11th International Symposium on Intelligent Distributed Computing, Belgrade, Serbia, October 11-13, 2017

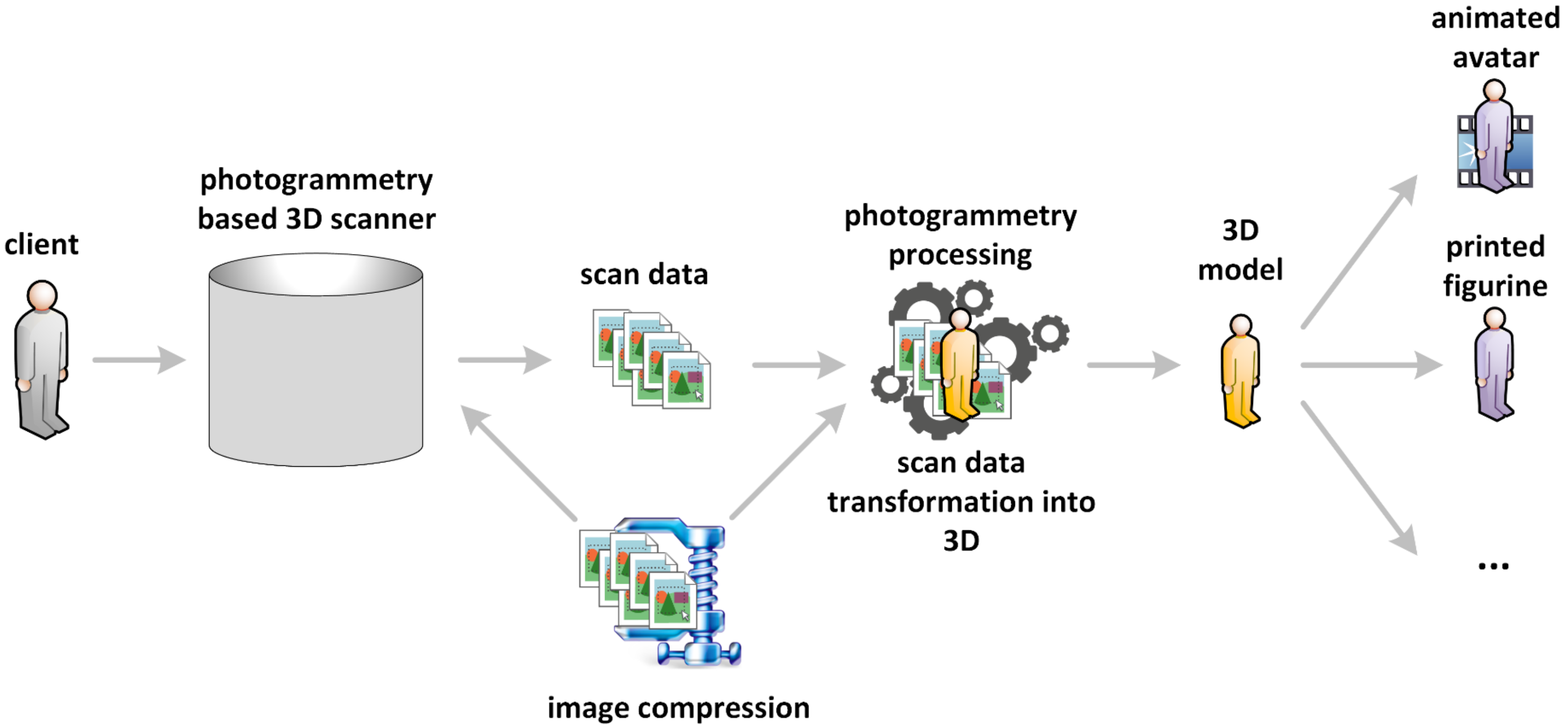
Binary Classification of Images for Applications in Intelligent 3D Scanning

Branislav Vezilić, Dušan Gajić, Dinu Dragan, Veljko Petrović, Srđan Mihić, Zoran Anišić, Vladimir Puhalac

