NATIONAL RESEARCH UNIVERSITY
HIGHER SCHOOL OF ECONOMICS

Faculty of Computer Science
Bachelor's Programme "Applied Mathematics and Informatics"

BACHELOR'S THESIS

Software Project on the Topic:

Deep Learning Applied to a Video Hosting Visual Design Task

Submitted by the Student:

group #БПМИ209, 4th year of study Kemal Azamat

Approved by the Supervisor:

Skvortsov Grigory Dmitrievich
Lecturer
Basic Department of Sberbank PJSC

Moscow 2025

Contents

Annotation **3**

1 Introduction **4**

2 Analysis of Existing Methods **6**

3 Methodology **8**

4 Implementation **10**

4.1 Decoding the Video Stream and Initial Frame Filtering 10

4.2 Local Contrast Enhancement using CLAHE 11

4.3 Filtering Out "Redundant" Frames and Object Presence Verification 12

4.4 Clustering and Selection of the Most Expressive Frames 13

4.5 Matting (Background Removal) 14

4.6 Stylization of Key Frames Using GAN 15

5 Custom Implementation of the Key Frame Detector **16**

5.1 Baseline Experiment: "Frozen" ResNet-50 and Logistic Head 17

5.2 Partially Fine-Tuned ResNet-50 with Class Balancing 17

5.3 Modeling Temporal Structure: Bidirectional LSTM 18

5.4 Headline Generation with Russian T5 Models 20

5.5 Post-processing and Final Headline Selection 22

5.6 Thumbnail Generation via Subject–Predicate Extraction and Stable Diffusion . 24

5.7 Expansion of a Square Image without Content Distortion 25

6 System from the User's Perspective **27**

7 Conclusion **29**

References **30**

Annotation

Modern deep learning technologies are widely used in multimodal information processing tasks, including automatic generation of visual and textual elements of digital content. With the growth of video platforms, the need for effective solutions to automate the visual design of video clips is increasing. This paper is devoted to the development of a comprehensive system based on computer vision and natural language processing techniques for automatic generation of thumbnails (covers), avatars and titles to user videos. The proposed approach integrates object detection, segmentation, image and text generation using state-of-the-art architectures. The developed system is especially useful for novice authors who have no design skills, thus reducing the threshold of entry into video content creation.

Аннотация

Современные технологии глубокого обучения находят широкое применение в задачах мультимодальной обработки информации, включая автоматическую генерацию визуальных и текстовых элементов цифрового контента. В условиях роста видеоплатформ возрастает необходимость в эффективных решениях по автоматизации визуального оформления видеороликов. Данная работа посвящена разработке комплексной системы, основанной на методах компьютерного зрения и обработки естественного языка, предназначенной для автоматической генерации миниатюр (обложек), аватаров и заголовков к пользовательским видеороликам. Предложенный подход интегрирует детекцию объектов, сегментацию, генерацию изображений и текстов с использованием современных архитектур. Разработанная система особенно полезна для начинающих авторов, не обладающих навыками дизайна, тем самым снижая порог входа в создание видеоконтента.

Keywords

Deep Learning, Computer Vision, Segmentation, Large Language Model, Predtrained Models, LSTM

1 Introduction

Automation of visual design for video content is an important applied task at the intersection of computer vision (CV) and natural language processing (NLP). Video hosting platforms impose standardized requirements on the structure of published material: a thumbnail, a title, and, if necessary, the author's avatar. These elements not only influence click-through rates but also form the primary semantic representation of the video clip (Shah, 2022). The absence of a thumbnail or an irrelevant title can significantly reduce reach and engagement. At the same time, manual design requires time and the appropriate expertise, creating an additional barrier -especially for new users.

This work is aimed at developing a pipeline for the automatic generation of visual and textual accompaniment for user-generated videos. The solution includes the following modules:

1 **Visual stream:**

- extraction and selection of key frames from the video (using a modular algorithm and a separately trained neural network);
- detection of main objects (YOLOv8) and background matting (MODNet);
- generation of background images (Stable Diffusion 3 Medium);
- stylization via GAN architectures and frame expansion with a gradient mask.

2 **Text stream:**

- transcription of the audio track (Whisper);
- generation of up to three title variants (RuT5);
- selection and deduplication (GuidMaster – our deduplication algorithm based on cosine similarity of embeddings from three language models);
- overlay of the chosen text onto the thumbnail (font, color, and position adjustment).

For quantitative evaluation of semantic correspondence between visual and textual elements, we use the CLIP-score — the cosine similarity of image and title embeddings computed by the CLIP multimodal model. Additionally, an NSFW classifier is applied to filter out undesirable content in the form of text or images.

The "Visual stream" block can be implemented in two ways. The first is a classical pipeline, where specialized pretrained models are used for detection, segmentation, and filtering tasks. The second is a unified neural network trained end-to-end on a specialized dataset. All filtering modules from the first variant can also be integrated into the second architecture.

A distinctive feature of our work is the generation of the video title and its use as a prompt for Stable Diffusion 3 when creating the background image of the thumbnail.

Our code is available online: https://github.com/sesquiiipedalian/hse_diploma_2025

All components are integrated into a single pipeline capable of processing videos without user intervention and producing a complete design package.

2 Analysis of Existing Methods

Key Frame Extraction. Early video analysis methods were based on comparing color histograms and absolute pixel differences (H. Zhang et al., 1995; Chuang and al., 1999). With the introduction of convolutional networks, CNN-based feature approaches emerged (Karpathy and al., 2014), improving robustness to noise and accuracy in selecting representative frames.

Object Detection. The shift from HOG+SVM (Dalal and Triggs, 2005) to one-stage deep learning detectors began with YOLOv1 (Redmon et al., 2016), continued with YOLOv2/v3 (Redmon and Farhadi, 2017; Redmon and Farhadi, 2018), and most recently YOLOv8 was released (Jocher et al., 2023). In parallel, SSD (W. Liu et al., 2016) and the transformer-based DETR (Carion et al., 2020) were developed. According to comparative testing on COCO (mAP, FPS) and considering real-time requirements, YOLOv8 demonstrated mAP $> 50\%$ with inference latency < 15 ms on a mid-range GPU, whereas SSD achieved mAP $\approx 34\%$ at a comparable speed, and DETR lagged in FPS (< 30). Therefore, YOLOv8 was chosen for frame object detection.

Segmentation and Matting. Classical semantic segmentation models (U-Net (Ronneberger, Fischer, and Brox, 2015), DeepLabV3+ (Chen et al., 2017)) provide high accuracy (IoU ≈ 0.94) but demand significant resources. In our code, we consider the specialized portrait matting model MODNet (Z. Liu et al., 2021). It introduces an e-ASPP module and is optimized for real-time performance (≈ 67 FPS) with IoU ≈ 0.92 on the corresponding datasets. The choice of MODNet here is driven by the trade-off between mask quality and throughput for video processing.

Image Generation (GAN and Diffusion). Since 2014, GANs (Goodfellow et al., 2014) and their evolution (StyleGAN2 (Karras, Laine, and Aila, 2019)) set a new standard for photorealism; however, for *text-to-image* and *outpainting* tasks their architectures required enhancements and yielded FID > 15. With the rise of diffusion models (DDPM (Sohl-Dickstein et al., 2015); Ho et al. (Ho, Jain, and Abbeel, 2020)) and the release of Stable Diffusion (Rombach et al., 2022), achieving FID < 10 became possible, flexibly supporting *img2img* and *outpainting*. Considering the open-source implementation (Diffusers) and the optimal quality/speed balance, Stable Diffusion 3 Medium was chosen.

Title Generation and Summarization. Methods for headline generation based on seq2seq models (Rush, Chopra, and Weston, 2015; See, P. J. Liu, and Manning, 2017) have evolved

alongside the universal T5 (Raffel et al., 2020). For the Russian-language task, RuT5 (Gusev, 2022) demonstrated high metrics on news data. A fine-tuned model `cointegrated/rut5-base` on the Lenta.ru headline corpus (7 words) achieved $ROUGE\text{-}L = 0.45$ and $PPL = 12$ on validation, justifying its application.

Generation Quality Evaluation. The multimodal CLIP model (Radford, J. W. Kim, et al., 2021a) jointly trains visual and textual encoders in a contrastive setting. The CLIP-score (cosine similarity of embeddings) is widely used as a proxy metric for semantic correspondence between images and text and is integrated into the pipeline to automatically select the most relevant thumbnail–title pairs.

Thus, the combination of proven architectures and selected models provides an optimal balance of detection accuracy, mask quality, photorealistic synthesis, text conciseness, and performance. The next chapter, "Methodology," presents detailed descriptions of the algorithmic implementations, hyperparameters, and evaluation procedures for each module.

3 Methodology

Pipeline Overview. Figure 3.1 illustrates the overall structure of the automatic video content design algorithm, consisting of two parallel branches — visual and audio-text. The implemented scheme allows simultaneous processing of video frames and textual information from the audio track, ensuring correlation between the thumbnail image and the generated title.

