VisionLabs LUNA PLATFORM 5

Release notes

Contents

LUNA PLATFORM v.5.67.0

LP changes

- The ability to perform video analytics on GPU has been added.

  GPU is enabled by specifying "gpu" in the "decoder_device_class" setting of the "LUNA_REMOTE_-SDK_VIDEO_SETTINGS" section.

- SDK has been updated to version 5.21.0.

  In this version LUNA PLATFORM:

  - HumanDetector has been updated to version v6.
  - DeepFakeDetector has been updated to v3.

  Important: The default threshold for redetection of FaceDetV3 detector has been updated from "0.3" to "0.5". When LUNA PLATFORM is updated, the threshold for redetection in the "redetect_-score_threshold" setting of the "LUNA_REMOTE_SDK_FACE_DETECTOR_SETTINGS" section **will not be updated automatically**. You must manually update the threshold to "0.5" to use the default redetection logic. If a custom value was previously set, you must update the threshold value yourself.

  When installing LUNA PLATFORM from scratch, the "redetect_score_threshold" value will also default to "0.3". It is necessary to use the dump file "platform_settings.json" from the distribution package, or update the threshold value after installation. In the next release, when installing from scratch, the value will be automatically set to "0.5".

- A mechanism for specifying trusted detections has been added, enabling you to explicitly specify which detections do not need to be redetected.

  It is assumed that trusted detections were obtained using VisionLabs algorithms. Marking a third-party detection as trusted may affect the evaluation results.

  You can indicate whether the detection is trusted using the "application/json" scheme (the "trusted_detections" parameter) or the "multipart/form-data" scheme (the "X-Luna-Trusted-Detections" header) in the following requests, where detections are explicitly specified instead of images:

  - "detect faces"
  - "sdk"
  - "iso"
  - "generate events"
  - "perform verification"
  - "generate stream events (beta)"

- In the "generate stream events (beta)" request with specifying the source image (`sources` > `source_type` > `raw_image`) an optional field `origin_bounding_box` has been added,

enabling to specify the coordinates of a bbox with a face or body in the coordinate system of the source image, which will be saved in the generated event.

Thus in the request you can specify both bbox coordinates in the Crop image coordinate system (`bounding_box` parameter) and bbox coordinates in the source image coordinate system (`origin_bounding_box` parameter).

Also, the parameter `sources > source > face/body > bounding_box`, available when specifying detections in a request (`sources > source_type > detections`), has been renamed to `origin_bounding_box`.

• The support for setting the DROI relative to the source frame has been added.

DROI specifies a region of interest relative to the source frame. DROI, unlike ROI, is not an estimation parameter, but works as a filter after estimation has been performed in accordance with the "targets" field. In other words, ROI is for estimation and optimization, DROI is for implementing business logic. You can use DROI together with ROI or separately. For example, if after execution the estimate taking into account the ROI of the number of people in the frame was equal to M, then after performing an additional filter on the DROI area, the number of people in the frame can be reduced to N.

The DROI on the source frame is specified by the following parameters:

  – "area" — Geometry of the region of interest. The parameter is represented as an array of polygons. Each polygon is represented by an array of objects, where each object contains the x and y coordinates of the polygon's vertex. For example, you can define a triangular area of interest.
  – "mode" – mode of specifying "x", "y". Two modes are available:
    * "abs" — Parameters "x", "y", "width" and "height" are set in pixels.
    * "percent" — Parameters "x", "y", "width" and "height" are set as percentages of the current frame size.
  – "form" — Format of the region of interest. In the current implementation there is only one possible value, "common". Future implementations will support other formats in addition to polygons.

• The ability to set the time to live of task results has been added to all tasks.

TTL is set in `result_storage_policy > ttl` parameter in task or schedule creation requests.

It is not possible to add TTL of task results to already created and executed tasks.

You can add the TTL of task results to an already created schedule by using the "replace tasks schedule" request. TTL of task results will not be applied to tasks created or running at the time of the request.

• Added automatic sending of the header `Set-Cookie: LUNA_AUTH_TOKEN=""; Path=/; Max -Age=0` to the client application when it receives a 401 error from the API service due to a rotten

authorization token placed in the Cookie.

LP fixed errors

- Fixed migration of Configurator settings with revision `c51cdfac`.

  This migration failed if the `bucket` field was not present in the S3 setting of the Image Store service.

- Removed unnecessary authorization for "get version", "get health", "set login cookie" and "clear login cookie" requests.

LUNA PLATFORM v.5.64.0

LP changes

- The ability to set time to live (TTL) for objects in bucket (both local and S3) has been added.

  TTL for objects is calculated relative to the GMT time format.

  TTL for objects is set in days in the following ways:

  - During the creation of a bucket for all objects at once (basic TTL bucket policy).
  - After creating a bucket for specific objects using requests to the corresponding resources.

  The number of days is selected from the list in the corresponding requests (see below).

  In addition to the number of days, the parameter can take the value "-1", meaning that objects should be stored indefinitely.

  Configuring basic TTL bucket policy

  The basic TTL bucket policy can be configured in the following ways:

  - Using the `--bucket-ttl` flag for the `lis_bucket_create.py` script. For example, "python3 ./base_scripts/lis_bucket_create.py -ii –bucket-ttl=2".
  - Using a request to the Image Store service. For example, "curl -X POST http://127.0.0.1:5020/1/buckets?buck samples?ttl=2".

  Configuring TTL for specific objects

  TTL for specific objects can be configured using the "ttl" parameter in the following places:

  - In the "storage_policy" > "face_sample_policy", "body_sample_policy" and "image_origin_-policy" handler policies.
  - In the requests "create object", "create images" and "save sample".

  If the "ttl" parameter is not specified, then the basic policy of the bucket in which the object is located (see above) will be applied.

  Supported cloud providers

Amazon S3 cloud providers, Yandex cloud storage and MinIO are supported.

Adding TTL to existing objects

You can add a TLL to an existing object using PUT requests to the `/objects`, `/images`, `/samples` `/{sample_type}` resources of the Image Store service.

You can add a TTL to an existing local bucket using a PUT request to the Image Store resource `/buckets`.

To add a TTL for a bucket located in S3, you need to perform a migration using the "base_-scripts/migrate_ttl_settings" script from the Image Store service. This is because for TTL objects in S3 is applied via tag related filters. The command to perform S3 bucket migration is given in the installation manual. See "Migration to apply TTL to objects in S3" for details on S3 bucket migration.

Expiration of TTL

When an object's TTL comes to an end, it is marked for deletion. For local buckets, the cleanup task is performed once a day (at 01:00 am). S3 buckets use internal ttl configuration rules.

Note that there may be a delay between the expiration date and the date the item is actually deleted. Both for local storage and S3.

Search for expiring objects

To find out when an object expires, you can use queries with the HEAD methods on the `/objects` and `/images` resources. These requests return `X-Luna-Expiry-Date` response headers, which indicate the date on which the object is no longer eligible for persistence.

See "TTL for objects" in the administrator manual for details.

- Now Handlers service will send only one notification if a user has two callbacks with the same settings.

  If a user has created two callbacks that differ only in filters, the system will send a notification only once, even if the event satisfies both filters.

  For example, if two callbacks are configured to one third-party system with the only difference being that one callback has filters on passing Liveness and the other on reaching a certain level of "similarity", then when an event is generated that satisfies both filters, only one notification will be sent to the external system.

  This makes it easier to implement the logic of sending notifications when certain filters are met.

- The ability to send notifications about the status of tasks and subtasks to Telegram has been added.

  Sending notifications to Telegram is implemented by adding a new type of callback `telegram` to the "notification_policy" of each task.

  To send notifications you should specify `chat_id` and `token` parameters.

Sending notifications to Telegram can also be configured when creating a schedule for tasks.

- In the containers of the Events and Licenses services, the Python version has been updated to 3.12.

  Support for older versions of Python has been discontinued.

- The "video" > "orientation" > "angle" parameter has been added to the "video analytics (beta)" query parameters, which is responsible for the ability to rotate the video by 90, 180 or 270 degrees before processing it.

- The ability to use a more secure algorithm to generate JWT tokens has been added.

  By default, Accounts service uses HS256 algorithm to sign JWT tokens. If necessary, you can use asymmetric cryptographic encryption using ES256 algorithm.

  For this purpose it is necessary to:

  – Generate ECDSA private key.
  – Encode the private key in Base64 format.
  – Set environment variables "ACCOUNTS_JWT_ECDSA_KEY" and "ACCOUNTS_JWT_ECDSA_-
    KEY_PASSWORD".

  Important: Switching the signature algorithm from HS256 to ES256 (or vice versa) has a significant impact on token validation. **All existing tokens signed with the previous algorithm will become invalid** after the change. This is because the token signature verification mechanism expects the structure and cryptographic basis of the token to match the newly specified algorithm.

  See "JWT token algorithms" in the administrator manual.

- The ability to encrypt passwords and tokens passed in the Estimator task and in the "notification_-
  policy" policy has been added.

  To do this, pass custom values to `FERNET_PASSPHRASE` and `SALT` environment variables when starting the Tasks service container.

  – `FERNET_PASSPHRASE` — Password or key used to encrypt data using the Fernet algorithm.
  – `SALT` — Random string added to the password before it is hashed.

  Important: When the container is started with the above environment variables specified, the old passwords and tokens will no longer work. You must specify the environment variables in the Tasks database migration command, and then you must start a new Tasks container with the environment variables specified to be able to use encryption when creating new objects.

  See "Additional protection for passwords and tokens" in the administrator manual for details.

- A section "FACE_QUALITY_DEFAULT_THRESHOLDS" of the Remote SDK service has been added, containing standard thresholds for the "face_quality" check group, "detect_policy", resources "/handlers" and "/verifiers".

  There are two types of standard thresholds in this section of parameters:

- Thresholds from the "threshold" field, which determine at what value LUNA PLATFORM will consider the check to have passed. These thresholds are similar to the thresholds in the "face_quality" section of the corresponding requests. Setting thresholds in requests overrides the standard thresholds set in the "FACE_QUALITY_DEFAULT_THRESHOLDS" section.
- Thresholds from the "config" field that determine how to interpret the response of Natural Light, Fish Eye, Color Type and Red Eyes estimators. Previously, these thresholds were not available.

An example of configuring thresholds for a Red Eyes estimator:

```
"red_eye": {
    "threshold": 0,
    "config": {
        "estimator": {
            "threshold": {
                "min": 0.5
}    }    }    }
```

Here:

- "threshold" field defines the value "0" or "1" at which LUNA PLATFORM will consider the check to have passed. Similar to the "threshold" parameter in "face_quality" requests.
- "config" > "estimator" > "threshold" section defines the response of the Red Eyes estimator from "0" to "1".

- Checking the availability of the custom Docker registry and Lambda base images during service startup has been added.

- No additional permissions are now required to query the `/lambdas/{lambda_id}/proxy` resource of the Lambda service.

LP fixed errors

- Fixed incorrect work of Cached Matcher with 64 version of neural network for face descriptors extraction.

  Cached Matcher did not cache descriptors of 64 version and performed matching using data from database.

- Fixed incorrect format of the "url" field for faces and attributes generated by the "generate events" request.

  For example, the address of a face location could be returned as `'url': 'http://127.0.0.1:5030/faces/359c8c12-b67d-4645-96dc-d8f1627f360f'`.

  Correct address: `'url': 'http://127.0.0.1:5030/3/faces/426542d6-5509-4e5b-8a01-e2abd5c0a8c6'`

- The "stream events (beta)" request now generates the value of the `image_origin` field based on detections if no `image_origin` parameter is specified in the body of the request, but a face or body sample is specified.

LUNA PLATFORM v.5.62.3

LP changes

- SDK has been updated to version 5.19.1.

  This version of LUNA PLATFORM:

  - 64th neural network model for face descriptor extraction is supported.
  - Default model was changed from 59th to 62nd (setting "DEFAULT_FACE_DESCRIPTOR_VERSION" of Remote SDK service).

  When migrating the Configurator service settings of the latest version, the "DEFAULT_FACE_DESCRIPTOR_VERSION" setting **will not be automatically updated**. If you want to use the 62nd version of the neural network, you must manually change the value of the "DEFAULT_FACE_DESCRIPTOR_VERSION" setting by first re-extracting the existing descriptors with the 62 model using the "Additional extraction" task.

- Helm charts for deploying LUNA PLATFORM in a Kubernetes cluster have been added to the distribution package.

  The administrator must have a Kubernetes cluster deployed and configured to use Helm charts. It is assumed that the user's Kubernetes cluster:

  - PostgreSQL/Oracle DBMS and InfluxDB and Redis databases are running.
  - There is access to S3-like object storage for storing buckets.

  See the example commands for deploying LUNA PLATFORM in a Kubernetes cluster using Helm charts in the new manual (document "LP_Kubernetes_Helm_Deployment_Example" in the distribution package) or at online documentation.

- A user interface has been added for LUNA PLATFORM, implemented as a web interface for the API service.

  The user interface is available at `<host:5000/ui>`.

  See the description of the user interface in the new manual (document "LP_User_Interface_Manual" in the distribution package) or at online documentation.

- In the "videosdk" request, the ability to track people in a video and estimate their attributes both in the context of the entire video and on individual frames has been added.

  For people tracking video analytics, the key concept is a track. A track is an object containing information about the position of one person in a sequence of frames. A track is created when the

detection of a face, body or body with a face (depends on the detector type in the "detector_type" parameter) appears on the last frame of the specified number of consecutive frames ("probe_count" parameter). There can be as many tracks as the number of people detected on the frame. Along with a track, a video analytics event is started.

Note: Currently there can be only one event for one track. In future releases, it will be possible to have multiple events for a track, which can be sent without waiting for the track to end.

For example, if "probe_count" = "3" and there are detections on two consecutive frames and no detections on the third frame, the track and event will not start. If at any moment of tracking a person there were detections on two consecutive frames and no detections on the third frame, the track and event will end.

During tracking of a person it is possible to estimate attributes of faces, bodies, images (list of estimations is set in the "targets" parameter), and also to get the image with the best detection. The estimation can be performed by one best image from the track or by several (parameter "face_samples"/"body_samples" > "count"). If necessary, you can filter the best shots by "score", "head_pitch", "head_roll", "head_yaw" parameters. All scores set in the "targets" field depend on the type of detector used.

If necessary, you can enable Re-identification of the body (ReID). The ReID feature is used to improve tracking accuracy by combining different tracks of the same person.

For this type of video analytics, you can also configure a rectangular region of interest (parameter "roi") on the frame (coordinates "x", "y", width, height, units - per cent or coordinates). The principle of ROI region of interest is similar to FaceStream.

Each people count video analytics event contains the following information:

- "event_id" — Event ID.
- "track_id" — Track ID.
- "video_segment" — Information about the video segment (start time of the event, end time of the event, first frame number, last frame number).
- "frames_estimations" — Array with information about estimates on each frame, containing the following information:
  * Frame number.
  * Time corresponding to the frame.
  * Bbox coordinates and score.
- "aggregated_estimations" — Result of the score specified in the "targets" parameter in the request body.
- "track_count" - Number of tracks.

See OpenAPI specification for details.

• The ability to obtain people coordinates and images with the maximum number of people has been added to the people count video analytics.

For this purpose, a new parameter "targets" has been added to the request body, which takes the following values:

- "coordinates" - Calculation of people's coordinates.
- "overview" - Obtaining an image with the maximum number of people.

It is possible to set both values or neither. If no values are specified, a basic analysis will be performed, containing only information about the video recording, the number of people and frames associated with them.

The coordinates of people are returned in the response body in the "events" > "aggregated_-estimations" > "overview" > "people_coordinates" field.

- The structure of the "generate stream events (beta)" request has been reworked.

The "sources" > "source" > "face_bounding_boxes" and "body_bounding_boxes" fields (available when "source_type" = "raw_image") have been renamed to "face_detection_data" and "body_-detection_data" respectively and reworked as follows:

- "height", "width", "x", "y" parameters moved to "face_detection_data"/"body_detection_-data" > "bounding_box" fields.
- "face_detection_data"/"body_detection_data" fields now contain the new optional fields "score", "angles" (contains pitch, yaw and roll parameters) and the mandatory "bounding_-box" field (see above).
- Added the ability to specify custom keys with values to the "face_detection_data"/"body_-detection_data" fields.

The "sources" > "source" > "face" field (available when "source_type" = "detections") now contains new optional fields "score" and "angles". Also added the ability to specify custom keys with values in the "face" field.

Also in the field "sources" > "source" added the ability to specify custom keys with values.

- The description of the "cron" field in the "create tasks schedule", "replace tasks schedule", "get tasks schedules" and "get tasks schedule" requests has been extended.
- Prometheus format request statistics is supplemented with the request execution time.

The data is presented in a histogram format, which enable you to analyze the distribution of request execution times and identify potential performance problems. Each time interval (bucket) in the histogram displays the number of requests whose execution time falls into the corresponding range.

See "Export metrics in Prometheus format" for details.

- The ability to set labels for running lambda using specific Kubernetes nodes using the "deploy_-parameters" > "selector" parameter has been added.

This enables for more granular control over lambda deployment, allowing administrators to specify the nodes on which lambda should be deployed based on their resource needs. By applying labels to Kubernetes nodes and using the "selector" parameter, administrators can manage resource allocation not only for individual lambdas, but also for groups of lambdas with similar resource requirements.

See the "create lambda", "put lambda", "get lambda", "get lambdas" requests.

- Notification sent to external system using callbacks mechanism has been extended with "result_-url" parameter containing external address to API service with task or subtask result.

- SDK version for Handlers-lambda has been updated from version 5.17.0 to version 5.19.0.

- The ability to specify the number of pods for lambda has been added.

The number of pods is set in the "pod_count" parameter of the "deploy_parameters" section of the "create lambda" request. You can specify from 1 to 32 pods.

You can also set CPU and memory limits for each lambda pod using the new parameters "cpu_limit", "ram_limit", "cpu_request", "ram_request" in the "deploy_parameters" > "resources" section.

In addition, the output format in the "get lambda status" and "get lambda status" requests has been updated. The logs are now returned as a dictionary, allowing information about multiple lambda pods to be displayed.

You can specify the number of pods and resources for an existing lambda using the "put lambda" request.

- Support for using Kubernetes namespaces has been added to the Lambda service to provide isolation of resource groups within a single cluster.

The namespace is specified in the "namespace" parameter of the "deploy_parameters" section of the "create lambda" request. See the namespace recommendations in the corresponding query parameter description.

You can set the namespace for an existing lambda using the "put lambda" request.

- The unused parameter "landmarks17_threshold" has been removed from the "LUNA_REMOTE_-SDK_BODY_DETECTOR_SETTINGS" section.

- A new `/metrics` resource has been added to lambda's created, allowing to collect and save metrics in Prometheus format as time series data that can be used to track lambda behavior.

See "Export metrics in Prometheus format" for details.

- The structure of the documentation in the distribution package has been redesigned.

The system requirements for LUNA PLATFORM have been moved to a separate document in the scope of delivery.

The installation manuals have been moved to the "InstallationManuals" directory.

LP fixed errors

- Fixed the error that caused some parameters not to be displayed when the "show_all" filter was enabled in the "Service name" field on the "Settings" page of the Configurator service user interface.

  In addition, the "show_all" filter is now set as the default value.

- Fixed the error where in some cases task schedule processing would not restart properly after an environment (e.g. database) was temporarily unavailable.

- Fixed the error in Backport 3 service that caused the list of closed web socket connections to not be cleared when web sockets were unavailable or not initialized.

LUNA PLATFORM v.5.59.0

LP changes

- **Note:** In the next release, the default value of the neural network for extracting face descriptors will be updated from version 59 to version 62. Version 59 will be removed from the Remote SDK container.

- A Storages utility has been developed that enables you to check and/or prepare the environment for LUNA PLATFORM v.5.46.1 and higher services before launching them.

  The preparation of the environment is understood as follows:

  - Preparing buckets in InfluxDB for monitoring work.
  - Preparing buckets for the Image Store service, enabling you to store user data (images, metadata, archives, etc.).
  - Preparing the InfluxDB for collection of aggregated statistics by the Admin service (see "Requests and estimations statistics gathering" section in the administrator manual).
  - Preparing databases, adding VLMatch functionality, creating and managing migration scripts for LUNA PLATFORM databases (PostgreSQL or Oracle).
  - Performing migration/loading settings into the Configurator database.
  - Loading dump files into the Configurator service database.

  The main advantage of Storages over manual preparation of the environment is that the utility knows revisions of the Configurator service settings and migration scenarios for LP service databases, which greatly simplifies the process of updating and downgrading LUNA PLATFORM. In addition, using the utility, you can evaluate the current state of the LUNA PLATFORM environment and determine what exactly needs to be updated.

  The utility is supplied in a Docker container and can be used as a tool for preparing the LUNA PLATFORM environment deployed in Docker containers, the Docker Compose script, the

Kubernetes orchestration system, etc. The main task of the administrator is to indicate to the Storages utility the addresses of databases, buckets, etc.

A new document "Storages utility manual" has been added to the package and to the online documentation site, describing the principles of using the utility and examples of use. Two new LUNA PLATFORM deployment manuals have also been added -"Install manual using Storages" and "Upgrade manual using Storages". The documents are similar to regular installation and upgrade manuals, except that all environment preparation is done using the Storages utility.

Important: The Storages utility is in beta testing. See detailed information about the utility in the Storages utility manual.

- SDK has been updated to version 5.18.0.

- The ability to enable sending notifications about the status of subtasks ("subtask_callbacks" section) has been added to the "notification_policy" policy of each task.

  The subtask state contains the following information:

  - Subtask ID
  - Subtask status ("in_progress", "failed", "cancelled", "done")
  - Number of completed subtasks

  In responses to the "get task notification policy" and "replace task notification policy" requests also contain the information described above.

  Notifications about the status of subtasks are also available when using the "notification_policy" in the schedule.

- A new resource `/metrics` has been added to all LUNA PLATFORM services, enabling you to collect and save metrics in the Prometheus format as time series data that can be used to track service behavior.

  By default, metrics collection is disabled. Metrics collection is enabled in the "enabled" parameter in the "LUNA_SERVICE_METRICS" section. If metrics collection is disabled, then a request to the `/metrics` resource will return an error with the code "12049" and the message "Forbidden, Resource is disabled".

  There are two types of metrics available:

  - `request_count_total` — Total number of requests.
  - `errors_count_total` — Total number of errors.

  Each of them has at least two labels for sorting:

  - `status_code` (or `error_code` for error metrics)
  - `path` — Path consisting of the request method and the endpoint route.

  If necessary, you can add custom label types by specifying the `label_name=label_value` pair in the "extra_labels" parameter in the "LUNA_SERVICE_METRICS" section.

See <u>"Export metrics in Prometheus format"</u> for details.

- The ability to obtain the X and Y coordinates of people when estimating the number of people has been added.

  The ability to obtain coordinates has been added to <u>"sdk"</u> request and the "detect_policy" of the <u>"create handler"</u> request. To do this, you need to enable the "people_count_coordinates" and "people_coordinates" parameters respectively.

  The generated event structure has also been updated.

- Examples for the request <u>"generate stream events (beta)"</u> have been added to the OpenAPI specification.

- The ability to run lambda GPU has been added.

  To use the GPU, you must enable the "deploy_parameters" > "enable_gpu" parameter in the <u>lambda generation</u> request.

  There are some requirements and limitations when using GPUs in Kubernetes. See <u>"Create GPU-enabled lambda"</u> administrator manual.

LP fixed errors

- Fixed lack of support for the `OPTIONS` method for the resource `/plugins`.

- Fixed the "Internal server error" that occurred when executing the Garbage collection task with the `events` filter enabled and the `remove_image_origins` parameter, when the event did not contain the source image (`None` value).

- Fixed the `value too long for type character varying(512)` error that occurred when downgrading Configurator settings from revision `d718e5954caf`.

  Changes made in revision `d718e5954caf` led to an increase in the length of the token settings value string from 512 characters to infinity. When downgrading from revisions where the length value in the corresponding setting was already greater than 512 characters, the error `value too long for type character varying(512)` was returned.

  Now, when downgrading from revision `d718e5954caf`, a setting value exceeding 512 characters will be deleted. You must re-set the token value for the appropriate setting.

- Fixed the error `Key (name)=(<setting>)is not present in table "limitation"` that occurred when downgrading Configurator settings from revision `dd2641137b42`.

  Changes made in revision `dd2641137b42` led to the mandatory binding of each setting to a specific template. For this purpose, a new table `limitation` was added containing restrictions. The error occurred when performing a downgrade from revisions where the Configurator database already contained data about restrictions in the `setting` table and it was impossible to establish a connection between the `limitation` table being deleted and the existing `setting` table.

Now, when performing a downgrade from revision `d718e5954caf`, all entries from the `setting` table will be deleted when the `limitation` table is deleted.

LUNA PLATFORM v.5.58.0

LP changes

- **Note:** After one release, the default value of the neural network for extracting face descriptors will be updated from version 59 to version 62. Support for version 52 of the neural network will also be discontinued.

- Support for using a token (Bearer authentication) via cookies has been added to simplify the authentication process in web browsers.

  To save Cookies, you need to perform the "set login cookie" request, specifying the token as authorization. The cookies are then sent back to the user's browser. Subsequent requests sent by the user's browser automatically include these Cookies, allowing the server to recognize the user's authorization, without having to explicitly send a token in each request.

  If necessary, you can clear Cookies by requesting "clear login cookie".

- A new environment variable "LUNA_SKIP_CHECK_CONNECTION" has been added to all services.

  The variable enables you to disable connection check (value "1") to all general services (Image Store, Faces, etc.), which is performed by default at the start of the service.

