UDC 004.6:33
DOI: 10.20535/SRIT.2308-8893.2025.2.04

A CONCEPTUAL MODEL AND A SYSTEM FOR REPLACING TEXT IN AN IMAGE WHILE PRESERVING THE STYLE

P. MASLIANKO, M. ROMANOV

Abstract. Text replacement in images, particularly while preserving its style, is a complex task that requires solving a range of scientific challenges and developing new technical solutions. One of the main issues is maintaining the authenticity and harmony of the image after modifications. The Research Objective is the development of a conceptual model and a system for text replacement in images with style preservation based on systems engineering methodology and the Eriksson-Penker business profile, ensuring the natural integration of new text elements into the image's context. Implementation Methodology – the systems engineering methodology and the Eriksson-Penker business profile are used to formalize the structured process of developing a system for text replacement in images with style preservation. Research Results – a method for developing the system based on systems engineering techniques was proposed, consisting of four main stages. In the first stage, the system structure is modeled as an Eriksson-Penker business profile. In the second stage, a set of processes is defined that are characteristic of the Data Science system class and the CRISP-DM international standard. Also, the structural and dynamic representations of the conceptual model, as well as the component interaction interfaces, are modeled. The third stage involves implementing a specific version of the system, while the fourth stage focuses on system verification and validation. A systems engineering method for the conceptual model and system for text replacement in images with style preservation has been proposed. It is based on a modified Eriksson-Penker business profile for metalevel system representation and international standards for Data Science and Data Mining processes.

Keywords: systems engineering method, Eriksson-Penker business profile, conceptual model, system for text replacement in images with style preservation.

BACKGROUND

Replacing text in an image, in particular, while preserving its style, is a complex task that requires solving a number of scientific problems and new technical solutions. One of the main challenges is to preserve the authenticity and harmony of the image after the changes have been made, as any modification of the text can disrupt the original style, fonts, colours and visual composition. To make such changes look natural, a mechanism is needed that can effectively identify the text style, recreate it, and adapt it to the new content.

Key aspects of the task:

1. Preserving the style: when replacing text, it is important not only to correctly reproduce the font, size, colour and other visual characteristics, but also to integrate it into the overall visual aesthetics of the image (including background, shadows, textures).

2. Adaptability to different styles: texts on images can be executed in a wide variety of styles — from handwritten text to complex graphic elements. The sys-

tem should be able to work with different text styles and formats, automatically detecting them.

3. Contextual replacement: Often, text is not just a separate element, but is integrated into a complex visual scene. Therefore, the replacement should take into account not only the style but also the context of the image to make the new text look appropriate.

4. Automation and scalability: manual text replacement is a time-consuming process in many cases. Automation of this task will significantly save time with a large volume of images, which is important for industries such as marketing, design or content localisation (Table 1).

Table 1. Overview and comparative analysis of existing solutions

Approach	Advantages	Disadvantages
Manual tools	High precision at the right skill level, full user control	Labour intensive, difficult to scale
Classic computer vision algorithms	Speed, automation of key processes	Limited ability to save complex text styles, need for additional customisation
Generative models (GAN)	Ability to automatically save complex stylistic elements of text	High cost of computing, dependence on data quality, long learning curve

Current methods for style-preserving text replacement in images are mostly based on generative adversarial networks and deep learning. StyleGAN remains one of the most effective solutions, thanks to its ability to accurately reproduce stylistic elements of text and integrate them into the overall context of the image.

Motivation for the development of the conceptual model and system to replace text in an image while preserving the style.

The main motivation is the need for a system that will automatically replace text in an image while maintaining its style. This is necessary for such cases of activity:

1. Content localisation: Many companies and brands need to adapt their visuals for different markets and languages. For example, advertising banners or posters may have text that needs to be translated and replaced in other languages without losing style and graphic authenticity.

2. Graphic design: Quickly changing text on design mockups can be important when working on prototypes or changes during the approval phase of projects.

Thus, the relevance of the problem lies in the need to develop a scientifically grounded conceptual model of this class of systems and implement an automated tool for replacing text in images while preserving the style. The solution to this problem opens up opportunities to speed up the content creation process, improve its quality and accuracy of visual adaptation for various purposes.

PROBLEM STATEMENT

Object of research. Conceptual ideas and approaches, existing mathematical models and algorithms for detecting, analysing and synthesising text in images, their theoretical aspects, architecture and existing software tools for image and text processing.

Subject of research. Conceptual model and system of text replacement in an image with style preservation. Structural and dynamic representation of the system, image generation algorithms. Models and algorithms for style control in the process of image generation.

Research objective. To develop a conceptual model and a system for replacing text in images while preserving its style, which allows to accurately identify, analyse and reproduce the visual characteristics of the text (font, size, colour, orientation, textures, shadows and other stylistic elements), ensuring the natural integration of new text elements into the context of the image.

Final result. A conceptual model and a system for replacing text in an image while preserving its style, as well as verification and validation results.

A method of system engineering of a conceptual model and a system for replacing text on images while preserving its style

The main idea behind the development of the conceptual model is based on [1–4] and consists in applying the systems engineering methodology and the Eriksson–Penker business profile to formalise an orderly way of developing a text-to-picture image replacement system with style preservation (Fig. 1).

One of the most common models of activity representation is the Eriksson–Penker business profile [5–8], in the context of which the authors formulated four main essences of the formal representation of the activity of any business system (Fig. 1):

– goals (represent the purpose of the system and are formulated as a rule. Goals can be broken down into sub-goals and achieved through the implementation of processes);

– processes (the main actions that make up the system's activities and are intended to achieve the goal in accordance with the established business rules. Processes are usually subject to rules, can change the state of input resources, and produce new resources — the system's output resources — in accordance with the conditions and requirements set by stakeholders);

– resources (physical, abstract or informational objects that the system consumes, uses, processes and produces throughout its activities to achieve the goal);

– rules (certain formalised restrictions, frameworks, conditions and requirements, etc. that are imposed on processes and describe the nature of the relationships between resources).

Such an ordered set of formalised (in particular, in UML notation) entities and system representations based on the Eriksson–Penker business profile formalises and systemises the conceptual model (meta-model) of a text-to-picture image replacement system with style preservation (Fig. 1).

Thus, the essence of the system engineering method of the conceptual model of the image text replacement system with style preservation is to apply the system approach and the Eriksson–Penker business profile to formalise and produce such systems

At the second stage, we formalise the structural and dynamic representation of the conceptual model of the system, interfaces for component interaction, technical requirements and specifications of all stakeholders.

At the third stage, we implement a specific version of the system based on technological and mathematical tools and in accordance with the technical requirements and specifications of all stakeholders.

The fourth stage involves verification and validation of a specific version of the system.