Outline

Introduction

- 3D scanning process



Introduction

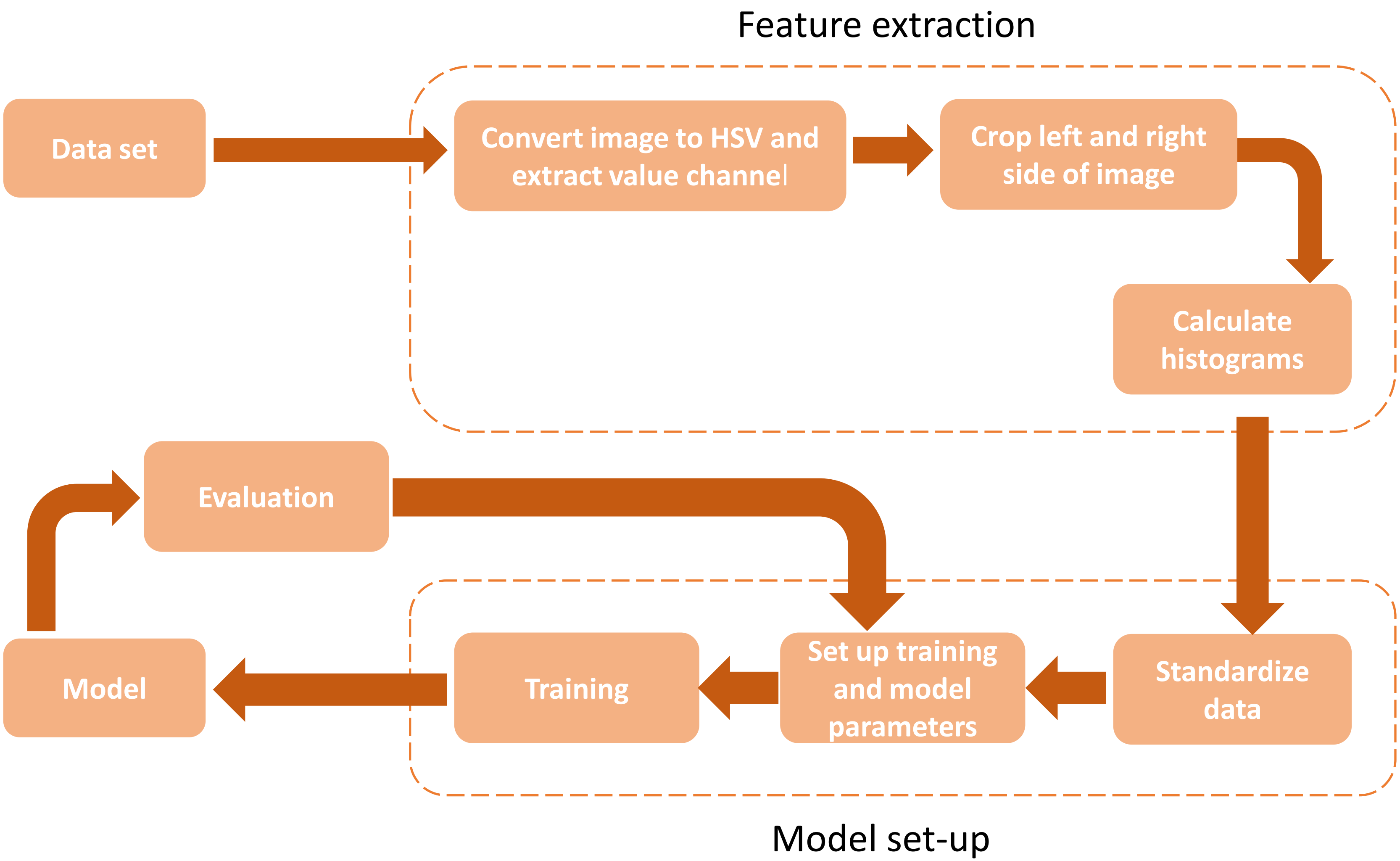
MISFIRED IMAGE

NORMAL IMAGE

Introduction

- Our goal is to create a system that is able to detect images with darker areas
- We differentiate two classes:
  - Normal images
  - Misfired images (as we like to call them)
- Our approach:
  - Manually extract features from images
  - Use them as input to selected machine learning algorithms