  Connection check is performed to exclude errors related to incorrect configuration of the LUNA PLATFORM, but it may slow down the process of lifting the container. In addition, in some cases, checking the connection can cause problems when lifting the container, especially if the connection is unstable.

  The environment variable can be passed using the `--env` argument when starting the container.

- Support for running services using the HTTPS protocol has been added.

  To utilize this feature, the following command line arguments need to be provided for the respective service:

  - `tls_cert` — Path to the SSL certificate.
  - `tls_key` — Path to the SSL private key.
  - `tls_key_pass` — Password for the SSL private key (optional).

  Example command: `python3 /srv/luna_<service>/run.py --tls_cert /srv/my_certificate.crt --tls_key /srv/my_private_key.key --tls_key_pass my_password`

  Note that the certificate and key must be mounted to the Docker container in the specified directories.

A list of all available arguments can be obtained using the following command: `python3 /srv/luna_<service>/run.py -h`

- The ability to specify the environment variable `--EXTEND_CMD` has been added to all LP services. This enables passing arguments to the service launch command that are not covered by existing environment variables.

  For example, you can explicitly set tagged settings when starting services: `--env=EXTEND_CMD="--INFLUX_MONITORING=TAG_1 --LUNA_EVENTS_DB=TAG_2"`

  See the "Service arguments" section in the installation manuals for details on environment variables and arguments.

- A new parameter "workers" has been added to the bodies of "create lambda" and "put lambda" requests, enabling you to allocate the number of workers to the specified Lambda instance.

- SDK version for Handlers-lambda has been updated from version 5.16.0 to version 5.17.0.

LP fixed errors

- In the descriptions of the "create handler", "validate handler policies" and "generate events" requests, the lack of information about the allowed length of characters in the "callbacks" policy parameters has been fixed.

- Fixed the error that caused the new value of a setting passed in an environment variable to not be updated if the setting was not in the Configurator service or configuration file.

- Fixed the "Internal server error" that occurred when trying to access resources `/1/buckets/{bucket}/objects/{object_id}` and `/1/buckets/{bucket}/objects` of the Image Store service, which are not supported methods.

  Correct errors are now returned.

- Fixed the error due to which, when creating an Estimator task with a non-existent handler, a corresponding entry was still created in the Handlers database.

LUNA PLATFORM v.5.57.0

LP changes

- A "notification_policy" has been added to all tasks, which enables you to send notifications about changes in the status of tasks using the callback mechanism.

  Callbacks enable you to send data to a third-party system at the specified URL.

  The parameters from the new policy are similar to the parameters set in the corresponding handler policy ("authorization", "content_type", "timeout", etc.).

  You can also set up notifications for tasks in the schedule settings.

If necessary, you can get information about the current status of the schedule or update it using the get task notification policy and replace task notification policy resources.

- In the "create token" request, the ability has been added to specify the "account" > "view" permission to be able to get the data of the account to which the token itself was issued.

- The "luna-handlers" client has been added for Tasks-lambda, which enables you to send user events to the Handlers service.

  See the detailed information about clients in the "Luna services clients" section of the development manual.

- The ability to create custom monitoring points for Lambda task instances has been added to the Lambda service.

  See detailed information about monitoring in the "Lambda monitoring" section of the development manual.

- An offline version of the website with online documentation has been added to the LUNA PLATFORM distribution package, replacing the previous HTML documents.

LP fixed errors

- Now, when executing "generate events" and "generate stream events (beta)" requests with samples, bounding boxes will not be sent to the Remote SDK service, because they are not used.

- The unused parameters `raw_event_detections_limit`, `raw_event_arrays_limit` and `result_candidate_limit` have been removed from the `LUNA_REMOTE_SDK_LIMITS` section. The only parameter left is `received_images_limit`.

- Fixed the error where the Python Matcher Proxy service was using 100% CPU when the "indexed_-matcher" plugin was enabled (when using LIM).

- Fixed the error due to which, if the location query parameters were not filled in in the "generate events", "save event", "generate stream events (beta)" requests, the values "null" were still written to the Events database.

  Now, if no location parameters are specified, then no record will be created in the "location" table.

LUNA PLATFORM v.5.56.0

Changes

- The ability to perform video analytics using the new resource "videosdk" has been added.

  Important: Currently, the resource is in beta testing status. Input and output schemes may be changed in future releases without backward compatibility support.

  Video analytics is a set of functions that process frame by frame and assess valuable data.

To perform video analytics, an external link to a video file not exceeding 1 GB in size must be specified.

During video analytics, a certain number of events are generated according to some rules, where each event has a start and end. Events are recorded in the response body and contain specific information about the particular video analytics. The response body also contains basic metadata about the video (number of frames, frame rate, video duration). The obtained information is only available in the response body and is not saved anywhere.

Important: An event in video analytics is in no way related to events generated by handlers.

In the "LUNA_REMOTE_SDK_VIDEO_SETTINGS" section of the Remote SDK service, you can configure video processing settings, such as the number of decoder working processes, temporary directory for storing videos, etc.

Note: In the current version, video decoding is only done on the CPU. GPU decoding will be added in future releases.

Currently, only people counting video analytics is supported.

People counting video analytics

The event of people counting video analytics starts when, on the last frame of the specified consecutive frames (parameter "probe_count"), the specified number of people (parameter "people_count_threshold") appears. For example, if the minimum number of people is 10, and the number of consecutive frames is 3, the event will start if there are 10 people in 3 frames, starting from the 3rd frame. If there are 10 people in 2 frames, and 9 people in the third frame, the event will not start.

By default, every 10 frames are processed. If necessary, you can adjust the frame processing frequency using the "rate" parameter (for example, process every 3 seconds or every 3 frames).

For this type of video analytics, you can also configure a rectangular region of interest (parameter "roi") on the frame (coordinates "x", "y", width, height, units of measurement — percentages or coordinates). The operating principle of ROI is similar to FaceStream.

Each event in people counting video analytics contains the following information:

- "event_id" — Event ID.
- "max_people_count" — Maximum number of people detected in the processed video segment.
- "video_segment" — Information about the video segment (start time of the event, end time of the event, first frame number, last frame number).
- "frames_estimations" — Array with information about estimates on each frame, containing the following information:
    * Frame number.
    * Time corresponding to the frame.

* Number of people.

See detailed information about the new resource in the OpenAPI specification.

- The SDK has been updated to version 5.17.0.

  In this version of the SDK, the LivenessOneShotRGB estimator has been updated.

- A built-in plugin has been added to the API service, which enables you to make some requests to the LUNA Streams service through the API service.

  Before using the plugin, you must enable and configure it (optional).

  Plugin "luna-streams.py" located in the API service container along the path `/srv/luna_api/plugins` and enabled using the setting "LUNA_API_ACTIVE_PLUGINS".

  The plugin can be configured in the "LUNA_API_PLUGINS_SETTINGS" setting. In the setting, you can set the address and API version of the launched LUNA Streams service, as well as connection timeouts.

  Example of the content of the "LUNA_API_PLUGINS_SETTINGS" setting for the "luna-streams.py" plugin:

```
{
    "luna-streams": {
        "luna-streams-address": {
            "origin": "http://127.0.0.1:5160/1",
            "api_version": 1
        },
        "luna-streams-timeouts": {
            "request": 60,
            "connect": 20,
            "sock_connect": 10,
            "sock_read": 60
        }
    }
}
```

If necessary, you can differentiate access rights for requests to LUNA Streams using token permissions. To do this, you need to create a new or update an existing token by passing a custom "streams" key with permissions for the token.

> To proxy requests from the LUNA API service to the LUNA Streams service, you need to specify a permission named "streams". Other custom key names will not work.

You can set the following permissions:

- "creation" (POST requests)

- "view" (GET requests)
- "modification" (PUT requests and PATCH requests)
- "deletion" (DELETE requests)

See the "Plugins" section of the developer manual to get a list of possible requests.

The information will also be added to the FaceStream administrator manual in the next release.

- A new "get list of plugins" request has been added to all LUNA PLATFORM services, which enables you to get a list of imported plugins and their status.

- A new type of Tasks has been added to the Lambda service-lambda, designed to implement additional custom long task types.

  Examples of possible functionality:

  - Unlink faces from lists if faces do not similar to the specified face.
  - Remove duplicates from list.
  - Recursively find events similar to the specified photo.

  Lambda-tasks are created in the same way as the other types using the "create lambda" request.

  After creating a lambda, you must perform the "lambda task" request to create a Lambda task. The results of the task/subtask can be obtained using standard requests from the Tasks service.

  The Lambda task is created according to the general task creation process, **except that** the Lambda service is used instead of the Tasks worker.

  See the "Tasks-lambda" section of the administrator manual for additional basic information.

  See the detailed information and code examples in the Tasks service developer manual.

  It is also possible to create a schedule for Lambda tasks.

- The error description for some cases of validation of the "generate events" request body when using the "multipart/form-data" scheme has been improved.

Fixed errors

- Fixed generation of incorrect token expiration time when creating/updating it in cases when the server's local time is set ("STORAGE_TIME" = "LOCAL").

- Fixed the error that caused excessive opening of database connections when creating a Linker task with a large number of people, which led to the task hanging.

- Fixed the error due to which it was impossible to perform a "create new events" request to the Events service with non-null "location" fields.

  An attempt to execute a similar request ended with an error with the code "12047" and the description "Failed to validate input msgpack. Path: '[0].location.<location_filed>', message: '[0].location.<location_filed> must be string'".

- Fixed the error `'a coroutine was expected, got None` that occurred when using the event dispatch plugin with a large number of arguments.

  See the "Plugins" section of the Handlers service developer manual for details and an example of an event sending plugin.

- Fixed the error of counting the current time ("now-time") for the "create_time__lt" and "create_-time__gte" filters specified in the "match_policy" > "candidates" policy when creating a handler.

  Previously, the current time was counted from the time when the handler was created. Now the current time will be counted from the time when the event was generated.

- Fixed the license check for the ISO feature.

  Previously, if the ISO licensed feature was disabled, requests to perform any estimations ended with an error with the code "11055" and the description "License problem: ISO feature is disabled".

- Fixed processing of detections in the generate stream events (beta) request.

  Now, if samples are transferred with the type "sources" > "source_type" > "detections", then detection and extraction of these samples will be performed.

- Fixed the error in the absence of the "unknown" field in the description of the liveness request in the OpenAPI specification of the Backport3 service.

LUNA PLATFORM v.5.54.0

Changes

- The query parameter "grant_all_permissions" has been added to the "create token" and "replace token" requests, which enables you to create a token with all permissions or add all permissions are already applied to an existing token.

- A new request "get accounts" has been added.

  You can specify the "targets" query parameter in this request to filter accounts by the fields "account_id", "login", "account_type", "description", "create_time", and "last_update_time".

  Filtering by the fields described above has also been added to the "get account" request.

- In the Admin and Tasks services, when performing the Additional extraction task, the "options" field is no longer mandatory if the "extraction_target" parameter is set to "basic_attributes".

- A new request "get list of plugins" has been added to the Events service, which enables you to get a list of imported plugins and their status.

  In the next release, this request will be added to all other services.

- The "Use tagged settings" section has been added to the administrator manual , describing the launch of services with settings other than the default ones.

  The "Service arguments" section has also been added to the installation manual, describing the main arguments of the services and how they are transferred.

Fixed errors

- Fixed display of "Path" when validating some request bodies with incorrect JSON structure.

- Fixed an issue where requests to the "/features" resource would fail with an "Internal server error" when the license did not contain any of the licensed features.

- Fixed incorrect processing of a sample or a binary image transferred as a source image in the generate stream events (beta) request.

- Fixed an error related to the Python Matcher service not checking for the presence of the VLMatch function when connecting to the Faces and Events databases.

LUNA PLATFORM v.5.53.0

Changes

- The VisionLabs image for PostgreSQL has been updated from version 12 to version 16.

  If you previously used this image, then to migrate to the new version you must perform the migration yourself according to official documentation. If necessary, you can continue to use PostgreSQL 12 by specifying the "postgis-vlmatch:12" image in the container launch command.

  Mounting PostgreSQL 12 data from the "/var/lib/luna/postgres" directory into a PostgreSQL 16 container will result in an error.

  The section "Migrate PostgreSQL 12 to PostgreSQL 16" has been added to the upgrade manual, containing a reminder about the need for migration.

- The ability to detect facial spoofing using DeepFake technology in photo images has been added.

  The "estimate_deepfake" parameter has been added to the requests "create handler", "create verifier" and "sdk".

  The "deepfake" field has been added to the event structure. This field can be used as a value for the "target" field or as a filter to receive an event using GET requests.

  Deepfake estimation may return the following results:

  - "prediction" = "fake" — The person is not real.
  - "prediction" = "real" — The person is real.
  - "score" = [0…1] — The degree of reliability of the estimation.

In requests "create handler" and "create verifier" it is possible to set the "real_threshold" and "mode". The "sdk" request will use the default values of these parameters without the possibility of explicitly specifying them.

Using the "real_threshold", you can set a value in the range [0…1], below which the system will consider that the person is not real.

Two operating modes are available:

– "mode" = "1" — Simplified operating mode.
– "mode" = "2" (default) — Operating mode using an additional neural network model for preliminary estimation. If the result of the preliminary estimation determines that the person is fake, then the result "score" = "0" and "prediction" = "fake" will be returned in the response body.

The following image requirements must also be met to perform the estimation:

– head pose: "pitch" = [-20…20]
– head pose: "yaw" = [-30…30]
– face width: "face_width" > 150

The "deepfake" filter can also be used in matching requests.

The "deepfake_states" parameter has also been added to the handler and verifier, which allows filtering events by the expected result of the Deepfake estimation.

The "deepfake" field has also been added to the filters for Clustering, Exporter, Cross-matching and Linker tasks. For the Exporter and Reporter tasks, the field is supported as a column.

The "deepfake" field has also been supported as a filter in the "ws handshake" request.

• A new request "get list of plugins" has been added to the API service, which enables you to get a list of imported plugins and their status.

• A new "callbacks" policy has been added to the "storage_policy" of the handler and verifier, with which you can send generated events (notifications) to the third-party system at the specified URL.

In fact, callbacks are analogous to sending notifications via web sockets, but with the key difference that they use the principles of HTTP webhooks, which provides a more flexible and customizable mechanism for sending notifications to third-party systems. You can configure the protocol type, external system address, request parameters and authorization data.

Events sent using the "callbacks" field have a format corresponding to the format of the "generate events" request.

For more information, see the requests "create handler" and "create verifier".

• The Handlers-lambda input data structure has been updated.

This means that all existing lambdas must be revised and recreated according to the new structure.

All examples have also been updated.

Example of the old input data structure: `{"body": getImage("empty.jpeg"), "filename" : "empty.jpeg", "source_type": 0}`

Example of a new input data structure: `{"source": {"body": getImage("empty.jpeg")}, "filename": "empty.jpeg", "source_type": "raw_image"}`

- Now the Kaniko executor image (the image for building lambda) should be in the registry specified in the "LAMBDA_REGISTRY" setting.

  Instructions for transferring the Kaniko executor image from the VisionLabs registry to the user registry have been added to the installation manual.

- New connection settings with Redis and Redis Sentinel have been added.

  The "user" parameter has been added to the settings groups "REDIS_DB_ADDRESS", "TASKS_-REDIS_DB_ADDRESS", "LUNA_ATTRIBUTES_DB", "BACKPORT3_EVENTS_DB_ADDRESS".

  The "user" and "password" parameters have been added to the "sentinel" section of the above settings.

- Guardant has been updated from version 3.15 to 3.21.

LUNA PLATFORM v.5.51.6

Changes

- Support for ECS logging format has been added.

  To use the new format, you need to set the value "ecs" in the "format" setting of the "LUNA_service_-LOGGER" section.

  When using the "ecs" value, the following fields will be used:

  - "http.response.status_code" — contains the HTTP response status code (e.g., 200, 404, 500, etc.).
  - "http.response.execution_time" — contains information about the time taken to execute the request and receive the response.
  - "http.request.method" — contains the HTTP request method (GET, POST, PUT, etc.).
  - "url.path" — contains the path in the request's URL.
  - "error.code" — contains the error code if the request results in an error.

- The ability to create a schedule for Reporter task has been added.

- A new parameter "base_image" has been added to the requests "create lambda" and "update lambda", allowing you to explicitly specify the name of the base image for building a Docker container.

Previously, it was possible to use only two images that were previously transferred to the user registry — "lpa-lambda-base" (basic functionality) and "lpa-lambda-base-fsdk" (basic functionality and functionality for using FSDK).

A custom lambda image is intended for complex lambdas that lack the capabilities implemented in the "lpa-lambda-base" and "lpa-lambda-base-fsdk" base images (for example, multi-stage creation of a Docker container or the inclusion of large libraries or data in a Docker container).

For a custom lambda image, the following conditions must be met:

- the image must be based on the "lpa-lambda-base" or lpa-lambda-base-fsdk images"
- the image must not remove or change any installed dependencies (python3, gcc, etc.)
- the image must be in the same registry as other base lambda images.

- The ability to create custom monitoring points for lambda units has been added to the Lambda service.

  For example, you can send data about how long it took to upload or process images.

  For more information, see the "Monitoring" section of the Lambda service development manual.

- The logic of processing bounding boxes in the generate stream events (beta) request has been updated.

  Now the source body image (the "body" parameter) is not required to save the bounding boxes of the face/body (field "events" > "detections" > "samples" > "face"/"body" > "detection" > "rect").

- In the LUNA PLATFORM license activation manual, a section "Vendor change" has been added with instructions on changing the HASP vendor to Guardant and vice versa.

Fixed errors

- Removed the unused parameter "FETCH_EXTERNAL_IMAGE_TIMEOUTS" from the API service settings.

- Fixed the "Internal server error" error occurring when performing a "get system info" request when the required measurements were missing in the Influx database.

LUNA PLATFORM v.5.51.4

Changes

- The ability to create a schedule for Estimator, Clustering, Exporter, Cross-matching, Roc-curve calculating and Additional extraction tasks has been added.

  When creating an Estimator task schedule, it is not possible to specify a ZIP archive as an image source.

- A new series "Matching-Process" has been added to the monitoring of the Python Matcher service, containing information on the matching — time of matching, time of receiving the candidate from the database, etc.

  See the Python Matcher service developer manual for a list of all the fields and tags in the new series.

- Request and error monitoring for the Lambda service has been added.

  See additional information in the Lambda service developer manual.

Fixed errors

- Fixed the error where requests to the "/features" resource would fail with an "Internal server error" when using a perpetual license.

- Fixed the error that caused the websocket connection to fail in some cases.

- Fixed the error in the OpenAPI specification where redundant candidate types were displayed in the request bodies for "matching faces" and "human body matching".

- Fixed the error that led to the "Internal server error" if an incorrect msgpack was passed in the request body "generate stream events" (request in beta testing state).

- Fixed the error in which an error with the code "12022" with the description "message: 'filters.origin must be one of ['faces', 'events', 'attributes']" was returned in the response to the "human body matching" request containing various types of candidates and references in filters.

  Now the error does not contain a list of all types, but a specific type that should be specified in the filter.

LUNA PLATFORM v.5.51.0

Changes

- The SDK has been updated to version 5.16.0.

  Support for the new 62 neural network model for face descriptor extraction has been added to LUNA PLATFORM.

- A new Lambda service has been added, intended to work with user modules that mimic the functionality of a separate service. This functionality is currently in beta testing.

  Full-fledged work with the Lambda service is possible when deploying LUNA PLATFORM services in Kubernetes. To use it, you must independently deploy LUNA PLATFORM services in Kubernetes or consult VisionLabs specialists. If necessary, you can use Minikube for local development and testing, thus providing a Kubernetes-like environment without the need to manage a full production Kubernetes cluster.

The service enables you to write and use your own handler or write an external service that will closely interact with LUNA PLATFORM and have several functions typical of LP services (such as logging, automatic reloading of configurations, etc.). Users just need to write the code and send it to the Lambda service, after which they can use their module without deploying additional containers, etc.

The service creates a Docker image based on a ZIP archive with developer code and then runs it on a Kubernetes cluster. The custom module running on the Kubernetes cluster is called **lambda**.

To work with the Lambda service, you need a **separate license feature**. In requests "get platform features", "get system info" and "get license", "lambdas" fields have been added, displaying the status of the license feature.

A new "lambda" permission has been added to the token, regulating the "creation", "view", "modification", and "deletion" rights for lambda.

To work with the Lambda service, the following environment requirements are required:

- Availability of running Licenses and Configurator services*.
- Availability of an S3 bucket for storing archives with developer code.
- Availability of a Docker registry for storing images.
- Availability of a Kubernetes cluster.

* during its operation, lambda will additionally interact with some LUNA PLATFORM services. The list of services depends on the type of lambda.

In addition to the above requirements for the environment, it is necessary to comply with the requirements for writing code and archiving, as well as properly configuring the service. See additional information in the documentation.

Lambda is created using the "create lambda" request, which specifies the archive, the name and its type (see below).

Lambda can be of two types:

Handlers-lambda

This type is intended to replace the functionality of the classic handler. The Lambda handler can be used in two cases:

- As a custom handler that has its response scheme, which may differ from the response of classic handlers and cannot be properly used in other LUNA PLATFORM services (for example, {"similarity": 0.123});
- As a custom handler that mimics the response of a classic handler (for example, { "images ": [{ ... }], "events": [{ ... }] ... }). Such a handler must match the response scheme of the event generation request and process the data correctly so that other services can use it.

Due to the addition of Handlers-lambda, the response body schema of the "create handler" request has been upgraded. A new parameter "handler_type" has been added to the request body, which accepts three values — "0" (static handler), "1" (dynamic handler) and "2" (lambda handler). To use the value "2", you also need to specify the "lambda_id" (see the logic below). The "is_dynamic" parameter is considered deprecated and will be ignored when using the "handler_type" parameter.

There are two ways to interact with Handlers-lambda:

1. Using the requests "generate events" or "estimator task". To use these requests, you must first create a handler by specifying "handler_type" = "2" and "lambda_id" obtained at the lambda creation stage. In response to the event generation, a custom result will be output, the format of which either matches the format of the classic event and enables it to be used by LP services in the future, or does not match.

2. Using the "proxy post request to lambda" request when it is not intended to mimic the response of a classic handler, since LP services assume a strict format for responding to the event generation request.

For example, with Handlers-lambda you can implement the logic for sending, extracting, and matching two descriptors in a single request. In the response, only the similarity of candidates can be given without unnecessary information. In the classic scenario of using the LUNA PLATFORM, the user cannot execute this scenario and is forced to write logic on the side of the external system. See the code examples in the developer manual.

Standalone-lambda

This type is intended to implement independent functionality to perform close integration with the LUNA PLATFORM. To work with this type, a request "proxy post request to lambda" with its own request and response scheme is used. This type is intended for a narrow target audience and is quite difficult to implement.

Using Standalone-lambda, you can write a service that implements the recording of a video stream to a file and saves it to the Image Store service for subsequent processing by the FaceStream application.

Dashboard creation for the Lambda service has also been added to the LUNA Dashboards service.

See lambda creation and processing sequence diagrams in the "Lambda diagrams" section.

See additional information for a basic introduction to the functionality of the Lambda service in the "Lambda service" section of the administrator manual.

See more information for a deeper dive in Lambda service developer documentation. The manual provides a set of steps to quickly get started with the service.

- The response to the "get system info" request to the Admin service has been expanded with additional information on statistics on the use of estimators and images.

The "estimators_performance_stats" field has been added, displaying the month, the name of the estimator, the average time of the estimate and the average size of the batch for each estimator. Example:

```
"estimators_performance_stats": [
    {
        "month": "2021-09",
        "name": "body_descriptor",
        "execution_time": 0.029135203011156546,
        "batch_size": 1.0952380952380951
    }
]
```

The "image_processing_stats" field has been added, displaying the month, the average decoding time of the image, the number of images by their size and the number of images by face height. Example:

```
"image_processing_stats": [
    {
        "month": "2021-09",
        "image_load_time": 0.006479793069339647,
        "image_size": {
            "w_1000_h_1050": 4,
            "w_1000_h_800": 212,
        },
        "face_detection_size": {
            "h_100": 2,
            "h_180": 197
        }
    }
]
```

Here:

- "w_1000_h_800": 212 – 212 images with a width of 1000 pixels and a height of 800 pixels
- "h_180": 197 – 197 images with a face height of 180 pixels

Images are rounded to the nearest 50 pixels. For example, if an image has a width of 105 pixels, it will be rounded to 150 pixels. All images with the same rounded values will be in the same bucket.

Note: To use the new statistics, run the `python influx2_cli.py create_usage_test --luna-config http://127.0.0.1:5070/1` after starting the Admin service (see the installation manual).

- The Schedules tab has been added to the Admin service user interface, which enables you to manage the task schedule and create a schedule for a Garbage collection task.

  The tab displays all created task schedules and all relevant information (status, ID, Cron string, etc.). In the tab, you can also create, delete and edit schedules, as well as manage delayed start (pause and start stopped schedules).

  For more information, see the section "Admin service user interface".

Fixed errors

- Fixed the absence of the following sorting parameters for accounts and tokens:

  - filters "create_time" and "create_time__gte" in the request "get tokens";
  - fields "create_time" and "last_update_time" in the response bodies of the requests "get account", "get tokens", "get token".

- Fixed the error in which during the launch of the Admin service, the connection and healthcheck to the Remote SDK service were not checked.

- Fixed the error where during the migration of the database of the Handlers service to version v.3.0.0, the field "detect_policy" > "estimate_people_count" was incorrectly updated in existing handlers.

  As a result, after the update, when trying to use handlers with such a field, an error could occur with the code "12022" and the description "Failed to validate input json. Path: 'policies.detect_-policy.estimate_people_count', message: 'value is not a valid dict'".