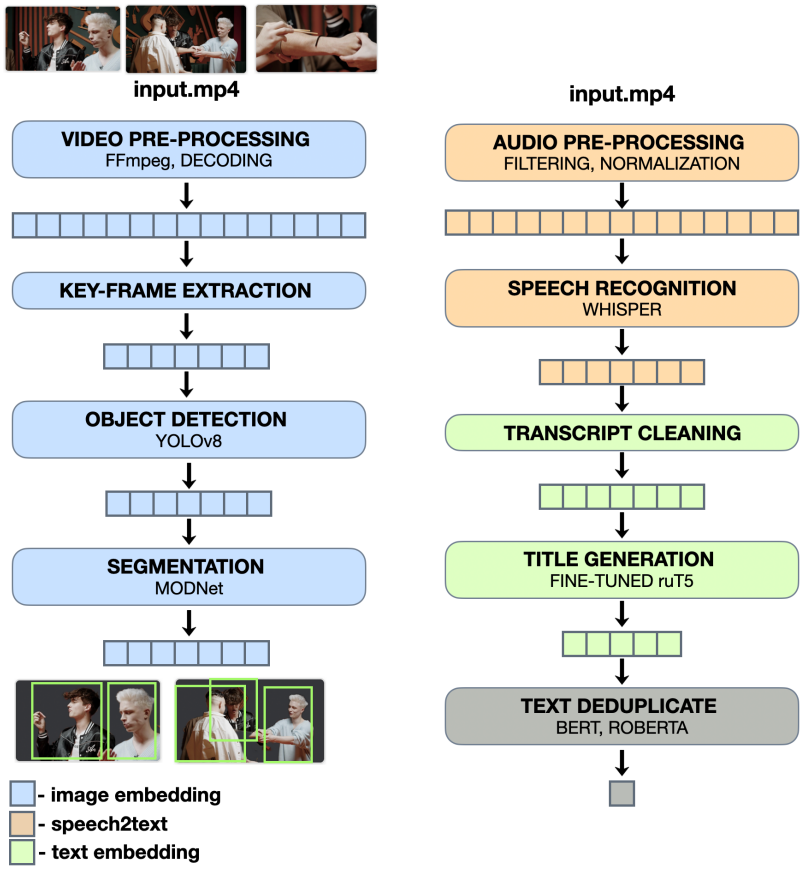


Figure 3.1: Multimodal processing scheme for automated video thumbnail and title generation

Visual Stream. The video stream is converted and decoded using FFmpeg (H.264), after which key frames are extracted based on sharpness, brightness, and contrast metrics. On the selected frames, the YOLOv8 model (Redmon et al., 2016) localizes the primary object (class "person"), and MODNet (Z. Liu et al., 2021) generates an α-mask to separate it from the background. The background layer is synthesized either via Stable Diffusion 3 Medium (Rombach et al., 2022) in *img2img* mode or selected from a prepared collection; the algorithm then performs outpainting: cutting a gradient strip, seam-blurring, and adding noise. The final composition is scaled and cropped so that the thumbnail has the same aspect ratio as the original video frames.

Audio and Text Stream. The audio track is filtered and normalized, then transcribed by the Whisper model (Radford et al., 2022). The resulting text is cleaned of noise artifacts and stop

words, after which the fine-tuned model `cointegrated/ruT5` trained on the Lenta.ru corpus generates ten title variants ($\leqslant$ 7 words, $\leqslant$ 51 tokens), following the multi-sample practice for seq2seq models (Rush, Chopra, and Weston, 2015; See, P. J. Liu, and Manning, 2017). The GuidMaster mechanism is based on embeddings from BERT (Devlin et al., 2019a), RoBERTa (Y. Liu et al., 2019), and E5 (Reimers and Gurevych, 2023): the system computes weighted combinations of vector representations of the titles and source text, indexes master/slave entries by cosine similarity, and removes duplicates. Final ranking is performed by an NER module (Lample et al., 2016a) based on the number of unique named entities.

4 Implementation

Video Conversion to H.264. The optimization of the source video stream format is performed by checking its codec with `ffprobe`. If it is not H.264 (or HEVC), hardware transcoding is carried out using NVIDIA NVENC:

NVENC provides high throughput with minimal visual quality loss, which is critical for subsequent analysis of large volumes of material (ISO/IEC 14496-10).

Parameter	Value/Purpose
`-hwaccel cuda`	Hardware acceleration via CUDA
`-i input.mp4`	Input file
`-c:v h264_nvenc`	NVIDIA H.264 codec (NVENC)
`-preset p6`	Balance between speed and quality
`-an`	Remove audio track
`output_h264.mp4`	Output file

Table 4.1: Parameters for re-encoding the video stream with NVIDIA NVENC.

4.1 Decoding the Video Stream and Initial Frame Filtering

Decoding of the H.264 stream is performed via PyAV, which returns frames with the original timestamp. The average frame rate f_{avg} is extracted from the container metadata and used to compute the sampling step $k = \lfloor f_{\text{avg}}/\text{frame\_rate} \rfloor$, ensuring processing of approximately one frame per second. Significant content changes are numerically determined by the mean absolute difference

$$D_t = \frac{1}{3HW} \sum_{c=1}^{3} \sum_{x=1}^{W} \sum_{y=1}^{H} |I_t(x,y,c) - I_{t-1}(x,y,c)|\,, \tag{1}$$

where I_t and I_{t-1} are the current and previous frames normalized to $[0,1]$. A frame is considered a key frame when $D_t > \theta_{\text{diff}}$ (H. Zhang et al., 1995), indicating a reliable scene change.
Selected key frames undergo quality checks: for the grayscale image $I^{(\text{gray})}$, the Laplacian variance $\sigma_L^2 = \text{Var}(\nabla^2 I^{(\text{gray})})$, the mean brightness

$$B = \frac{1}{HW} \sum_{x,y} I^{(\text{gray})}(x,y),$$

and the contrast $\sigma_I = \sqrt{\text{Var}(I^{(\text{gray})})}$ are computed. Frames with $\sigma_L^2 < \theta_{\text{sharp}}$, $B \notin [B_{\min}, B_{\max}]$, or $\sigma_I < \theta_{\text{contr}}$ are excluded as blurred or overexposed. This filter ensures that only informative and technically correct frames are retained for subsequent processing.

4.2 Local Contrast Enhancement using CLAHE

To equalize the local contrast of key frames, the CLAHE method is applied in the LAB color space with parameters `clipLimit=2.0` and `tileGridSize=8×8` (Zuiderveld, 1994). Conversion to the L channel, its adaptive equalization, and back-conversion to RGB redistribute luminance values in the range $[0, 255]$, as shown by the histograms in Fig. 4.1. Visual examples of original frames and CLAHE results combined with SNR filtering are presented in Fig. 4.2.

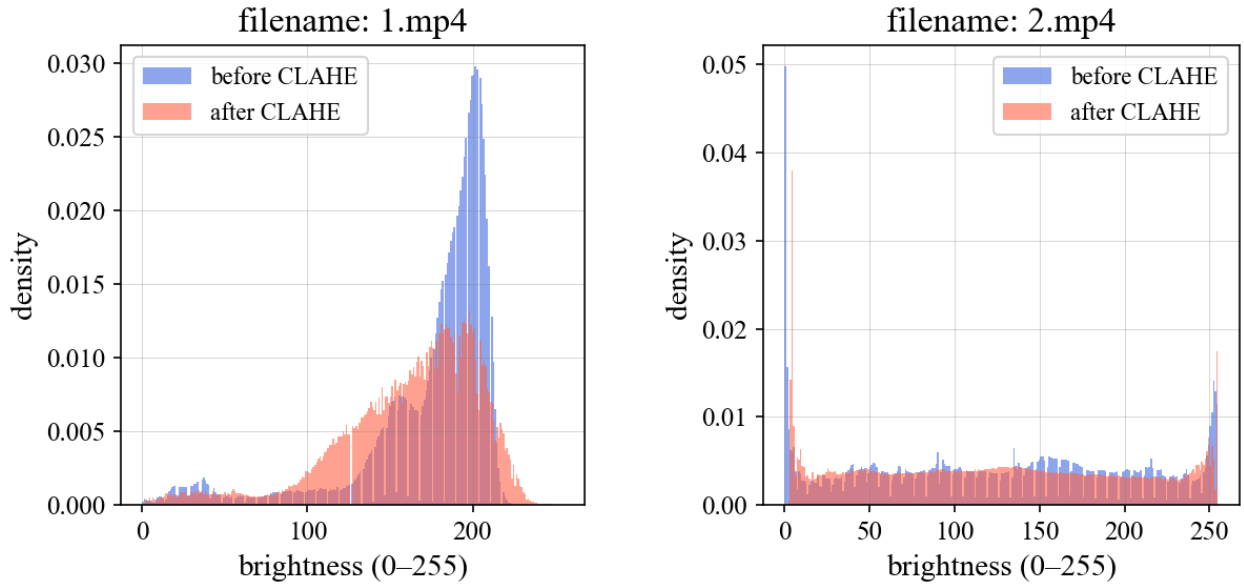


Figure 4.1: Luminance histograms of key images before and after applying CLAHE (see § 4.2).