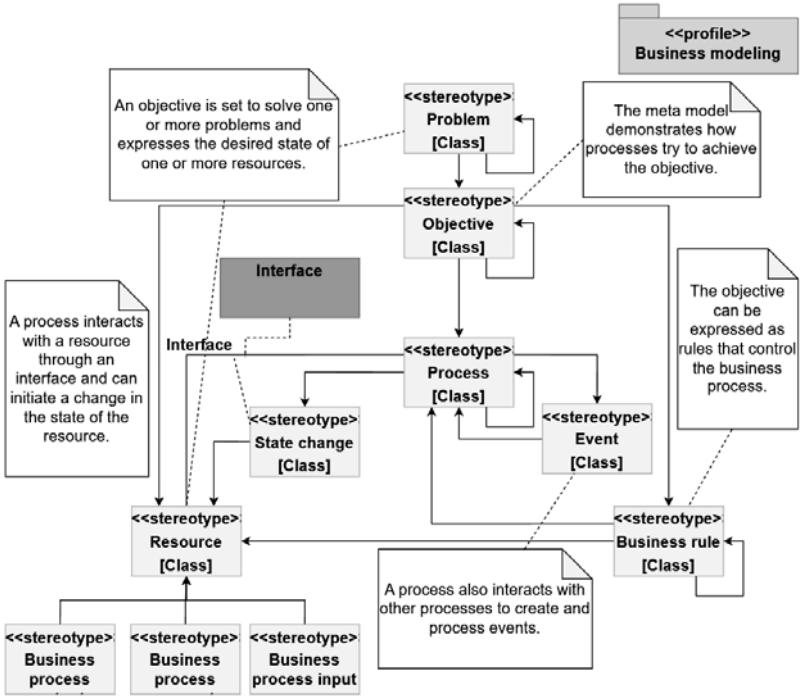

Fig. 1. Improved Eriksson–Penker business profile. Class diagram in UML notation [1]

Formalisation of Eriksson–Penker business profile classes for a conceptual model and a system for replacing text on images while preserving its style

Let's define the content of each of the classes of the diagram (Fig. 1) in terms of the system engineering problem, namely classes:

1. Problem (an actual issue that requires appropriate solutions, the main motivation for developing a system that leads to the formulation of a specific goal. The problem of this work: the need to develop a conceptual model and an automated tool for replacing text in images while preserving the style. The solution to this problem opens up opportunities for significantly speeding up the content creation process, improving its quality and accuracy of visual adaptation for various purposes).

2. Purpose (expresses the global goal of the work designed to solve the problem. The purpose of this work: To develop a system for automatically replacing text in images while preserving its style, which allows for accurate identification, analysis and reproduction of visual text characteristics (font, size, colour, orientation, textures, shadows and other stylistic elements), ensuring natural integration of new text elements into the image context).

3. Process (a set of processes of the system's activity, which results in achieving the goal, a clearly defined sequence of actions/subprocesses that leads to the fulfilment of a certain task. The processes of this system are: Loading training data, Pre-processing training data, Training the model, Replacing text in an image with preservation of style, Functioning of the web application).

4. State change (possible changes in certain resources as a result of the processes. The system has three state changes: Image with text and word coordinate file → Segmented images (the Pre-process training data process), Segmented images with text labels → Transformed images and text labels (the Replace text in image with preservation of style process), Initialised model → Trained model (the Train model process)).

5. Resource (any entities (tangible or intangible) that are consumed and produced by the system under development.

The resources of the lowest level of the hierarchy, which are directly involved in the processes, are also divided into the following three classes by the nature of their influence on the processes:

– Business process output (resources produced by the system, the end result of its operation. This includes the generated image with new text).

– Business process support resource (resources that support the execution of processes, but are not the final result of work: Trained model, Computer hardware and computing resources, Pre-processing algorithms, Metadata files, Software).

– Business process input (primary input resources of the initial processes that initialise the system cycle: Image with text, Metadata file for segmentation, Image for text replacement, New text).

6. Event (occurs due to certain external factors or as a result of interaction between processes. The potential events of this system are Uploading an image with text by a user, Entering a new text to replace it, Loading training data, Completing preprocessing, Completing model training, Transforming input parameters, Outputting results via the interface).

7. Business Rule (BR): formal instructions that regulate, limit, establish the context and framework for the functioning of processes. Example of a business rule: the format of images must be JPG, JPEG, PNG).

Thus, on the basis of the conceptual model (meta-model) of the image text replacement system with style preservation shown in Fig. 1, we can reasonably formalise the structural representation of the image text replacement system with style preservation in the form of a component diagram (Fig. 2).

Formalisation of the structural representation of the conceptual model of the system

Such a conceptual model of structural representation (Fig. 2) formalises the class of systems for replacing text in an image with preserving the style in the form of an ordered set of entities and relations between them. An important property of such a conceptual model is the ability to implement the internal structure of the system components on the basis of various engineering and mathematical tools necessary for the implementation of a particular system without changing the structure and interfaces of interaction between the components.

Here is a short list of the system's interfaces and their functionality:

- IDD (Interface Dataset Download) — interface for downloading an external data set;
- IIP (Interface Image Preprocessing) — interface for transferring the initially processed downloaded data set;
- ISIP (Interface Segmented Image Processing) — interface for transferring processed segmented images with annotations;
- IMT (Interface Model Training) — interface for transferring the trained model;
- IIITP (Interface Input Image and Text Processing) — an interface for transmitting the result of processing images and text entered by the user.

Based on the model of structural representation, we formalise the model of dynamic representation of the system, which shows the internal structure of components and the algorithm of interaction between them (Figs. 3, 4).

