Яндекс

Video Analysis: the Yandex approach

Alexander Shishenya
Deep Learning for Computer Vision Senior Developer

Video analysis and processing

The Tasks:

1. Ranking
2. *Recommendations*
3. Classification/tagging
    - Rubrication
    - Brand-safety
4. *Duplicates detection*
5. Scene cut detection
6. Highlights generation
7. Superresolution

Video analysis and processing

The Tasks:

1. Ranking
2. *Recommendations*
3. Classification/tagging
    - Rubrication
    - Brand-safety
4. *Duplicates detection*
5. Scene cut detection
6. Highlights generation
7. Superresolution

Universal embeddings:

- Multidomain
- Robust to various lengths
- Covering the full video
- Have fixed length

Datasets: Opensource

Datasets

- HMDB51 / UFC101
- Kinetics
- Sports 1M
- COIN
- Moments In Time
- HowTo 100M

Datasets: Opensource

Datasets

- HMDB51 / UFC101
- Kinetics
- Sports 1M
- COIN
- Moments In Time
- HowTo 100M

Common problems

- Small size
- Narrow domain
- Little intra-domain variation
- Short segments
- Fixed length videos

Datasets: Opensource

Datasets

- HMDB51 / UFC101
- Kinetics
- Sports 1M
- COIN
- Moments In Time
- HowTo 100M
- YouTube 8M

Common problems

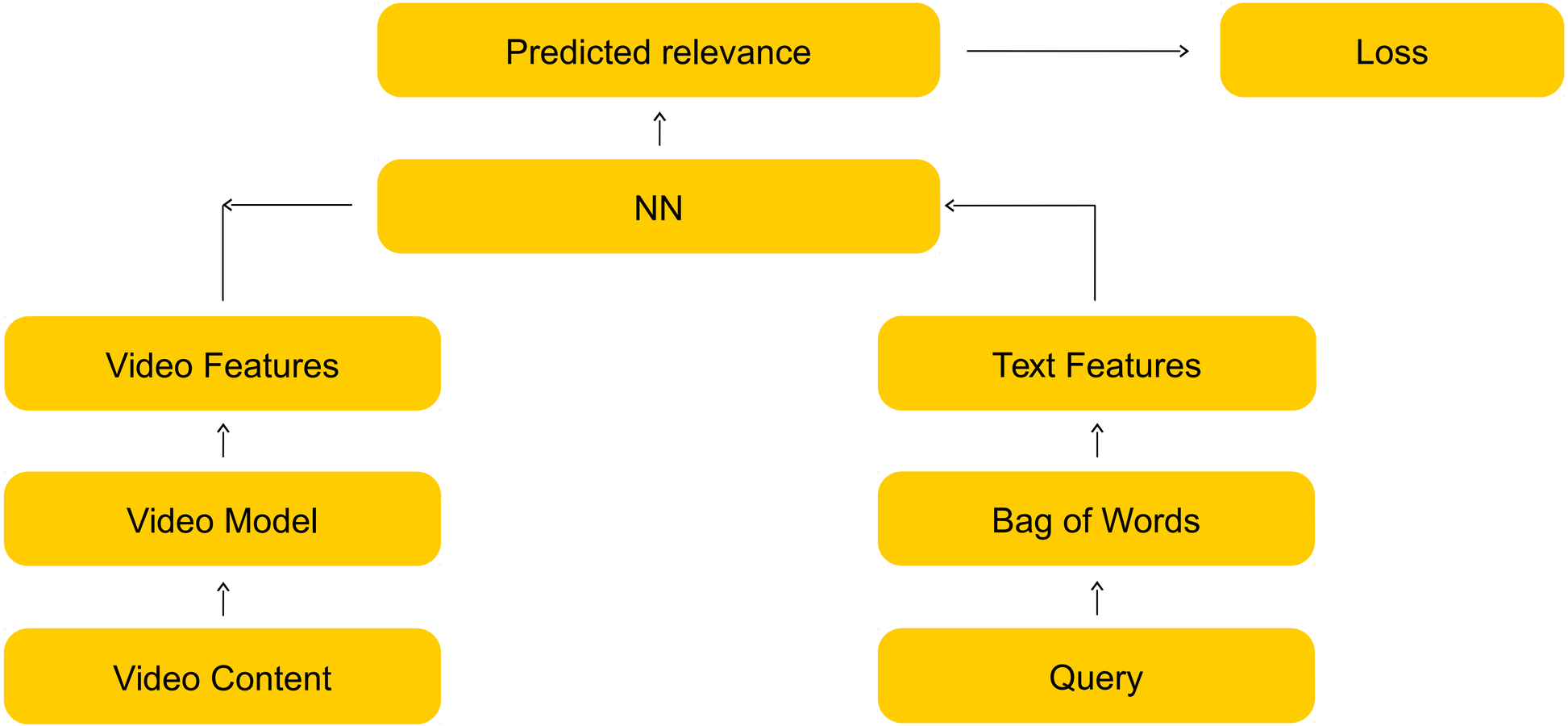
- Small size
- Narrow domain
- Little intra-domain variation
- Short segments
- Fixed length videos

Datasets: In-house dataset

Search engines datasets (query + video)

- Search engine results top elements
  - Replicates the existing ranking
  - Too noisy
- User clicks
  - Little correlation with video content
- User deep views
  - 120M videos, 200M queries, 1G positive samples
  - Mostly clean data, labeled by human users
  - Mining negative samples is tough