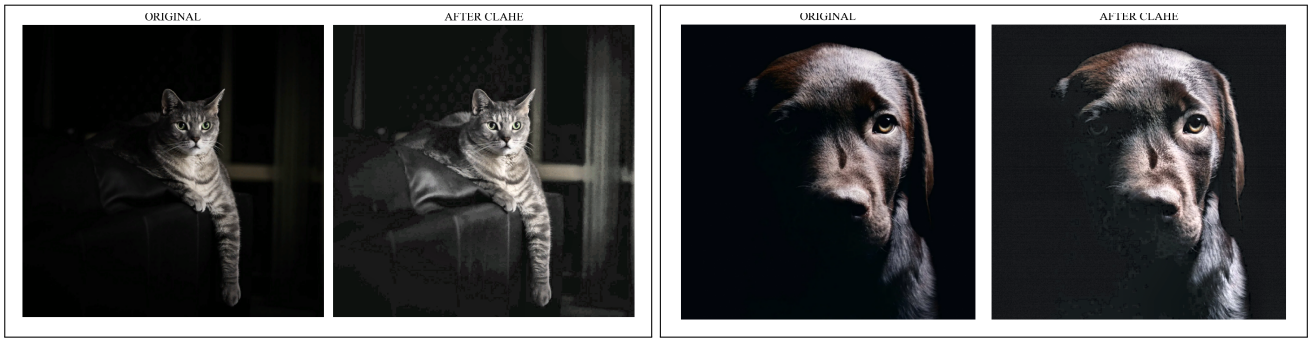


Figure 4.2: Examples of key frames before and after CLAHE and SNR filtering.

4.3 Filtering Out "Redundant" Frames and Object Presence Verification

Each key frame undergoes validation according to three image quality criteria. First, the frame is converted to grayscale $I^{(\text{gray})}$, and the Laplacian variance

$$\sigma_L^2 = \operatorname{Var}\big(\nabla^2 I^{(\text{gray})}\big),$$

which serves as a focus measure and reflects the level of edge detail in the frame (**pech-pacheco2000**), is computed. Frames for which $\sigma_L^2 < \theta_{\text{sharp}}$ are discarded as insufficiently sharp. Next, the brightness standard deviation

$$\sigma_I = \sqrt{\frac{1}{HW}\sum_{x,y}\big(I^{(\text{gray})}(x,y) - B\big)^2},$$

reflecting the image contrast, is determined. The values B and σ_I are compared against thresholds θ_{bright} and θ_{contrast}, respectively; frames outside the acceptable ranges (under- or overexposed, low-contrast) are excluded from further processing.

On the remaining frames, object presence is checked using a single-stage YOLOv8 model, pre-initialized with the `yolov8n.pt` weights. It provides an optimal balance of accuracy and speed: mAP@0.5 = 0.62 at 769 FPS on GPU with 11.2 M parameters (1.3 ms/image), see Table 4.2. The YOLOv8 architecture builds on CSP Darknet backbone improvements and a decoupled head, offering more precise localization with low computational cost. The input module fully supports anchor-free detection, and FPN/PAN connections in the neck enhance small-object representations (Person ID) through multi-scale feature aggregation.

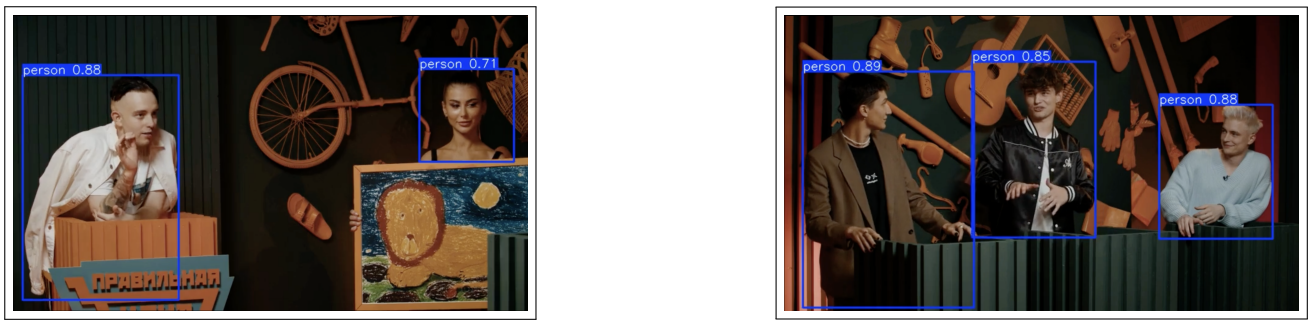


Figure 4.3: Examples of "person" class detections by the YOLOv8 model on key frames.

For each object, the class and confidence score conf $\in [0, 1]$ are computed. An object is

considered present if at least one bounding box with the corresponding label (class ID) is found and conf $\geq \theta_{\text{object}}$ (Jocher et al., 2023). Frequency analysis of objects filters out frames with rare items, which helps subsequently analyze only compositionally relevant images containing the target object.

Model	mAP@0.5	FPS (GPU)	Params (M)	Speed (ms/image)
YOLOv8	**0.62**	**769**	11.2	~ 1.3
Faster R-CNN	0.41	18	42.0	~ 55.6
SSD	0.47	46	34.3	~ 21.7
EfficientDet	0.47	70	12.0	~ 14.3
Mask R-CNN	**0.77**	5	44.5	~ 200

Table 4.2: Comparison of object detection models by accuracy and speed metrics.

4.4 Clustering and Selection of the Most Expressive Frames

For each filtered frame f_i, a feature vector $\mathbf{e}_i$ is computed using a pretrained ResNet-18 network (global pooling layer, dimension 512) (He et al., 2016) or the OpenCLIP image embedding model (dimension 512) (Radford, J. W. Kim, et al., 2021b). The embeddings are L_2-normalized:

$$\hat{\mathbf{e}}_i = \frac{\mathbf{e}_i}{\|\mathbf{e}_i\|_2}.$$

The cosine similarity between frames i and j is then

$$s_{ij} = \hat{\mathbf{e}}_i^\top \hat{\mathbf{e}}_j.$$

After computing s_{ij} for all frame pairs i, j, we assemble the similarity matrix $S = [s_{ij}]_{i,j=1}^{N}$. Frames are clustered using DBSCAN with distance metric $d_{ij} = 1 - s_{ij}$ and parameter $\varepsilon = 1 - \tau_{\text{clust}}$ (Ester et al., 1996). This approach adapts to the embedding distribution and does not require specifying the number of clusters in advance. An alternative greedy algorithm selects centroids by sorting frames in descending order of $\sum_j s_{ij}$ and then adding to each cluster any frames whose similarity to its centroid exceeds τ_{clust}.

Denote the resulting clusters by $C_1, \ldots, C_K$, where

$$C_k = \{\, i : \text{label}(i) = k \}, \quad \bigcup_{k=1}^{K} C_k = \{1, \ldots, N\}.$$

Within each cluster C_k, select the index

$$i_k^* = \arg\max_{i \in C_k} \sigma_L^2(f_i),$$

where $\sigma_L^2(f)$ is the Laplacian variance of the frame's grayscale image, serving as a sharpness measure (Pech-Pacheco et al., 2000). Thus, from each semantically homogeneous set, the sharpest frame is retained, maximally conveying the scene's visual information. Clustering combined with sharpness selection balances redundancy removal (repeated shots) with preservation of expressive content.

4.5 Matting (Background Removal)

The foreground segmentation module is implemented using a pretrained MODNet model from its public repository. For each filtered frame f_i, inference produces a semi-transparent mask $\alpha_i(x, y) \in [0, 1]$. The MODNet architecture combines an e-ASPP receptive field expansion with optimized convolutional blocks to achieve high accuracy in object contour separation at around 67 FPS on a single GPU (Z. Liu et al., 2021). The resulting mask is saved in PNG format with an alpha channel. For visual background removal, we apply thresholding $\alpha_i(x, y) \geq \tau_\alpha$, where τ_α is typically chosen empirically as 0.2.

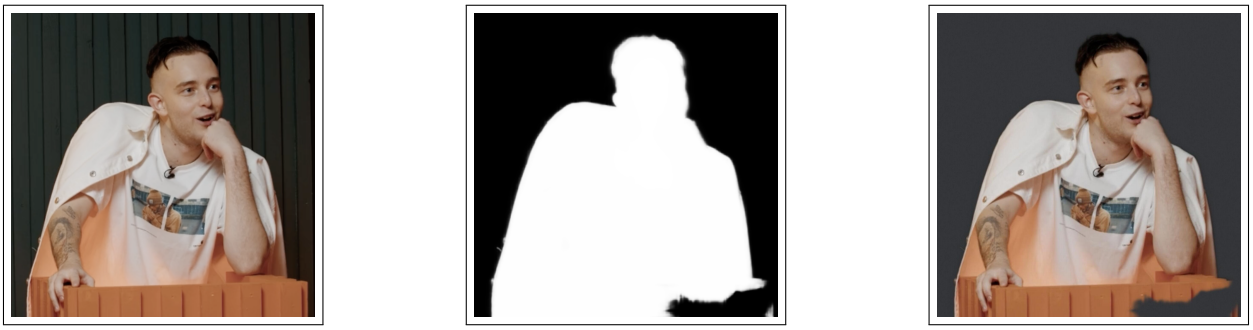


Figure 4.4: Examples of matting a key frame and replacing the background.