LUNA PLATFORM v.5.49.1

Changes

- The ability to schedule Garbage collection and Linker tasks has been added.

  Using a schedule, you can flexibly manage the start time of tasks. For example, you can set up a regular schedule to clear events older than one month every Friday night.

  The schedule is created using the request "create tasks schedule" to the API service, which specifies the contents of the task being created and the time interval for its launch. To specify the time interval, Cron expressions are used.

  If necessary, you can create a delayed schedule, and then activate it using the "action" = "start" parameter of the "patch tasks schedule" request. Similarly, you can stop the scheduled task using "action" = "stop". To delete a schedule, you can use the "delete tasks schedule" request.

  Permissions to work with schedules are specified in the token with the "task" permission. This means that if the user has permission to work with tasks, then he will also be able to use the schedule.

An example of a CURL request has been added to the installation manual and launch manual using Docker Compose to launch a daily schedule for the Garbage collection task for events older than 30 days with the removal of samples and source images.

A new table "schedule" has been added to the Tasks database, containing all information about the task execution schedule. You can get information from the database about the created schedule using the requests "get tasks schedule" and "get tasks schedules".

See the "Running scheduled tasks" section of the administrator manual for details.

- Now if the request format is "application/msgpack" and the input data does not match the MessagePack format, then an error will be returned with the code "12047" and the description "Failed to validate input msgpack. Path: '{}', message: '{}'".

  Previously, the error code "12022" was returned with the description "Failed to validate input json. Path: '{}', message: '{}'".

- The "generate events" request now supports specifying custom metadata for an image passed using the "multipart/form-data" schema in the "image" field.

  Previously, specifying custom metadata was only available for source images ("image_origin" fields). Also fixed was a bug due to which the description of the "generate events" request did not display a description about specifying custom metadata for source images.

  If metadata is provided simultaneously in both fields, the value from the "image_origin" field will be used.

  Metadata is transmitted using headers of the form "X-Luna-Meta-<user_defined_key>:<user_-defined_value>", which are sent to the Image Store service when an image is saved during event generation.

- LUNA Dashboards service has been updated to version 0.0.8.

  This version adds the missing dashboard for the Remote SDK service.

  Also included in the package is an archive with the "grafana-piechart-panel" plugin, designed for manual installation of dashboards.

Fixed errors

- Fixed a number of the following errors when making requests to the Events service:

  – Fixed an error with status code 500 when requesting the resource "/events/stats" with the parameter "filters" > "value" of type "bool".

  – Fixed an error with status code 500 when requesting the resource "/events/stats" and using invalid operators in a filter that was set to "null".

  – Fixed an error with status code 500 when requesting the "/events" resource when the JSON request contained a number exceeding the "int64" limit.

- Fixed an error bug with incorrect filtering of event statistics by "null" using the "neq" and "nin" operators.

- The request body schema for the Estimator task for the "zip" source type in the OpenAPI API service specification has been updated.

- Fixed an error where loading API service settings values from environment variables did not work for:

  - nested setting keys, for example, "VL_SETTINGS.LUNA_SERVICE_NAME_DB.DB_SETTINGS.CONNECTION_ POOL_SIZE=10"
  - array type values, for example, "VL_SETTINGS.OTHER.LUNA_SERVICE_NAME_ACTIVE_- PLUGINS=[logger]"

- Fixed an error in the "save event" request of the OpenAPI specification of the API service, in which the parameter "detections" > "samples" > "face" > "detection" > " face_quality" > "checks" > "check_face_properties_request" > "max" was shown as required.

LUNA PLATFORM v.5.47.4

Changes

- Database settings `<service_name>_DB` of all services have been expanded with a new optional parameter "dsn", which specifies a DSN string that can contain various settings for managing the connection to the database, such as multiple hosts, authentication data, port, and others (settings depend on the type of database).

  Due to the implementation of a new setting, the classic settings for connecting to the database ("db_host", "db_port", "db_name", "db_user" and "db_password") in all settings of all services have become optional.

  If necessary, you can combine the DSN string and the classic settings, but the DSN string is a higher priority. You can partially fill in the DSN string (for example, "postgres01,postgres02/luna_- handlers"), and then the missing parameters will be filled in from the values of the parameters "db_host", "db_port", "db_name", "db_user" and "db_password".

  After updating to the new version of LUNA PLATFORM, the "dsn" parameter will not appear in the "Settings" tab in the Configurator. To use DSN, you must manually specify the appropriate parameter. Below is an example of specifying the "dsn" parameter in the "LUNA_FACES_DB" section:

```
{
"dsn": "luna:luna@postgres01:5432,postgres02:5432/luna_faces?some_option
    =some_value"
```

```
  "db_settings": {
      "connection_pool_size": 5
      }
  }
```

For more information, see the section "Connect to database using DSN" of the administrator manual.

- Now in response to the requests "get system info" and "get license" in the "expiration_time" field, the value will be displayed "perpetual" for a perpetual license.

- Now for the "Thin faces" matching plugin, you can configure the maximum size of the list to perform the matching. If the list exceeds the set number, then Python Matcher or LIM Indexed Matcher will be used for matching.

The maximum size of the list is set by specifying the environment variable "VL_SETTINGS.THIN_-FACE.MAX_LIST_LENGTH" when starting the Python Matcher service.

Example of specifying a variable when starting the Python Matcher service using Docker: `docker run \ -e VL_SETTINGS.THIN_FACE.MAX_LIST_LENGTH=100`.

Fixed errors

- Fixed the occurrence of error 31006 "Unexpected behavior of the {matcher_type} matcher: {plugin_error_description}" when performing a matching using a reference attribute and a matching plugin.

LUNA PLATFORM v.5.47.1

Changes

- The internal mechanism of interaction of the Tasks service with workers has been updated.

Now, instead of sending HTTP requests to workers, the Tasks service will interact with them using Redis.

A new section "TASKS_REDIS_DB_ADDRESS" has been added to the Tasks service settings, where:

  - "host" — Redis IP address
  - "port" — Redis port
  - "password" — password for authorization in Redis
  - "number" — the number of the Redis database (from 0 to 15). Each number corresponds to a separate database, which enables you to separate the data.

When upgrading to the current version of the LUNA PLATFORM, the values of the above settings will be filled in according to the "LUNA_ATTRIBUTES_DB" section of the Faces service. This will enable you to use the same Redis instance for the Tasks service that is used for the Faces service. If it is

necessary to separate the data of the Faces and Tasks services in Redis, you can specify user settings in the "TASKS_REDIS_DB_ADDRESS" section after launching the service (for example, specify the database number of the current Redis instance or specify the address of another Redis instance).

Also, the Tasks workers now do not have access to the Tasks database.

See the updated Tasks service sequence diagram in the "Tasks diagrams" section.

- In the containers of the Events and Licenses services, the Python version has been updated to 3.11.

Support for older versions of Python has been discontinued.

Fixed errors

- Fixed the error in Python Matcher, Handlers, Sender and Backport 3 services, due to which the connection check to Redis Sentinel did not pass the first time.

- Fixed the error due to which the Sender service did not restore the connection with Redis after a restart.

LUNA PLATFORM v.5.46.1

Changes

- The functionality for working with neural networks (detection, estimation and extraction) has been transferred from the Handlers service to the new Remote SDK service. This made it possible to make the Handlers service optional and disable it in the "ADDITIONAL_SERVICE_USAGE" setting when there is no need to work with handlers.

Now, when working with handlers, the Handlers service redirects requests for detection, estimation and extraction to the Remote SDK service, and then processes the result.

If the Handlers service is disabled, then:

  – Launching the API service will lead to the unavailability of using the following requests: "detect faces", "extract attributes", "estimator task", all resource requests "/handlers", all resource requests "/verifiers".
  – Launching the Tasks service will lead to the unavailability of performing the tasks "Additional extraction" and "Estimator".
  – Launching the Admin service will lead to the unavailability of performing the task "Additional extraction".

All neural networks and settings related to detection, estimation and extraction have also been transferred to the Remote SDK service. Now, to select such settings, it is necessary to enter the value "luna-remote-sdk" in the "Service name" field in the Configurator user interface instead of "luna-handlers".

Requests to resources "/iso", "/sdk", "/liveness" are now performed directly to the Remote SDK service without the participation of Handlers.

Disabling unused neural networks is now performed by passing the appropriate environment variable (for example, `--env=EXTEND_CMD="--enable-all-estimators-by-default=0 --enable-face-detector=0`) in the Remote SDK container launch command, not the Handlers container.

Now, instead of launching Handlers on the GPU, you need to run the Remote SDK on the GPU (flag `--gpus device=0`).

- The SDK has been updated to version 5.15.0. Key SDK changes affecting LUNA PLATFORM 5:

  - The following 109, 110 neural network models for extracting body descriptors have been supported.
  - The 105th, 106th, 107th neural network models for extracting body descriptors are considered outdated;
  - The estimator CrowdEstimatorV2 has been built in.

In this version of LUNA PLATFORM:

  - The 105th model **has been removed** from the Remote SDK container.
  - The default model was changed from the 107th to the 110th (setting "DEFAULT_HUMAN_- DESCRIPTOR_VERSION" of the Remote SDK service).

Upgrade from the version where the 107th model was used (the default model from versions 5.34.0 and above)

If you are upgrading from a version where the 107th model was used, then it is recommended to specify the 110th neural network model in the "DEFAULT_HUMAN_DESCRIPTOR_VERSION" setting during the upgrade and execute the "Additional extraction" task after launching the Admin service (see section "Launch Additional extraction task" in the administrator manual) to continue matching by old body descriptors.

Upgrade from the version where the 105th model was used

If you are upgrading from a version where the 105th model was used, then **launching the Remote SDK service will fail** if you do not perform one of the following actions before launching the Remote SDK container:

  - Manually change the setting value "DEFAULT_HUMAN_DESCRIPTOR_VERSION" from "105" to "110". After changing the version of the neural network for extracting body descriptors, you should perform the "Additional extraction" task after starting the Admin service (see "Launch Additional extraction task" section in the administrator manual). Otherwise, search and matching by old descriptors will be unavailable.
  - Disable the use of a neural network to extract the body descriptors by passing the "–enable- body-descriptor-estimator=0" argument when starting the Remote SDK container.

- Request the 105th neural network model from VisionLabs and transfer it to the Remote SDK container according to the instructions described in the "Use non-delivery neural network model" section in the administrator manual.

All of the above information has been added to the Upgrade manual in the "Change the neural network model for extracting descriptors" section.

When launching LUNA PLATFORM from scratch, no additional actions are required.

- A new parameter "estimate_people_count" has been added to the resources "sdk" and "/handlers/{handler_id}/events", which enables you to estimate the number of people in the image.

This functionality is licensed separately.

Note that such estimation cannot be compared in accuracy with individual face or body detectors. It should be used to approximate the number of people. See SDK documentation for more details.

In the body of the response, the result is returned in a separate field "image_estimations", since this estimation does not apply to faces or bodies of events.

- The following parameters have been added to the Estimator task for the ZIP archive:

  - "prefix" — Prefix of the file key. It can be used to download images from a specific directory.
  - "postfix" — Postfix of the file key. Can be used to upload images with a specific extension.
  - "recursive" — Recursive retrieval of images from nested directories.

- The ability to specify the relative time (now-time format) has been added in the parameters "create_time__gte","create_time__lt","end_time__gte","end_time__lt","insert_time__gte","insert_time__lt" in the following tasks:

  - Clustering
  - Exporter
  - Linker
  - Garbage collection
  - Additional extraction
  - Cross-matching task

This can be useful for filtering data for a certain time interval relative to the current time. For example, you can perform the task "Garbage collection" for the last few days.

- In the containers of the API and Admin services, the Python version has been updated to 3.11.

Support for older versions of Python has been discontinued.

- In the Docker Compose script "start_platform.sh" the lines concerning the launch of Backport 3, Backport 4, User Interface 3 and User Interface 4 are commented out.

Now these services will not start when executing the script.

- Instructions for activating the Guardant license without a graphical interface have been added to the license activation manual.

  To do this, you should additionally install a package intended to run interface applications without physical output to the screen.

Fixed errors

- Fixed the error that caused the Redis Sentinel connection check to fail the first time.

LUNA PLATFORM v.5.45.4

Changes

- The load on the Faces database has been reduced when performing license requests to check the maximum number of faces with linked descriptors or basic attributes.

- New resources have been added to the Image Store service:

    - HEAD "/1/buckets/{bucket}", which enables you to check the existence of a bucket
    - GET "/1/buckets/{bucket}", which enables you to get the creation time of the specified bucket

- The unused "bucket" setting has been removed from the "S3" section of the Image Store service.

- In the containers of the Tasks, Accounts, Configurator, Sender, Backport 3 and Backport 4 services, the Python version has been updated to 3.11.

  Support for older versions of Python has been discontinued.

Fixed errors

- The missing migration of the default setting value "score_threshold" in the "LUNA_HANDLERS_-FACE_DETECTOR_SETTINGS" section of the Handlers service has been added.

  When updating to the current version, the previous default value "0.42" will automatically update to the current default value "0.5".

  If the value of the "score_threshold" setting differs from the default one, the migration will not be performed.

- Incorrect structure of the field "face_detections" > "detection" of the event in the Events service has been changed to the correct form, corresponding to the OpenAPI specification.

LUNA PLATFORM v.5.45.3

Changes

- The resources "/matcher/faces", "/matcher/bodies" and "/matcher/raw" have added support for the Accept header, which defines the MIME type of response that should be expected from the client.

  There are two values available for the header: `application/json` (default) and `application/msgpack`.

- In the containers of the Faces, Image Store, Handlers, Python Matcher and Python Matcher Proxy services, the Python version has been updated to 3.11.

  Support for older versions of Python has been discontinued.

  The transition to the new version of Python allowed to increase the speed of comparison by 10-20% in some cases.

- The "verify_ssl" parameter has been added to the Estimator task, which enables you to disable SSL certificate verification for S3-like storage.

  This enables you to use a self-signed SSL certificate.

- The face parameters estimation has been accelerated in some cases when the parameters responsible for filtering are not specified in the request ("yaw_threshold", "fd_score_threshold", "liveness_threshold", etc.).

- The execution of matching requests has been accelerated when the "target" fields from the list below are used for candidate events:

  - "match_result"
  - "body_detections"
  - "face_detections"
  - "attach_result"
  - "tags"
  - "location"

Fixed errors

- Fixed an error where the request to create S3 bucket to the Image Store service was successful, but the bucket was not created if the "region" setting was not specified in the service settings.

LUNA PLATFORM v.5.45.1

Changes

- The SDK has been updated to version 5.14.0. Key SDK changes affecting LUNA PLATFORM 5:

  - updated FaceDetV3 detector;
  - updated Eyebrows estimator.

The default value of the `score_threshold` setting in the "LUNA_HANDLERS_FACE_DETECTOR_-SETTINGS" section of the Handlers service settings was changed from **0.42** to **0.5**. Migrating settings automatically update this value (see the section "Configurator database migration" in the upgrade manual). Check the face recognition logic if you use a `score_threshold` value other than the default value.

Note: In the upcoming releases, the default version of the neural network for extracting body descriptors will be changed. You will need to manually change the version in the settings of the Handlers service, otherwise its launch will fail.

- The LUNA PLATFORM license activation section has been moved to a separate license activation manual "LP_License_Activation_Manual.pdf/html".

- The ability to activate a license using a Guardant key has been added.

  This activation method requires a graphical system interface and access to the Internet. If the server where you plan to use LUNA PLATFORM does not meet these requirements, then you can perform part of the steps on a secondary server on Windows OS or Linux OS.

  The old HASP key activation method remains available.

  See the "License activation using Guardant key" section in the new license activation manual for details.

- The ability to filter by `null` values (the value means that the attribute was not estimated) has been added for candidate events in the following requests:

  - "matching faces";
  - "human body matching";
  - "create handler";
  - "generate events";
  - "save event";

  Also, filtering by `null` values has been added for event filters for the following tasks:

  - "Clustering";
  - "Exporter";
  - "Cross-matching";
  - "Linker";
  - "Estimator".

  Filtering has been added for the following attributes:

  - meta
  - source
  - emotion
  - mask

- ethnic_group
- liveness
- gender
- apparent_gender
- headwear_state
- sleeve_length
- upper_clothing_colors
- lower_garment_type
- lower_garment_colors
- shoes_apparent_color
- backpack_state
- city
- district
- street
- house_number
- area
- geo_position
- track_id

This enables you to filter events generated by different handlers with different policies, where the first one performed the estimation of a certain attribute (for example, the mask state is `occluded`), and the second one did not perform the estimation (for example, the mask state is `null`), but you need to get both events.

- The "get events" and "save event" requests have been sped up.

- A new parameter "verify_ssl" has been added to the "S3" section of the Image Store service settings, which enables you to disable SSL certificate verification for S3-like storage.

  This enables you to use a self-signed SSL certificate.

- The mechanism for checking the connection with the Image Store service has been updated.

  Previously, the Admin service performed validation by getting a list of all buckets, which could result in an error due to the user not having access to the buckets. Now the connection check is performed without getting a list of all buckets.

- The ability to license the LUNA PLATFORM using Guardant dongles has been added.

  This licensing method requires a graphical system interface and access to the Internet. If the server where you plan to use LUNA PLATFORM does not meet these requirements, then you can perform part of the steps on a secondary server on Windows OS or Linux OS.

  The old HASP keys licensing method remains available.

  See the "License activation" section of all installation/upgrade/migrate manuals for details.

- A new subsection `CACHED_DATA` has been added to the `DESCRIPTORS_CACHE` section of the Python Matcher service settings, which enables you to set data for caching.

  In the `face_lists` field, you can configure which lists will be cached and which ones will be ignored. There are two values available for this field:

  - `include` — only lists specified in this section will be cached (to disable, set `null`);
  - `exclude` — lists from this section will be ignored.

- A new example of the "Thin face" built-in plugin has been added to the Python Matcher service.

  The "Thin face" plugin is provided as an example to quickly match faces (objects) with simplified faces (objects). Simplified faces are stored in a separate database table "luna_faces" with three required columns ("face_id", "descriptor", "descriptor_version"). If necessary, you can configure a number of additional columns: "account_id", "lists", "create_time", "external_id", "user_data", "event_id", "avatar".

  See a detailed description of the "Thin face" plugin and instructions for writing custom plugins in the "PythonMatcherDevelopmentManual" document in the distribution package.

Fixed errors

- Fixed default values for some parameters in the following API service OpenAPI specification requests:

  - "create handler";
  - "extract attributes";
  - "iso";
  - "sdk";
  - "create attributes";
  - "get attributes".

  Also fixed the default value for the "extract_descriptor" parameter of the "create descriptors" request in the Backport 3 service OpenAPI specification and the default value for the "policies" > "create_face_policy" > "set_sample_as_avatar" parameter of the "create handler" request in the Backport 4 service OpenAPI specification.

- Fixed an error in the "Estimator" task, which returned status code 500 when trying to connect to a non-existent endpoint of an S3-like server.

  Now the status code 400 and error code 12031 are issued with the content "Specified bucket not available".

- Fixed an error in the "Estimator" task, which returned status code 500 when trying to connect to a Samba server without authorization.

  Now the status code 400 and error code 12031 are issued with the content of the Samba error.

- Fixed a behavior where, in some cases, access rights errors detected during license initialization might not be displayed in the Licenses service logs.

  Now, with such errors in the logs of the Licenses service, messages like `Failed to init licensing` will always be issued.

- Fixed `object of type 'Image'` has no `len()` error with status code 500 when estimating some rotated images with `use_exif_info` parameter enabled.

- Fixed an error in the Python Matcher service that caused the descriptor cache to be reloaded when the logging settings were changed.

- Fixed an error where it was not possible to specify more than 36 characters for the `source` filter for candidates in matching requests.

  Now you can specify a maximum of 128 characters.

- Fixed an error that occurred when migrating accounts and tokens of the Backport 3 service when upgrading from versions 5.2.0...5.28.0 to versions 5.30.0 and higher.

LUNA PLATFORM v.5.42.0

Changes

- Now the Image Store service can store any files as objects (for example, a video file).

  Previously, only the following types of objects were available: `json`, `text`, `zip`, `pdf`.

  You can upload files using the "create objects" request. The bytes of the file must be specified in the request body, and the `Content-Type` header must contain the MIME type of the file (for example, `video/mp4`). The response to the "get object" request contains the header Content-Disposition. This header contains the file name of the attachment object (for example, `video_1.mp4`). The file name is generated based on the `object_id` and the MIME type.

  If the MIME type of the file could not be determined, then the file extension will be set as '.bin'.

  Now the `Accept` header is ignored for the "get object" request. Previously, the service returned the error `12023, Content type is unacceptable` with the status code 406, if the object content type did not match the MIME type of the file.

- Now, when executing any request with correct authorization, information about the corresponding account is displayed in the logs of the API service.

  This functionality enables you to determine who exactly executed a particular request. This may be required for information security and system administrators.

  If the request was executed with BasicAuth or LunaAccountIdAuth type authorization, the following message will be displayed in the logs:

```
Request invoked by user (account_id: '270531af-e52e-4538-9181-628
    d9900a0db')
```

If the request was executed with BearerAuth type authorization, the following message will be displayed in the logs:

```
Request invoked by user (account_id: '270531af-e52e-4538-9181-628
    d9900a0db' token_id: 'd57e16f5-e243-47d2-aa85-8b200c12d86f')
```

If the request was executed without authorization, the following message will be displayed in the logs:

```
Request invoked by user (account_id: null)
```

The logs of the Accounts service additionally display information about the creation of tokens by specific users:

```
User with account_id: '270531af-e52e-4538-9181-628d9900a0db' create
    token: 'd57e16f5-e243-47d2-aa85-8b200c12d86f'
```

Logging information about the creation of tokens enables you to track where the token came from and which user it belonged to, even after it was deleted.

- A new check `shoulders_position` has been added to the `face_quality` and `iso` sections, which determines the most predominant state of the shoulder position from the following:

  – non-parallel
  – parallel
  – hidden

- The execution of the "get faces" request with all `target` fields has been sped up.

- The memory usage by the Task service has been reduced in some cases of using the "Cross-matching" task.

Fixed errors

- Fixed an error where the script `db_create.py` didn't work for non-default configuration values of the Configurator service.

- Fixed an error where the cache warmed up when starting the Python Matcher service with the "DESCRIPTORS_CACHE" setting turned off.

- Fixed an error where the Python Matcher service started and continued to work when some workers stopped with an error.

A "worker" is the Python Matcher service process that handles incoming (HTTP) requests.

- Fixed behavior that could result in the deletion of one or all new event subscriptions if the new subscription was created and the old one was deleted at the same time.

- Fixed an error where using EXIF data evaluation (parameter `use_exif_info`) could lead to an error of the form `Internal server error`.

r # LUNA PLATFORM v.5.40.0 {-}

Changes

- Starting from the current release, commands for launching PostgreSQL, InfluxDB, and Image Store containers specify paths of directories for mounting located in the root directory `/var/lib/luna/<db_or_bucket_folder>`, unlike previous versions where paths were specified for a specific LUNA PLATFORM version `/var/lib/luna/current/example-docker/<db_or_bucket_folder>`.

  This means that PostgreSQL, InfluxDB, and Image Store data will now be stored in the root directory and will no longer need to be transferred to the directory with the new version of LUNA PLATFORM when upgrading.

  The Docker Compose script has also been updated to reflect the information above.

  Note: When upgrading to the current version, you must migrate the old PostgreSQL, InfluxDB, and Image Store data to the root directory, and then delete and re-create the containers with new mount directory paths. See "Move data" in the upgrade manual.

- Added support for LUNA PLATFORM services without the Image Store service.

  The service can be disabled in the "ADDITIONAL_SERVICES_USAGE" setting of the Configurator service.

  Resources that require the Image Store service to be disabled will return the error `11070, Luna Image Store service is disabled`.

  When the Image Store service is disabled, there are some specific features to note:

  - Objects of the type `images`, `objects`, `samples`, and sample save policies in handlers/verifiers will be unavailable.

  - All tasks, except Garbage Collection, Linker, and Estimator, will become unavailable. However, there are some limitations for these tasks:

    * Garbage Collection, Estimator, Linker: after the subtask completes, the task status will be updated to `Done`, and the task result ID will be `None`.
    * Garbage Collection: deleting samples will become unavailable.

  If the Image Store service is disabled after events with the `image_origin_policy` are generated, when using the Garbage Collection task and the `remove_image_origins`

parameter, the Tasks service will still attempt to delete the source images with an external URL.

When the Image Store service is disabled, samples and portraits, as well as the "get portrait" and "get portrait thumbnail" resources, become unavailable.

A new setting "BACKPORT3_ENABLE_PORTRAITS" has been added to the Backport 3 service, which enables you to disable the ability to use portraits but leave the ability to use the rest of the functionality of the Image Store service. If the use of the Image Store service is disabled in the "ADDITIONAL_SERVICES_USAGE" setting, then the above setting must also be disabled.

See the "Disableable services" section of the administrator manual for details.

- Added support for specifying user metadata for the source image to the "generate events" request.

  The metadata is passed using headers like "X-Luna-Meta-<user_defined_key>:<user_defined_-value>", which are sent to the Image Store when the source image is saved during event generation.

  Headers must be specified in the `image_origin` part when using the `multipart/form-data` request content type.

  For more information about using user metadata when saving images to the Image Store service, see the section "Source images saving" of the administrator manual.

Fixed errors

- Fixed error where when specifying the source image in the body of the request "generate events" as not a URL, it was saved to the Image Store service regardless of the state of the `image_origin_policy`.

- Fixed description of incorrect "Content-Type" response headers for requests "detect faces" and "get list count".

- Added new `Exclude-Header` header to almost all requests of the Admin service.

