Deep Tree Learning for Zero-shot Face Anti-Spoofing

Karen Lu, Siyuan Yao, Jingyuan Li

Background

What are some of the attacks?

Face anti-spoofing? Zero-Shot Face Anti-spoofing?

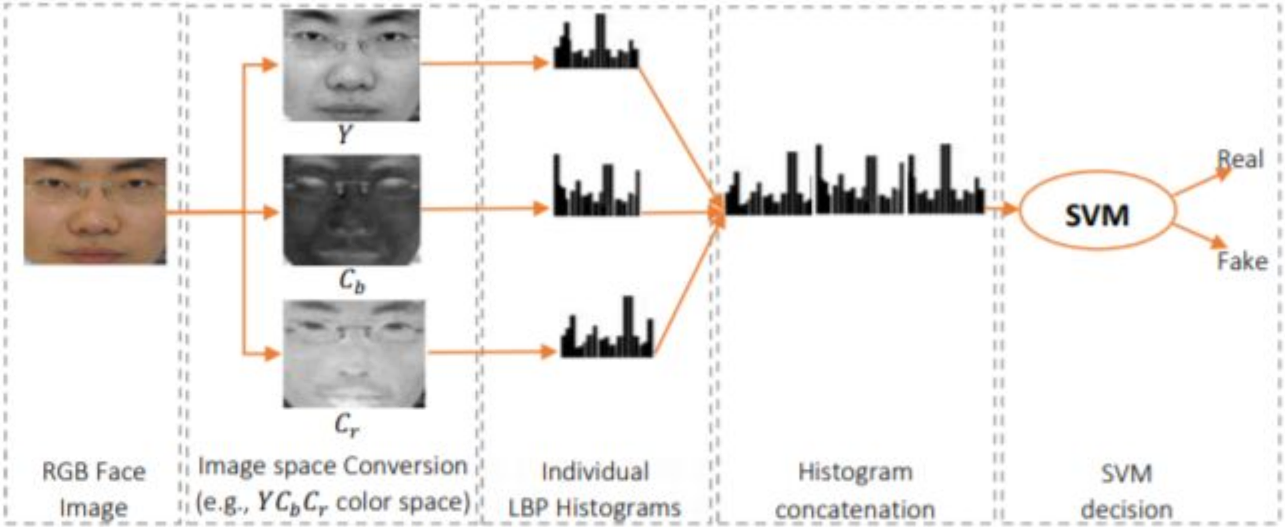
Face anti-spoofing - Designed to prevent face recognition systems from recognizing fake faces

Zero-Shot Face Anti-spoofing - detection of unknown spoof attacks

Unknown: never seen during the training

Prior ZSFA works:

handcrafted features → traditional classifiers → binary decision



Drawbacks:

Lacking spoof type variety — 2 types -> 13 types

No spoof knowledge — Semantic embedding

Limitation of feature selection — Hierarchical features

Deep Tree Network

Datasets

Table 1: Comparing our SiW-M with existing face anti-spoofing datasets.

Dataset	Year	Num. of subj./vid.	Face variations			Spoof attack types					Total num. of spoof types
			pose	expression	lighting	replay	print	3D mask	makeup	partial	
CASIA-FASD [50]	2012	50/600	Frontal	No	No	1	2	0	0	0	3
Replay-Attack [15]	2012	50/1, 200	Frontal	No	Yes	1	1	0	0	0	2
HKBU-MARs [30]	2016	35/1, 008	Frontal	No	Yes	0	0	2	0	0	2
Oulu-NPU [9]	2017	55/5, 940	Frontal	No	No	1	1	0	0	0	2
SiW [32]	2018	165/4, 620	$[-90°, 90°]$	Yes	Yes	1	1	0	0	0	2
SiW-M	2019	493/1, 630	$[-90°, 90°]$	Yes	Yes	1	1	5	3	3	13

Contributions:

- Conduct an extensive study of zero-shot face anti-spoofing on 13 different types of spoof attacks;
- Propose a Deep Tree Network (DTN) to learn features hierarchically and detect unknown spoof attacks;
- Collect a new database for ZSFA and achieve the state-of-the-art performance on multiple testing protocols.

Deep Tree Networks

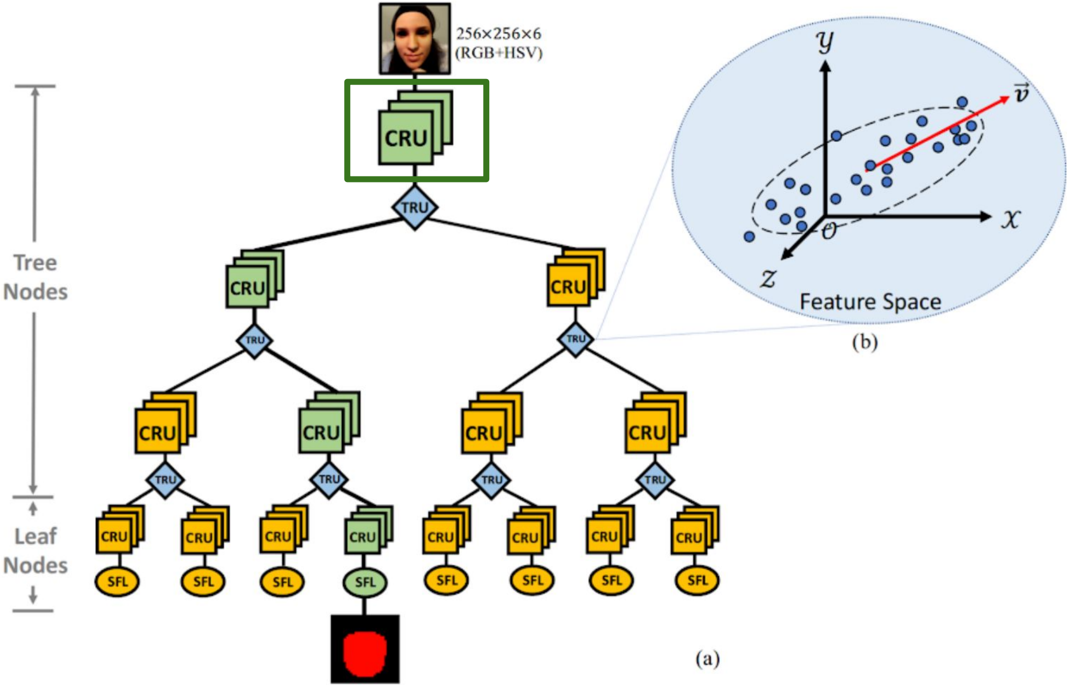
Deep Tree Network

Assumptions:

1. For each spoof type, we have homogenous features
2. Among different spoof types, there are distinct features

Goal

1. Discover semantic subgroups for known spoofs
2. Create a hierarchical structure to learn the features

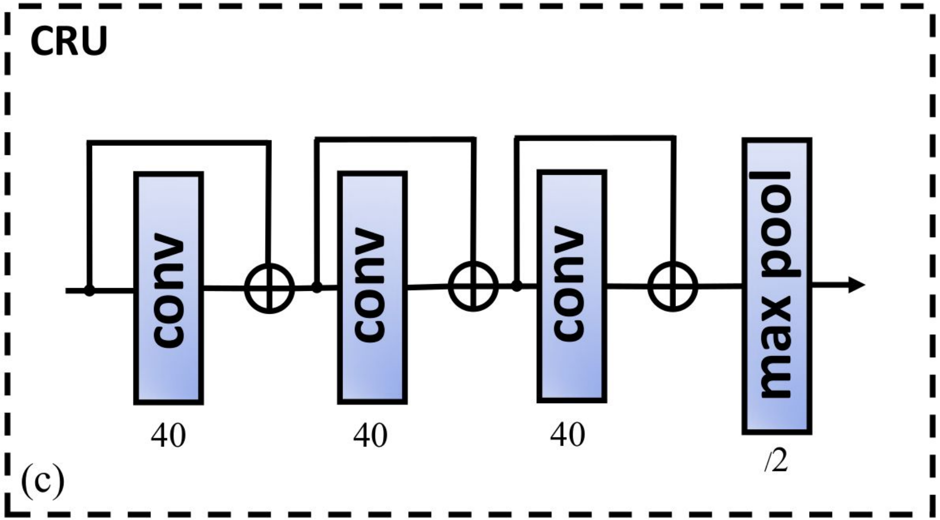


Convolutional Residual Unit

Image

$\mathbf{I} \in \mathbb{R}^{256\times256\times6}$

$I \in \mathbb{R}^{128*128*40}$



- conv layer is 3 x 3 x 40
- maxpool has stride 2

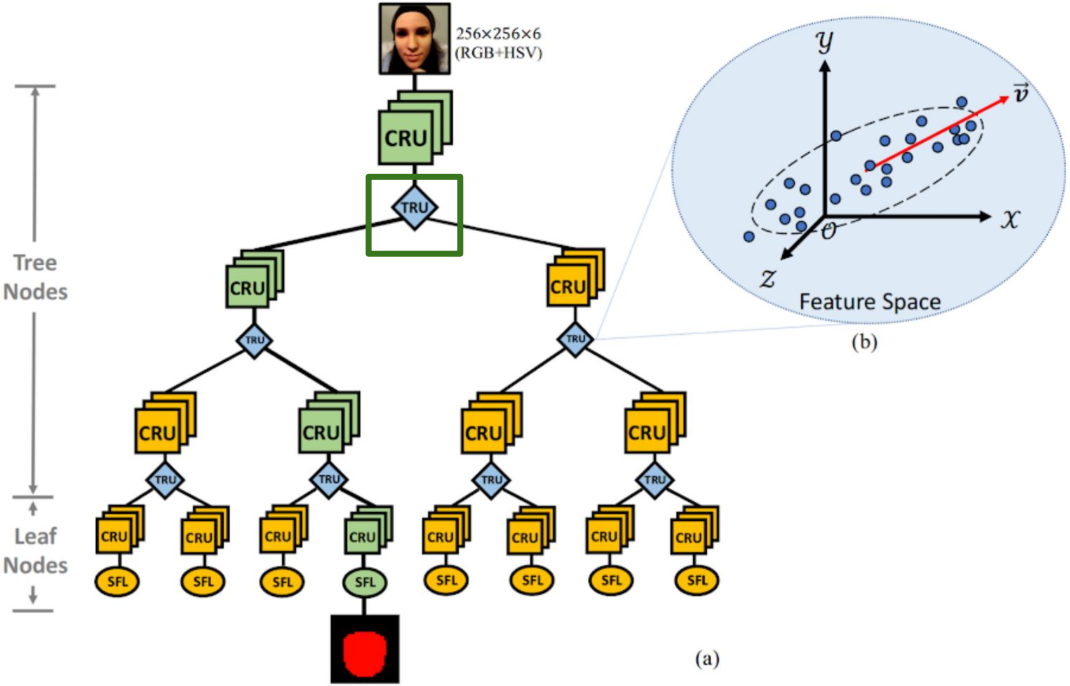
Deep Tree Network

Assumptions:

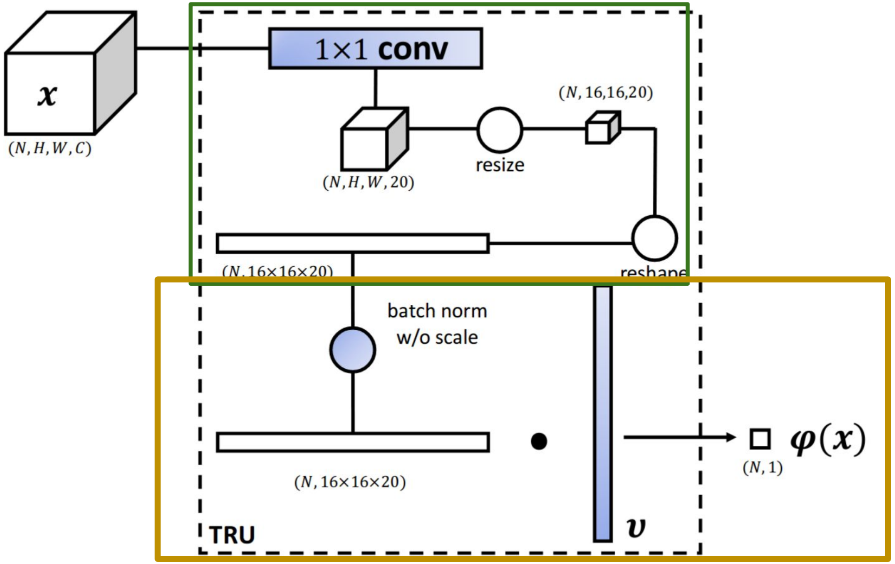
1. For each spoof type, we have homogenous features
2. Among different spoof types, there are distinct features

Goal

1. Discover semantic subgroups for known spoofs
2. Create a hierarchical structure to learn the features



Tree Routing Network



Step 1: Compression

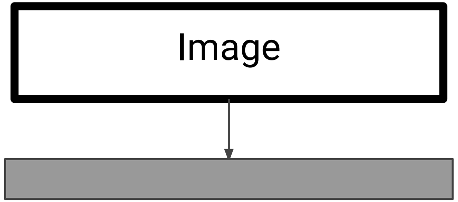
- reduces the computing burden
- 400GB ~ 0.1GB

