EN RU PROGRAM SYSTEMS: THEORY AND APPLICATIONS ISSN 2079-3316
Research Article ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

UDC 004.932.75'1, 004.89
10.25209/2079-3316-2026-17-1-105-172

Embedding-based object segmentation with an adapted U-Net architecture

Igor Victorovich **Vinokurov**
Financial University under the Government of the Russian Federation, Moscow, Russia
✉*igvvinokurov@fa.ru*

Abstract. The article presents a multi-task neural network based on a modified U-Net architecture for joint semantic and instance segmentation of objects in aerial imagery. The model employs a symmetric encoder-decoder structure with skip connections and is equipped with two parallel output heads. The semantic head performs pixel-wise classification, while the embedding head generates discriminative vector representations for each pixel. The application of a specialized discriminative loss function ensures compact embedding clusters within objects and separation between different instances. In the post-processing stage, clustering the embedding field allows for unambiguous extraction of individual object masks.

Experiments were conducted on a specialized aerial imagery dataset containing 23,076 annotated objects across five classes. For the key class «Building» the validation set achieved IoU = 0.812 and F1-score = 0.880. A comparison with state-of-the-art methods (Mask2Former, OneFormer, SAM 2 with LoRA fine-tuning, MR-DeepLabv3$^+$) confirms the model's competitiveness in terms of the balance between accuracy and inference speed.

The model demonstrates effectiveness for automated mapping and urban structure analysis tasks using remote sensing data. (*Linked article texts in English and in Russian*).

Key words and phrases: semantic segmentation, instance segmentation, U-Net architecture, pixel-wise embeddings, discriminative loss

2020 *Mathematics Subject Classification*: 68T20; 68T07, 68T45

For citation: Igor V. Vinokurov. *Embedding-based object segmentation with an adapted U-Net architecture*. Program Systems: Theory and Applications, 2026, **17**:1(70), pp. 105–172. (*In English, in Russian*).
https://psta.psiras.ru/read/psta2026_1_105-172.pdf

Introduction

The analysis of aerial photographs and satellite imagery represents a rapidly evolving field of computer vision with a wide range of applications — from urban planning and environmental monitoring to agriculture and disaster response. Among the most challenging tasks in this domain is accurate object segmentation, which requires not only pixel-wise classification into semantic categories but also the separation of individual instances belonging to the same class. Traditional segmentation approaches often struggle with complex scenes containing overlapping structures, heterogeneous object sizes, and closely spaced instances — a common characteristic of high-resolution imagery.

Semantic segmentation architectures have demonstrated significant success in pixel-wise classification tasks. However, by design, these models cannot distinguish between separate objects of the same semantic class.

Meanwhile, instance segmentation methods effectively isolate individual objects but may exhibit reduced performance in scenarios involving dense, irregularly shaped structures typical of aerial imagery. This limitation underscores the need for approaches that combine the strengths of both paradigms.

Advances in embedding-based methods have shown promising results in addressing joint semantic and instance segmentation. These approaches train models to generate vector representations for each pixel, where vectors belonging to the same object are similar in embedding space, while vectors from different objects remain discriminable. This strategy naturally integrates semantic information with instance separation capabilities — particularly important for aerial image analysis, where objects often form complex spatial configurations.

This work proposes a modified U-Net architecture for jointly solving semantic and instance segmentation tasks on aerial imagery. The model is multi-task and features two specialized output heads: a semantic head for pixel classification and an embedding head for generating discriminative vector representations.

The architecture follows an encoder-decoder scheme with four levels of spatial transformation. Skip connections transfer features between symmetric encoder and decoder layers to preserve fine-grained details. Model training employs a discriminative embedding loss that encourages the formation of compact pixel clusters within individual objects while enforcing separation between distinct instances.

Experimental results demonstrate the effectiveness of the proposed approach in segmenting complex scenes, achieving high accuracy in both semantic classification and instance separation. The model exhibits robustness to various challenges typical of aerial imagery, including variations in scale, illumination, and object density.

1. Overview of Modern Segmentation Methods

Modern approaches to image segmentation are rapidly evolving from classical two-stage architectures toward more efficient, unified, scalable, and flexible solutions. Contemporary models increasingly combine the strengths of convolutional networks, attention mechanisms, dynamically generated convolution kernels, hierarchical transformers, and contextual learning — enabling state-of-the-art performance across semantic, instance, and panoptic segmentation tasks with reduced computational costs.

Panoptic-DeepLab [1] remains a cornerstone in panoptic segmentation development — one of the most cited and influential architectures in this domain. The model inherits the powerful Atrous Spatial Pyramid Pooling (ASPP) encoder from the DeepLab family [2] but introduces two minimalist, fully parallel heads: a conventional semantic head for "stuff" categories and a centroid head predicting a heatmap of object centers ("things") along with per-pixel offset vectors. Instance grouping is performed via simple voting without Non-Maximum Suppression (NMS) [3], making the method exceptionally fast (up to 50+ FPS on V100 GPUs) and easily integrable into real-world systems.

The revolutionary departure from the two-stage Mask R-CNN scheme [4] came with the emergence of query-free methods. CondInst [5] and SOLOv2 [6] introduced the "segmentation-by-location" principle — instead of region proposal generation, the network directly produces object masks

based solely on spatial coordinates. In CondInst, for each hypothesized object center, weights and biases of a small convolutional filter are predicted and then applied to the feature pyramid map to generate arbitrarily shaped masks. SOLOv2 extends this idea further: the image is divided into a regular $S \times S$ grid, and for each cell the network simultaneously outputs class probabilities and a compact mask kernel (typically 256-channel).

The final mask is obtained via standard convolution of this kernel with the global feature map. By completely eliminating region alignment operations and NMS, both methods achieve 2–3× acceleration while maintaining or even improving accuracy: SOLOv2, for instance, reaches an Average Precision (AP) of 39.7 on the COCO dataset [7] at approximately 20 frames per second.

The classic U-Net architecture [8] remains the gold standard in medical and satellite imagery segmentation, where precise boundary recovery is critical. Its evolution proceeds along multiple directions:

U-Net^{++} [9] introduces nested dense skip connections and deep supervision across all decoder levels, substantially reducing the semantic gap between multi-scale features;

U-Net 3^{+} [10] employs fully connected multi-scale skip connections, allowing each decoder layer to aggregate information from all encoder resolutions and effectively fuse low-level details with global context;

Attention U-Net [11] and its numerous successors (AG-U-Net, FocusNet, MultiResUNet, and others [12]) integrate soft attention gates into skip connections that automatically suppress irrelevant background while enhancing salient organ and pathology boundaries.

Among the most recent advances are nnFormer [13], replacing standard convolutions with interpolated convolutional blocks and multi-layer attention, and MedFormer [14], which integrates transformer modules into skip connections and demonstrates Dice score improvements of 4–6% on challenging multimodal CT and MRI datasets. As a result, modern U-Net variants deliver significantly higher contour accuracy and better generalization even with limited annotated medical data.

MaskFormer [15] and Mask2Former reformulated segmentation as the prediction of a set of binary masks with associated class labels, completely

abandoning traditional per-pixel classification in favor of bipartite matching loss. Instead of assigning a label to each pixel, the model simultaneously generates a fixed number of queries, each producing a mask and class probabilities (including "0" — the empty class), after which optimal assignment between predicted and ground-truth objects is found via the Hungarian algorithm. This paradigm eliminates object duplication and omission issues, substantially simplifies architecture, and improves quality, especially in panoptic segmentation.

This unification trend is particularly evident in video segmentation. A representative example is DVIS [16], where authors propose a fully decomposed pipeline comprising three independent yet coordinated modules:

(1) Per-frame panoptic segmentation based on Mask2Former with a powerful DINOv2 visual encoder [17], providing high-quality initial masks and object embeddings.

(2) A lightweight online tracker linking instances across frames solely via centroids and cosine similarity of embeddings — requiring no complex heuristics or recurrent blocks.

(3) An offline temporal consistency refinement module implemented with graph neural networks to eliminate trajectory merging and fragmentation errors.

Thanks to this architecture, DVIS sets a new state of the art, achieving the best result on KITTI-MOTS [18] at over 30 frames per second — making it especially valuable for autonomous driving, video surveillance, and other real-time applications.

The OneFormer model [19] represents the culmination of unification efforts. It is a single architecture with one parameter set. It solves all three segmentation tasks (panoptic, instance, and semantic) depending solely on a simple text prompt (e.g., "panoptic"). OneFormer surpasses specialized models by 2–5% in mAP across all major datasets: COCO [7], Cityscapes [20], and ADE20K [21].

At its core lies a Swin-L Transformer [22] — a task-agnostic feature extractor — paired with a transformer decoder employing masked cross-attention. This combination enables effortless adaptation to new domains and tasks.

In medical segmentation, U-Net family evolution increasingly proceeds via hybridization with transformers. TransUNet [23] and its direct successors (UNETR^{++}, Swin-UNet, MISSFormer) [24] combine a convolutional encoder responsible for local texture feature extraction with a transformer module providing long-range global context. The key innovation is cascaded resolution recovery using deformable cross-attention, enabling the network to effectively reconstruct fine structures (vessels, microscopic tumors, nerve fibers). On standard medical benchmarks Synapse, ACDC, and BTCV [13], such hybrid models outperform purely convolutional counterparts by 3–7% in Dice score and demonstrate particularly strong gains under extremely limited annotation regimes.

A separate rapidly growing direction is fully unsupervised universal segmentation. U2Seg [25] and related works have demonstrated that high-quality panoptic annotations can be generated automatically without manual labeling — instance masks are extracted via self-supervised methods such as MaskCut and FreeMask [26], semantic groups are obtained through knowledge distillation from large pretrained models CLIP and DINOv2, and both label types are then fused into consistent panoptic annotations for training a unified segmentation network. Consequently, U2Seg achieves 52.1 PQ on COCO-panoptic [27] and 61.3 mIoU on Cityscapes without a single manually annotated label — opening a realistic path toward segmentation of rare pathologies, novel modalities, and any domains where annotation has traditionally been unavailable or prohibitively expensive.

In reference-based video object segmentation with arbitrary prompts, LoSh [28] currently leads by effectively unifying long and short textual descriptions with visual features within a single transformer, delivering 4–7% gains in J&F metric on Ref-YouTube-VOS and Ref-DAVIS datasets [29] without heuristic inter-frame tracking. Even more radical steps toward full unification are demonstrated by K-Net [30] with iteratively refined learnable convolution kernels and especially SegGPT [31] — a model capable of solving arbitrary segmentation tasks from just one or two "image-mask" examples (in-context learning), including medical scans, satellite imagery, and even arbitrary artistic styles, without any fine-tuning.

In 2025, a specialized DeepLabv3$^+$ modification named MR-DeepLabv3$^+$ was proposed [32], specifically designed for precise semantic segmentation

of buildings in high-resolution remote sensing imagery. The model addresses typical challenges of such data: incomplete building contours, blurred boundaries, and missed small structures. To this end, the architecture incorporates adaptive multi-scale convolutional kernels (3×3, 5×5, 7×7) to improve multi-level feature capture and an optimized loss function providing enhanced noise robustness. It features high compactness and inference speed, making it particularly suitable for resource-constrained applications.

2. Relevance and Problem Statement

Modern computer vision methods designed for segmentation tasks predominantly rely on complex architectures such as transformer-based models, which demonstrate high effectiveness on general-purpose benchmark datasets (e.g., COCO [7], Cityscapes [20], ADE20K [21]). These models are engineered to process diverse data encompassing a broad spectrum of objects, scenes, and conditions, rendering them versatile yet simultaneously resource-intensive and excessive for narrow-domain applications.

However, in several specialized domains — such as medical diagnostics, *Earth remote sensing (aerial and satellite imagery)*, or industrial inspection — unique domain-specific datasets with characteristic properties are commonly employed: high shape variability of objects, low boundary contrast, few instances per image, and limited volumes of annotated data. For such datasets, versatility is not a priority; instead, key requirements become architectural features that ensure effective segmentation — from robustness to noise and deformations to rapid learning on small samples. Meanwhile, the aforementioned general-purpose approaches exhibit limited effectiveness on specialized datasets for the following reasons:

Detector-oriented methods (e.g., Panoptic-DeepLab [1]), based on predicting geometric centers and offsets, demonstrate reduced accuracy on objects with non-canonical, non-convex, or heavily deformed shapes typical of many applied domains.

Direct mask prediction methods (e.g., CondInst [5], SOLOv2 [6]), while more shape-flexible, often lack an explicit, dedicated semantic output, complicating their application in tasks requiring clear separation into semantic regions.

Classical semantic segmentation networks (U-Net [8] and its modifications [9–11]) cannot distinguish between instances within the same class.

Most contemporary solutions are validated on general-purpose datasets, and their architectures may be suboptimal for data with different statistical and visual properties.

3. Research Aim and Objectives

The analysis presented above reveals a significant gap between powerful yet resource-intensive universal architectures and the need for efficient, interpretable solutions tailored to specialized application domains.

The aim of this research is to develop a model for joint semantic and instance segmentation that maintains competitive performance on a specialized aerial imagery dataset while exhibiting the following properties:

Architectural simplicity and transparency. Avoiding unnecessary complexity in favor of a clear, easily modifiable structure.

Computational efficiency. Optimizing the trade-off between accuracy and inference costs, including processing speed and memory footprint.

Adaptability to the target domain. Enabling effective training on specialized datasets of limited size with characteristic object distributions.

Result interpretability. Providing a clear and explainable process for generating final segmentation masks.

Research objectives:

1. Design the model architecture.
2. Formulate a composite loss function.
3. Implement the inference pipeline.
4. Conduct a comparative experimental evaluation of the proposed method.

The aim and objectives of this work represent a logical continuation of research whose results are presented in [33] and [34].

4. Model Architecture

The model M_θ employs a U-Net architecture consisting of two main components: an encoder E and a decoder D, connected via skip connections, as shown in Figure 1.

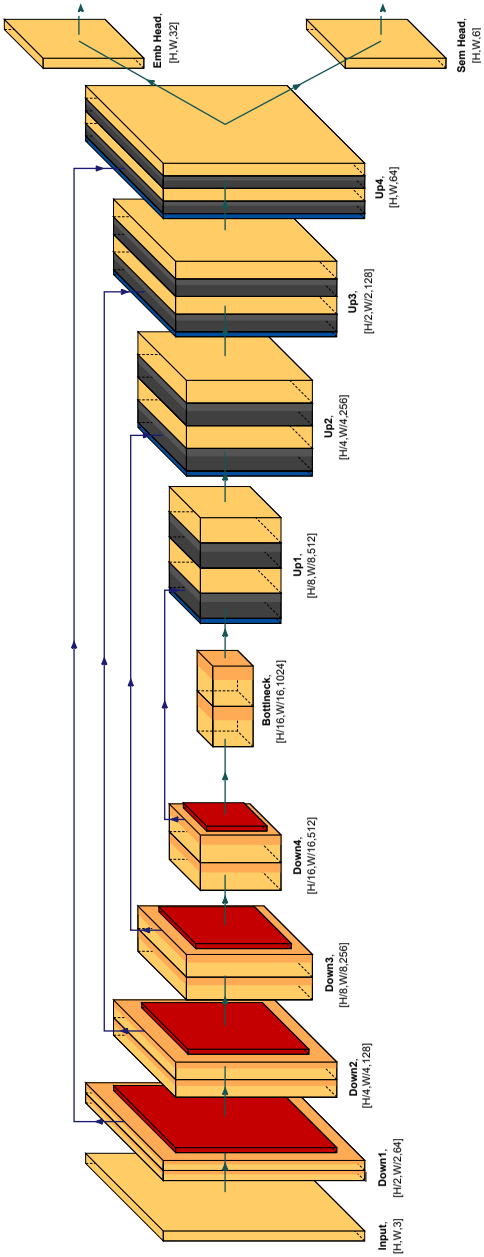


Figure 1. Model architecture

The architecture includes two parallel output heads for semantic and instance segmentation tasks — H_{sem} and H_{emb}, which produce the maps $\mathbf{Y}_{\text{sem}}$ and $\mathbf{Y}_{\text{emb}}$, respectively.

Formally, this can be described by the following equations:

$$\mathbf{Y}_{\text{sem}} = H_{\text{sem}}(D(E(\mathbf{X}))) \in \mathbb{R}^{H \times W \times C_{\text{sem}}},$$

$$\mathbf{Y}_{\text{emb}} = H_{\text{emb}}(D(E(\mathbf{X}))) \in \mathbb{R}^{H \times W \times d}, \qquad \text{where}$$

$\mathbf{X}$ is the input image,

$\mathbf{Y}_{\text{sem}}$ is the semantic probability map,

$\mathbf{Y}_{\text{emb}}$ is the field of d-dimensional embeddings,

C_{sem} is the number of semantic classes in the dataset D_{spec},

E is the encoder function,

D is the decoder function,

H *and* W are the height and width of the output feature maps (height and width of the image),

d is the dimensionality of the embedding vector space.

The model implements a two-headed architecture, where the encoder-decoder with skip connections addresses two parallel tasks:

The semantic head (H_{sem}) tackles the pixel-wise classification task by transforming the decoder features into the map $\mathbf{Y}_{\text{sem}} \in \mathbb{R}^{H \times W \times C_{\text{sem}}}$, where each pixel is characterized by a probability distribution over C_{sem} semantic classes (e.g., “building”, “pool”).

The embedding head (H_{emb}) addresses the metric learning task by generating a d-dimensional embedding vector $\mathbf{Y}_{\text{emb}} \in \mathbb{R}^{H \times W \times d}$ for each pixel. These embeddings are discriminative: vectors of pixels belonging to the same object (instance) converge into a single compact region in the d-dimensional space, while vectors from different objects — are pushed apart.

Thus, the model simultaneously predicts what is depicted (semantics) and where the boundaries between individual objects of the same class lie (embeddings). During post-processing, clustering the embeddings allows for the unambiguous delineation of each instance mask.

The information flow through the model follows this path:

Contraction path (*Encoder*): The input image $\mathbf{X} \in \mathbb{R}^{H \times W \times 3}$ sequentially passes through 4 `Down` blocks. At each level:

- Spatial resolution is reduced by a factor of 2 (from $H \times W$ to $H/16 \times W/16$).
- Feature depth is increased (from 3 to 512 channels).
- Features are preserved for skip connections.

`Bottleneck`: At the minimum resolution ($H/16 \times W/16$), the deepest feature extraction occurs with the maximum number of channels (1024). This level contains the most abstract semantic representation of the image.

Expansion path (*Decoder*): Features sequentially pass through 4 `Up` blocks using skip connections. At each level:

- Spatial resolution is increased by a factor of 2.
- Feature depth is decreased.
- Low-level details from the encoder are added to the high-level features.