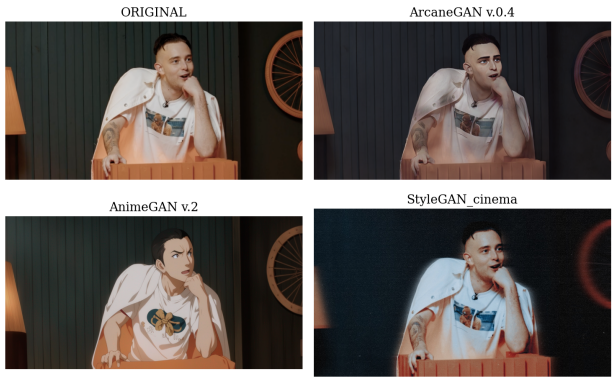
The output of this stage is a set of binary and semi-transparent masks that separate foreground objects from the background. Subsequently, the user will be given the option to "replace" the background.

4.6 Stylization of Key Frames Using GAN

To modify the visual style of key frames, three generative adversarial network (GAN) architectures are employed: ArcaneGAN v0.4 (Y. Zhang et al., 2022), AnimeGAN v2 (Y. Wang et al., 2021), and StyleGAN (H. Kim et al., 2020).The selection of these models is largely due to their practicality, small model sizes, and frame processing speed. ArcaneGAN v0.4 uses residual blocks with multi-scale features for artistic style transfer. AnimeGAN v2 is optimized with perceptual losses and gradient constraints, which helps maintain the clarity of character outlines. StyleDAN introduces adaptive normalization layers (AdaLIN) and an attention mechanism for selective style transfer without distorting color transitions.

Inference for all models is performed on a GPU using PyTorch; the average processing time per frame is 12–18 ms, enabling integration of stylization into a real-time pipeline. For each frame f_i, three stylized variants $\hat{f}_i^{(\text{arc})}$, $\hat{f}_i^{(\text{anime})}$, and $\hat{f}_i^{(\text{uga})}$ are generated.

(a) ArcaneGAN v0.4 and AnimeGAN+Cinema



(b) StyleDAN v0.4 and StyleDAN+Cinema

Figure 4.5: Examples of key frame stylization

5 Custom Implementation of the Key Frame Detector

Selection of Training Data. The TVSum–50 dataset (Song, Castrejon, and Jaimes, 2015) is used to validate the key frame detector. The collection consists of 50 video clips grouped into ten thematic scenes (five videos per category), providing even coverage of domestic, sports, and social topics. Summary statistics are shown in Table 5.1.

Metric	Value	Notation
Number of videos	50	N
Total duration	$1\,h\,19\ min$	$\sum_{v=1}^{N} T_v$
Frame rate	$25\ fps$	f
Shot length	$\tau = 2\ s$	—
Average # of shots	≈ 49	$\bar{K}$
Rating scale	$s_{i,j} \in \{1, \ldots, 5\}$	—

Table 5.1: Key characteristics of TVSum–50

Each video is divided into equal-duration shots $\{\mathcal{S}_i\}_{i=1}^{K_v}$ with $|\mathcal{S}_i| = \tau$. Twenty independent annotators assigned each shot an integer score $s_{i,j}$. In our case, the average

$$\bar{s}_i = \frac{1}{M} \sum_{j=1}^{M} s_{i,j}, \quad M = 20,$$

where M is the number of annotators.

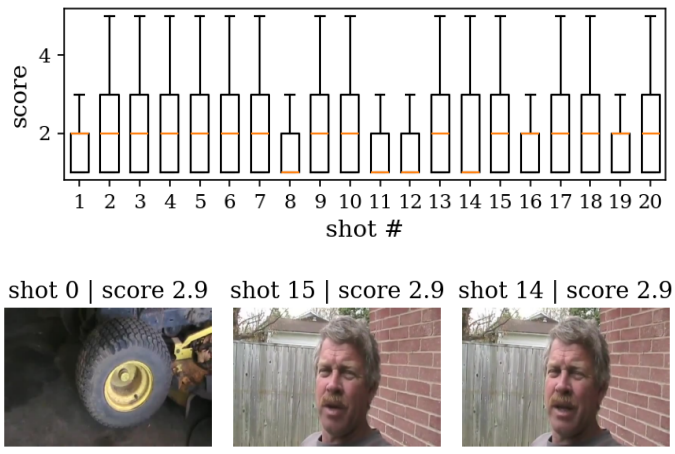
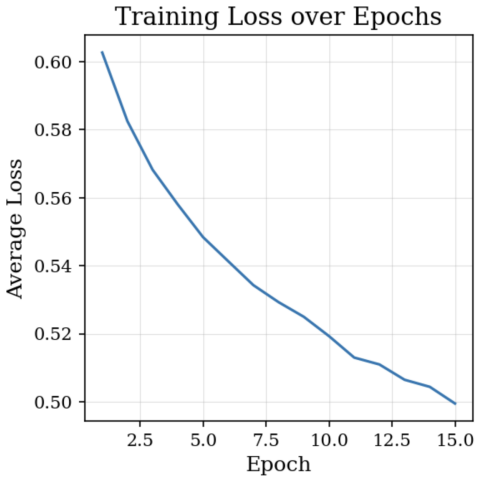


Figure 5.1: Distribution of frame ratings and top-3 frames: `AwmHb44_ouw.mp4`

We observe that for each video we can compile the distribution of ratings as shown in Fig. 5.5. It was also calculated that only $\approx 15\%$ of shots have $\bar{s}_i \geq 4$, which may lead to class imbalance when binarizing into "key/non-key." In any case, the availability of a common metric (summary-level F_1 at a 15% duration budget) allows us to compare our results with published works, which is convenient.

5.1 Baseline Experiment: "Frozen" ResNet-50 and Logistic Head

For an initial hypothesis test, we implemented the simplest possible pipeline: from the pre-extracted frames we construct the `TVSumDataset`; all layers of ResNet-50 pretrained on ImageNet are frozen, and the final fully connected layer is replaced by a single-neuron logistic head. The classification threshold was set to $\tau = 0.5$.



(a) Training Loss Dynamics

Metric	Value
Accuracy	0.771
Precision	0.717
Recall	0.185
F_1	0.294

(b) Quality Metrics on the Entire Dataset ($\tau = 0.5$)

Review. The model demonstrates decent precision (precision ≈ 0.72): most frames labeled as "key" are indeed important. However, recall is only 18%, which leads to low coverage of informative shots and, consequently, $F_1 = 0.29$. In this case, we face class imbalance: only $\approx 15\%$ of shots per video belong to the positive class, and a binary BCE loss without weight adjustment does not incentivize the model to detect these rare examples. The fixed threshold $\tau = 0.5$ also contributes to this issue. We should consider unfreezing `layer4`, since otherwise the model trains only a 2048×1 linear classifier, which may be insufficient for semantically separating diverse scenes. We must also split out a validation set.

5.2 Partially Fine-Tuned ResNet-50 with Class Balancing

After the baseline experiment, a minimal but justified set of improvements was implemented.

Feature Space Adaptation. The base ResNet-50 remains almost unchanged: only the weights of the fourth convolutional block $\theta_{l4} \subset \Theta$ are left trainable. Overall, this reduces the number of trainable parameters from 25 million to approximately 2.5 million.

Class Imbalance Compensation and Generalization. The proportion of key shots is only $\simeq 15\%$, so in the training set we apply weighted stochastic sampling with class weight ratio $w_0 : w_1 \approx 6 : 1$, ensuring equal-probability presentation of positives and negatives. In addition, light augmentation is introduced (`RandomResizedCrop(224)` and horizontal flip), while valida-

tion is performed on a fixed center crop.

Stable Training. Classifier parameters are trained with learning rate $\eta_{fc} = 10^{-3}$, whereas the unfrozen block uses a reduced rate of $0.5\,\eta_{fc}$. Every five epochs, the learning rate is halved. The model is checkpointed at the best validation F_1 score with a patience of three epochs to prevent overfitting.