Step 2: Routing Function

- batch norm

Tree Routing

Previous Work



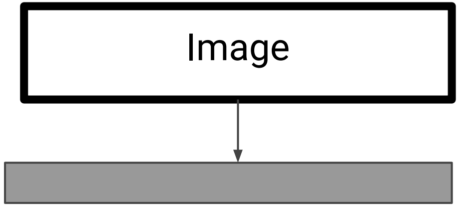
Routing Function

$$\varphi(\boldsymbol{x}) = \boldsymbol{x}^T \cdot \boldsymbol{v} + \tau,$$

dim = H x W x 6

$$\boldsymbol{x} = f(\mathbf{I} \,|\, \theta) \in \mathbb{R}^m$$

Contribution



Routing Function

$$\varphi(\boldsymbol{x}) = (\boldsymbol{x} - \underline{\boldsymbol{\mu}})^T \cdot \boldsymbol{v}, \quad \underline{\|\boldsymbol{v}\|} = 1,$$

PCA

Recap: Principal Components Analysis

Principal Components Analysis is a linear algebra method that given a data matrix **maps** the vectors into a new space which the direction of **highest variance** is extracted.

$$t_{k(i)} = \mathbf{x}_{(i)} \cdot \mathbf{w}_{(k)} \qquad \text{for} \qquad i = 1, \dots, n \qquad k = 1, \dots, l$$

$$\mathbf{w}_{(1)} = \arg\max_{\|\mathbf{w}\|=1} \left\{ \sum_i (t_1)^2_{(i)} \right\} = \arg\max_{\|\mathbf{w}\|=1} \left\{ \sum_i \left(\mathbf{x}_{(i)} \cdot \mathbf{w}\right)^2 \right\}$$

https://en.wikipedia.org/wiki/Principal_component_analysis

Contribution: Adding PCA

$$\varphi(\boldsymbol{x}) = (\boldsymbol{x} - \boldsymbol{\mu})^T \cdot \boldsymbol{v}, \quad \|\boldsymbol{v}\| = 1$$

set of data samples

$$\arg\max_{\boldsymbol{v},\theta} \lambda = \arg\max_{\boldsymbol{v},\theta} \boldsymbol{v}^T \underline{\bar{\boldsymbol{X}}}_{\mathcal{S}}^T \bar{\boldsymbol{X}}_{\boxed{\mathcal{S}}} \boldsymbol{v}.$$

demeaned data X

$$\mathcal{L}_{route} = \exp(-\alpha \boldsymbol{v}^T \bar{\boldsymbol{X}}_{\mathcal{S}}^T \bar{\boldsymbol{X}}_{\mathcal{S}} \boldsymbol{v}) + \beta \underbrace{\mathrm{Tr}(\bar{\boldsymbol{X}}_{\mathcal{S}}^T \bar{\boldsymbol{X}}_{\mathcal{S}})}_{\text{Regularizer}}$$

What data should we use for training the tree?

How do we leverage the existing data to train the spoof tree?

- use all spoof data to learn semantic subgroups of known spoofs
- use general data tree to learn spoof vs live data

Problems?

- Live tree does not convey semantic meaning and doesn’t help find the route
- General data may result in imbalanced subgroups → cause bias

Solutions against Bias

- Only use spoof samples to construct X_s
- Suppress the responses of live data to 0 (aka. Ignore live data when training routing function)
- Suppress the responses of spoof data that doesn't visit the node

$$\mathcal{L}_{uniq} = -\frac{1}{N} \sum_{\mathbf{I}_k \in \mathcal{S}} \boxed{\left\| \bar{\boldsymbol{x}}_k^T \boldsymbol{v} \right\|^2} + \frac{1}{N^-} \sum_{\mathbf{I}_k \in \mathcal{S}^-} \boxed{\left\| \bar{\boldsymbol{x}}_k^T \boldsymbol{v} \right\|^2} \qquad (6)$$

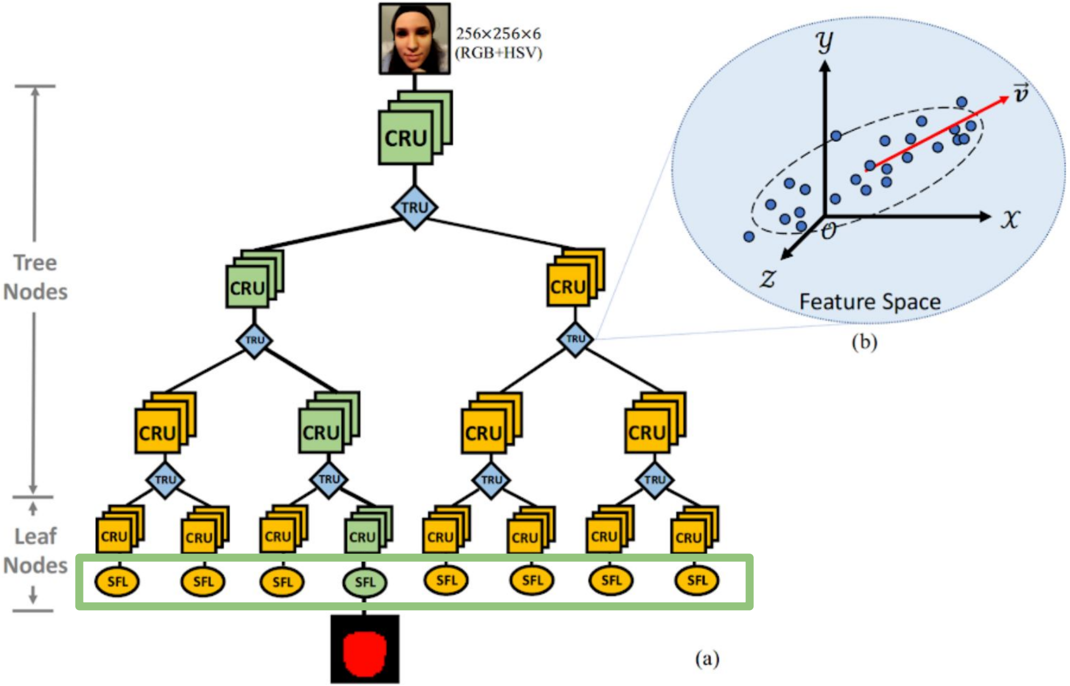
Deep Tree Network

Assumptions:

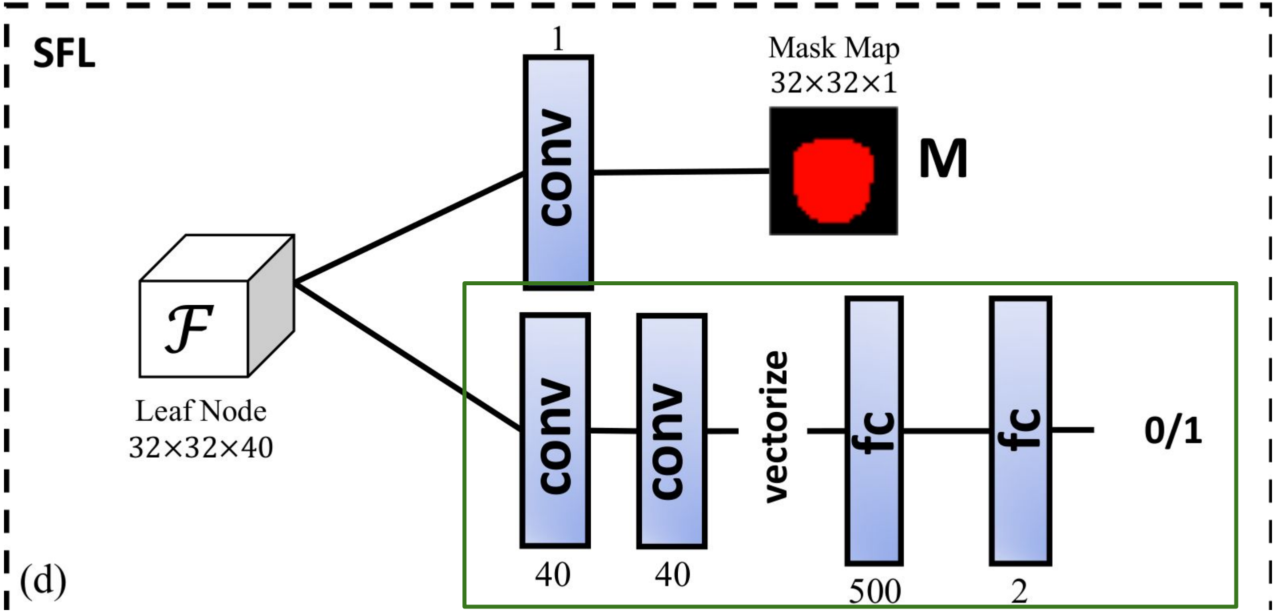
1. For each spoof type, we have homogenous features
2. Among different spoof types, there are distinct features

Goal

1. Discover semantic subgroups for known spoofs
2. Create a hierarchical structure to learn the features



Supervised Feature Learning (SFL)



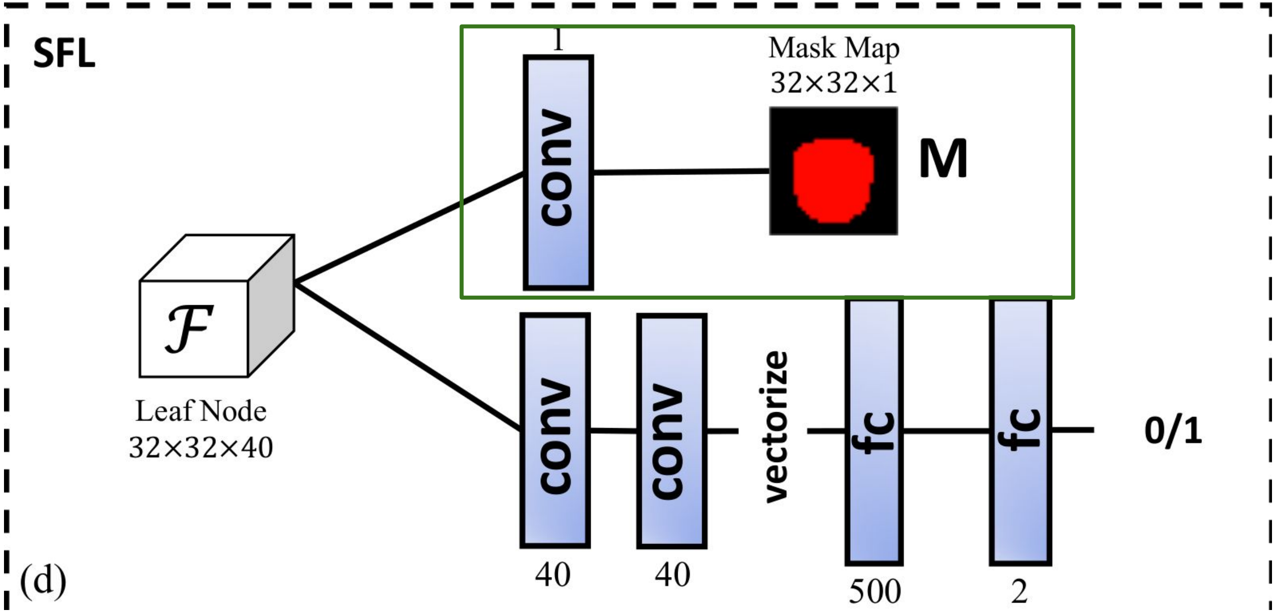
Classification Supervision

$$\mathcal{L}_{class} = \frac{1}{N} \sum_{I_k \in \mathcal{S}} \left\{ (1 - y_k)\log(1 - p_k) - y_k \log p_k \right\}$$

$$p_k = \frac{\exp(\mathbf{w}_1{}^T \mathbf{c}_k)}{\exp(\mathbf{w}_0{}^T \mathbf{c}_k) + \exp(\mathbf{w}_1{}^T \mathbf{c}_k)},$$

$$\mathbf{c}_k \in \mathbb{R}^{500}$$

Supervised Feature Learning (SFL)



Pixel-wise Supervision

$$\mathcal{L}_{mask} = \frac{1}{N} \sum_{I_k \in \mathcal{S}} \| \boxed{\mathbf{M}_k} - \boxed{\mathbf{D}_k} \|_1$$

