

A decorative network diagram in the top-left corner, consisting of interconnected nodes and lines, some of which are highlighted in blue.

Children safety and privacy protection through facial characteristics censorship in photos

Alexandros Zerntev
Nikos Episkopos
Spyros Nikolakis

A decorative network diagram in the bottom-right corner, consisting of interconnected nodes and lines, some of which are highlighted in blue.

1. Introduction



Social media revolution (2010 -)

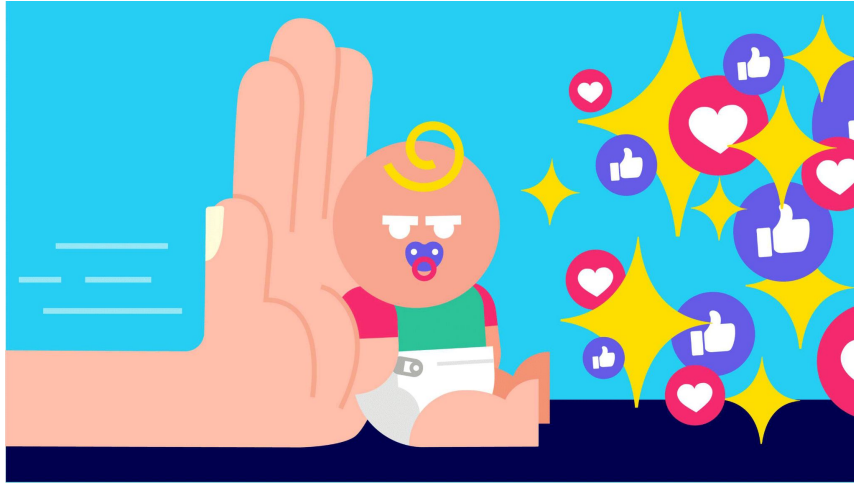
- ◎ Access from everywhere
- ◎ Write whatever you want
- ◎ Share whatever you want
- ◎ Meet new people
- ◎ Chat with friends
- ◎ New services are offered all the time



Privacy concerns



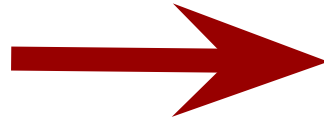
Family with young children online privacy & dangers



Family with young children online privacy & dangers



Photo censorship





2.

Dataset

Data collection

Data Description and Sources

Sources to create custom dataset:

- ◎ Gathered images from search engines
- ◎ Added personal photographs

Type of images:

- ◎ Random infants photographs found online
- ◎ Photographs of families with children

Examples of Photographs



Process of creating the dataset

3 datasets created (training, validation, testing)

Process:

1. Collect images
2. Extract faces from images as separate images
3. Manually classify the images of the extracted faces into two categories, placing each one in the respective directory (*babies* and *not-babies*)

A decorative network diagram in the top-left corner, consisting of a complex web of interconnected nodes and lines, rendered in a light gray color.

3.

Face Detection

Extracting faces from images

YOLOFace

- ◎ Deep learning based Face Detection in images
- ◎ Uses YOLOv3 real-time object detector
- ◎ Open-source software (MIT license, hosted on GitHub)
- ◎ Trained on “Wider Face” dataset:



Faces Extraction Example



→ **YOLOFace**



Other methods tried

- ◎ OpenCV Haar cascades
- ◎ OpenCV DNN Face Detector
- ◎ HOG + Linear SVM Face Detector
- ◎ MMOD + CNN Face Detector
- ◎ MTCNN

Did not work as well for us.



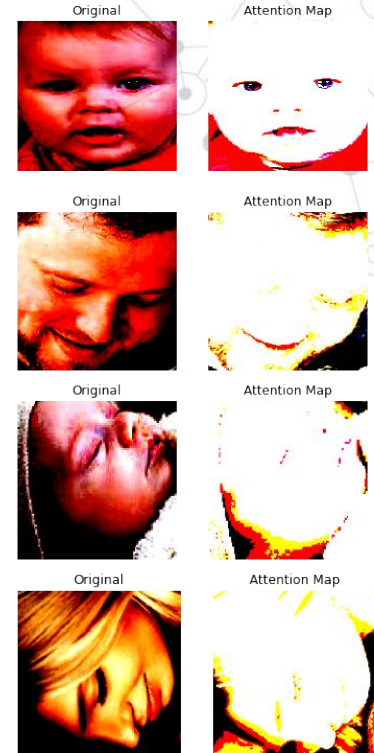
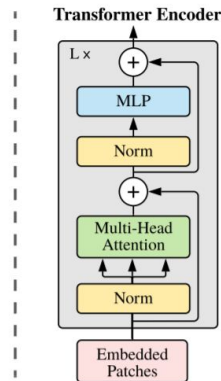
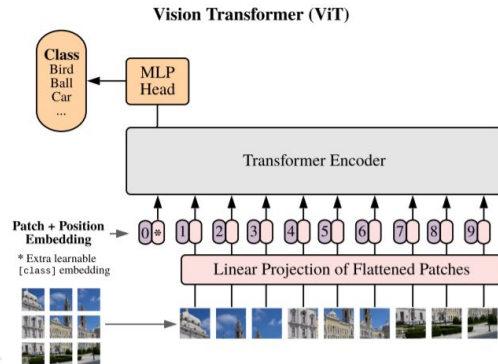
4.