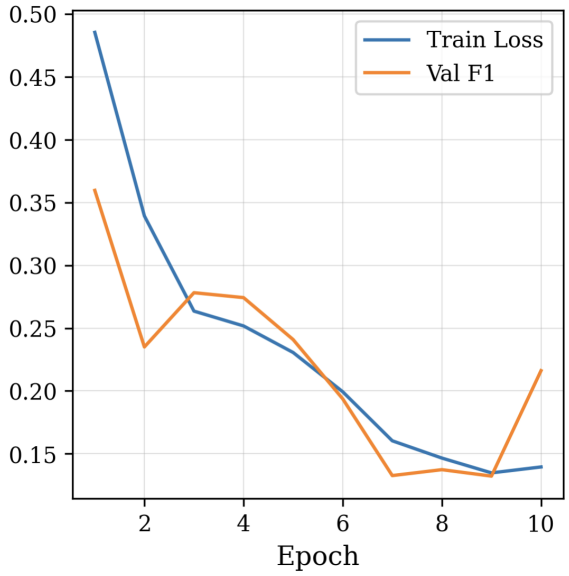


Figure 5.3: 10 epochs

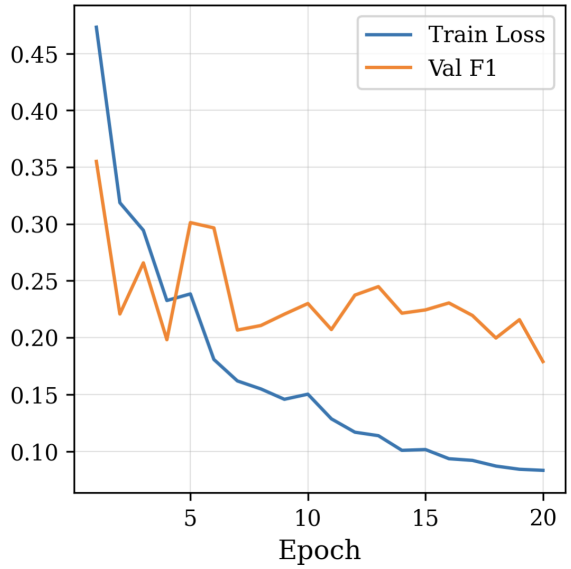


Figure 5.4: 20 epochs

The optimum is reached by the 7th epoch:$F_1^{\text{val}} = 0.31$. Extending training to 20 epochs does not improve the metric, confirming the onset of overfitting in `layer4`. Compared to the fully frozen network, an improvement of $+$**7%** is achieved without changing the loss function and without accounting for temporal context.

5.3 Modeling Temporal Structure: Bidirectional LSTM

To overcome the limitation of frame-wise classification without context, a lightweight recurrent head was added, capable of modeling narrative smoothness and local transitions between adjacent shots. We describe it in more detail below. In the proposed approach, in the first stage a feature vector is extracted from each frame x_t:

$$f_t = \varphi_{\text{ResNet50}}(x_t) \in \mathbb{R}^{2048},$$

where $\varphi_{\text{ResNet50}}$ is a pretrained ResNet-50 without the fully connected output layer. The sequence $\{f_t\}_{t=1}^{T}$ is fed into a bidirectional LSTM:

$$\overrightarrow{h}_t = \overrightarrow{\text{LSTM}}(f_t, \overrightarrow{h}_{t-1}), \quad \overleftarrow{h}_t = \overleftarrow{\text{LSTM}}(f_t, \overleftarrow{h}_{t+1}), \quad h_t = \left[\overrightarrow{h}_t; \overleftarrow{h}_t\right] \in \mathbb{R}^{2H},$$

after which for each time step the logits and probabilities are computed:

$$\ell_t = W\, h_t + b, \qquad p_t = \sigma(\ell_t),$$

where σ is the sigmoid function, $W \in \mathbb{R}^{1\times 2H}$, and $b \in \mathbb{R}$. Training was performed using the binary cross-entropy with logits loss:

$$\mathcal{L} = -\frac{1}{N}\sum_{n=1}^{N}\sum_{t=1}^{T_n}\big[y_{n,t}\ln p_{n,t} + (1-y_{n,t})\ln(1-p_{n,t})\big].$$

In the experiment, the TVSum-50 dataset was split by video into 70%/15%/15% for train/validation/test:

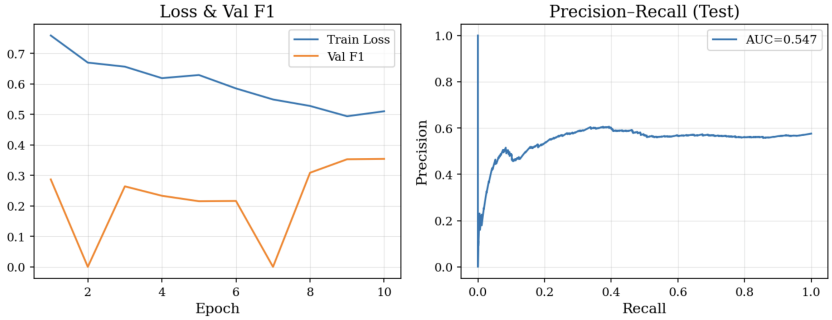


Figure 5.5: Loss, F_1, and PR-curve

After 10 epochs, the best validation performance reached

$$F_1^{\text{val}} = 0.3405,$$

while on the final test set the bidirectional LSTM head achieved

$$F_1^{\text{test}} = 0.481, \qquad \text{PR-AUC} = 0.5466.$$

This indicates that accounting for local temporal dependencies significantly increased recall while maintaining acceptable precision, resulting in a twofold higher F_1 compared to the best purely frame-based solution. The increase in PR-AUC reflects the overall improvement in the model's ability to distinguish key and background shots across different thresholds.

Thus, integrating the bidirectional LSTM allowed the capture of video stream dynamics without significantly complicating the backbone, demonstrating a substantial metric gain with minimal additional parameters.

5.4 Headline Generation with Russian T5 Models

Headline generation is a special case of abstractive summarization, where a short headline (usually a single sentence) must be generated from the full article/text. We consider the T5 family of models (Transformer-based seq2seq) for Russian, fine-tuned to generate headlines. In particular, we compare our fine-tuned model `cointegrated/ruT5` on the Lenta.ru corpus (Yutkin, 2020). The dataset covers five main topics: *Politics*, *Society*, *Economics*, *Science*, and *Sports*. After filtering by the condition $|\text{headline}| \leq 51$, we obtained 78,497 text–headline pairs. The split was performed in a 70%/15%/15% ratio for train/val/test. Quality was evaluated using ROUGE-L, BLEU, and perplexity (PPL). For the experiment, we further divided the initial corpus based on headline length. Thus, we identified three training data groups:

Group	Condition	Size	Mean	σ
7 words	≤ 51 tokens	78,497	43.2	6.28
9–11 words	> 65 tokens	35,832	70.4	4.25
11+ words	> 80 tokens	20,590	82.9	4.23

Table 5.2: Corpus split by headline length (in tokens).

Hypothesis. Restricting the target length during seq2seq model training encourages the generation of more concise headlines without losing informativeness by shifting the output distribution toward compact sequences.

Below is the plot of the loss during fine-tuning of `cointegrated/ruT5` on the headline generation task (epochs 1–30), see Fig. 5.6.

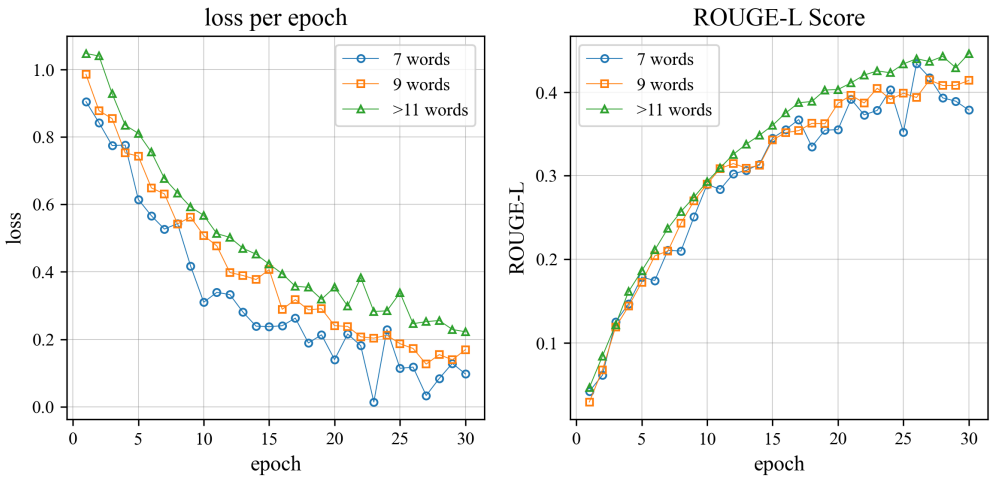


Figure 5.6: Loss of our fine-tuned `cointegrated/ruT5` on Lenta.ru (epochs 1–30) and ROUGE-L score

During fine-tuning, the loss steadily decreases from over 1.0 to below 0.25 by the 30th epoch.

The model trained on headlines longer than 11 words exhibits the best convergence—its loss stabilizes at a minimal level already after the 15th epoch. The 7-word configuration shows the largest fluctuations. At the same time, ROUGE-L on the validation set increases most markedly for the long-headline model (ROUGE-L > 0.44 by the final epoch).
Now we compare these results with two other models from the same family:

Model	ROUGE-L	ROUGE-2	BERTScore	PPL
cointegrated/ruT5 (51 tokens)	**40.7**	24.9	81.2	9.6
wanderer-msk/ruT5	39.4	**36.8**	**88.4**	8.1
IlyaGusev/rut5$_s um$	33.2	22.0	84.6	**7.8**
Pointer-Generator RNN	28.5	10.3	66.2	—

Table 5.3: Headline generation quality with a 51-token limit.