Ranking architecture: Late Fusion



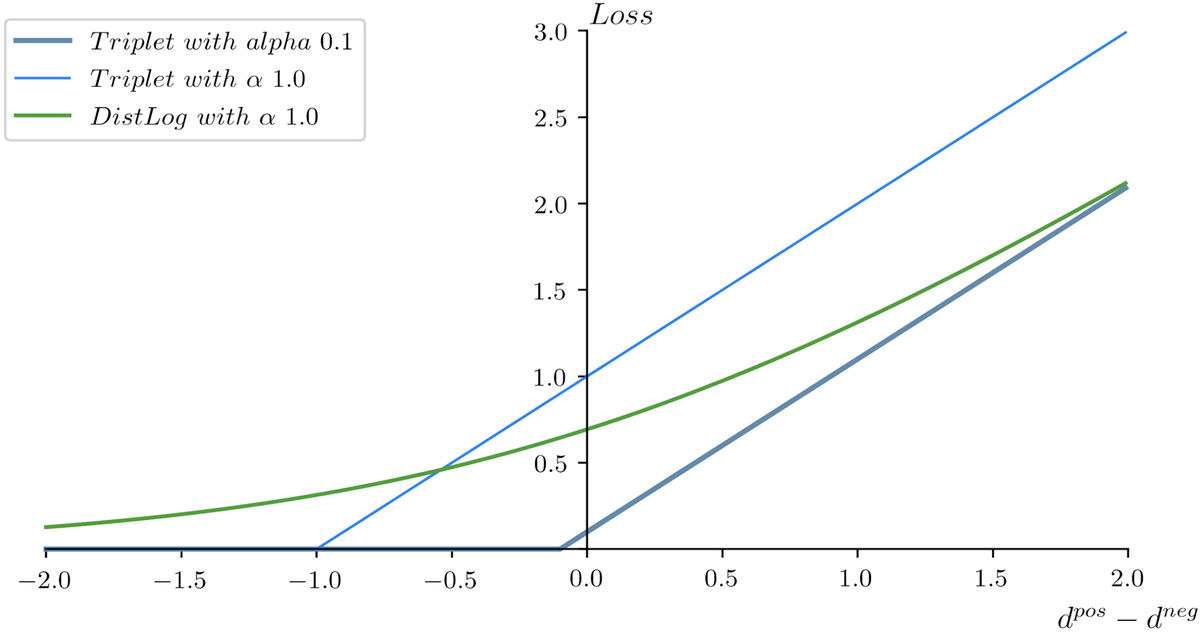
Loss function

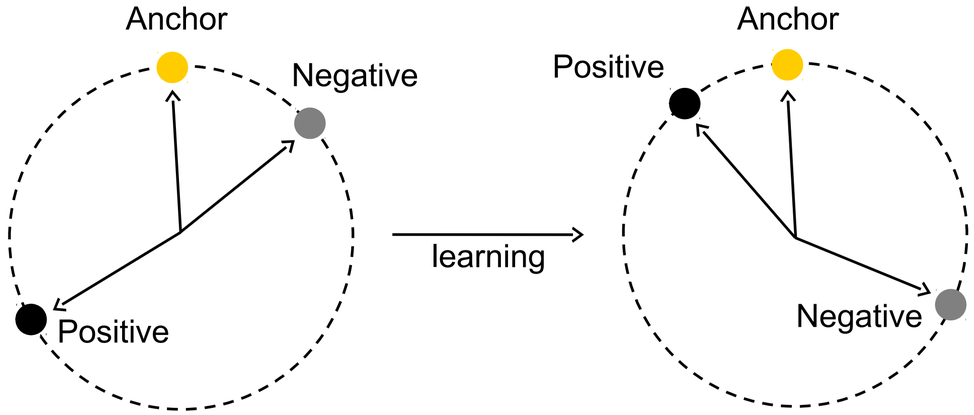
$$d_i^{pos} = 1 - cos\left(f(x_i^{anc}), f(x_i^{pos})\right)$$

$$d_i^{neg} = 1 - cos\left(f(x_i^{anc}), f(x_i^{neg})\right)$$

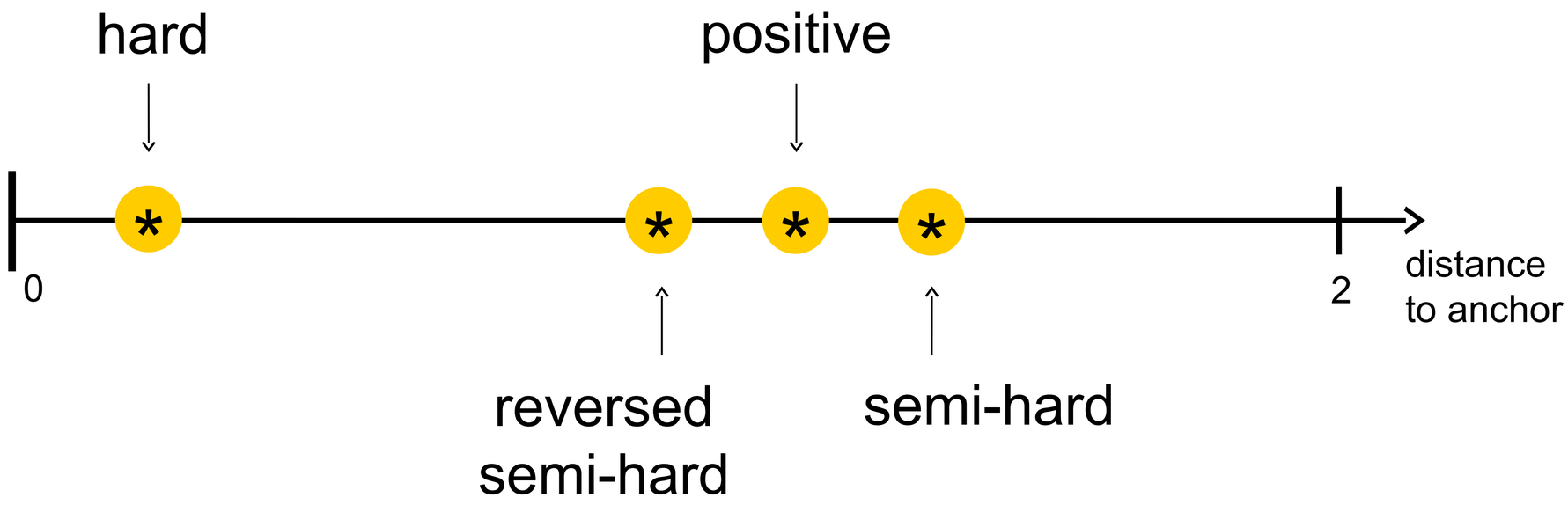
$$Loss_{triplet} = [dpos - dneg + \alpha]_+$$