  This fixed an issue in the Admin UI that caused the user to log out after reloading the page.

- Fixed license error that occurred when using a trial license for the specified number of days.

LUNA PLATFORM v.5.38.3

Changes

- SDK was updated to version 5.13.0. Major SDK changes affecting LUNA PLATFORM:

  - updated body detector
  - updated FishEye estimator

The default value of the `score_threshold` setting in the "LUNA_HANDLERS_BODY_DETECTOR_-SETTINGS" section of the Handlers service settings was changed from **0.3** to **0.5** . Migrating settings automatically update this value (see the section "Configurator database migration" in the upgrade manual). Check the body recognition logic if you use a `score_threshold` value other than the default value.

The default value of the `redetect_face_target_size` setting in the "LUNA_HANDLERS_-FACE_DETECTOR_SETTINGS" section of the Handlers service settings was changed from **45** to **64** . Migrating the settings will automatically update this value (see the section "Configurator database migration" in the upgrade manual). Check the face recognition logic if you use a value `redetect_face_target_size` other than the default value.

- Now, during the initial license check, only the license expiration time is checked, instead of all license checks. The remaining checks are now performed when the Licenses service receives requests.

  This change speeds up the loading of the Licenses service.

Fixed errors

- The missing description for the `msgpack` format was added to the OpenAPI specification in the matching requests "matching faces" and "matching bodies".

- The error was fixed, in which not all date and time formats were read in the monitoring. If an unintended format was transferred, then a reading error occurred.

  Now the ability to read all date formats from the Influxdb was added.

- The Accounts service error was fixed, in which some errors related to processing requests to the Accounts database were not processed properly and were not output to the service logs.

- The error was fixed where some data could be missing from tasks when the parameter `tasks_to_faces_requests_concurrency` of the Tasks service was set to 1.

- If the matching request contains the value of the `target` field, which the plugin does not support, the service adds a key value for the `target` field to the request (`face_id` for matching by faces and `event_id` for matching by events).

  Previously, the plugin did not support the `face_id` or `event_id` field as the value of the `target` field and the request could fail or return an empty list of candidates. Now the Python Matcher Proxy service will not redirect such requests to the matching plugin.

  For more information, see the section "Matching targets" in the administrator manual.

LUNA PLATFORM v.5.38.1

Changes

- The ability to transfer source images in URL or Base64 format in the "generate events" request has been added.

  The source images can be transferred by specifying the `Content-Type = application/json` or `Content-Type = multipart/form-data`.

- Docker Compose script `start_logging.sh` has been added to the LUNA PLATFORM distribution package, which launches the LUNA Dashboards, Grafana Loki and Promtail service, enabling you to flexibly work with LUNA PLATFORM logs in Grafana.

  The script is located in the `example-docker` directory.

  In the LUNA PLATFORM installation manual using Docker Compose, the relevant information has been added to the launch section of the main Docker Compose script.

  See detailed information about logging visualization in the "Grafana Loki" section of the administrator manual.

- Sending data to InfluxDB has been speeded up.

- The ability to filter by `null` values (the value means that the attribute was not estimated) for the following attributes in the requests "get events" and "get statistics on events" has been added:

  - `meta`
  - `source`
  - `emotion`
  - `mask`
  - `ethnic_group`
  - `liveness`
  - `gender`
  - `apparent_gender`
  - `headwear_state`
  - `sleeve_length`
  - `upper_clothing_colors`
  - `lower_garment_type`
  - `lower_garment_colors`
  - `shoes_apparent_color`
  - `backpack_state`
  - `city`
  - `district`
  - `street`
  - `house_number`
  - `area`
  - `geo_position` (поля `origin_longitude` и `origin_latitude` для запроса "get events")
  - `track_id`

This enables you to filter events generated by different handlers with different policies, where the first one performed the estimation of a certain attribute (for example, the mask state is `occluded`), and the second one did not perform the estimation (for example, the mask state is `null`), but you need to get both events.

- Section of the form `<service_name>_HTTP_SETTINGS` has been added to the settings of each service, containing settings responsible for processing HTTP connections.

  The following settings are available:

  - `request_timeout` — the duration of time between the instant when a new open TCP connection is passed to the server. Value (in seconds) is integer number, default 60.
  - `response_timeout` — the duration of time between the instant the server passes the HTTP request to the app, and the instant a HTTP response is sent to the client. Value (in seconds) is integer number, default 600.
  - `request_max_size` — how big a request may be (bytes). Value (in bytes) is integer number, default 1gb.
  - `keep_alive_timeout` — http keep alive timeout. Value (in seconds) is integer number, default 15.

  See the following link for details: https://sanic.dev/en/guide/deployment/configuration.html#builtin-values

- Improved performance of "get faces" request with `list_id` and `face_id__gte/lt` filters.

- The ability to redefine the settings of services at their start using environment variables has been added.

  The `VL_SETTINGS` prefix is used to redefine the settings. Examples:

  - `VL_SETTINGS.INFLUX_MONITORING.SEND_DATA_FOR_MONITORING=0`. Using the environment variable from this example will set the `SEND_DATA_FOR_MONITORING` setting for the `INFLUX_MONITORING` section to `0`.
  - `VL_SETTINGS.OTHER.STORAGE_TIME=LOCAL`. For non-compound settings (settings that are located in the `OTHER` section in the configuration file), you must specify the `OTHER` prefix. Using the environment variable from this example will set the value of the `STORAGE_TIME` setting (if the service uses this setting) to `LOCAL`.

  The environment variable can be specified using the `ENV` argument when running services in Docker containers.

- The "get license" request of the Licenses service now has the ability to get the value of a specific licensing feature using the new `targets` field.

- Support for dashboards for LIM has been added.

Fixed errors

- In the example responses of requests "get events", "get event", "get task result" and "ws handshake" in the OpenAPI specification, the description of the fields from `body_basic_attributes` to `basic_attributes`, from `upper_body_attributes` to `upper_body`, from `lower_body_attributes` to `lower_body` and from `body_accessories` to `accessories`.

- Fixed the error where the "Exporter" task would continue to run with incomplete data even if it encountered an error.

  Now, when an error occurs while executing the Exporter task, errors with codes 28038 (failed to load data) and 28039 (failed to load data, but there were repeated attempts to reconnect to the Faces or Events services) will be returned.

- Fixed the error due to which the loading of a large task result was interrupted.

LUNA PLATFORM v.5.36.5

Changes

- Now the address of the licensing server is set in the Licenses service configurations, and not in the "hasp_30147.ini" file.

  Thus, it is no longer necessary to mount this file when launching the Licenses container.

  Instructions have been added to the installation and upgrade manuals for specifying the address of the licensing server using the Configurator user interface before launching the Licenses service.

  The installation manual also includes instructions for specifying the address of the licensing server using the file "platform_settings.json" before starting the launch process.

  Note. When upgrading from previous versions, you must specify the licensing address in the new configuration, otherwise the Licenses service will not start.

- The appearance of the Admin service user interface has been updated.

  The logic of working with the user interface has remained the same.

- Added the ability to record the coordinates of the face or body bounding boxes in the generated event when using a sample as an image source. The coordinates can be passed either using an external application or manually set in a request for generating the event (for example, using the "multipart/form-data" schema).

  The specified coordinates are recorded in the `face_detections/body_detections > detection > rect` field of events.

  Setting the coordinates of the bounding boxes will be ignored in other requests where you can specify a sample as an image source.

  Previously, only the bounding boxes of the face or body of the source images could be saved to the event.

- Support for filtering candidates by the `meta` field of the event has been added to the "match_-policy" of requests "create handler" and "validate handler policies".

- Filters by the `meta` event field have been added to the tasks "Clustering", "Exporter", "Cross-matching" and "Linker".

- Validation of the filter by the "meta" field for matching has been improved. This filter is used in the following requests:

  - "matching faces",
  - "matching bodies",
  - "cross-matching task".

- Support for filtering and aggregation by the "meta" field of event has been added to the request "get statistics on events".

- A new request "get platform features" has been added to the API service, in response to which you can get information about whether the license is active and whether the license period has expired, as well as information about the license functions enabled ("face_quality", "body_-attributes" and "liveness") and the availability of functionality for working with the Events, Tasks and Sender services. The use of these services is enabled in the "ADDITIONAL_SERVICES_USAGE" configuration of the Configurator service.

- The ability to select faces as an object for garbage collecting (the "target" field) has been added to the task "Garbage collection".

  Filters "create_time__lt","create_time__gte","user_data","list_id" are also available, as well as parameters for storing results (field "store_results") and removing samples of faces (field "remove_samples").

  Selecting faces as an object for garbage collecting is also available in the Admin user interface.

- The logic of the dynamic range estimator has been updated in the group of checks "face_quality".

- The resource "/attributes/batches" of the Faces service has been moved to "/descriptors/batches".

- In all LUNA PLATFORM services, logging to a file has been disabled by default (the "log_to_file" setting of each service).

- The documentation included in the distribution package has been updated.

  The "General concepts" section of the administrator manual has been redesigned.

  The "Before launch/upgrade" sections of all installation/upgrade/migration manuals have been redesigned. Some of the descriptive information has been moved to the "Additional information" section located at the end of each manual.

Fixed errors

- The enumeration in the "content > filter > object_type" field from "face"/"event" to "faces"/"events" of the "create additional extract task" request has been corrected in the OpenAPI specification of the Admin service.

- The error has been fixed, which outputs unnecessary information to the logs when using the "create_usage_task" statistics counting command.

- The error in the OpenAPI specification of the Python Matcher service has been fixed, due to which the meta filter did not have a nullable type in the requests "face matching", "human body matching", "cross matching faces" and "cross matching bodies".

- The error has been fixed, due to which in the request "face matching" of the Python Matcher service, the object "candidates > filters" was not required.

LUNA PLATFORM v.5.35.0

Changes

- Support for the old Services for index building and searching by index has been discontinued.

  The services have been removed from the documentation and service settings.

  Now only the LUNA Index Module is used to create the index and search the index.

- Functionality for saving and retrieving user-defined image metadata using custom headers has been added.

  You can save an image with user-defined metadata in the "create images" resource by setting the header X-Luna-Meta-<user_defined_key> with the value <user_defined_value>. In the source image bucket in the Image Store, the metadata is stored in a separate <image_id>.meta. json file that is located next to the source image.

  In response to "get image" request, you should specify the "with_meta=1" header to get the image metadata in the response header.

  To store metadata values for multiple keys, you should specify multiple headers.

- The LUNA Dashboards version has been updated to 0.0.5. The Grafana version in the LUNA Dashboards container has been updated to 8.5.20.

  Now you don't need to use a separate command to launch dashboards, they are created automatically when the LUNA Dashboards container is launched.

- Grafana Loki containers (log aggregation system) and Promtail (agent delivering LUNA PLATFORM logs to Grafana Loki) have been added.

  Running containers requires Grafana to be running and the "Loki" data source configured. The "Loki" data source has already been created in the LUNA Dashboards container.

In the "Grafana Loki" section of the installation manual, instructions for launching Grafana Loki and Promtail have been added.

See the "Grafana Loki" section of the administrator manual for more information.

- Support for Samba network file system as an image source (the "source_type" parameter) was added to the Estimator task ("/tasks/estimator" resource).

  For this type of source, the following parameters can be set in the request body for connecting to the Samba:

  - host — Samba IP address (required);
  - port — Samba port;
  - user, password — authorization data. If there is no authorization data, the connection to Samba will be performed as a guest.

  As in Estimator tasks using FTP server, S3-like storage or network disk as image sources, it is possible to set the path to the directory with images, recursively receive images from nested directories, select the type of transferred images, and specify the prefix and postfix.

  To obtain correct results of image processing using the Estimator task, all processed images should be either in the source format or in the format of samples.

  See the OpenAPI specification for related examples and more information.

- The "extract_exif" parameter has been added to the "sdk" resource, which enables extracting EXIF data from an image.

- Filters "geo_position" (parameters "origin_longitude", "origin_latitude", "longitude_delta" and "latitude_delta") and "user_data" have been added to the "ws handshake" resource.

- Support for filtering candidates by the "meta" field of the event has been added to the "matching faces", "matching bodies" resources and the "match_policy" of the "create handler" request.

- A table has been added to the "Upgrade notes" section of the upgrade manual with key changes from previous versions of LUNA PLATFORM that affect the installation and operation of LUNA PLATFORM. This table may be useful when trying to upgrade LUNA PLATFORM from a version other than the previous one.

LUNA PLATFORM v.5.34.0

Changes

- The SDK has been updated to version 5.12.0. Key SDK changes affecting LUNA PLATFORM 5:

  - the following estimators have been updated: LivenessOneShotRGB, FishEye, Orientation, HeadWear;

– the 54th, 56th, and 57th face descriptor extraction neural network models and the 104th and 107th body descriptor extraction neural network models have been removed. Support for these models remains available. If necessary, you can request them from VisionLabs specialists.

Now the default body descriptor extraction neural network model is 107 (setting "DEFAULT_-HUMAN_DESCRIPTOR_VERSION" of the Handlers service).

Important information for upgrading from previous versions

You should do one of the following before starting the Handlers container:

– manually change the setting value "DEFAULT_HUMAN_DESCRIPTOR_VERSION" from "104" or "106" to "107". After changing the version of the neural network for extracting body descriptors, you should perform the "Additional extraction" task after starting the Admin service (see "Launch re-extraction task" section in the administrator manual). Otherwise, search and matching by old descriptors will be unavailable.
– disable the use of a neural network to extract the body descriptors by passing the "–enable-body-descriptor-estimator=0" argument when starting the Handlers container;
– request the 104th neural network model from VisionLabs and transfer it to the Handlers container according to the instructions described in the [Use non-delivery neural network model](https://docs.visionlabs.ai/luna/v.5.34.0/standard-distribution/admin-manual/additional-information/#use-non-delivery-neural-network-model] section in the administrator manual.

If one of the above actions is not performed, then **starting the Handlers service will fail**.

All of the above information has been added to the update manual in the "Changing the model of the neural network of bodies" section.

When launching LUNA PLATFORM from scratch, no additional actions are required.

• New "meta" field has been added to the event structure, designed to store arbitrary user data in JSON format (no more than 2 MB).

It is assumed that with the help of this functionality, the user will create his own data model (event structure) and use it to store this data. Note that if you plan to store multiple structures, you must explicitly separate them to avoid overlapping fields. For example, as follows:

```
{
    "struct1": {
        ...
    },
    "struct2": {
        ...
    }
```

```
    }
```

Data in the "meta" field can be set in the following ways:

- in the "generate events" request body with the content type "application/json" or "multipart/form-data";
- in the "save events" request body;
- using a custom plugin or client application.

In the "generate events" request body, it is possible to set the "meta" field both for specific images and for all images at once (mutual meta-information). For requests with aggregation enabled, only mutual meta-information will be used for the aggregated event, and meta-information for specific images will be ignored. See the detailed information in the "generate events" request body in the OpenAPI specification.

The "meta" field can be used as a filter in a "get events" request or as a value for the "target" parameter in a "get event" request.

The "meta" column has been added to the Reporter and Exporter tasks.

Support for event meta-information has also been added to the "ws handshake" resource.

If necessary, you can build an index to improve the search.

See the detailed description and operation features in the "Events meta-information" section of the administrator manual.

- Support for the Liveness V1 service has been discontinued.

The Liveness V1 service has been removed from the documentation and service settings. Liveness V2 has been renamed to Liveness.

- New argument "enable-all-estimators-by-default" has been added to the launch arguments of the Handlers container, enabling/disabling the initialization of all default estimators and detectors.

Previously, in order to use certain estimators or detectors, it was necessary to specify the status of each existing estimator. Now it's enough to disable the initialization of all estimators "enable-all-estimators-by-default=0" by default, and then specify only those estimators or detectors that you need to enable. An example of a command to start the Handlers service using only a face detector and estimators of face sample and emotions.

```
docker run
...
--env=EXTEND_CMD="--enable-all-estimators-by-default=0 --enable-face-
    detector=1 --enable-face-warp-estimator=1 --enable-emotions-
    estimator=1"
...
```

See the "Enable/disable several estimators and detectors" section of the administrator manual for details.

- Now the filter by "account_id" is optional for the Clustering (request body filter), Cross-matching (request body filter for candidates or references) and Estimator (matching filters in handler policies) tasks. This enables you to match across objects from different accounts.

- For the resources listed below, a new "external_url" parameter has been added to the response body, indicating the absolute address to the object:

    - "create account",
    - "create token",
    - "replace token",
    - "create images",
    - "create objects",
    - "extract attributes",
    - "create face",
    - "create list",
    - "save event",
    - "create verifier",
    - "create handler".

  The absolute address is the address of the API service specified in the "EXTERNAL_LUNA_API_-ADDRESS" setting of the API service. The default setting value is http://127.0.0.1:5000/6/. This change enables you to use links from API service responses for your own purposes, without knowing the exact address of the service. The change also enables you to send links in a convenient format, by which you can get their contents.

  Relative link example ("url" parameter): "/6/objects/4a870804-0cd6-4c13-9c78-98ad167dc4ec"

  Absolute link example ("external_url" parameter): "http://127.0.0.1:5000/6/objects/4a870804-0cd6-4c13-9c78-98ad167dc4ec"

- Support for the "Accept" header has been added to the "get face descriptor batches" resource of the Faces service, which takes two values — "application/x-flutbuf" (default) and "application/x-msgpack".

- The unused "max_face_size" parameter of the "LUNA_HANDLERS_FACE_DETECTOR_SETTINGS" section has been removed from the Handlers service settings.

  The "max_face_size" parameter is calculated as "min_face_size * 32".

LUNA Index Module changes

- The LUNA Index Module installation manual has been updated.

  Now by default there are commands to install the module in "/var/lib/luna/" directory instead of in "/var/lib/luna/current/" directory.

The index storage is now created in the "/var/lib/luna/" directory to simplify the upgrade process.

- The LUNA Index Module upgrade manual has been added.

- LUNA Index Module now takes into account changes in the version of face descriptors. It means that:

    – the Indexer service builds an index from descriptors of the version specified in the "DEFAULT_-FACE_DESCRIPTOR_VERSION" setting of the Index Manager service;

    – the Index Manager service automatically rebuilds the index if it does not contain information about versions of descriptor;

    – the Indexed Matcher service loads only those indexes that contain descriptors of the version specified in the "DEFAULT_FACE_DESCRIPTOR_VERSION" setting of the Index Manager service.

In this regard, the mandatory field "descriptor_version" has been added to the index metastructure ("meta.json"). The "get indexes" and "get most relevant indexes" requests also return the "descriptor_version" parameter.

Important information for upgrading from previous version

After starting the Index Manager service, it will automatically start rebuilding all indexes that do not contain information about descriptors, i.e. all indexes created in a previous version of LIM. Rebuilding the index can take a long time, depending on the number of faces on the lists. In order to avoid the lengthy process of rebuilding the index, you can add the "descriptor_version" field with the corresponding version of the descriptors to the "meta.json" files of all previously created indexes before starting the Index Manager service.

Reminder added to "Important information" section of the new LIM upgrade manual.

- Now, by default, the Indexed Matcher service monitors changes in lists with faces. If new changes are made to the list, the Indexed Matcher service updates the corresponding indexes in its memory by gradually adding a small number of descriptors.

The use of this functionality is controlled by the "enabled" setting of the "LIM_MATCHER_REFRESH" section of the Configurator service.

When a cached index is updated, the Indexed Matcher service stops matching on that index, but continues to accept new match requests for that index. By adding a small number of descriptors to the cached index, the matching process is performed with minimal interruption. However, it should be taken into account that if elements are inserted into the list too often (dozens and hundreds of additions), then this can cause a significant degradation in the speed of work, up to an almost complete stop of the matching process.

If this functionality is used, then it is not necessary and not recommended to perform frequent index rebuilds. Accordingly, it is recommended to increase the planning routine period ("planning_period" setting of the "LIM_MANAGER_INDEXING" section of the Configurator

service. However, adding new faces to the cached index is slower than rebuilding the index, so it makes no sense to use this function if a very large number of faces have been added to the list. In this case, it is easier to rebuild the index again.

See "Cached index refreshing" section in the LIM administrator manual.

- Downloading descriptors from the Faces service has been optimized. The "Accept" header has been changed from `application/x-flutbuf` to `application/x-msgpack`.

- "Warming up" (test matching) the first created index before starting to use it has been added. Subsequent indexes are not "warmed up".

  "Warm-up" is performed after the index is loaded into the memory of the Indexed Matcher service and the service is waiting for a matching request.

- The Python version has been updated to 3.10. Support for other versions has been discontinued.

Fixed errors

- The API service OpenAPI specification has been corrected and extended the response code examples 403, 408, 413, 500, 503, and 504. See the response examples in the OpenAPI specification.

- The "perform verification" request now has a previously missing parameter for specifying the type of image to be processed "image_type". Two values are available for this parameter — 0 (source image) and 1 (face sample).

- Incorrect thresholds for the "mouth_occluded", "mouth_smiling" and "mouth_open" parameters of the "face_quality" parameter group in the "create handler", "get handlers", "get handler" and "replace handler" requests have been fixed in the OpenAPI specifications of API services and Handlers.

  The specification previously stated that the allowed thresholds for the above parameters are [-1..1], although in fact the allowed thresholds are [0..1].

LUNA PLATFORM v.5.33.0

Changes

- Starting with the next release, Liveness V1 support is discontinued. The container launch commands and description will be removed from the documentation.

- New parameter "estimate_lower_body" was added to the "/sdk" resource and the "detect_policy" > "body_attributes" of the "/handlers" resource, which allows to perform estimation of the following bodies attributes:

  – "lower_garment" (type: trousers, shorts, skirt, undefined; color: orange, purple, red, white, yellow, pink, brown, beige, khaki, multicolored, undefined);

– "shoes" (color: white, black, other, undefined).

This estimation is disabled by default in both resources. Also now the "estimate_upper_body" parameter determines the color of the headwear ("headwear_apparent_color" parameter). The colors are similar to the colors of the lower garment. New body attributes were added to the event structure (see "detections" > "samples" > "body" > "detection" > "attributes"). Body attribute filters "lower_garment_types", "lower_garment_colors", "shoes_apparent_colors" and "headwear_apparent_color" were added to the "match_policy" policy of the "/handlers" resource and to the "human body matching" resource, specified when using events as candidates. The ability to set these body attributes as values for the "targets" field was also added.

Filters on these body attributes can also be used when sending events via websockets (see the "ws handshake" resource).

Appropriate event filters were added to Cross-matching, Clustering, Reporter, Exporter and Linker tasks. The corresponding columns were added to the Reporter and Exporter tasks.

The ability to specify these body attributes when creating a new handler in the Estimator task is supported.

The structure of the event in the "generate events" or "save events" requests is also extended by these body attributes. In addition, in these requests and the "sdk" request, aggregation by these body attributes is available. The aggregated attribute values are displayed in the "aggregate_-estimations" fields of the respective resources.

• Settings of the Handlers service were updated.

Now you can set both individual runtime settings for each estimator/detector, and global runtime settings for all estimators/detectors.

Global runtime settings are set in the "LUNA_HANDLERS_RUNTIME_SETTINGS" section and contain three settings:

– global_device_class — device class ("cpu" or "gpu") for all estimators/detectors that have the parameter value "device_class" = "global" (see below);
– num_threads — number of worker threads for all estimators/detectors;
– num_compute_streams — number of streams for all estimators/detectors.

Individual runtime settings are set in the section like "LUNA_HANDLERS__SETTINGS.aintime_-settings" and contain three settings:

– device_class — device class to perform estimation ("cpu", "gpu" или "global");
– optimal_batch_size — batch size for estimation;
– worker_count — amount of workers to perform estimation.

• In the Handlers service, the ability to enable/disable the use of individual estimators and detectors was added. By default, all estimators are enabled.

Previously, it was possible to enable/disable only all estimators at once. The ability to enable individual estimators allows you to save RAM or GPU memory, since when the Handlers service launches, the ability to perform these estimates is checked and data is loaded into memory. If you disable an estimator or detector, you can also remove its neural network from the Handlers container.

Disabling estimators or detectors is possible by passing arguments with the names of estimators or detectors to the launch command of the Handlers service. Arguments are passed to the container using the "EXTEND_CMD" variable. A list of all the arguments and an example of launching the container were added to the LUNA PLATFORM 5 administrator manual.

- Support for images with CMYK scheme was added.

- The "ws handshake" resource was added to the Backport 3 service for the possibility of receiving events via web sockets.

  This functionality is similar to the functionality of the "Event & Statistic" service of LUNA PLATFORM 3.

- Settings of the API service were updated.

  The "LUNA_HANDLERS_LIVENESS_SETTINGS" section was removed from the API service settings because it was not used.

  The "LIVENESS_THRESHOLD" parameter was renamed to "LIVENESSV1_THRESHOLD".

- The "estimate_glasses" parameter was added to the "detect_policy" of the "/handlers" and "/verifiers" resources, which enables you to estimate the type of glasses (glasses, sunglasses, no glasses).

- The "estimate_attributes" parameter in the "create descriptors" and "search" requests of the Backport 3 service were expanded with the ability to estimate the type of glasses (glasses, sunglasses, no glasses). Now, when performing attribute estimation, gender, age, and points will be estimated.

  The filters "glasses__gt" (lower threshold) and "glasses__lt" (upper threshold) are available in the "ws handshake" resource.

- The "account_id" filter was added to the request parameters of most resources with GET methods that return multiple objects in the response.

  Using this filter enables you to get data (tokens, faces, attributes, etc.) belonging to a specific account. The exceptions are the "get accounts" resource, service resources ("get health", "get configs", etc.) and the "ws handshake" resource.