Overview of the Research Approach



Data Set

- Data set consists of 26,027 images with resolution of 5184 x 3456 in JPEG format
- All images are resized to resolution 400 x 266
- Only 2,729 out of 26,027 images (10.49%) are marked as misfired
- Balanced data set is produced by randomly sampling normal images from remaining data
- Balanced data set = equal number of samples per class
- Additional manual removal of outliers
- **FINAL SET:** 2,589 images spread across both classes

Feature Extraction

Convert image to HSV and extract value channel → **Crop left and right side of image** → **Calculate histograms**

Feature Extraction

Convert image to HSV and extract value channel → **Crop left and right side of image** → **Calculate histograms**

Color

Grayscale

Feature Extraction

Convert image to HSV and extract value channel → Crop left and right side of image → Calculate histograms

Color

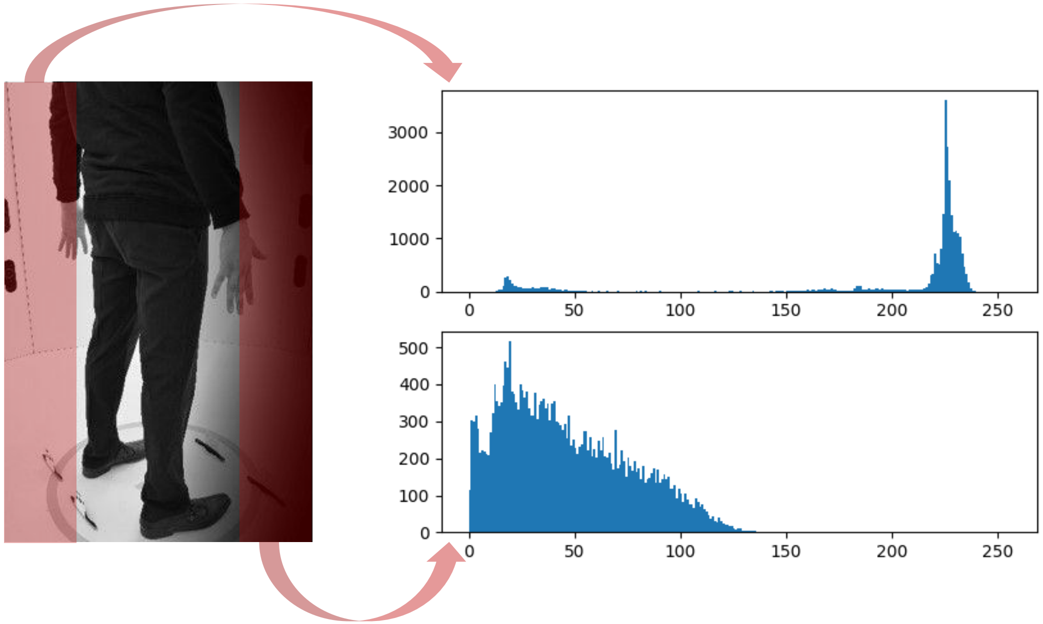
Grayscale

Analysed 4 different crop ratios:

1. 5%
2. 12.5%
3. 25%
4. 40%

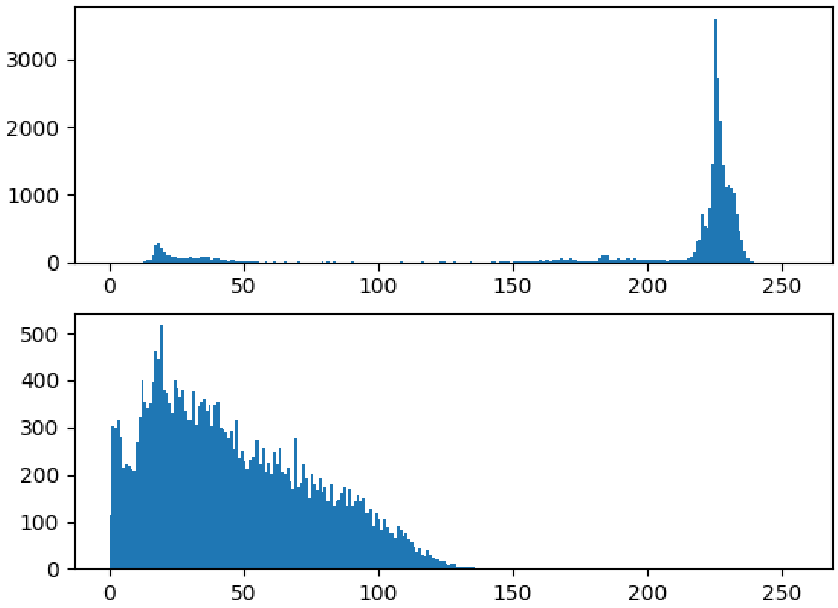
Feature Extraction

Convert image to HSV and extract value channel → Crop left and right side of image → **Calculate histograms**