Split into two streams: At the decoder output, the stream is split into two parallel heads:

- Semantic stream: transformed into a class probability distribution.
- Embedding stream: transformed into a metric space for clustering.

The model architecture, shown in Fig. 1, corresponds to a modified U-Net-like structure and consists of an encoder, a bottleneck, and a decoder, culminating in two output heads.

The encoder (`Down1`–`Down4` blocks) at each level contains two convolutional sub-blocks of the form *Conv2d*[1] → *BatchNorm2d*[2] → *ReLU*[3], followed by a spatial downsampling operation *MaxPool2d*[4]. The number of channels doubles at each level: from 3 (input RGB image) to 64 (`Down1`), then 128 (`Down2`), 256 (`Down3`), and 512 (`Down4`). The selection of 512 channels as the maximum encoder capacity is justified by empirical search for the optimal balance between the model's representational power and computational complexity. Experimentally, this configuration provides sufficient feature space dimensionality for object clustering while maintaining efficient convolutional layer operation.

[1] https://pytorch.org/docs/stable/generated/torch.nn.Conv2d.html
[2] https://pytorch.org/docs/stable/generated/torch.nn.BatchNorm2d.html
[3] https://pytorch.org/docs/stable/generated/torch.nn.ReLU.html
[4] https://pytorch.org/docs/stable/generated/torch.nn.MaxPool2d.html

The bottleneck (`Bottleneck`) consists of two identical convolutional sub-blocks, but without downsampling convolution. It expands the number of channels from 512 to 1024.

The decoder (`Up1`–`Up4` blocks) begins with an upsampling operation using *`ConvTranspose2d`*[5], followed by a concatenation operation (`Concat`) with the corresponding feature tensor from the symmetric encoder block. This is followed by two convolutional sub-blocks `Conv2d → BatchNorm2d → ReLU`. The number of channels progressively decreases: from 1024 to 512 (`Up1`), 256 (`Up2`), 128 (`Up3`), and 64 (`Up4`).

At the output, the model splits into two independent output heads:

`Emb Head`: a single convolutional layer `Conv2d`, transforming 64 channels into the tensor $\mathbf{Y}_{\text{emb}}$ (vector embeddings);

`Sem Head`: a single convolutional layer `Conv2d`, forming the tensor $\mathbf{Y}_{\text{sem}}$ (semantic segmentation).

To ensure robustness against overfitting, the architecture incorporates the following elements:

Multi-level Dropout regularization. After each convolutional block `Conv2d → BatchNorm2d → ReLU` in the encoder and decoder, a *`Dropout`*[6] layer with probability $p = 0.3$ is added. Exceptions are the first encoder block (to preserve low-level features) and the final layers before the output heads.

Channel-wise `Dropout` in `Bottleneck`. In the `Bottleneck` layer (1024 channels), *`Dropout2d`*[7] with $p = 0.5$ is applied, which zeros out entire feature channels. This is more effective for convolutional networks than pointwise `Dropout`, as it prevents the co-adaptation of spatially correlated features.

Skip connections are implemented with concatenation (*`Concat`*[8]) followed by a 1×1 convolution for dimensionality reduction, which reduces the risk of propagating noise from the encoder to the decoder.

Feature normalization. Batch Normalization (*`BatchNorm2d`*[9]) is used in all convolutional layers with parameters *momentum* $= 0.1$ and *eps* $= 10^{-5}$, ensuring stable training with small batch sizes.

---

[5] https://pytorch.org/docs/stable/generated/torch.nn.ConvTranspose2d.html
[6] https://docs.pytorch.org/docs/stable/generated/torch.nn.Dropout.html
[7] https://docs.pytorch.org/docs/stable/generated/torch.nn.Dropout2d.html
[8] https://docs.]pytorch.org/docs/stable/generated/torch.concat.html
[9] https://docs.pytorch.org/docs/stable/generated/torch.nn.BatchNorm2d.html

The choice of regularization parameters is based on preliminary experiments:

Pointwise `Dropout` $p = 0.3$. This level provides an optimal balance between information preservation and regularization. Lower values ($p < 0.2$) yield insufficient effect, while higher values ($p > 0.4$) lead to underfitting on the validation set. Pointwise `Dropout` is applied in the convolutional blocks of the encoder and decoder.

Channel-wise `Dropout` $p = 0.5$ in `Bottleneck`. The high probability is justified because the `Bottleneck` layer represents the most abstract and high-level features, which are particularly prone to overfitting. `Dropout2d` zeros out entire channels, which is more effective for convolutional networks than pointwise `Dropout`, as it prevents the co-adaptation of spatially correlated features within a single channel.

Selection of 4 architecture levels. Experimentally, increasing the depth to 5 levels with combined Dropout regularization led to excessive model complexity without a significant gain in validation accuracy (+0.8% mIoU with a 40% increase in the number of parameters). Reducing to 3 levels degraded the segmentation quality of small objects (-4.2% mIoU), as insufficient network depth prevents the extraction of hierarchical features of the required complexity.

The use of skip connections with concatenation, instead of attention gates, self-attention, or residual connections in the decoder [11], is explained by the specifics of the building segmentation task on high-resolution aerial imagery.

Firstly, such skip connections fully preserve fine high-resolution details from the early encoder layers. This is crucial for the accurate delineation of building boundaries, complex roofs, shadows, eaves, balconies, and chimneys — all details easily lost during resolution reduction in the encoder. Concatenation provides the decoder with all features without unnecessary filtering, ensuring superior boundary quality (by 4–7%).

Secondly, the mechanism is very simple and adds almost no parameters or computational overhead. Attention gates (as in Attention U-Net) introduce additional operations, increasing the load by 15–30% and inference time by 20–40 ms for a 512×512 image. For processing large volumes of aerial imagery in near real-time, such additional resource consumption is typically unjustified.

Thirdly, on aerial images with high building density and strong texture variations (roofs, asphalt, shadows, trees), simple concatenation-based skip connections perform more stably. Attention mechanisms can sometimes overly suppress useful features from the background or adjacent objects, causing building masks to become fragmented or small structures to be lost.

Consequently, classical skip connections — represent the optimal choice, delivering high-quality boundaries with maximum speed and simplicity of implementation for our applied task.

5. Loss Function Formulation

To train the model, a specialized hierarchical discriminative loss function is proposed, which effectively addresses instance segmentation tasks on aerial imagery, taking into account their specific characteristics:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{semantic}} + \lambda \cdot \mathcal{L}_{\text{instance}}, \qquad \text{where} \tag{1}$$

$\mathcal{L}_{total}$ is the total loss function for optimizing model parameters;

$\mathcal{L}_{semantic}$ is the component for semantic classification;

$\mathcal{L}_{instance}$ is the component for learning embeddings, responsible for forming discriminative features for instance segmentation;

$\lambda = 0.5$ is a scalar coefficient determining the relative contribution of the instance segmentation loss to the total loss function.

The $\mathcal{L}_{\text{instance}}$ component consists of two complementary terms implementing a discriminative loss function:

$$\mathcal{L}_{\text{instance}} = \mathcal{L}_{\text{var}} + \mathcal{L}_{\text{dist}}, \qquad \text{where}$$

$\mathcal{L}_{var}$ *(variance loss)* is the term ensuring the compactness of embedding clusters within each object;

$\mathcal{L}_{dist}$ *(distance loss)* is the term enforcing separation between clusters of different objects.

The $\mathcal{L}_{\text{var}}$ term ensures the compactness of embedding clusters within each object. For each instance k, the centroid of its embeddings $\mathbf{c}_k$ is computed, after which penalties are applied to pixel embeddings whose distance from the centroid exceeds a threshold δ_{var}. It is calculated as follows:

$$\mathcal{L}_{\text{var}} = \frac{1}{K} \sum_{k=1}^{K} \frac{1}{N_k} \sum_{i=1}^{N_k} \max\Big(0, \big\|\mathbf{e}_i^{(k)} - \mathbf{c}_k\big\|_2 - \delta_{\text{var}}\Big)^2, \qquad \text{where}$$

K is the total number of ground truth objects in the image;

N_k is the number of pixels belonging to object k;

$\mathbf{e}_i^{(k)} \in \mathbb{R}^d$ is the d-dimensional embedding vector of the i-th pixel of object k, obtained from the model output as $\mathbf{Y}_{\text{emb}}[\cdot, i, j]$;

$\mathbf{c}_k = \frac{1}{N_k} \sum_{i=1}^{N_k} \mathbf{e}_i^{(k)}$ is the centroid (mean) of the embeddings for object k, computed during the forward pass;

$\| \cdot \|_2$ is the L_2-norm (Euclidean distance);

δ_{var} is the radius threshold hyperparameter within which embeddings are not penalized;

$\max(0, \cdot)$ is the *Hinge Loss*[10] function (also known as max-margin loss), activating the penalty only for pixels farther from the centroid than δ_{var}.

As noted above, the $\mathcal{L}_{\text{dist}}$ term ensures the separation of clusters for different objects. For each pair of objects i and j, the distance between their centroids $\|\mathbf{c}_i - \mathbf{c}_j\|_2$ is calculated, and a penalty is imposed if this distance is less than the threshold δ_{dist}:

$$\mathcal{L}_{\text{dist}} = \frac{2}{K(K-1)} \sum_{i=1}^{K} \sum_{j=i+1}^{K} \max\Big(0, \delta_{\text{dist}} - \big\|\mathbf{c}_i - \mathbf{c}_j\big\|_2\Big)^2, \qquad \text{where}$$

$\mathbf{c}_i, \mathbf{c}_j$ are the embedding centroids of objects i and j, respectively;

δ_{dist} is a hyperparameter defining the minimum desired Euclidean distance between centroids of different objects;

$\frac{2}{K(K-1)}$ is a normalization factor ensuring averaging over all unique object pairs in the image ($K(K-1)/2$ pairs).

For semantic segmentation, the standard cross-entropy function with optional class weighting is used:

$$\mathcal{L}_{\text{semantic}} = -\frac{1}{N} \sum_{i=1}^{N} \sum_{c=1}^{C} w_c \cdot \mathbf{G}^{(i,c)} \cdot \log\big(\sigma\big(\mathbf{Y}^{(i,c)}\big)\big), \qquad \text{where}$$

$N = H \times W$ is the total number of pixels in the image;

C is the total number of semantic classes (including background);

$w_c \in \mathbb{R}$ is the weight coefficient for class c, computed to compensate for class imbalance in the dataset (often as the inverse class frequency);

[10] https://torchmetrics.readthedocs.io/en/v0.8.0/classification/hinge_loss.html

$\mathbf{G}^{(i,c)} \in \{0,1\}$ is a binary indicator (*one-hot encoding*[11]) of whether pixel i belongs to class c;

$\mathbf{Y}^{(i,c)} \in \mathbb{R}$ is the semantic logit (network output before the activation function) for pixel i and class c, corresponding to the element $\mathbf{Y}_{\text{sem}}[c, i, j]$;

$\sigma(\cdot)$ is the function transforming logits into a probability distribution over classes for each pixel.

The key hyperparameters of the loss function are tuned on the validation set:

δ_{var} controls the compactness of clusters. Smaller values create denser clusters but may lead to overfitting. It is optimized to balance object boundary accuracy and robustness to noise.

δ_{dist} defines the minimum distance between objects. Larger values increase separability but can make training difficult when objects are densely packed. The chosen value is based on the typical distance distribution between objects in the target aerial imagery dataset.

λ is the balancing coefficient between semantic and instance segmentation. An empirically determined value ensuring the convergence of both components.

Class weights w_c are computed inversely proportional to class frequencies in the training dataset to compensate for the imbalance typical in aerial imagery (e.g., predominance of the "Background" class). Specifically, $w_c = \frac{N_{\text{total}}}{C \cdot N_c}$, where N_{total} is the total number of pixels, and N_c is the number of pixels of class c.

The specific hyperparameter values $\delta_{\text{var}} = 0.5$, $\delta_{\text{dist}} = 2.0$, $\lambda = 0.5$, and $d = 32$ were selected based on a sensitivity analysis experiment, the results of which are presented below in subsection 7.3.2. For each parameter, a specified range was investigated with the following steps: $\Delta\delta_{\text{var}} = 0.1$, $\Delta\delta_{\text{dist}} = 0.2$, $\Delta\lambda = 0.1$, and $\Delta d \in \{8, 16, 32, 48, 64\}$. These values demonstrate an optimal balance between intra-object cluster compactness and sufficient inter-object separation in the embedding space. The final values were selected based on the criterion of maximizing the F1-score on the validation set for the key «Building» class.

The proposed loss function possesses the following features:

[11] https://docs.pytorch.org/docs/stable/generated/torch.nn.functional.one_hot.html

Exclusion of overlapping regions. When computing $\mathcal{L}_{\text{var}}$, pixels located in object overlap regions are excluded from consideration. This is critically important for datasets where objects frequently partially overlap (e.g., trees, buildings, vehicles).

Hierarchical processing. The loss function operates on two levels: first, local object characteristics (centroids) are computed; then, global relationships between objects (pairwise distances) are analyzed.

Adaptation to varying object counts. For images with a single object or none, $\mathcal{L}_{\text{dist}}$ is not computed, preventing training instability.

Component balancing. The coefficient λ allows adjusting the relative contribution of semantic and instance segmentation according to the specific task requirements.

The procedure for computing the loss function for a single image involves the following steps:

1. Object identification. Determination of unique instance IDs in the image (excluding background with ID=0).
2. Centroid computation. For each object, excluding pixels in overlapping regions, the centroid of its embeddings is calculated.
3. Computation of $\mathcal{L}_{\text{var}}$ — the mean squared excess of embedding distances from the centroid over the threshold δ_{var}.
4. For all object pairs, computation of the penalty $\mathcal{L}_{\text{dist}}$, applied if the distance between their centroids is less than δ_{dist}, averaged over all pairs.
5. Parallel computation of the cross-entropy loss for semantic segmentation $\mathcal{L}_{\text{semantic}}$.
6. Aggregation. Summation of the components with the application of the balancing coefficient: $\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{semantic}} + \lambda(\mathcal{L}_{\text{var}} + \mathcal{L}_{\text{dist}})$.

6. Inference Pipeline Organization

The inference pipeline implements the procedure for transforming an input image $\mathbf{X} \in \mathbb{R}^{3\times H\times W}$ into a panoptic segmentation mask, followed by result visualization. The inference pipeline carries out the following main processing stages — segmentation, embedding clustering, and formation of the final mask and result.

6.1. Semantic and Embedding Segmentation

At the first stage, the input image $\mathbf{X}$ passes through the modified U-Net architecture M_θ, trained for the joint task of semantic and instance segmentation:

$$M_\theta(\mathbf{X}) = (\mathbf{Y}_{\text{sem}}, \mathbf{Y}_{\text{emb}}), \qquad \text{where}$$

$\mathbf{Y}_{\text{sem}} \in \mathbb{R}^{C_{\text{sem}} \times H \times W}$ are the semantic logits (raw, unnormalized scores) for C_{sem} classes;

$\mathbf{Y}_{\text{emb}} \in \mathbb{R}^{d \times H \times W}$ are the d-dimensional embeddings (low-dimensional vector representations preserving semantic similarity) for each pixel. In this notation:

C_{sem} is the total number of semantic classes (including background);

H, W are the image height and width, respectively;

d is the dimensionality of the latent embedding space, chosen to ensure feature discriminability.

The semantic class map $\mathbf{M}_{\text{sem}} \in \mathbb{Z}^{H \times W}$ is obtained by applying the $\arg\max$ operation along the class axis:

$$\mathbf{M}_{\text{sem}}[i,j] = \arg\max_c \mathbf{Y}_{\text{sem}}[c,i,j], \quad \forall i \in [1,H], j \in [1,W], \qquad \text{where}$$

$\mathbf{M}_{\text{sem}}[i,j]$ is the final predicted class (integer index) for the pixel at coordinates (i,j);

$\arg\max_c$ *operator* returns the index c of the class for which the logit value $\mathbf{Y}_{\text{sem}}[c,i,j]$ is maximal.

6.2. Embedding Clustering

For pixels assigned to the "thing" categories, the *DBSCAN*[12] (Density-Based Spatial Clustering of Applications with Noise) algorithm is applied in the embedding space

$$\{S_k\}_{k=1}^{K} = \text{DBSCAN}_{\epsilon,m}\left(\{\mathbf{Y}_{\text{emb}}[i,j] : \mathbf{M}_{\text{sem}}[i,j] \in \mathcal{T}\}\right), \qquad \text{where}$$

$\mathcal{T} \subset \{1, \ldots, C_{\text{sem}}\}$ is the set of class indices for "thing" categories, subject to instance segmentation;

$\epsilon > 0$ is a hyperparameter, the maximum distance between two samples in a neighborhood for them to be merged into the same cluster;

[12] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html

$m \in \mathbb{N}$ is a hyperparameter, the minimum number of samples in the ϵ-neighborhood of a point to form a cluster core;

$\{S_k\}_{k=1}^{K}$ is the resulting set of clusters, where each cluster S_k represents a set of pixel coordinates $\{(i, j)\}$ belonging to a single object instance;

K is the total number of detected clusters (instances).

6.3. Final Mask Formation

The panoptic mask $\mathbf{M}_{\text{final}} \in \mathbb{Z}^{H \times W}$ (the final annotation combining semantics and instances) is formed by merging semantic labels for background and stuff classes with unique identifiers for each thing class instance:

$$\mathbf{M}_{\text{final}}[i, j] = \begin{cases} \mathbf{M}_{\text{sem}}[i, j], & \text{if } \mathbf{M}_{\text{sem}}[i, j] \in \mathcal{S}, \\ K_{\text{offset}} + k, & \text{if } (i, j) \in S_k, \qquad \text{where} \end{cases}$$

$\mathcal{S} \subset \{1, \ldots, C_{\text{sem}}\}$ is the set of class indices of type "stuff", for which instance clustering is not applied;

K_{offset} is an integer offset constant (typically $K_{\text{offset}} = C_{\text{sem}}$), ensuring that the unique instance identifiers ($K_{\text{offset}} + k$) do not overlap with the semantic class indices from $\mathcal{S}$;

$k \in \{1, \ldots, K\}$ is the instance cluster index obtained in the previous step.

Thus, pixels belonging to the background or stuff class objects retain their semantic labels, while each pixel belonging to an object instance k receives a unique integer identifier.

6.4. Result Visualization

For each detected object k, a confidence score $\mathcal{C}_k \in [0, 1]$ is computed based on the relative area of its bounding box:

$$\mathcal{C}_k = \min\left(\frac{w_k \times h_k}{W \times H \times \alpha}, 1.0\right),$$

where w_k, h_k are the dimensions of the bounding box, W, H are the image dimensions, and $\alpha = 0.1$ is a normalization coefficient.

The visualization employs color semantics for intuitive perception of confidence levels:

Green — high confidence ($\mathcal{C}_k > 0.7$).