  Also, the "account_id" query parameter was added to the "face matching" and "human body matching" resources.

These query parameters will only work if the "visibility_area" parameter of the token is set to "all". Otherwise, an error will be returned.

- The ability to use a user SQL function for matching was added.

  This function can be useful if it is not possible to use C-extensions of the PostgreSQL database. The function may be needed, for example, to deploy LUNA PLATFORM 5 on AWS and use Amazon Aurora PostgreSQL.

  See the detailed description in the "Alternative matching options" section in the file "luna_-v.5.33.0/extras/VLMatch/postgres/readme.md" of the distribution package.

- The Python version was updated to 3.10 in the containers of the API and Licenses services.

- The Redis database was updated to version 7.0.5-alpine3.16.

Fixed errors

- The error was fixed due to which in the request to get statistics on events (resource "/events/statistics") the filter with the current "account_id" was always passed, due to which statistics were returned only according to the data of the user account that performed the request.

  This resulted in the fact that users with the account type "admin" and "advanced_used" could not receive statistics on other users.

- The error was fixed, due to which in the request to the resource users with the account type "advanced_used" or "admin" could not get statistics on other users.

- The error was fixed, due to which it was impossible to get both dynamic and static handlers in the "get handlers" request using the "is_dynamic" filter at the same time. Now if you do not specify this filter, both types of handlers will be returned.

- The error in the "detect face" request was fixed in the API specification where a non-existent "liveness" field was displayed in the response body.

- The error was fixed, due to which, when making a request to the "/3/attributes//samples" resource of the Faces service, the filter by "account_id" was not taken into account.

- The error was fixed where the "clear authorization" request from the Admin service would return an "Internal server error" when there was no cookie.

  Now the "Cookies for current session not found" error is returned.

- The "Internal server error" was fixed, which occurred when loading an image with a color scheme that the SDK does not support.

  Now if an attempt is made to upload an image with an unsupported color scheme, the correct error code 18003 will be returned.

- The error in the Python Matcher service with the cache enabled was fixed, in which unexpected errors could occur in the logs after changing the settings in the Configurator service.

Now, the service process shutdown routine takes care of cancellation of open connections and errors in the logs do not appear.

- The error of the "account_id" filter in the Python Matcher service was fixed.

If in the "face matching" and "human body matching" requests with the specified "account_id" filter, the value of the "account_id" field of the event reference differed from the value of the "account_id" field of candidates, then an incorrect message was returned that only the candidate was filtered. Now a correct message is returned that the specified reference has not been found.

- The error in the Python Matcher service was fixed that occurred in "face matching" and "human body matching" requests with Basic authorization and "user" account type when specifying the account ID in the "candidates" > "filter" > "account_id" field, different from the account ID by which the request is made.

Now, when trying to use other "account_id" in filters with the "user" account type, the correct message will be returned.

- The error in the "upgrade attribute" request of the Handlers service was fixed when an image processing error (for example, if you specify a body sample instead of a face sample) returned an empty field, or an "Internal server error". Now, if image processing errors occur, the correct error code 11043 will be returned.

LUNA PLATFORM v.5.32.0

Changes

- The ability to match by body descriptors when generating an event was added.

The "match_policy" policy of the handler was expanded with a new filter "descriptor" > "descriptor_type", which enables you to explicitly specify what type of descriptor will be matched — by the face descriptor ("face" value) or by the body descriptor ("body" value).

If the face matching is performed, but the descriptor type is specified as "body", then an error will be returned.

Face descriptors are linked to the "face" object, and body descriptors are linked to the "event" object. If a special plugin is not used, then matching by events may take a long time.

Note that body matching is not as accurate as face matching. With a large number of candidate events, the probability of false determinations of the best matches is higher than with a large number of candidate faces.

- Service containers are now named the same when manually launched and when launched with Docker Compose.

For example, previously the container for the Configurator service was created under the name `example-docker_configurator_1`, and now it will be created under the name `luna-configurator`.

- The Python version was updated to 3.10 in the containers of the services Faces, Image Store, Accounts, Tasks, Events, Configurator, Sender, Handlers, Backport3 and Backport4.

  All commands related to using Python inside containers were updated in the documentation, namely, commands like `python3.9` were replaced with `python3`.

- The "include_luna_services" parameter was added to the "healthcheck" resource of the Faces service, with which you can enable or disable the healthcheck for the LUNA PLATFORM services on which this service depends. If this option is enabled, then additional requests are sent to the "/healthcheck" resources of these services.

- Support for the neural network 60 was added to the services for building an index and searching by index.

Fixed errors

- Missing information about the "Additional extract" task was added to the response bodies of the "get tasks", "get task" resources of the OpenAPI specification.

  As before, the "Additional extract" task must be performed using the Admin service.

- The "tasks" > "content" > "filters" > "account_id" parameter in the response bodies of the "get tasks", "get task" resources for the "Garbage collection" task is now optional, since "account_id" may be missing if the task was launched from the Admin service.

- The "tasks" > "content" > "target" parameter in the response bodies of the "get tasks", "get task" resources for the "Garbage collection" task was extended by the "event_descriptors" and "face_-descriptors" parameters for cases of deleting face and body descriptors by version and type.

- The "tasks" > "content" > "options" > descriptor_version" parameter for the "Additional extract" task was removed from the response bodies of the resources "get tasks", "get task" for the case when basic attributes were used as extraction ("extraction_target" = "basic_attributes").

- Missing "Content-Type" headers were added to the "create account", "patch account", "create token", "replace token" and "verify credentials" resources of the OpenAPI specification.

- The error was fixed, in which the Image Store service did not start and did not issue any logs if the S3 storage URL scheme was not specified. Now, with such an error, the corresponding logs will be returned.

LUNA Index Module

Starting with this build, new LUNA Index Module (LIM) is available for LUNA PLATFORM 5, which significantly speeds up the matching of a large number of descriptors. The module pre-builds indexes on

a set of lists of faces and performs a matching on them. The user either specifies the lists for processing himself, or sets up automatic processing of all existing lists in LP.

LIM repeats the basic functionality of the previously existing "Index building and index search" services, while its performance is higher. Unlike the old services, LIM does not require an SSH connection due to the processing of the index delivery mechanism. At the moment, LIM lacks the index completion functionality available in the previously existing "Index building and index search" services, i.e. if a face is attached to the list after the index has started to be built, then it will not appear in the matching results. This functionality is in the works and will be available in the upcoming releases. If this functionality is critical, it is recommended to wait for its release in the next releases.

LIM is delivered as a separate distribution package containing an administrator manual with information about new services and their operation, installation manuals (manual and using Docker Compose), OpenAPI specifications and Docker Compose scripts. More detailed information about the distribution package can be found in the document "LIM_Quick_Start_Guide.pdf" from the LIM distribution package.

To use LIM, a separate parameter is required in the LUNA PLATFORM 5 license key. You need to contact VisionLabs for the possibility of adding a new parameter for working with LIM to an existing license key.

LIM works with all versions of neural networks extracting descriptors.

The previous "Index building and index search" remain available until 2023, after which support will be discontinued and the services will be removed from distribution.

For more information, see the LIM administrator manual in the corresponding distribution package

LUNA PLATFORM v.5.31.0

Changes

- SDK was updated to version 5.10.0.

- Support for the 60th neural network model for extracting face descriptors and the 105th, 106th and 107th neural network models for extracting body descriptors was added. By default, models 59th and 104th are used.

  The 60th neural network model is not supported when using index building and index search services.

  To reduce the size of the Handlers container, the 54th, 56th and 57th neural network models for extracting face descriptors and the 102nd, 103rd and 104th neural network models for extracting body descriptors will be removed from the delivery in the next LP 5 builds. To use them, you will need to download the necessary neural network separately and place it in the Handlers service container. This procedure is described in the section "Use non-delivery neural network model" of the LP 5 administrator manual.

- Support for the HumanFace detector was added. Detector is intended for simultaneous detection of the face and body in the image.

  The detector is used automatically when performing POST requests for resources "handlers/{handler_-id}/events" and "/tasks/estimator", if both "detect_face" and "detect_body" options are specified in the handler. At the same time, the speed of image processing is significantly increased compared to the use of separate detectors.

  In this regard, the names of fields in the Influxdb used for monitoring were updated in the Handlers service:

  - detection_width -> face_detection_width (face_detection series);
  - detection_height -> face_detection_height (face_detection series);
  - detection_width -> body_detection_width (body_detection series);
  - detection_height -> body_detection_height (body_detection series).

- The Accounts service was added to the LUNA Dashboards monitoring data visualization tool.

Fixed errors

- The error was fixed, due to which the Handlers service in some cases did not return the connection to the Postgres database connection pool and could output the error with the code 10017 (Database connection timeout error) until the service was restarted.

- The "logbook" dependency was removed from the "migrate_4_to_5.py" samples migration script, the absence of which resulted in the "ModuleNotFoundError: No module named 'logbook'" error.

- The absence of the "remove_image_origins" field description was fixed in the "create gc task" request in the OpenAPI documentation of the Admin service.

- The error was fixed, in which an incorrect API version was returned in the response body of the "/5/accounts" request for the Backport4 service.

LUNA PLATFORM v.5.30.0

Changes

- New mechanism was added to implement the role model in LP 5. It provides the ability to create accounts with a certain data visibility area and issue tokens for them with differentiation of rights to access LP resources within the account for which the token was created.

  Now, most requests require an account, except for requests that do not require authorization.

  All accounts created using the Admin service will be automatically migrated. To save the ability to work with data previously created using the "Luna-Account-Id" header, you need to create a new account, specifying the previously used "account_id" when creating it. For the Backport 3 service, you need to run the migration script separately.

See a detailed description of the new authorization system and account migration below or in the "Accounts, tokens and authorization types" section of the administrator manual.

Accounts

The account is required to delimit the visibility areas of objects for a particular user. Each created account has its own unique "account_id".

The account can be created using a POST request "create account" to the API service, or by requesting "register account", or using the Admin user interface. When creating the account, you must specify the following data: login (email), password and account type.

The account type determines what data is available to the user.

– user — the type of account with which you can create objects and use only your account data.
– advanced_user — the type of account for which rights similar to "user" are available, and there is access to the data of all accounts. Access to data from other accounts means the ability to receive data (GET requests), check their availability (HEAD requests) and perform comparison requests based on data from other accounts.
– admin — the type of account for which rights similar to "advanced_user" are available, and there is also access to the Admin service (see "Admin service" below).

Using the "Luna-Account-Id" header in the "create account" request, you can set the desired account ID. It should also be used if it is necessary to preserve the ability to work with data created in LP versions up to 5.30.0 (see "Migrating accounts from previous builds of LP 5" below).

Tokens

Token is linked to an existing account with any type and enables you to impose extended restrictions on the requests being made. For example, when creating the token, you can give the user permission only to create and modify all lists and faces, or you can prevent the use of certain handlers by specifying their ID.

The token and all its permissions are stored in the database and linked to the account by the "account_id" parameter.

When creating the token, you can set the following parameters:

– expiration_time – expiration time of the token in RFC 3339 format. You can specify an infinite token expiration time using the value "null"
– permissions – permissions that are available to the user
– visibility_area – token visibility of data from other accounts (see the section "Viewing other account data" in the administrator manual)

See the list of all permissions and detailed information on tokens in the "Permissions set in token" section of the administrator manual.

Authorization types for accessing resources

There are three types of authorization available in LUNA PLATFORM:

- **BasicAuth**. Authorization by login and password (set during account creation).
- **BearerAuth**. Authorization by JWT token (issued after the token is created).
- **LunaAccountIdAuth**. Authorization by "Luna-Account-Id" header, which specifies the "account_id" generated after creating the account (this method was adopted as the main one before version 5.30.0).

LunaAccountIdAuth authorization has the lowest priority compared to other methods and can be disabled using the "ALLOW_LUNA_ACCOUNT_AUTH_HEADER" setting in the "OTHER" section of the API service settings in the Configurator (enabled by default). In the OpenAPI specification the "Luna-Account-Id" header is marked with the word **Deprecated**.

Admin service

The account can still be managed using the Admin service. The service's user interface was updated. Now the account creation dialog contains an expanded set of fields.

The old administrator account was converted to the account with the "admin" type. The default login is now **root@visionlabs.ai** instead of **root**.

When deploying LP 5 from scratch, the database is no longer created for the Admin service. Now account data is stored in the Accounts database (see "Admin database migration" below).

Migrating accounts from previous builds of LP 5

Previously, the account was created using the "/accounts" resource of the Admin service or the user interface of the Admin service.

All accounts created before the current version will be automatically migrated when updating to the current version. The administrator account will be assigned the type "admin", and the accounts created when requesting the resource "/accounts" will be assigned the type "advanced_user". The email address will be used as the login and password. The name of the organization will be written in the "description" field.

It was also previously possible to make requests to LP 5 services by specifying the "account_id" in the "Luna-Account-Id" header. In order to preserve the ability to work with data created earlier by specifying the "account_id" in the "Luna-Account-Id" header, it is necessary to specify "login", "password", "account_type" and the old "account_id" in the "Luna-Account-Id" header in the account creation request. Thus, the old "account_id" will be linked to the account being created.

Admin database migration

Previously, account data was stored in the Admin database.

If the LP is deployed for the first time, the Accounts database will be created with which the Accounts service interacts. If LP is updated from version 5.28.0 or lower, the Admin database will

be converted to work with accounts, and the Admin service will no longer interact with the Admin database.

The section "Admin DB transformation" was added to the LP 5 upgrade manual, as well as the section "Account creation using API service", where an example of a CURL request for creating a new account and migrating the account if its "account_id" was used in the "Luna-Account-Id" header without creating account in the Admin service.

Migrating accounts from previous versions of Backport 3

When upgrading from the previous version of Backport 3, to migrate accounts, you must additionally run the migration script after migrating the Backport 3 database (see the "Accounts and tokens migration" section in the LP 5 upgrade manual).

Migrating accounts from LP 3.3.8

When upgrading from LUNA PLATFORM 3.3.8, accounts will be migrated when running the "start_-migration.py" script (see "Migration script description" in the LUNA PLATFORM 3.3.8 migration manual).

Migrating accounts from LP 4.5.4

When upgrading from LUNA PLATFORM 4.5.4, accounts are migrated in the same way as described in "Migrating accounts from previous builds of LP 5" and "Admin database migration" above. The section "Admin DB transformation" was added to the LUNA PLATFORM 4.5.4 migration manual, as well as the section "Account creation using API service", where an example of a CURL request for creating a new account and migrating the account if its "account_id" was used in the "Luna-Account-Id" header without creating account in the Admin service.

Backport 4 service

The ability to authorize by login and password (BasicAuth) and by account ID (LunaAccountIdAuth) was added to the Backport 4 API . Authorization by token (BearerAuth) is not possible.

The User Interface 4 service now uses login and password authorization ("BASIC_AUTH" parameter in the .env file) instead of "account_id" ("LUNA_ACCOUNT_ID" parameter in the .env file). Login and password must be specified in the "login:password" format in Base64.

- The number of received and deleted faces, tasks, handlers, events and LP settings specified in the "page_size" parameters of various resources and services was increased from 100 to 1000.

- New section "EXTERNAL_LUNA_API_ADDRESS" was added to the settings of the Handlers and Tasks services, intended to correctly process links to objects created using the "/images" and "/objects" resources in the API service. This section specifies the address and API version of the API service.

If as input for resources "/detector", "handlers/{handler_id}/events", "/iso", "/sdk" and "verifiers/{verifier_id}/verification" specifies the URL and version of the API service of the

"images" type object that matches the address and version of the API from the "EXTERNAL_LUNA_-API_ADDRESS" section of the Handlers service settings, then these objects will be loaded using the Image Store service directly, and not send a request to the API service from subsequent redirection to the Image Store service.

Format example: "http://10.15.3.144:5000/6/images/141d2706-8baf-433b-82eb-8c7fada847da", where "http://10.15.3.144:5000" must match the value from the "origin" setting ", and the value 6 must match the value of the"api_version" setting in the "EXTERNAL_LUNA_API_ADDRESS" section.

If the "content" > "source" > "reference" parameter of the resource "/tasks/estimator" specifies the URL and version of the API service of the "object" (ZIP archive) type object that matches the address and version of the API from the section "EXTERNAL_LUNA_API_ADDRESS" of the Tasks service settings, then this object will be loaded using the Image Store service directly, rather than sending a request to the API service with subsequent redirection to the Image Store service.

To avoid errors, you must configure this section in the Handlers or Tasks settings before using URLs to objects of type "objects" or "images" as an input source.

- Vendor libraries for HASP key and HASP utility were updated.

Now the "haspvlib_x86_64_111186.so" and "haspvlib_111186.so" libraries are replaced by the "haspvlib_x86_64_30147.so" and "haspvlib_30147.so" libraries, and the "aksusbd-8.23-1.x86_-64.rpm" utility is replaced by the "aksusbd-8.43-1.x86_64.rpm" utility.

When upgrading to a new LP build, you need to remove the old libraries and the HASP utility and install the new ones. The Licenses service will not start if the utility and libraries of previous versions are used.

You also need to contact technical support to reissue the LUNA PLATFORM 5 key.

Fixed errors

- The error was fixed, in which the request to the "4/healthcheck" resource returned the error "Internal server error" when the Tasks service was disabled in the "ADDITIONAL_SERVICES_-USAGE" section of the Configurator service.

- The error was fixed, in which the request to the resource "/luna_sys_info" returned the error "Internal server error" if the license file did not have the function of performing image checks for compliance with standards in the resources "/iso", "/detector", "/handlers" and "/verifiers".

- The error in the OpenAPI specification was fixed, in which the "get task result" and "generate events" response schemes displayed the type "Nullable" for some "face_quality" thresholds.

LUNA PLATFORM v.5.28.0

Changes

- SDK was updated to version 5.9.0. Key SDK changes affecting LUNA PLATFORM 5:
  - Updated Liveness V2
  - Updated face detector FaceDetV3

  The default value of the "score_threshold" setting in the "FACE_DETECTOR_V3" section of the Configurator service was changed from **0.89** to **0.42**. Migrating settings will automatically update this value (see "Configurator database migration" in the upgrade manual). Check the face recognition logic if you are using the "score_threshold" value other than the default value.

Fixed errors

- The error was fixed that ignored the "real_threshold" in Liveness checks on "/liveness" and "/sdk" resources. This error could lead to unexpected system behavior when estimating Liveness.

LUNA PLATFORM v.5.27.0

Changes

- New query parameter "estimate_face_quality" was added to the "/detector" resource which allows performing all "face_quality" checks from the "/handlers" and "/verifiers" resources with default thresholds.

  You can perform checks on this resource using the appropriate licensed feature.

- New "background_lightness" and "background_uniformity" checks were added to the "face_quality" checks group of the "/handlers" and "/verifiers" resources and to the "/iso" and "/detector" resources.

  The "background_lightness" check enables you to determine the lightness of the background from 0 to 1, where:

  - [0…0.1] — black background
  - [0.1…0.3] — dark background
  - [0.3…0.97] — light background
  - [0.97…1] — white background

  The "background_unformity" check enables you to determine the degree of background uniformity from 0 to 1, where 0 — the background is non-uniform, 1 — the background is uniform.

  New checks were also added to the "face_quality" field of the event structure in the "ws handshake" resource.

  Use of these checks requires the appropriate licensed feature.

- It is now possible to change the "quality_threshold" for Liveness V2, which is set by default in the system. To do this, use the corresponding setting in the "LUNA_HANDLERS_LIVENESS_SETTINGS"

section of the Configurator service. Previously, this threshold could not be changed for "/sdk" and "/liveness" resources.

The default threshold value is 0.5.

Previously, this threshold was located in the "config.py" setting of the Handlers service and had the "LIVENESS_V2_QUALITY_THRESHOLD" name.

When using "/handlers" or "/verifiers" resources, the threshold value from the Configurator will be redefined by the corresponding setting in the request parameter.

- The ability to specify body attributes in the "filters" and "targets" fields was added to the request body for event statistics.

- The monitoring of the Handlers service was extended. Now, in addition to monitoring requests (Requests series), requests completed with an error (Errors series), number of SDK estimations performed (Usages_statistic series) and licensing (Licensing series), data on each SDK esimation performed is monitored.

  For example, for each estimation of the mask presence on the face, the execution time of this estimation in seconds is monitored.

  See the "Monitoring" section of the Handlers developer manual for details and a list of estimates for which data is monitored.

Fixed errors

- The error was fixed that caused the URLs of face samples and attributes to be returned in the wrong format when requesting the "perform verification" resource.

- In the OpenAPI specification, the response scheme of the "stats" field in the response body to the "get statistics on events" request was fixed.

  It was previously specified that arrays of the "string", "number", "integer", "boolean", "Array of any" and "object" types can be returned as values for the "stats" field. The "stats" field is now specified to return a non-empty array containing arrays of the "integer", "number", and "string" types.

- Fixed the "internal server error" that occurred when processing images that had the NaN value in the coordinate fields of the EXIF data.

LUNA PLATFORM v.5.26.0

Changes

- SDK was updated to version 5.8.0.

- The mask state estimation was updated. In addition to the three main states, the following additional properties are now estimated:

- "correct" — there is mask on the face, the mouth and nose are occluded by the mask
- "mouth" — there is mask on the face that occludes only the mouth
- "clear" — there is no mask on the face
- "chin" — there is mask on the face and is located under the chin, without occluding the area from eyes to mouth
- "partially" — the face is partially occluded, but not by medical mask and not by mask with full face occlusion
- "full" — there is mask on the face, in which the face is completely occluded, for example, balaclava/ski mask

Each basic mask state corresponds to one of two properties. The most likely property is returned in the "predominant_occlusion" field:

- "medical_mask" state corresponds to the "correct" or "mouth" property
- "missing" state corresponds to the "clear" or "chin" property
- "occluded" state corresponds to the "partially" or "full" property

For each of the properties, a probabilistic score is returned in the range [0..1].

Additional mask properties are not saved to the database and no filtering is performed on them.

Note: The presence of the chin mask from this version refers to the "missing" state. In previous versions, it referred to the "medical_mask" state. Additional properties are returned when estimating mask states in the "sdk" and "detect faces" resources, as well as when creating events ("save event" and "generate events" requests) and performing verification ("perform verification" request). Additional mask properties are also taken into account when sending events via web sockets ("ws handshake" request).

Example of estimating the updated mask state in the "/sdk" resource:

```
"mask": {
    "predominant_mask": "occluded",
    "estimations": {},
    "face_occlusion": {
        "predominant_occlusion": "correct",
        "estimations": {
            "full": 0.019,
            "clear": 0.02,
            "correct": 0.6108324766,
            "partially": 0.31,
            "mouth": 0.0209,
            "chin": 0.019097
        }
    }
},
```

- The body attribute columns "apparent_gender", "apparent_age", "headwear_state", "sleeve length", "upper_clothing_colors" and "backpack_state" were added to the report structure in CSV format (see "reporter" and "exporter task" tasks).

- For the "num_threads" parameters from the "LUNA_HANDLERS_HUMAN_EXTRACTOR_RUNTIME_- SETTINGS", "LUNA_HANDLERS_WARP_ESTIMATOR_RUNTIME_SETTINGS", "LUNA_HANDLERS_- EXTRACTOR_RUNTIME_SETTINGS", "LUNA_HANDLERS_DETECTOR_RUNTIME_SETTINGS" and "LUNA_HANDLERS_HUMAN_DETECTOR_RUNTIME_SETTINGS" settings, the default value was changed from 4 to 6. This change improves the default performance for servers with a large number of CPU cores.

Fixed errors

- The error was fixed in which in tasks it was possible to set empty values for the filters "handler_ids", "masks", "face_ids", etc., which led to filtering by all relevant objects.

- The absence of the error when entering an incorrect login/password in the Admin service user interface was fixed.

- Now, in the absence of the licensed feature, the name of the missing feature is displayed in the Licenses service logs.

LUNA PLATFORM v.5.25.0

Changes

- The ability to define body attributes was added to the "/sdk" resource and the "detect_policy" of the "/handlers" resource (see the SDK documentation for more information about body attributes).

  The following attributes can be estimated:

  – gender (male, female, undefined) and age by body image. The estimation of gender and age in this way is less accurate than estimation by face image.
  – presence of headwear, type of sleeves (long, short, undefined), clothing color (black, blue, green, gray, orange, purple, red, white, yellow, undefined). The following colors will be added in future updates: brown, pink, khaki, beige, multi-colored.
  – presence of backpack.

  A separate license is required to use the new functionality. You can get information about the license status by requesting the "/luna_sys_info" resource ("license_info" > "body_attributes" field).

  New body attributes were added to the event structure (see "detections" > "samples" > "body" > "detection" > "attributes").

  Example of estimating body attributes in the generated event, with "estimate_upper_body", "estimate_accessories" and "estimate_basic_attributes" parameters enabled in the handler:

```
   "attributes": {
       "basic_attributes": {
           "apparent_age": 25,
           "apparent_gender": 0
       },
       "upper_body": {
           "headwear": {
               "state": 0
           },
           "sleeve": {
               "length": "short"
           },
           "upper_clothing": {
               "colors": [
                   "white",
                   "black"
               ]
           }
       },
       "accessories": {
           "backpack": {
               "state": 0
           }
       }
   }
```

The "apparent_gender", "apparent_age__gte","apparent_age__lt","headwear_states","sleeve_-
lengths","upper_clothing_colors", and"backpack_states" body attribute filters specified when
using events as candidates were added to the "match_policy" of the "/handlers" resource and the
"human body matching" resource. The ability to set body attributes as values for the "targets"
field was also added.

Body attribute filters can also be used when sending events via web sockets (see the "ws
handshake" resource).

Appropriate event filters were added to the Cross-matching, Clustering, Reporter, Exporter and
Linker tasks.