Classification Models

Get a good classifier with a few data?

Vision Transformers

- Based on Transformers from NLP
- Pre Trained on *imagenet21k*
- “Understands” context
- Does not assume locality of features
- Generalizes well even with small dataset



VGGFace2

- ◎ Task-specific pretrained model *ResNet-50*
- ◎ 3M images from 9000+ people from all over the world
- ◎ 2 uses
 - Feature extraction and then classification
 - Fine tuning

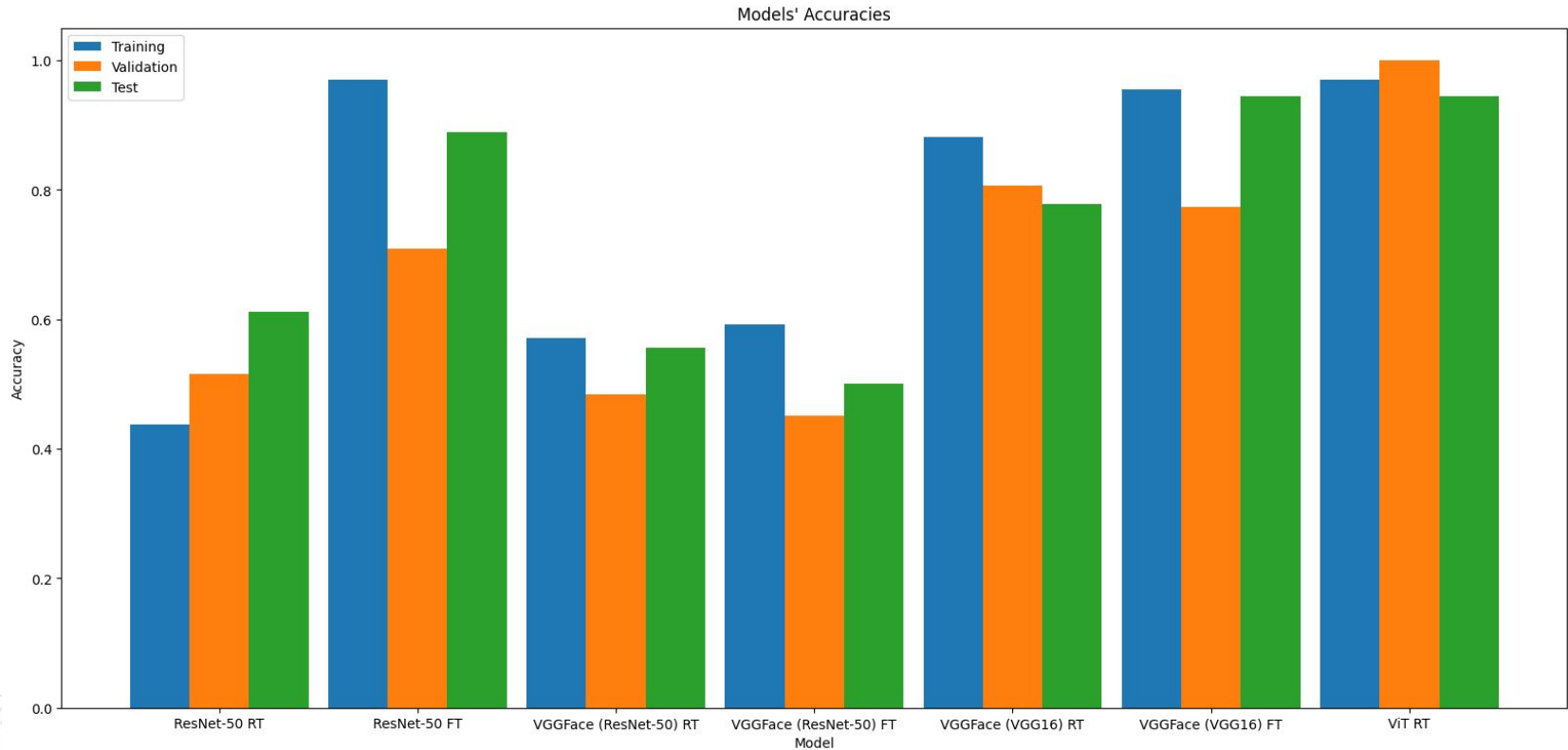


A decorative network diagram in the top-left corner, featuring a complex web of interconnected nodes and lines. The nodes are represented by small circles, some of which are larger and have concentric circles, suggesting a hierarchical or multi-layered structure. The lines are thin and gray, connecting the nodes in a non-linear fashion.

