Multistage Fusion of Face Matchers

Sergey Tulyakov Nishant Sankaran Deen Mohan Srirangaraj Setlur
Venu Govindaraju
Center for Unified Biometrics and Sensors, University at Buffalo, Buffalo, New York, USA
tulyakov,n6,dmohan,setlur,govind@buffalo.edu

Abstract

Multistage, or serial, fusion refers to the algorithms sequentially fusing an increased number of matching results at each step and making decisions about accepting or rejecting the match hypothesis, or going to the next step. Such fusion methods are beneficial in the situations where running additional matching algorithms needed for later stages is time consuming or expensive. The construction of multistage fusion methods is challenging, since it requires both learning fusion functions and finding optimal decision thresholds for each stage. In this paper, we propose the use of single neural network for learning the multistage fusion. In addition we discuss the choices for the performance measurements of the trained algorithms and for the selection of network training optimization criteria. We perform the experiments using three face matching algorithms and IJB-A and IJB-C databases.

1. Introduction

The benefits of fusing the recognition results of multiple pattern recognition in biometrics area are well understood and expected, and a great variety of the fusion algorithms have been presented in the literature [18, 5]. Two different fusion architectures could be distinguished: parallel and serial. Parallel architecture assumes that recognizers execute independently and all recognition results are available before fusion; fusion algorithm takes all the information and derives fused comparison scores in a single procedure. Serial architecture runs recognizers sequentially; after next recognizer's results are available, they are fused with the results of previously run recognizers, and the decision is made on whether to accept recognition results and finish, or reject recognition results and proceed to the next recognizer. Most of the developed fusion algorithms are parallel and research of serial fusion methods is rather sparse. For example, the overview article on fusion in biometric applications [5] has only three references on serial fusion works, and [13] describes serial fusion as novel and not fully explored.

The main reason for the use of the serial fusion is to save the time or other costs of running multiple recognition algorithms. Faster, and possibly less reliable, recognizers could be deployed first; if their results have sufficient confidence, there is no need to run slow, but possibly more precise, remaining recognizers. Generally, it is expected that the parallel fusion algorithm taking matching results from all recognizers should have higher recognition accuracy than the serial fusion algorithm, since earlier stages of serial fusion do not utilize all the information and their performance should suffer. We can also view serial fusion as a particular type of parallel fusion implemented in a decision tree like manner, where upper nodes of decision tree correspond to earlier stages of serial fusion; such restriction on fusion algorithm should have detrimental effect on performance. Though it might be possible that implemented serial fusion algorithm have superior accuracy, it is likely that the corresponding parallel fusion algorithm was not implemented properly, or only a limited number of suboptimal parallel fusion algorithms was used for comparison [21].

The example of 3-stage fusion system is presented in Fig. 1. At each stage k, the results of corresponding biometric matcher are obtained and a fusion function F_k calculates fused score using the results of matchers $1, \dots, k$. The fused score is then compared to thresholds θ_k^m and θ_k^n determining whether match or non-match decision could be made. If no decision is made, the algorithm proceeds to the next stage. In general, to train such serial fusion system one needs to learn all fusion functions F_k and thresholds θ_k^m and θ_k^n. The difficulty in training such systems is caused by interdependence between these learnable functions and parameters. For example, if we change some threshold, then the distribution of samples at later stages will change. If we change the fusion function, then the corresponding thresholds will also have to be changed. If current stage is improved, then the previous stage needs to change, so that more samples proceed to current stage.

Given the complexity of training procedure might explain the low number of works investigating this type of fusion. The question on how the performance of serial fusion

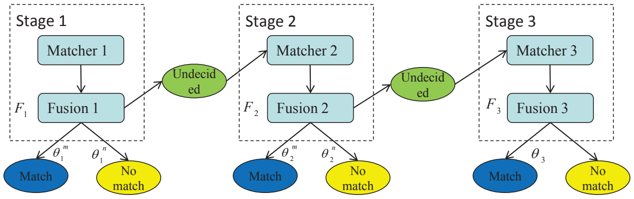

Figure 1. Sample serial fusion architecture consisting of three stages.

algorithms should be measured stands as another barrier for their development. Whereas parallel fusion has clear performance objective, such as accuracy in recognition results, the time costs have to be incorporated into performance measures for serial fusion. It appears, that currently there is no standard way to report the performance of serial fusion algorithms with accuracy and time cost trade-offs.

In this paper, we consider a construction of serial (or multistage) fusion algorithm using a single neural network. The advantage of such approach is the simplicity and uniformity in learning different stages of fusion. We also discuss how the performance of multistage fusion algorithms could be measured by optimizing a cost function including both accuracy and time.

2. Previous Work

The motivation for the earlier works on multistage fusion was the weakness of existing computing hardware and the desire to reduce the matching time while maintaining accuracy advantages by utilizing multiple recognition algorithms. For example, El-Shishiny et al. [4] stress the improvements of matching time in a three stage pattern classification system. Some later works emphasized the need to reduce the time in some particular big scale or time sensitive applications. Hong and Jain [7] introduced the multistage system to address the large time required to perform the biometric recognition in large databases. In this work, the faster face recognizer runs first to produce the small list of possible match candidates, and the slower fingerprint matcher runs next on reduced list to get final results. Note, that it is possible that face matcher could fail to list the genuine person among candidates, and the accuracy performance of such fused system could be lower than the performance of slower parallel fusion system, or the fingerprint matcher alone. In another example, Cordella et al. [2] investigates the use of multistage fusion in a time sensitive task of computer intrusion detection.

