Data organisation in video surveillance systems using deep learning

Anastasiia D. Sokolova
Email: adsokolova96@mail.ru
Outline

• Image recognition and faces ordering problem
• Proposed two-stage approach of organizing information in video surveillance systems
• Experimental results in face recognition
• Concluding comments and future plans
Relevance

• Automatic recognition of objects in the field of public safety
• Grouping of video data for statistics
• Face verification as a part of the identity authentication procedure
Face detection

- Haar cascades
- Tensorflow
Feature extraction

- Lightened CNN (version C)
  
  ![Lightened CNN Diagram](image)
  
  Size: 119 Mb
  
  Out: vector of 256 elements

- VGGNet
  
  ![VGGNet Diagram](image)
  
  Size: 553 Mb
  
  Out: vector of 4096 elements
Proposed approach

- Face detection
- Face tracking
- Extraction of D bottleneck features
- Update clusters
- Normalize track descriptors
- Extract homogeneous tracks

Video stream
Algorithms

• Computation of average track features
  \[ \rho(X(m_1), X(m_2)) = \rho(\bar{x}(m_1), \bar{x}(m_2)), \quad \bar{x}(m_i) = \frac{1}{\Delta t(m_i)} \sum_{t=t_i(m_i)}^{t_2(m_i)} x(t) \] (1)

• Pair-wise distance between all frames:
  \[ \rho(X(m_1), X(m_2)) = \frac{1}{\Delta t(m_1) \Delta t(m_2)} \sum_{t=t_1(m_1)}^{t_2(m_1)} \sum_{t'=t_1(m_2)}^{t_2(m_2)} \rho(x(t), x(t')) \] (2)

• Distance between medoids of tracks:
  \[ \rho(X(m_1), X(m_2)) = \rho(x^*(m_1), x^*(m_2)), x^*(m_i) = \arg\min_{x(t), t \in [t_1(m_i), t_2(m_i)], t' \in [t_1(m_i)]} \sum_{t=t_i(m_i)}^{t_2(m_i)} \rho(x(t), x(t')) \] (3)

• Distance between median features of tracks:
  \[ \rho(X(m_1), X(m_2)) = \rho(x'(m_1), x'(m_2)) \] (4)
Datasets

- LFW (Labeled Faces in the Wild)
  - 1680 people
  - 13000 images
  - 1-10 frames
- YTF (YouTube Faces)
  - 1595 people
  - 3425 videos
  - 48-6070 frames
Homogeneous segmentation

0.4285

0.521

0.8934
Roc-curves (Lightened CNN)
Area under curve (Lightened CNN)

- AvePool(1) -> norm
- norm -> AvePool(1)
- AvePool(1)
- norm -> Distance(2)
- Distance(2)
- Medoid(3) -> norm
- Medoid(3)
- Median(4) -> norm

Bar chart comparison of t-test and L2 norms.
Area under curve (VGGNet)

AvePool(1)->norm
norm->AvePool(1)
AvePool(1)
norm->Distance(2)
Distance(2)
Medoid(3)->norm
Medoid(3)
median(4)

[Bar chart showing comparisons between t-test and L2 for different operations]
Clustering results

Clusters (Lightened CNN)
- 35
- 2457

Clusters (VGGNet)
- 44
- 2594

Threshold = 0.0569
Conclusion

• Our algorithm is based on the ways to efficiently compute the dissimilarity of video tracks by using rather simple aggregation techniques.
• The most accurate and computationally cheap technique involves the $L_2$-normalization of average unnormalized features of individual frames.

Future work

• Research more sophisticated distances between video tracks, e.g., metric learning or statistical homogeneity testing.
• Usage other clustering methods, e.g., approximate nearest neighbor search.
Thank you for attention!