The ability to specify body attributes when creating new handler in the Estimator task is supported.

Body attributes can be aggregated when "generating" or "saving" events and performing
estimation in the "/sdk" resource. The aggregated attribute values are displayed in the "aggregate_-
estimations" fields of the respective resources.

- Now the "/healthcheck" resource is used for checking the status of a launched service container

when launching using start_platform.sh script.

Fixed errors

- Validation of events generated by the "save event" request was fixed. If checks from the "face_-quality" section were set in the event, then the event could be sent to the Sender service in the wrong format.

- The error was fixed where it was possible to set non-empty values for the "emotions", "masks" and "liveness" filters when specifying events as candidates in the "match_policy" of the "/handlers" resource.

- Fixed was the start_platform.sh fail upon changing the "DATA" variable value in the .env file.

LUNA PLATFORM v.5.24.2

Fixed errors

- The "reference_limit" setting of the Python Matcher service is now taken into account when matching with candidates of type "event_external_id". Ignoring this setting resulted in the "Database connection timeout error" error that occurred when matching a large number of faces with the same "external_id".

- The logging in LUNA Handlers service related to initialization, estimating and other events was fixed.

- The logging of the MatcherLib module of the Python Matcher service was fixed. Previously, logs were not written at the INFO level.

  Also, when Cached Matcher was enabled, the MatcherLib module saved logs to the "/tmp" directory and this behavior was not disabled using the "LUNA_PYTHON_MATCHER_LOGGER.LOG_-TO_FIE=false" setting.

- The work of the MatcherLib module of the Python Matcher service with version 52 of the neural network was fixed.

LUNA PLATFORM v.5.24.1

Changes

- The "link" field was added to the response bodies of requests performed with an error, containing a link to online-documentation with a detailed description of the error received. Links to error descriptions were also added to the response samples in the OpenAPI documentation.

  To follow the links, you need an Internet connection.

Fixed errors

- The message "sanic.error: <ApiRequest: OPTIONS /6/objects> body not consumed" was removed from the LUNA API service logs. The message was displayed if there was a non-empty body in the OPTIONS "/6/objects" request.

- Processing empty ZIP archives and directories from S3-like storage with the Estimator task no longer returns any errors. Previously, such processing returned an "Unsupported media type" error.

- The error that occurred when saving an event with the field value "detect_ts=0" was fixed. When trying to get an event with this value, the value "detect_ts=null" was returned.

- The error was fixed where updating a limitation or setting with an empty string in the "description" field in the Configurator service would result in "null" in the "description" field. The error occurred when using an Oracle database.

- "Internal server error" was fixed when saving an event ("save event" request) with "predominant_-mask": "medical_mask" value.

- The error was fixed where all estimators (DETECTOR, EXTRACTOR, WARP, HUMAN_EXTRACTOR, HUMAN_DETECTOR) used the values of the `device_class` ("gpu" or "cpu") parameter from the "LUNA_HANDLERS_DETECTOR_RUNTIME_SETTINGS" setting, ignoring the rest of the settings corresponding to the estimators, which led to the launch of all estimators GPU only or CPU only.

  Known Issue: The "num_threads" and "num_compute_streams" settings are still defined for all estimators in the "LUNA_HANDLERS_DETECTOR_RUNTIME_SETTINGS" section. Corresponding settings in other sections are ignored.

LUNA PLATFORM v.5.24.0

Changes

- Support for FTP server as an image source (the "source_type" parameter) was added to the estimator task ("/tasks/estimator" resource).

  For this type of source, the following parameters can be set in the request body for connecting to the FTP server:

    - host — FTP server IP address or hostname (required);
    - port — FTP server port;
    - max_sessions — maximum number of allowed sessions on the FTP server;
    - user, password — authorization parameters (required).

  As in Estimator tasks using S3-like storage or network disk as image sources, it is possible to set the path to the directory with images, recursively receive images from nested directories, select the type of transferred images, and specify the prefix and postfix.

To obtain correct results of image processing using the Estimator task, all processed images should be either in the source format or in the format of samples.

See the OpenAPI specification for related examples and more information.

- New "detect_ts" parameter was added for events, which enables you to store a timestamp relative to something, for example, relative to the beginning of a video file (see the response body of the generate events request).

  The value of the "detect_ts" parameter can also be set manually using the "save event" request.

- Memory consumption was significantly reduced for the Clustering, ROC-curve calculating and Cross-matching tasks, as well as for the Python Matcher and Image Store services.

  Also, to increase performance in the "cross matching faces" and "cross matching bodies" resources of the Python Matcher service, the Content-Type of the response body was changed from "application/json" to "application/msgpack".

- New "sorting" query parameter was added to the "cross matching faces" and "cross matching bodies" resources of the Python Matcher service to disable sorting of match results in lexicographic order.

- Support for the Vertica database for the Events service was discontinued. The settings for changing the database to Vertica was removed from the Configurator service settings, and the corresponding information was removed from the documentation.

- Checking for the presence of avx2 instructions on the processor when starting containers using the Docker Compose script was added. If the processor does not have avx2 support, then the "avx2 not supported" error will be displayed when starting containers.

Fixed errors

- The error was fixed where only real numbers were processed for some request parameters with type "number". The "Failed to validate input json" error occurred when entering integers.

LUNA PLATFORM v.5.23.1

Fixed errors

- The error was fixed, due to which invalid results were given when performing some estimates using 8-bit png images.

  It is relevant for the following resources:

    - detect faces
    - iso resource
    - generate events

- – perform verification
- – sdk

- The error was fixed in the Python Matcher service. The service did not restart after entering invalid parameters in the Configurator service settings when the `oracle` database type was specified as a `db_type` in the "LUNA_FACES_DB" section.

- The error in the "Thin Event" plugin example that caused the plugin to initialize twice was fixed.

- Now the Handlers service does not check the connection with the Events service if the second one is disabled in the "ADDITIONAL_SERVICES_USAGE" setting in the Configurator service.

LUNA PLATFORM v.5.23.0

Changes

- Support for a network disk as an image source ("source_type" parameter) was added to the Estimator task ("/tasks/estimator" resource).

  For this type of source, the following parameters can be specified in the request body:

  - – The "path" parameter — absolute path to the directory with images in the container (required);
  - – The "follow_links" parameter — enables/disables symbolic link processing;
  - – The "prefix" parameter — file key prefix;
  - – The "postfix" parameter — file key postfix.

  See an example of using prefixes and postfixes in the "/tasks/estimator" resource.

  When using a network disk as an image source and launching Tasks and Tasks Worker services through Docker containers, it is necessary to mount the directory with images from the network disk to the local directory and synchronize it with the specified directory in the container. You can mount a directory from a network disk in any convenient way. After that, you can synchronize the mounted directory with the directory in the container using the following command when launching the Tasks and Tasks Worker services:

```
docker run \
-v /var/lib/luna/current/images:/srv/images
```

  /var/lib/luna/current/images — path to the previously mounted directory with images from the network disk.

  /srv/images — path to the directory with the images in the container where they will be moved from the network disk. This path should be specified in the request body of the Estimator task (the "path" parameter).

As in the Estimator task using an S3-like storage as an image source, it is possible to recursively download images from nested bucket directories ("recursive" parameter) and select the type of transferred images ("image_type" parameter). To get correct processing results, it is necessary to use the same type of images (source image, face/body sample).

See the OpenAPI specification for related examples and more information.

- Two new image checks for compliance with the ICAO standard were added to the "face_-quality" group of checks of "/handlers" and "/verifiers" resources of API and Handlers services — "illumination_uniformity" and "dynamic_range".

The "illumination_uniformity" check enables you to check the uniformity of the illumination of the face in the image. The ICAO standard recommends the use of color images. When using black and white images, the results may be unexpected. It is recommended to use this check if it is necessary to get results that comply with the ICAO standard. In other cases, you can use the VisionLabs algorithm to check the uniformity of the illumination of the face in the image ("illumination_quality" check).

The "dynamic_range" check enables you to check the dynamic range of facial skin tone.

It is not possible to use a sample as an input image for these checks.

See the detailed description of the checks in the "Face and image parameters" section in the administrator manual.

- The default (recommended) thresholds were updated for the following checks in the "face_quality" check group and the "/iso" resource:

Check name	Old threshold values	New threshold values
mouth_occluded	min=0, max=0.3	min=0, max=0.5
mouth_open	min=0, max=0.64	min=0, max=0.5

It is recommended to set updated thresholds for these checks in previously created handlers.

- The execution time of requests for matching bodies was reduced when an event is specified as a reference.

Fixed errors

- The error was fixed where in the response body of the Estimator task result, the value "raw image" was returned in the "filename" field of the "detections" group instead of the file name.

LUNA PLATFORM v.5.22.0

Changes

- The SDK was updated to version 5.6.0.

- The "face_quality" group of checks was added to the "detect_policy" policy of the "/handlers" and "/verifiers" resources of the API and Handlers services, which enables you to configure the check of incoming images and faces in these images in accordance with predefined conditions. With "face_quality" group of checks it is possible to check face images in accordance with the requirements of ISO/IEC 19794-5:2011 or similar.

Available checks

The following face and image checks are available:

- **Head pose angles**: "head_yaw", "head_pitch", "head_roll"
- **Gaze angles**: "gaze_yaw", "gaze_pitch"
- **Presence of smile**: "mouth_smiling"
- **Presence of occluded mouth:** "mouth_occluded"
- **Presence of opened mouth:** "mouth_open"
- **Type of smile that is determined when there is smile** (no smile, closed-mouth smile, open-mouth smile): "smile_properties"
- **Eye status** for each eye (opened, closed, occluded): "left_eye", "right_eye"
- **Red eye effect**: "red_eyes"
- **Distance between eye centers**: "eye_distance"
- **Eyebrows state** (neutral, raised, squinting, frowning): "eyebrows_state"
- **Naturalness of lighting**: "natural_light"
- **Presence of radial distortion (Fisheye effect)**: "radial_distortion"
- **Highlights and specularity:** "specularity_quality"
- **Blurriness:** "blurriness_quality"
- **Contrast and saturation** (insufficient or too large exposure): "dark_quality", "light_quality"
- **Type of color by face** (color, grayscale, infrared (Near infrared range)): "face_color_type"
- **Position of face center point** on the image horizontally and vertically: "head_horizontal_center", "head_vertical_center"
- **Vertical and horizontal head size relative to size of image**: "head_width", "head_height"
- **Face width and height**: "face_width", "face_height"
- **Indents of face from image edges**: "indent_upper", "intend_lower", "intend_left", "intend_right"
- **Image size in bytes:** "image_size"
- **Image height and width:** "image_width", "image_height"
- **Image aspect ratio:** "aspect_ratio"
- **Image format** (JPG, JPG2000, PNG): "image_format"

– **Headwear** (none, baseball cap, beanie, peaked cap, shawl, hat with ear flaps, helmet, hood, hat, other): "headwear_type"

For a description of each check, see the "Face and image parameters" section in the administrator manual.

Enabling face_quality checks

Checks are performed for each face detection found in the photo. The results of the checks listed above are not aggregated. It is possible to enable and disable image processing with multiple faces using the "multiface_policy" option. To enable checks, you should specify the value "1" in the "estimate" field for "face_quality". Image check is disabled by default. To enable filtering based on the checks results, you should specify the value "1" in the "filter" field. If one or more checks for face detection fail, no further policies for that detection will be performed. In this case, the results obtained in accordance with the "detect_policy" policy for this detection will be returned in the response. Parameters of the "face_quality" check group:

```
"face_quality": {
    "estimate": 0,
    "filter": 0,
    "checks": {}
    }
```

The checks to be performed and their parameters are listed in the "checks" group.

Each check can be enabled or disabled. This enables you to specify the set of checks required for a particular scenario.

To disable checking, set `"estimate": 0` for a particular check. By default, all checks are enabled and will be performed when "face_quality" is enabled.

Depending on the type of check, the user can specify the minimum and maximum values of the threshold, or allowable values for this check. For this, the "threshold" field is used. An example of setting check parameters:

```
"checks": {
    "image_format": {
        "estimate": 1,
        "threshold": [
            "JPEG",
            "JPEG2000",
            "PNG"
        ]
    },
    "illumination_quality": {
```

```
                "estimate": 1,
                "threshold": {
                        "min": 0.3,
                        "max": 0.9
            }},
        }
```

Returned response

The name of the check ("name"), the defined value ("object_value"), the set thresholds/allowed values ("threshold_value") and the final verdict ("result") are output in the response to the request "handlers/{handler_id}/events" in "events" > "detection" > "samples" > "face" > "detection" > "face_quality".

Based on the results of all checks, a general verdict is displayed ("status" field), which is equal to "1" if all checks pass successfully and equal to "0" if at least one check fails.

The results of the checks are not stored in the database, they are returned only in the response.

An example of a system response for two checks:

```
"face_quality": {
    "status": 1,
    "checks": [
        {
            "name": "image_format",
            "object_value": "PNG",
            "threshold_value": [
                "JPEG",
                "JPEG2000",
                "PNG"
            ],
            "result": 1
        },
        {
            "name": "illumination_quality",
            "object_value": 0.5182177424430847,
            "threshold_value": {
                "min": 0.3,
                "max": 1.0
            },
            "result": 1
        }]
```

Licensing

This functionality is licensed separately, the ISO estimation option should be specified in the key. The "face_quality" parameter group and the "/iso" resource use the same license.

If the license does not have the ISO estimation option, then `"face_quality"> "estimate": 1` will return the error "License problem: 'ISO license feature is disabled.'".

Difference between checks for "face_quality" and "/iso" resource

The set of checks for "face_quality" and the "/iso" resource is different. See the "Image check" section in the administrator manual. The section provides a comparison table of available checks.

- The following new checks was added to the "/iso" resource:

  – Eyebrows state ("eyebrows_state")
  – Headwear ("headwear_type")
  – Type of smile that is determined when there is smile ("smile_properties")
  – Naturalness of lighting ("natural_light")
  – Presence of radial distortion (Fisheye effect) ("radial_distortion")
  – Red eye effect ("red_eyes")
  – Type of color by face ("face_color_type")

  These checks are performed in accordance with the default threshold values set in the system. If you want to change the threshold or disable the check, you should use the "face_quality" group of checks in the "/handlers" or "/verifiers" resources.

  For a description of each of the checks, see the "Face and image parameters" section in the administrator manual.

- The internal estimation mechanism for the Handlers service was updated, which significantly reduced the consumption of resources (CPU, GPU, RAM) for performing calculations. You can now run more instances of the Handlers service, or increase the number of workers per instance at the same capacity, resulting in faster query execution.

  In some cases, this change may cause performance degradation with the old Handlers configuration. In this case, you need to increase the number of instances or Handlers workers to improve performance.

- Matching by a large list (more than 100,000 faces) was accelerated with a large number of simultaneously performed requests. The speed of performing of such requests in some cases is increased up to two times.

- In the examples of the OpenAPI specification, the "exif" field is expanded with orientation information for all resources that support exif extraction.

LUNA PLATFORM v.5.21.0

Changes

- The SDK was updated to version 5.5.1.

- Support for S3-like storage as an image source (the "source_type" parameter) was added to the estimator task ("/tasks/estimator" resource).

  The following parameters can be set for this type of source:

  – Bucket name/Access Point ARN/Outpost ARN (required);
  – Endpoint (when specifying the bucket name);
  – Bucket region (when specifying the bucket name);
  – File key prefix. It can also be used to download images from a specific directory, for example, "2022/January".

  The following parameters are used to configure authorization:

  – Public access key (required);
  – Secret access key (required);
  – Signature version ("s3v2"/"s3v4").

  It is also possible to recursively extract images from the nested directories of the bucket (the "recursive" parameter) and save the source images (the "save_origin" parameter).

  See the relevant examples and additional information in the "APIReferenceManual.html" document.

- The performance of face processing in the Linker task was improved.

  Previously, the Tasks service requested a full set of targets from the Faces service, which reduced the faces processing performance. Now the service requests only the "face_id" target.

- The default number of connections to the LUNA Admin database was increased from 1 to 5 (the "connection_pool_size" setting, the "LUNA_ADMIN_DB" section in the Configurator service).

- Memory consumption was significantly reduced when using the cross-matching task.

Fixed errors

- The error of connecting to the service databases was fixed, in which in some scenarios the connections did not return to the pool of connections, which led to the return of the status code 500 and error 10017 (database connection timeout error).

LUNA PLATFORM v.5.20.0

Changes

- The ability to delete the source images associated with the event was added to the garbage collection task. To do this, use the "remove_image_origins" parameter, which is set in the request body.

The option was added to the following resources:

- "/tasks/gc" of the API service.
- "/tasks/gc" of the Admin service.
- "/tasks/gc" of the Tasks service.

The ability to delete source images was also added to the Admin service user interface: "Tasks" > "Garbage Collecting tasks" > "start descriptors gc" > "Remove image origins".

The system will try to delete images if they are stored in an external source.

- The "image_origins" field, containing the URL of all source images associated with the event being deleted, can now be returned in the response to an "event deletion" request from the Events service. These URLs can be used to delete the source images later.

  The "tagret" query parameter was added to the resource. You can specify the following values for it:

  - "face_samples".
  - "body_samples".
  - "image_origins".

  Depending on the specified targets, after deleting an event, the response will return information about the objects associated with this event: IDs of face/body samples, URLs of source images.

  By default, the response body displays information about all the objects associated with the event: "face_samples", "body_samples" and "image_origins".

  If you leave the "target" field empty, then only the "event id" of the event to be deleted will be returned.

- New reference types for "face matching" and "human body matching" resources were added:

  - "event_track_id" enables you to perform matching with each event containing the specified "track_id".
  - "external_event_id" enables you to perform matching with each event containing the specified "external_id".

  A search among several events related to the same face or body is performed when using these types. It reduces the effect of filming conditions (lighting, position relative to the camera, etc.) on the recognition accuracy.

  The "event_id" field is returned in the response for each event used as a reference.

- The "Advanced PostgreSQL setting" section with information about the configuration of PostgreSQL for working with LUNA PLATFORM was added to the administrator manual.

Fixed errors

- In the Events service, an error was fixed where special characters entered in the "label" ("matches" > "label") and "external_id" ("matches" > "candidates" > "face"/"event" > "external_id") fields were incorrectly processed in the "create new events" request.

- The error of aggregating the Liveness V2 estimation results for the "/liveness" resource was fixed. When sending one low-quality photo image to the resource, for which the system returned "prediction" = 2, a different value "prediction" was returned in the aggregated results for the same photo image.

LUNA PLATFORM v.5.19.0

Changes

- The maximum size of the archive transferred to the "/tasks/estimator" was increased from 1 GB to 100 GB. It is not recommended to transfer larger archives, as this may affect system performance.

- The execution of requests for getting faces ("/faces" resource) and getting the face count ("/faces/count" resource) when filtering by the list was accelerated in the Faces service.

Fixed errors

- The error was fixed when the Handlers service logs showed a message that the number of Liveness transactions is unlimited when Liveness was disabled.

LUNA PLATFORM v.5.18.0

Changes

- A new resource "/iso" was added to the API and Handlers services. The resource checks the face image for compliance with the requirements of the ISO/IEC 19794-5 standard. The response of the resource returns a verdict for each check (1 — the image corresponds to the standard, 0 — the value obtained does not correspond to the standard) and a general verdict based on all the checks performed. The total verdict is 1 if the image has successfully passed all the checks.

  For each check, the received value and the threshold/value with which the comparison is performed are returned. The thresholds are set in the system by the requirements of the ISO/IEC 19794-5 standard and are not changed by the user.

  Response example:

```
  {
   "name": "head_roll",
   "object_value": 5.434040069580078,
   "threshold_value": {
```

```
       "min": -8,
     "max": 8
   },
  "result": 1
  },
```

The following checks are now available:

- Quality of the image: contrast and saturation (insufficient or too large exposure), blurring, specularity, uniformity of illumination.
- Mouth state (opened, closed, occluded).
- Glasses state (no glasses, glasses, sunglasses).
- Eyes state (for each eye: opened, closed, occluded).
- Gaze.
- Head rotation angles (pitch, yaw and roll angles).
- Position of the face central point in the image horizontally and vertically.
- Vertical and horizontal head size relative to the image size.
- Distance between the centers of the eyes.
- Image formats.

The requirements can be found on the official website: https://www.iso.org/obp/ui/#iso:std:iso-iec:19794:-5:en.

In the request, you can additionally enable the extraction of EXIF data of the image.

By default, checks are performed for images that have one face present. You can enable estimation for multiple faces in the image using the "multiface_policy" parameter. For each of the found faces, the estimates and coordinates of the found face will be returned. It should be noted that many ISO checks assume the presence of one person in the frame, so not all checks for multiple faces will be performed successfully.

The order of the returned responses after processing corresponds to the order of the transferred images.

If one or more of the images transferred in the request is damaged, an error will be returned. The rest of the images in the request will be processed as usual.

This functionality is licensed separately. If there is no ISO option in the license, an error will be returned when using the "/iso" resource.

For more information, see the "Image checking according to ISO/IEC 19794-5" section in the administrator manual and the OpenAPI documentation.

- Removed were the "profile" and "limitations-file" options for the "db_create.py" script. The "profile" option was used to fill the Configurator service database with settings based on the default profile. The "limitations-file" option was used to load settings templates from a file.

Now it is recommended to use the "db_create.py" command to create an empty database without additional parameters, then load the necessary settings from the file using the "dump-file" flag or migrate the settings with the "python3.9 -m configs.migrate –config /srv/luna_-configurator/configs/config.conf –profile platform head" command.

- The vendor's library for the HASP key "haspvlib_x86_64_111186.so" was added to the distribution package and in the documentation.

LUNA PLATFORM v.5.17.0

Changes

- Functionality for gathering statistics on completed requests was added. The statistics received will enable the analysis of LP usage to further improve the system. The following data is gathered:

    - the number of requests for LP resources, for example, "POST: 'handlers/handler_id/events'".
    - the total number of estimations of the same type (for example, "estimate_liveness").
    - the number of successful requests and requests that failed with an error.

Statistics are maintained for each error code for each resource for each service. Statistics are sorted by month.

Statistics gathering works only when monitoring is enabled and InfluxDB of version 2.0.8 and later is installed.

Monitoring is now enabled in services by default. Sending the above data to monitoring is enabled by default. The information is impersonal and contains only quantitative data.

To get statistics, send a request to the "/luna_sys_info" resource of the Admin service or go to the "help" tab in the Admin service GUI and click "Get LUNA PLATFORM system info". The necessary information is contained in the "stats" section.

Response example:

```
{
    "stats": {
    "estimators_stats": [
        {
            "count": 1,
            "month": "2021-09",
            "name": "body_descriptor_extractor_usages"
        }
    ],
    "routes_stats": [
        {
```

```
            "service": "luna-api",
            "route": "GET:/version",
            "month": "2021-09",
            "errors": [
                {
                    "count": 1,
                    "error_code": "12012"
                }
            ],
            "request_stats": [
                {
                    "count": 1,
                    "status_code": "200"
                }
            ]
        }
    [...]
    ]
}
```

Statistics are gathered in InfluxDB based on data from the "luna_monitoring" bucket and stored in the "luna_monitoring_aggregated" bucket. The buckets are created in InfluxDB. Do not delete data from this bucket, otherwise, it will be impossible to get statistics.

Tasks for gathering statistics can be found in the InfluxDB GUI on the "Tasks" tab.

Statistics are gathered once a day, so they are not displayed immediately after the LP is launched. You can manually start the performing of tasks on the "Tasks" tab in the InfluxDB GUI.

To enable statistics gathering, run the `python influx2_cli.py create_usage_task --luna-config http://127.0.0.1:5070/1` command after launching the Admin service. The relevant information was added to the installation manual. The command automatically creates the necessary package "luna_monitoring_aggregated". If this command is not performed, the response "/luna_sys_info" will not display statistics.

If necessary, you can disable statistics gathering by deleting or disabling the corresponding tasks on the "Tasks" tab in the InfluxDB GUI. See "Requests and estimations statistics gathering" in "LP_-Administrator_Manual".

- Support for InfluxDB version 1 was discontinued. The parameters for this version were removed from the configuration files. Now InfluxDB 2 is used to monitor LP operation (starting from version 2.0.8).

If you previously used InfluxDB 1, you can migrate to a new version. A description of the migration is given on the official InfluxDB website:

https://docs.influxdata.com/influxdb/v2.0/upgrade/v1-to-v2/docker/

- Filtering by Liveness states (spoof, real, unknown) was added to the "/sdk" resource.

- Changes were made to licensing.

  For Liveness V2, the ability to license the number of completed transactions was added. Now, when licensing Liveness V2, you can choose between an unlimited license and a license with a limited number of transactions.

  Each use of Liveness in requests reduces the transaction count. It is impossible to use the Liveness score in requests after the transaction limit is exhausted. Requests that do not use Liveness and requests where the Liveness estimation is disabled are not affected by the exhaustion of the limit. They continue to work as usual.

  The HASP utility was updated to version 8.23. When upgrading to a new LP build, you need to remove the old HASP utility and install a new one. The Licenses service will not start if the utility of previous versions is used.

  When using Liveness V2 with transactions, you will need to update the license when switching to a new build, because previously issued licenses do not support this feature.

  Information about the current license parameters is displayed in the response to the GET request to the "/license" resource of the Licenses service, in the response to the request to the "/luna_sys_-info" resource of the Admin service and in the GUI of the Admin service on the "help" tab.

  See "Information about license" in "LP_Administrator_Manual" for details.

- The Licenses service no longer depends on other LP services other than Configurator (when getting settings from it) and can be run independently of them. But LP services, such as Faces, now depend on Licenses. In this regard, the order of launching services in the installation documentation was changed.