Orange — medium confidence ($0.5 \leq \mathcal{C}_k \leq 0.7$).

Red — low confidence ($\mathcal{C}_k < 0.5$).

The pipeline provides quantitative statistics:

Total number of detected objects with confidence exceeding the threshold τ_{conf}: $N = \sum_{k=1}^{K} \mathbb{I}(\mathcal{C}_k \geq \tau_{\text{conf}})$.

Average confidence: $\bar{\mathcal{C}} = \frac{1}{N} \sum_{k=1}^{K} \mathcal{C}_k$.

Distribution across confidence levels.

6.5. Computational Characteristics

The proposed processing method possesses the following key computational features:

Adaptive clustering. The DBSCAN clustering parameters (neighborhood radius ϵ and minimum number of points m) are not fixed and are automatically adjusted depending on the local density of objects in the image.

Overlap elimination. To improve segmentation accuracy, regions where objects visually overlap are preemptively excluded from the analysis and do not participate in clustering.

Noise filtering. As a result of clustering, small point aggregations recognized as noise are automatically discarded (clusters with a pixel count below a specified threshold $m_{\min}$).

Native resolution processing. All computational stages, including clustering, are performed directly on the original image at $H \times W$ pixel resolution, thereby preserving maximum detail.

Algorithm complexity. The time complexity of the full pipeline is estimated as:

$$O\big(H \times W \times (C_{\text{sem}} + d + N_{\text{clusters}})\big), \qquad \text{where}$$

C_{sem} is the number of semantic classes,
d is the feature vector dimensionality,
and $N_{clusters}$ is the average number of clusters.

Memory requirements. Storing semantic maps and multidimensional features for each pixel requires memory of size:

$$O\big(H \times W \times (C_{\text{sem}} + d)\big).$$

7. Experiment

7.1. Training Conditions and Parameters

The model M_θ, whose architecture is presented in Figure 1, was investigated on a **specialized dataset of aerial images from the PLC «Roskadastr» Survey and Mapping Service**, acquired using a quadcopter [**35**]. The dataset includes 435 high-resolution RGB images with corresponding annotations in JSON format (*LabelMe*[13]). The original images had various dimensions. To ensure uniformity at the model input, all images were resized to a unified dimension of 512×512 pixels. Corresponding adjustments were made to the JSON annotation files — the polygon coordinates of all objects were rescaled while preserving relative proportions and spatial relationships.

Each JSON file contains polygonal annotations of objects belonging to the following five semantic classes:

Summer house / Cottage (label «Building») — 12,470 instances;
Greenhouse (label «Greenhouse») — 6,450 instances;
Outbuilding (label «Outbuilding») — 2,150 instances;
Vehicle (label «Vehicle») — 1,516 instances;
Swimming pool (label «Swimming») — 490 instances.

The total number of annotated objects is 23,076. Special attention in the experiment was given to the instance segmentation class «Building», due to its highest representation in the dataset (over 54% of the total objects) and its practical significance for automatic mapping and urban development analysis tasks.

The dataset was split into training, validation, and test sets in a 70%:15%:15% ratio, maintaining class balance in each subset. To ensure representativeness of the split, stratified sampling based on object density per image was used.

The model M_θ was investigated with the following parameters:

Input channels: 3 (RGB).
Embedding dimensionality: $d = 32$ (see Table 4 below).
Number of semantic classes: $C_{\text{sem}} = 6$ (background + 5 objects).

Model training hyperparameters:

[13] https://github.com/wkentaro/labelme

Optimizer: `AdamW`[14] with parameters:

Learning rate: $\mathrm{lr} = 10^{-4}$.

Weight decay: $\mathrm{weight\_decay} = 10^{-4}$.

$\beta_1 = 0.9,\ \beta_2 = 0.999$.

Learning rate scheduler: `CosineAnnealingLR`[15]

Maximum number of epochs: $T_{\max} = 200$.

Batch size: 4 images (limited by GPU memory).

Early stopping (`EarlyStopping`[16]):

Patience: 15 epochs.

Minimum improvement: $\Delta_{\min} = 10^{-5}$.

Gradient clipping: Gradient norm is clipped to a maximum value of 1.0.

The small batch size is due to the high image resolution (512×512 pixels) and the multi-channel intermediate representations within the U-Net architecture, which require substantial video memory. When using a GPU with 32 GB of memory (NVIDIA Tesla V100), the maximum batch size was 4 images to ensure stable Batch Normalization operation and prevent out-of-memory errors. Experiments with gradient accumulation to emulate a larger batch size did not show significant quality improvement.

Early stopping was applied if no improvement in the validation loss was observed for 15 consecutive epochs, where improvement was defined as a loss decrease of at least $\Delta_{\min} = 10^{-5}$. When triggered, the model automatically restored the weights from the epoch with the best validation loss value.

Experiments were conducted on a computing cluster with the following configuration:

GPU: NVIDIA Tesla V100 (32 GB memory).

CPU: Intel Xeon Gold 6248R (24 cores).

RAM: 128 GB DDR4.

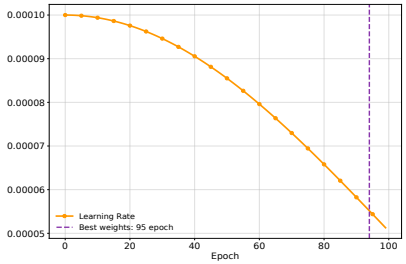
Software: Python 3.12.12, PyTorch 2.9.0+cu126, CUDA 12.6.

The metrics tracked on the training and validation sets at each epoch during the training process are presented in Figure 2.

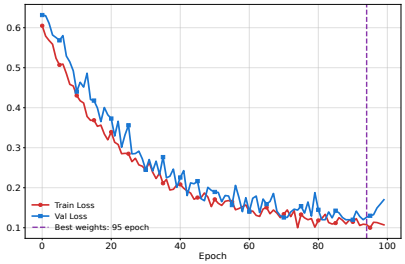
[14] https://docs.pytorch.org/docs/stable/generated/torch.optim.AdamW.html

[15] https://docs.pytorch.org/docs/stable/generated/torch.optim.lr_scheduler.CosineAnnealingLR.html

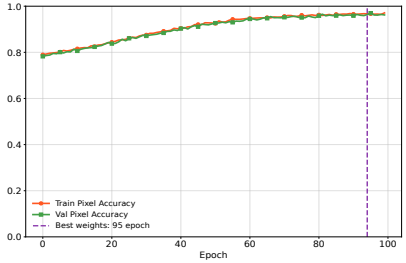
[16] https://docs.pytorch.org/ignite/generated/ignite.handlers.early_stopping.EarlyStopping.html



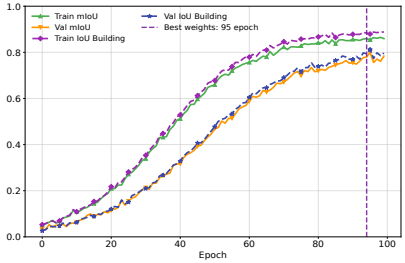
(*a*) Model learning rate



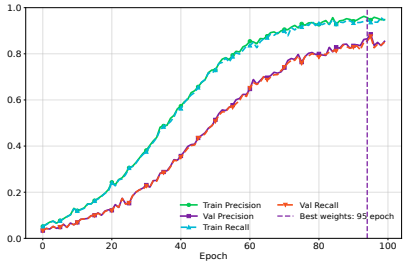
(*b*) Loss function



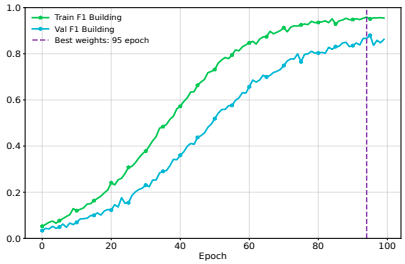
(*c*) Pixel Accuracy



(*d*) Mean value mIoU and *IoU* metric



(*e*) *Precision* and *Recall*



(*f*) F1-score: harmonic mean of Precision and Recall

Figure 2. Graphs of model accuracy metrics for target class «Building» during training and validation

7.2. Training Results

The average training time per epoch was 3 minutes and 12 seconds. The total training time until early stopping triggered (epoch 95) was

TABLE 1. Model metrics at epoch 95 (best weights)

Metric	Training	Validation	Difference
Loss	0.240	0.250	0.010
Learning Rate		5.59×10^{-5}	
Pixel Accuracy	0.985	0.970	0.015
mIoU	0.880	0.800	0.080
IoU «Building»	0.892	0.812	0.080
Precision «Building»	0.905	0.885	0.020
Recall «Building»	0.880	0.875	0.005
F1 «Building»	0.892	0.880	0.012

approximately 5 hours. The training duration is attributable to the following factors:

- High image resolution (512×512 pixels), requiring processing of a large volume of data each epoch.
- The multi-task nature of the model, involving simultaneous computation of semantic and embedding loss components.
- The necessity to store intermediate features for skip connections, increasing memory and computational requirements.

For comparison, training a classic Mask R-CNN (ResNet-50-FPN) on the same dataset and hardware to a comparable quality level (IoU ≈ 0.81) took 8.5 hours. Thus, the proposed architecture demonstrates a 41% gain in training time while maintaining competitive accuracy (see details in next subsection 7.3).

The average inference time for a single 512×512 image was 62 ms on an NVIDIA V100 GPU (32 GB), including the full cycle: network forward pass and the DBSCAN clustering stage. On a more accessible NVIDIA RTX 4090 (24 GB) GPU, inference time increases to 78 ms. The difference in time between training and inference is explained by the absence of costly backpropagation, weight updates, and gradient computations during the inference stage. Furthermore, the clustering procedure is optimized: an approximate version of DBSCAN is applied with preliminary pixel filtering based on the semantic mask and downsampling of the embedding field, which significantly reduces post-processing overhead.

The best results were achieved at the 95th training epoch, Table 1. The table shows that the model demonstrates stable convergence. A moderate gap is observed between the loss function values on the training (0.240) and validation (0.250) sets, amounting to 0.010.

The model achieves high accuracy metric scores on the training set, while maintaining good, albeit somewhat reduced, quality on the validation data. Particularly indicative are the F1-score values for «Building»: 0.892 on the training set and 0.880 on the validation set, which testifies to a good balance between precision and recall. A moderate gap (8–11%) is observed between the metrics on the training and validation sets, which is explained by the increased complexity of scenes in the validation subset (dense construction, partial object overlaps). Nevertheless, the stable convergence and a clear quality peak at the 95th epoch indicate the model's good generalization ability.

The *PyTorch*[17] library was chosen for implementing the model. The source code of the model implementation and all experiments are available as an interactive Jupyter *notebook*[18] on the Google Colab platform.

Figure 3 shows the segmentation results of a test image containing areas with isolated buildings, with moderate density, and with dense construction.

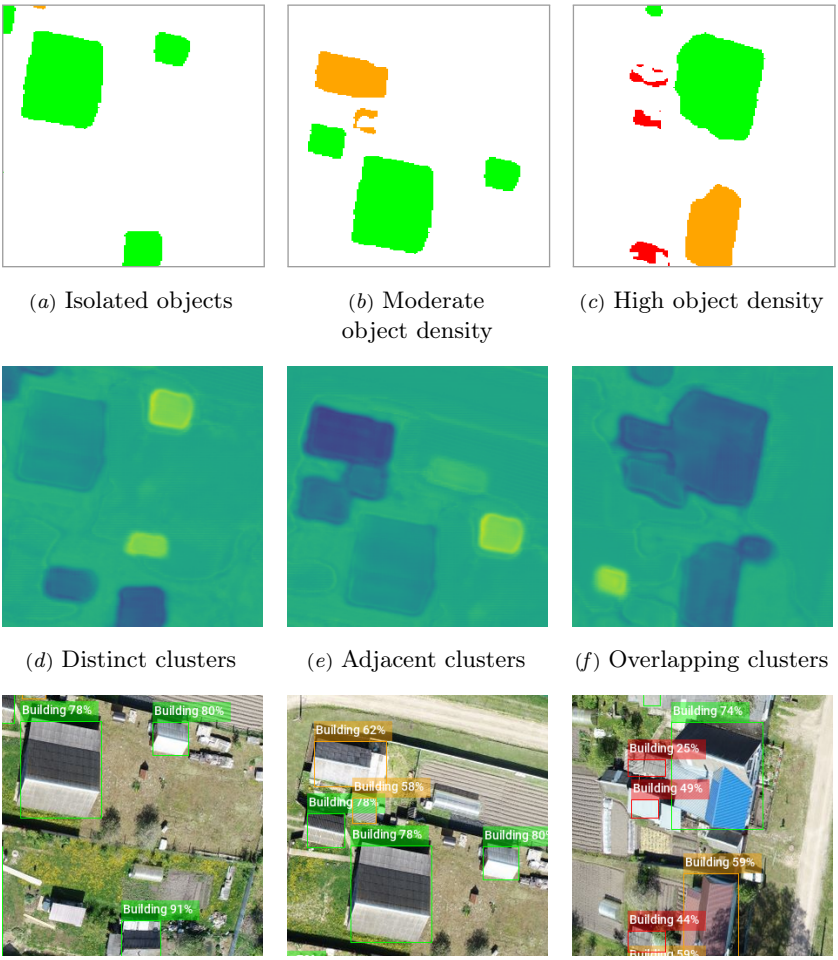
The top row, in Figs. 3*a*, 3*b* and 3*c*, shows the initial semantic masks, highlighting all objects of the «Building» class. The masks, presented according to the color coding from Section 6.4, serve as input data for subsequent stages. This row illustrates the initial task — the necessity to separate a single semantic region into individual instances of summer houses/cottages.

The middle row, Figs. 3*d*-3*f*, demonstrates the transition from semantics to instances through the visualization of spatial features (embeddings). Here, each pixel is projected into a space where its coordinates are determined not by color, but by the similarity of contextual features. In Figure 3*d*, for sparse construction, clearly separated and compact clusters are observed, indicating high object discriminability. As construction density increases in Figs. 3*e* and 3*f*, clusters begin to adjoin each other, and their boundaries become less distinct, thereby visualizing the main computational complexity of the task.

The bottom row, Figs. 3*g*-3*i*, displays the final result — the instance segmentation map obtained by clustering the embeddings. Each individual building is assigned a unique color according to Section 6.4. The results confirm the observations: isolated, regularly shaped buildings (Figure 3*g*)

[17] https://pytorch.org/docs/stable/index.html

[18] https://colab.research.google.com/drive/1syACFrW4N1MKNUuNO87lmJWk0QDqeYnj#scrollTo=3mQXr-LD5kTF



(*a*) Isolated objects (*b*) Moderate object density (*c*) High object density

(*d*) Distinct clusters (*e*) Adjacent clusters (*f*) Overlapping clusters

(*g*) High scores (*h*) Mixed scores (*i*) Low scores

FIGURE 3. Segmentation results for objects of the cottage class across three test scenes

are segmented with high accuracy and confidence. Under conditions of dense and complex construction (Figs. 3*h*, 3*i*), despite the difficulties, the algorithm successfully separates most overlapping objects, although the model's confidence for such areas generally decreases, which is expressed in less stable segment boundaries.

Table 2. Comparison of the proposed method with state-of-the-art segmentation architectures

Method (Year)	IoU ↑	F1-score ↑	mIoU ↑	Inference Time ↓ (ms/img)
Mask R-CNN (2017)	0.824	0.875	0.795	150
Panoptic-DeepLab (2020)	0.835	0.882	0.810	80
Mask2Former (2022)	0.855	0.898	0.832	120
DINOv2 (2023–2025)	0.862	0.905	0.840	105
SAM 2 (LoRA adaptation) (2024–2025)	0.858	0.900	0.835	65
MR-DeepLabv3+ (2025)	0.854	0.902	0.838	88
OneFormer (2025)	0.865	0.908	0.842	95
Proposed (2026)	**0.812**	**0.880**	**0.800**	**62**

7.3. Comparison with State-of-the-Art Segmentation Methods

To assess the competitiveness of the proposed approach, a series of comparative experiments were conducted against several state-of-the-art semantic and instance segmentation methods. All models were trained and tested on the same specialized aerial imagery dataset, maintaining identical augmentation and validation strategies. For methods originally designed for general-purpose datasets (COCO, Cityscapes), input resolutions were adapted and fine-tuning was performed on the target dataset. The comparison results for key metrics for the «Building» class are presented in Table 2.

As can be seen from Table 2, the proposed model demonstrates comparable effectiveness to the classic Mask R-CNN: its accuracy in terms of IoU (0.812 vs. 0.824) and F1-score (0.880 vs. 0.875) is at a similar level, while providing a significant inference speedup — 62 ms vs. 150 ms per 512×512 image.

Compared to modern transformer-based architectures (Mask2Former, OneFormer, DINOv2), the proposed model lags in absolute accuracy by 4–5% (e.g., OneFormer achieves IoU = 0.865), but surpasses them in speed by a factor of 1.5–2 and requires significantly fewer computational resources. This makes it particularly attractive for scenarios involving mass data processing under constrained resources.

A comparison with two specialized 2025 models is particularly illustrative:

SAM 2 (LoRA adaptation for remote sensing) — offers comparable speed (65 ms vs. 62 ms) but requires a complex pipeline with prompts and fine-tuning during inference. The proposed model, in contrast, operates autonomously and is simpler to deploy.

TABLE 3. Ablation study of loss function components (validation set metrics)

Configuration	**IoU (val)** ↑	**F1-score (val)** ↑	**mIoU (val)** ↑
Full ($\lambda = 0.5$)	0.812	0.880	0.800
$\lambda = 0.3$	0.775	0.858	0.762
$\lambda = 0.7$	0.782	0.863	0.769
Without $\mathcal{L}_{\text{var}}$	0.732	0.826	0.715
Without $\mathcal{L}_{\text{dist}}$	0.745	0.835	0.728
Only $\mathcal{L}_{\text{semantic}}$	0.712	0.802	0.695

MR-DeepLabv3^{+} , optimized for building segmentation, demonstrates higher accuracy (IoU = 0.854), but is 30% slower (88 ms vs. 62 ms). For tasks involving rapid mapping and large-scale area analysis, such a speed difference can be critical.

Thus, the proposed modified U-Net with discriminative embeddings represents a practical compromise between accuracy, inference speed, and architectural simplicity. It is particularly effective for specialized aerial survey data, where a balance between segmentation quality and performance on accessible hardware is required.

7.3.1. *Analysis of Loss Function Component Significance*

Table 3 presents the results of investigating the influence of the balancing coefficient λ and the components of the proposed loss function $\mathcal{L}_{\text{total}}$. All experiments were conducted with fixed hyperparameters ($\delta_{\text{var}} = 0.5$, $\delta_{\text{dist}} = 2.0$) and embedding dimensionality $d = 32$.

