IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 2 November 2022, accepted 28 December 2022, date of publication 30 December 2022, date of current version 5 January 2023.

Digital Object Identifier 10.1109/ACCESS.2022.3233463

RESEARCH ARTICLE

Memory Efficient Local Features Descriptor for Identity Document Detection on Mobile and Embedded Devices

DANIIL P. MATALOV[1,2], ELENA E. LIMONOVA[1,2], (Member, IEEE), NATALYA S. SKORYUKINA[1,2], AND VLADIMIR V. ARLAZAROV[1,3], (Member, IEEE)

[1]Smart Engines Service LLC, 117312 Moscow, Russia
[2]Federal Research Center Computer Science and Control, RAS, 119333 Moscow, Russia
[3]Institute for Information Transmission Problems (Kharkevich Institute), RAS, 127051 Moscow, Russia

Corresponding author: Daniil P. Matalov (mataloff@gmail.com)

This work was supported by the Grant from the Ministry of Science and Higher Education of the Russian Federation (internal number 00600/2020/51896) under Grant 075-15-2022-319.

ABSTRACT In this paper, we propose a data-driven approach to training a memory-efficient local feature descriptor for identity documents location and classification on mobile and embedded devices. The proposed algorithm for retrieving a dataset of patches is based on the specifics of document detection in smartphone camera-captured images with a template matching approach. The retrieved dataset of patches relevant to the domain, which includes splits for features training, features selection, and testing, is made public. We train a binary descriptor using the retrieved dataset of patches, each bit of the descriptor relies on a single computationally-efficient feature. To estimate the influence of different feature spaces on the descriptor performance, we perform descriptor training experiments using gradient-based and intensity-based features. Extensive experiments in identity document location and classification benchmarks showed that the resulting 128 and 192-bit descriptors which use gradient-based features outperformed a state-of-the-art 512-bit BEBLID descriptor for arbitrary keypoints matching in all cases except the cases of extreme projective distortions, being significantly more efficient in cases of low lighting. The 64-bit gradient-based descriptor obtained within the approach showed better quality than 128 and 256-bit BinBoost descriptors in scanned document images. To evaluate the influence of the descriptor size on the matching speed, we propose a model based on the required number of processor instructions for computing the Hamming distance between a pair of descriptors on various energy-efficient processor architectures.

INDEX TERMS Binary descriptors, document classification, identity documents recognition, image retrieval, local feature descriptors, local features learning.

I. INTRODUCTION

ID (identity document) recognition systems are already deeply integrated into human activity, and the pace of integration is only increasing [1]. Almost everyone faces document recognition systems in banking, sharing economy, border crossing, hospitality, medicine, insurance, and other areas requiring ID document authentication. The very first fundamental image document recognition problems are document location and classification [2], [3]. A high-quality solution to these problems plays a significant role in the document recognition process. Specific knowledge about the document type allows defining optimal parameters for image processing to achieve high recognition accuracy and decrease the size of the input for character recognition algorithms, which affects the required amount of computing resources. For example, modern text-in-the-wild recognition algorithms, which are supposed to recognize all text characters regardless of font size and semantics, require significant computational resources [4], [5]. Several document location approaches are

The associate editor coordinating the review of this manuscript and approving it for publication was Jiju Poovvancheri.

based on extraction and analysis of the document boundaries, lines, segments, and corners [6], [7], [8]. These methods test a set of hypotheses about the consistency of the extracted geometric primitives with the distortion model of the document rectangle. Some modern deep learning-based approaches are used for document classification and retrieval [9], [10]. However, training a high accuracy algorithm requires tens of thousands of training samples, which is complicated to obtain for regulatory and privacy reasons. Such algorithms also contain millions of parameters that significantly affect computation time and memory consumption, including both random-access memory (RAM) and persistent storage.

Identity documents often have a fixed geometry and a set of predefined elements such as static text filling, logos, and fixed background. Chiron et al. [11] state that visual-based template matching approaches are the best suited for ID images recognition. The main advantages of the methods which follow local features matching are industrial precision and performance on a wide class of devices regardless of computational power [2], [12]. In this paper, we consider the problem of local features description for identity documents location and classification. The paper extends previous work [13] and provides an extensive justification of how to create computationally and memory-efficient local feature descriptors for the task, including training data preparation and the choice of feature space. Moreover, we experimentally show that depending on the type of source of identity document images to be recognized, for example, a scanner, it is achievable to train a significantly more compact descriptor that provides a comparative localization and classification quality. A compact descriptor allows for reducing requirements for computing resources and thereby expand the class of computing devices capable of solving the detection problem using the template matching approach.

The main contributions of this paper are:

- a novel data-driven approach for training local features descriptors for identity documents detection and classification on mobile and embedded devices with local features matching based methods;
- a new public available dataset of patches considering the identity documents matching domain;
- for the first time, provide a model for estimating the performance of calculating the distance between a pair of binary descriptors depending on the descriptors' size on various CPU architectures.

The rest of the paper is organized as follows. Section II is devoted to the description of the application of local features for the problem of identity documents localization and classification. In Section III we present an overview of related works dedicated to different descriptor algorithms used in identity document images matching and provide an overview of modern efficient descriptors for arbitrary image matching. Section IV presents the proposed descriptors and the learning algorithm while Section V presents a used dataset of domain-specific patches and an algorithm for its retrieval. In Section VI we present and discuss experimental results

FIGURE 1. Regions of static document type dependent elements (green) and personal data (red).

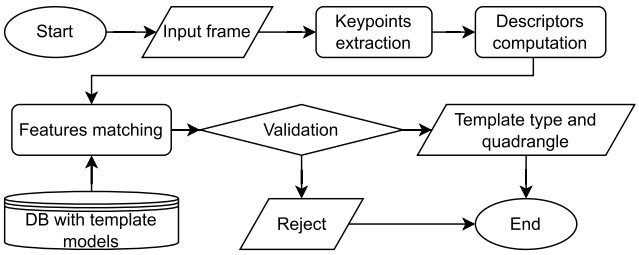


FIGURE 2. Identity document localization and classification using template matching approach.

and Section VII proposes a model to compare the matching speed of different descriptors. In conclusion, the results are summarized and future work plans are discussed.

II. TEMPLATE MATCHING FOR IDENTITY DOCUMENTS DETECTION

First of all, we consider the usage of the template matching approach to identity document classification and localization in the necessary degree of detail and refer to [12] and [14] for the extensive description.