Binary Mask to Produce

Provided Binary Mask

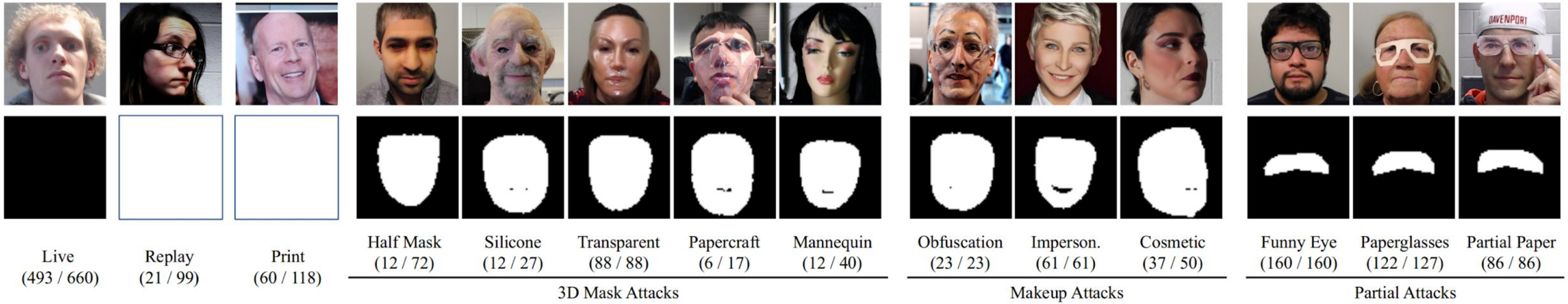
Putting it all Together

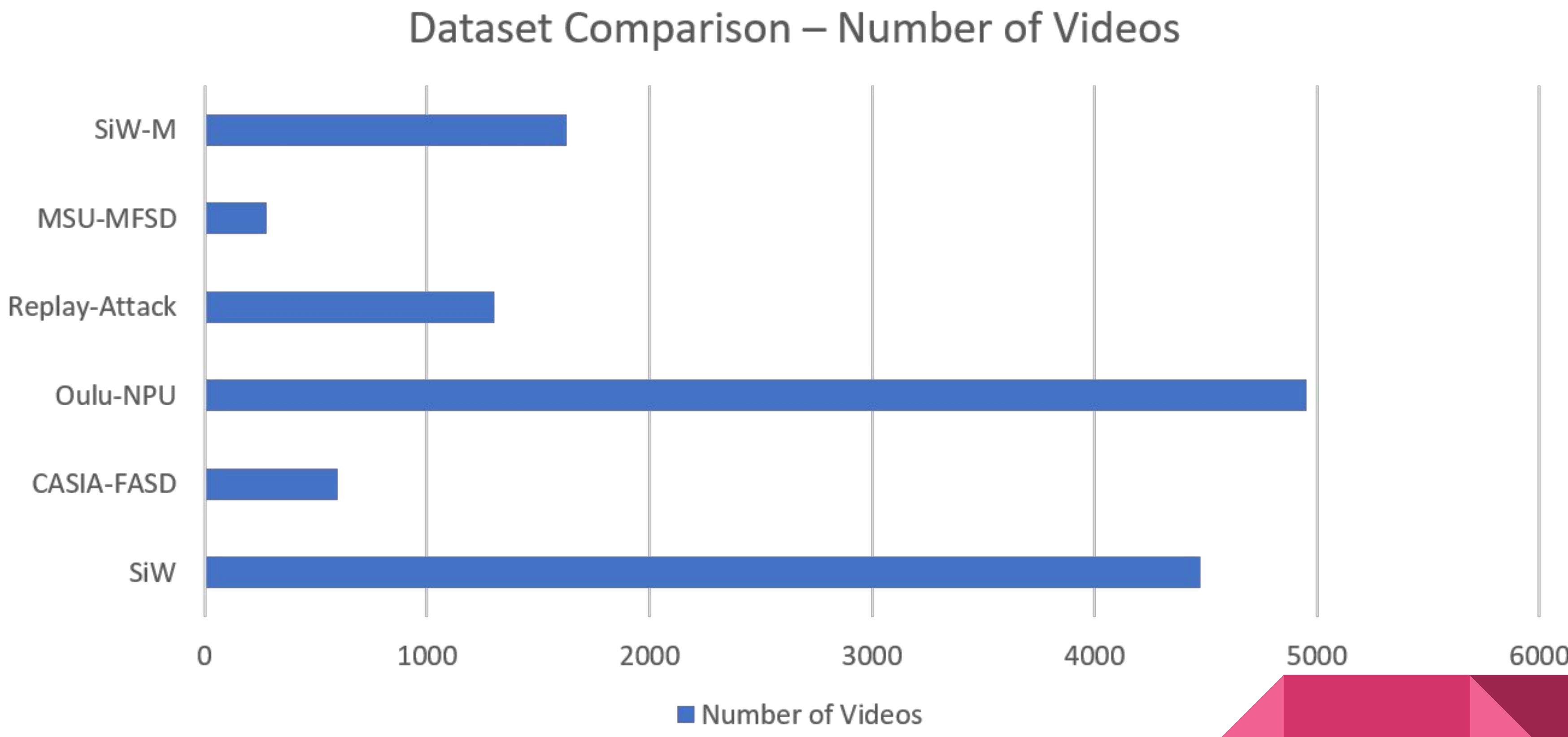
$$\mathcal{L} = \sum_{i=1}^{p}(\alpha_1\mathcal{L}^i_{class}+\alpha_2\mathcal{L}^i_{mask})+\sum_{j=1}^{q}(\alpha_3\mathcal{L}^j_{route}+\alpha_4\mathcal{L}^j_{uniq})$$