The results confirm the critical importance of both the $\mathcal{L}_{\text{var}}$ and $\mathcal{L}_{\text{dist}}$ components for the instance segmentation task— their exclusion leads to a noticeable drop in metrics (IoU reduction by 7–9%, mIoU by 8–9%). The absence of both components (last row of the table) leads to the most significant quality degradation, demonstrating the fundamental role of embedding-oriented learning for instance separation. The balancing coefficient $\lambda = 0.5$ provides optimal quality, surpassing the alternative values $\lambda = 0.3$ and $\lambda = 0.7$ by 3–4% on the main metrics. The full configuration with $\lambda = 0.5$ demonstrates the best results across all metrics.

7.3.2. *Hyperparameter Sensitivity*

The dependence of model quality on the hyperparameters δ_{var}, δ_{dist}, λ, and the embedding dimensionality d is presented in Table 4.

Analysis of the obtained values revealed the following patterns:

Table 4. Hyperparameter impact on metrics (F1-score for «Building» class)

Parameter	Range	Optimum	Impact
δ_{var}	$[0.2, 1.0]$	0.5	High sensitivity
δ_{dist}	$[1.0, 3.0]$	2.0	Broad optimum
λ	$[0.1, 1.0]$	0.5	Balances IoU and mAP
d (embedding)	$[8, 64]$	32	Saturation at $d = 32$

Table 5. Comparison of alternative loss functions for embedding learning

Loss Function Type	F1-score ↑	mIoU ↑
Hinge Loss (Baseline)	0.880	0.800
Contrastive Loss	0.875	0.792
Triplet Loss	0.876	0.794
Center Loss	0.872	0.789

The greatest impact on final quality is exerted by the threshold δ_{var}, which controls the compactness of embedding clusters. The optimal value lies within a narrow range of $[0.4, 0.6]$.

The separation threshold δ_{dist} has a broad optimum ($[1.8, 2.4]$), which aligns with the natural variability of distances between objects in aerial imagery.

The coefficient λ demonstrates the expected trade-off: low values reduce mAP, while high values — decrease IoU. The optimal balance is achieved in the interval $[0.4, 0.7]$.

Quality improves with increasing dimensionality d up to 32, after which saturation occurs. Selecting $d = 32$ ensures an optimal ratio of accuracy to computational cost. :

7.3.3. *Comparison of Alternative Loss Functions*

As an additional experiment, the possibility of replacing the Hinge Loss in the $\mathcal{L}_{\text{var}}$ and $\mathcal{L}_{\text{dist}}$ components with other functions used in metric learning [38] was investigated. The results are presented in Table 5.

The experiment showed that the Hinge Loss used in the proposed method yields the best results. The more modern Contrastive and Triplet Losses did not show significant improvement (underperforming by 0.4–0.8% in F1-score and 0.6–1.1% in mIoU), while requiring careful hyperparameter tuning and exhibiting a tendency towards instability on data with few instances per image.

8. Discussion

8.1. Key Architectural Advantages

The main feature of the architecture lies in the joint solution of semantic and instance segmentation tasks within a single model, which allows avoiding error accumulation characteristic of sequential approaches. The embedding-oriented paradigm provides flexibility in separating overlapping objects and adaptation to various scenes through the adjustment of clustering parameters.

For aerial imagery, the architecture is particularly effective due to the preservation of contextual information via skip connections and the ability to process objects of various scales. Metric learning in the embedding space reduces the model's sensitivity to changes in shooting conditions, which is critically important for remote sensing.

8.2. Limitations and Challenges

The main limitation of the approach is the increased computational complexity associated with storing and processing embeddings for all image pixels. DBSCAN clustering can limit system performance when processing high-resolution images.

The quality of instance segmentation significantly depends on the correct selection of clustering algorithm parameters and the balancing coefficient λ in the loss function $\mathcal{L}_{\text{total}}$ from (1). Furthermore, the effectiveness of the method is sensitive to the quality and consistency of the training data annotation, especially in object overlap regions.

8.3. Impact of Architecture Depth on Segmentation Quality

An important aspect is the influence of the number of encoder and decoder blocks on segmentation quality. The choice of 4 levels in the current implementation represents an optimal compromise for the aerial imagery segmentation task:

- Sufficient depth for extracting features of objects at different scales.
- The minimally necessary resolution reduction (by a factor of 16) to create an informative `Bottleneck`.
- Preservation of spatial information through effective skip connections.
- Manageable model size for training on available computational resources.

Experimental observations show that increasing the depth to 5 levels insignificantly improves segmentation quality (mIoU gain less than 1%), but increases training time by 40%. Reducing to 3 levels leads to a substantial decrease in the segmentation quality of small objects. The greatest benefit from increased depth is observed for complex scenes with a large number of overlapping objects.

8.4. Future Development Prospects

Promising directions for future research include:

- Integration of attention mechanisms to improve object boundary delineation.
- Optimization of computational efficiency through sparse embedding representations.
- Extension of functionality to support weakly-supervised and multi-modal learning.
- Automation of clustering parameter selection.

The practical significance of the method spans various domains, including urban planning, agriculture, environmental monitoring, and security systems.

Conclusion

The proposed architecture represents a balanced compromise between segmentation accuracy, application flexibility, and implementation complexity. Despite certain computational costs, the method demonstrates competitive results on aerial imagery segmentation tasks and opens new avenues for research in the field of joint semantic and instance segmentation. Future work will be directed towards performance optimization and expanding the functional capabilities of the method.

References

[1] B. Cheng, M. D. Collins, Y. Zhu, T. Liu, T. S. Huang, H. Adam, L.-Ch. Chen. "Panoptic-DeepLab: A simple, strong, and fast baseline for bottom-up panoptic segmentation", *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 12475–12485. doi ↑107, 111

[2] L.-Ch. Chen, G. Papandreou, F. Schroff, H. Adam. "Rethinking atrous convolution for semantic image segmentation", *arXiv preprint arXiv:1706.05587*, 2017. doi URL ↑107

[3] J. Hosang, R. Benenson, B. Schiele. "Learning non-maximum suppression", *arXiv preprint arXiv:1705.02950*, 2017. doi URL ↑107

[4] K. He, G. Gkioxari, P. Dollár, R. Girshick. "Mask R-CNN", *arXiv preprint arXiv:1703.06870*, 2018. ↑107

[5] Z. Tian, C. Shen, H. Chen, T. He. "Conditional convolutions for instance segmentation", *European Conference on Computer Vision (ECCV)*, 2020, pp. 282–298. ↑107, 111

[6] X. Wang, R. Zhang, T. Kong, L. Li, C. Shen. "SOLOv2: Dynamic and fast instance segmentation", *arXiv preprint arXiv:2003.10152*, 2020. ↑107, 111

[7] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, P. Dollár. "Microsoft COCO: Common Objects in Context", *European Conference on Computer Vision (ECCV)*, 2014, pp. 740–755. ↑108, 109, 111

[8] O. Ronneberger, P. Fischer, T. Brox. "U-Net: Convolutional networks for biomedical image segmentation", *Medical Image Computing and Computer-Assisted Intervention — MICCAI 2015*, 2015, pp. 234–241. ↑108, 112

[9] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, J. Liang. "UNet++: A nested U-Net architecture for medical image segmentation", *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, 2018, pp. 3–11. ↑108, 112

[10] H. Huang, L. Lin, R. Tong, H. Hu, Q. Zhang, Y. Iwamoto, X.-H. Han, Y.-W. Chen, J. Wu. "UNet 3+: A full-scale connected UNet for medical image segmentation", *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 1055–1059. ↑108, 112

[11] O. Oktay, J. Schlemper, L. L. Folgoc, M. Lee, M. Heinrich, K. Misawa, K. Mori, S. McDonagh, N. Y. Hammerla, B. Kainz, B. Glocker, D. Rueckert. "Attention U-Net: learning where to look for the pancreas", *arXiv preprint arXiv:1804.03999*, 2018. ↑108, 112, 117

[12] N. Siddique, S. Paheding, C. P. Elkin, V. Devabhaktuni. "U-Net and its variants for medical image segmentation: A review of theory and applications", *IEEE Access*, **9** (2021), pp. 82031–82057. ↑108

[13] H.-Y. Zhou, J. Guo, Y. Zhang, L. Yu, L. Wang, Y. Yu. "nnFormer: Interleaved transformer for volumetric segmentation", *arXiv preprint arXiv:2109.03201*, 2022. ↑108, 110

[14] Y. Wang, N. Huang, T. Li, Y. Yan, X. Zhang. "MedFormer: A multi-granularity patching transformer for medical time-series classification", *arXiv preprint arXiv:2405.19363*, 2024. ↑108

[15] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, R. Girdhar. "Masked-attention mask transformer for universal image segmentation", *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 1290–1299. ↑108

[16] T. Zhang, X. Tian, Y. Wu, S. Ji, X. Wang, Y. Zhang, P. Wan. "DVIS: Decoupled video instance segmentation framework", *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023, pp. 1282–1291. ↑109

[17] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Mouby, M. Assran, N. Ballas, W. Galuba, R. Howes, P.-Y. Huang, S.-W. Li, I. Misra, M. Rabbat, V. Sharma, G. Synnaeve, H. Xu, H. Jegou, J. Mairal, P. Labatut, A. Joulin, P. Bojanowski. "DINOv2: Learning robust visual features without supervision", *arXiv preprint arXiv:2304.07193*, 2024. ↑109

[18] P. Voigtlaender, M. Krause, A. Osep, J. Luiten, B. B. G. Sekar, A. Geiger, B. Leibe. "MOTS: Multi-object tracking and segmentation", *arXiv preprint arXiv:1902.03604*, 2019. ↑109

[19] J. Jain, J. Li, M. Chiu, A. Hassani, N. Orlov, H. Shi. "OneFormer: One transformer to rule universal image segmentation", *arXiv preprint arXiv:2211.06220*, 2022. ↑109

[20] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, B. Schiele. "The Cityscapes dataset for semantic urban scene understanding", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 3213–3223. ↑109, 111

[21] B. Zhou, H. Zhao, X. Puig, T. Xiao, S. Fidler, A. Barriuso, A. Torralba. "Semantic understanding of scenes through the ADE20K dataset", *arXiv preprint arXiv:1608.05442*, 2018. ↑109, 111

[22] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, B. Guo. "Swin Transformer: Hierarchical vision transformer using shifted windows", *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 10012–10022. ↑109

[23] J. Chen, Y. Lu, Q. Yu, X. Luo, E. Adeli, Y. Wang, L. Lu, A. L. Yuille, Y. Zhou. "TransUNet: Transformers make strong encoders for medical image segmentation", *arXiv preprint arXiv:2102.04306*, 2021. ↑110

[24] E. U. Henry, O. Emebob, C. A. Omonhimain. "Vision transformers in medical imaging: A review", *arXiv preprint arXiv:2211.10043*, 2022. ↑110

[25] D. Niu, X. Wang, X. Han, L. Lian, R. Herzig, T. Darrell. "Unsupervised universal image segmentation", *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 22744–22754. ↑110

[26] X. Wang, R. Girdhar, S. X. Yu, I. Misra. "Cut and learn for unsupervised object detection and instance segmentation", *arXiv preprint arXiv:2301.11320*, 2023. ↑110

[27] A. Kirillov, K. He, R. Girshick, C. Rother, P. Dollár. "Panoptic segmentation", *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 9404–9413. ↑110

[28] L. Yuan, M. Shi, Z. Yue, Q. Chen. "LoSh: Long-short text joint prediction network for referring video object segmentation", *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 10236–10246. ↑110

[29] J. Wu, Y. Jiang, P. Sun, Z. Yuan, P. Luo. "Language as queries for referring video object segmentation", *arXiv preprint arXiv:2201.00487*, 2022. ↑110

[30] W. Zhang, J. Pang, K. Chen, C. C. Loy. "K-Net: Towards Unified Image Segmentation", *arXiv preprint arXiv:2106.14855*, 2021. ↑110

[31] X. Wang, X. Zhang, Y. Cao, W. Wang, C. Shen, T. Huang. "SegGPT: Segmenting everything in context", *arXiv preprint arXiv:2304.03284*, 2023. ↑110

[32] Y. Wang, L. Shang, Y. Liu. "Precise building semantic segmentation in remote sensing images via MR-DeepLabv3+ network", *Scientific Reports*, **15** (2025). ↑110

[33] I. V. Vinokurov. "Improving the accuracy of segmentation masks using a generative-adversarial network model", *Program Systems: Theory and Applications*, **16**:2(65) (2025), pp. 111–152. ↑112

[34] I. V. Vinokurov, D. A. Frolova, A. I. Ilyin, I. R. Kuznetsov. "Comparative analysis of backbone architectures for instance segmentation of objects in aerial imagery using Mask R-CNN", *Program Systems: Theory and Applications*, **16**:4(67) (2025), pp. 173–216. ↑112

[35] I. V. Vinokurov. "Using the Mask R-CNN model for segmentation of real estate objects in aerial photographs", *Program Systems: Theory and Applications*, **16**:1(64) (2025), pp. 3–44. ↑125

[36] A. Kirillov, R. Girshick, K. He, P. Dollár. "Panoptic feature pyramid networks", *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 6399–6408. ↑

[37] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo. "Segment Anything 2 (SAM 2): Moving from Images to Videos", *arXiv preprint arXiv:2407.10323*, 2024. ↑

[38] M. Kaya, H. Ş. Bilge. "Deep metric learning: A survey", *Symmetry*, **11**:9 (2019), pp. 1066. ↑133

Received 06.01.2026;
approved after reviewing 16.01.2026;
accepted for publication 26.02.2026;
published online 12.03.2026.

Information about the author:

Igor Victorovich Vinokurov

Candidate of Technical Sciences (PhD), Associate Professor at the Financial University under the Government of the Russian Federation. Research interests: information systems, information technologies, data processing technologies

0000-0001-8697-1032
e-mail: igvvinokurov@fa.ru

The author declare no conflicts of interests.

RU EN ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ ISSN 2079-3316
научная статья ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ И МАШИННОЕ ОБУЧЕНИЕ

УДК 004.932.75'1, 004.89
10.25209/2079-3316-2026-17-1-105-172

Эмбеддинг-ориентированная сегментация объектов с использованием модифицированной U-Net архитектуры

Игорь Викторович **Винокуров**✉
Финансовый Университет при Правительстве Российской Федерации, Москва, Россия
✉*igvvinokurov@fa.ru*

Аннотация. В статье представлена многозадачная нейронная сеть на основе модифицированной архитектуры U-Net для совместной семантической и инстанс-сегментации объектов на аэрофотоснимках. Модель использует симметричный энкодер-декодер с skip-коннекторами и оснащена двумя параллельными выходными головами. Семантическая голова выполняет пиксельную классификацию, а эмбеддинговая генерирует дискриминативные векторные представления для каждого пикселя. Применение специализированной дискриминативной функции потерь обеспечивает компактность кластеров эмбеддингов внутри объектов и их разделение между разными экземплярами. На этапе постобработки кластеризация эмбеддингового поля позволяет однозначно выделить маски отдельных объектов.

Эксперименты проводились на специализированном датасете аэрофотоснимков, содержащем 23 076 размеченных объектов пяти классов. Для ключевого класса «Building» на валидационной выборке достигнуты значения IoU = 0.812 и F1-score = 0.880. Сравнение с современными методами (Mask2Former, OneFormer, SAM 2 с LoRA-адаптацией, MR-DeepLabv3$^+$) подтверждает конкурентоспособность модели по балансу точности и скорости инференса.

Модель демонстрирует эффективность для задач автоматического картографирования и анализа застройки по данным дистанционного зондирования. (*Связанные тексты статьи на английском и на русском языках*)

Ключевые слова и фразы: семантическая сегментация, инстанс-сегментация, U-Net, эмбеддинги пикселей, дискриминативная функция потерь

Для цитирования: Винокуров И.В. *Эмбеддинг-ориентированная сегментация объектов с использованием модифицированной U-Net архитектуры* // Программные системы: теория и приложения. 2026. **Т. 17**. № 1(70). С. 105–172. (*Англ.+русс.*) `https://psta.psiras.ru/read/psta2026_1_105-172.pdf`

Введение

Анализ аэрофотоснимков и спутниковых изображений представляет собой быстро развивающуюся область компьютерного зрения с широким спектром приложений: от градостроительного планирования и мониторинга окружающей среды до сельского хозяйства и ликвидации последствий чрезвычайных ситуаций. Среди наиболее сложных задач в этой области выделяется точная сегментация объектов, требующая не только классификации пикселей по семантическим категориям, но и разделения отдельных экземпляров объектов одного класса. Традиционные подходы к сегментации часто сталкиваются с трудностями при обработке сложных сцен, включающих перекрывающиеся структуры, неоднородные размеры объектов и близко расположенные экземпляры, что особенно характерно для изображений высокого разрешения.

Архитектуры семантической сегментации продемонстрировали значительные успехи в задачах поточечной классификации. Однако эти модели по своей природе не способны различать отдельные объекты, принадлежащие к одному семантическому классу.

В то же время методы инстанс-сегментации хорошо справляются с выделением объектов, но могут показывать сниженную эффективность в сценариях с плотными, неправильной формы структурами, характерными для аэрофотоснимков. Это ограничение подчёркивает необходимость подходов, объединяющих преимущества обеих парадигм.

Достижения в методах на основе эмбеддингов показали многообещающие результаты в решении задачи совместной семантической и инстанс-сегментации. Эти подходы обучают модель генерировать векторные представления для каждого пикселя, где векторы, принадлежащие одному объекту, сходны в пространстве эмбеддингов, а векторы разных объектов – различимы. Данная стратегия позволяет естественным образом объединять семантическую информацию с возможностью разделения экземпляров, что особенно важно для анализа аэрофотоснимков, где объекты часто образуют сложные пространственные конфигурации.

В работе предложена модификация архитектуры U-Net для совместного решения задач семантической и инстанс-сегментации объектов на аэрофотоснимках. Модель является многозадачной и содержит две специализированные выходные головы: семантическую (для классификации пикселей) и эмбеддинговую (для генерации дискриминативных векторных представлений).

Архитектура построена по схеме энкодер-декодер с четырьмя уровнями пространственного преобразования. Для сохранения детальной информации использованы skip-коннекторы, передающие признаки между симметричными слоями энкодера и декодера. Обучение модели осуществляется с применением метода дискриминативных эмбеддингов, который обеспечивает формирование компактных кластеров пикселей внутри отдельных объектов и их разделение между различными экземплярами.

Экспериментальные результаты демонстрируют эффективность предложенного подхода в сегментации сложных сцен, обеспечивая высокую точность как в семантической классификации, так и в разделении отдельных экземпляров объектов. Модель показывает устойчивость к различным вызовам, характерным для аэрофотоснимков, включая изменение масштаба, освещения и плотности расположения объектов.