An identity document type within the approach is represented as a template model. According to the fact, that the design of an ID is strictly regulated, every instance of the particular document type differs only in the regions containing personal data. Other regions, which don't contain personal data, usually consist of lots of static texts, logos, and other graphical elements, which are rich with local features (keypoints). As a result, a template model comprises a set of keypoints and their descriptors specific to a particular identity document type may be constructed. Fig. 1 shows the regions containing static texts (green rectangles) and personal data (red rectangles). To create an identity document template model, its ideal image and regions specifying document type distinctive keypoints are used. An ideal template image is an image usually obtained with a flatbed scanner and cropped according to document borders. The image size is normalized, in such a way, that 1 physical millimeter occupied 10 pixels. The distinctive regions are used to filter keypoints that are supposed to form a template model.

A diagram summarizing the main blocks of the document detection algorithm with precomputed template models is

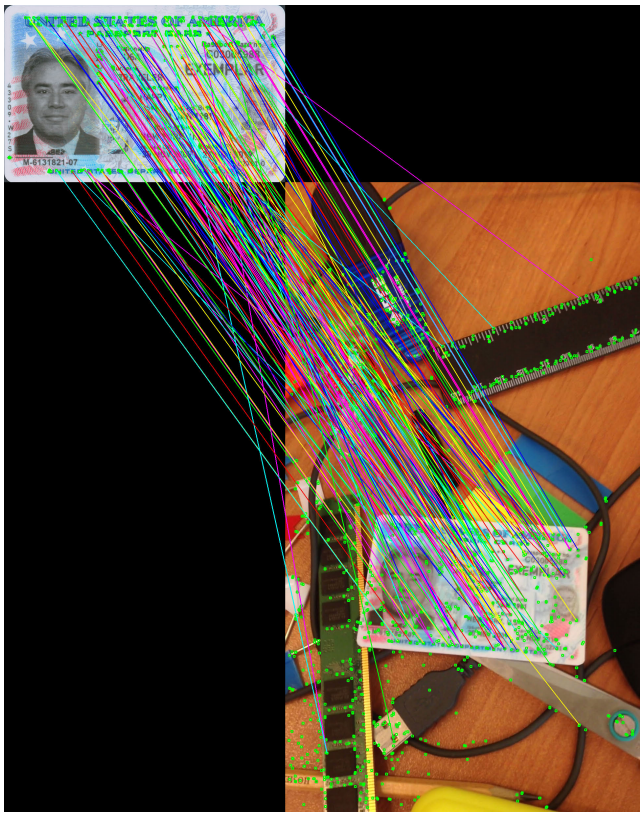


FIGURE 3. Template features matching.

shown in Fig. 2. At first, the same keypoint extraction and descriptors computing algorithms that are used to construct an identity document model are executed on a query image. Each keypoint of the input image is matched with the points that form the template models based on the distance between the descriptors. Then, all the template models are arranged in accordance with the number of points which were matched with the query image. The best models are geometrically verified using RANSAC to find a transformation that maps the query points to the corresponding points of the template according to a distortion model. The template and the transformation matrix considers the best if the number of points which are correctly mapped overcome a specified threshold and the number is the highest among the other models. The document type and the document quadrangle in the input image are trivially derived and form the answer of the detection algorithm. A more detailed description of the method is presented in [12]. The example of the matching process is shown in Fig. 3.

III. RELATED WORKS

Many research papers use various combinations of different algorithms of local features detection and description for solving identity document detection problem.

In paper [15] a set of SURF [16] descriptors is used to create a so-called vocabulary of representative visual words to train an SVM classifier for different document classes. SURF keypoints and descriptors were also used in paper [14] to build a complex algorithm for simultaneous identity document detection and localization with the local features matching. The algorithm capable of detecting 64 different document classes achieved an accuracy of 96.6% on a private dataset of real documents. Later, another implementation of local features matching approach proposed in paper [12] showed better performance than [14] on public MIDV-500 dataset [17] consisting of 15000 images of 50 different document types. The algorithm, that used a combination of SURF [16] keypoints and RFD [18] descriptors, was claimed to find document coordinates and its class on iPhone 6 smartphone for 0.35 seconds. Despite the lots of advantages of the local features matching approach, which includes industrial precision and performance on energy-efficient devices, the approach has several limitations.

First of all, with the development of recognition systems, the number of supported documents increases along with the number of indexed document template models, which affects memory consumption. Document models are usually stored in persistent storage and the amount of bytes required to describe a keypoint directly affects I/O devices response. Skoryukina et al. [19] show that usage of BinBoost [20] binary descriptors instead of real-valued SURF allowed to reduce memory consumption dramatically without noticeable changes in classification quality. Therefore, the compactness of the descriptor is one of the key requirements.

Secondly, the process of searching the template models which contain the descriptor similar to the query one is accompanied by an intensive calculation of the distance between a query descriptor and the descriptors in the models. To speed up the search, a randomized k-d tree is usually used. However, the amount of memory required to store the indexing structure depends on the descriptor size, and its construction and the search is done with intense calculation of distance between descriptors. Binary descriptors have another advantage over real-valued ones since distance computation between a pair of binary descriptors is done with the Hamming distance, which may be efficiently implemented with hardware `popcnt` instruction.

Such binary descriptors as ORB [21], FREAK [22], and BRISK [23] were proposed as an alternative to real-valued and computationally intense SIFT and SURF. Recently, statistical methods and machine learning have been widely used to build algorithms for computing local descriptors. Fan et al. [18] train and select a set of gradient-based features in RFD using classical statistical methods. Modern machine learning approaches, such as BinBoost [20] and BEBLID [24] use modification of AdaBoost algorithm to train and select the most accurate and discriminative features. In BinBoost [20] approach, each bit of the descriptor is produced by a linear combination of weak classifiers. In BEBLID [24] each bit of the descriptor is computed by a single weak classifier. Some of the deep learning approaches perform L_2 loss minimization [25], [26], while the most modern ones train Siamese networks [27], [28] using triplet loss optimization. Despite

a significant increase in quality, neural network models produce real-valued descriptors and require significant computing resources.

Recent works are aimed to the creation of a universal local feature descriptor suitable for any recognition task. However, the descriptor's ability to extract a vast set of image characteristics with different nature and map it to some metric space affect computational complexity and memory consumption, making it improper to execute on energy-efficient devices. Thus, a more practical approach is to develop a method for building a domain specific local features descriptor, considering the distortion model related to matching methods, which can be used on a wide class of computing devices. The scope of applicability of our descriptor is the identity document detection with template matching approach on devices with low computational resources. Unfortunately, there are no publicly available datasets of patches extracted from identity documents. This means, a domain-specific training dataset of aligned patches should be created in accordance with the matching approach and the distortion model specific to the capturing conditions.