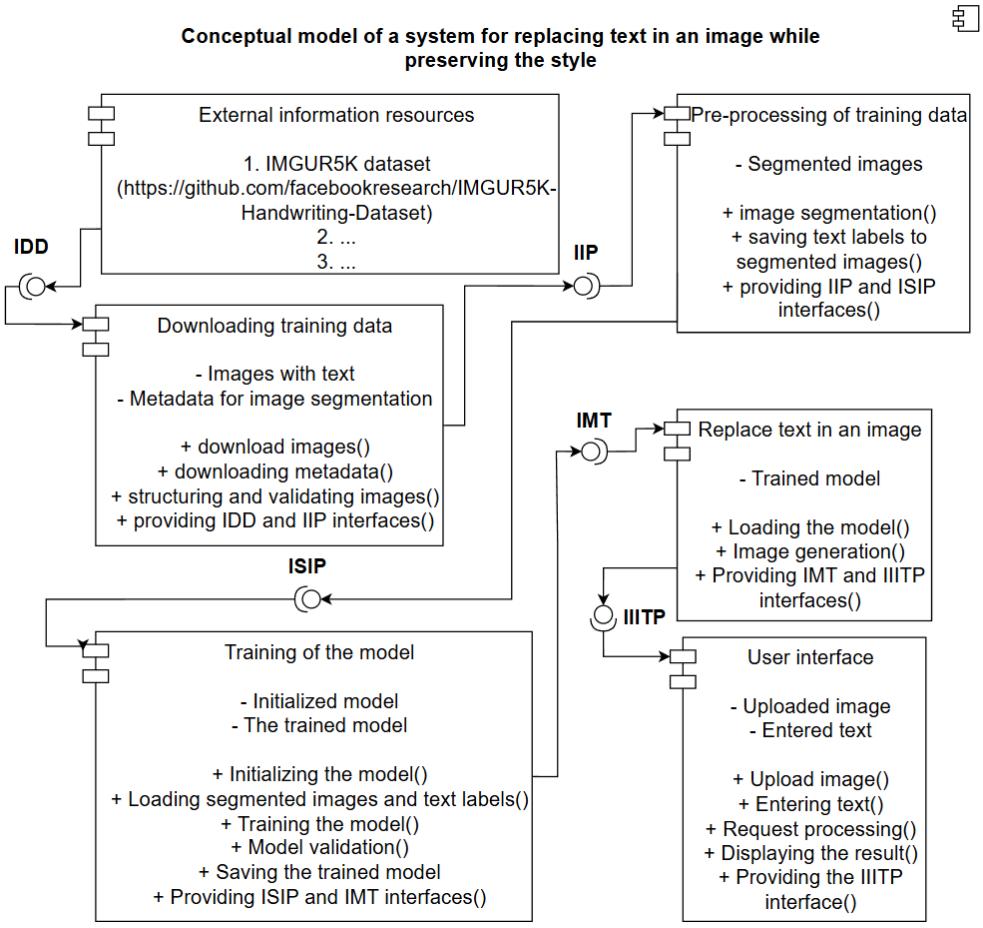

Fig. 2. Conceptual model of a system for replacing text in an image while preserving the style. Component diagram in UML notation

Formalisation of the dynamic representation of the conceptual model of the system

The formalisation of the dynamic representation is determined on the basis of a set of processes specific to the class of Data Science systems, according to the Data Science process defined by O'Neill and Schutt [2] and the international standard CRISP-DM as interpreted by Foster and Fawcett [3] (Figs. 3, 4).

At this level, the organisation of activities can be refined to take into account the specifics of the system and, as a result, decomposed into the following three sub-stages:

1. Collection, analysis, and processing of training text data to be used in model training according to the Data Science process model proposed by O'Neill and Schutt [2] or the Data under standing and Data processing stages of the CRISP-DM (CRoss Industry Standard Process for Data Mining) analysed by Foster and Fawcett [3].

2. The actual construction (architecture development) and training of the model is analogous to the Machine Learning Algorithms Statistical Models stage [2] or the Modeling stage of the standard Data Mining process [3].

3. Determining metrics for evaluating the effectiveness of both trained models and the system as a whole is analogous to the Report Findings stage [2] or the Evaluation stage of the standard Data Mining process [3].

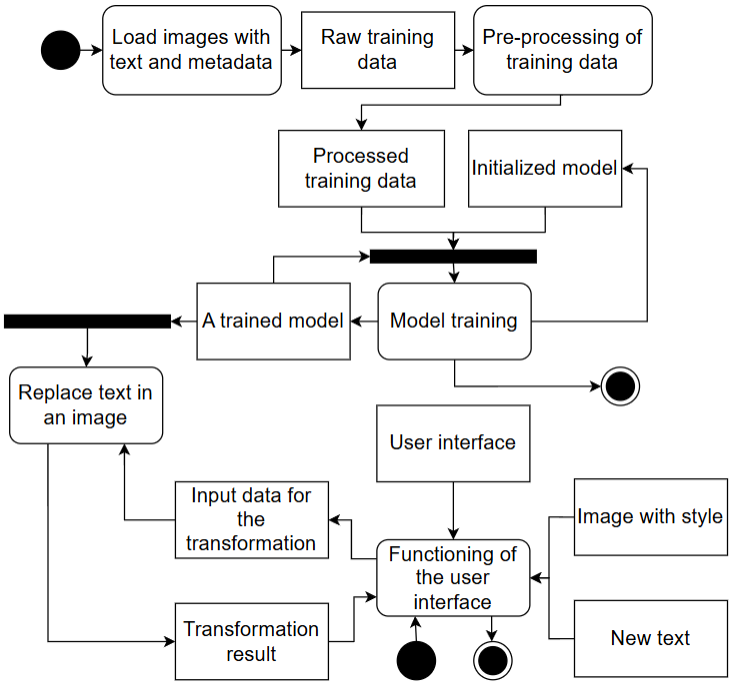

Fig. 3. Conceptual model of a system for replacing text in an image while preserving the style. The first level activity diagram in UML notation

Thus, the Eriksson–Penker business profile is a system of classes and relationships between them that are necessary and sufficient to represent and develop a conceptual model of the system. And the corresponding set of Data Science technologies are tools for implementing the components of the conceptual model of the system.

An exhaustive list of business rules (specifications) for a business profile regulates the functionality of the user interface of a particular system, for example:

BR1 — The user must enter new text and select a replacement image before starting the process. If this data is not provided, the system does not start processing and sends a request to fill in all fields.

BR2 — If the user uploads an image for the style, it must comply with the established formats (JPEG, JPG, PNG) and be no larger than 10 MB.

BR3 — After the text replacement is complete, the user can preview the result before uploading it.

BR4 — If an error occurs (incorrect image loading, segmentation errors, insufficient data for training, etc.), the system should stop the process and send a clear error message to the user indicating possible ways to fix it.

Implementation of the conceptual model for a specific version of the system for replacing text in an image while preserving the style. Implementation of the structural representation of the system

Since the conceptual model of the system is a system of classes and relations between them, necessary and sufficient for the functioning of the system (Fig. 1), and the conceptual model of the system in the form of a diagram of components has the form shown in Fig. 2, then the purpose and functionality of the system components will be as follows:

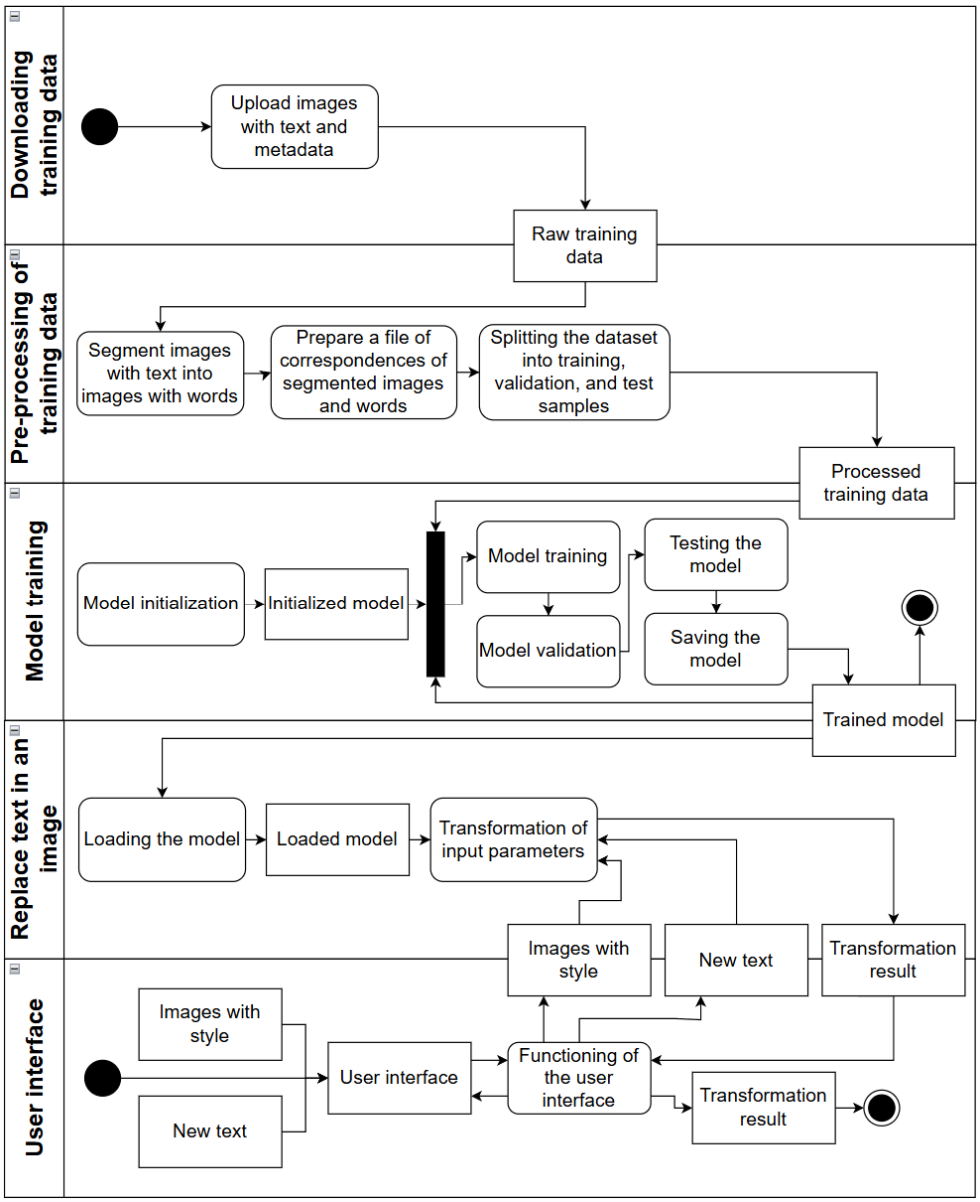

Fig. 4. Conceptual model of a system for replacing text in an image while preserving the style. A detailed process diagram based on a second-level activity diagram in UML notation

1. Training data loading. This module is responsible for loading images from the IMGUR5K dataset [9], checking their integrity using hashes, and loading annotations for use in further processing steps.

2. Training data preprocessing. This module is responsible for segmenting the uploaded images and saving them to the appropriate directories for training, validation and testing. It is also possible to reduce the size of the training dataset by random sampling, which is useful in situations of lack of computing power.

3. Model training. The main module of the system, which results in a trained model for transforming the entered images and text. For training, pre-segmented images are used with corresponding annotations that describe the text on the segmented image.

4. Replacing text with images. A module that serves as a facade for interacting with the trained model, where the trained model is loaded into memory, input parameters are passed to the model, and the results are returned in the form of generated images.

5. User interface. A module that is an interface (web page) where a user can upload their image with a style and enter text to generate an image with text in the style of the uploaded image.

Implementation of a dynamic system representation. The dynamic representation of the system in the form of an activity diagram (Fig. 3) is a visualisation of the main cycles of the processes and their first-level interaction.

Fig. 4 shows a diagram of the processes of the conceptual model of the system in the form of a diagram of interaction of components (see Fig. 2) and shows their internal structure. Such a representation makes it possible to group the main processes of the system by the relevant components. The process diagram is an integral part of the conceptual model based on the Eriksson–Penker business profile, as it combines both types of representations of the system, structural and dynamic, and formalises its activities.

Implementation of the Training Data Loading and Training Data Preprocessing components

Data download module. This module is responsible for loading images from the IMGUR5K dataset [9], checking their integrity using hashes, and loading annotations for use in further processing steps.

Uploading images: For each image, the module generates a URL, uses the requests library to download it, and saves the file in a specific directory. After the image is uploaded, its hash is checked to ensure authenticity.

Parallel uploading: To optimise the uploading process, the module uses the ThreadPoolExecutor from the futures library, which allows you to upload several images simultaneously, which significantly speeds up processing.

Loading annotations: After the images are successfully uploaded, the module loads annotations for each image in JSON format, containing information about the image, its hash, and the coordinates of text elements. The annotations are also divided into three sets: training, validation, and test.

Training data processing module. This module is responsible for processing the uploaded images, extracting text features, and preparing the dataset for model training.

Processing of text elements: For each image, the module uses annotations to find text features in the images. The crop_minAreaRect() function is used to crop the image around the text, taking into account its coordinates and orientation. This function is based on the perspective transformation algorithm cv2.getPerspectiveTransform() from the OpenCV library, which allows you to accurately select text taking into account its slope.

Saving processed images: The cropped text elements are saved in PNG format, and a corresponding record is created in the JSON file with the text on the image for each of them.

Reducing the data sample: If you specify the reduce option, the module randomly selects a part of the images for processing, which allows you to work with a smaller sample for testing or speeding up the work.

Implementation of the Model Training component. Model training process:

1. Load the training dataset (segmented images with annotations).
2. Creating an initialised model of the initial architecture using available python libraries and frameworks / loading a pre-trained model — the latest version of the saved model.
3. Training the model using the GAN algorithm on the training data set by optimising the loss function.
4. Conducting validation to determine the optimal model architecture and model hyperparameters on the validation dataset with the calculation of MSE, SSIM, PSNR and FID metrics.
5. Final testing on the test data set.
6. Saving the current version of the model, selecting the most efficient ve
7. rsion for processing user requests.

Mathematical support of the model training component

In Fig. 5 we can see a high-level diagram of the model training process. This diagram is based on the TextStyleBrush architecture [9]. It shows the main components of the model, and we will describe them:

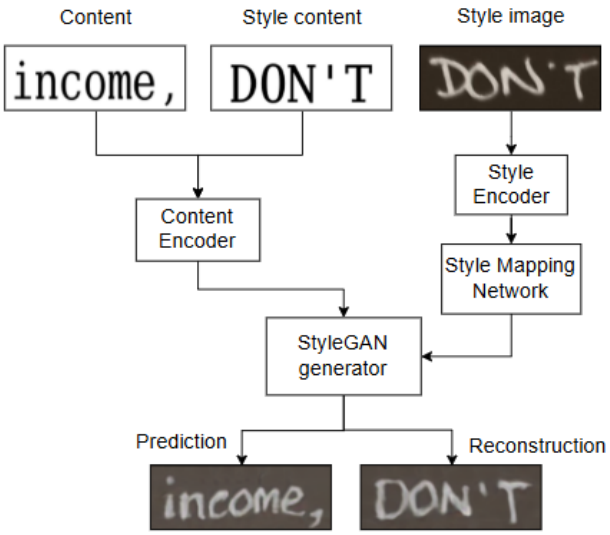

Fig. 5. High-level diagram of the model training process

1. Content — an image of the text we want to see on the generated image. This image is formed from the annotations to the images, namely, the text is taken and transformed into an image using a file that describes the font (VerilySerifMono) and functions from the PIL library. The size of all images is fixed — 192 by 64 pixels. This is done in order to have the same data types in all modules of our model, which greatly simplifies implementation.
2. Style content — an image of the text that appears on the image with the desired style.
3. Style image — an image that contains the desired style for the generated image.
4. Content Encoder — it is a pre-trained ResNet18 model that does not include average pooling layers and all subsequent ones, which is done to preserve the spatial properties of the image.
5. Style Encoder — it is a pre-trained ResNet18 model that lacks fully-connected layers.