$$Loss_{distLog} = -log[\sigma(\alpha(d^{neg} - d^{pos}))]$$



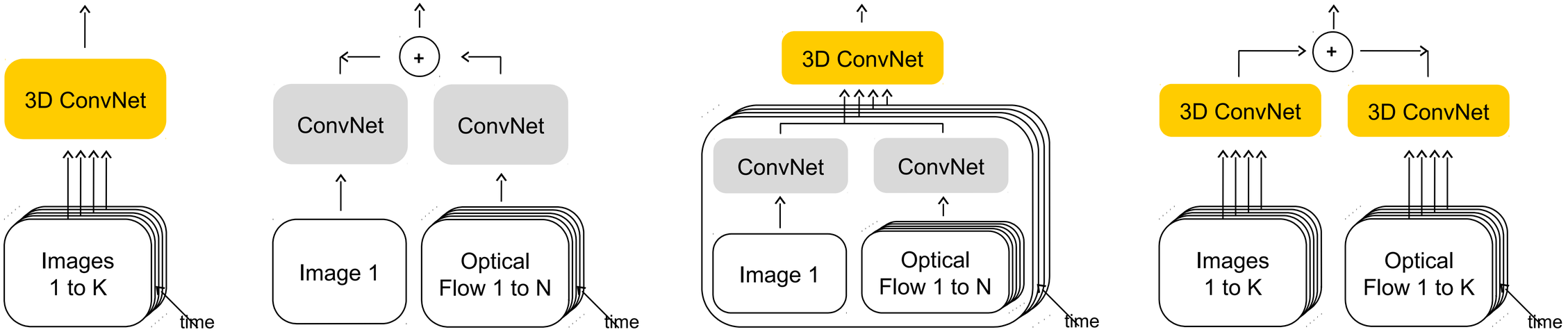


Negative mining



Video feature extractor

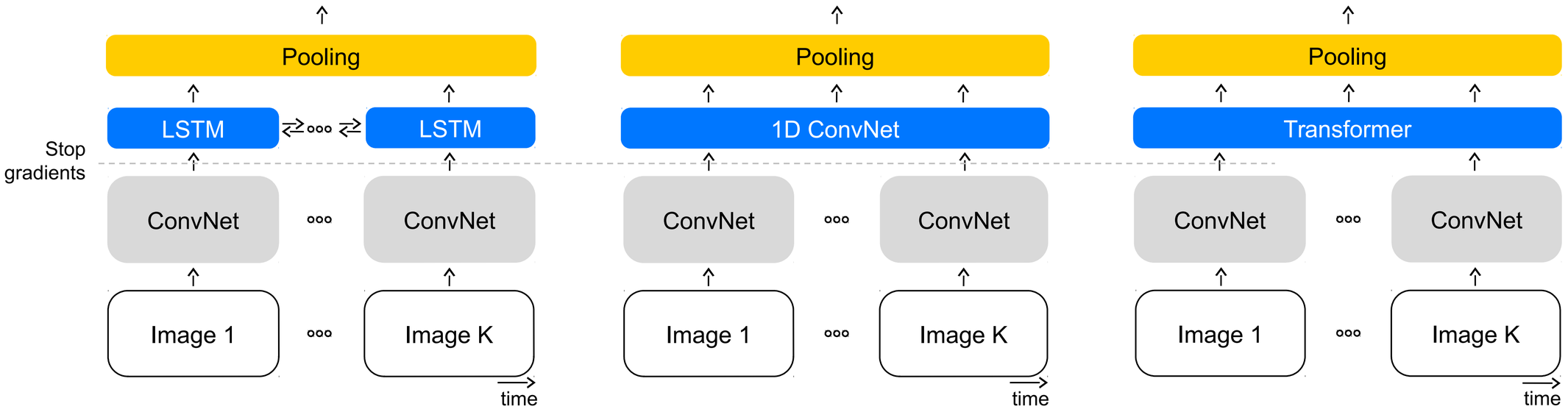
1. **Training on frame data (2D/3D conv, Optical flow)**
   - Trains from scratch, no information is lost
   - Captures temporal structure, e.g. motion, naturally
   - Painfully slow
   - **Requires tremendous amounts of GPU RAM**
   - 3D conv/OptFlow make the network bigger and slower



Video feature extractor

2. Training on frame features (ImageNet like)

- Trains fast, can use larger batch size
- **Looses some information**
- Depends on the backbone network
- More complex data & training pipeline



Ranking results

Model	hard@64, % (less is better)	Yandex.Video ranking score delta (more is better)
BiLSTM (batch size 128)	16.4	320
BiLSTM (batch size 128) (more data)	12.6	630
BiLSTM (batch size 2048) (even more data)	8.2	1060
1D CNN	7.75	1220
1D CNN + Transformer	8.05	-

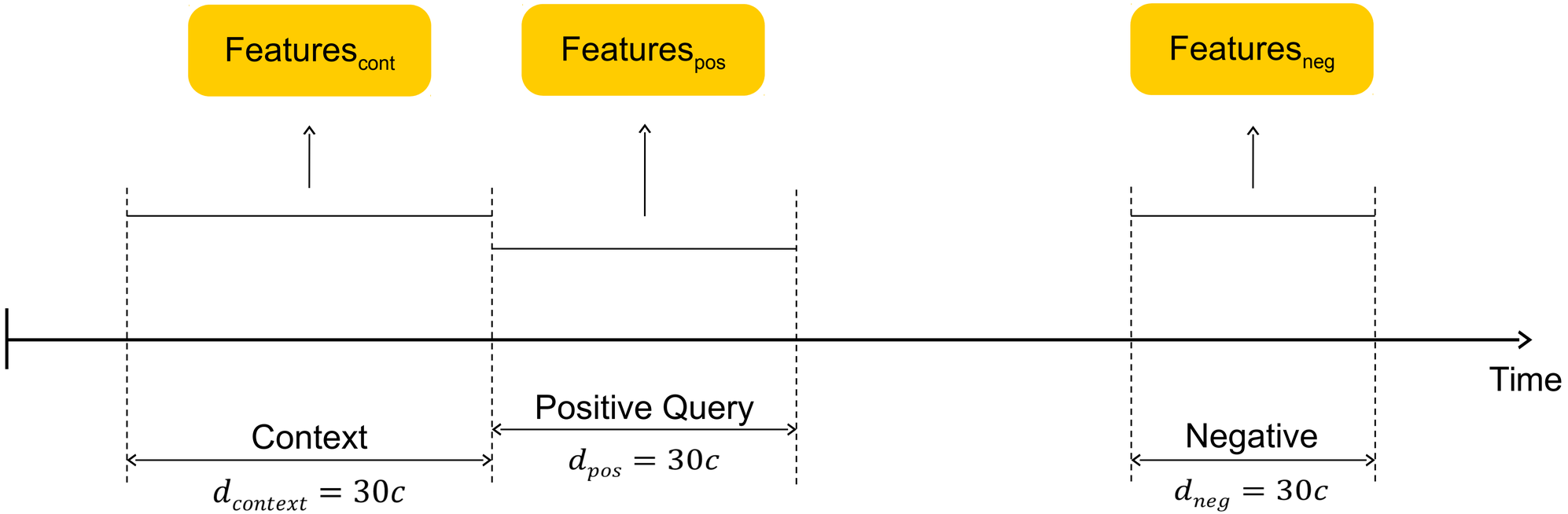
Global embegging. Problems

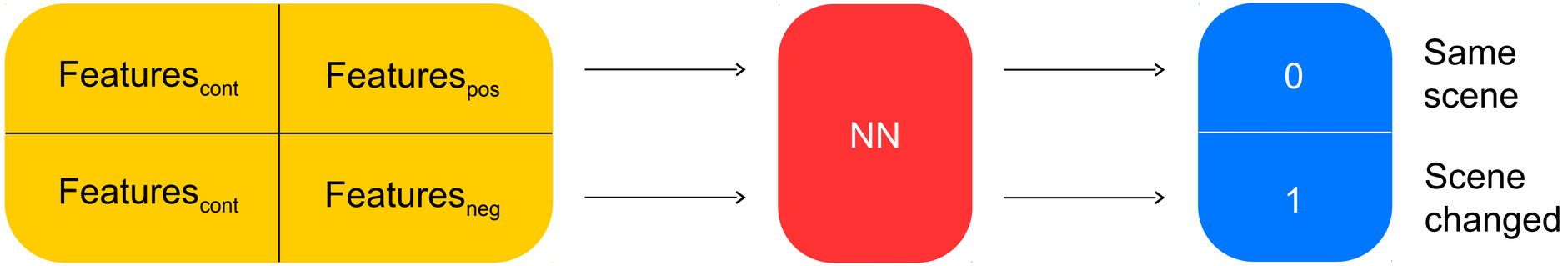
- Easy to use
- Fixed size per video
  - Fails to capture nuances of long videos
  - Fails to described weakly connected content
  - Hard to do segment-based analysis

We need a local descriptor

- Fixed size
- Describes some self-contained period
- Is part of a global descriptor $D_{global} = PullingOp(\{D_{local}\})$

Scene cut detection





Scene cut detection: Results

- **A diverse dataset of 50 different length videos**
- **Scene change is determined by crowd sourcing**

Model	Precision, %	# detected points
random	~3	190
cnn	22.61	2900
cnn multihead attention fusion	46.53	1300
cnn multihead attention fusion fine-tuning on small labeled dataset	73.18	600