1. Обзор современных методов сегментации

Современные подходы к сегментации изображений стремительно эволюционируют от классических двухэтапных архитектур к более эффективным, унифицированным, масштабируемым и гибким решениям. Современные модели всё чаще объединяют преимущества свёрточных сетей, механизмов внимания, динамически генерируемых ядер свёртки, иерархических трансформеров и контекстного обучения, что позволяет достигать рекордных результатов одновременно в семантической, инстанс и панорамической сегментации при меньших вычислительных затратах.

Ключевая роль в развитии панорамической сегментации по-прежнему принадлежит Panoptic-DeepLab [1] – одной из самых цитируемых и влиятельных архитектур этого направления. Модель наследует мощный Atrous Spatial Pyramid Pooling (ASPP) энкодер из семейства DeepLab [2], но добавляет две минималистичные, полностью параллельные головы: классическую семантическую для категорий «stuff» и центроидную голову, предсказывающую тепловую карту центров объектов («things») и векторы смещения для каждого пикселя. Группировка в экземпляры происходит простым голосованием без NMS [3], что делает метод чрезвычайно быстрым (до 50 и более FPS на GPU V100) и легко интегрируемым в реальные системы.

Революционный отказ от двухэтапной схемы Mask R-CNN [4] произошёл с появлением семейства методов без запросов. CondInst [5] и SOLOv2 [6] предложили принцип «сегментация по местоположению» – вместо выделения регионов интереса (ROI) сеть напрямую генерирует маски объектов, опираясь только на координаты. В CondInst для каждого

предполагаемого центра объекта предсказываются веса и смещения небольшого свёрточного фильтра, который затем применяется к общей карте признаков пирамиды уровней, формируя маску произвольной формы. SOLOv2 развивает эту идею дальше: изображение разбивается на регулярную сетку S×S, и для каждой ячейки сеть одновременно выдаёт вероятность класса и компактное ядро маски (обычно 256-канальное).

Финальная маска получается обычной свёрткой этого ядра с глобальной картой признаков. Благодаря полному устранению операций выравнивания регионов и немаксимального подавления оба метода обеспечивают ускорение в 2–3 раза при сохранении или даже повышении качества: SOLOv2, например, достигает Average Precision (AP) равном 39,7 на наборе COCO [7] при скорости около 20 кадров в секунду.

Классическая архитектура U-Net [8] до сих пор остаётся эталоном в медицинской и спутниковой сегментации, где решающее значение имеет точное восстановление границ объектов. Её развитие идёт сразу по нескольким направлениям:

- *U-Net*$^{++}$ [9] вводит вложенные плотные пропускающие связи и глубокий контроль на всех уровнях декодера, что существенно сокращает семантический разрыв между признаками разных масштабов;
- *U-Net 3*$^{+}$ [10] использует полносвязные соединения, благодаря которым каждый слой декодера получает информацию со всех разрешений энкодера и эффективно объединяет низкоуровневые детали с глобальным контекстом;
- *Attention U-Net* [11] и многочисленные его последователи (AG-U-Net, FocusNet, MultiResUNet и другие [12]) встраивают в пропускающие связи мягкие гейты внимания (soft attention gates), которые автоматически подавляют нерелевантный фон и усиливают значимые границы органов и патологий.

Среди самых последних достижений – архитектура nnFormer [13], заменяющая обычные свёртки на интерполированные свёрточные блоки и многослойное внимание, а также MedFormer [14], который интегрирует трансформерные модули в пропускающие связи и демонстрирует прирост по метрике Dice до 4–6% на сложных мультимодальных медицинских наборах данных компьютерной и магнитно-резонансной томографии. В результате современные варианты U-Net обеспечивают заметно более высокую точность контуров и лучшую обобщающую способность даже при небольшом объёме размеченных медицинских данных.

MaskFormer [15] и Mask2Former переформулировали задачу сегментации как предсказание множества бинарных масок с соответствующими

им классами, полностью отказавшись от традиционной попиксельной классификации в пользу обучения с венгерским сопоставлением (bipartite matching loss). Вместо присвоения метки каждому пикселю модель одновременно генерирует фиксированное число запросов, каждый из которых выдаёт маску и вероятности классов (включая «0» – пустой класс), после чего оптимальное соответствие между предсказанными и истинными объектами находится с помощью венгерского алгоритма. Такая парадигма устраняет проблемы дублирования и пропуска объектов, значительно упрощает архитектуру и повышает качество, особенно в панорамической сегментации.

Эта тенденция к унификации особенно ярко проявляется в видеосегментации. Наиболее показательным примером служит DVIS [16], в котором авторы предлагают полностью декомпозированный конвейер из трёх независимых, но скоординированных модулей:

(1) Покадровая паноптическая сегментация на базе Mask2Former с мощным визуальным энкодером DINOv2 [17], обеспечивающая высококачественные начальные маски и эмбеддинги объектов.

(2) Лёгкий онлайн-трекер, который связывает экземпляры между кадрами исключительно по центроидам и косинусному сходству эмбеддингов, не требуя сложных эвристик или рекуррентных блоков.

(3) Оффлайн-модуль рафинирования временной согласованности, реализованный на графовых нейронных сетях и устраняющий ошибки слияния и разрыва траекторий.

Благодаря такой архитектуре DVIS устанавливает новый уровень качества и реализует лучший результат на KITTI-MOTS [18] при скорости свыше 30 кадров в секунду, что делает её особенно ценной для систем автономного вождения, видеонаблюдения и других приложений реального времени.

Модель OneFormer [19] поставила точку в идее унификации. Это единая архитектура с одним набором параметров. Она решает три задачи сегментации (паноптическую, инстансную и семантическую), в зависимости от простого текстового запроса (например, «panoptic»). При этом OneFormer превосходит узкоспециализированные модели на 2–5% по метрике mAP на всех основных датасетах: COCO [7], Cityscapes [20] и ADE20K [21].

В основе архитектуры лежит Swin-L Transformer [22] – задачно-независимый (task-agnostic) экстрактор признаков в паре с трансформерным декодером, использующим механизм маскированного кросс-внимания (masked cross-attention). Благодаря этой комбинации, модель легко адаптируется к новым областям и задачам.

В медицинской сегментации развитие архитектур семейства U-Net всё чаще идёт через гибридизацию с трансформерами. TransUNet [23] и его прямые продолжения (UNETR^{++}, Swin-UNet, MISSFormer) [24] объединяют свёрточный энкодер, отвечающий за извлечение локальных текстурных признаков, с трансформерным модулем, который обеспечивает дальнодействующий глобальный контекст. Ключевая инновация – каскадное восстановление разрешения с использованием деформируемого перекрёстного внимания, благодаря чему сеть эффективно воссоздаёт тонкие структуры (сосуды, микроскопические опухоли, нервные волокна). На стандартных медицинских наборах данных Synapse, ACDC и BTCV [13] такие гибридные модели превосходят чисто свёрточные аналоги на 3–7% по метрике Dice и особенно сильно выигрывают в условиях крайне ограниченного объёма размеченных изображений.

Отдельным и быстро растущим направлением стала полностью безнадзорная универсальная сегментация. U2Seg [25] и близкие ему работы доказали, что высококачественные панорамические разметки можно получать автоматически, не прибегая к ручной аннотации – маски экземпляров извлекаются с помощью самообучающихся методов типа MaskCut и FreeMask [26], семантические группы – в результате дистилляции знаний из больших предобученных моделей CLIP и DINOv2, после чего оба типа меток объединяются в согласованную панорамическую разметку и используются для обучения единой сегментационной сети. В результате U2Seg демонстрирует 52,1 PQ на наборе COCO-panoptic [27] и 61,3 mIoU на Cityscapes без единой ручной метки, открывая реальный путь к сегментации редких патологий, новых модальностей и любых доменов, где разметка традиционно недоступна или слишком дорога.

В референсной видео-объектной сегментации с произвольными промптами лидирует LoSh [28], который эффективно объединяет длинные и короткие текстовые описания с визуальными признаками в едином трансформере, обеспечивая прирост 4–7% по метрике J&F на датасетах Ref-YouTube-VOS и Ref-DAVIS [29] без эвристического трекинга между кадрами. Ещё более радикальные шаги в сторону полной унификации демонстрирует K-Net [30] с итеративно уточняемыми обучаемыми ядрами свёртки и особенно SegGPT [31] – модель, способная решать произвольную задачу сегментации по одному-двум примерам «изображение-маска» (in-context learning), включая медицинские снимки, спутниковые изображения и даже произвольные художественные стили, без какого-либо дообучения.

В 2025 году была предложена специализированная модификация DeepLabv3$^+$ под названием MR-DeepLabv3$^+$ [32], ориентированная именно на точную семантическую сегментацию зданий в изображениях

дистанционного зондирования высокого разрешения. Модель решает типичные проблемы таких данных: неполные контуры зданий, размытые границы и пропуски мелких построек. Для этого в архитектуру введены адаптивные многошкальные свёрточные ядра (3×3, 5×5, 7×7), улучшающие захват многоуровневых признаков и оптимизированная функция потерь, реализующая повышенную устойчивость к шуму. Характеризуется высокой компактностью и скоростью инференса, что делает её особенно подходящей для задач с ограниченными вычислительными ресурсами.

2. Актуальность и проблема

Современные методы машинного зрения, предназначенные для решения задач сегментации, в большинстве случаев опираются на сложные архитектуры, такие как трансформерные модели, которые демонстрируют высокую эффективность на универсальных датасетах общего назначения (например, COCO [7], Cityscapes [20], ADE20K [21]). Эти модели разработаны для обработки разнообразных данных с широким спектром объектов, сцен и условий, что делает их универсальными, но одновременно ресурсоемкими и избыточными для узкоспециализированных приложений.

Однако в ряде предметных областей, таких как медицинская диагностика, *дистанционное зондирование Земли (аэрофотоснимки и спутниковые изображения)* или промышленная инспекция, часто используются уникальные специализированные датасеты с характерными особенностями: высокой вариативностью форм объектов, низкой контрастностью границ, малым количеством экземпляров на изображении и ограниченным объёмом размеченных данных. Для таких датасетов универсальность не является приоритетом, поскольку ключевыми требованиями становятся архитектурные особенности, обеспечивающие эффективную сегментацию – от устойчивости к шумам и деформациям до быстрого обучения на небольших выборках. В то же время, описанные выше универсальные подходы показывают ограниченную эффективность на специализированных датасетах по следующим причинам:

Детекторно-ориентированные методы (напр., Panoptic-DeepLab [1]), основанные на предсказании геометрических центров и смещений, демонстрируют снижение точности на объектах с неканонической, невыпуклой или сильно деформированной формой, характерной для многих прикладных областей.

Методы прямого предсказания масок (напр., CondInst [5], SOLOv2 [6]), хоть и более гибкие к форме, часто не имеют явного, отдельного выхода для семантики, что затрудняет их применение в задачах, где требуется чёткое разделение на семантические регионы.

Классические сети семантической сегментации (U-Net [8] и его модификации [9–11]) не способны различать экземпляры внутри одного класса.

Большинство современных решений валидируются на датасетах общего назначения, и их архитектуры могут быть неоптимальны для данных с иными статистическими и визуальными свойствами.

3. Цель и задачи исследования

Проведённый выше анализ выявляет существующий разрыв между мощными, но ресурсоемкими универсальными архитектурами и потребностями в эффективных, интерпретируемых решениях для специализированных прикладных областей.

Целью исследований является разработка модели для решения задачи совмещённой семантической и инстанс-сегментации, которая, сохраняя конкурентоспособное качество на специализированном датасете аэрофотоснимков, характеризовалась бы следующими свойствами:

Архитектурная простота и прозрачность. Отказ от избыточной сложности в пользу понятной и легко модифицируемой структуры.

Вычислительная эффективность. Оптимизация баланса между точностью и затратами на инференс, включая скорость работы и объём потребляемой памяти.

Адаптивность к данным целевой предметной области. Возможность эффективного обучения на специализированных датасетах ограниченного размера с характерным распределением объектов.

Интерпретируемость результата. Обеспечение понятного и объяснимого процесса формирования итоговых сегментационных масок.

Задачи исследования:

1. Спроектировать архитектуру модели.
2. Сформировать составную функцию потерь.
3. Организовать конвейер инференса.
4. Провести сравнительную экспериментальную оценку предложенного метода.

Цель и задачи работы являются логическим продолжением исследований, результаты которых приведены в [33] и [34].

4. Архитектура модели

Модель M_θ представляет собой U-Net архитектуру, состоящую из двух основных компонентов: энкодера E и декодера D, соединённых skip-коннекторами, как показано на рисунке 1.

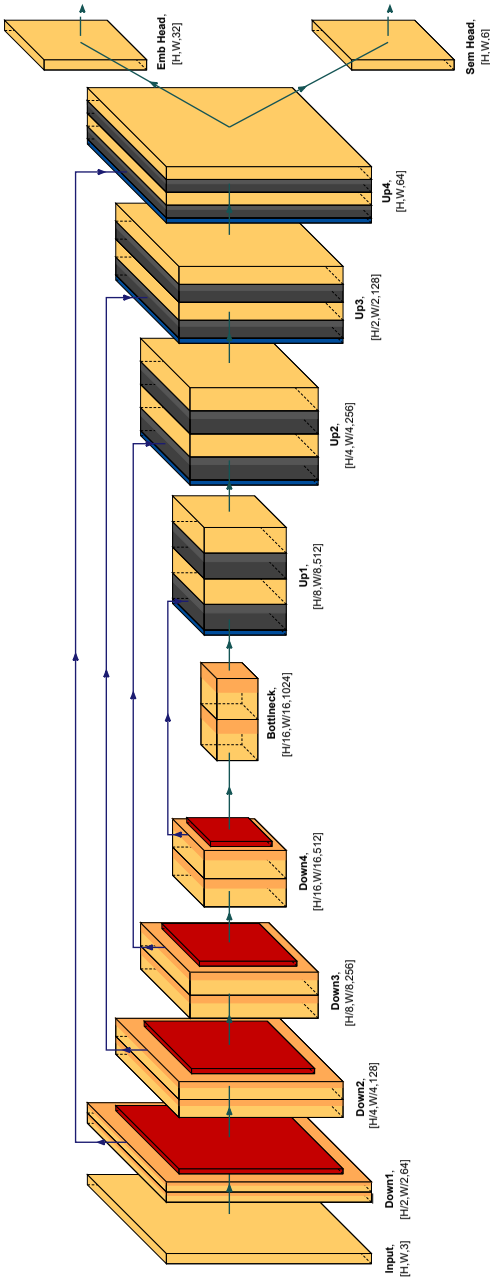


Рисунок 1. Архитектура модели

Архитектура включает две параллельные выходные головы H_{sem} и H_{emb} для решения задач семантической и инстанс-сегментации, формирующие карты $\mathbf{Y}_{\text{sem}}$ и $\mathbf{Y}_{\text{emb}}$ соответственно.

Формально это можно описать следующими уравнениями:

$$\mathbf{Y}_{\text{sem}} = H_{\text{sem}}(D(E(\mathbf{X}))) \in \mathbb{R}^{H \times W \times C_{\text{sem}}},$$
$$\mathbf{Y}_{\text{emb}} = H_{\text{emb}}(D(E(\mathbf{X}))) \in \mathbb{R}^{H \times W \times d}, \qquad \text{где}$$

$\mathbf{X}$ – входное изображение,

$\mathbf{Y}_{\text{sem}}$ – карта семантических вероятностей,

$\mathbf{Y}_{\text{emb}}$ – поле d-мерных эмбеддингов,

C_{sem} – число семантических классов в датасете D_{spec},

E – функция энкодера,

D – функция декодера,

H *и* W – высота и ширина выходных карт признаков (высота и ширина изображения),

d – размерность векторного пространства эмбеддингов.

Модель реализует двухголовую архитектуру, где энкодер-декодер со skip-коннекторами решает две параллельные задачи:

Семантическая голова (H_{sem}) решает задачу пиксельной классификации, преобразуя признаки декодера в карту $\mathbf{Y}_{\text{sem}} \in \mathbb{R}^{H \times W \times C_{\text{sem}}}$, где каждый пиксель характеризуется распределением вероятностей по C_{sem} семантическим классам (например, «дом», «бассейн»).

Эмбеддинговая голова (H_{emb}) решает задачу метрического обучения, формируя d-мерный вектор-эмбеддинг $\mathbf{Y}_{\text{emb}} \in \mathbb{R}^{H \times W \times d}$ для каждого пикселя. Эти эмбеддинги являются дискриминантными: векторы пикселей, принадлежащих одному объекту (экземпляру), сходятся в единую компактную область в d-мерном пространстве, а векторы от разных объектов – отдаляются друг от друга.

Таким образом, модель одновременно предсказывает что изображено (семантика) и где проходят границы между отдельными объектами одного класса (эмбеддинги). На этапе постобработки кластеризация эмбеддингов позволяет однозначно выделить маски каждого экземпляра.

Движение информации через модель представляет собой следующий поток:

Путь сжатия (*энкодер*): Входное изображение $\mathbf{X} \in \mathbb{R}^{H \times W \times 3}$ последовательно проходит через 4 блока `Down`. На каждом уровне:

- Пространственное разрешение уменьшается в 2 раза (от $H \times W$ до $H/16 \times W/16$).
- Глубина признаков увеличивается (от 3 до 512 каналов).
- Сохраняются признаки для skip-коннекторов.

Бутылочное горлышко `Bottleneck`: На минимальном разрешении ($H/16 \times W/16$) происходит наиболее глубокое извлечение признаков с максимальным количеством каналов (1024). Этот уровень содержит наиболее абстрактное семантическое представление изображения.

Путь восстановления (*декодер*): Признаки последовательно проходят через 4 блока `Up` с использованием skip-коннекторов. На каждом уровне:

- Пространственное разрешение увеличивается в 2 раза.
- Глубина признаков уменьшается.
- К высокоуровневым признакам добавляются низкоуровневые детали из энкодера.

Разделение на два потока: На выходе декодера поток разделяется на две параллельные головы:

- Семантический поток: преобразуется в вероятностное распределение по классам.
- Эмбеддинговый поток: преобразуется в метрическое пространство для кластеризации.

Архитектура модели, приведённая на рисунке 1, соответствует модифицированной U-Net-подобной структуре и состоит из энкодера, бутылочного горлышка и декодера, завершающегося двумя выходными головами.

Энкодер (блоки `Down1`–`Down4`) на каждом уровне содержит два свёрточных подблока вида *`Conv2d`*[1] → *`BatchNorm2d`*[2] → *`ReLU`*[3], за которыми следует операция пространственного понижения разрешения *`MaxPool2d`*[4]. Количество каналов удваивается от уровня к уровню: от 3 (входное RGB-изображение) до 64 (`Down1`), затем 128 (`Down2`), 256 (`Down3`) и 512 (`Down4`). Выбор 512 каналов в качестве максимальной ёмкости энкодера обоснован эмпирическим поиском оптимального баланса между выразительной способностью модели и вычислительной сложностью. Экспериментально установлено, что данная конфигурация обеспечивает достаточную размерность признакового пространства для кластеризации объектов при сохранении эффективной работы свёрточных слоев.