IV. DESCRIPTOR DESIGN

To address the problem of limited computing resources the descriptor we build is binary rather than a real-valued one. To train the descriptor we followed the approach based on RFD [18]. It consists of training an extensive set of thresholding functions and further selecting the best-performing and discriminant ones. Each of the selected functions provides a bit in the descriptor and compares the value of a computationally efficient feature with a trained threshold. The selected set of functions defines the descriptor computing function. One of the most significant advantages of the descriptor training algorithm is the undemanding enormous amount of training data and clarity of implementation [18].

A. FEATURE SPACE

The traditional approach to constructing a descriptor includes defining a hand-crafted feature space which extracts low-level information about the neighborhood of a keypoint. Such low-level information feature extraction functions may be briefly divided into intensity-based and gradient-based. In this paper we consider intensity-based features used in [24] and gradient-based features used in [18].

1) AVERAGE BOX DIFFERENCE

Following the notation in [24], Average Box Difference feature is calculated as follows:

$$g(p_1, p_2, s) = \frac{1}{s^2}\left(\sum_{q \in R(p_1,s)} I(q) - \sum_{r \in R(p_2,s)} I(r)\right), \quad (1)$$

where $I(q)$ is pixel intensity, p_1 and p_2 define top left coordinates of the squared regions of size s. Fig. 4 shows a descriptor value computation based on the average box difference features.

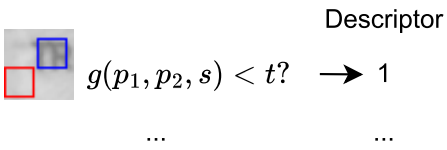


FIGURE 4. A descriptor value computing using Average Box Difference features.

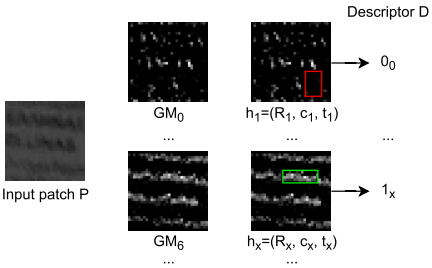


FIGURE 5. Descriptor computing with edge map features.

This feature type provides a measure of the difference between the average intensity of pixels in different regions of the neighborhood of a keypoint. Such feature space can computationally efficiently approximate a highly sampled image gradient orientation computed with various gradient calculation operators and may be implemented with integral images.

2) EDGE MAP FEATURES

Edge Map Features, that were originally proposed in [29], are widely used in different descriptors [18], [20]. Input image patch P is mapped into 8 images, each produced with a bilinear soft assignment of image gradient orientation. GM_0 and GM_6 in second column in Fig. 5 shows the magnitude of gradients which orientations mostly drop to sectors with $[0; \frac{\pi}{4}]$ and $[\frac{6\pi}{4}; \frac{7\pi}{4}]$ relatively.

A feature value computation is performed as follows:

$$g(R, c) = \frac{\sum_{(x,y)\in R} GM_c(x, y)}{\sum_{i=0}^{7} \sum_{(x,y)\in R} GM_i(x, y)}. \quad (2)$$

Such feature space is a sort of low-cost detector of edges of a particular length and orientation and may be efficiently implemented with integral images.

Moreover, such features were used in the 293-bit version of RFD descriptor [18], that demonstrated excellent performance in identity documents matching on mobile devices.

B. FEATURES TRAINING AND SELECTION

We followed RFD binary descriptor training approach, which was proposed by Fan et al. [18], with slight modifications. The objective of a local feature descriptor algorithm is to create as much as possible close descriptions for similar patches and to provide distant descriptions for different image patches. In the case of binary descriptors, Hamming distance is used to estimate the similarity of descriptors. RFD descriptor training is organized in 2 steps: 1) independent training of a set of thresholding functions leading to the

best discriminative ability and 2) selecting the most accurate features considering the potentially excessive correlation of their responses. The only parameter that affects the final size of the descriptor in the original algorithm is the correlation threshold since it inspects all the available features. Unlike the original [18] descriptor training algorithm, we use a different way of calculating the feature correlation and limit the maximum number of resultant features. Given M labeled pairs of patches (P_i^1, P_i^2, l_i), where $l_i \in \{1, -1\}$ indicates matching and non-matching pairs, a feature selection is performed according to Algorithm 1. The *corr* value models the frequency of coincidence of the responses of the classifiers on the corresponding patches, and its values are normalized to the range [0, 1]. We also ignore features for which the correlation equals 1 with one of the features already added to the descriptor since such features do not affect the discriminative power of the descriptor. Unlike many other gradient orientation based-features, we use the L_1 norm of gradient magnitude in our Edge Map features implementation to increase robustness, computation speed and reduce memory consumption. In our paper we used accuracy to train thresholds and to score features' discriminative ability. The accuracy of a feature with a given threshold is calculated as follows:

$$\mathrm{Acc}(g(t)) = \frac{\mathrm{TPR}(g(t)) + \mathrm{TNR}(g(t))}{2}, \tag{3}$$

where TPR and TNR are True Positive Rate and True Negative Rate, respectively. A pair of patches is considered as True Positive, if the thresholding results of the feature are equal for the both patches from the matching pair and different for non-matching pair.

In our paper we propose and study performance of 3 types of descriptors. The first, which uses Edge Map Features, we call RFDoc. The other one using ABD features we called ABDoc. And ABDRFDoc descriptor, which considers both feature spaces.

Algorithm 1: Feature Selection Algorithm

Input: Set of labeled pairs of patches M, set of features H, maximum descriptor size N, correlation threshold t_c
Output: Set of selected features S

```
Compute accuracy q_i for each h_i from H on the set of
  patches M and sort H in descending order of q_i;
S ← S ∪ {h_0}, j ← 1;
while |S| ≤ N and j < |H| do
    foreach j ∈ {1, ..., |H|} do
        g_ik ← true;
        foreach k ∈ {1, ..., |S|} do
            nm_jk ← 0, nd_jk ← 0;
            foreach m ∈ {1, ..., |M|} do
                if h_j(P_m^1) = h_k(P_m^1) then
                    nm_jk ← nm_jk + 1
                else
                    nd_jk ← nd_jk + 1
                end
                if h_j(P_m^2) = h_k(P_m^2) then
                    nm_jk ← nm_jk + 1
                else
                    nd_jk ← nd_jk + 1
                end
            end
            corr_jk ← (nm_jk − nd_jk)/(4 · |M|) + 0.5;
            if corr_jk ≥ t_c or corr_jk = 1 then
                g_ik ← false;
                break;
            end
        end
        if g_ik = true then
            S ← S ∪ {h_j};
        end
    end
end
```