Highlights: Intro

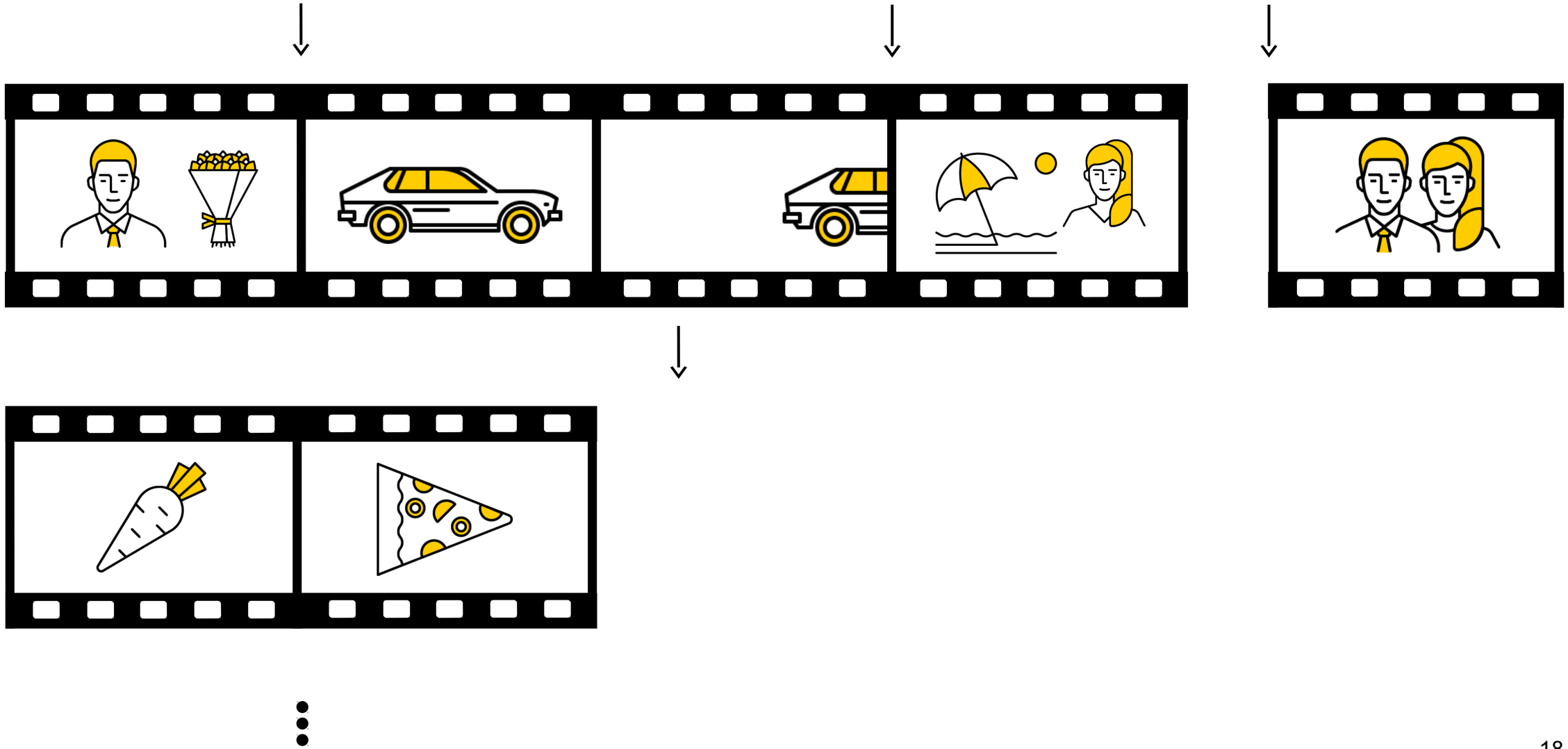
The way people use to watch videos

- Funny, fairly short content

Typical videos on Yandex

- Produced by vloggers 20–40 minutes long

Highlights



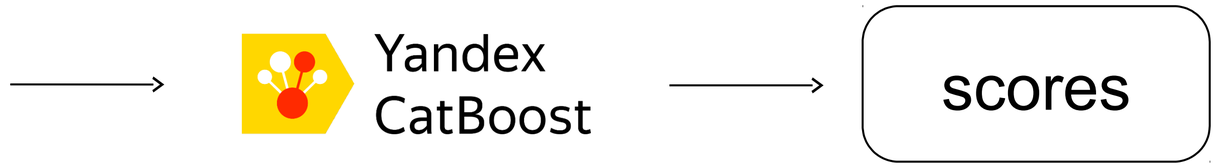
Highlights: Ranking

Video embeddings

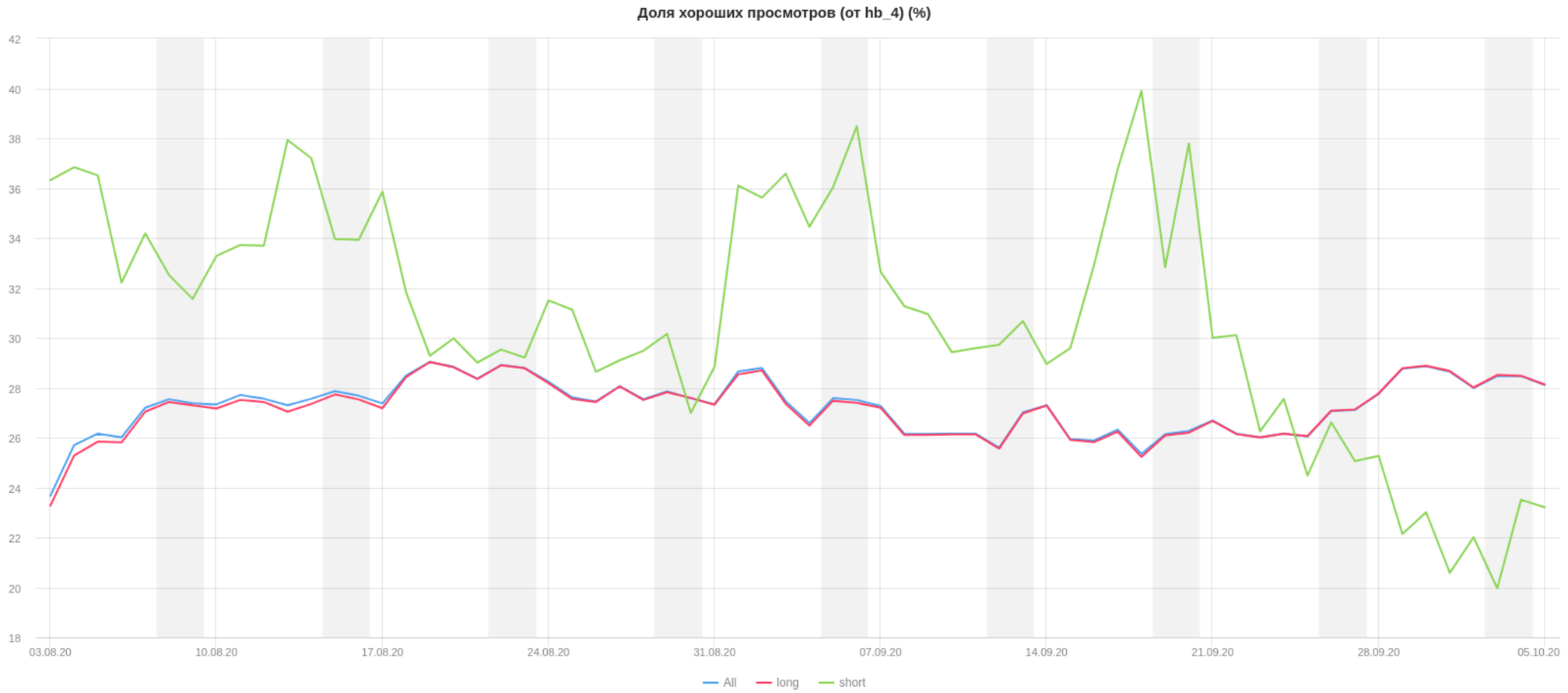
Audio embeddings

Per-frame features

Similiarity to the title



Highlights: Results



- **Basically on par with original videos!**

Classifiers

Rubrication

- Humor
- Travel
- DIY
- Beaty
- Auto
- …

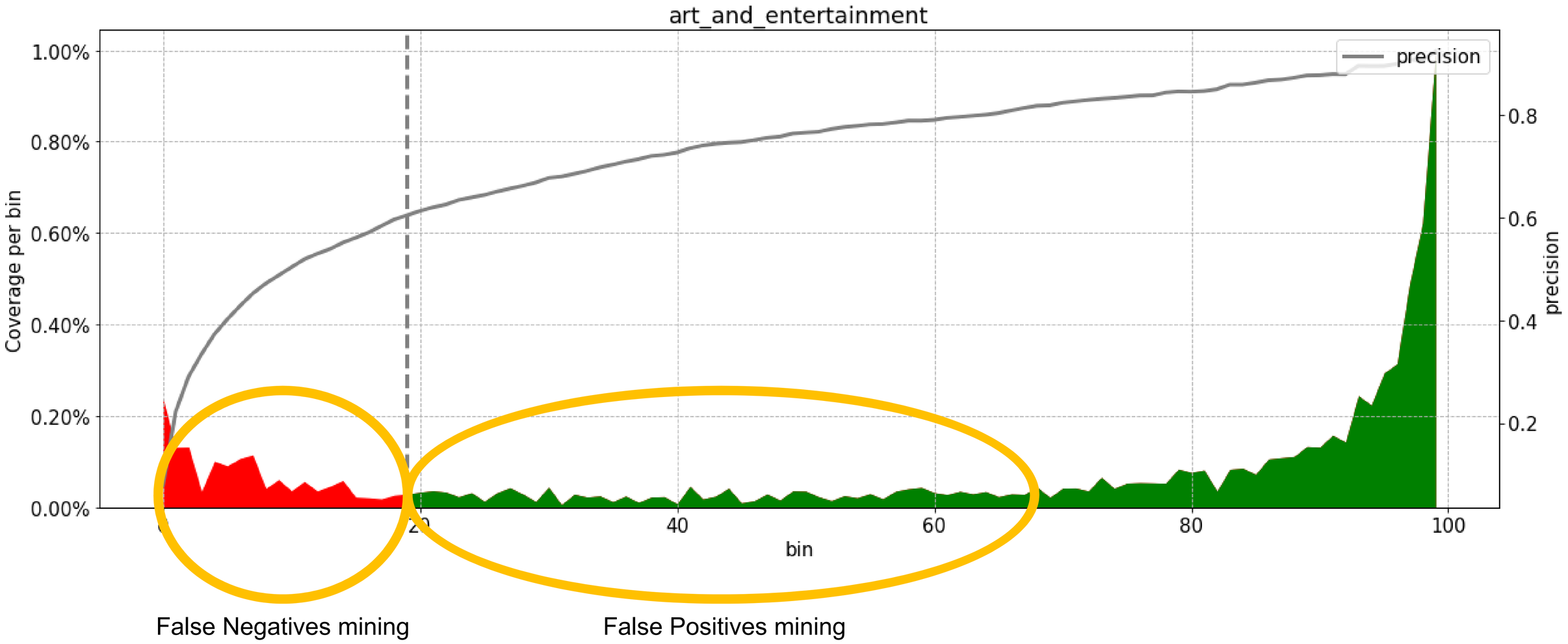
Moderation

- Violence
- Hatespeech
- Smoking
- Gambling
- 18+
- …

Classifiers: Problems

- Video is really hard to label
- One often needs both local and global context
- Low rate of positive samples in data (~1%)