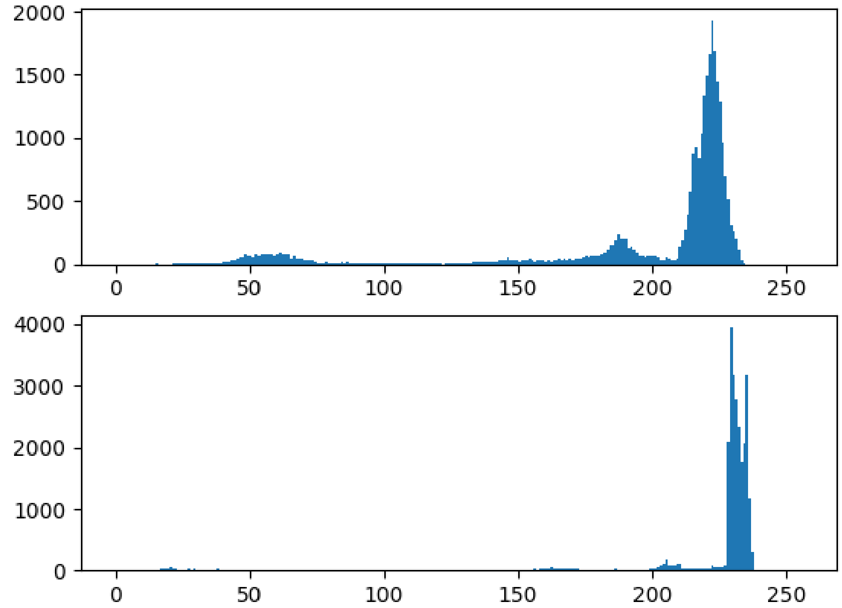


Feature Extraction

MISFIRED IMAGE

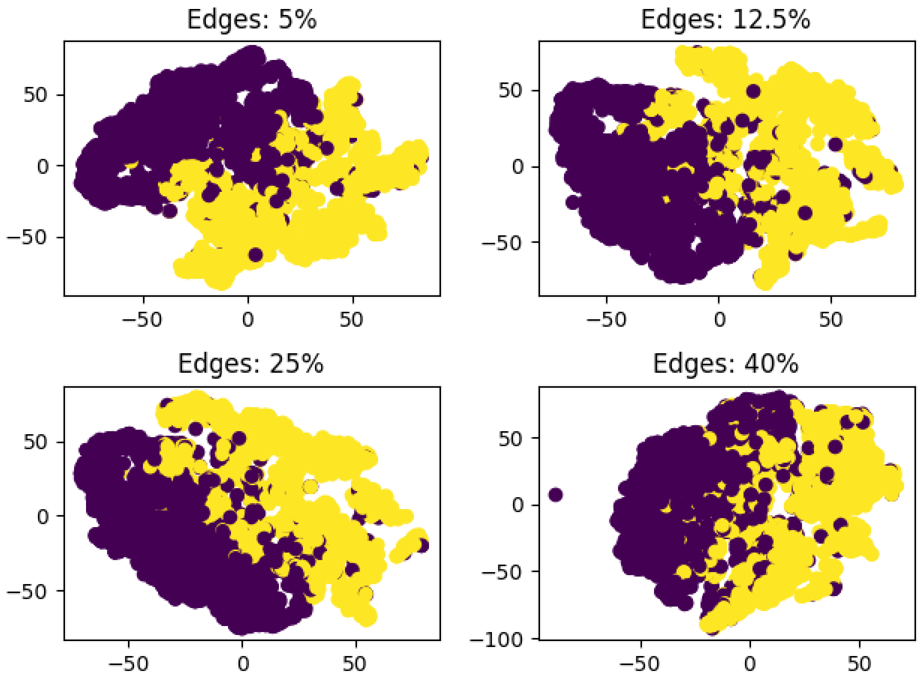


NORMAL IMAGE



t-SNE (t-distributed Stochastic Neighbor Embedding)