6. Style mapping network — the key component of the StyleGAN model, which transforms the style vector obtained by the Style Encoder into a vector of parameters used to control the style on each layer of the generator using fully connected layers. The network structure consists of the following layers: PixelNorm (for normalizing the values of the vector obtained from the StyleEncoder component), Linear (fully connected layer), and LeakyReLU (nonlinear activation function).

7. StyleGAN generator — is the main component of the model that is responsible for generating images, taking as input an image with text and a vector of parameters obtained using the Style mapping network.

The synthesizing network consists of several convolutional units that gradually increase the image resolution starting from a small initial feature map. Each block uses stylized layers, such as AdaIN [10], to control the style of the image.

AdaIN (Adaptive Instance Normalization) — is a key component of StyleGAN that is responsible for styling images based on the style derived from the mapping network. AdaIN normalizes the feature activations for each feature map separately, and then scales and shifts their values according to the parameters obtained from the style vector.

Formally, the work of this layer can be represented as follows:

Let's say we have: x — input feature tensor for the convolutional layer; w — style vector obtained from the mapping network; $\mu(x)$ and $\sigma(x)$ — mean and standard deviation for any channel.

First, we calculate the affine transformation:

$$y_\sigma = \mathrm{A}_\sigma w + b_\sigma, \quad y_\mu = \mathrm{A}_\mu \cdot + b_\mu$$

where A_σ and A_μ — weight matrices for multiplying the vector w, which provide different scaling depending on the style; b_σ and b_μ — displacement vectors that shift the parameters that control the mean value (for y_μ) and standard deviation (for y_σ) of features in the image space.

After the affine transformation, the style vector w creates the parameters y_σ and y_μ, which are then passed to AdaIN. Each feature channel in the tensor is normalized and modified according to these parameters, allowing each channel to respond to the style w.

Next, the features are normalized, where the mean $\mu(x)$ and standard deviation $\sigma(x)$ are calculated for each channel, and then the value is normalized:

$$x'_{ij} = \frac{x_{ij} - \mu(x)}{\sigma(x)}.$$

After normalization, each channel is scaled by y_σ and shifted by y_μ, which allows you to change the style:

$$x_{AdaIN} = y_\sigma x' + y_\mu$$

Thus, the affine transformation helps to transfer style parameters from the style vector w to the feature level, where they determine the color, texture, and other characteristics of the generated image. This allows for flexible control of visual attributes through the latent vector w.

8. Prediction — an image generated by the generator that combines a style from the Style image and text from the Content image.

9. Reconstruction — is an image generated by a generator that combines the style from the Style image and the text from the Style content image; under ideal conditions, the Reconstruction image should be identical to the Style image.

Next, we will describe additional components of the system that help calculate the loss function for our model.

Let's start with the discriminator (Fig. 6), as we can see, using the Discriminator component we calculate two loss functions — Discriminator adversarial loss and Generator adversarial loss. To calculate these two functions, we use the classical approach for GAN models, namely:

$$L_{discriminator} = \frac{1}{2}(MSE(D(z_{style\ image}),1) + MSE(D(G(z_{style\ content}, z_{style\ image})),0)),$$

where MSE — mean squared error; D — discriminator; G — generator; $z_{style\ image}$ and $z_{style\ content}$ — the results of the Style Encoder and Content Encoder components, respectively.

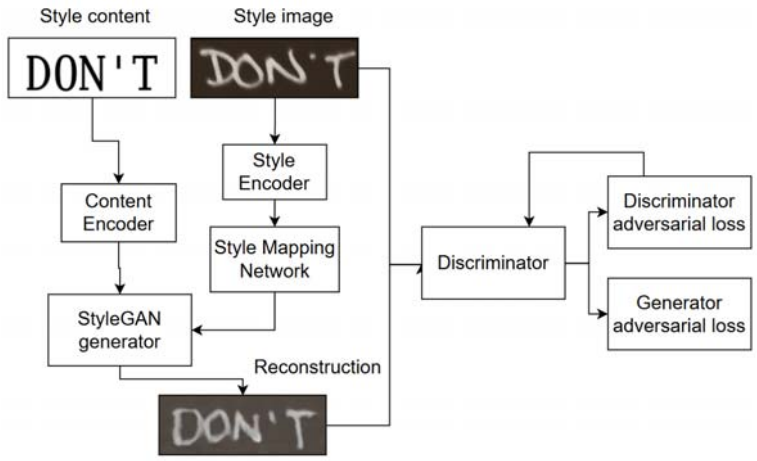

Fig. 6. Diagram of the process of calculating the competitive loss function for the discriminator and generator

The generator loss function is calculated as follows:

$$L_{generator} = MSE(D(G(z_{style\ content}, z_{style\ image})),1)).$$

To calculate the OCR loss function (Fig. 7), we use the trained TRBA (Text Recognition by Attention) model, which is denoted as OCR Recognizer in the diagram, Content label and Style content label are the text in the image Content and Style content, respectively, which are available during training.

In order to calculate the OCR loss function, we follow the following steps:

- Normalize the image size.
- Encode the text labels Content label and Style content label.
- Pass the Prediction and Reconstruction images to the model and get the predicted text on these images in the encoded form.
- Calculate the cross-entropy for the pairs (Prediction, Content label) and (Reconstruction, Style content label).
- Sum the obtained values and divide them by two.

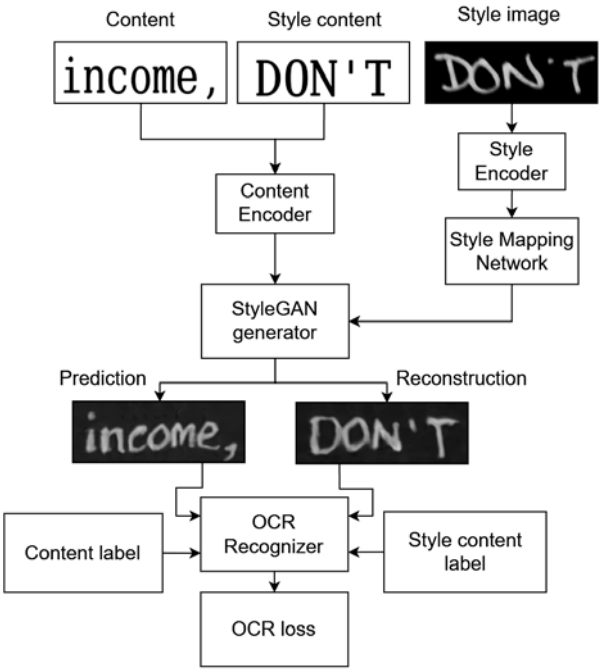

Fig. 7. Diagram of the OCR loss function calculation process

Formally, the process of calculating the OCR loss function can be represented as follows:

$$L_{OCR} = \frac{1}{2}(CE(TRBA(Prediction), Content\,label) +$$
$$+ CE()TRBA(Reconstruction), Style\,content\,label),$$

where CE — cross entropy; $TRBA$ — pretrained model TRBA.