On the other hand, some works on multistage fusion stress accuracy performance benefits with smaller emphasis on time improvements. For example, Pudil et al. [14] present a serial fusion algorithm based on particular values of error rates FAR and FRR achieved at different stages, and show that fusion has better accuracy performance than any single combined recognizer. Similarly, Marcialis et al. [11] calculate error and time costs for proposed serial fusion algorithm, and show that overall costs, including time, are better than costs for individual recognizers, as well as, parallel fusion method.

Another idea for utilizing serial fusion is the attempt to build a better performing classifier ensembles. For example, Last et al. [10] use serial fusion to combine the recognition results of the classifiers utilizing increasing subsets of features. The classifiers with smaller feature sets work faster, and are, possibly, more stable at the tails of class distributions; incorporating them first into serial fusion system might achieve time and accuracy gains. This idea is closely related to general framework of boosting [6], but the performance gains of boosting methods are explained differently and have little relation to the task of time reduction investigated in serial fusion methods.

Note, that most works on serial fusion propose only specific algorithms based on particular performance points of the individual recognizers, such as the thresholds delivering some predetermined values of FAR and FRR. Essentially, such fusion methods are constructed in a heuristic manner with defined structure of the algorithm and some parameters obtained from training data. The trade-off between recognition accuracy and the time costs is not considered, and the optimization of the total system cost is not achieved.

More theoretically sound framework for optimizing multistage fusion learning based on relative costs of recognition error rates and times was presented in Trapeznikov et al. [20]. In contrast to our cost function defined later, this work used cost parameters defined for each particular stage. The optimization procedure was decomposed as the combination of individual optimizations for each stage. Since such optimizations depend on other stages, these optimizations for all the stages were done sequentially and repetitively until the convergence of the whole system.

In our proposed algorithm we utilize similar framework of total system cost optimization. Instead of training fusion functions separately at each stage, we propose building a single neural network modeling fusion functions at all stages. We also propose to separate the threshold optimization delivering best accuracy/time trade-off from the main fusion learning algorithm, so that the performance of serial fusion methods would be evaluated similar to other fusion methods.

3. Performance Measures

In this work we assume that we are dealing with biometric verification system, and the system's task is to confirm or deny the claim of biometric identity. The implementation of biometric system typically includes the cal-

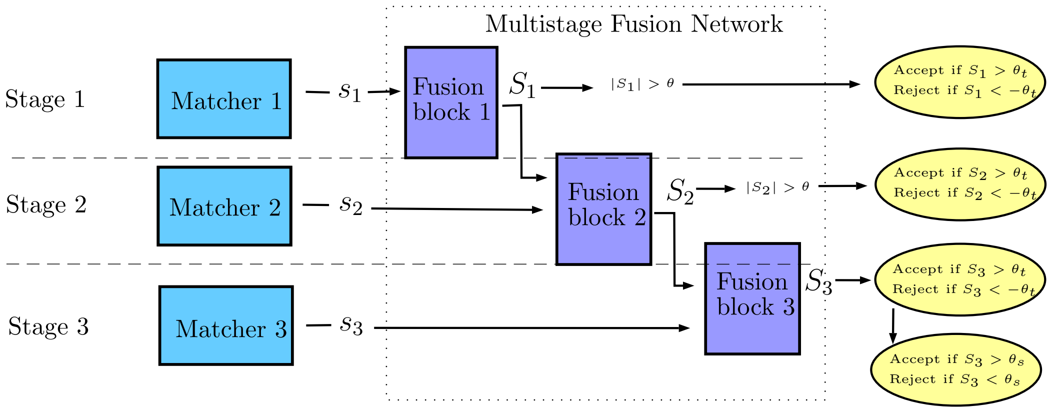
Multistage Fusion Network
Stage 1
Stage 2
Stage 3
Matcher 1
Matcher 2
Matcher 3
s_1
s_2
s_3
Fusion block 1
Fusion block 2
Fusion block 3
S_1
S_2
S_3
$|S_1| > \theta$
$|S_2| > \theta$
Accept if $S_1 > \theta_t$
Reject if $S_1 < -\theta_t$
Accept if $S_2 > \theta_t$
Reject if $S_2 < -\theta_t$
Accept if $S_3 > \theta_t$
Reject if $S_3 < -\theta_t$
Accept if $S_3 > \theta_s$
Reject if $S_3 < \theta_s$

Figure 2. Proposed multistage fusion network architecture.

culation of the comparison score between the probe and reference templates. In verification systems this score is compared to some predetermined threshold to confirm or deny the verification claim. Two types of errors are present when making this decision - confirming the match for non-mated pair of probe and reference templates, and denying the match for mated pair. We label the error rates of this decision correspondingly as $FAR(\theta)$, false accept rate, and $FRR(\theta)$, false reject rate, both depending on chosen comparison score threshold θ.