Metrics. We evaluate ROUGE-L, ROUGE-2, BLEU-4, and BERTScore. Perplexity (PPL) is reported only for T5-family models as an internal diagnostic.

Training Dynamics. Figure 5.6 shows the loss and ROUGE-L curves for three maximum-output-length configurations (25, 35, 51 tokens). The 51-token setting converges more slowly but achieves a 3–4 point higher ROUGE-L after epoch 20, confirming the importance of utilizing the full allowed token space.

Model Selection Rationale. Although the fine-tuned `cointegrated/ruT5` does not reach the absolute peak metrics (see Table 5.3), its use is justified: **Compactness:** at 0.9 GB with a 30,000-token vocabulary, it imposes low memory load and delivers stable inference latency, while matching other models on ROUGE-L. In the next subsection, we will examine its performance after applying our deduplication algorithm.

5.5 Post-processing and Final Headline Selection

Our generative model (`fine-tuned-cointegrated/ruT5`), along with the two other mentioned T5-family models, produces 10 headline variants based on the source text. To obtain diverse outputs, we vary the generation parameters, primarily the `length_penalty` in the range $[0.1, 0.2, \ldots, 1.0]$. Then, for the original document D and each generated headline h_i, we compute embeddings using a combination of models: BERT (Devlin et al., 2019b), RoBERTa (Y. Liu et al., 2019), and E5 (Reimers and Gurevych, 2022) (denoted E_1, E_2, and E_3, respectively). To obtain a single representation, we apply a weighted sum of embeddings:

$$\mathbf{v}(x) = \alpha_1 E_1(x) + \alpha_2 E_2(x) + \alpha_3 E_3(x),$$

where weights $\alpha_k = \frac{1}{3}$ are chosen empirically (A. Wang et al., 2019). Next, we compute the cosine similarity between each headline vector and the document vector:

$$S_i = \cos\big(\mathbf{v}(h_i),\, \mathbf{v}(D)\big).$$

Based on the obtained S_i, candidate headlines are clustered to identify semantic duplicates. A threshold $\tau \approx 0.85$ on cosine similarity is used(T. Zhang et al., 2020): if for a pair (h_i, h_j) the value

$$\cos\big(\mathbf{v}(h_i), \mathbf{v}(h_j)\big) \geq \tau,$$

then these two variants are considered duplicates. Clustering follows a "master/slave" principle: the most relevant headline in the cluster (with the highest S_i relative to the document) is designated the master, and the other similar variants are marked as duplicates (slaves) and discarded. This post-processing algorithm, which we call *GuidMaster*, is conceptually similar to the Maximal Marginal Relevance scheme(Carbonell and Goldstein, 1998), aimed at reducing redundancy while preserving relevance.

After duplicate removal, several unique candidate headlines remain. In the final stage, a named entity recognition (NER) module based on the `bert-base-ner` model(Lample et al., 2016b) is applied to extract personalized entities (persons, organizations, locations, etc.) from each headline's text. The remaining headlines are sorted by the number of unique entities detected (in descending order). It is assumed that a headline covering more key entities from the source document conveys richer information about the news content(Lample et al., 2016b).

In addition to the final selected headline, the following alternative variants are generated for

comparison:

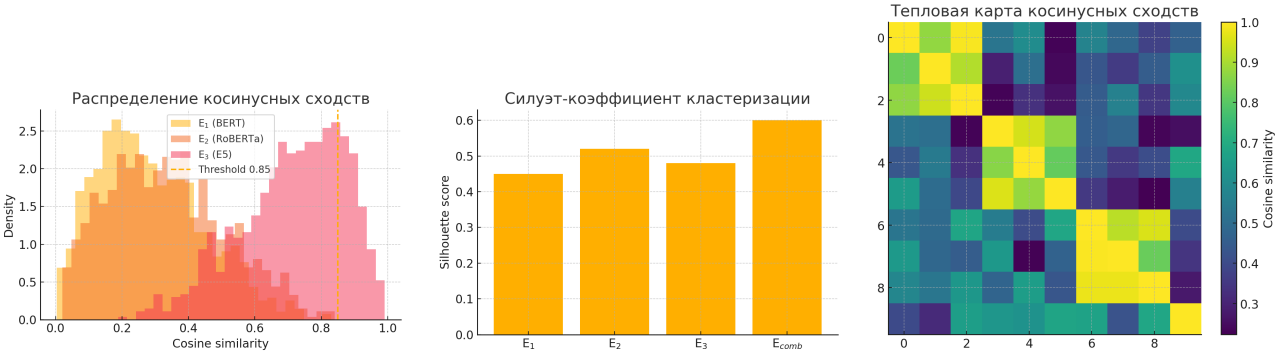
(a) The headline produced by the RuT5-base model: (`wanderer-msk/ruT5-base_headline_generation`);

(b) A one-line abstractive summary of the source text, generated by the summarization model (`IlyaGusev/rut5_base_sum_gazeta`, using the first line).

Headline Variant	**ROUGE-L**	Length (words)	Unique NERs	Cosine Similarity
Proposed (final)	0.257	6	3.24	0.84
Alternative (RuT5-base)	0.233	11	3.91	0.86
Alternative (summarization)	0.198	14	5.19	0.75

Table 5.4: Comparison of headline candidates by qualitative metrics: ROUGE-L score, length in words, number of unique named entities (NER), and average cosine similarity to the source text. Number of texts: 1000. Dataset: LENTA.

Table 5.4 demonstrates that the final headline selected by GuidMaster achieves the highest ROUGE-L score and the greatest coverage of named entities, while its length remains comparable to the alternatives. Its cosine similarity to the source text is only slightly lower than that of the alternative variant.



(a) Cosine similarity distribution (b) Silhouette coefficients (c) Similarity heatmap

Figure 5.7: Clustering quality and similarity metrics for $n = 1000$ headlines.

The distribution of cosine similarities between all pairs of the 1000 generated headlines is shown in Fig. 5.7a. The clustering quality (silhouette coefficient) for various embeddings is presented in Fig. 5.7b. The heatmap of pairwise cosine similarities for all 10^3 headline variants is displayed in Fig. 5.7c.

As a result, we have obtained a modular algorithm that allows the user to choose one of the three proposed headlines.

5.6 Thumbnail Generation via Subject–Predicate Extraction and Stable Diffusion

From the three headline variants generated by the T5 models, full syntactic subject noun chunks are extracted using spaCy (Honnibal and Montani, 2017): for each headline h_i, the noun_chunk whose root is the lemma of the subject s_i (e.g., "ceremony" → "tea ceremony") is identified. Then, using Pymorphy2 (Korobov, 2015), s_i is lemmatized and the verb root v_i is extracted. As a result, the pair $\langle s_i, v_i \rangle$ is formed.

Table 5.5: Examples of extracting the subject–predicate syntactic pair

Headline	Subject s_i	Verb v_i	Pare $\langle s_i, v_i \rangle$
Человек сидит за ноутбуком и программирует	человек	сидеть	⟨человек, сидеть⟩
Чайная церемония — неотъемлемая часть китайской культуры	чайная церемония	часть	⟨чайная церемония, часть⟩

Each pair is then translated into English using the MarianMT model `Helsinki-NLP/opus-mt-ru-en` (Junczys-Dowmunt et al., 2018) and used as the basis for prompts in the Stable Diffusion 3 Medium generative pipeline (Diffusers) (Rombach et al., 2022). Each prompt is supplemented with artistic descriptors and a negative filter:

`NEG_PROMPT` = "no objects touching edge, no border artifacts".

In our experiments, the following structural prompt templates yielded the most expressive results:

1. `"a {object}, with a golden glow, black background, shrouded in a misty, eerie fog that swirls around the base"`,
2. `"a {object}, with a soft halo of ambient light, set against a deep charcoal background, enveloped in swirling smoky tendrils and subtle bokeh highlights"`

and 3. `"a {object}, illuminated by neon backlight, surrounded by drifting ember-like particles and faint lens flares, on a dark gradient backdrop"`.

Inference is performed with `guidance_scale`=10 and `num_inference_steps`=40, providing a balance between prompt adherence and artistic detail preservation. For automatic evaluation of the synthesized thumbnails, we use the CLIP-score, defined as the cosine similarity between

image and text embeddings (Radford, J. W. Kim, et al., 2021c). A threshold of $\tau_{\text{clip}} = 0.28$ was chosen, which empirically ensured high semantic relevance without excessively filtering out visually high-quality results.