V. A DATASET OF ALIGNED PATCHES FROM ID DOCUMENT IMAGES

To create a dataset of aligned patches, we used a subset of document types from MIDV-500 [17] and MIDV-2019 [30] datasets. MIDV-500 [17] contains video clips of 50 document types captured with a smartphone camera. The dataset addresses such problems of mobile camera-based document recognition as complex background, projective distortions, highlights, and other hand-held capturing caused distortions. MIDV-2019 [30] is an extension which includes the documents captured in 4K resolution under higher projective distortions and low-lighting. Each video clip in the datasets consists of 30 frames and the corresponding ground-truth, including coordinates of the document corners. We randomly selected 10 document types (chn.id, deu.passport.new, dza.passport, esp.id.new, fin.drvlic, grc.passport, hun.passport, irn.drvlic, prt.id, usa.passportcard) and randomly selected no more than 7 frames from each clip, rejecting those frames with document corners falling out of the frame for more than 10% of the average diagonal length of a document. The selected frames were used to extract matches. The matches extraction procedure is described as Algorithm 2.

Quadrangle N_b defines document coordinates in a fixed-size coordinate system where a document plane occupies the entire image. The size of the coordinate system, which is the same as the size for an ideal template image, is computed according to the physical sizes of the document type to ensure that 1 physical millimeter occupies 10 pixels. The set of rectangles R specify document regions supposed to contain personal data (red rectangles in Fig. 1). In our implementation we used YACIPE [31] algorithm for keypoint extraction since it has already shown great performance for document detection problem [12] and is suitable for embedded devices. Point filtering is done to avoid overfitting, since

Algorithm 2: Retrieving a Set of Matching Document Patches From a Single Frame

Input: F – an input frame, Q – a document quadrangle in the input frame coordinate system, N_b – a document quadrangle in the normalized frame coordinate system (specific to document type), R – a set of rectangles in N_b (specific to document type);

Output: A set of pairs of image patches Y

Compute transformation matrix $H : Q \to N_b$;

Compute scale $s \leftarrow \frac{A_Q}{A_{N_b}}$ as a ratio of areas N_b and Q;

Compute normalized document image D applying H to F;

Run keypoint extraction algorithm to D and get a set of keypoints $K \leftarrow \{(x, y, size)\} \in D$;

Remove points from the set K that lie outside all the rectangles;

foreach $k_i = (x, y, size)$ *in* K **do**

  $k_i' \leftarrow (x_i', y_i', size_i')$, where $(x_i', y_i') \leftarrow H^{-1}(x_i, y_i)$, $size' \leftarrow size \cdot s$;

  Extract aligned image patches (P_i, P_i') for keypoints k_i and k_i' from the original frame and the normalized one;

  $Y \leftarrow Y \cup (P_i, P_i', 1)$;

end

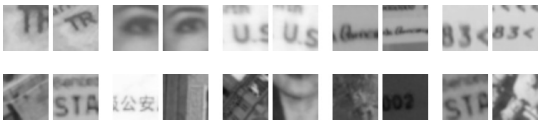


FIGURE 6. Examples of matching (top row) and non-matching (bottom row) pairs from the dataset.

the ID type model in the template matching approach is built from the descriptors of the points belonging to the static elements specific to the document type (green rectangles in Fig. 1).

To complete the dataset of patches with non-matching pairs, we used a well-known Brown dataset [32]. A single pair was obtained by random sampling a patch from Y and a patch from sets of 500k patches in the Brown dataset. All the patches from Y used in non-matching pair construction were removed from the matching part of the dataset. The retrieved dataset of patches were divided into 3 sets: 75k pairs for features training, 75k for features selection, and 350k for testing. All the sets were balanced (50% of matching and 50% of non-matching pairs) and did not intersect. Examples of pairs of patches from the collected dataset are presented in Fig. 6. The dataset is available for download at ftp://smartengines.com/rfdoc. The structure of the dataset and the way of indexing patches is similar to that used in the Brown dataset [32].

TABLE 1. FPR@95% of the trained descriptors on the test set of 350k patches.

Descriptor	Num bits	Error rate, %
ABDoc	64	0.731
	128	0.636
	192	0.652
RFDoc	64	0.305
	128	**0.177**
	192	0.208
ABDRFDoc	64	8.619
	128	7.246
	192	6.658

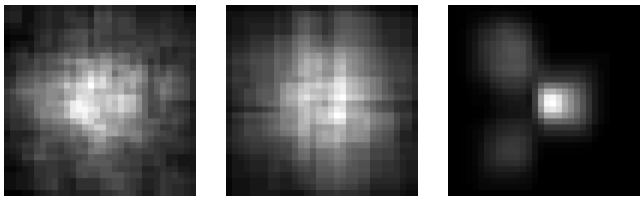
VI. EXPERIMENTAL EVALUATION

A. TRAINING AND FEATURES SELECTION

In our experiments, we built a descriptor that took a square image with a size of 32 pixels. Many rectangles may be generated in such a coordinate system, though a huge number of them is usually highly correlated [18]. In our experiments, we generated rectangles defining Edge Map Features with sides greater or equal to 3 pixels and an area less or equal to 256. For Average Box Difference features, we generated all possible non-overlapping squares of size from 3 to 17 pixels with a step of 1. Threshold fitting was performed on a set of patches from ID documents (see in Sec. V) that are indexed in `train_75k.txt` file at ftp://smartengines.com/rfdoc. To ensure compactness and high speed of comparison of a pair of descriptors (see in Sec. VI-D, VII), we used the maximum number of the selected features N equal to 64, 128 and 192 in Algorithm 1. In the features selection algorithm, we fixed the feature correlation threshold t_c to 0.7 since smaller values of t_c did not allow to select N features in some cases, but greater values led to less discriminant descriptors and showed worse performance. Features selection was performed using the set of patches that are indexed in `feature_selection_75k.txt` file at ftp://smartengines.com/rfdoc.

B. PATCH VERIFICATION PERFORMANCE

To measure the quality of the descriptors for patch verification problem, we used a set of 350k pairs of patches from documents (see Sec. V) that are indexed in `test_350k.txt` file. The descriptors that use Edge Map features we call RFDoc (following the notation in [13]), and descriptors that are based on Average Box Difference features we call ABDoc. Since the considered feature spaces describe different low-level image characteristics, we assumed that its simultaneous usage in the descriptor could increase the overall performance. Based on that we executed Algorithm 1 with H representing a merged set of trained functions both of the mentioned feature spaces and built the descriptors which we call ABDRFDoc. False Positive Rates at 95% Recall (FPR@95%) is the main characteristic to evaluate the patch verification performance. The idea behind that is to get an error rate at such a threshold on the distance between a pair



(a) ABDoc (b) RFDoc (c) ABDRFDoc

FIGURE 7. Distribution of rectangles that define features in the resultant 128-bit descriptors.