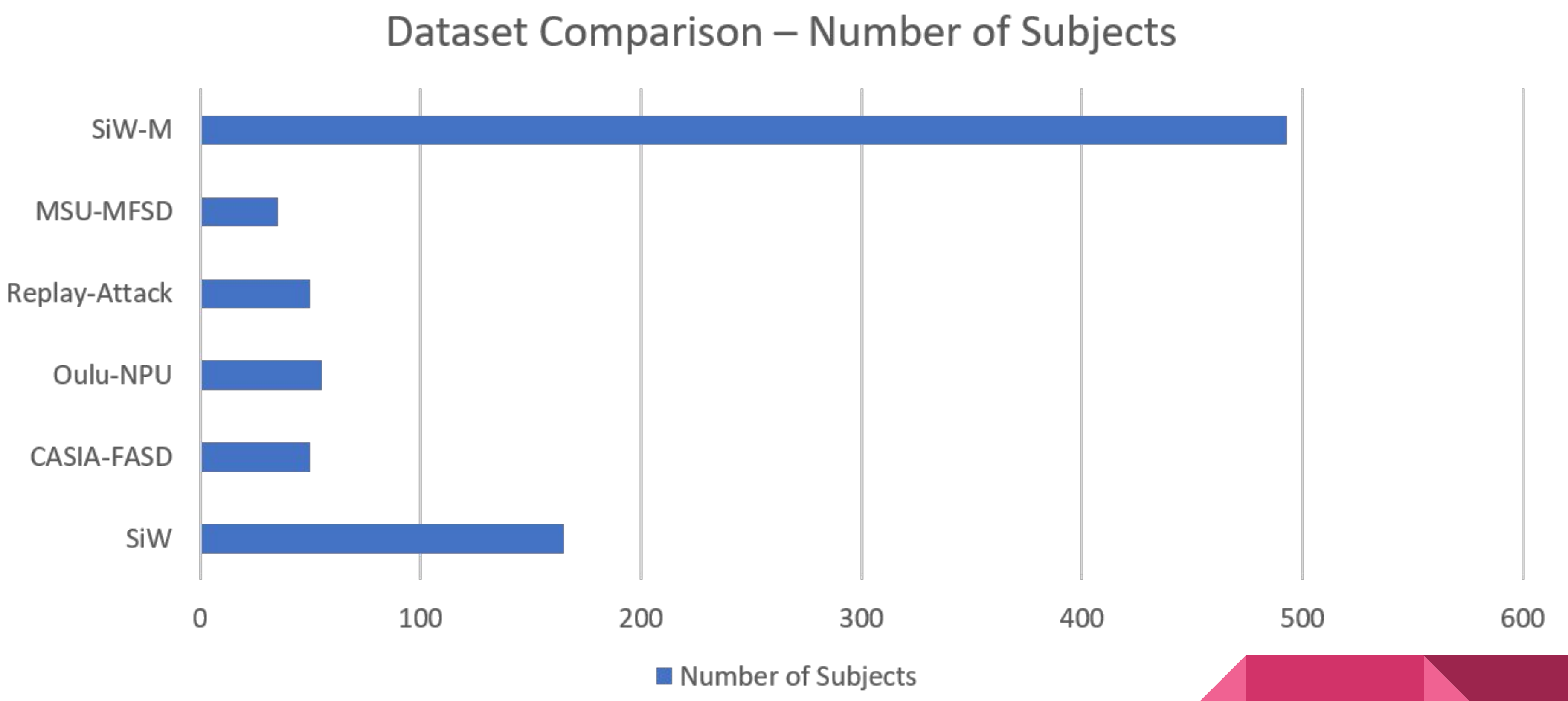
Spoof in the Wild Database

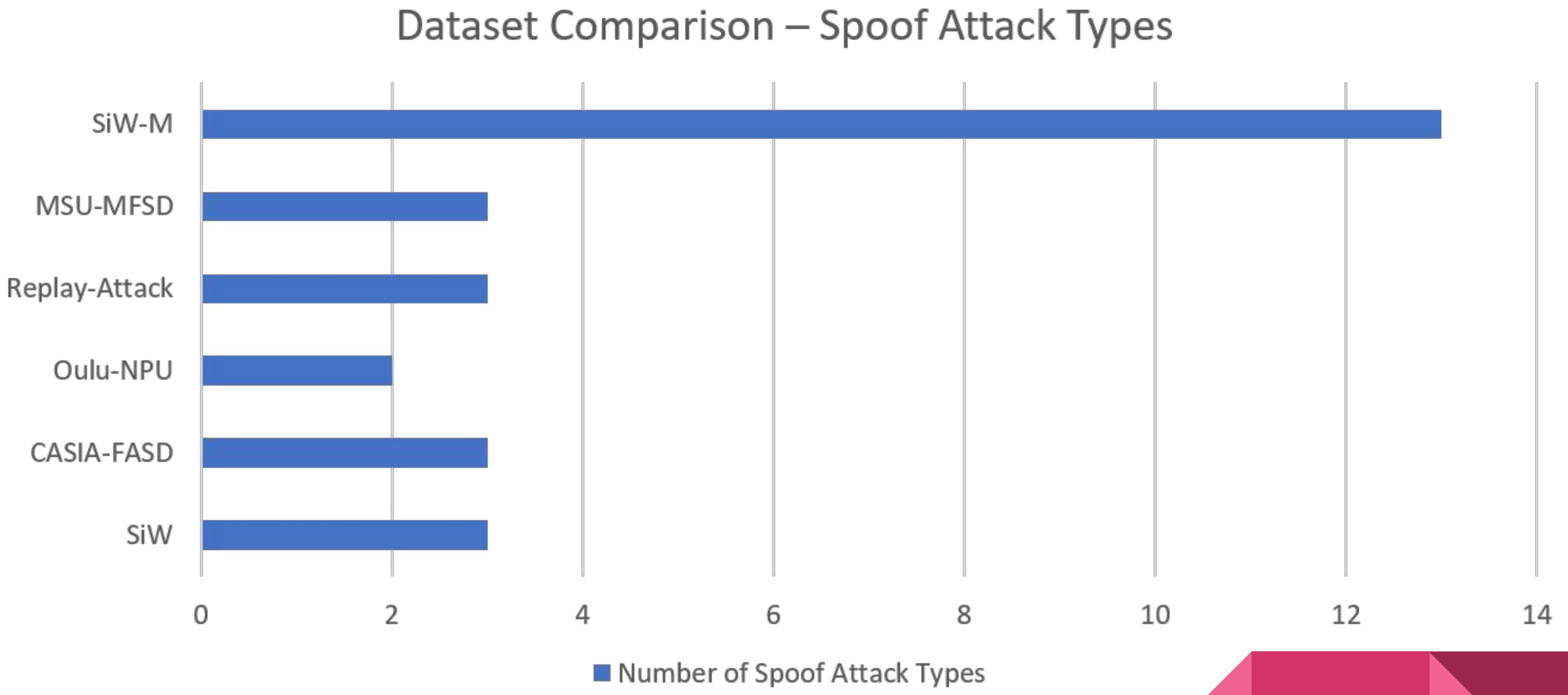
Database Composition

Live - 493 subjects, 660 videos
Spoof - 13 types, 968 videos




Dataset Comparison – Number of Videos
SiW-M
MSU-MFSD
Replay-Attack
Oulu-NPU
CASIA-FASD
SiW
0
1000
2000
3000
4000
5000
6000
Number of Videos


Dataset Comparison – Number of Subjects
SiW-M
MSU-MFSD
Replay-Attack
Oulu-NPU
CASIA-FASD
SiW
0
100
200
300
400
500
600
Number of Subjects


Dataset Comparison – Spoof Attack Types
SiW-M
MSU-MFSD
Replay-Attack
Oulu-NPU
CASIA-FASD
SiW
0
2
4
6
8
10
12
14
Number of Spoof Attack Types

Leave-one-out Test Protocol

- Training
  - 12 types of attacks
  - 80% of the live video
- Testing
  - 1 type of attacks
  - 20% of the live video

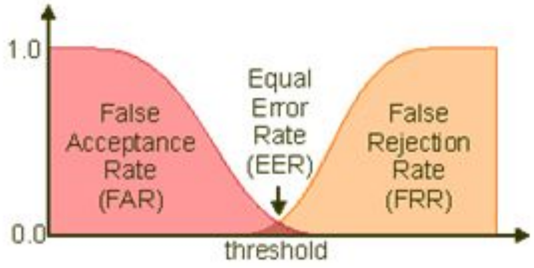
Experiment Setup and Results

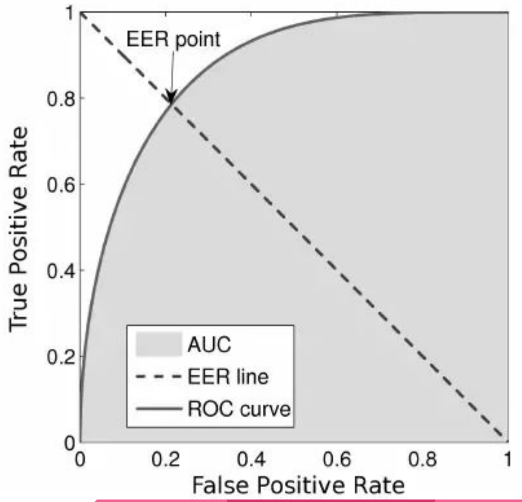
Experimental Setup

- Databases
  - SiW-M
  - CASIA
  - Replay-Attack
  - MSU-MFSD

Experimental Setup

- Metrics
  - APCER
    - Attack Presentation Classification Error Rate
      - False Acceptance Rate (FAR)
  - BPCER
    - Bona Fide Presentation Classification Error Rate
      - False Rejection Rate (FRR)
  - ACER
    - Average Classification Error Rate
  - EER
    - Equal Error Rate
  - AUC
    - Area Under Curve