  Services now write license data to the monitoring database themselves:

  - The Faces service writes data about created faces with attached descriptors to the monitoring database in the "license_faces_limit_rate" field.

  - The Handlers service writes data on the number of available Liveness V2 transactions to the monitoring database in the "liveness_balance" field.

    A warning about the exhaustion of the number of available transactions is sent to the monitoring and logs of the service when the remaining 2000 transactions of Liveness V2 are reached (this threshold is set in the system).

  - The Licenses service writes data about the expiration date of the license to the monitoring database in the "license_period_rest" field. This behavior was used in previous builds as well.

  See sections "Licenses service" and "Information about license" in "LP_Administrator_Manual" for details.

Fixed errors

- The request to get faces with attributes in the Faces service was expedited. Previously, the Licenses service could incorrectly process the value of the FacesLimit parameter on large databases.

- In the Index Manager service, a problem was fixed when an index was not rebuilt. The index was manually deleted or was not built due to a service crash, but was listed in the database as existing, so LP did not rebuild it.

  Now in these cases, the status of lists with a missing index will be set as "failed". Next, the index building task will be added to the scheduler, and the index will be built.

LUNA PLATFORM v.5.16.0

Changes

- The "top_similar_external_id" field was added to the events, containing the "external_id" of the most similar candidate (event or face) with whom the face was matched.

  The "external_id" field was added to the "top_match" event object, which contains the "external_id" of candidates for matching in responses to the following requests:

  - "generate events".
  - "get event".
  - "ws handshake".

  These fields enable you to immediately receive the "external_id" of the most similar faces or events after performing requests. For example, previously, the event did not contain the "external_id" of the most similar face, either in the response to the request or in the event database. An additional request was required to determine the "external_id" of a face by his "face_id".

  Appropriate filters were added to the requests listed below to sort events by the external ID of the most similar object:

  - The "top_similar_external_ids" filter was added to the "match_policy" policy for events (see the policy description in the "create handler" request).

  - Filtering by the value of the "top_similar_external_ids" parameter for receiving events in API and Events services was added (see the "get events" request).

  - The "top_similar_external_id" filter and target were added to the request for getting statistics on events in API and Events services (see the "get statistics on events" request).

  - The "top_similar_external_id" filter was added when specifying events as candidates for "match faces" and "match bodies" requests.

  - The "top_similar_external_id" filter was added to the event filters for the following tasks:

* "linker task"
* "clustering task"
* "cross-matching task"
* "exporter task"

The "top_match" column in the results of reporter task and exporter task now contains the "external_id" of the top matching candidate.

- For candidates in the "matching faces" and "human body matching" requests, the "order" parameter was added, which determines the sorting order of the results. Sorting is available in ascending order of event creation time (option "create_time_asc"), in descending order of event creation time (option "create_time_desc") and by candidate similarity (option "similarity").

  Using the new sorting parameters enables, for example, to get events and faces in the order in which they transferred into the system. In this way, you can determine the first occurrence of the object.

  For the time sorting parameters to work correctly, you need to set a threshold based on the objects similarity. When specifying new sorting parameters, database matching is used. The request processing speed will be lower than when matching a list sorted by "similarity".

  By default, the "similarity" option is used.

- The "limit" parameter was added to the "clustering task", which enables you to set the maximum number of candidates returned for each matching. The default value was changed from 5 to 20 000.

  The parameter controls the number of candidates returned in the response. The more candidates come back, the better. But with a large cluster size, there may be performance and accuracy issues.

Fixed errors

- The error was fixed, in which the specified "targets" parameters were not returned in the results of Backport 4 requests for matching by indexed list, and the default set of fields was returned as "targets". The problem occurred in the Python Matcher Proxy service when performing a matching of one reference with several candidates.

- The image loading behavior was changed. Now each image is loaded independently. If an error occurs when loading one of the images, the loading of other images is not interrupted.

- The "result" field in the response to the "get task result" request for the estimator task is no longer required in the Tasks service.

  If an error occurs during the performing of one subtask, then only the "errors" field with an error will be returned in the response body. Previously, an empty "result" field was returned along with "errors".

- In the Tasks service, the limitations for markup the "ROC-curve calculating task" were fixed. The maximum value can now be set to 20 000 elements.

  Previously, when transferring more than 20 000 elements, the error "Uncaught exception occurred" was returned in the logs of the Tasks Worker service. An error with the level "Error" and a description of the error are now returned.

- The unused "CLUSTERING_MATCH_LIMIT" setting was removed from the Configurator service settings.

- The information returned in the "LUNA_CONFIGURATOR" section in response to the "get service configuration" request was fixed. It now contains the current state of the parameter.

- In the "EventsReferenceManual.html" document, the description for the following fields in the "create new events" request schema now specifies that the fields are not "Nullable":

  - matches/candidates/face/external_id
  - matches/candidates/face/user_data
  - matches/candidates/event/external_id
  - matches/candidates/event/handler_id
  - matches/candidates/event/user_data

- In the "EventsReferenceManual.html" document, the description for the following fields in the "get event" and "get events" request schemas now specifies that the fields are "Nullable":

  - top_match/event_id
  - top_match/face_id
  - match_result/candidates/event/event_id
  - match_result/candidates/event/create_time

- In the "EventsReferenceManual.html" document, the description for the following fields in the "face matching" and "human body matching" request schemas now specifies that the fields are "Nullable":

  - matches/result/event/top_match/event_id
  - matches/result/event/top_match/face_id

- In the "SenderReferenceManual.html" document, the description for the following fields in the "ws handshake" request schema now specifies that the fields are "Nullable":

  - event/matches/candidates/event/match_result/candidates/event/external_id
  - event/matches/candidates/event/match_result/candidates/event/user_data
  - event/matches/candidates/event/match_result/candidates/event/top_match/face/face_id
  - event/matches/candidates/event/match_result/candidates/event/top_match/event/event_-id

- In the "HandlersReferenceManual.html" document, the description for the following field in the "generate events" request schema now specifies that the fields are "Nullable":

– events/matches/candidates/event/match_result/candidates/event/user_data

LUNA PLATFORM v.5.15.0

Changes

- Dynamic handler support for the estimator task was added (see the "/tasks/estimator" request).

  Previously, the resource supported static handlers only. Now you can specify the ID of a static or dynamic handler in the `handler_id` field.

  For a dynamic handler, you can specify the policies directly in the request. The same dynamic handler can be used in different estimator tasks, and it is not required to create a new handler for each task.

  If you specify the ID of the static handler and try to specify policies, an error will be returned.

- A new "/configs" resource was added to all the LP services. It enables you to get the settings used by the corresponding service.

  If the service is launched with the option for automatic settings reloading and the settings are updated, then the resource will return the actual values and not the settings with which the service was initially launched.

  To get a list of settings, set the `Accept` header, which takes the values `application/json` (settings will be returned in JSON format for Configurator) or `text/plain` (settings will be returned in the format of the config.conf configuration file).

  Passwords and tokens specified in configuration files will be hidden in the received response.

- A new "/handlers/validator" resource was added. The resource enables you to check the correctness of the specified handler policies.

  The resource can be useful for checking the correctness of dynamic handlers.

- The Grafana docker container, which contains the scripts for creating LUNA PLATFORM dashboards, was added. See the LP docker installation manual for the description of container launching and dashboards creating.

- The instruction for using GPU when running LUNA PLATFORM using Docker Compose was added to the documentation.

Fixed errors

- The bug when an error code 500 was returned when generating events was fixed. The error was returned if the tag was set without specifying filters in the [conditional_tags_policy[policy (see the "/handles/handler_id/events" request).

- The description of the "page_size" parameter in the "get events" request was corrected in the OpenAPI documentation.

  Now it is specified that the number of objects cannot exceed 1000. Previously, the maximum value of 100 was specified.

LUNA PLATFORM v.5.14.0

Changes

- SDK was updated to version 4.5.1.

  – Liveness V2 algorithm was updated. The default thresholds for "liveness_threshold" and "quality_threshold" are now equal to "0.5". The recommended threshold for "liveness_-threshold" is now equal to "0.5" instead of "0.88".
  – The problem when SDK returned "Invalid detection" and "Invalid image size" in case of receiving an invalid bounding box was fixed. Now the "Invalid rectangle" error is returned in these cases.

- The support for matching plugins was added to the Python Matcher service.

  Plugins may significantly improve matching processing performance. For example, it is possible to organize the storage of the data required for matching operations and additional objects fields in separate storage using plugins, which will speed up access to the data compared to the use of the standard LUNA PLATFORM database.

  The general steps for custom plugin creation and an example of "Thin event" are given in the Python Matcher Proxy documentation ("ServiceManuals/PythonMatcherDevelopmentManual").

  "Thin event" is used for rapid comparison of face descriptors with descriptors of simplified events. Simplified events contain fewer fields compared to events from the "luna_events" database. All the data for them is stored in the same table.

  Requirements for the launch of "Thin event" are provided in its documentation. By default, the plugin is not used.

  Note that plugins are not provided as a ready-made solution for matching. It is required to implement the logic required for solving particular business tasks.

  The Python Matcher Proxy service should be installed for working with plugins. It decides on the use of standard LP matching mechanisms or plugins based on the complexity of the request (cost). The activation of plugins should be performed in this service.

  See the general description of plugins in the "LP_Administrator_Manual" in the "Matching plugins" section and detailed instructions with examples of plugins in the Python Matcher Proxy documentation in the "Plugins" section.

- The possibility to specify the end time for events using the "Luna-Event-End-Time" header was added to the "generate events" request. All the events will be generated with the "end_time" field if the header is specified.

  The field can be used to send event end time from external systems.

  The "end_time" field is supported for:

  - Specifying "target" for events in the matching request (see "generate events" and "create handler").
  - Saving events (see "save event").
  - Specifying "target" for receiving events and their statistics (see "get events" and "get statistics on events").
  - Returning events using WebSockets (see "ws handshake"). The creation time and end time will be returned in the "event-create-time" and "event-end-time" fields, respectively.
  - Columns in the CSV document in reporter and exporter tasks (see "reporter task" and "exporter task") Filters "end_time__gte" and "end_time__lt" are available for:
  - Matching policy, when events are specified as candidates (see "generate events" and "create handler").
  - Events receiving (see "get events").
  - Performing linker, cross-matching and clustering tasks (see "linker task", "cross-matching task", "clustering task").

- The "CONNECTION_POOL_SIZE" setting was added to Events, Faces, Tasks, Configurator, Admin, Handlers, Python Matcher, and Backport3 services. The setting enables you to set the size of the connection pool to the database.

Fixed errors

- The invalid field name for specifying the version of the descriptor in the requests that receive descriptors was fixed in the "APIReferenceManual". Previously, the field was called "descriptor_-version", now the field is called "version".

LUNA PLATFORM v.5.13.0

Changes

- The aggregation of received values of mask states, emotions and Liveness was added to the "/sdk" resource. When performing a request with the "aggregate_attributes" parameter enabled, the aggregated values of these estimations are returned to the "aggregate_estimations" field in the response body.

- The application/msgpack content type for creation of a face with attributes was added. See the "create face" request.

MessagePack packs data more efficiently than JSON. It is a binary format and therefore does not require decoding from BASE64.

Fixed errors

- An excess description of creating tasks to delete descriptors of the specified version for faces and events was removed from the OpenAPI document of the API service. Only event deletion is now available for the garbage collection task in the API service. See the "garbage collection task" request.

  The task of deleting descriptors of the required version for faces and events is available in the API of the Admin service.

- An unhandled error that occurred when the "account_id" field was incorrectly set in the "match_policy" when sending POST requests to the "/handlers" resource and PUT requests to the "/handlers/handler_id" resource of the Handlers service was fixed. The error occurred if "account_id" was not set in the UUID format.

LUNA PLATFORM v.5.12.2

Changes

- The request body description for faces creation with application/msgpack content-type was fixed in the Faces service.

LUNA PLATFORM v.5.12.1

Changes

- A new "/healthcheck" resource was added to LUNA PLATFORM services. The resource can be used to actively check the status of the service, namely, whether the service can perform its functions in full or not. The possibility of connecting this service to the LP and databases on which it depends is checked.

  - "/healthcheck API"
  - "/healthcheck Admin"
  - "/healthcheck Handlers"
  - "/healthcheck Events"
  - "/healthcheck Backport3"
  - "/healthcheck Backport4"
  - "/healthcheck Tasks"
  - "/healthcheck Image Store"
  - "/healthcheck Python Matcher"

- "/healthcheck Licenses"
- "/healthcheck Faces"
- "/healthcheck Sender"
- "/healthcheck Configurator"

It is possible to set up a periodic resource check using HAProxy, NGINX or another system. This will enable you to determine that the service is unavailable and decide whether to disconnect the service from the contour or restart it.

Using the "include_luna_services" option, you can enable and disable healthcheck for the LUNA PLATFORM services on which this service depends. If this option is enabled, additional requests are sent to the "/healthcheck" resources of these services.

The "include_luna_services" option is disabled in order not to perform recursive checking of the same services. For example, when several services on which this service depends at once will send requests to the Faces service and thereby increase the load on it.

If the healthcheck is successful, only the connection execution time in the "execution_time" field is returned.

If one or more services are unavailable, an error code 502 "Unhealthy" is returned. The response body lists the components, check statuses, and errors that have occurred.

The error code 500 in the response body does not necessarily mean a problem with the service. A long request may fail due to exceeded timeouts, increased server load, network problems or other reasons.

When performing a request to the "/healthcheck" resource, it is recommended to set a timeout of several seconds. If the request does not have time to be processed, this is a sign that problems have arisen during the operation of the system.

- The "account_id" parameter value is now used when receiving images from the Image Store service for the "detect face", "extract attributes", "generate events", and "perform verification" requests execution when they are sent directly to the Handlers service.

- Descriptions of migrations from LUNA PLATFORM 3 and LUNA PLATFORM 4 were corrected following the latest changes in LUNA PLATFORM 5.

LUNA PLATFORM v.5.12.0

Changes

- Now the filters "create_time__lt" and "create_time__gte" for matching candidates (faces and events) in the "match_policy" in handlers can be set relative to the current time. For example, this filter format enables you to select for matching events created in the last hour only. It can be

used for cases of face payment in transport to exclude re-payments. In this format, the time is set using the following template — `now-(\d+)[smhdwMy]`, where `\d+` is a number, `[smhdwMy]` is the required period: m (minutes), h (hours), d (days), w (weeks), M (months), y (years). Examples:

- The entry `"create_time__gte": "now-3h"` means that all objects created during the last three hours will be selected.
- The entry `"create_time__lt": "now-4w"` means that all objects created earlier than four weeks ago.

- A new "wait_saving" parameter was added to the "handlers/handler_id/events/raw" resource of API and Handlers services, which allows you to enable and disable waiting for events to be saved in the Events DB before sending a response. If the option is disabled, the response to the "handlers/handler_id/events/raw" request is returned faster, because the system will not wait for the event to be saved in the database. However, the system does not send any notifications if the saving failed. When this option is enabled, the system waits for events to be saved before sending a response. If the saving is successful, the status code 201 will be returned. If for some reason the event was not saved, the error code 500 will be returned. This option is enabled by default.

- A new "version" parameter was added to the request for receiving current LUNA PLATFORM settings from the Configurator service ("/dump" request), containing the migration version of these settings. The version changes when upgrading to new LUNA PLATFORM builds. You should not manually update the system settings from the previous build file, because the system settings are migrated automatically when you update to a new build. If you try to use a dump file from the previous LP build with different version, an error will occur. It is recommended to use files with settings from the previous build for checking the correctness of settings migration only.

- The repeated description of the WebSocket response message was removed from the "Callback" section from the Sender and API services OpenAPI documentation.

- The ability to specify the value "application/msgpack" as the "Content-Type" header for the following Events service requests was added:

  - GET to "/events"
  - PATCH to "/events/event_id". MessagePack packs data more efficiently than JSON. This is a binary format, so it does not require decoding from BASE64.

Fixed errors

- The bug when the license became unavailable the day before its expiration was fixed.

- An error with exceeding the maximum number of clients for the Vertica database was fixed in the LUNA Events service.

LUNA PLATFORM v.5.11.0

Changes

- A new "estimator" task has been added. It enables you to perform batch processing of images using the specified policies. This task can be created using the API of API or Tasks services.

  As a result of the task performing, JSON is returned with data for each of the processed images and information about the errors that have occurred.

  In the request body, you can specify the ID of an already saved handler or set processing policies manually.

  The resource accepts a link to a ZIP archive with images for processing. An external URL or the URL to an archive saved in the Image Store can be used as a link to the archive. In the second case, the archive should first be saved to the LP using a POST request to the "/objects" resource.

  The archive can be password protected. The password can be passed in the request using the "authorization" -> "password" parameter.

  To get correct processing results, use images of the same type (original image, face/body sample). The type of transferred images is specified in the request in the "image_type" parameter.

- The "remove_samples" parameter has been added to the event removing task ("/tasks/gc"), when enabled, face/body samples are removed along with events.

  Please note that after removing samples, it will become impossible to re-extract the basic attributes and descriptor for that event.

  This parameter has been added to the resources of the LUNA API, LUNA Admin and LUNA Tasks services, and can also be used in the user interface of the LUNA Admin service.

- When sending requests to other services, the API service can now send the accept-encoding header with the "identity", "deflate", "gzip" directives (previously, only the "deflate" and "gzip" directives could be specified). The "identity" directive enables you to disable automatic compression (on the side of other services) and automatic decompression (on the API service side) for request bodies with images or ZIP archives.

- The API service now reuses connections to other platform services. This change decreases the number of open connections and speeds up requests to other services.

- In the Events service, the response when deleting events has been changed. Previously, the IDs of deleted samples for faces and bodies were returned in one array — "samples". Now two arrays are returned in the response — "face_samples" and "body_samples". See the "event deletion" resource.

- The *skip_missing_descriptors* parameter has been added to LP 3 migration script "start_-migration.py". This parameter enables you to ignore the missing descriptors in the LP 3 database

during migration. Information about the script can be found in the "Migration launch" section of the "LP_Migration_from_LP3" manual.

Fixed errors

- Changes have been made to the processing of the "external_ids" parameter for the "/verifiers/verifier_-id/verifications" resource. Previously, the value of this parameter could only be specified in the "uuid" format, but now input in the "string" format is available.

- The error when the Backport 3 service stopped after automatically updating the service settings was fixed.

LUNA PLATFORM v.5.10.0

Changes

- A new "notification_policy" was added to the "storage_policy" set of policies of the "/handlers" service. It is used to enable and disable notifications sending about events creation to the Sender service. You can set filters for the notifications sending using this policy.

  The policy is enabled by default.

- "LUNA_HANDLERS_LIMITS" group of parameters was added to the Handlers settings.

  These parameters enable you to set:

    - Received images limit.
    - Max detections in raw event.
    - Arrays limit in raw event.
    - Matching result candidates limit.

  Previously, these values could only be changed using a special configuration file.

  It should be noted that an increase in limits can lead to problems in LP.

- All the values received for the "mask" and "emotions" parameters are now aggregated when "aggregate_attributes" is enabled in the request to the "handlers/handler_id/events" resource. The aggregated valuesof the parameters are returned in the response in the "aggregate_-estimations" group of parameters.

  The values defined for each detection are also returned to the response in the "Detections" group.

  The aggregated values of the parameters can be specified in the "aggregate_estimations" field when manually creating events using the "/handles/handler_id/events/raw".

  The aggregated parameters values are returned when receiving notifications about created events from the Sender service using WebSockets.

- SDK was updated to version 5.3.0.

  The "redetect_score_threshold" parameter value was set equal to "0.3".

- General description of working with LP plugins and basic information about their utilization was added to the administrator's manual in the "Plugins" section.

Fixed errors

- Truncated JPEG image files support was added. These images have not got end of image marker bites. Previously, an internal server error was returned when processing such images.

- The filter display error was fixed for the GC task in the Admin GUI. When the faces object was selected, a redundant "descriptor type" filter was displayed.

- An error with the POST request to the "/faces/attributes/descriptors" resource in the Faces service was fixed. The service returned a blank list of faces if there was a face with a descriptor, the version of which was equal to the version specified in the "missing_version" field.

- Fixed was an error with descriptor version validation in the Events services upon descriptors deletion. Now the version of descriptors will not be checked when deleting. Any specified versions of descriptors can be removed, including those that are no longer supported by the system.

- Error description when receiving an object with invalid Accept header has been fixed.

- Response schema for "luna_sys_info" request was fixed.

- A failure when automatically updating the Handlers service settings was fixed. The fail occurred when the settings for this service were changed in the Configurator service.

- Missing description of several monitoring fields was added to the Handlers service documentation.

- The "mimetype" field validation was fixed for the events generation request in the Handlers service. An internal error is no longer returned when image type specified in the request differs from the type of images provided in the request.

LUNA PLATFORM v.5.9.0

Changes

- Dashboards with information about LP requests and errors were added to the distribution package.

  Grafana is used to display dashboards. Use port 3000 to access the Grafana GUI.

  InfluxDB version 2 is required for dashboard creation. Dashboards are not working with version 1. InfluxDB 2 is now used by default during LP installation. InfluxDB 1 is still supported, but the recommended version is 2. The default token and password are used for connection to InfluxDB 2 during installation. Change them if necessary.

Grafana, Influx 2, and dashboards creation script launching were added to the installation manual.

You can find additional information about monitoring and dashboards in the "Monitoring" section of the administrator's manual.

- Resources "/objects" and "/objects/{object_id}" for objects storing and receiving were added to Image Store.

  The following request body schemas are available:

    - "application/json"
    - "application/pdf"
    - "application/zip"
    - "text/plain"

  The "/objects" resource provides the possibility to save an object of one of the listed types under unique ID in Image Store.

  The "/objects/{object_id}" resource enables you to:

    - Get an object.
    - Delete an object.
    - Check the object existence.

- The possibility to perform the GC task to delete face and body descriptors for events was added.

  You should specify "event_decriptors" as a "target". Next, you should specify the event descriptors type (face, body) and their version.

  Note that the "/tasks/gc" request body was changed. Now it is required to set "face_descriptors" instead of "descriptors" as a "target" for face descriptors deletion.

  Now it is possible to run events descriptors deletion using Admin GUI.

  The "/events/descriptors" resource was added to the Events service. It provides the possibility to delete event descriptors.

- "PLATFORM_LIMITS" parameters group was added to the Tasks and Python Matcher services. They provide possibility to set limits for "/matcher/face", "/matcher/body", "/tasks/clustering", and "/tasks/cross_match" requests. You can set the maximum number of:

    - References and candidates.
    - Filters values that are provided in the request.
    - Candidates returned in the response.

  Previously, these values could only be changed using a special configuration file.

  It should be noted that an increase in limits can lead to problems in LP.

- Python 3.9 is now used for Python Matcher, Admin, and Index Manager services. Older versions of Python are no longer supported.

- The list of libraries required for the migration from LUNA PLATFORM 3 to LUNA PLATFORM 5 is now stored in a separate requirements.txt file. It can be used for migration outside of the Backport 3 container.

- Attributes can now be specified as candidates for verification in the "/verifiers/verifier_-id/verifications" resource.

  The attribute has a limited period of existence, so verification will be impossible after the attribute is deleted.

  For example, this will enable you to write a pass for 24 hours (set the corresponding attribute TTL) and a person can use the pass only during this period.

- The following information was added to the OpenAPI documentation of the API service:

  - The task results examples were returned to the "/tasks/{task_id}/result" resource description.
  - The "exporter" task result description was added to the "/tasks/{task_id}/result" resource description. Select the "application/zip" response schema in the response description.

Fixed errors

- A bug when the "Exporter" task was performed without the delimiter specified in the query in the "csv_delimiter" parameter was fixed.

- Event IDs without attributes are now returned sorted in response to the "/events/attributes/missing" resource of the Events service.

- A bug when the sample ID was returned with the "Null" value in response to the DELETE request to the "/events" resource was fixed in the Events service.

- The error when changing the password in the Admin service using the PATCH request to the "/login" resource was fixed.

- The error with the `WARNING: sanic.root: Message body set in response on /2/ events. A 204 response may only have headers, no body.` message returned in the Events service log was fixed. The service no longer returns the request body when specifying gzip or deflate as an accept-encoding when status code 204 is returned or HEAD method is used.

- The database scheme for the Events service was updated in the documentation.

- Backport 3 service:

  - The status code for a successful response to POST "/handlers/verify/raw" was updated.
  - The response schema for the PATCH "/storage/persons/{person_id}" for status code 400 was added.
  - The response schema for the GET "/version" was updated.
  - The default values for the persons "user_data" and "external_id" fields were changed to the empty strings.

LUNA PLATFORM v.5.8.0

Changes

- The parameters for using Liveness V2 were added to the "detect_policy" of resources "/handlers" and "/verifiers". When Liveness estimation is enabled, the requirements of Liveness V2 for the incoming images should be considered. See "Liveness V2 Requirements" for details.

When Liveness V2 license is absent the "License problem: 'Liveness v.2 feature disabled'" error is returned in the response to the "handlers/handler_id/events" and "/verifiers/verifier_-id/verifications" requests. The "estimate_liveness > estimate" should be set to "0" when there is no Liveness V2 license.

Liveness estimation is not provided for the Backport 4 service handlers.

Liveness in handlers and verifiers

The "estimate_liveness" group of parameters provides the possibility to enable the estimation of Liveness and the status of the incoming image:

- "estimate" — enable Liveness estimation in the incoming images. The default value is set to "0" and Liveness V2 is not used.
- "liveness_threshold" — set Liveness threshold. The face in the incoming image will be considered real ("real" state) if the Liveness value is greater or equal to the specified threshold Otherwise, the face will be considered a spoof ("spoof" state). The default value is "0.88".
- "quality_threshold" — set the image quality threshold for Liveness estimation. If the quality is lower than the specified threshold, the "unknown" state will be set. Default value: "0". The "liveness_states" parameter enables you to set a filter by Liveness states. You should set Liveness states values. Only events with an estimated liveness state equal to one of the specified liveness states will be processed.

