{i; j\}$ is defined as the Euclidean distance, *i.e.*, $a_{i;j} = \|\mathbf{p}_i - \mathbf{p}_j\|_2^2$, where $\mathbf{p}_i$ and $\mathbf{p}_j$ are the coordinates of point i and j respectively. As some midpoints might be overlapping, we eliminate redundant points, resulting in 250 interpolated points. Together with the original landmarks, we finally extract a total of 318 points with RGB and depth cues to represent each face for the subsequent processing. Note that, as the detected landmarks reside on irregular grids in general, all the extracted points are also irregular, which is nontrivial to represent via traditional signal representation.

Hypergraph Construction. In order to describe the high-order relationship of the extracted irregular landmarks, we propose to construct a hypergraph over each face. Specifically, we treat the extracted points as vertices on the hypergraph, and connect each point with its k-nearest-neighbors using a hyperedge, which leads to a $(k+1)$-uniform hypergraph. The weight of each hyperedge is assigned 1, as we assume all the hyperedges are of equal importance to characterize local facial regions.

4.3. Hypergraph Convolution

The core of HGCNN is hypergraph convolution, which is an extension to graph convolution. Unlike Euclidean data (*e.g.*, images or videos), it is difficult to define convolution over graphs/hypergraphs in the vertex domain, because a meaningful translation operator in the vertex domain is nontrivial to define due to the irregularity of vertices. As in [8], we start from filtering of hypergraph signals in the spectral domain, and then deploy Chebyshev approximation to reduce the computational complexity.

Spectral filtering of hypergraph signals. The convolution operator on a graph $*_G$ is first defined in the spectral domain [4], specifically in the Graph Fourier Transform (GFT) [9] domain. The GFT basis is the eigenvector matrix of the graph Laplacian matrix. We extend traditional GFT to hypergraphs. Specifically, different from the Laplacian of a simple graph, the Laplacian of a hypergraph originates from the characterization of the total variation in a graph signal $\mathbf{f}$ with respect to the hypergraph [19]:

$$\sum_{e \in E} \sum_{\{u,v\} \subseteq e} \frac{w(e)}{\delta(e)} \left(\frac{f(u)}{\sqrt{d(u)}} - \frac{f(v)}{\sqrt{d(v)}} \right)^2 = 2\mathbf{f}^\top \mathcal{L} \mathbf{f}; \quad (2)$$

where $\mathcal{L} = \mathbf{I} - \mathbf{D}_v^{-\frac{1}{2}} \mathbf{H} \mathbf{W} \mathbf{D}_e^{-1} \mathbf{H}^\top \mathbf{D}_v^{-\frac{1}{2}}$ is the normalized hypergraph Laplacian as in (1). (2) describes the variation between vertices that belong to the same hyperedge, which is weighted by the corresponding hyperedge weight and degree. As an extension to GFT, the goal of Hypergraph Fourier Transform (HGFT) is to build an orthogonal basis from the Laplacian so as to minimize the total variation. It can be derived that the eigenvectors of the Hypergraph Laplacian form

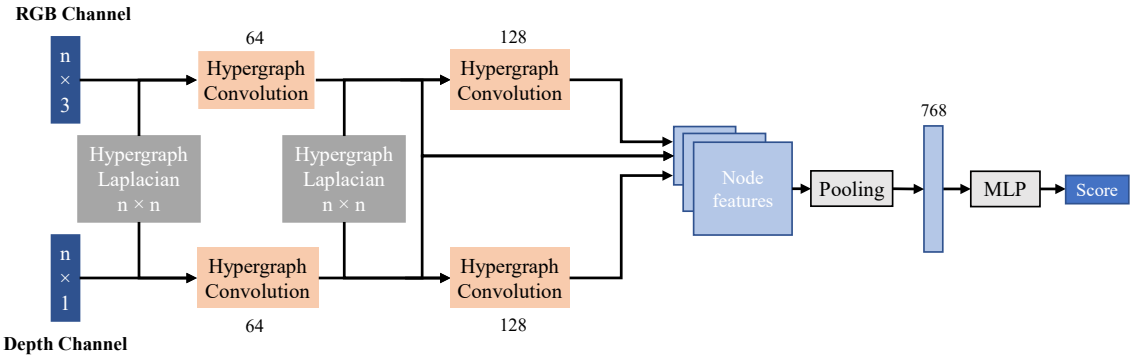

Fig. 4. The architecture of the proposed HGCNN. Both RGB and depth channels are fed into two hypergraph convolution layers of (64; 128) hidden nodes. Multi-level features are concatenated and average pooled, resulting in a 768-dimensional vector.

such a set, therefore the HGFT basis $\mathbf{U}$ is exactly the eigenvector set of the Laplacian matrix. The HGFT of a graph signal $\mathbf{x}$ is thus defined as $\hat{\mathbf{x}} = \mathbf{U}^\top \mathbf{x}$, and the inverse HGFT follows as $\mathbf{x} = \mathbf{U}\hat{\mathbf{x}}$.