- Method for visualizing data set and distribution of classes among different features
- Cropped parts of image at different widths: 5%, 12.5%, 25%, 40%



Model Set-up

- 4 different machine learning methods:
  - k-Nearest Neighbors
  - Support Vector Machines
  - Random Forest
  - Artificial Neural Networks (Multi-layer perceptron)

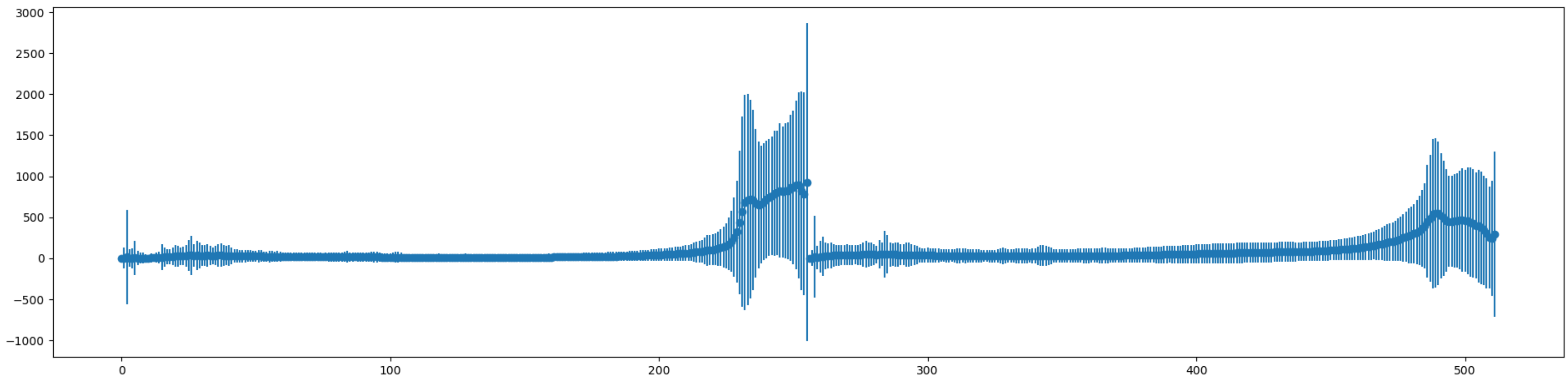
Model Set-up

Standardize data → Set up training and model parameters → Training

Model Set-up

Standardize data → **Set up training and model parameters** → **Training**

- Zero-centered by feature
- Scale by feature variance



Model Set-up

- **k-Nearest Neighbors**
  - Parameters: k-number of neighbors
- **Support Vector Machines**
  - Parameters: kernel, C, gamma
- **Random Forest**
  - Parameters: criteria, number of trees, max depth, minimal samples for split
- **Artificial Neural Network**
  - Parameters: network architecture, optimizer and learning rate, loss function

Model Set-up

Standardize data → Set up training and model parameters → **Training**

- **k-Nearest Neighbors**
  - Parameters: k-number of neighbors
- **Support Vector Machines**
  - Parameters: kernel, C, gamma
- **Random Forest**
  - Parameters: criteria, number of trees, max depth, minimal samples for split
- **Artificial Neural Network**
  - Parameters: network architecture, optimizer and learning rate, loss function

Evaluation

- F1-measure

$$precision = \frac{tp}{tp + fp}$$

$$recall = \frac{tp}{tp + fn}$$

$$F1 = 2 * \frac{precision * recall}{precision + recall}$$

- 11-fold cross validation
- Split on train and test set, 80-20 ratio

Results for k-Nearest Neighbors

- Parameter optimization was done using 11-fold cross validation
- Parameters:
  - k – number of neighbors
  - Euclidean distance