To calculate the Reconstruction of the loss function (Fig. 8), we need to calculate the difference between the Style image and the Reconstruction image. Formally, this is written as follows:

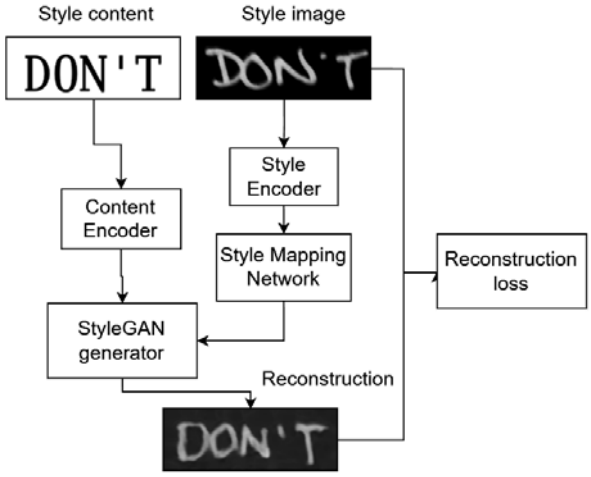

Fig. 8. Diagram of the process of calculating the Reconstruction loss function

$$L_{reconstruction} = L1(Style\,image, Reconstruction),$$

where $L1$ — is the $L1$ loss function or mean absolute error.

To calculate the Cycle of the loss function (Fig. 9), we first need to get a Reconstruction image, and then pass it to the model input and get another image. The idea is that we should lose as little data as possible during the transformations driven by the model. Once we have two images, we calculate the average absolute error between them:

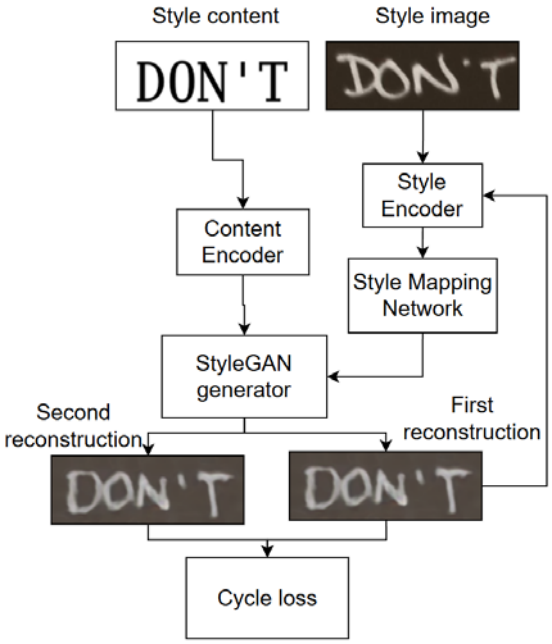

Fig. 9. Diagram of the process of calculating the Cycle of the loss function

To calculate the Typeface loss function (Fig. 10), we use the pre-trained VGG16 model, in which the last classification layer is removed, which means that the model will not perform classification, but will simply return a feature vector. After that, we pass two images to the model input — Style image and Reconstruction and calculate the L1 loss function between them, formally written as follows:

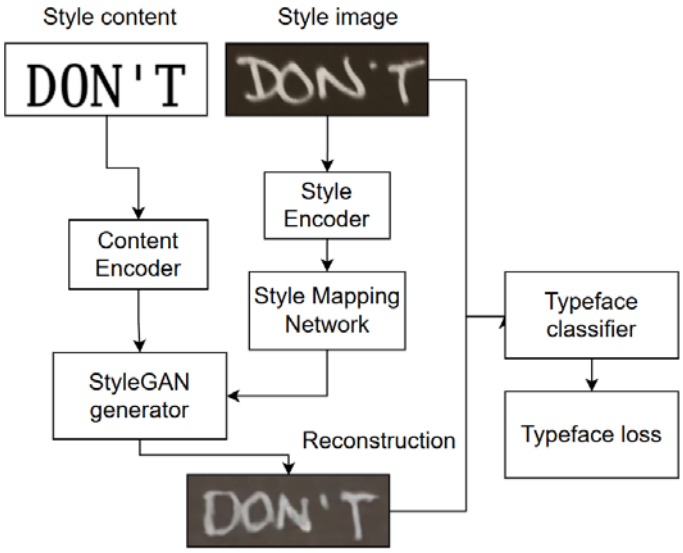

Fig. 10. Diagram of the Typeface loss function calculation process

$$L_{typeface} = L1(VGG16(Style\,image), VGG16(Reconstruction)).$$

To calculate the Perceptual and Texture loss functions (Fig. 11), we use the trained VGG16 model. For each block of VGG16, the difference between the activations of the corresponding layers of the input and target images is calculated, and the difference is measured using $L1$ loss, thus we obtain the Perceptual loss function:

$$L_{perceptual} = \frac{L1(X,Y)}{h\,w},$$

where X and Y — are activations at the *i*-th layer for the input and target images, respectively; h and w — height and width of the corresponding activations.

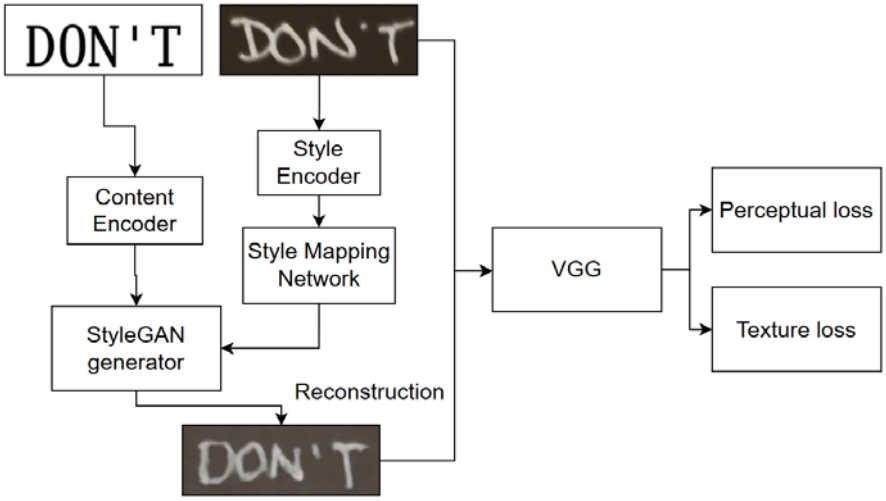

Fig. 11. Diagram of the process of calculating Perceptual and Texture loss functions

Texture losses are calculated in a similar way, but additionally, Gram matrices are calculated (The element G_{ij} represents the mutual correlation between the *i*-th and j-th channels. It calculates how similar these two channels are on average, i.e. how much the pixel value variations of one channel are related to the variations of the other. The gram matrix provides an idea of how different parts of the filters "cooperate" with each other to reflect the style or texture of the image):

$$L_{texture} = \frac{L1(G(X), G(Y))}{h\;w},$$

where $G(X)$ and $G(Y)$ — are Gram matrices, which are defined as $G(X) = X\,X^{\mathrm{T}}$.

The final loss function of the generator is as follows:

$$L_{final} = 0.06L_{generator} + 0.07L_{OCR} + 2.0L_{reconstruction} + 2.0L_{cycle} +$$

$$+ 0.0L_{typeface} + 25.0L_{perceptual} + 7.0L_{texture}.$$

The coefficients were selected empirically.