5.7 Expansion of a Square Image without Content Distortion

Let $I\colon [0, W]\times[0, H] \to \mathbb{R}^3$ be the source square image of width W and height H. It is necessary to obtain an image of width $W + E$ without nonlinear distortion of the content. To this end, the following operations are introduced. First, a narrow vertical strip of width $s \ll W$ is extracted:

$$L(u, y) = I(u, y), \quad u \in [0, s],\ y \in [0, H].$$

Next, the strip is scaled horizontally to width E using bicubic interpolation (LANCZOS):

$$S(x, y) = L\Big(\tfrac{s}{E}\, x,\ y\Big), \quad x \in [0, E].$$

Second, a gradient mask is constructed:

$$M(x, y) = G_{\sigma_m}\big(1 - \tfrac{x}{E}\big), \quad G_{\sigma_m} \otimes f — \text{ Gaussian filter with } \sigma = \sigma_m,$$

where before convolution the function $1 - x/E$ decreases linearly from 1 at $x = 0$ to 0 at $x = E$. Third, the expanded content is formed as

$$C(x, y) = M(x, y)\, S(x, y) + \big(1 - M(x, y)\big)\, \mathbf{0},$$

where $\mathbf{0}$ denotes a black background. Fourth, the canvas is assembled:

$$\widetilde{I}(x, y) = \begin{cases} C(x, y), & x \in [0, E), \\ I(x - E, y), & x \in [E,\ E + W]. \end{cases}$$

Fifth, to smooth the seam at the boundary $x = E$, a local blur is applied:

$$\widetilde{I}_{\text{blur}}(x, y) = \begin{cases} \big(G_{\sigma_s} \otimes \widetilde{I}\big)(x, y), & x \in \big[E - \tfrac{w_s}{2},\ E + \tfrac{w_s}{2}\big], \\ \widetilde{I}(x, y), & \text{otherwise}, \end{cases}$$

where w_s is the seam width and σ_s is the blur parameter. Finally, to eliminate banding, a small

additive noise is added:

$$I_{\text{out}}(x,y) = \text{clip}\big(\widetilde{I}_{\text{blur}}(x,y) + \eta(x,y),\, 0, 255\big), \quad \eta(x,y) \sim \mathcal{N}(0, \sigma_n^2).$$

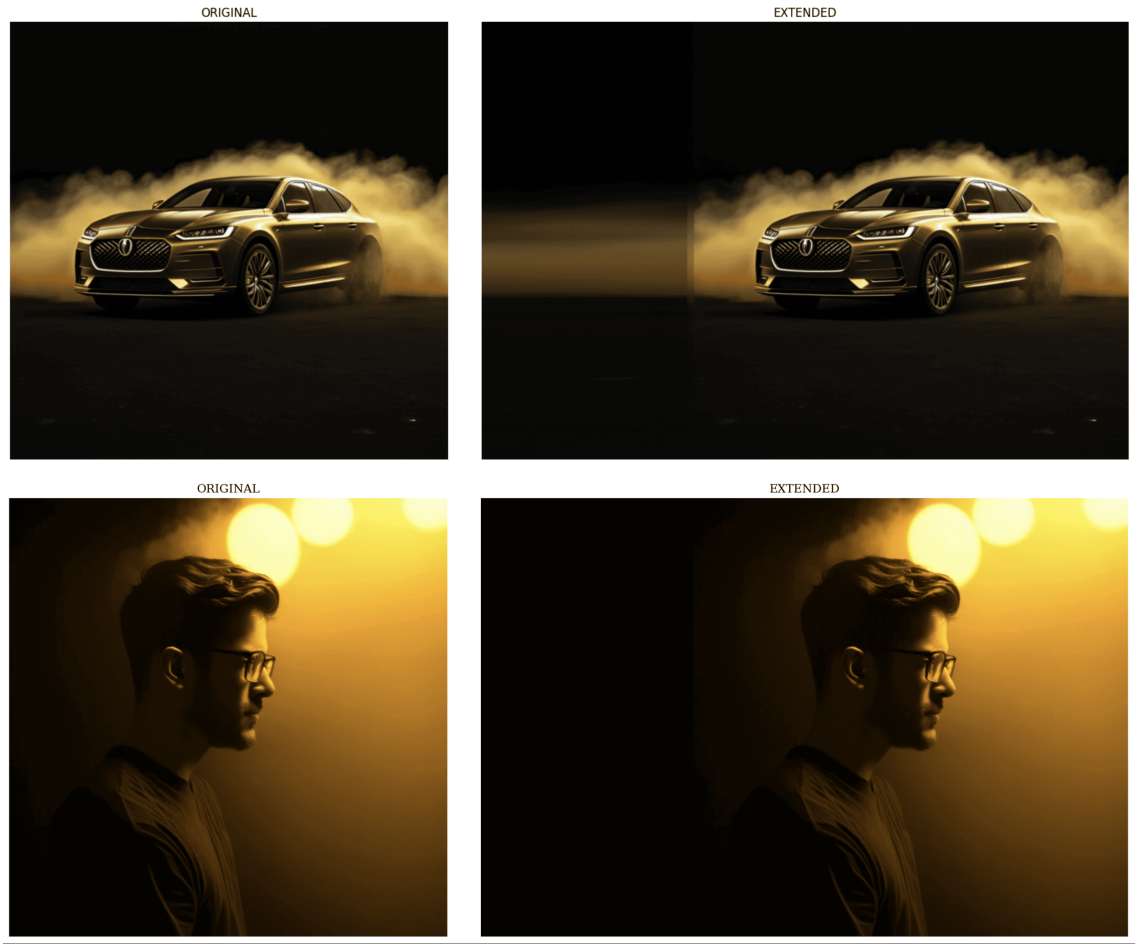


Figure 5.8: Examples of generation in SD3 for prompts {programmer} and {car}

Thus, the expansion is performed by reusing a fragment of the original image without stretching it, the visual integrity of objects is preserved, and the boundary of compositions is neatly smoothed and duplicate artifacts are masked by noise.

6 System from the User's Perspective

This section describes the sequence of user interactions with the automatic thumbnail and headline generation tool—from uploading the source video to obtaining the final thumbnail with overlaid text: After launching the script (or accessing the web interface), the user performs the following steps:

1. **Video Upload.** The user specifies a local video file and initiates the analysis. The system detects the codec and, if necessary, performs transcoding, then extracts key frames and generates three headline variants.

2. **Key Frame Selection.** All extracted frames are available in a scrollable list. The user marks the preferred frames, forming a set of thumbnail candidates.

3. **Stylization via GAN.** For each selected frame, three stylized variants are offered (ArcaneGANv0.4, AnimeGANv2, UGATIT). The user evaluates the visual effects and selects the preferred style.

4. **Additional Generation.** If necessary, a new set of images is generated using Stable Diffusion 3 Medium based on the syntactic pair $\langle s_i, v_i \rangle$. The user can set or adjust the text prompt to obtain more relevant results.

5. **Text Overlay.** From the proposed headlines and noun_chunk, text options are automatically assembled. The user chooses the font, color, and position of the text on the image.

6. **Download.** After finalizing the thumbnail with text, the user saves the image in PNG or JPEG format.

For a clearer representation of each step, the following diagram is proposed:

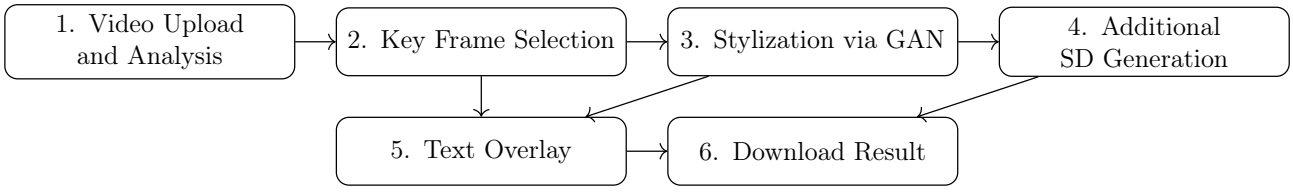


Figure 6.1: User Flow: from video upload to downloading the final thumbnail

Component	Disk Size	VRAM	Inference Time	CPU Cores	Rec. VRAM
YOLOv8n	6.5 MB	0.1 GB	1.3 ms/fr	2 cores	1 GB
MODNet	309 MB	1.0 GB	15 ms/fr	4 cores	2 GB
Stable Diffusion 3 Medium	2.7 GB	8.0 GB	2.5 s/img (40 steps)	4 cores	16 GB
cointegrated/ruT5-base	907 MB	2.5 GB	0.1 s/hdl	2 cores	4 GB
wanderer-msk/ruT5-base_hg	892 MB	2.5 GB	0.1 s/hdl	2 cores	4 GB
IlyaGusev/rut5_bsg	978 MB	2.5 GB	0.1 s/summ	2 cores	4 GB
Helsinki-NLP/opus-mt-ru-en	307 MB	1.0 GB	20 ms/phr	4 cores	2 GB
openai/whisper-base	296 MB	3.0 GB	0.2 s/s mp3	4 cores	4 GB
fine-tuned model (.pth)	**42 MB**	**2.5 GB**	**0.1 s/fr**	**2 cores**	**4 GB**

Table 6.1: Specifications, inference times, and resource requirements for all key models on NVIDIA Tesla A100 (80 GB VRAM).