Classifiers: active learning



SuperResolution: problem statement

Having low-quality and low-resolution videos transform them into high-quality and high-resolution sharp and realistic videos.

- How to measure quality of the video?
- How to measure that video is realistic?

SuperResolution: quality metrics

$$PSNR(x,y) = 10log_{10}\left(\frac{MAX_I^2}{MSE(x,y)}\right)$$

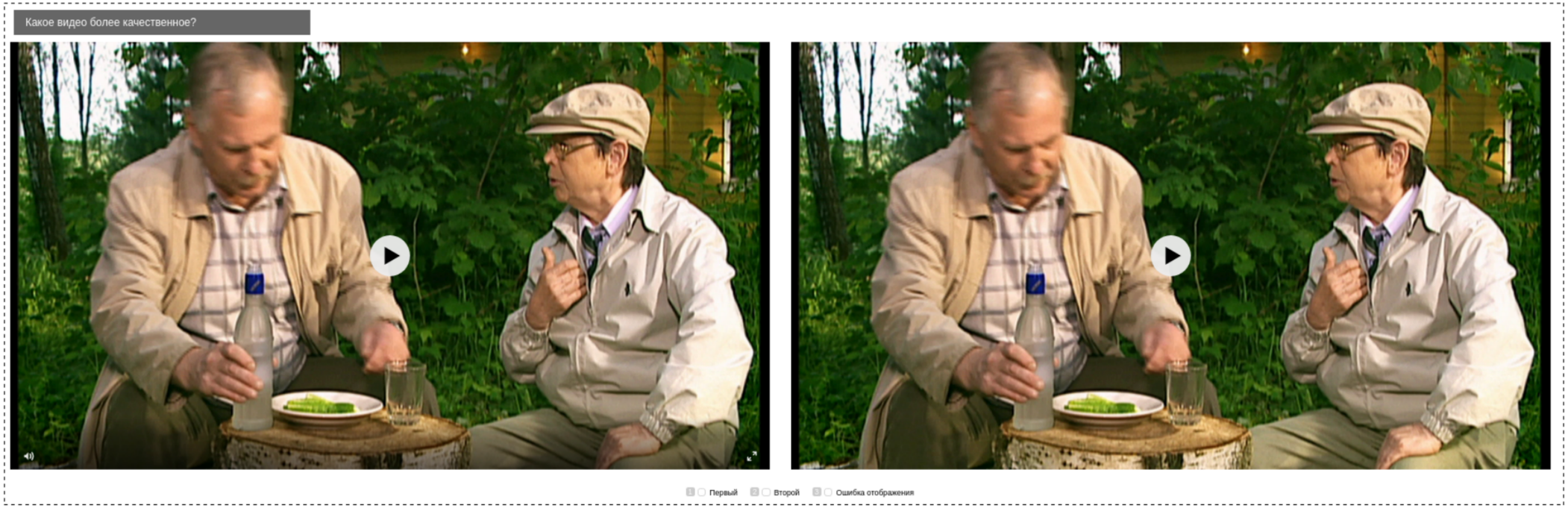
$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_1)}$$

SuperResolution: quality metrics

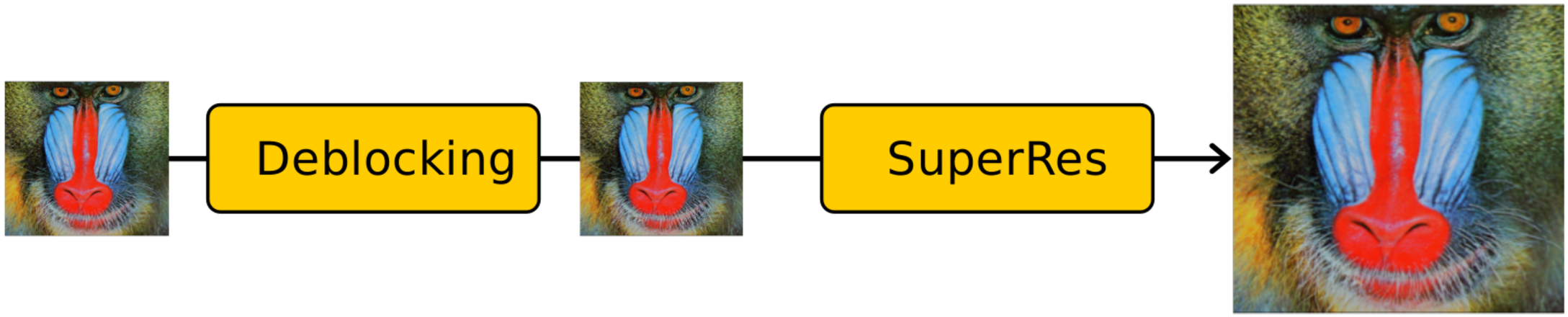
Limitations of SSIM and PSNR

- Weakly correlate with human sence of good quality.
- The restored video may differ from the original one with “noisy” details and still have a good quality.
- Require original high-quality videos.