Verification and validation of the system engineering method of the system

The fourth stage involves verification and validation of the developed system to ensure that all technical specifications and stakeholder requirements are met. First of all, the model used to implement the Model Training component should be taken into account. It can be one of the following types of neural network architectures (this list is not exhaustive):

- convolutional neural networks;
- recurrent neural networks;
- generative-adversarial neural networks.

Next, you can analyze the functionality, adequacy, and performance of the developed system based on the selected benchmarking metric.

So, based on the definition of the concept of "method" [4], we can formulate the definition of the System Engineering Method of the conceptual model and the system of text replacement in images with style preservation: it is an ordered set of classes of tasks, processes, resources, business rules and relationships between them to produce a system based on the system engineering methodology, the Eriksson–Penker business profile and Data Science technologies.

Defining criteria for evaluating system performance

When designing a specific text-to-picture image replacement system, it is important to use appropriate metrics to evaluate the quality of image generation. The goal of such metrics is to measure how well the new images match the original in terms of visual similarity, stylistic authenticity, and detail accuracy. The metrics used in this paper are: mean square error (MSE), structural similarity (SSIM), peak signal-to-noise ratio (PSNR), and Frechet distance between distributions (FID). Each of them assesses different aspects of system quality and is effective for evaluating the performance of generative models.

1. Mean squared error (MSE)

Mean Squared Error (MSE) — is a metric that evaluates the difference between the pixel values in the original image and the generated image. It is calculated as the arithmetic mean of the square deviations between the corresponding pixels of the two images. The formula for calculating MSE looks like this:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(I_{\text{orig}}(i) - I_{\text{gen}}(i))^2,$$

where N — number of pixels in the image; $I_{\text{orig}}(i)$ and $I_{\text{gen}}(i)$ — pixel values of the original and generated images, respectively.

Advantages of MSE:

Easy to implement and interpret;

Well suited for evaluating pixel-by-pixel rendering accuracy.

Why it's useful: MSE allows you to measure the exact difference between the original and the generated image, which is important for assessing the quality of preservation of text elements and image details after text replacement.

2. Structural similarity (SSIM)

Structural Similarity Index Measure (SSIM) — is a metric that evaluates the similarity between two images in terms of their structure, brightness, and contrast.

SSIM tries to model human perception, which makes it more sensitive to changes in image structure [11]. SSIM calculation formula:

$$SSIM(I_{\text{orig}}, I_{\text{gen}}) = \frac{(2\mu_{\text{orig}}\mu_{\text{gen}} + C_1)(2\sigma_{\text{orig-gen}} + C_2)}{(\mu_{\text{orig}}^2 + \mu_{\text{gen}}^2 + C_1)(\sigma_{\text{orig}}^2 + \sigma_{\text{gen}}^2 + C_2)},$$

where μ_{orig} and μ_{gen} — average pixel values of the original and generated images; σ_{orig} and σ_{gen} — covariance between the original and generated image; C_1 and C_2 — small constants to stabilize calculations.

Advantages of:

Good at showing structural changes in an image that are difficult to detect with MSE.

Simulates visual perception, making it more suitable for assessing image quality from the perspective of the human eye.

Why it's useful: SSIM helps to assess how well the basic structures and textures of the original image are preserved after text replacement, which is important for maintaining the style and authenticity of visual content.

3. Peak Signal-to-Noise Ratio (PSNR)

Peak Signal-to-Noise Ratio (PSNR) — is a metric used to evaluate the differences between an original and a generated image by measuring the level of noise or distortion. PSNR is calculated based on MSE and is expressed in decibels (dB) [12]:

$$\text{PSNR} = 10 \cdot \log_{10}\left(\frac{\text{MAX}^2}{\text{MSE}}\right),$$

where MAX — is the maximum possible pixel value in the image (for example, 255 for 8-bit images), and MSE is the root mean square error.

Advantages of PSNR:

Sensitive to small changes in brightness and colour.

A high PSNR value indicates that the image is close to the original.

Why it's useful: PSNR allows you to evaluate how well the system preserves brightness and contrast after text replacement, which is important for ensuring the visual quality of the image after editing.

4. Fréchet Inception Distance (FID)

Fréchet Inception Distance (FID) — is a metric used to evaluate the quality of generated images based on the similarity of distributions between real and generated images. FID measures the distance between the Gaussian distributions of features from images that are extracted using a pre-trained neural network (usually InceptionV3) [13]. FID calculation formula:

$$\text{FID} = \left\|\mu_{\text{orig}} - \mu_{\text{gen}}\right\|^2 + \text{Tr}\,(\Sigma_{\text{orig}} + \Sigma_{\text{gen}} - 2(\Sigma_{\text{orig}}\Sigma_{\text{gen}})^{1/2}),$$

where μ_{orig} and μ_{gen} — average feature values for original and generated images, Σ_{orig} and Σ_{gen} — covariation of these features.

Advantages of:

Evaluates not only pixel similarity, but also deeper features of images, making it more flexible for evaluating high-level stylistic characteristics.

Metric sensitivity to small changes that can be important for generating visually realistic images.

Why it's useful: FID is one of the main metrics for evaluating the quality of generative models such as StyleGAN. It allows you to evaluate how well the system is able to generate realistic images while preserving the visual and stylistic characteristics of the text and overall composition.

Conclusion. The use of MSE, SSIM, PSNR, and FID metrics provides a compreh ensive approach to evaluating the performance of a style-preserving text-to-picture system. MSE and PSNR evaluate the pixel-by-pixel reproduction accuracy, SSIM takes into account structural similarity, and FID allows you to determine how realistic the generated images look. Together, these metrics help to comprehensively evaluate the quality of the system and its ability to accurately reproduce the stylistic and visual features of images after text replacement.

Verification and validation of the system engineering method

The fourth stage involves verification and validation of the conceptual model and the developed version of the image text replacement system with style preservation. The model was built on the basis of the TextStyleBrush model architecture and trained for 50 epochs on a dataset consisting of 20 thousand segmented images. The system was also verified and validated, and the results are shown in the following Table 2.

Table 2. Comparative analysis of the use of metrics to assess the quality of the developed system

Metric	Test set 1	Test set 2	Test set 3	Average
MSE	0.0196	0.0229	0.0288	0.0238
SSIM	0.5071	0.5489	0.4460	0.5007
PSNR	15.3077	15.8087	15.4113	15.5092
FID	1.3876	0.7256	1.1510	1.0881

Interpreting the results of metrics such as MSE, SSIM, PSNR, and FID can give you an idea of the quality of the generated images compared to the original ones. Here is a breakdown of what each metric typically shows:

1. Mean squared error (MSE)

MSE quantifies the root mean square difference between the pixel values of two images. Lower values indicate better similarity. Although there is no absolute scale, an MSE value close to zero indicates high similarity, while higher values indicate greater divergence.

2. Structural similarity index (SSIM)

The SSIM ranges from -1 to 1, where 1 indicates complete similarity. An SSIM value of 0.5 indicates moderate structural similarity between images. In general, values above 0.7 are considered acceptable, and values above 0.9 are considered excellent.