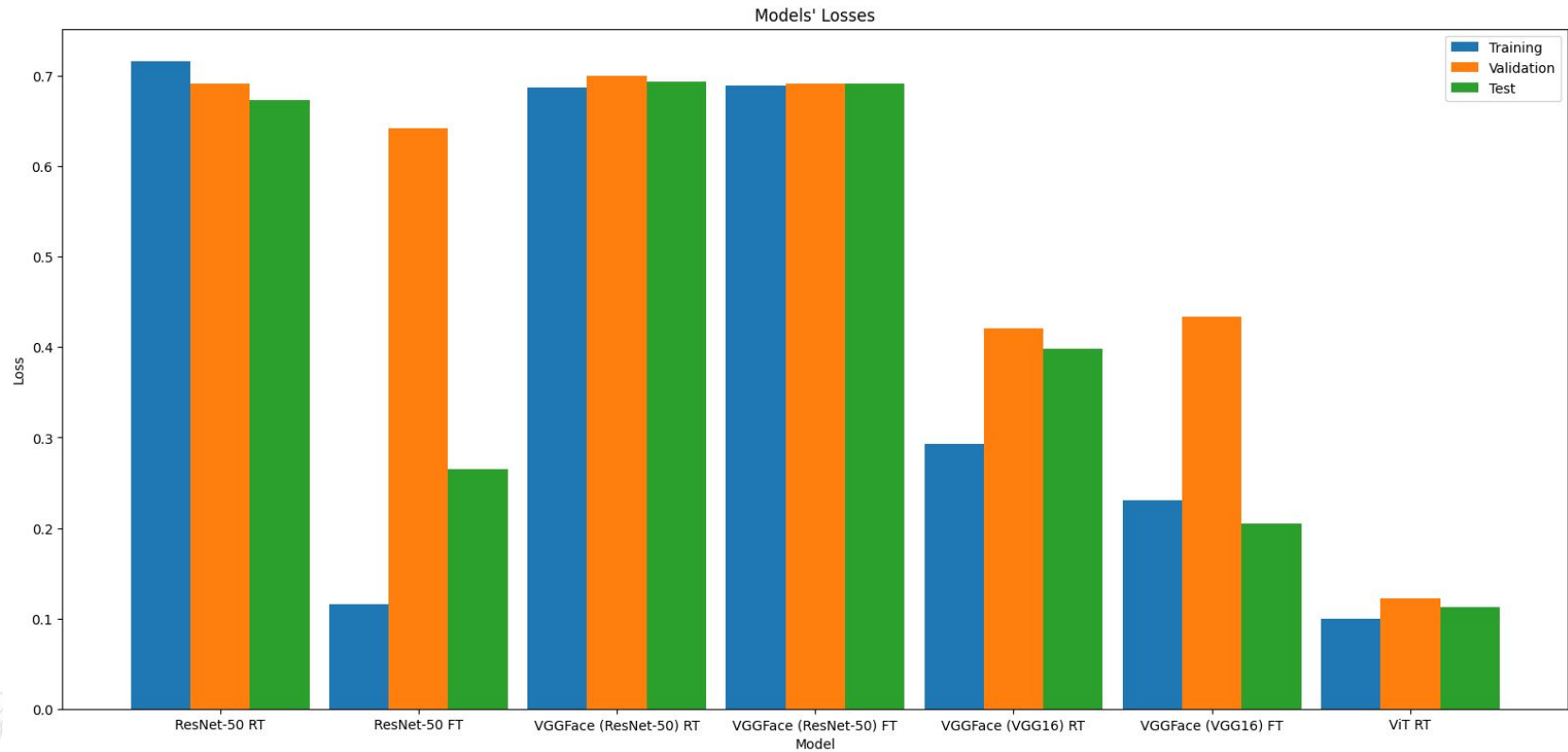
5.

Results and Conclusions

Models evaluation



Models evaluation



Overall results



Successful result



Result with small
error

Further improvements

- ◎ More data is always welcome
- ◎ More diverse and adversarial examples in trainset (adults with baby faces or babies with aged faces)
- ◎ More complex data (pose, angle, age, scale, etc...)
- ◎ Improvement of Face detection model for more accurate bounding boxes

A decorative network diagram in the top-left corner, featuring a complex web of interconnected nodes and lines. The nodes are represented by small circles, some of which are larger and have concentric circles, suggesting different levels or types of connectivity. The lines are thin and gray, creating a mesh-like structure.

A little show off

Let's watch a demo video

A decorative network diagram in the bottom-right corner, similar to the one in the top-left. It shows a cluster of nodes connected by lines, with some nodes being larger and more prominent than others, all rendered in a light gray color.



Thanks!

Any questions?

You can find us at:

`alexzerntev @ di.uoa.gr`

`episko @ di.uoa.gr`

`snikola @ di.uoa.gr`