If an event is filtered by "liveness_states", the estimated face properties are returned for the face detection but the "extract_policy", "match_policy", "storage_policy", "conditional_tags_policy" policies will not be performed for the event.

One or several values from the list can be set for the "liveness_states" filter:

- spoof — 0

- real — 1
- unknown — 2

The default parameter value is set to "null", and filtration is not performed. The field is ignored if Liveness estimation is disabled in the "estimate_liveness > estimate".

The "liveness" filter was added to the following policies:

- "match_policy".
- "storage_policy" for all the objects.
- "conditional_tags_policy".

The "liveness" field was added to the "match_policy" as a possible "target" value.

Liveness results aggregation

When the "aggregate_attributes" option is enabled in the "handlers/handler_id/events" request, the aggregation of the Liveness estimation results will be performed for several processed images to receive more precise data. The data will be returned in the "aggregate_estimations" field.

The "aggregate_estimations" field is mandatory and is always returned in the response. The response also includes Liveness estimation values for each of the detections. When the "aggregate_attributes" option is disabled, the Liveness values for the face detection and values in the "aggregate_estimations" field will be the same.

When an event is saved, the aggregated Liveness result from the "aggregate_estimations" field is saved in the database.

New liveness field

The "liveness" field was added to events. The following features are provided:

- Receive events using the "liveness" filter and set "liveness" as a "target" in the GET "/events" request.
- Receive statistics on events and set "liveness" as a target and filter in the GET "/events/statistics" request.
- Use "liveness" as a filter for events in the request to the "/ws" resource and get "liveness" values in response to this resource.

The "liveness" field was supported as:

- A filter for events in "/matcher/face" and "/matcher/body" matching requests.
- A filter for events in cross-matching ("/tasks/cross_match"), clustering ("/tasks/clustering"), and linker ("/tasks/linker") tasks.
- A column for a report in the reporter task ("/tasks/reporter").

You can set liveness parameters values when manually save events using the "/handles/handler_-id/events/raw" resource. A separate liveness value can be set for each face detection and an aggregated value can be set.

- The "/tasks/additional_extract" request from the Admin service now supports the re-extraction of basic attributes and descriptors for faces and bodies saved in events. The transition to a new version of the neural network for events is now supported.

The following features were added to the "/tasks/additional_extract":

- Set events as objects for descriptors and basic attributes extraction.

    – Re-extract body descriptors for events.

You should specify descriptors (faces or bodies) or basic_attributes to be re-extracted. Then it is required to specify a filter for objects (faces or events) for which the descriptors should be extracted (for descriptors only).

The re-extraction of events attributes was added to the Admin user interface. You should select "Events" as "Object type" and set "Face" or "Body" as "Descriptor type".

The PATCH request to the "/events/{event_id}" resource for updating of basic attributes and descriptors of existing events was added.

The resources for receiving events with samples and attributes "/events/attributes/missing" and calculation of the number of such events "/events/attributes/missing/count" were added to the Events service.

See additional information about source images saving for events in the "Launch re-extraction task" in LP_Administrator_Manual.

- Python 3.9 is now used for API, Faces, Image Store, Tasks, Events, Configurator, Sender, Handles, Backport 3, Backport 4, Licenses services. Older versions of Python are no longer supported.

Python 3.7 and newer can still be used for "luna3" client library, the "folder_uploader.py" script for images downloading, and the scripts for migration from LP 3.

- The "image_origin" field was added to the POST request to the "handlers/handler_id/events" resource for "application/json" and "multipart/form-data" body schemas.

One can specify a link to the source image for each of the images provided in the request. The URL will be added to the "image_origin" field of the created event.

The image provided in the "image_origin" field will not be processed in the "handlers/handler_-id/events" request. It is used as a source image only.

If the "image_origin" is not empty, the provided URL will be used in the created event regardless of the "image_origin_policy" policy.

See additional information about source images saving for events in the "Saving source images" in LP_Administrator_Manual.

- The support of UTF8 symbols was added for EXIF tags. Recognition of Cyrillic was added.

Symbols in response are limited by ASCII encoding. Screening is used for all characters in EXIF tags, which are not included in the ASCII set.

An example of screening for the "artist" EXIF field: "'exif': {'artist':'\u041f\u043e\u043f\u043e\u0432'"".

Fixed errors

- Missing response schemas were added in the OpenAPI documentation of the Image Store service.

- Fixed was the error when EXIF tags were causing "Internal server error".

LUNA PLATFORM v.5.7.0

Changes

- The new resource "/tasks/exporter" was added with which you can collect event and/or face data and export them from LP to CSV file.

  The input is a set of filters to determine the objects that need to be exported to a file. The output returns a ZIP archive, which stores a CSV file with data about objects and images for each object (optional).

  This possibility is supported in API and Tasks services. See "Exporter task" section in Administrator's manual.

- The option to use external image URL when saving source image was added.

  The use_external_references setting was added to the policy for source image saving. With this setting, you can enable saving a link to an external image in the "image_origin" field to avoid duplicating the same image in the database.

  - If the request contains a sample and it was saved in the Image Store, then the field will contain a link to it.
  - If the request contains an URL to the image, then the field will contain the URL.

  If the URL is longer than 256 characters, the source image will be saved instead.

  This possibility is supported in API and Handlers services.

- The possibility to transfer the detection time along with images was added.

  In the request for creating events in the "handlers/handler_id/events" resource, the possibility to explicitly specify the time of face detection for each of the transferred images was added. It is available for "multipart/form-data" and "application/json" request content types.

  This possibility is required for cases when images are not sent immediately, but after a while.

- For API, Configurator and Backport 4 services, the settings reload mechanism was changed. The reload is now performed by restarting the respective processes. The mechanism provides a more reliable update of service settings.

  Note that requests made when the settings are changed may end with an error. The service may be unavailable for some time.

- When specifying a non-existent list_id in the clustering task, two errors are now returned, instead of one: "List with id {ID} not found" and "Objects for clustering not found (empty set)".

Fixed errors

- The error message "Check connection to Influxdb: connection refused" is now returned when the LP services connection to InfluxDB fails. Previously, instead of an error, the call list was returned before the error occurred.

- The slowdowns in the Image Store service were fixed.

- An error when the value of the face_bounding_boxes parameter passed in the request to the "/verifiers/verifier_id/verifications" resource was not processed has been fixed in the Handlers service.

- The URL format of an avatar for a face created using a handler was fixed in the Handlers service. Previously, it was returned in the "/samples/{sample_id}" format. The URL is now returned in the "/samples/faces/{sample_id}" format.

- The user_data and external_id fields were added to the OpenAPI specification of the Backport 4 service, which are returned in the response of creating events using the "handlers/handler_-id/events" resource.

LUNA PLATFORM v.5.6.0

Changes

- SDK was updated to version 5.2.0.

- The support for the 59 neural network version for the face descriptors extraction was added.

  Starting from build 5.6.0, the 59 version is used by default for new LP installations.

  If LP is already installed, the default neural network version will not be updated to 59 automatically. The currently used neural network version will remain in the Configurator service. In this case, it is required to perform re-extraction of the already existing face descriptors to update to the new neural network. The process description is given in the "Switch neural network version" section of the administrator's manual.

  Note that re-extraction of events descriptors is not available. When switching to a new neural network, existing events cannot be used for matching operations.

  The distribution of Index building and searching by index does not support the 59 neural network version. The neural network of version 56 is used by default for these services.

- The support for 102, 103, and 104 neural networks for the body descriptors extraction was added. The 104 version is now used by default.

  The neural network of version 101 that was used in the previous releases is not supplied and is not supported anymore. The already existing body descriptors cannot be re-extracted using the 104 neural network version when switching to LP build 5.6.0, and they cannot be used for matching operations.

- The "user_data" and "external_id" parameters values are now set to " " by default for all the services and cannot be set to null.

- The "event_id__gte" and "event_id__lt" filters were added to the "Cross-match", "Clustering", and "Linker tasks". The filters enable you to set the upper and lower bounds for the "event_id" parameter values. The subsequent processing will be performed only for the events whose IDs are included in the range specified by the filter. They can be used, for example, to divide events processing into several tasks and perform them in parallel.

- The configuration reload mechanics have changed in Image Store, Tasks, Events, Admin, and Backport 3 services. Now, it's done mostly by restarting appropriate processes. The mechanism provides a more reliable update of service settings.

  Please note that requests made at the time of changing the settings may end with an error. The service may be unavailable for some time.

Fixed errors

- An error with an incomplete log was fixed in the API service. The record about the request processing was not written to the log in case of client disconnect. Now the service will write a message about the request end with status code 499 if the client disconnected before the response.

- New possible status codes were added to services responses:

  - The service will return status code 408 if the service did not receive a complete request message within the specified period (60 seconds by default).

  - The service will return status code 503 if the service did not process a request within the specified period (600 seconds by default).

  - The service will return status code 413 if the request payload is higher than the service can process.

  The error with the SDK descriptor type was fixed in the OpenAPI specification for the API service. Previously, the SDK descriptor was displayed as an Object type with "descriptor" and "version" fields. Now the type of object is set as string `<byte\>`.

LUNA PLATFORM v.5.5.0

Changes

- New resource "/handles/handler_id/events/raw" was added to API and Handlers services. Request fields are filled in when sending the request.

  The format of the generated event is similar to the format returned by the "handlers/handler_-id/events" resource. The "event_id" and "url" fields are not specified when creating a request. They are returned in the response after the event is created.

  Notifications using web sockets are sent when events are created using this resource.

The resource enables you to set your logic for filling in event fields, which is different from the logic using handlers. For example, when you want to extract descriptors only for a part of the detections and not for all the detections.

- The resource "/ws" was added to the API service. Now web sockets configuration and Sender responses proxying are performed via the API service. This provides a single entry point to the LP through the API service and enables you to avoid sending requests directly to the Sender service.

  Configuring web sockets directly via Sender is still available. It can be used to reduce the load on the API service. In other cases, it is recommended to use the API service.

- The new filters "event_id__gte" and "event_id__lt" are supported for receiving events using the "/events" resource.

  Using these filters, you can perform pagination that is:

  – Faster than the pagination by "page" and "page_size" parameters. It does not slow down with a large number of events.
  – More stable than pagination by "page" and "page_size" parameters. When events number is changed during the pagination process, it does not cause events to be lost or duplicated in the response.

- The configuration reload mechanics have changed in Faces and Python Matcher services. Now, it's done mostly by restarting appropriate processes. The mechanism provides a more reliable update of service settings.

  Please note that requests made at the time of changing the settings may end with an error. The service may be unavailable for some time.

- The values of "user_data" and "external_id" fields are now set to "" by default.

- Cache sharing was added for the Python Matcher service when running multiple worker processes (workers) of the same service. Now each of the worker processes uses the same descriptors cache. Previously, when creating multiple workflows, each of them had a separate descriptors cache.

  This change can both speed up and slow down the service. If you need to ensure that the cache is stored in each of the Python Matcher processes, you should run each of the server instances separately.

- InfluxDB OSS 2.x was supported for monitoring for all LP services. The section "InfluxDB OSS 2", which describes how to configure LP to work with the second version of the database, was added to "LP_Administrator_Manual". The database in the LP package was not updated to the new version.

Fixed errors

- The face patch by "null" value of "event_id" field is supported in the Faces service in PATCH request to "/faces/{face_id}".

- The problem with the delimiter character escaping in the report columns has been fixed in the Tasks service. If the delimiter character is found in the report column, it is now escaped.

- Proper handling of the services communication errors that occurred during tasks processing was added to the Tasks service. For example, if the service is unavailable or there are problems with the connection, the corresponding errors are correctly displayed in the list of errors after the task execution.

LUNA PLATFORM v.5.4.0

Changes

- The "detect_time" and "image_origin" fields were added to events:
  - The "detect_time" field includes the time of detection on the source frame.
  - The "image_origin" field includes the UTL of the saved source image where the detection was performed.

  These fields are returned in the response on the POST request to the "handlers/handler_id/events" resource.

  When an event is stored, the fields data is added to the Events database in the "face_detect_result" and "body_detect_result" tables.

  The format of input (POST "/events") and output (GET "/events", GET "/events/{event_id}") data for events was changed in the Events service due to this update.

  These new fields are returned in the response on the "/ws" resource when the Sender service sends events.

- The policy for the source image saving was added to the "/handlers" resource.

  It is required to set the "store_image" parameter value to "1" for "image_origin_policy" to enable source images storage. Source images are not stored by default.

- Attributes can be now set as candidates and references for cross-matching task. See "/tasks/cross_-match".

  Attributes can now be used for ROC curves creation. See "/tasks/roc".

  Attributes usage provides the possibility to perform experiments (for example, to create a ROC curve by a given selection) and receive results without storing redundant data in the database.

- The "Stored and estimated data" section was added to the "LP_Administrator_Manual". It describes data that is estimated and stored by LP.

- The possibility to write LP services logs to the server was added when using Compose. Previously, when installing using Compose, it was impossible to save logs to the file. This feature was added

for standard LP distribution and not available for the services for index building and searching by index.

It is required to create directories for logs and set permissions for them (see "Create logs directory"). Logs cannot be written to the directories, and services will return errors if permissions are not set. After containers launching, logging settings should be updated in the Configurator service (manually or using the "logging.json" file) to enable logging to file (see "Enable logging to server directory"). Files will be saved in the "/srv/logs" directory in containers.

Logging to files is disabled by default. Logs are output to stdout only.

Fixed errors

- An issue with displaying full URLs for events and *faces* instead of relative URLs in the response to a POST request to the "handlers/handler_id/events" resource was fixed.

- The automatic setting of the time filter is disabled in the GET "/events" request if the event IDs are specified. Previously, the default value for this filter was set implicitly if it was not explicitly set in the request.

- The ethnicity table was fixed in the OpenAPI documentation of the Events service.

- The creation of a Docker container for the Handlers service was fixed. The container size decreased by 5 gigabytes.

LUNA PLATFORM v.5.3.0

Changes

- The possibility to save source images was added. The stored images can have the following formats: JPEG, PNG, BMP, TIFF, Portable pixmap.

  The "/images" resource is used for images storage.

  The "/images/{image_id}" resource is used for deletion of images and for receiving stored images.

  The images are stored in the "visionlabs-image-origin" bucket of the Image Store.

  The URL of the saved image can be specified upon execution of requests to the "/detector", "handlers/handler_id/events", "/verifiers/verifier_id/verifications" resources. The "Content-Type" header value should be set to "application/json".

  Example of the image URL: "http://server_ip:5000/6/images/10bc2cb4-db84-410d-adc7-ff2ac17e4b2d".

- The logic of the LUNA PLATFORM services launch inside containers was changed. Now applications are launched by the *luna* user instead of the *root* user.

This change was not applied to the containers of the following services: UI 3, UI 4, services for index building and searching by index.

- FaceDetV1 and FaceDetV2 detectors are not supported anymore.

FaceDetV3 will be automatically set as the detector used in the "LUNA_HANDLERS_DETECTOR_-TYPE" setting after the Configurator service settings migration if another detector was used. The migration is performed during the Configurator database update.

- The "wait_saving" option was added to the event storage policy ("storage_policy" > "event_-policy") of the "/handlers" resource. It enables and disables waiting for events to be saved in the Events database before sending a response.

  – When the option is disabled, the response from the "handlers/handler_id/events" resource is returned faster, but the system does not send any notifications in case of failure during the event saving. This behavior was set by default in the previous LUNA PLATFORM versions.
  – When the option is enabled, the system waits until the events are saved before sending a request. The 500 error code is returned when events saving fails.

The option is enabled by default.

The option will be automatically added to the "storage_policy" of already existing handlers during the Handlers service database migration. The option will be disabled for the already existing handlers, so the behavior of the system will not change. The migration is performed during the Handlers database update.

- The "application/msgpack" value can be now set for the "Content-Type" header of the following resources:

  – "/matcher/faces"
  – "/matcher/bodies"

MessagePack packages data more efficiently than JSON. It is not a binary format, so it does not require decoding from BASE64.

- CUDA version was updated to version 11.1 in the Handlers container.

- The "Switch to 46 or 52 neural network" section was updated in LP_Administrator_Manual.

- The speed of cross-matching tasks execution was increased.

Fixed errors

- The error during matching using the 58 neural network version was fixed.

- The error when authorizing using cookies was fixed in the Admin service.

- The error when opening the account page was fixed in the Admin service.

- The crash during the new line symbol processing in the request to the Events service was fixed. The error had occurred when the Vertica database was utilized. Now an error is returned when the "\n" symbol is found in the request.

LUNA PLATFORM v.5.2.1

Changes

- API service:

    – Time fields validation was fixed for all the resources utilizing `create_time__gte` and `create_time__lt` filters. An error occurred when time was set in UTC format (the "Z" suffix was used in the field). Example: "2018-08-11T09:11:41.674Z".

    – Fixed was validation for "match_policy" with events set as candidates. The wrong error message "unexpected value; permitted: 'faces'" was returned when filters for events were set incorrectly.

- Admin service:

    – The internal error with status code 500 was fixed for requests without any authorization data.

    – The error "BadAdminAuth" is now returned in the case of authorization with an incorrect login/password pair.

LUNA PLATFORM v.5.2.0

Changes

- LUNA PLATFORM now supports 57 and 58 neural networks for descriptors extraction. LUNA PLATFORM still uses the 56 neural network by default. See the "Switch neural network version" section of LP_Administrator_Manual for information about changing the default neural network version.

    The services for index building and searching by index do not support 57 and 58 neural networks in this LUNA PLATFORM build.

    Neural networks of versions 46 and 52 that may be required for backward compatibility are not provided in the distribution package. They are provided by VisionLabs upon request. See the "Switch to 46 or 52 neural network" section in LP_Administrator_Manual for information about adding these neural networks to the Handlers container.

- New Liveness V2 mechanism was added. Now it can be used in "/liveness" and "/sdk" resources.

    The Liveness that utilizes a separate service is now called Liveness V1.

Liveness V2 does not require a separate service. It is part of the Handlers service. Therefore, the "liveness" option in "ADDITIONAL_SERVICES_USAGE" should be disabled when using Liveness v2.

The new Liveness v2 and Liveness v1 can be used in the "/liveness" resource. The request (except for the metadata section) and the returned result in this resource will have the same format for both Liveness versions. The metadata section is not used for Liveness V2.

Liveness v1 is not used in "/sdk" resource. If a request with "estimate_liveness = 1" is sent on the resource when the Liveness V1 is utilized, an error occurs.

It is not allowed to use both Liveness versions at the same time. Liveness versions are changed in the LUNA PLATFORM license file. The following values are available:

- 0 — Liveness feature is not used
- 1 — Liveness V1 is used
- 2 — Liveness V2 is used

It is not required to update LUNA PLATFORM license if you already have one and you are not going to use Liveness V2. Otherwise, the existing key should be updated.

Liveness versions description is given in the "Liveness description" section of LP_Administrator_-Manual. Tables of the working conditions of Liveness versions for the "/sdk" and "/liveness" resources are given in the section.

See image requirements for Liveness V2 in the "Liveness V2 requirements" section.

- Images rotation:

    - The ability to automatically rotate the image using its EXIF data was added. This feature ("use_exif_info") was added to the following resources:

        * "/detector"
        * "/verifiers/verifier_id/verifications"
        * "handlers/handler_id/events"
        * "/sdk"

    The option is enabled by default.

    If the request specifies that the processed image is a sample, the option is ignored.

    - The "LUNA_HANDLERS_USE_AUTO_ROTATION" option was added to the configurations of the Handlers service. It enables you to turn on the automatic rotation mode of the image if it is rotated by 90, 180, 270 degrees.

    This neural network consumes a significant amount of server resources, so it is disabled by default.

    The option is not applied to the samples. If the request specifies that the processed image is a sample, but the image is rotated, then the automatic rotation of such an image will not be performed.

The two above options for automatic rotation and rotation based on EXIF data can be used together.

- The CORE Matcher service is no longer supported. All the operations for matching by lists are now performed using the Python Matcher service.

  To increase performance, the "cache_enabled" setting for this service should be set to "True" (set by default). In this case, the descriptors will be stored in RAM and processed faster by Python Matcher.

- When using an external PostgreSQL database (not the PostgreSQL container included in the package), you should recreate the matching function in the database with the addition of the "PARALLEL SAFE" directive. This directive speeds up database matching.

  This directive will be added automatically during the basic installation of the PostgreSQL container from the distribution package.

  Example of a string for deleting a function from the database: `DROP FUNCTION VLMatch;`.

  Example of a line for adding the function: `CREATE FUNCTION VLMatch(bytea, bytea, int) RETURNS float8 AS 'VLMatchSource.so', 'VLMatch'LANGUAGE C PARALLEL SAFE;`.

LUNA PLATFORM v.5.1.3

Changes

- Accept header was added for the "/sdk" resource. It enables you to specify the request content-type: JSON or MessagePack. MessagePack packages data more efficiently than JSON. It is not a binary format, so it does not require decoding from BASE64. See "sdk" > "sdk resource".

- Glasses estimation was added for the "/sdk" resource. See "sdk" > "sdk resource".

- Filters `list_id__gte` and `list_id__lt` were added for resources "/lists" and "/lists/count" (method "GET"). Received lists are ordered by "list_id".

  Using these filters, you can perform pagination that is:

  - faster than the pagination by "page" and "page_size" parameters. It does not slow down with a large number of lists
  - more stable than pagination by "page" and "page_size" parameters. When lists are changed during the pagination process, it does not cause the lists to be lost or duplicated in the response.

  See "lists" > "get lists"

  See "lists" > "get lists count"

- The "/lists/deletions" resource was added to the Faces service for tracking of the deleted lists. See "administration" > "get lists deletions".

The request for removing deleted lists log was added. See "administration" > "clear lists deletions log".

- The possibility to extract basic attributes for faces without basic attributes was added to the "/additional_extract" resource. Biometric samples should be available for the samples. See "tasks processing" > "additional extract task".

- The "wait_events_saving" query parameter was added for method POST on the "/events" resource. See "events" > "create new events".

  When this parameter is enabled, the system will wait for the event to be saved to the database before returning a response (this default behavior was used earlier).

  When this option is disabled, the system does not wait for events to be saved to the database. It sends a response immediately after validating the event and adding it to the buffer.

- The memory consumption when caching a large number of small lists was reduced for the Python Matcher service.

LUNA PLATFORM v.5.1.2

Changes

- A new workers command line argument was added for LP services. A service will automatically spin up multiple processes and route traffic between the processes. You can change the number of workers in Docker containers of services using the `WORKER_COUNT` parameter.

  It is recommended to use additional worker processes when increasing the number of service instances on the same server.

  It is not recommended to use additional worker processes for the Handlers service when it utilizes GPU. Problems may occur if there is not enough GPU memory, and the workers will interfere with each other.

  See the "Worker processes" section in LP_Administrator_Manual.pdf.

- Configurations reload support was added for LP services. If a setting value has been updated, it will be applied to the service without restarting it.

  This feature is enabled for services Docker containers using the `RELOAD_CONFIG` option.

  Configurations check period is specified using the `RELOAD_CONFIG_INTERVAL` option.

  You should carefully use this feature while changing important service settings (DB setting, work plugins list, and others). Do not send any requests to the service while applying these settings.

  See the "Automatic configurations reload" section in LP_Administrator_Manual.pdf and the "Configuration" section in "ServiceManuals" of each service.

- The "Matcher" setting was added to "ADDITIONAL_SERVICES_USAGE" config section. It is used to enable CORE Matcher service utilization.

- The functionality `missing basic attributes` was added to the Faces service. The functionality enables you to:

  - Get faces that do not have basic attributes "administation" > "get faces without basic attributes";
  - Count faces without basic attributes "administation" > "get face count without basic attributes";
  - Count basic attributes linked to faces "administation" > "get basic attributes count info".

- The "DB_CONNECT_TIMEOUT" setting was added to the Events service config.

- The dump_emails.py script was removed from the Admin service.

LUNA PLATFORM v.5.1.1

Changes

- The possibility to specify *attributes* as candidates in the "/matcher/faces" resource ("matcher" > "matching faces" in APIReferenceManual.html) was added. It enables you to perform matching without saving candidates to the database.

- The mechanism for migration of settings in Configurator service is available. Settings migration is performed without changing user values. Additional actions are required to apply this mechanism to LP build 5.1.0. These actions are described in the "LP_Upgrade_Manual.html" manual.

- The "track_id" field was added for events. It enables you to mark events belonging to the same person for future analysis. You should manually specify the "track_id" in the "handlers/handler_-id/events" request. The field can be used:

  - As a filter for matching candidates;
  - For cross-matching, clustering, linker, and reporter tasks. In the clustering task, you can now specify whether to add events with the same "track_id" to the same cluster.

- You can specify settings from Configurator DB when performing migration using Alembic.

```
alembic -x luna-config=http://127.0.0.1:5070/1 upgrade head
```

- Examples for the "ws" resource were added to "SenderReferenceManual.html".

- An exception was added for the case when the Events service is disabled, but the event saving is enabled in "storage_policy". The following error is returned: 11040 "Luna Events service is disabled"

- The incorrect link to the sample image in response when generating an event using "events" > "generate events".

Link in previous LP build: `"url": "http://image-store:5020/1/buckets/visionlabs-samples/images/2e8045f2-14cd-4c8b-af3a-7c959e85fe6f",`

Fixed link: `"url": "/6/samples/faces/939fca8a-753f-4ee4-8450-9c1bb3c1f76c",`