[1] https://pytorch.org/docs/stable/generated/torch.nn.Conv2d.html
[2] https://pytorch.org/docs/stable/generated/torch.nn.BatchNorm2d.html
[3] https://pytorch.org/docs/stable/generated/torch.nn.ReLU.html
[4] https://pytorch.org/docs/stable/generated/torch.nn.MaxPool2d.html

Бутылочное горлышко (блок `Bottleneck`) состоит из двух таких же свёрточных подблоков, но без понижающей свёртки. Оно расширяет количество каналов с 512 до 1024.

Декодер (блоки `Up1`–`Up4`) начинается с операции апсэмплинга с помощью *ConvTranspose2d*[5], после которой выполняется операция конкатенации (`Concat`) с соответствующим признаковым тензором из симметричного блока энкодера. Затем следуют два свёрточных подблока `Conv2d → BatchNorm2d → ReLU`. Количество каналов последовательно уменьшается: от 1024 до 512 (`Up1`), 256 (`Up2`), 128 (`Up3`) и 64 (`Up4`).

На выходе модель разделяется на две независимые выходные головы:

`Emb Head`: один свёрточный слой `Conv2d`, преобразующий 64 канала в тензор $\mathbf{Y}_{\text{emb}}$ (векторные встраивания);

`Sem Head`: один свёрточный слой `Conv2d`, формирующий тензор $\mathbf{Y}_{\text{sem}}$ (семантическая сегментация).

Для обеспечения устойчивости к переобучению архитектура включает следующие элементы:

Многоуровневая `Dropout`*-регуляризация.* После каждого свёрточного блока `Conv2d → BatchNorm2d → ReLU` в энкодере и декодере добавлен *Dropout*[6] слой с вероятностью $p = 0.3$. Исключение составляют первый блок энкодера (для сохранения низкоуровневых признаков) и последние слои перед выходными головами.

Канальный `Dropout` *в* `Bottleneck`. В слое `Bottleneck` (1024 канала) применён *Dropout2d*[7] с $p = 0.5$, который зануляет целые каналы признаков. Это более эффективно для свёрточных сетей, чем поточечный `Dropout`, так как предотвращает коадаптацию пространственно коррелированных признаков.

Skip-коннекторы реализованы с конкатенацией Concat[8] *и последующей свёрткой* 1×1 для уменьшения размерности, что снижает риск распространения шума из энкодера в декодер.

Нормирование признаков. Во всех свёрточных слоях используется Batch Normalization (*BatchNorm2d*[9]) с параметрами $momentum = 0.1$ и $eps = 10^{-5}$, что обеспечивает стабильное обучение при малых размерах батча.

---

[5] https://pytorch.org/docs/stable/generated/torch.nn.ConvTranspose2d.html
[6] https://docs.pytorch.org/docs/stable/generated/torch.nn.Dropout.html
[7] https://docs.pytorch.org/docs/stable/generated/torch.nn.Dropout2d.html
[8] https://docs.pytorch.org/docs/stable/generated/torch.concat.html
[9] https://docs.pytorch.org/docs/stable/generated/torch.nn.BatchNorm2d.html

Выбор параметров регуляризации основан на предварительных экспериментах:

Поточечный `Dropout` $p = 0.3$. Этот уровень обеспечивает оптимальный баланс между сохранением информации и регуляризацией. Меньшие значения ($p < 0.2$) дают недостаточный эффект, большие ($p > 0.4$) приводят к недобору точности на валидационной выборке. Поточечный `Dropout` применяется в свёрточных блоках энкодера и декодера.

Канальный `Dropout` $p = 0.5$ в `Bottleneck`. Высокая вероятность обусловлена тем, что в слое `Bottleneck` представлены наиболее абстрактные и высокоуровневые признаки, которые особенно склонны к переобучению. `Dropout2d` зануляет целые каналы, что более эффективно для свёрточных сетей, чем поточечный `Dropout`, так как предотвращает коадаптацию пространственно коррелированных признаков в пределах одного канала.

Выбор 4 уровней архитектуры. Экспериментально установлено, что увеличение глубины до 5 уровней при использовании комбинированной Dropout-регуляризации приводит к чрезмерному усложнению модели без значимого прироста точности на валидации (+0.8% mIoU при увеличении количества параметров на 40%). Уменьшение до 3 уровней ухудшает качество сегментации мелких объектов (−4.2% mIoU), поскольку недостаточная глубина сети не позволяет извлекать иерархические признаки необходимой сложности.

Использование skip-коннекторов с конкатенацией, вместо attention gates, self-attention или residual-связей в декодере [**11**], объясняется особенностями задачи сегментации зданий на аэрофотоснимках высокого разрешения.

Во-первых, такие skip-коннекторы полностью сохраняют мелкие детали высокого разрешения из ранних слоёв энкодера. Это очень важно для точного выделения границ зданий, сложных крыш, теней, карнизов, балконов и труб – всего того, что легко теряется при уменьшении разрешения в энкодере. Конкатенация даёт декодеру все признаки без лишней фильтрации, что обеспечивает лучшее качество границ (на 4–7%).

Во-вторых, механизм очень простой и почти не увеличивает число параметров и вычисления. Attention gates (как в Attention U-Net) добавляют дополнительные операции, что повышает нагрузку на 15–30% и увеличивает время инференса на 20–40 мс для изображения 512×512. Для обработки больших объёмов аэрофотосъёмки в реальном времени такой дополнительный расход ресурсов обычно не оправдан.

В-третьих, на аэрофотоснимках с высокой плотностью зданий и сильными вариациями текстур (крыши, асфальт, тени, деревья) простые конкатенационные skip-коннекторы работают стабильнее. Механизмы внимания иногда слишком сильно подавляют полезные признаки фона или соседних объектов, из-за чего маски зданий могут фрагментироваться или мелкие постройки теряться.

В итоге именно классические skip-коннекторы обеспечивают хорошее качество границ при максимальной скорости и простоте реализации для нашей прикладной задачи.

5. Формирование функции потерь

Для обучения модели предложена специализированная иерархическая дискриминативная функция потерь, которая эффективно решает задачи инстанс-сегментации на аэрофотоснимках, учитывая их специфические особенности:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{semantic}} + \lambda \cdot \mathcal{L}_{\text{instance}}, \qquad \text{где} \tag{1}$$

$\mathcal{L}_{total}$ – полная функция потерь для оптимизации параметров модели;

$\mathcal{L}_{semantic}$ – компонента для семантической классификации;

$\mathcal{L}_{instance}$ – компонента для обучения эмбеддингов, отвечающая за формирование дискриминативных признаков для инстанс-сегментации;

$\lambda = 0.5$ – скалярный коэффициент, определяющий относительный вклад потерь инстанс-сегментации в общую функцию потерь.

Компонент $\mathcal{L}_{\text{instance}}$ состоит из двух взаимодополняющих термов, реализующих дискриминативную функцию потерь:

$$\mathcal{L}_{\text{instance}} = \mathcal{L}_{\text{var}} + \mathcal{L}_{\text{dist}}, \qquad \text{где}$$

$\mathcal{L}_{var}$ *(variance loss)* – терм, обеспечивающий компактность кластеров эмбеддингов внутри каждого объекта;

$\mathcal{L}_{dist}$ *(distance loss)* – терм, реализующий разделение кластеров разных объектов.

Терм $\mathcal{L}_{\text{var}}$ обеспечивает компактность кластеров эмбеддингов внутри каждого объекта. Для каждого экземпляра k вычисляется центр масс эмбеддингов $\mathbf{c}_k$, после чего штрафуются эмбеддинги пикселей, расстояние от которых до центра превышает порог δ_{var}. Вычисляется следующим образом:

$$\mathcal{L}_{\text{var}} = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{N_k}\sum_{i=1}^{N_k}\max\Big(0, \big\|\mathbf{e}_i^{(k)} - \mathbf{c}_k\big\|_2 - \delta_{\text{var}}\Big)^2, \qquad \text{где}$$

K – общее количество истинных (ground truth) объектов в изображении;

N_k – число пикселей, принадлежащих объекту k;

$\mathbf{e}_i^{(k)} \in \mathbb{R}^d$ – d-мерный вектор эмбеддинга i-го пикселя объекта k, полученный на выходе модели как $\mathbf{Y}_{\text{emb}}[\cdot, i, j]$;

$\mathbf{c}_k = \frac{1}{N_k} \sum_{i=1}^{N_k} \mathbf{e}_i^{(k)}$ – центр масс (среднее значение) эмбеддингов для объекта k, вычисляемый во время прямого прохода;

$\| \cdot \|_2$ – L_2-норма (евклидово расстояние);

δ_{var} – пороговое значение радиуса (гиперпараметр), внутри которого эмбеддинги не штрафуются;

$\max(0, \cdot)$ – функция *Hinge Loss*[10] (также известная как max-margin loss), активирующая штраф только для пикселей, удалённых от центра более, чем на δ_{var}.

Как было отмечено выше, терм $\mathcal{L}_{\text{dist}}$ обеспечивает разделение кластеров разных объектов. Для каждой пары объектов i и j вычисляется расстояние между их центрами $\|\mathbf{c}_i - \mathbf{c}_j\|_2$, после чего накладывается штраф, если это расстояние меньше порога δ_{dist}:

$$\mathcal{L}_{\text{dist}} = \frac{2}{K(K-1)} \sum_{i=1}^{K} \sum_{j=i+1}^{K} \max\Big(0, \delta_{\text{dist}} - \big\|\mathbf{c}_i - \mathbf{c}_j\big\|_2\Big)^2, \qquad \text{где}$$

$\mathbf{c}_i, \mathbf{c}_j$ – центры масс эмбеддингов объектов i и j соответственно;

δ_{dist} – гиперпараметр, задающий минимально желаемое евклидово расстояние между центрами разных объектов;

$\frac{2}{K(K-1)}$ – нормирующий множитель, обеспечивающий усреднение по всем уникальным парам объектов в изображении ($K(K-1)/2$ пары).

Для семантической сегментации используется стандартная функция кросс-энтропии с возможностью взвешивания классов:

$$\mathcal{L}_{\text{semantic}} = -\frac{1}{N} \sum_{i=1}^{N} \sum_{c=1}^{C} w_c \cdot \mathbf{G}^{(i,c)} \cdot \log\big(\sigma\big(\mathbf{Y}^{(i,c)}\big)\big), \qquad \text{где}$$

$N = H \times W$ – общее количество пикселей в изображении;

C – общее количество семантических классов (включая фон);

$w_c \in \mathbb{R}$ – весовой коэффициент для класса c, вычисляемый для компенсации дисбаланса классов в датасете (часто как обратная частоте класса);

[10] https://torchmetrics.readthedocs.io/en/v0.8.0/classification/hinge_loss.html

$\mathbf{G}^{(i,c)} \in \{0,1\}$ – бинарный индикатор (*one-hot encoding*[11]) принадлежности пикселя i к классу c;

$\mathbf{Y}^{(i,c)} \in \mathbb{R}$ – семантический *логит* (значение непосредственно на выходе слоя перед применением функции активации) для пикселя i и класса c, соответствующий элементу $\mathbf{Y}_{\text{sem}}[c, i, j]$;

$\sigma(\cdot)$ – функция, преобразующая логиты в вероятностное распределение по классам для каждого пикселя.

Ключевые гиперпараметры функции потерь настраиваются на валидационном наборе:

δ_{var} контролирует компактность кластеров. Меньшие значения создают более плотные кластеры, но могут привести к переобучению. Оптимизируется для баланса между точностью границ объектов и устойчивостью к шуму.

δ_{dist} определяет минимальное расстояние между объектами. Большие значения увеличивают разделимость, но могут затруднить обучение при плотном расположении объектов. Выбор значения обусловлен типичным распределением расстояний между объектами в целевом датасете аэрофотоснимков.

λ – балансирующий коэффициент между семантической и инстанс-сегментацией. Эмпирически установленное значение, обеспечивающее сходимость обеих компонент.

Веса классов w_c вычисляются обратно пропорционально частоте классов в тренировочном датасете для компенсации дисбаланса, характерного для аэрофотоснимков (например, преобладание фона «Background»). Конкретнее, $w_c = \frac{N_{\text{total}}}{C \cdot N_c}$, где N_{total} – общее число пикселей, а N_c – число пикселей класса c.

Конкретные значения гиперпараметров $\delta_{\text{var}} = 0.5$, $\delta_{\text{dist}} = 2.0$, $\lambda = 0.5$ и $d = 32$ были выбраны по итогам эксперимента на анализ чувствительности, описанным ниже в подразделе 7.3.2. Для каждого параметра исследовался заданный диапазон со следующими шагами: $\Delta\delta_{\text{var}} = 0.1$, $\Delta\delta_{\text{dist}} = 0.2$, $\Delta\lambda = 0.1$ и $\Delta d \in \{8, 16, 32, 48, 64\}$. Данные значения демонстрируют оптимальный баланс между компактностью кластеров внутри объектов и их достаточным разделением в пространстве эмбеддингов. Окончательные значения были отобраны по критерию максимума F1-меры на валидационной выборке для ключевого класса «Building».

Предложенная функция потерь обладает следующими особенностями:

[11] https://docs.pytorch.org/docs/stable/generated/torch.nn.functional.one_hot.html

Исключение областей перекрытий. При вычислении $\mathcal{L}_{\text{var}}$ пиксели, находящиеся в областях перекрытий объектов, исключаются из рассмотрения. Это критически важно для датасетов, где объекты часто частично перекрываются (например, деревья, здания, транспортные средства).

Иерархическая обработка. Функция потерь работает на двух уровнях: сначала вычисляются локальные характеристики объектов (центры масс), затем анализируются глобальные отношения между объектами (попарные расстояния).

Адаптация к разному количеству объектов. Для изображений с одним или отсутствием объектов $\mathcal{L}_{\text{dist}}$ не вычисляется, предотвращая нестабильность обучения.

Балансировка компонент. Коэффициент λ позволяет регулировать относительный вклад семантической и инстанс-сегментации в соответствии с требованиями конкретной задачи.

Процедура вычисления функции потерь для одного изображения включает следующие шаги:

1. Идентификация объектов. Определение уникальных ID экземпляров на изображении (исключая фон с ID=0).
2. Вычисление центров масс. Для каждого объекта с исключением пикселей в областях перекрытий вычисляется центр масс его эмбеддингов.
3. Вычисление $\mathcal{L}_{\text{var}}$ – среднего квадратичного превышения расстояний эмбеддингов от центра над порогом δ_{var}.
4. Для всех пар объектов вычисление штрафа $\mathcal{L}_{\text{dist}}$, если расстояние между их центрами меньше δ_{dist}, с усреднением по всем парам.
5. Параллельное вычисление кросс-энтропийной потери для семантической сегментации $\mathcal{L}_{\text{semantic}}$.
6. Агрегация. Суммирование компонент с применением балансирующего коэффициента: $\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{semantic}} + \lambda(\mathcal{L}_{\text{var}} + \mathcal{L}_{\text{dist}})$.

6. Организация конвейера инференса

Конвейер инференса реализует процедуру преобразования входного изображения $\mathbf{X} \in \mathbb{R}^{3\times H\times W}$ в панорамическую маску сегментации с последующей визуализацией результатов. Конвейера инференса реализует следующие основные этапы обработки – сегментацию, кластеризацию эмбеддингов, формирование итоговой маски и результата.

6.1. Семантическая и эмбеддинговая сегментация

На первом этапе входное изображение $\mathbf{X}$ проходит через модифицированную U-Net архитектуру M_θ, обученную для решения совместной задачи семантической и инстанс-сегментации:

$$M_\theta(\mathbf{X}) = (\mathbf{Y}_{\text{sem}}, \mathbf{Y}_{\text{emb}}), \qquad \text{где}$$

$\mathbf{Y}_{\text{sem}} \in \mathbb{R}^{C_{sem} \times H \times W}$ – семантические логиты (сырые, ненормализованные оценки) для C_{sem} классов;

$\mathbf{Y}_{\text{emb}} \in \mathbb{R}^{d \times H \times W}$ – d-мерные эмбеддинги (векторные представления низкой размерности, сохраняющие семантическую близость) для каждого пикселя. В этой записи:

C_{sem} – общее количество семантических классов (включая фон);

H, W – высота и ширина изображения соответственно;

d – размерность скрытого эмбеддинг-пространства, выбранная для обеспечения дискриминативности признаков.

Карта семантических классов $\mathbf{M}_{\text{sem}} \in \mathbb{Z}^{H \times W}$ получается применением операции arg max по классовой оси:

$$\mathbf{M}_{\text{sem}}[i,j] = \arg\max_c \mathbf{Y}_{\text{sem}}[c,i,j], \quad \forall i \in [1,H], j \in [1,W], \qquad \text{где}$$

$\mathbf{M}_{\text{sem}}[i,j]$ – итоговый предсказанный класс (целочисленный индекс) для пикселя с координатами (i,j);

$\arg\max_c$] – оператор, возвращающий индекс c класса, для которого значение логита $\mathbf{Y}_{\text{sem}}[c,i,j]$ максимально.

6.2. Кластеризация эмбеддингов

Для пикселей, отнесённых к приоритетной категории, используется алгоритм [12]DBSCAN (Density-Based Spatial Clustering of Applications with Noise) в пространстве эмбеддингов

$$\{S_k\}_{k=1}^{K} = \text{DBSCAN}_{\epsilon,m}\left(\{\mathbf{Y}_{\text{emb}}[i,j] : \mathbf{M}_{\text{sem}}[i,j] \in \mathcal{T}\}\right), \qquad \text{где}$$

$\mathcal{T} \subset \{1, \ldots, C_{\text{sem}}\}$ – множество индексов классов типа «объект» (things), подлежащих инстанс-сегментации;

$\epsilon > 0$ – гиперпараметр, максимальное расстояние между двумя образцами в окрестности для их объединения в один кластер;

$m \in \mathbb{N}$ – гиперпараметр, минимальное количество образцов в ϵ-окрестности точки для образования ядра кластера;

[12] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html

$\{S_k\}_{k=1}^{K}$ – итоговое множество кластеров, где каждый кластер S_k представляет собой множество координат пикселей $\{(i, j)\}$, принадлежащих одному экземпляру объекта;

K – общее количество обнаруженных кластеров (экземпляров).