Edge[%]		5	12.5	25	40
k – neighbors	5	**93.69 (+/- 1.41)**	93.65 (+/- 1.29)	92.71 (+/- 2.15)	91.53 (+/- 1.95)
	10	93.17 (+/- 1.96)	93.40 (+/- 2.02)	92.73 (+/- 2.53)	**91.73 (+/- 1.69)**
	15	93.52 (+/- 1.10)	**93.76 (+/- 1.75)**	92.71 (+/- 1.66)	91.68 (+/- 2.20)
	35	92.91 (+/- 2.12)	93.18 (+/- 2.14)	**92.76 (+/- 1.79)**	91.07 (+/- 2.07)

Results for Support Vector Machines

- Parameter optimization was done using 11-fold cross validation
- Best parameters:
  - kernel function: rbf
  - C = 10
  - Gamma = 0.001

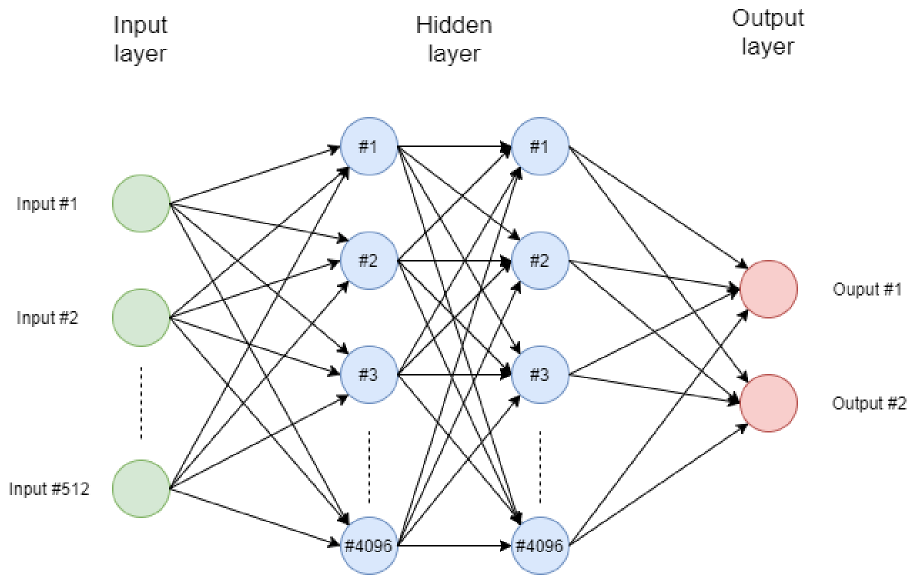
Edge [%]	**5**	**12.5**	**25**	**40**
Accuracy [%]	94.33 (+/- 1.47)	**94.42 (+/- 1.48)**	94.23 (+/- 1.76)	93.53 (+/- 1.61)

Results for Random Forest

- Parameter optimization was done using 11-fold cross validation
- Parameters:
  - criteria, number of trees, max depth, min samples for split

Edge[%]	5	12.5	25	40
Criteria	**gini**	gini	entropy	gini
Number of trees	**64**	512	64	512
Max depth	**512**	512	512	60
Min samples for split	**5**	2	2	2
Accuracy [%]	**94.55 (+/- 1.38)**	94.38 (+/- 1.40)	93.71 (+/- 1.82)	93.51 (+/- 0.95)

Results for ANN (Artificial Neural Network)



Network architecture

- Architecture same as classification block in well-known networks (VGG16, ResNet50, etc.)
- Dropout is increased to 0.75, from regular 0.5
- ReLU activation function in hidden layers
- Softmax function in output layer