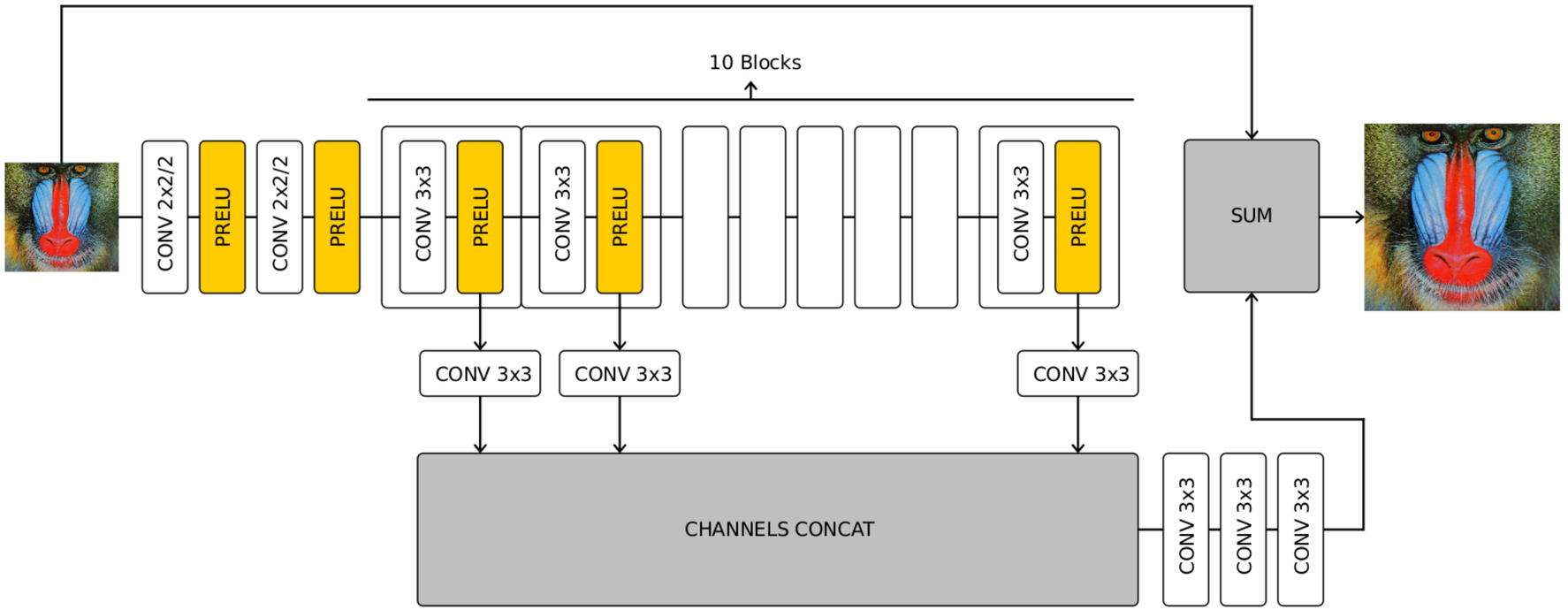
SuperResolution: Side-by-Side



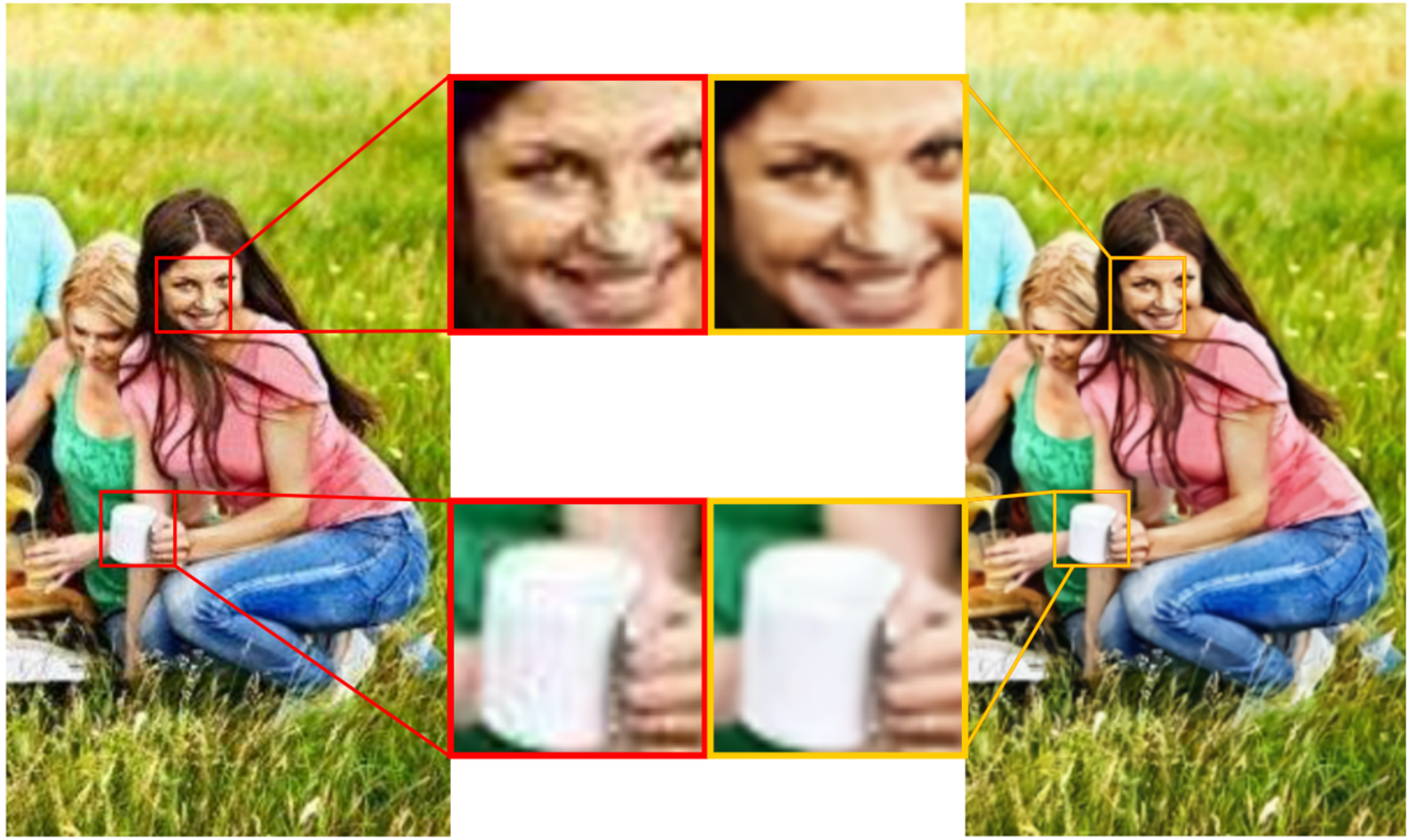
SuperResolution: General Architecture



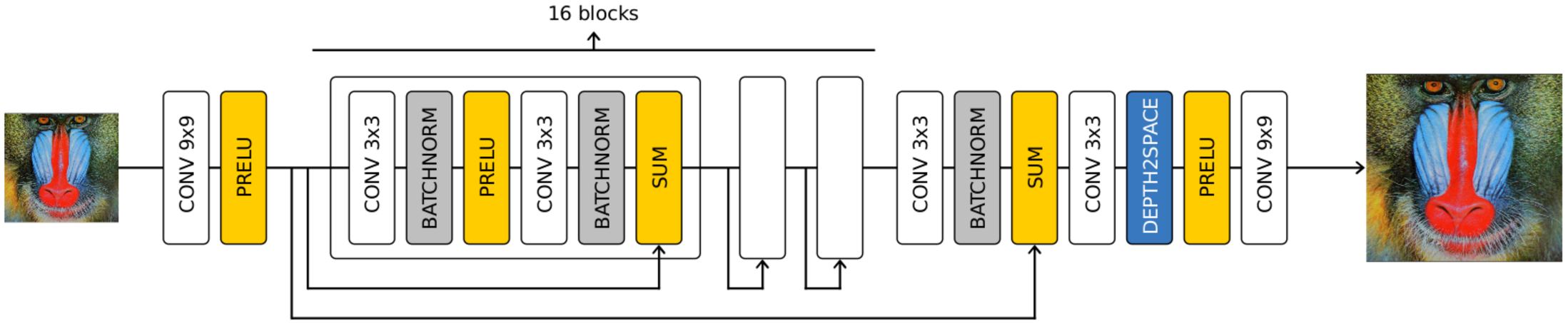
SuperResolution: Deblocking



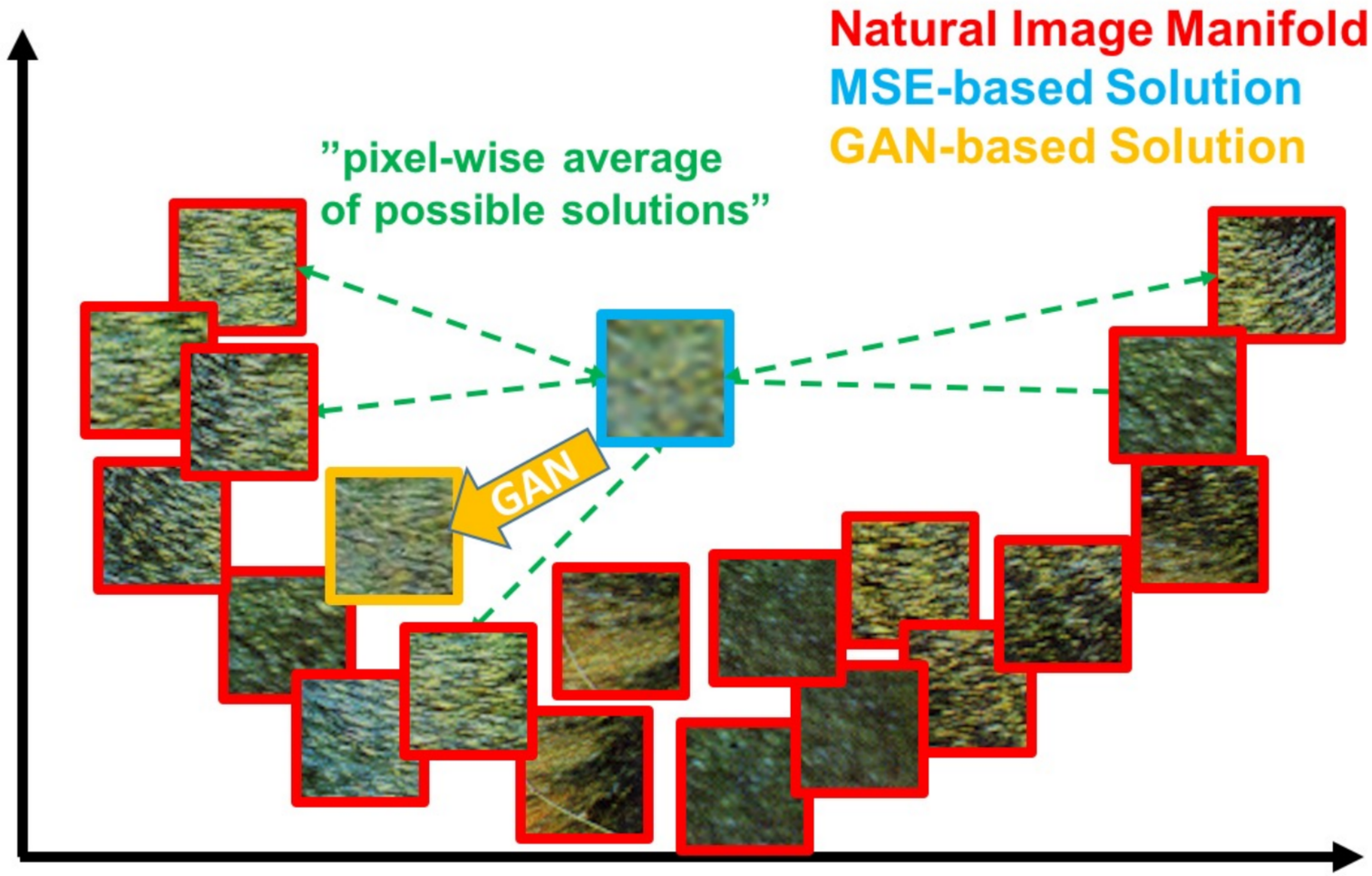