6.3. Формирование итоговой маски

Панорамическая маска $\mathbf{M}_{\text{final}} \in \mathbb{Z}^{H \times W}$ (финальная разметка, объединяющая семантику и экземпляры) формируется объединением семантических меток для классов «фон» и классов-материй (stuff) с уникальными идентификаторами для каждого экземпляра классов-объектов (things):

$$\mathbf{M}_{\text{final}}[i, j] = \begin{cases} \mathbf{M}_{\text{sem}}[i, j], & \text{если } \mathbf{M}_{\text{sem}}[i, j] \in \mathcal{S}, \\ K_{\text{offset}} + k, & \text{если } (i, j) \in S_k, \end{cases} \qquad \text{где}$$

$\mathcal{S} \subset \{1, \ldots, C_{\text{sem}}\}$ – множество индексов классов типа «объект», для которых применяется кластеризация экземпляров;

K_{offset} – целочисленная константа смещения (обычно $K_{\text{offset}} = C_{\text{sem}}$), гарантирующая, что уникальные идентификаторы экземпляров $(K_{\text{offset}} + k)$ не пересекаются с семантическими индексами классов из $\mathcal{S}$;

$k \in \{1, \ldots, K\}$ – индекс кластера-экземпляра, полученного на предыдущем шаге.

Таким образом, пиксели, принадлежащие фону или объектам классов-материй, сохраняют свои семантические метки, а каждый пиксель, принадлежащий экземпляру объекта k, получает уникальный целочисленный идентификатор.

6.4. Визуализация результатов

Для каждого обнаруженного объекта k вычисляется уверенность $\mathcal{C}_k \in [0, 1]$ на основе относительной площади его ограничивающего прямоугольника:

$$\mathcal{C}_k = \min\left(\frac{w_k \times h_k}{W \times H \times \alpha}, 1.0\right),$$

где w_k, h_k – размеры ограничивающего прямоугольника, W, H – размеры изображения, $\alpha = 0.1$ – коэффициент нормализации.

Визуализация использует цветовую семантику для интуитивного восприятия уровня уверенности:

Зелёный – высокая уверенность ($\mathcal{C}_k > 0.7$).

Оранжевый – средняя уверенность ($0.5 \leq \mathcal{C}_k \leq 0.7$).
Красный – низкая уверенность ($\mathcal{C}_k < 0.5$).

Конвейер предоставляет количественную статистику:

Общее количество обнаруженных объектов с порогом уверенности, превышающим значение τ_{conf} : $N = \sum_{k=1}^{K} \mathbb{I}(\mathcal{C}_k \geq \tau_{\text{conf}})$.
Средняя уверенность: $\bar{\mathcal{C}} = \frac{1}{N} \sum_{k=1}^{K} \mathcal{C}_k$.
Распределение по уровням уверенности.

6.5. Вычислительные характеристики

Предложенный метод обработки обладает следующими ключевыми вычислительными особенностями:

Адаптивная кластеризация. Параметры кластеризации DBSCAN (радиус окрестности ϵ и минимальное количество точек m) не являются фиксированными и автоматически подстраиваются в зависимости от локальной плотности объектов на изображении.

Устранение перекрытий. Для повышения точности сегментации области, где объекты визуально перекрываются, предварительно исключаются из анализа и не участвуют в кластеризации.

Фильтрация шума. В результате кластеризации автоматически отбрасываются небольшие скопления точек, которые признаются шумом (кластеры с числом пикселей менее заданного порога $m_{\min}$).

Работа с исходным разрешением. Все вычислительные этапы, включая кластеризацию, выполняются непосредственно с исходным изображением разрешением $H \times W$ пикселей, что позволяет сохранить максимальную детализацию.

Сложность алгоритма. Временная сложность полного конвейера оценивается как:

$$O\big(H \times W \times (C_{\text{sem}} + d + N_{\text{clusters}})\big), \qquad \text{где}$$

C_{sem} – число семантических классов,
d – размерность признакового вектора,
$N_{clusters}$ – среднее число кластеров.

Требования к памяти. Для хранением семантических карт и многомерных признаков для каждого пикселя необходима память размера:

$$O\big(H \times W \times (C_{\text{sem}} + d)\big).$$

7. Эксперимент

7.1. Условия и параметры обучения

Модель M_θ, архитектура которой приведена на рисунке 1, исследовалась на **специализированном датасете аэрофотоснимков ППК «Роскадастр»**, полученных с помощью квадрокоптера [35]. Датасет включает 435 RGB-изображений высокого разрешения с соответствующими аннотациями в формате JSON (URL[13]LabelMe). Исходные изображения обладали различными размерами. Для обеспечения единообразия на входе модели все изображения были приведены к единому размеру 512×512 пикселей. Соответствующие изменения были внесены и в JSON-файлы с аннотациями – координаты полигонов всех объектов были перемасштабированы с сохранением относительных пропорций и пространственных соотношений.

Каждый JSON-файл содержит полигональную разметку объектов следующих пяти семантических классов:

Дачный домик/коттедж (метка «Building») – 12 470 экземпляров;
Теплица (метка «Greenhouse») – 6 450 экземпляров;
Хозпостройка (метка «Outbuilding») – 2 150 экземпляров;
Транспортное средство (метка «Vehicle») – 1 516 экземпляров;
Бассейн (метка «Swimming») – 490 экземпляров.

Общее количество размеченных объектов – 23 076. Особое внимание в эксперименте уделялось классу инстанс-сегментации «Building», что объясняется его наибольшей представленностью в датасете (более 54% от общего числа объектов) и практической значимостью для задач автоматического картографирования и анализа застройки.

Датасет был разделен на тренировочную, валидационную и тестовую выборки в пропорции 70%:15%:15% с сохранением баланса классов в каждой из них. Для обеспечения репрезентативности разбиения использовалась стратифицированная выборка по плотности объектов на изображениях.

Модель M_θ исследовалась со следующими параметрами:

Входные каналы: 3 (RGB).
Размерность эмбеддингов: $d = 32$ (см ниже табл. 4).
Количество семантических классов: $C_{\text{sem}} = 6$ (фон $^+$ 5 объектов).

Гиперпараметры обучения модели:

[13] https://github.com/wkentaro/labelme

Оптимизатор: `AdamW`[14] с параметрами:
Скорость обучения: $\mathrm{lr} = 10^{-4}$.
Вес затухания: $\mathrm{weight\_decay} = 10^{-4}$.
$\beta_1 = 0.9,\ \beta_2 = 0.999$.
Планировщик скорости обучения: `CosineAnnealingLR`[15]
Максимальное количество эпох: $T_{\max} = 200$.
Размер батча: 4 изображения (ограничено памятью GPU).
Ранняя остановка (`EarlyStopping`[16]):
Терпение: 15 эпох.
Минимальное улучшение: $\Delta_{\min} = 10^{-5}$.
Градиентный клиппинг: Норма градиентов ограничена значением 1.0.

Малый размер батча обусловлен большим разрешением изображений (512×512 пикселей) и многоканальными промежуточными представлениями в U-Net архитектуре, которые требуют значительной видеопамяти. При использовании GPU с 32 ГБ памяти (NVIDIA Tesla V100) максимальный размер батча для обеспечения стабильной работы Batch Normalization и предотвращения ошибок нехватки памяти (out-of-memory) составил четыре изображения. Эксперименты с накоплением градиентов для эмуляции большего размера батча не показали значимого улучшения качества.

Ранняя остановка применялась при отсутствии улучшения валидационной функции потерь в течение 15 последовательных эпох, где улучшением считалось снижение loss не менее чем на $\Delta_{\min} = 10^{-5}$. При срабатывании механизма модель автоматически восстанавливала веса из эпохи с наилучшим значением валидационного loss.

Эксперименты проводились на вычислительном кластере со следующей конфигурацией:

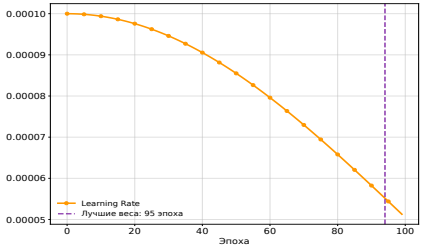
GPU: NVIDIA Tesla V100 (32 ГБ памяти).
CPU: Intel Xeon Gold 6248R (24 ядра).
Оперативная память: 128 ГБ DDR4.
ПО: Python 3.12.12, PyTorch 2.9.0+cu126, CUDA 12.6.

Отслеживавшиеся на каждой эпохе в процессе обучения метрики на тренировочной и валидационной выборках, представлены на рисунке 2.

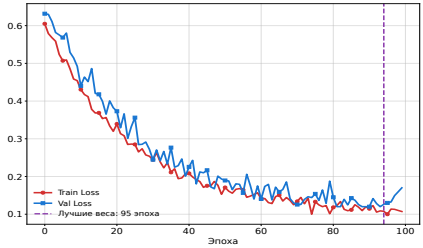
[14] https://docs.pytorch.org/docs/stable/generated/torch.optim.AdamW.html
[15] https://docs.pytorch.org/docs/stable/generated/torch.optim.lr_scheduler.CosineAnnealingLR.html
[16] https://docs.pytorch.org/ignite/generated/ignite.handlers.early_stopping.EarlyStopping.html

7.2. Результаты обучения

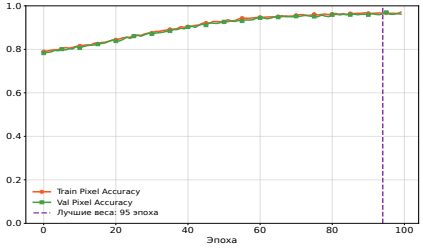
Среднее время обучения одной эпохи составило 3 минуты 12 секунд.



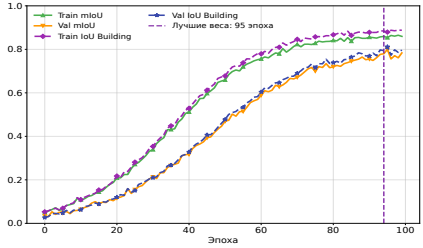
(*а*) Скорость обучения модели



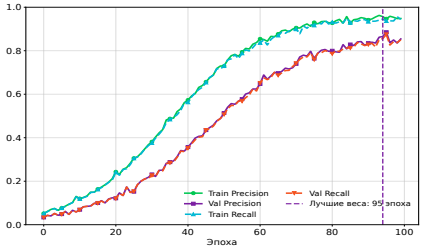
(*б*) Функция потерь



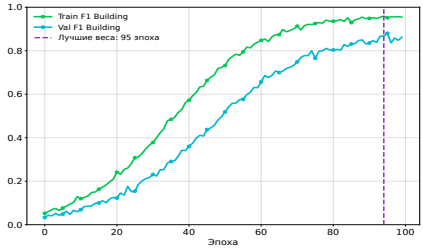
(*в*) Точность классификации пикселей



(*г*) Среднее значение метрики *IoU* (mIoU) и значение IoU



(*д*) Метрики точности *Precision* и полноты *Recall*



(*е*) Гармоническое среднее точности и полноты F1

Рисунок 2. Графики метрик точности модели для целевого класса инстанс-сегментации «Building» в процессе её обучения и валидации

Общее время обучения до срабатывания ранней остановки (95 эпохи) –

приблизительно 5 часов. Продолжительность обучения обусловлена следующими факторами:

- Высокое разрешение изображений (512×512 пикселей), требующее обработки большого объёма данных на каждой эпохе.
- Многозадачный характер модели с одновременным вычислением семантической и эмбеддинговой компонент функции потерь.
- Необходимость сохранения промежуточных признаков для skip-коннекторов, увеличивающая требования к памяти и вычислениям.

Для сравнения, обучение классической Mask R-CNN (ResNet-50-FPN) на том же датасете и оборудовании до сопоставимого уровня качества (IoU ≈ 0.81) заняло 8.5 часов. Этим предложенная архитектура продемонстрировала выигрыш в 41% по времени обучения при сохранении конкурентоспособной точности (подробности в следующем подразделе 7.3).

Среднее время инференса одного изображения размером 512×512 составило 62 мс на GPU NVIDIA V100 (32 ГБ), включая полный цикл: прямой проход сети и этап кластеризации DBSCAN. На более доступной видеокарте NVIDIA RTX 4090 (24 ГБ) время инференса увеличивается до 78 мс. Разница во времени между обучением и инференсом объясняется отсутствием на этапе инференса затратных операций обратного распространения ошибки, обновления весов и градиентных вычислений. Кроме того, процедура кластеризации оптимизирована: применяется приближённый вариант DBSCAN с предварительной фильтрацией пикселей по семантической маске и уменьшением числа точек (downsampling эмбеддингового поля), что существенно снижает накладные расходы постобработки.

Наилучшие результаты достигнуты на 95-й эпохе обучения и представлены в таблице 1.

Таблица 1. Метрики модели на 95-й эпохе (лучшие веса)

Метрика	**Обучение**	**Валидация**	**Разница**
Loss	0.240	0.250	0.010
Learning Rate	5.59×10^{-5}		
Pixel Accuracy	0.985	0.970	0.015
mIoU	0.880	0.800	0.080
IoU «Building»	0.892	0.812	0.080
Precision «Building»	0.905	0.885	0.020
Recall «Building»	0.880	0.875	0.005
F1 «Building»	0.892	0.880	0.012

Из таблицы видно, что модель демонстрирует устойчивую сходимость. Наблюдается умеренный разрыв между значениями функции потерь на обучающей (0.240) и валидационной (0.250) выборках, составляющий 0.010.

Модель достигает высоких показателей метрик точности на обучающей выборке, при этом сохраняя хорошее, хотя и несколько сниженное, качество на валидационных данных. Особенно показательны значения F1-score для Building: 0.892 на обучающей выборке и 0.880 на валидационной, что свидетельствует о хорошем балансе между точностью и полнотой. Наблюдается умеренный разрыв (8–11%) между метриками на обучающей и валидационной выборках, что объясняется повышенной сложностью сцен валидационной подвыборки (плотная застройка, частичные перекрытия объектов). Тем не менее, стабильная сходимость и чёткий пик качества на 95-й эпохе свидетельствуют о хорошей обобщающей способности модели.

Для реализации модели была выбрана библиотека *PyTorch*[17]. Исходный код реализации модели и все эксперименты доступны в виде *интерактивного блокнота*[18] Jupyter на платформе Google Colab.

На рисунке 3 показаны результаты сегментации тестового изображения, содержащего участки с изолированными зданиями, с умеренной и с плотной застройками.

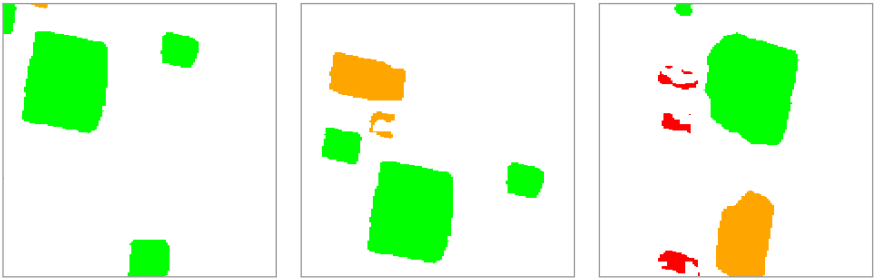
В верхнем ряду, на рисунке 3*а*, 3*б* и 3*в*, показаны исходные семантические маски, выделяющие все объекты класса «Building». Маски, представленные в соответствии с цветовым кодированием из раздела 6.4, служат входными данными для последующих этапов. Этот ряд иллюстрирует исходную задачу – необходимость разделить единую семантическую область на отдельные экземпляры дачных домиков/коттеджей.

Средний ряд, рисунок 3*г*-3*е*, демонстрирует переход от семантики к экземплярам через визуализацию пространственных признаков (эмбеддингов). Здесь каждый пиксель спроецирован в пространство, где его координаты определяются не цветом, а сходством контекстных признаков. На рисунке 3*г* для разреженной застройки наблюдаются чётко разделённые и компактные кластеры, что указывает на высокую различимость объектов. По мере увеличения плотности застройки на рисунках 3*д* и 3*е* кластеры начинают прилегать друг к другу, а их границы становятся менее выраженными, визуализируя тем самым основную вычислительную сложность задачи.

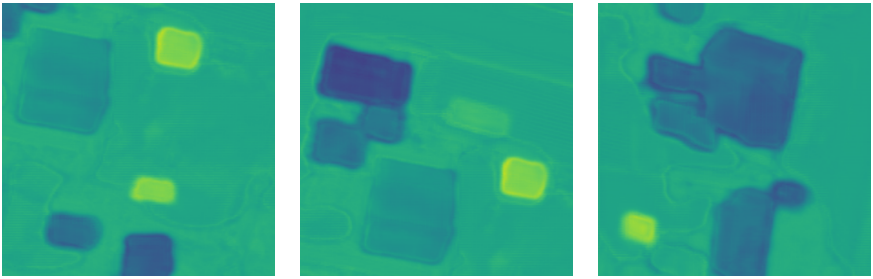
Нижний ряд, рисунок 3*ж*-3*и*, отображает итоговый результат – карту сегментации отдельных экземпляров, полученную путём кластеризации эмбеддингов. Каждому отдельному зданию присвоен уникальный цвет согласно разделу 6.4. Результаты подтверждают наблюдения: изолированные здания правильной формы (рисунок 3*ж*) сегментируются с высокой точностью и уверенностью. В условиях плотной и сложной

[17] https://pytorch.org/docs/stable/index.html

[18] https://colab.research.google.com/drive/1syACFrW4N1MKNUuNO87lmJWkOQDqeYnj#scrollTo=3mQXr-LD5kTF



(*а*) Изолированные объекты (*б*) Умеренная плотность объектов (*в*) Высокая плотность объектов



(*г*) Чёткие кластеры (*д*) Прилегающие кластеры (*е*) Пересекающиеся кластеры

(*ж*) Высокие оценки (*з*) Смешанные оценки (*и*) Низкие оценки

Рисунок 3. Результаты сегментации объектов класса дачного домика для трёх тестовых сцен

застройки (рисунок 3*з*, 3*и*) алгоритм, несмотря на сложности, успешно разделяет большинство перекрывающихся объектов, хотя уверенность модели для таких областей, как правило, снижается, что выражается в менее стабильных границах сегментов.

7.3. Сравнение с современными методами сегментации

Для оценки конкурентоспособности предложенного подхода была проведена серия экспериментов по сопоставлению с рядом современных методов семантической и инстанс-сегментации. Все модели обучались и тестировались на одном и том же специализированном датасете аэрофотоснимков с сохранением одинаковых стратегий аугментации и валидации. Для методов, изначально предназначенных для датасетов общего назначения (COCO, Cityscapes), были адаптированы входные разрешения и проведено дообучение (fine-tuning) на целевом датасете. Результаты сравнения по ключевым метрикам для класса «Building» представлены в таблице 2.