Hence, the convolution between two hypergraph signals $\mathbf{x}$ and $\mathbf{y}$ can be defined as the multiplication of the corresponding HGFT coefficients, followed by the inverse HGFT, *i.e.*,

$$\mathbf{x} *_G \mathbf{y} = \mathbf{U}(\mathbf{U}^\top \mathbf{x}) \odot (\mathbf{U}^\top \mathbf{y}); \tag{3}$$

where $\odot$ is the element-wise Hadamard product. Then the spectral filtering of a graph signal $\mathbf{x}$ by g is

$$\mathbf{y} = g(\mathcal{L})\mathbf{x} = g(\mathbf{U} \quad \mathbf{U}^\top)\mathbf{x} = \mathbf{U}g(\quad)\mathbf{U}^\top \mathbf{x}; \tag{4}$$

where is a diagonal matrix of the eigenvalues of $\mathcal{L}$.

Chebyshev approximation for fast filtering. Inspired by the truncated Chebyshev polynomials for the approximation of spectral filtering, we employ the first-order Chebyshev approximation of hypergraph convolution similar to [8], which is formulated as:

$$\mathbf{Y} = (\mathbf{I} - \mathbf{D}_v^{-\frac{1}{2}}\mathbf{H}\mathbf{W}\mathbf{D}_e^{-1}\mathbf{H}^\top\mathbf{D}_v^{-\frac{1}{2}})\mathbf{X} \quad ; \tag{5}$$

where $\mathbf{X} \in \mathbb{R}^{n \quad F_1}$ is the input F_1-dimensional data feature of n points. $\in \mathbb{R}^{F_1 \quad F_2}$ is a matrix of learnable weight parameters, and F_2 is the dimension of the output feature. We then obtain the learned features with respect to each vertex by employing a bias $\mathbf{b}$ and ReLU activation:

$$\mathbf{X}^0 = \mathrm{ReLU}((\mathbf{I} - \mathbf{D}_v^{-\frac{1}{2}}\mathbf{H}\mathbf{W}\mathbf{D}_e^{-1}\mathbf{H}^\top\mathbf{D}_v^{-\frac{1}{2}})\mathbf{X} \quad + \mathbf{b}); \tag{6}$$

The architecture of the proposed HGCNN is demonstrated in Fig. 4.

5. THE PROPOSED DATASET

In order to overcome the deficiency of 3D face data, we collect a 3D face anti-spoofing database, referred to as FA3D, with color, depth and IR information. The database contains 285 videos of 19 subjects recorded by Intel RealSense SR300. The videos include RGB videos of resolution 1920×1080, the corresponding aligned depth videos and IR videos of the same resolution. For each subject, we collect five sections, each corresponding to a different posture. In section 1, the subject blinks several times; in section 2 the subject moves horizontally and vertically; in section 3 the subject moves back and forth; in section 4 people are asked to make expressions such as smile; in section 5 the subject yaws within -15 to 15 . The attacks include all the aforementioned categories we aim to address: print attacks, replay attacks and 3D mask attacks. Print attacks are based on high-resolution photos printed by Canon LBP7100. Replay attacks originate from real video clips, replayed by Macbook Pro under the same environment. For mask attacks, we employ a unique 3D latex mask and let different people wear the mask and record videos.

6. EXPERIMENTS

6.1. Experimental Setup

We evaluate the proposed framework on multiple attack databases for generalizability, including 3DMAD [6] and the proposed FA3D. 3DMAD is the only existing publicly available 3D spoofing database, containing 17 subjects and 255 video clips with 3D masks from Thatsmyface.com. Each video is recorded by Kinect with resolution 640×480. For each database, we follow the training-validation-test protocol as described. In our experiments, we deploy the commonly used metrics, which are presented in [11].