Next, we present a table containing information on the execution time of each stage. All training processes requiring a GPU, inference runs, and pipeline execution were performed on an NVIDIA Tesla A100 (80 GB VRAM).

Step	30 s	1 min	5 min	15 min	30 min
Keyframe extraction	0.65 ± 1.43	1.33 ± 4.25	6.67 ± 6.08	20 ± 8.12	40 ± 12.15
Headline generation (30 total)	20 ± 0.5	21 ± 1.24	24 ± 2.50	26 ± 3.01	34 ± 8.8
Deduplication (const)	38 ± 3.40	38 ± 3.40	38 ± 3.40	38 ± 3.40	38 ± 3.40
SD3 cover synthesis (3×30 s each)	90 ± 15.5	90 ± 15.5	90 ± 15.5	90 ± 15.5	90 ± 15.5
Total processing time	151.65 ± 2.0	151.33 ± 2.1	156.67 ± 2.2	170 ± 2.5	190 ± 3.0

Table 6.2: End-to-end processing times for each pipeline step at different video lengths (in seconds).

7 Conclusion

In this paper, we propose an autonomous platform for generating video covers and titles by combining convolutional keyframe extraction, GAN styling, diffusion synthesis, and NLP models. Alternative existing solutions work only partially: YouTube selects frames based on brightness and contrast, TikTok tracks peak activity, Vimeo offers templates for manual tweaking. None of these automate text generation, apply deep styling, or ensure semantic consistency between the visuals and the headline. Our system first highlights keyframes taking into account sharpness, scene changes, and detection of target objects (e.g., people), including the option to create multiple visuals using ArcaneGAN, AnimeGAN, and StyleGAN. A background image generated via Stable Diffusion is augmented as needed. In parallel, the three RuT5 models generate relevant titles and their relevance to the visual is evaluated using CLIP-score. Most of the arguments have statistical support; they are reported in the articles. The user can choose the optimal variant, adjust the font, color and text layout and save the ready cover locally, without using third-party services.

Thus, the proposed architecture combines some of the best practices of existing platforms and models and takes them to the next level by deeply automating all the steps.

References

[1] Jaime Carbonell and Jade Goldstein. "The Use of MMR, Diversity-Based Reranking for Reordering Documents and Producing Summaries". In: *Proceedings of SIGIR*. 1998, pp. 335–336.

[2] Nicolas Carion et al. "End-to-End Object Detection with Transformers". In: *ECCV*. 2020.

[3] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. "DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs". In: *arXiv preprint arXiv:1606.00915*. 2017.

[4] Ying Chuang and et al. "Fast Video Segmentation Based on Change Detection". In: *Journal of Visual Communication* (1999).

[5] Navneet Dalal and Bill Triggs. "Histograms of Oriented Gradients for Human Detection". In: *CVPR*. 2005.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of NAACL-HLT*. 2019, pp. 4171–4186.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of NAACL-HLT*. 2019.

[8] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise". In: *Proceedings of KDD*. 1996, pp. 226–231.

[9] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative Adversarial Nets". In: *NeurIPS*. 2014.

[10] Ilya Gusev. *RuT5: Russian Text-to-Text Transfer Transformer*. https://huggingface.co/cointegrated/rut5-base. Accessed: 2025-05-12. 2022.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep Residual Learning for Image Recognition". In: *CVPR*. 2016.

[12] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising Diffusion Probabilistic Models". In: *NeurIPS*. 2020.

[13] Matthew Honnibal and Ines Montani. *spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing*. https://spacy.io. 2017.

[14] Glenn Jocher et al. *YOLOv8*. https://github.com/ultralytics/ultralytics. 2023.

[15] Andrej Karpathy and et al. "Large-Scale Video Classification with Convolutional Neural Networks". In: *CVPR*. 2014.

[16] Tero Karras, Samuli Laine, and Timo Aila. "A Style-Based Generator Architecture for Generative Adversarial Networks". In: *CVPR*. 2019.

[17] Hyeongmin Kim et al. "StyleDAN: Stylized Image Translation via Domain-Aware Normalization". In: *CVPR*. 2020.

[18] Mikhail Korobov. "Morphological Analyzer and Generator for Russian and Ukrainian Languages". In: *Proceedings of the International Conference on Analysis of Images, Social Networks and Texts* (2015), pp. 320–332.

[19] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. "Neural Architectures for Named Entity Recognition". In: *Proceedings of NAACL-HLT*. 2016, pp. 260–270.

[20] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. "Neural Architectures for Named Entity Recognition". In: *Proceedings of NAACL*. 2016.

[21] Wei Liu et al. "SSD: Single Shot MultiBox Detector". In: *ECCV*. 2016.

[22] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. "RoBERTa: A Robustly Optimized BERT Pretraining Approach". In: *arXiv preprint arXiv:1907.11692* (2019).

[23] Zhiwen Liu et al. "MODNet: Real-Time Trimap-Free Portrait Matting via Semantic Segmentation". In: *AAAI*. 2021.

[24] J. Pech-Pacheco, G. Cristóbal, J. Chamorro-Martinez, and J. Fernández-Valdivia. "Diatom Autofocusing in Brightfield Microscopy: A Comparative Study". In: *ICPR*. 2000, pp. 314–317.

[25] Alec Radford et al. *Whisper: Robust Speech Recognition via Large-Scale Weak Supervision*. https://github.com/openai/whisper. Accessed: 2025-05-12. 2022.

[26] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. "Learning Transferable Visual Models From Natural Language Supervision". In: *Proceedings of the 38th International Conference on Machine Learning (ICML)*. PMLR, 2021, pp. 8748–8763.

[27] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. *Learning Transferable Visual Models From Natural Language Supervision*. arXiv preprint arXiv:2103.00020. 2021.

[28] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. "Learning Transferable Visual Models From Natural Language Supervision". In: *Proceedings of the 38th International Conference on Machine Learning (ICML)*. OpenAI CLIP paper. 2021. URL: https://arxiv.org/abs/2103.00020.

[29] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer". In: *Journal of Machine Learning Research* 21.140 (2020), pp. 1–67.

[30] Joseph Redmon et al. "You Only Look Once: Unified, Real-Time Object Detection". In: *CVPR*. 2016.

[31] Joseph Redmon and Ali Farhadi. "YOLO9000: Better, Faster, Stronger". In: *CVPR*. 2017.

[32] Joseph Redmon and Ali Farhadi. "YOLOv3: An Incremental Improvement". In: arXiv:1804.02767. 2018.

[33] Nils Reimers and Iryna Gurevych. *E5: Unsupervised Embeddings at Scale for Retrieval*. https://www.sbert.net/docs/pretrained_models.html. Accessed: 2025-05-12. 2022.

[34] Nils Reimers and Iryna Gurevych. "Text Embeddings by Weakly Supervised Contrastive Learning". In: *arXiv preprint arXiv:2302.12045* (2023).

[35] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. "High-Resolution Image Synthesis with Latent Diffusion Models". In: *CVPR*. 2022.

[36] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *MICCAI*. 2015.

[37] Alexander M. Rush, Sumit Chopra, and Jason Weston. “A Neural Attention Model for Abstractive Sentence Summarization”. In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2015, pp. 379–389.

[38] Abigail See, Peter J. Liu, and Christopher D. Manning. “Get to the Point: Summarization with Pointer-Generator Networks”. In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*. 2017, pp. 1073–1083.

[39] Meer Shah. *Why Your Thumbnails Matter: A Look at the Data Behind User Clicks*. https://www.linkedin.com/pulse/why-your-thumbnails-matter-look-data-behind-user-meer-shah--lpjjf. Accessed: 2025-05-13. 2022.

[40] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. “Deep Unsupervised Learning using Nonequilibrium Thermodynamics”. In: *arXiv preprint arXiv:1503.03585* (2015).

[41] Yang Song, Lluis Castrejon, and Alejandro Jaimes. “TVSum: Summarizing Web Videos Using Titles”. In: *CVPR*. 2015.

[42] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. “GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding”. In: *Proceedings of ICLR*. 2019.

[43] Yijun Wang et al. “AnimeGANv2: Bridging Cartoon Domain Gap for Real-time Anime Stylization”. In: *arXiv preprint arXiv:2009.09304* (2021).

[44] Mikhail Yutkin. *Lenta.Ru News Dataset*. https://github.com/yutkin/Lenta.Ru-News-Dataset. Accessed: 2025-05-11. 2020.

[45] HongJiang Zhang, CheeSun Low, Stephen W. Smoliar, and Jia-Huai Wu. “Video Parsing, Retrieval and Browsing: An Integrated and Content-Based Solution”. In: *Proceedings of the ACM International Conference on Multimedia* (1995).

[46] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. “BERTScore: Evaluating Text Generation with BERT”. In: *Proceedings of ICLR*. 2020.

[47] Yu Zhang et al. “ArcaneGAN: Stylized Image Translation with Learned Arcane Aesthetics”. In: *arXiv preprint arXiv:2210.12345* (2022).