SuperResolution: Deblocking



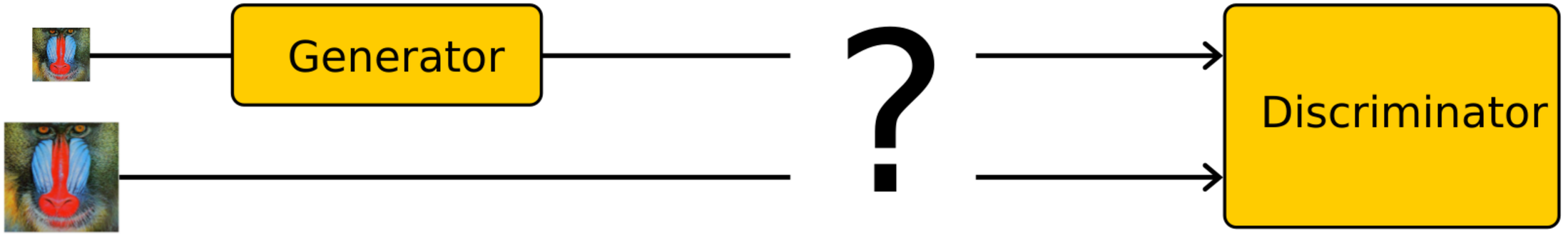
SuperResolution: SuperRes



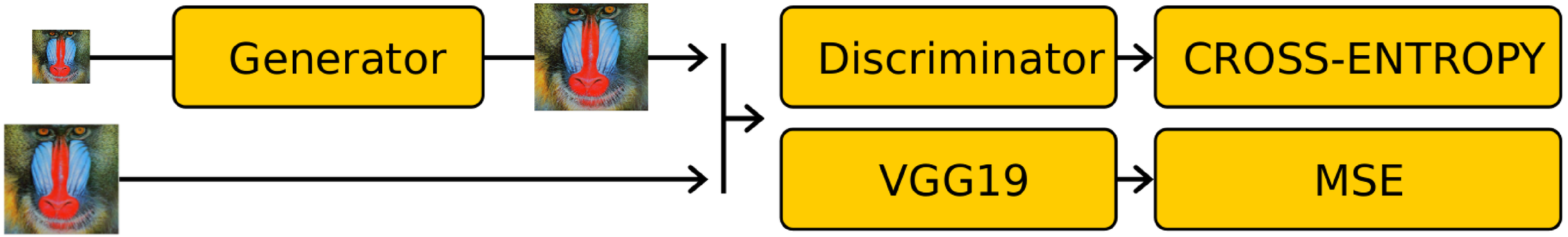
SuperResolution: GAN



SuperResolution: GAN

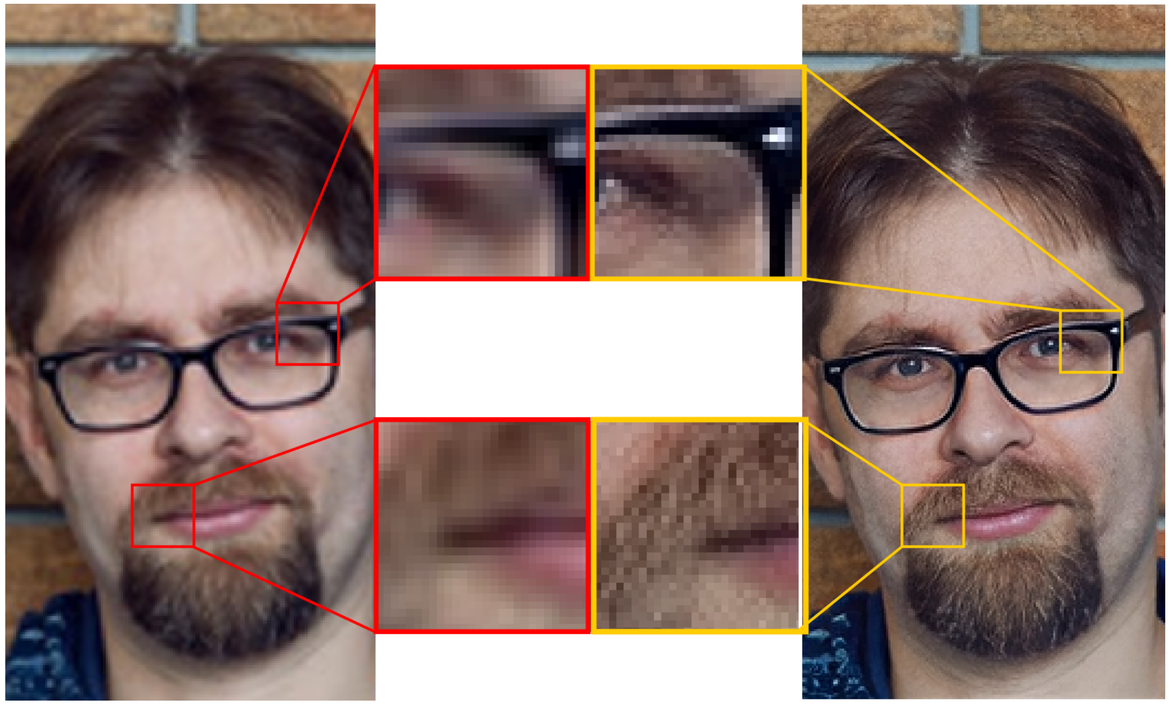


SuperResolution: GAN



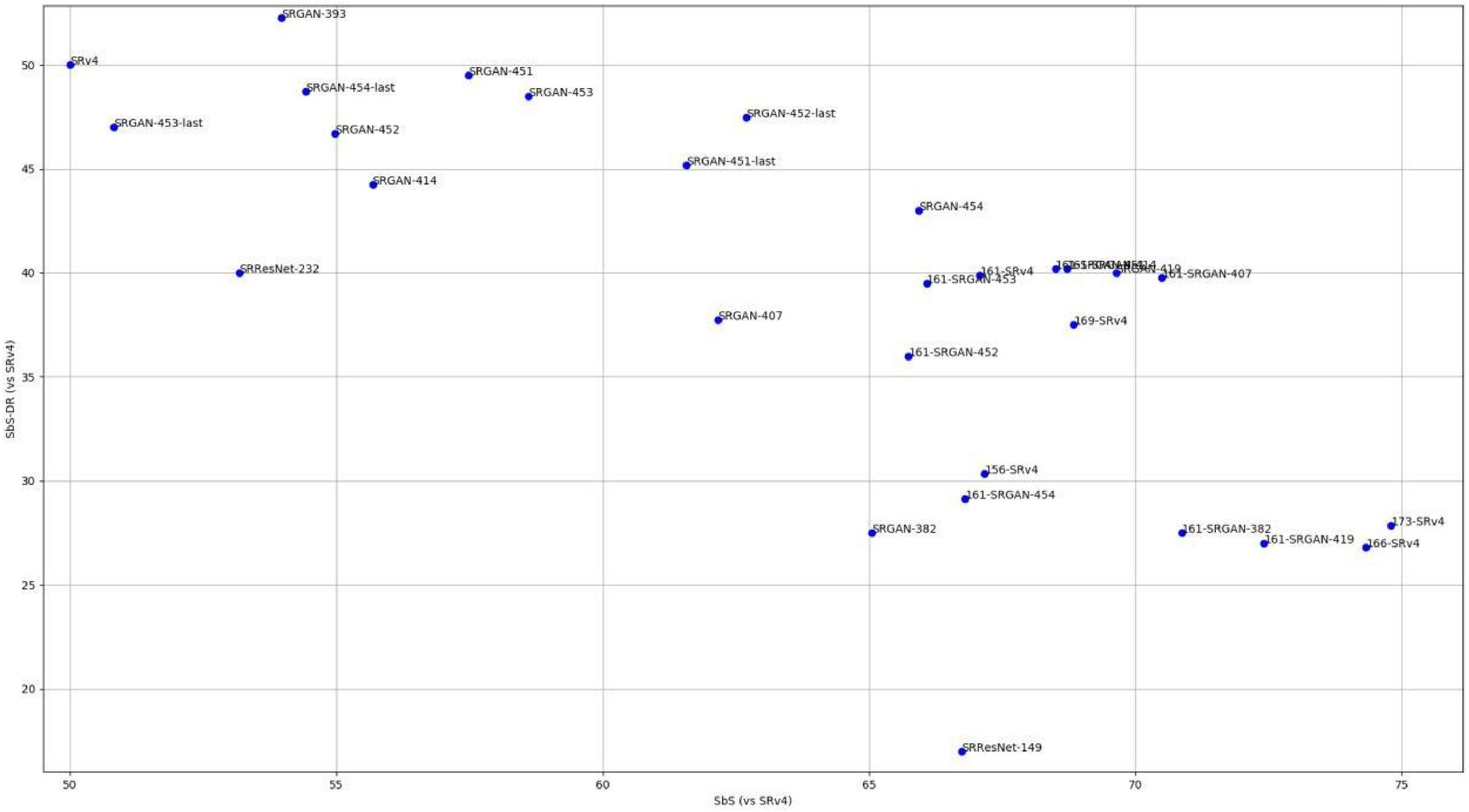
SuperResolution: Examples

SuperResolution: Results



- ~90% wins over original
- GAN-based
- Performance-quality tradeoff

SuperResolution: Visually appealing vs Natural



Яндекс

Thank you for your attention!

Alexander Shishenya

shishenya@yandex-team.ru