Experimental Setup

- Parameter Setting
  - Constant learning rate - 0.001
  - Batch size - 32
  - 15 epochs
  - Randomized weights
    - 0 mean
    - 0.02 standard deviation

Ablation Study - Fusion Method

- Two values for final classification
  - Norm of the mask maps
  - Binary spoof scores
- Comparing ACER (lower is better)
  - Norm of the mask maps alone - 31.7%
  - Binary spoof scores alone - 20.5 %
  - Maximum of two - 21%
  - Average of two - 19.3%
- Result - Average of two performs the best

Ablation Study - Routing Function

Proving the necessity of routing function

Table 3: Compare models with different routing strategies.

Strategies	APCER	BPCER	ACER	EER
Random routing	37.1	**16.1**	26.6	24.7
Pick-one-leaf	51.2 ± 20.0	18.1 ± 4.9	34.7 ± 8.8	24.1 ± 3.1
Proposed routing function	**17.0**	21.5	**19.3**	**19.8**

Ablation Study - Loss Function

Showing the effect of route loss, and the unique loss

Methods	APCER	BPCER	ACER	EER
MPT [44] Limited routing	31.4	24.2	27.8	27.3
Live data $\surd$, Spoof data $\surd$, Unique Loss $\times$	**1.4**	73.3	37.3	31.2
Live data $\times$, Spoof data $\surd$, Unique Loss $\times$	70.0	12.7	41.3	44.8
Live data $\surd$, Spoof data $\surd$, Unique Loss $\surd$	54.2	**12.5**	33.4	36.2
Live data $\times$, Spoof data $\surd$, Unique Loss $\surd$	17.0	21.5	**19.3**	**19.8**

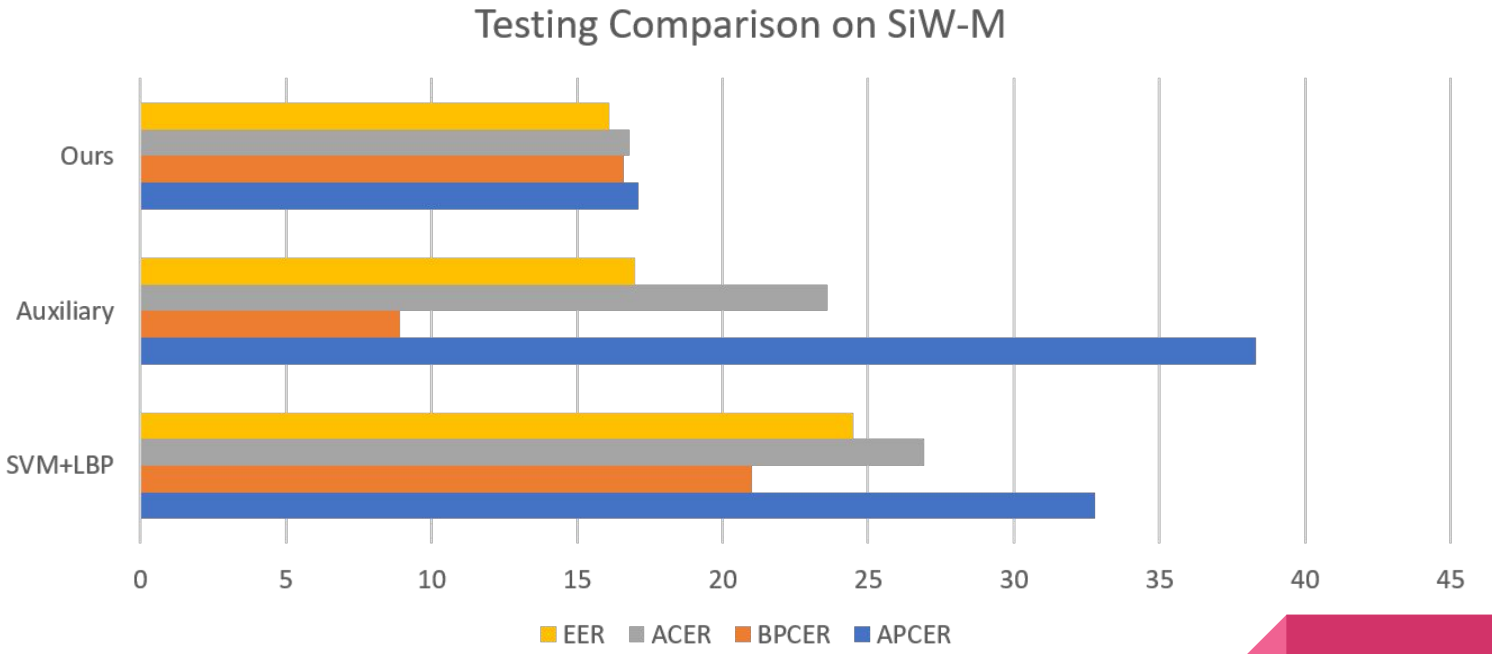
Testing - Existing Databases

Consistent and superior performance

Table 2: AUC (%) of the model testing on CASIA, Replay, and MSU-MFSD.

Methods	CASIA [50]			Replay-Attack [15]			MSU [42]			Overall
	Video	Cut Photo	Warped Photo	Video	Digital Photo	Printed Photo	Printed Photo	HR Video	Mobile Video	
OC-SVM$_{RBF}$+BSIF [3]	70.7	60.7	95.9	84.3	88.1	73.7	64.8	87.4	74.7	78.7 ± 11.7
SVM$_{RBF}$+LBP [9]	91.5	91.7	84.5	99.1	98.2	87.3	47.7	99.5	**97.6**	88.6 ± 16.3
NN+LBP [45]	**94.2**	88.4	79.9	99.8	95.2	78.9	50.6	99.9	93.5	86.7 ± 15.6
Ours	90.0	**97.3**	**97.5**	**99.9**	**99.9**	**99.6**	**81.6**	**99.9**	97.5	$\mathbf{95.9} \pm \boxed{\mathbf{6.2}}$

Testing - SiW-M



lower is better

Testing - SiW-M

Table 5: The evaluation and comparison of the testing on SiW-M.