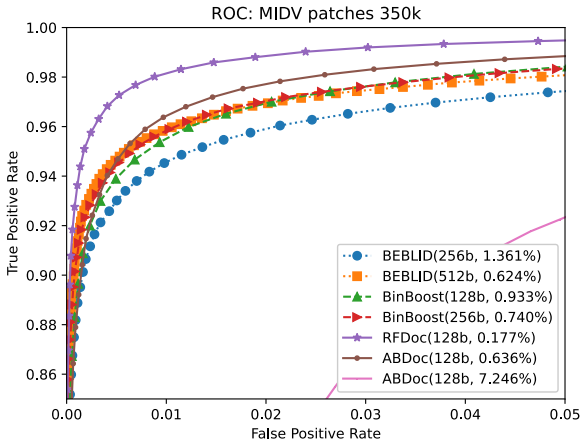


FIGURE 8. ROC curves of 128-bit RFDoc and ABDoc descriptors, relative to modern binary descriptors. The amount of bits required to store a keypoint representation and error rate to provide 95% True Positive Rate are stated in the legend.

of descriptors that provides a successful matching of 95% of positive patches. Patch verification performance of the trained descriptors on the test set of patches is presented in Table 1.

According to the experimental data, using only Edge Map features in the selection algorithm lead to the best-performing descriptor. Surprisingly, the descriptors that consider both feature spaces showed the worst performance. The reason for this could be that after selecting the first few high-quality and uncorrelated features, the algorithm scans and adds features with significantly much less accuracy. Moreover, although classified feature responses from different feature spaces may be uncorrelated based on our definition, they may still be internally dependent and overly complementary. The visualization of how rectangles defining the features are distributed over the patch coordinate system in the resultant 192-bit descriptors is presented in Fig. 7. Based on the visualization, the features in the descriptors that use only one specific feature type (Fig. 7(a) for ABDoc and Fig. 7(b) for RFDoc) are distributed over almost all the area of the patch. On the other hand, features in ABDRFDoc descriptors are highly concentrated in a particular area, which explains the poor performance of such descriptors.

To compare the trained descriptors to state-of-the-art binary descriptors we used the implementations provided in the OpenCV of version 4.5.1. Unfortunately, to our knowledge, there was no publicly available authorized RFD [18] implementation, and we were unable to test original RFD performance in our test configurations. To retain the descriptiveness of the ROC curves presented in Fig. 8, we show results only for 128-bit versions of the trained descriptors. According to the ROC curve, 128-bit version of RFDoc descriptors significantly outperformed all of the considered trainable binary descriptors, including the most efficient binary descriptor for arbitrary image matching BEBLID-512 [24].

C. COMPLEX CAMERA-BASED DOCUMENT RECOGNITION

To explore the performance of the trained descriptors for complex identity document location and classification, we evaluated the approach proposed in [14] using images of document types, which was not used for aligned patches dataset creation, from MIDV datasets (see Sec. V). Unlike [14] we used a bruteforce matching instead of FLANN to achieve reproducible results and to exclude potential collisions caused by approximate nearest neighbors search, and we did not use additional restrictions for homography matrix. All the parameters related to ideal template images, keypoint extraction, and RANSAC were fixed according to [12]. The algorithm of computing the detection performance is based on the calculation of the maximal deviation of the computed document corner coordinates divided by the length of the shortest document boundary side [12]. As an algorithm for keypoints detection, SURF [16] algorithm is used for all of the considered descriptors.

Matching performance on the hand-held captured identity documents of the MIDV datasets is presented in Table 2. BEBLID-512 descriptor contains more than 2.5 times more bits and belongs to the significantly memory-consuming class, however, we present the results for it to understand our position relative to state-of-the-art. MIDV-2020 dataset [3] (see Fig. 9 (a,b)) consists of 1000 different documents (100 document samples per type), all with unique artificially generated document content such as faces, signatures, text fields data. In addition to smartphone captured video clips, MIDV-2020 also contains high-quality ideal scans Fig. 9 (c) and photos of arbitrarily placed documents with different backgrounds. Although MIDV-2020 did not participated in training data extraction, images and clips containing passport of Greece was not used in testing, since images of that document type presented in MIDV-500 and MIDV-2019 was used for patch verification setup. Unlike other MIDV datasets, MIDV-LAIT [33] consider identity documents with Perso-Arabic, Thai, and Indian Scripts, and consists of 180 unique synthetically generated documents. Experimental results on MIDV-LAIT dataset were performed excluding ''ind.aadhaar'' document type since this document type does not have enough distinguishing keypoints to use the template-matching approach without significant modifications.

According to experimental results on MIDV-500, the RFDoc-192 descriptor showed similar document classification and localization performance to the best performed BEBLID-512. However, the RFDoc-192 descriptor

TABLE 2. Experimental results on the hand-held captured parts of the MIDV datasets.

Descriptor	MIDV-500		MIDV-2019		MIDV-LAIT		MIDV-2020				Bits
	Clips								Photo		
	Class.	Locat.	Class.	Locat.	Class.	Locat.	Class.	Locat.	Class.	Locat.	
64-128 bit descriptors											
RFDoc-64 [our]	87.708	78.838	73.333	57.597	96.277	76.061	60.408	45.744	89.444	76.064	**64**
RFDoc-128 [our]	**92.866**	**84.194**	**86.062**	**71.697**	**98.888**	**79.804**	**73.094**	**58.463**	**97.000**	**88.385**	128
BinBoost-128 [20]	85.958	73.588	68.791	50.262	92.222	70.854	42.551	31.304	70.666	56.837	128
192-256 bit descriptors											
RFDoc-192 [our]	**93.458**	85.128	**88.875**	**75.535**	**98.777**	**80.741**	77.821	63.176	**98.666**	89.599	**192**
ABDoc-192 [our]	93.358	**85.139**	84.333	70.768	97.555	78.747	73.886	58.056	96.111	84.015	**192**
BEBLID-256 [24]	92.783	84.072	83.833	72.368	97.611	80.020	**81.013**	**67.930**	98.333	**90.312**	256
BinBoost-256 [20]	91.116	81.132	79.916	63.074	96.333	76.132	58.221	45.366	81.111	70.620	256
512 bit descriptors											
BEBLID-512 [24]	93.508	85.226	85.854	75.001	98.611	81.959	83.718	70.378	99.000	92.580	512