3. Peak signal-to-noise ratio (PSNR)

PSNR is a logarithmic measure that compares the maximum possible signal power with the power of distorting noise. In image processing, a PSNR of 20–30 dB is generally considered acceptable, while a value of more than 30 dB is good and more than 40 dB is excellent. A PSNR value of 15 dB indicates poor quality, often indicating that the generated images are very different from the originals.

4. Fréchet Inception Distance (FID)

FID compares the distribution of generated images to real images. Lower FID scores indicate better quality and diversity of the generated images. Scores below 10 are generally considered good, while scores above 50 indicate poor quality. A FID score of 1.0881 indicates that your GAN is generating high quality images that are very similar to the real dataset.

Overall score:

- MSE: Low, which is good.
- SSIM: Moderate; can be improved.
- PSNR: Low; indicates noticeable differences in image quality.
- FID: Excellent; indicates good GAN performance.

Conclusion. MSE, SSIM, and PSNR metrics are used in almost every work related to generative networks, they are simple and quick to calculate, but such simplicity is usually not suitable for evaluating such systems, it is more done to be able to compare results with older works. It is better to focus on the results of the FID metric, which more accurately reflects the result for our particular problem.

CONCLUSIONS

1. A method of system engineering of a conceptual model and a system for replacing text in images with preservation of style is proposed, based on the modified Eriksson–Penker business profile [5–8] of the system's representation at the meta-level, as well as international standards of DataScience [2] and DataMining [3] processes, which is the basis for algorithmizing the development of specific system components. The effectiveness of the method is investigated on the example of developing a system for automatic text replacement in an image while preserving the style.

2. The proposed method of system engineering is aimed at developing specialized systems designed to replace text on design layouts and various products of companies and brands that need to adapt their visual materials for different markets and languages for promoting goods and services.

3. The use of the system engineering method significantly speeds up and streamlines the implementation of a particular system and reduces the cost of its development, verification and validation.

4. Prospects for further research are aimed at applying the system engineering method to implement a system based on other mathematical models, forming performance evaluation metrics and scientifically sound methods for verifying and validating the method.

REFERENCES

1. P.P. Maslianko, O.S. Maystrenko, "The system engineering of organizational system informatization projects," *KPI Sci. News*, no. 6, pp. 34–42, 2008.
2. C. O'Neil, R. Schutt, *Doing data science: Straight talk from the frontline*. O'Reilly Media, Inc., 2013, 406 p.
3. F. Provost, T. Fawcett, *Data science for business: What you need to know about data mining and data-analytic thinking*. O'Reilly Media, Inc., 2013.
4. Pavlo P. Maslianko, Yevhenii P. Sielskyi, "Method of system engineering of neural machine translation systems," *KPI Science News*, no. 2, pp. 46–55, 2021. doi: https://doi.org/10.20535/kpisn.2021.2.236939
5. H.-E. Eriksson, M. Penker, *Business modeling with UML*. New York: John Wiley & Sons, 2000, 459 p.

6. A. Kossiakoff, W. Sweet, S. Seymour, S. Biemer, *System Engineering Principles and Practice*. M.: DMK Press, 2014, 624 p.
7. D.K. Hitchins, *Systems Engineering: A 21st Century Systems Methodology*. Wiley, 2007, 528 p.
8. S.Krymskyi, "Metod," in *Filosofskyi Entsyklopedychnyi Slovnyk; V.I. Shynkaruk, Ed*. Kyiv, Ukraine: Abrys, 2002, 742 p. doi: https://doi.org/10.20535/ kpisn.2021.2.236939
9. Praveen Krishnan, Rama Kovvuri, Guan Pang, Boris Vassilev, Tal Hassner, "TextStyleBrush: Transfer of Text Aesthetics from a Single Example," *Journal of Latex Class Files*, vol. 14, no. 8, August 2015. Available: https://arxiv.org/pdf/ 2106.08385
10. X. Huang, S. Belongie, "Arbitrary Style Transfer in Real-Time with Adaptive Instance Normalization," *2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 2017*, pp. 1510–1519. doi: https://doi.org/10.1109/ICCV.2017.167
11. Peter Ndajah, Hisakazu Kikuchi, Masahiro Yukawa, Hidenori Watanabe, Shogo Muramatsu, "SSIM image quality metric for denoised images," *International Conference on Visualization, Imaging and Simulation(VIS '10)*, pp. 53–57, 2010.
12. A. Horé, D. Ziou, "Image Quality Metrics: PSNR vs. SSIM," *20th International Conference on Pattern Recognition, Istanbul, Turkey, 2010*, pp. 2366–2369. doi: https://doi.org/10.1109/ICPR.2010.579
13. Yaniv Benny, Tomer Galanti, Sagie Benaim, Lior Wolf, "Evaluation Metrics for Conditional Image Generation," *International Journal of Computer Vision*, 129, pp. 1712–1731, 2021. doi: https://doi.org/10.1007/s11263-020-01424-w

Received 25.11.2024

INFORMATION ON THE ARTICLE

Pavlo P. Maslianko, ORCID: 0000-0003-4001-7811, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Ukraine, e-mail: masliankop@gmail.com

Mykola D. Romanov, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Ukraine, e-mail: kolya.romanov8@gmail.com

КОНЦЕПТУАЛЬНА МОДЕЛЬ ТА СИСТЕМА ДЛЯ ЗАМІНИ ТЕКСТУ НА ЗОБРАЖЕННІ ЗІ ЗБЕРЕЖЕННЯМ СТИЛЮ / П.П. Маслянко, М.Д. Романов

Анотація. Заміна тексту на зображенні, зокрема, зі збереженням його стилю, є складним завданням, яке потребує вирішення низки наукових задач та нових технічних рішень. Однією з основних проблем є збереження автентичності та гармонійності зображення після внесення змін. Мета дослідження — розроблення концептуальної моделі та системи заміни тексту на зображеннях зі збереженням його стилю на основі методології системної інженерії та бізнес-профілю Еріксона–Пенкера, забезпечуючи природну інтеграцію нових текстових елементів у контекст зображення. Методика реалізації — методологія системної інженерії і бізнес-профіль Еріксона–Пенкера для формалізації впорядкованого процесу розроблення системи для заміни тексту на зображенні зі збереженням стилю. Результати дослідження — метод розроблення системи на основі застосування технік системної інженерії, який складається з чотирьох основних етапів. На першому етапі структуру системи моделюють як бізнес-профіль Еріксона–Пенкера, на другому — визначають множину процесів, характерну для класу систем Data Science та міжнародного стандарту CRISP-DM, моделюють структурне і динамічне представлення концептуальної моделі системи та інтерфейси взаємодії компонентів, на третьому етапі виконують імплементацію конкретної версії системи, а на четвертому — верифікація та валідація системи. Запропоновано метод системної інженерії концептуальної моделі та системи заміни тексту на зображеннях зі збереженням його стилю, що ґрунтується на модифікованому бізнес-профілі Еріксона–Пенкера подання системи на метарівні, а також міжнародних стандартів процесів Data Science та Data Mining.

Ключові слова: метод системної інженерії, бізнес профіль Еріксона–Пенкера, концептуальна модель, система для заміни тексту на зображенні зі збереженням стилю.