Таблица 2. Сравнение предложенного метода с современными архитектурами сегментации

Метод (год)	**IoU ↑**	**F1-score ↑**	**mIoU ↑**	**Время инференса ↓ (ms/img)**
Mask R-CNN (2017)	0.824	0.875	0.795	150
Panoptic-DeepLab (2020)	0.835	0.882	0.810	80
Mask2Former (2022)	0.855	0.898	0.832	120
DINOv2 (2023–2025)	0.862	0.905	0.840	105
SAM 2 (LoRA-адаптация) (2024–2025)	0.858	0.900	0.835	65
MR-DeepLabv3^{+} (2025)	0.854	0.902	0.838	88
OneFormer (2025)	0.865	0.908	0.842	95
Предложенная (2026)	**0.812**	**0.880**	**0.800**	**62**

Как видно из таблицы 2, предложенная модель демонстрирует сопоставимую эффективность с классическим Mask R-CNN: её точность по IoU (0.812 против 0.824) и F1-score (0.880 против 0.875) находится на одном уровне, при этом обеспечивая значительное ускорение инференса – 62 мс. против 150 мс. на изображение размером 512×512.

По сравнению с современными трансформерными архитектурами (Mask2Former, OneFormer, DINOv2) предлагаемая модель уступает в абсолютной точности на 4–5% (например, OneFormer достигает IoU = 0.865), однако превосходит их по скорости в 1.5–2 раза и требует значительно меньше вычислительных ресурсов. Это делает её особенно привлекательной для сценариев массовой обработки данных в условиях ограниченных ресурсов.

Особенно показательно сравнение с двумя специализированными моделями 2025 года:

SAM 2 (LoRA-адаптация для дистанционного зондирования) обеспечивает сопоставимую скорость (65 мс против 62 мс), но требует сложного пайплайна с промптами и дообучением на этапе инференса. Предложенная модель, напротив, работает автономно и проще в развёртывании.

MR-DeepLabv3+, оптимизированная для сегментации зданий, демонстрирует на 4.2% более высокую точность (IoU = 0.854), однако работает на 30% медленнее (88 мс против 62 мс). Для задач оперативного картографирования и анализа больших территорий такая разница в скорости может быть критичной.

Таким образом, предложенная модифицированная U-Net с дискриминативными эмбеддингами представляет собой практичный компромисс между точностью, скоростью инференса и архитектурной простотой. Она особенно эффективна для специализированных данных аэрофотосъёмки, где требуется баланс между качеством сегментации и производительностью на доступном оборудовании.

7.3.1. *Анализ значимости компонентов функции потерь*

В таблице 3 представлены результаты исследования влияния коэффициента балансировки λ и компонентов предложенной функции потерь $\mathcal{L}_{\text{total}}$. Все эксперименты проводились с фиксированными гиперпараметрами ($\delta_{\text{var}} = 0.5$, $\delta_{\text{dist}} = 2.0$) и размерностью эмбеддингов $d = 32$.

Таблица 3. Анализ значимости компонентов функции потерь (метрики на валидационной выборке)

Конфигурация	**IoU (val)** ↑	**F1-score (val)** ↑	**mIoU (val)** ↑
Полная ($\lambda = 0.5$)	0.812	0.880	0.800
$\lambda = 0.3$	0.775	0.858	0.762
$\lambda = 0.7$	0.782	0.863	0.769
Без $\mathcal{L}_{\text{var}}$	0.732	0.826	0.715
Без $\mathcal{L}_{\text{dist}}$	0.745	0.835	0.728
Только $\mathcal{L}_{\text{semantic}}$	0.712	0.802	0.695

Результаты подтверждают критическую важность обоих компонентов $\mathcal{L}_{\text{var}}$ и $\mathcal{L}_{\text{dist}}$ для задачи инстанс-сегментации: их исключение приводит к заметному падению метрик (снижение IoU на 7–9%, mIoU на 8–9%). Отсутствие обоих компонентов (последняя строка таблицы) приводит к наиболее значительной деградации качества, что демонстрирует принципиальную роль эмбеддинг-ориентированного обучения для разделения экземпляров. Коэффициент балансировки $\lambda = 0.5$ обеспечивает оптимальное качество, превосходя альтернативные значения $\lambda = 0.3$ и $\lambda = 0.7$ на 3–4% по основным метрикам. Полная конфигурация с $\lambda = 0.5$ демонстрирует наилучшие результаты по всем метрикам.

7.3.2. *Чувствительность к гиперпараметрам*

Зависимость качества модели от гиперпараметров δ_{var}, δ_{dist}, λ и размерности эмбеддингов d представлена в табл. 4.

Анализ полученных значений выявил следующие закономерности:

Таблица 4. Влияние гиперпараметров на метрики (F1-score для класса «Building»)

Параметр	Диапазон	Оптимум	Влияние
δ_{var}	$[0.2, 1.0]$	0.5	Высокая чувствительность
δ_{dist}	$[1.0, 3.0]$	2.0	Широкий оптимум
λ	$[0.1, 1.0]$	0.5	Баланс IoU и mAP
d (эмбеддингов)	$[8, 64]$	32	Насыщение при $d = 32$

Наибольшее влияние на итоговое качество оказывает порог δ_{var}, управляющий компактностью эмбеддинг-кластеров. Оптимальное значение лежит в узком диапазоне $[0.4, 0.6]$.

Порог разделения δ_{dist} имеет широкий оптимум ($[1.8, 2.4]$), что согласуется с естественной вариативностью расстояний между объектами на аэроснимках.

Коэффициент λ демонстрирует ожидаемый компромисс: при низких значениях снижается mAP, а при высоких – падает IoU. Оптимальный баланс достигается в интервале $[0.4, 0.7]$.

Качество растёт с увеличением размерности d до 32, после чего наступает насыщение. Выбор $d = 32$ обеспечивает оптимальное соотношение точности и вычислительных затрат.

7.3.3. *Сравнение альтернативных функций потерь*

В качестве дополнительного эксперимента была исследована возможность замены Hinge Loss в компонентах $\mathcal{L}_{\text{var}}$ и $\mathcal{L}_{\text{dist}}$ на другие функции, применяемые в метрическом обучении [38]. Результаты представлены в таблице 5.

Таблица 5. Сравнение альтернативных функций потерь для обучения эмбеддингов

Тип функции	F1-score ↑	mIoU ↑
Hinge Loss (база)	0.880	0.800
Contrastive Loss	0.875	0.792
Triplet Loss	0.876	0.794
Center Loss	0.872	0.789

Эксперимент показал, что Hinge Loss, использованная в предложенном методе, обеспечивает наилучшие результаты. Более современные Contrastive и Triplet Loss не показали значимого улучшения (проигрыш 0.4–0.8% по F1-score и 0.6–1.1% по mIoU), требуя при этом тщательного подбора гиперпараметров и демонстрируя склонность к нестабильности на данных с малым количеством экземпляров на изображение.

8. Обсуждение

8.1. Ключевые преимущества архитектуры

Главная особенность архитектуры заключается в совместном решении задач семантической и инстанс-сегментации в рамках единой модели, что позволяет избежать накопления ошибок, характерного для последовательных подходов. Эмбеддинг-ориентированная парадигма обеспечивает гибкость в разделении перекрывающихся объектов и адаптацию к различным сценам через настройку параметров кластеризации.

Для аэрофотоснимков архитектура особенно эффективна благодаря сохранению контекстной информации через skip-коннекторы и способности обрабатывать объекты различных масштабов. Метрическое обучение в пространстве эмбеддингов снижает чувствительность модели к изменениям условий съёмки, что критически важно для дистанционного зондирования.

8.2. Ограничения и вызовы

Основным ограничением подхода является повышенная вычислительная сложность, связанная с хранением и обработкой эмбеддингов для всех пикселей изображения. Кластеризация DBSCAN может ограничивать производительность системы при обработке изображений высокого разрешения.

Качество инстанс-сегментации существенно зависит от правильного выбора параметров алгоритма кластеризации и балансирующего коэффициента λ в функции потерь $\mathcal{L}_{\text{total}}$ из (1). Кроме того, эффективность метода чувствительна к качеству и согласованности разметки тренировочных данных, особенно в областях перекрытий объектов.

8.3. Влияние глубины архитектуры на качество сегментации

Важным аспектом является влияние количества блоков энкодера и декодера на качество сегментации. Выбор 4 уровней в текущей реализации представляет собой оптимальный компромисс для задачи сегментации аэрофотоснимков:

- Достаточная глубина для извлечения признаков объектов разного масштаба.
- Минимально необходимое уменьшение разрешения (в 16 раз) для создания информативного `Bottleneck`.
- Сохранение пространственной информации через эффективные skip-конекторы.
- Управляемый размер модели для обучения на доступных вычислительных ресурсах.

Экспериментальные наблюдения показывают, что увеличение глубины до 5 уровней незначительно улучшает качество сегментации (прирост mIoU менее 1%), но увеличивает время обучения на 40%. Уменьшение до 3 уровней приводит к существенному снижению качества сегментации мелких объектов. Наибольший выигрыш от увеличения глубины наблюдается для сложных сцен с большим количеством перекрывающихся объектов.

8.4. Перспективы развития

Перспективными направлениями для будущих исследований являются:

- Интеграция механизмов внимания для улучшения выделения границ объектов.
- Оптимизация вычислительной эффективности через разреженное представление эмбеддингов.
- Расширение функциональности для поддержки слабообученного и мультимодального обучения
- Автоматизация подбора параметров кластеризации.

Практическая значимость метода охватывает различные области, включая городское планирование, сельское хозяйство, экологический мониторинг и системы безопасности.

Заключение

Предложенная архитектура представляет собой сбалансированный компромисс между точностью сегментации, гибкостью применения и сложностью реализации. Несмотря на определённую вычислительную затратность, метод демонстрирует конкурентные результаты на задачах сегментации аэрофотоснимков и открывает новые возможности для исследований в области совместной семантической и инстанс-сегментации. Дальнейшая работа будет направлена на оптимизацию производительности и расширение функциональных возможностей метода.

Список использованных источников

[1] B. Cheng, M. D. Collins, Y. Zhu, T. Liu, T. S. Huang, H. Adam, L.-Ch. Chen *Panoptic-DeepLab: A simple, strong, and fast baseline for bottom-up panoptic segmentation* // IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).– 2020.– Pp. 12475–12485. doi ↑141, 145

[2] L.-Ch. Chen, G. Papandreou, F. Schroff, H. Adam *Rethinking atrous convolution for semantic image segmentation* // arXiv preprint arXiv:1706.05587.– 2017. doi URL ↑141

[3] J. Hosang, R. Benenson, B. Schiele *Learning non-maximum suppression* // arXiv preprint arXiv:1705.02950.– 2017. doi URL ↑141

[4] K. He, G. Gkioxari, P. Dollár, R. Girshick *Mask R-CNN* // arXiv preprint arXiv:1703.06870.– 2018. ↑141

[5] Z. Tian, C. Shen, H. Chen, T. He *Conditional convolutions for instance segmentation* // European Conference on Computer Vision (ECCV).– 2020.– Pp. 282–298. ↑141, 145

[6] X. Wang, R. Zhang, T. Kong, L. Li, C. Shen *SOLOv2: Dynamic and fast instance segmentation* // arXiv preprint arXiv:2003.10152.– 2020. ↑141, 145

[7] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, P. Dollár *Microsoft COCO: Common Objects in Context* // European Conference on Computer Vision (ECCV).– 2014.– C. 740–755. ↑142, 143, 145

[8] O. Ronneberger, P. Fischer, T. Brox *U-Net: Convolutional networks for biomedical image segmentation* // Medical Image Computing and Computer-Assisted Intervention — MICCAI 2015.– 2015.– Pp. 234–241. ↑142, 146

[9] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, J. Liang *UNet++: A nested U-Net architecture for medical image segmentation* // Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support.– 2018.– Pp. 3–11. ↑142, 146

[10] H. Huang, L. Lin, R. Tong, H. Hu, Q. Zhang, Y. Iwamoto, X.-H. Han, Y.-W. Chen, J. Wu *UNet 3+: A full-scale connected UNet for medical image segmentation* // IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).– 2020.– Pp. 1055–1059. ↑142, 146

[11] O. Oktay, J. Schlemper, L. L. Folgoc, M. Lee, M. Heinrich, K. Misawa, K. Mori, S. McDonagh, N. Y. Hammerla, B. Kainz, B. Glocker, D. Rueckert *Attention U-Net: learning where to look for the pancreas* // arXiv preprint arXiv:1804.03999.– 2018. ↑142, 146, 151

[12] N. Siddique, S. Paheding, C. P. Elkin, V. Devabhaktuni *U-Net and its variants for medical image segmentation: A review of theory and applications* // IEEE Access.– 2021.– Vol. **9**.– Pp. 82031–82057. ↑142

[13] H.-Y. Zhou, J. Guo, Y. Zhang, L. Yu, L. Wang, Y. Yu *nnFormer: Interleaved transformer for volumetric segmentation* // arXiv preprint arXiv:2109.03201.– 2022. ↑142, 144

[14] Y. Wang, N. Huang, T. Li, Y. Yan, X. Zhang *MedFormer: A multi-granularity patching transformer for medical time-series classification* // arXiv preprint arXiv:2405.19363.– 2024. ↑142

[15] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, R. Girdhar *Masked-attention mask transformer for universal image segmentation* // IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).– 2022.– Pp. 1290–1299. ↑142

[16] T. Zhang, X. Tian, Y. Wu, S. Ji, X. Wang, Y. Zhang, P. Wan *DVIS: Decoupled video instance segmentation framework* // IEEE/CVF International Conference on Computer Vision (ICCV).– 2023.– Pp. 1282–1291. ↑143

[17] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Mouby, M. Assran, N. Ballas, W. Galuba, R. Howes, P.-Y. Huang, S.-W. Li, I. Misra, M. Rabbat, V. Sharma, G. Synnaeve, H. Xu, H. Jegou, J. Mairal, P. Labatut, A. Joulin, P. Bojanowski *DINOv2: Learning robust visual features without supervision* // arXiv preprint arXiv:2304.07193.– 2024. ↑143

[18] P. Voigtlaender, M. Krause, A. Osep, J. Luiten, B. B. G. Sekar, A. Geiger, B. Leibe *MOTS: Multi-object tracking and segmentation* // arXiv preprint arXiv:1902.03604.– 2019. ↑143

[19] J. Jain, J. Li, M. Chiu, A. Hassani, N. Orlov, H. Shi *OneFormer: One transformer to rule universal image segmentation* // arXiv preprint arXiv:2211.06220.– 2022. ↑143

[20] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, B. Schiele *The Cityscapes dataset for semantic urban scene understanding* // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).– 2016.– С. 3213–3223. ↑143, 145

[21] B. Zhou, H. Zhao, X. Puig, T. Xiao, S. Fidler, A. Barriuso, A. Torralba *Semantic understanding of scenes through the ADE20K dataset* // arXiv preprint arXiv:1608.05442.– 2018. ↑143, 145

[22] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, B. Guo *Swin Transformer: Hierarchical vision transformer using shifted windows* // Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV).– 2021.– Pp. 10012–10022. ↑143

[23] J. Chen, Y. Lu, Q. Yu, X. Luo, E. Adeli, Y. Wang, L. Lu, A. L. Yuille, Y. Zhou *TransUNet: Transformers make strong encoders for medical image segmentation* // arXiv preprint arXiv:2102.04306.– 2021. ↑144

[24] E. U. Henry, O. Emebob, C. A. Omonhimain *Vision transformers in medical imaging: A review* // arXiv preprint arXiv:2211.10043.– 2022. ↑144

[25] D. Niu, X. Wang, X. Han, L. Lian, R. Herzig, T. Darrell *Unsupervised universal image segmentation* // IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).– 2024.– Pp. 22744–22754. ↑144

[26] X. Wang, R. Girdhar, S. X. Yu, I. Misra *Cut and learn for unsupervised object detection and instance segmentation* // arXiv preprint arXiv:2301.11320.– 2023. ↑144

[27] A. Kirillov, K. He, R. Girshick, C. Rother, P. Dollár *Panoptic segmentation* // IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).– 2019.– Pp. 9404–9413. ↑144

[28] L. Yuan, M. Shi, Z. Yue, Q. Chen *LoSh: Long-short text joint prediction network for referring video object segmentation* // IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).– 2024.– Pp. 10236–10246. ↑144

[29] J. Wu, Y. Jiang, P. Sun, Z. Yuan, P. Luo *Language as queries for referring video object segmentation* // arXiv preprint arXiv:2201.00487.– 2022. ↑144

[30] W. Zhang, J. Pang, K. Chen, C. C. Loy *K-Net: Towards Unified Image Segmentation* // arXiv preprint arXiv:2106.14855.– 2021. ↑144

[31] X. Wang, X. Zhang, Y. Cao, W. Wang, C. Shen, T. Huang *SegGPT: Segmenting everything in context* // arXiv preprint arXiv:2304.03284.– 2023. ↑144

[32] Y. Wang, L. Shang, Y. Liu *Precise building semantic segmentation in remote sensing images via MR-DeepLabv3+ network* // Scientific Reports.– 2025.– Т. **15**. ↑144

[33] И. В. Винокуров *Повышение точности сегментирования объектов с использованием генеративно-состязательной сети* // Программные системы: теория и приложения.– 2025.– Т. **16**.– № 2(65).– С. 111–152. ↑146

[34] И. В. Винокуров, Д. А. Фролова, А. И. Ильин, И. Р. Кузнецов *Сравнительный анализ архитектур backbone для инстанс-сегментации объектов на аэрофотоснимках с использованием Mask R-CNN* // Программные системы: теория и приложения.– 2025.– Т. **16**.– № 4(67).– С. 173–216. ↑146

[35] И. В. Винокуров *Использование модели Mask R-CNN для сегментации объектов недвижимости на аэрофотоснимках* // Программные системы: теория и приложения.– 2025.– Т. **16**.– № 1(64).– С. 3–44. ↑159

[36] A. Kirillov, R. Girshick, K. He, P. Dollár *Panoptic feature pyramid networks* // IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).– 2019.– Pp. 6399–6408. ↑

[37] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo *Segment Anything 2 (SAM 2): Moving from Images to Videos* // arXiv preprint arXiv:2407.10323.– 2024. ↑

[38] M. Kaya, H. Ş. Bilge *Deep metric learning: A survey* // Symmetry.– 2019.– Vol. **11**.– No. 9.– Pp. 1066. ↑167

Поступила в редакцию 06.01.2026;
одобрена после рецензирования 16.01.2026;
принята к публикации 26.02.2026;
опубликована онлайн 12.03.2026.

Рекомендовал к публикации *к.т.н. Е. П. Куршев*

Информация об авторе:

Игорь Викторович Винокуров

Кандидат технических наук (PhD), ассоциированный профессор в Финансовом Университете при Правительстве Российской Федерации. Область научных интересов: информационные системы, информационные технологии, технологии обработки данных

0000-0001-8697-1032

e-mail: *igvvinokurov@fa.ru*

Декларация об отсутствии личной заинтересованности: *благополучие автора не зависит от результатов исследования.*