(a) Extreme projective (b) Low light (c) Scan, arbitrary placed (d) Saudi Arabia ID

FIGURE 9. Images from MIDV-2020 and MIDV-LAIT.

requires 2.65 times less memory to describe one local feature of the image. 128-bit version of RFDoc, being 2 times less memory consuming, showed the same quality as BEBLID-256. 64-bit RFDoc significantly outperformed BinBoost-128 both in terms of classification and localization, but remarkably lost to other RFDoc and BEBLID. On a more challenging MIDV-2019 dataset, which includes documents captured under low-lighting in 4K, RFDoc-192 descriptor outperformed all the competitors and showed 21% fewer classification errors than BEBLID-512. The reason for this may be the gradient-based nature of RFDoc features. BEBLID-512, RFDoc-192 and RFDoc-128 showed very close classification performance to each other on MIDV-LAIT, however the 512-bits length version of BEBLID outperformed RFDoc in the accuracy of finding document corners coordinates. RFDoc showed worse performance than BEBLID descriptors in the clip setup of MIDV-2020. Errors analysis showed that the most problematic cases for RFDoc are extreme projective distortions (see Fig. 9(a)). The reason for this is that the feature space used in RFDoc descriptors does not incorporate spatial information and use only gradient directions. The same trend was seen on the "photo" part, however the document classification performance was indistinguishable, and there was no such gap in localization performance. When comparing the quality of RFDoc and ABDoc descriptors, the latter shows significantly worse performance, regardless of the descriptor size.

Table 3 shows descriptors performance in images of MIDV-2020 obtained with a flatbed scanner. All the BEBLID and RFDoc versions showed the best performance in cases, when a document was placed in the flatbed scanner without any rotations. However, the influence of the spatial information encoding feature space on the accuracy of finding document quadrangles may be observed. In cases of arbitrary placed and rotated documents (see (Fig. 9(c))), 128 and

TABLE 3. Experimental results on the scan parts of MIDV-2020 dataset.

Descriptor	Upright scans		Rotated scans		Bits
	Class.	Locat.	Class.	Locat.	
64-128 bit descriptors					
RFDoc-64 [our]	99.888	94.389	97.222	87.667	**64**
RFDoc-128 [our]	**100.000**	**96.674**	**99.555**	**94.535**	128
BinBoost-128	80.333	75.203	77.222	69.446	128
192-256 bit descriptors					
RFDoc-192 [our]	**100.000**	96.499	**100.000**	96.191	**192**
ABDoc-192 [our]	99.555	92.478	95.555	87.734	**192**
BEBLID-256	**100.000**	**98.782**	99.888	**96.934**	256
BinBoost-256	87.444	81.046	84.333	78.884	256
512 bit descriptors					
BEBLID-512	100.000	98.673	100.000	97.838	512

TABLE 4. Required amount of memory to index descriptors of SURF keypoints specific to 50 document templates presented in MIDV-500 dataset.

Descriptor	Bits	Raw, MB	Zip, MB
RFDoc/ABDoc	64	2.9	0.7
	128	5.8	1.4
	192	8.6	2.2
BinBoost [20]	128	5.8	1.4
	256	11.5	2.9
BEBLID [24]	256	11.5	2.9
	512	22.7	5.9

192 bit RFDoc showed almost the highest possible classification accuracy.

D. MEMORY CONSUMPTION

A required amount of memory to index different types of considered descriptors with OpenCV brute force matcher for descriptors of SURF keypoints specific to 50 document types presented in MIDV-500 dataset is shown in Table 4. If we scale the number of document types supposed to be detected to a number of supporting documents by industrial recognition systems, for example, several thousand document types, a difference in memory consumption becomes critical. For example, consider a system which is supposed to support recognition of 3000 different document types. To estimate required amount of memory to index the document templates, the amount of memory presented in Table 4 may be multiplied by 60. In this case, to only index the document types represented as a set of keypoint descriptors of the BEBLID-512 type, approximately 1.3 GB of memory is required, which may become unfeasible for edge computing devices. At the same time, if each document model is represented with RFDoc-128 descriptors, which demonstrated relatively similar performance to BEBLID-512 in most cases, the indexing algorithm will require around 0.4 GB RAM.

VII. ESTIMATION OF THE MATCHING SPEED

It is rather difficult to reliably assess the impact of the type of descriptors on the matching speed. Things like the complexity of the descriptor computing algorithm, descriptor comparison metric, and the descriptor's data type, the number of descriptors, and the indexing algorithm directly affect the matching speed. Therefore, we do not compare the overall matching time of different descriptors implemented with different degrees of code optimization. A patch description speed depending on the used descriptor type may be estimated using the data presented in papers [24] and [18]. Instead, we evaluate the influence of different types of binary descriptors on matching speed in terms of the number of instructions required to calculate the distance between a pair of descriptors.

TABLE 5. Hardware support of population count instruction.

Arch	Instruction	Input width, bit	Processed width, bits
x86-64	POPCNT	32/64	32/64
x86-64 (AVX-512)	VPOPCNTW	128/256/512	16/32/64
ARM	VCNT	64/128	8
MIPS	PCNT	128	8/16/32/64

TABLE 6. A number instructions to compute Hamming distance for various binary descriptors on different architectures in xor/popcnt/add format (less is better).

Descriptor	Bits	x86-64		ARM		MIPS	
		x/y/z	total	x/y/z	total	x/y/z	total
BEBLID	256	4/4/3	**11**	2/2/5	**9**	2/2/2	**6**
	512	8/8/7	**23**	4/4/7	**15**	4/4/4	**12**
BinBoost	128	2/2/1	**5**	1/1/4	**6**	1/1/1	**3**
	256	4/4/3	**11**	2/2/5	**9**	2/2/2	**6**
RFDoc	64	1/1/0	**2**	1/1/3	**5**	1/1/0	**2**
	128	2/2/1	**5**	1/1/4	**6**	1/1/1	**3**
	192	3/3/2	**8**	2/2/5	**9**	2/2/2	**6**

In the case of binary descriptors, the Hamming distance is used as a similarity metric. The Hamming distance computation can be implemented with bitwise XOR and population count instructions, usually included in the instruction set of modern processors. We consider CPUs of x86-64, ARM, or MIPS architectures typical for end-user devices. These architectures provide hardware instruction for the population count (`popcnt`) with a varying input width and processed width (see Table 5) [34], [35], [36]. Input width is the bitwidth of the data chunk processed by one instruction, while processed width is the size of the element inside this data chunk for which the `popcnt` is computed. Thus, to calculate Hamming distance between two binary vectors, we need to (1) pack them into integer values, (2) compute bitwise XOR between them, (3) split its result into chunks equal to input width and perform `popcnt` for each of them, (4) sum up the results, (5) perform horizontal addition of the elements inside the sum data chunk (to take into account processed width different from input width).