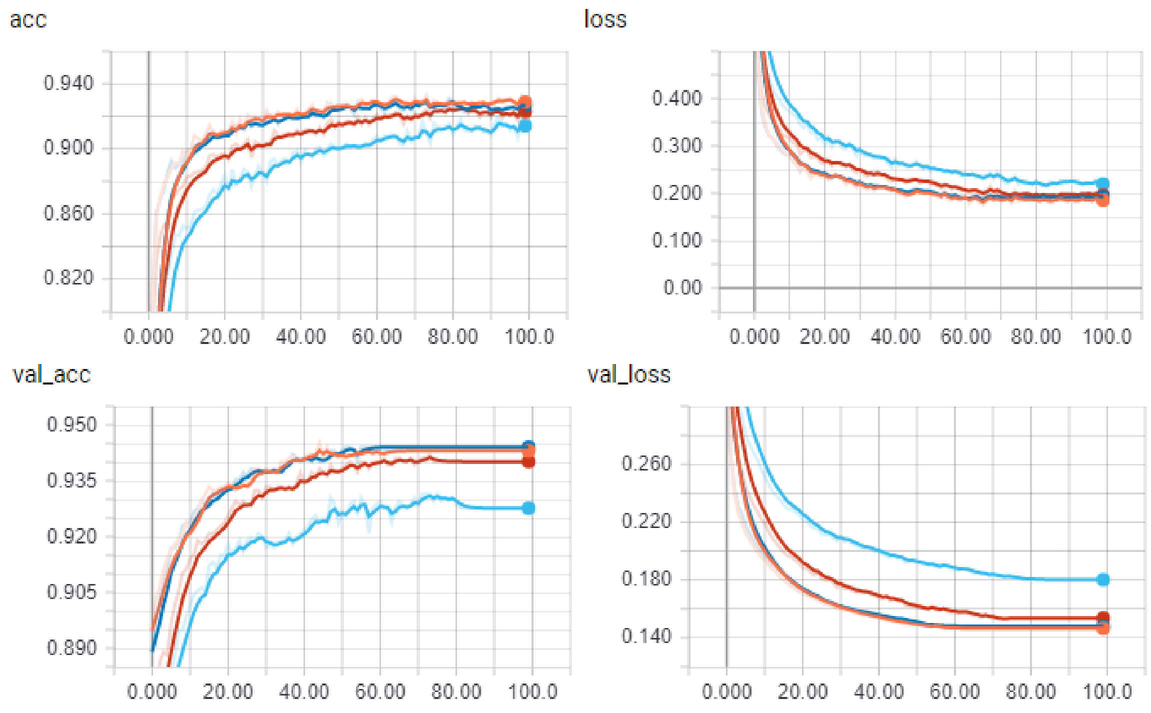
Results for ANN (Artificial Neural Network)

- SGD (Stohastic Gradient Descent)
  - learning rate = 0.0001
  - Momentum = 0.9
  - Nesterov = True
  - Reduce learning rate by factor of 0.1, if value loss haven’t been improved for 2 consecutive epochs
- Training
  - 100 epochs (~1.5h)
- Loss
  - Categorical cross entropy

Edge [%]	5	12.5	25	40
Accuracy [%]	94.47	**94.59**	94.36	93.04

Results for ANN (Artificial Neural Network)

- Training process



Edges: • 5% •12.5% •25% •40%

Final Results

Comparison of proposed methods on test data

Edge [%]	5	12.5	25	40
RF	**95.14**	94.76	94.96	93.60
ANN	94.47 (-0.67)	**94.59 (-0.17)**	94.36 (-0.60)	93.04 (-0.56)
SVM	**94.75 (-0.39)**	94.59 (-0.17)	94.23 (-0.73)	93.40 (-0.20)
k-NN	**94.58 (-0.56)**	94.56 (-0.20)	94.31 (-0.65)	93.40 (-0.20)

Conclusion

- Proposed a method for detecting misfired images
- Based on histograms as global descriptors and machine learning algorithms
- All 4 considered algorithms showed very good and comparable performance
- Random Forest showed best performance with 95.14% accuracy on edge area set to 5%
- **Possible improvements:**
  - Add additional feature descriptors
  - Clear and expand the data set
- **Future work:**
  - Final goal is development of complete quality control system for 3D scanning
  - System is expected to detect: misfiring, empty scene, blur
  - Reconstruction of missing 3D model body parts

11th International Symposium on Intelligent Distributed Computing, Belgrade, Serbia, October 11-13, 2017

Binary Classification of Images for Applications in Intelligent 3D Scanning

Branislav Vezilić, Dušan Gajić, Dinu Dragan, Veljko Petrović, Srđan Mihić, Zoran Anišić, Vladimir Puhalac

UNIVERSITAS STUDIORUM NEOPLANTENSIS MCMLX

doob innovation studio