6.2. Experimental Results

6.2.1. Ablation Study

In order to study advantages of different modules of the proposed HGCNN comprehensively, we design the following incomplete models. Model 1 is the model without hypergraphs, which is equivalent to the original model except the number of vertices in each hyperedge $k = 0$. Model 2 is our model with the depth channel removed, *i.e.*, only the RGB cue is taken as the input. In Model 3, we replace the hypergraph representation with simple complete graphs, where the weight of

Table 1. Ablation study of different models on 3DMAD

	TDR			
FDR	1%	5%	10%	20%
Model 1	54.8%	58.4%	69.9%	88.4%
Model 2	75.7%	80.5%	84.6%	93.3%
Model 3	86.3%	93.6%	97.8%	98.2%
Model 4	**97.8%**	**100%**	**100%**	**100%**

Table 2. Comparison with different methods on 3DMAD

Method	HTER
Erdogmus *et al.* (LBP) [6]	0.95%
Feng *et al.* (OFM) [7]	4%
Edmunds *et al.* (Motion) [5]	3.53%
Liu *et al.* (rPPG) [13]	4.22%
Menotti *et al.* (CNN) [16]	0%
HGCNN	**0%**

each edge is assigned as the exponential function of the Euclidean distance between two connected vertices i and j, *i.e.*, $w_{i;j} = \exp\{-\|\mathbf{c}_i - \mathbf{c}_j\|_2^2\}$, with $\mathbf{c}_i$ and $\mathbf{c}_j$ denoting the coordinates of i and j respectively. The hypergraph Laplacian is then replaced with the simple graph Laplacian accordingly as used in [10]. Model 4 is the proposed complete model.

We test these models on the 3DMAD database and follow the protocol in [6]. Specifically, since the documentation doesn't define the index of validation, we follow the leave-one-out-cross-validation (LOOCV) settings and calculate the average value. The results are reported in Tab. 1, where we calculate TDR at different FDR, the higher the better. The results validate the importance of the hypergraph representation and the depth cue.

6.2.2. Results on 3D datasets

3DMAD We test on 3DMAD evaluated by HTER, with results reported in Tab. 2. We follow the test protocol in [6], *i.e.*, selecting 8 subjects for training, 5 for validation, and 5 for testing. From the results we observe that our model achieves competitive performance compared with model-driven [7, 5, 13] and CNN-based methods [16]. The model-driven methods exploit motion (optical flow map (OFM), rPPG), texture (LBP) or multi-cue integration (HOOF), while the CNN-based one deploys the VGG-16 model and fine-tuning.

FA3D In order to evaluate more comprehensive performance on our dataset, we design different protocols for test. Protocol 1 is designed to test the generality in terms of unseen subjects, which means subjects appearing in the training data are absent in the testing and vice versa. Specifically, we randomly select 10 subjects as training data and 7 subjects for test. In protocol 2, we test the robustness to different postures in order to *validate the posture-invariance of the proposed representation*. We extract one posture from each subject in the training stage, and employ the other postures for testing.

Table 3. Comparison with different methods on FA3D

Method	APCER	BPCER	ACER	ACC
LBP+SVM [6]	10.1%	65.4%	38.1%	66.5%
FASNet [15]	0.7%	9.2%	4.5%	97.6%
Patch-based [2]	0.1%	0.3%	0.2%	99.6%
HGCNN	0.1%	1.6%	0.7%	99.6%

Table 4. Our results under different protocols on FA3D

Variation	HTER	ACER	ACC
Subjects	0.5%	0.7%	99.6%
Posture	0.6%	0.9%	99.3%
Attack Types	0.3%	0.2%	99.9%

In protocol 3, we exemplify the efficiency of depth data by splitting attack types. As listed in Tab. 4, our accuracy is high for all the protocols. In particular, with the depth auxiliary our model only misclassifies 46 out of 12000 true frames, *i.e.*, resulting in BPCER of 0.4%.

Furthermore, in order to *compare our method with CNN-based methods on 3D datasets*, we implement the state-of-the-art CNN-based method FASNet [15] and patch-based method [2] with RGB-only input, and conduct experiments on FA3D of protocol 1. As reported in Tab. 3, our method achieves competitive results over all the metrics. Also, note that BPCER is higher compared with APCER, which is due to the data skew—fake samples are 4 times more than real ones, *i.e.*, it is aimed to minimize false acceptance rate.

6.2.3. Visualization and Analysis

To interpret the graph structure more vividly, we visualize the latent feature space of different layers in Fig. 5, where darker colors represent smaller distance. We observe that point features in the input are not quite distinguished from each other, but after hypergraph convolution points tend to keep similar features with adjacent ones, especially in certain regions like the mouth, nose and eyes. Further, we list the time cost in Tab. 5 to validate our computation efficiency. Although our preprocessing is a bit time-consuming due to the keypoint extraction, the forward time is much less than FASNet, leading to less total time.

7. CONCLUSION

We exploit hypergraph representation of 3D faces with RGB-D information to detect 3D mask attacks via the pre-

Table 5. Time complexity of our method and FASNet

Method	Forward	Preprocessing	Total
HGCNN	23ms	56ms	79ms
FASNet [15]	67ms	32ms	97ms