To obtain more accurate algorithm's computational efficiency, we should consider the number of arithmetic logic units (ALUs) able to perform each operation and its latency. Such an estimate should be determined for each specific processor, which is outside the scope of this work. Instead,

we estimate the total number of instructions corresponding to the case with one ALU and the same execution time for all instructions. The number of instructions that process a fixed-size bit subset to calculate the Hamming distance for different architectures and binary descriptors is shown in Table 6. According to the overall number of instructions, Hamming distance computation between a pair of any RFDoc-128 descriptors is from 2.5 to 4.6 times faster, then the closest in terms of quality BEBLID-512 descriptor. RFDoc-64 requires from 1.2 to 2.5 times fewer instructions than RFDoc-128 and demonstrates the best computational efficiency among all considered descriptors.

VIII. CONCLUSION

In this paper, we consider the problem of training a local feature descriptor for identity documents detection and classification on devices with limited computational resources. To train a memory-efficient descriptor, we developed a problem-specific framework, which considers the key features of the identity documents detection with a template matching approach. First, we create a training dataset of patches, which incorporates the distortions specific to hand-held capturing and agrees with a document template construction model. The domain-specific dataset of patches extracted from camera captured documents was published. Experiments show that the proposed approach using features that represents a low-cost line segment detector makes it possible to build high-quality descriptors for the considered problem. The resulting 64-bit RFDoc descriptor being 2 times more memory efficient significantly outperformed 128-bit BinBoost in complex identity document location and classification benchmarks performed on all known for today MIDV datasets. RFDoc-128 performed better than all the considered BinBoost descriptors and showed very close results to BEBLID-256, except the cases of extreme projective distortions. In almost all cases, with the exception of images taken in low-light conditions, 512 bit BEBLID showed the best results. However, 192 bit version of RFDoc, which is more than 2.5 times memory efficient, achieved slightly worse performance in most cases than BEBLID-512 and completely outperformed all the considered descriptors in smartphone camera captured identity documents under low-lightning case.

As future work we plan to modify the algorithm for selection functions that provide a bit in the descriptor to overcome limitations related to poor performance in combining feature spaces and expand the family of decision rules. Slightly more complex rules could increase the distinctiveness of a bit in the descriptor and may allow to construct even more memory-efficient descriptors.

REFERENCES

[1] A. Goode, "Digital identity: Solving the problem of trust," *Biometric Technol. Today*, vol. 2019, no. 10, pp. 5–8, Dec. 2019.

[2] K. Bulatov, V. V. Arlazarov, T. Chernov, O. Slavin, and D. Nikolaev, "Smart IDReader: Document recognition in video stream," in *Proc. 14th IAPR Int. Conf. Document Anal. Recognit. (ICDAR)*, Nov. 2017, pp. 39–44.

[3] K. B. Bulatov, E. V. Emelianova, D. V. Tropin, N. S. Skoryukina, Y. S. Chernyshova, A. V. Sheshkus, S. A. Usilin, Z. Ming, J.-C. Burie, M. M. Luqman, and V. V. Arlazarov, "MIDV-2020: A comprehensive benchmark dataset for identity document analysis," *Comput. Opt.*, vol. 46, no. 2, pp. 252–270, Apr. 2022.

[4] S. Long, X. He, and C. Yao, "Scene text detection and recognition: The deep learning era," *Int. J. Comput. Vis.*, vol. 129, no. 1, pp. 161–184, 2020.

[5] D. Yu, X. Li, C. Zhang, T. Liu, J. Han, J. Liu, and E. Ding, "Towards accurate scene text recognition with semantic reasoning networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 12113–12122.

[6] D. Tropin, I. Konovalenko, N. Skoryukina, D. Nikolaev, and V. V. Arlazarov, "Improved algorithm of ID card detection by a priori knowledge of the document aspect ratio," in *Proc. 13th Int. Conf. Mach. Vis.*, Jan. 2021, pp. 407–415.

[7] A. Zhu, C. Zhang, Z. Li, and S. Xiong, "Coarse-to-fine document localization in natural scene image with regional attention and recursive corner refinement," *Int. J. Document Anal. Recognit. (IJDAR)*, vol. 22, no. 3, pp. 351–360, Sep. 2019.

[8] E. Puybareau and T. Geraud, "Real-time document detection in smartphone videos," in *Proc. 25th IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2018, pp. 1498–1502.

[9] A. W. Harley, A. Ufkes, and K. G. Derpanis, "Evaluation of deep convolutional nets for document image classification and retrieval," in *Proc. 13th Int. Conf. Document Anal. Recognit. (ICDAR)*, Aug. 2015, pp. 991–995.

[10] A. Das, S. Roy, U. Bhattacharya, and S. K. Parui, "Document image classification with intra-domain transfer learning and stacked generalization of deep convolutional neural networks," in *Proc. 24th Int. Conf. Pattern Recognit. (ICPR)*, Aug. 2018, pp. 3180–3185.

[11] G. Chiron, N. Ghanmi, and A. M. Awal, "ID documents matching and localization with multi-hypothesis constraints," in *Proc. 25th Int. Conf. Pattern Recognit. (ICPR)*, Jan. 2021, pp. 3644–3651.

[12] N. Skoryukina, V. Arlazarov, and D. Nikolaev, "Fast method of ID documents location and type identification for mobile and server application," in *Proc. Int. Conf. Document Anal. Recognit. (ICDAR)*, Sep. 2019, pp. 850–857.

[13] D. P. Matalov, E. E. Limonova, N. S. Skoryukina, and V. V. Arlazarov, "RFDoc: Memory efficient local descriptors for id documents localization and classification," in *Document Analysis and Recognition—ICDAR* (Lecture Notes in Computer Science), vol. 12822, S. U. J. Lladós and D. Lopresti, Ed. Cham, Switzerland: Springer, 2021, pp. 209–224.

[14] A. M. Awal, N. Ghanmi, R. Sicre, and T. Furon, "Complex document classification and localization application on identity document images," in *Proc. 14th IAPR Int. Conf. Document Anal. Recognit. (ICDAR)*, Nov. 2017, pp. 426–431.