Methods	Metrics (%)	Replay	Print	Mask Attacks					Makeup Attacks			Partial Attacks			Average
				Half	Silicone	Trans.	Paper	Manne.	Obfusc.	Imperson.	Cosmetic	Funny Eye	Paper Glasses	Partial Paper	
SVM$_{RBF}$+LBP [9]	APCER	19.1	15.4	40.8	20.3	70.3	**0.0**	4.6	96.9	35.3	**11.3**	53.3	58.5	0.6	32.8 ± 29.8
	BPCER	22.1	21.5	21.9	21.4	20.7	23.1	22.9	21.7	12.5	22.2	18.4	20.0	22.9	21.0 ± 2.9
	ACER	20.6	18.4	31.3	21.4	45.5	11.6	13.8	59.3	23.9	16.7	35.9	39.2	11.7	26.9 ± 14.5
	EER	20.8	18.6	36.3	21.4	37.2	7.5	14.1	51.2	19.8	16.1	34.4	33.0	7.9	24.5 ± 12.9
Auxiliary [32]	APCER	23.7	7.3	27.7	**18.2**	97.8	8.3	16.2	100.0	18.0	16.3	91.8	72.2	0.4	38.3 ± 37.4
	BPCER	**10.1**	**6.5**	**10.9**	**11.6**	**6.2**	**7.8**	**9.3**	11.6	**9.3**	**7.1**	**6.2**	**8.8**	**10.3**	$\mathbf{8.9 \pm 2.0}$
	ACER	16.8	6.9	19.3	**14.9**	52.1	8.0	12.8	55.8	13.7	**11.7**	49.0	40.5	**5.3**	23.6 ± 18.5
	EER	14.0	4.3	**11.6**	**12.4**	**24.6**	7.8	10.0	72.3	10.1	**9.4**	21.4	**18.6**	**4.0**	17.0 ± 17.7
Ours	APCER	**1.0**	**0.0**	**0.7**	24.5	**58.6**	0.5	**3.8**	**73.2**	**13.2**	12.4	**17.0**	**17.0**	**0.2**	$\mathbf{17.1 \pm 23.3}$
	BPCER	18.6	11.9	29.3	12.8	13.4	8.5	23.0	**11.5**	9.6	16.0	21.5	22.6	16.8	16.6 ± 6.2
	ACER	**9.8**	**6.0**	**15.0**	18.7	**36.0**	**4.5**	**7.7**	**48.1**	**11.4**	14.2	**19.3**	**19.8**	8.5	$\mathbf{16.8 \pm 11.1}$
	EER	**10.0**	**2.1**	14.4	18.6	26.5	**5.7**	**9.6**	**50.2**	**10.1**	13.2	**19.8**	20.5	8.8	$\mathbf{16.1 \pm 12.2}$

Analysis - Visualization of the Tree Routing

Analysis - Tree Routing Distribution

	Leaf 1	Leaf 2	Leaf 3	Leaf 4	Leaf 5	Leaf 6	Leaf 7	Leaf 8
Live	9	9	1	11	16	28	13	13
Replay	0	0	45	33	0	20	1	0
Print	0	3	11	78	0	6	2	0
Half Mask	3	6	24	4	0	21	42	0
Silicone Mask	0	3	20	17	0	54	5	0
Trans. Mask	0	0	0	0	0	43	57	0
Paper Mask	0	0	51	0	0	49	0	0
Manne	11	16	18	5	0	47	4	0
Ob. Makeup	0	4	14	0	0	79	3	0
Im. Makeup	0	73	6	0	0	14	6	0
Co. Makeup	8	0	8	10	4	59	10	1
Funny Eye	2	0	0	1	34	0	27	36
Paper Glasses	3	1	0	1	45	0	9	41
Partial Paper	91	0	0	0	8	0	0	0
Live (Test)	8	7	3	13	9	34	19	7

(a)

Print Model

	Leaf 1	Leaf 2	Leaf 3	Leaf 4	Leaf 5	Leaf 6	Leaf 7	Leaf 8
Live	35	32	9	4	7	7	2	4
Replay	0	2	15	0	35	47	0	1
Print	0	3	6	0	61	28	2	0
Half Mask	46	0	0	19	0	20	12	2
Silicone Mask	44	0	0	3	0	24	29	1
Trans. Mask	56	40	0	2	0	0	2	1
Paper Mask	1	0	0	46	0	38	15	0
Manne	1	1	0	56	0	15	27	1
Ob. Makeup	96	0	0	0	0	4	0	0
Im. Makeup	1	0	0	52	2	5	41	0
Co. Makeup	17	1	0	56	0	12	15	0
Funny Eye	23	40	31	1	1	1	1	3
Paper Glasses	11	74	7	0	2	1	2	3
Partial Paper	1	0	7	0	1	2	25	63
Live (Test)	44	29	3	1	8	8	2	6

(b)

Trans. Mask Model

Analysis - t-SNE Visualization

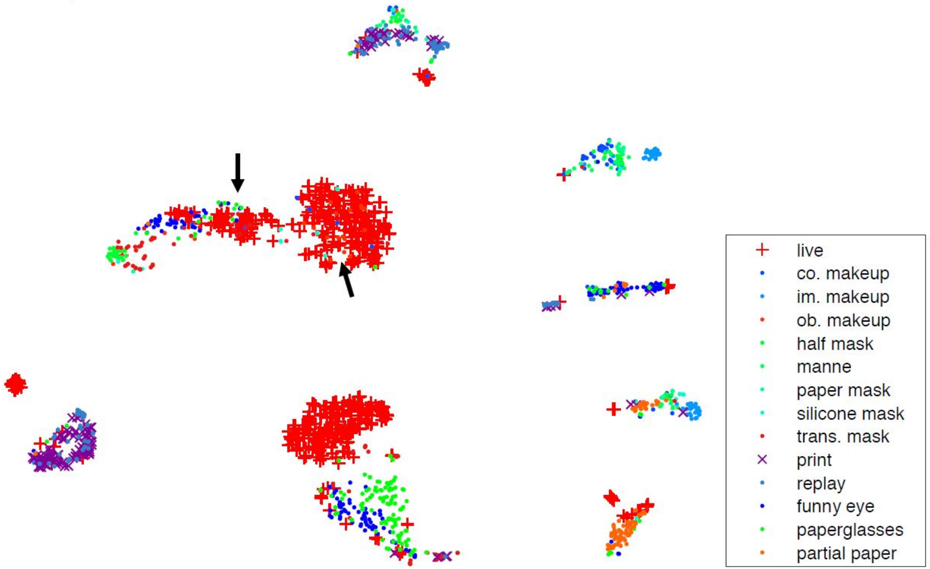


Figure 7: t-SNE Visualization of the DTN leaf features.

Future Development

Future Development

- Expand the size of SiW-M
- Expand the tree by adding more semantic sub-groups and tree layers