[15] L.-P. De Las Heras, O. R. Terrades, J. Llados, D. Fernandez-Mota, and C. Canero, "Use case visual bag-of-words techniques for camera based identity document classification," in *Proc. 13th Int. Conf. Document Anal. Recognit. (ICDAR)*, Aug. 2015, pp. 721–725.

[16] H. Bay, T. Tuytelaars, and L. V. Gool, "SURF: Speeded up robust features," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2006 pp. 404–417.

[17] V. V. Arlazarov, K. Bulatov, T. Chernov, and V. L. Arlazarov, "MIDV-500: A dataset for identity document analysis and recognition on mobile devices in video stream," *Comput. Opt.*, vol. 43, no. 5, pp. 818–824, Oct. 2019.

[18] B. Fan, Q. Kong, T. Trzcinski, Z. Wang, C. Pan, and P. Fua, "Receptive fields selection for binary feature description," *IEEE Trans. Image Process.*, vol. 23, no. 6, pp. 2583–2595, Jun. 2014.

[19] N. Skoryukina, V. V. Arlazarov, and A. Milovzorov, "Memory consumption reduction for identity document classification with local and global features combination," in *Proc. SPIE*, vol. 11605, Jan. 2021, Art. no. 116051.

[20] T. Trzcinski, M. Christoudias, and V. Lepetit, "Learning image descriptors with boosting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 3, pp. 597–610, Mar. 2015.

[21] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 2564–2571.

[22] A. Alahi, R. Ortiz, and P. Vandergheynst, "FREAK: Fast retina keypoint," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 510–517.

[23] S. Leutenegger, M. Chli, and R. Y. Siegwart, "BRISK: Binary robust invariant scalable keypoints," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 2548–2555.
[24] I. Suárez, G. Sfeir, J. M. Buenaposada, and L. Baumela, "BEBLID: Boosted efficient binary local image descriptor," *Pattern Recognit. Lett.*, vol. 133, pp. 366–372, May 2020.
[25] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, P. Fua, and F. Moreno-Noguer, "Discriminative learning of deep convolutional feature point descriptors," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 118–126.
[26] Y. Tian, B. Fan, and F. Wu, "L2-Net: Deep learning of discriminative patch descriptor in Euclidean space," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 661–669.
[27] B. G. V. Kumar, G. Carneiro, and I. Reid, "Learning local image descriptors with deep Siamese and triplet convolutional networks by minimizing global loss functions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 5385–5394.
[28] X. Han, T. Leung, Y. Jia, R. Sukthankar, and A. C. Berg, "MatchNet: Unifying feature and metric learning for patch-based matching," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3279–3286.
[29] K. Ali, F. Fleuret, D. Hasler, and P. Fua, "A real-time deformable detector," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 2, pp. 225–239, Feb. 2012.
[30] K. Bulatov, D. Matalov, and V. V. Arlazarov, "MIDV-2019: Challenges of the modern mobile-based document OCR," in *Proc. SPIE*, vol. 11433, pp. 717–722, Jan. 2020.
[31] A. Lukoyanov, D. Nikolaev, and I. Konovalenko, "Modification of YAPE keypoint detection algorithm for wide local contrast range images," in *Proc. SPIE*, vol. 10696, pp. 305–312, Apr. 2018.
[32] S. A. J. Winder and M. Brown, "Learning local image descriptors," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2007, pp. 1–8.
[33] Y. S. Chernyshova, E. V. Emelianova, A. V. Sheshkus, and V. V. Arlazarov, "MIDV-LAIT: A challenging dataset for recognition of IDs with perso-arabic, thai, and Indian scripts," in *Document Analysis and Recognition—ICDAR* (Lecture Notes in Computer Science), vol. 12822, S. U. J. Lladós and D. Lopresti, Ed. Cham, Switzerland: Springer, 2021, pp. 258–272.
[34] *ARM NEON Documentation*. Accessed: Jun. 4, 2022. [Online]. Available: https://developer.arm.com/documentation/ihi0073/latest/
[35] *MIPS SIMD Documentation*. Accessed: Jun. 4, 2022. [Online]. Available: https://www.mips.com/products/architectures/ase/simd
[36] *Intel Intrinsics Guide*. Accessed: Jun. 4, 2022. [Online]. Available: https://www.intel.com/content/www/us/en/docs/intrinsics-guide/index.html

DANIIL P. MATALOV was born in Moscow, Russia, in 1996. He received the bachelor's degree in applied mathematics from the National University of Science and Technology MISiS, Moscow, in 2017, and the master's degree in physics, mathematics and computer science from the Moscow Institute of Physics and Technology, in 2019. He is currently pursuing the Ph.D. degree with the Federal Research Center "Computer Science and Control," Russian Academy of Sciences. Since 2016, he has been employed with Smart Engines Service LLC, Moscow. Since 2020, he has been employed with the Federal Research Center "Computer Science and Control," Russian Academy of Sciences. His research interests include neural network compression and image recognition on mobile devices.

ELENA E. LIMONOVA (Member, IEEE) was born in Dolgoprudny, Moscow, Russia, in 1993. She received the master's degree in physics, mathematics and computer science from the Moscow Institute of Physics and Technology, in 2017. She is currently pursuing the Ph.D. degree with the Federal Research Center "Computer Science and Control," Russian Academy of Sciences. Since 2016, she has been working as a Programmer and a Technician at Smart Engines Service LLC. Her research interests include neural network compression and image recognition on mobile devices.

NATALYA S. SKORYUKINA was born in 1991. She received the Specialist degree in applied mathematics from the National University of Science and Technology MISiS, Moscow, Russia, in 2013. Since 2014, she has been employed with the Federal Research Center "Computer Science and Control," Russian Academy of Sciences. Since 2016, she has been employed with Smart Engines Service LLC, Moscow.

VLADIMIR V. ARLAZAROV (Member, IEEE) was born in Moscow, Russia, in 1976. He received the Specialist degree in applied mathematics from the Moscow Institute of Steel and Alloys, in 1999, and the Ph.D. degree in computer science, in 2005. Since 1999, he has been working at the Institute for Systems Analysis, Russian Academy of Sciences (currently Federal Research Center "Computer Science and Control," Russian Academy of Sciences), Moscow, as a Researcher, a Senior Researcher, and the Head of the Laboratory. Since 2012, he has been working at the Moscow Institute of Physics and Technology (National Research University), Moscow, as an Associate Professor. Since 2016, he has been the General Director of Smart Engines Service LLC, Moscow. Since 2018, he has been working at the Institute for Information Transmission Problems, Russian Academy of Sciences, as a Senior Researcher. He has published over 90 articles and authored seven patents. His research interests include computer vision and document analysis systems.