The trade-off between two error rates is typically represented by ROC curve, and the operating threshold for the biometric system deployment can be chosen in a number of ways. Frequently, the threshold is chosen so that the FAR is no larger than some small number, e.g. $FAR(\theta) = .1\%$. It is a reasonable approach, but a more theoretically optimal method of choosing operating threshold will rely on Bayesian risk or cost minimization [19]. If we denote the cost of making false accept error as λ_{FA}, and the cost of making the false reject error as λ_{FR}, then the total cost, or risk, of our decision is

$$C(\theta) = \lambda_{FA}FAR(\theta) + \lambda_{FR}FRR(\theta) \tag{1}$$

The optimal operating threshold θ is found by minimizing the cost $C(\theta)$. Now, if we want to incorporate the matching time into the cost equation, then we would have to add a third term into like this:

$$C(\theta) = \lambda_{FA}FAR(\theta) + \lambda_{FR}FRR(\theta) + \lambda_T T(\theta) \tag{2}$$

where $T(\theta)$ is the total time of running the system, and λ_T is the time cost. But there is a problem with such equation: using the single score threshold parameter θ, we would not be able to properly balance both matching error rate and time costs. For example, if we set θ so that the matching time is limited by some constant, then both FAR and FRR will be set. Alternatively, if θ is used as a threshold to the final comparison score of the system, then it would have little influence on its running time. From a more general point of view, if our matching system has three variable performance characteristics, then it is reasonable to assume that their relationships are controlled by two parameters; the set of possible performance values thus constitutes a two-dimensional surface in three-dimensional performance value space. In contrast, traditional matching systems of eq. 1 have two performance values, whose relationships is represented by ROC curve - a one-dimensional curve in two-dimensional performance value space.

Given these considerations, we will assume that the multistage fusion system has to be controlled by two parameters, say θ_s and θ_t. In our implementation we will primarily associate θ_s with the trade-off between two matching error rates, FAR and FRR, and θ_t will mostly control the matching time, and thus will control the trade-off between time and two error rates. Thus, we will rewrite the cost equation of our matching system as

$$\begin{aligned} C(\theta_s, \theta_t) = &\lambda_{FA}FAR(\theta_s, \theta_t) + \lambda_{FR}FRR(\theta_s, \theta_t) \\ &+ \lambda_T T(\theta_s, \theta_t) \end{aligned} \tag{3}$$

Note, that all performance values (FAR, FRR, T) still depend on both threshold parameters. For example, if we change the threshold θ_t to reduce the running time T, then we might expect that matching error rates, FAR and FRR, might increase. From the other side, if we change the threshold associated with matching rates, θ_s, then it is possible that the fusion score will reach the decision threshold set by θ_s at earlier or later stage, and, correspondingly, the running time T will be reduced or increased.

In contrast to traditional matching systems, finding optimal threshold parameters in proposed multistage fusion system requires a little more computation. Instead of iterating over the possible values of a single threshold, we have to iterate over values of threshold pair (θ_s, θ_t) and calculate

FAR, FRR, T for each pair. Effectively, the time to calculate the optimal system thresholds is $O(n^2 \log(n))$, while for traditional systems it is $O(n \log(n))$, where n is the total number of test samples.

4. Multistage Fusion Network

In this work we propose the use of the single neural network for fusing biometric comparison scores in the multistage systems. The diagram of the proposed system is given in Fig. 2. The structure of the network mirrors the work of the multistage fusion system - each stage is represented by a separate fusion block, and the execution flows from the first stage to the last.

The comparison score from the mth matching algorithm is not available to the fusion blocks at stages $1, \dots, m-1$, but only at stages m and later. The fused score from each stage is available as a separate output of the neural network. Such network structure allows the calculation of fused score at stage m even if matching algorithms $m+1, \dots, M$ have not run and the comparison scores from these algorithms are not available (M is the total number of stages). In practice, during testing run, after mth matching algorithm executes and its comparison score is available, we restart the fusion network with all currently available scores from algorithms $1, \dots, m$ while substituting some dummy values for algorithms $m+1, \dots, M$; the fused score from mth fusion block is not affected by dummy values and we use it to decide if fusion should be terminated or continued to the next stage.

The base implementation of the network uses cross-entropy loss function, in which all output fused scores are trained to approximate the probability of genuine or impostor verification attempt. Note, that since we want to use a single threshold θ_t to advance to next stage, we map the fused scores it to interval $[-1, 1]$ for decision. Output values with large absolute values indicate more confident fuse scores; we accept the fused score s_k at stage k if its absolute value is larger than the time threshold: $|s_k| > \theta_t$. The uniform training of the network implies that the fused scores at each stage are comparable, and using same threshold θ_t for early fusion decision is justified. At the end, the final fused score is the score s_k of the last run stage; it is compared against θ_s to make accept or reject decision of the system.

4.1. Time cost sensitive optimization

Although the base implementation of the multistage fusion network does offer a unified approach to training different stages of the fusion function, it does not really account for the time costs of different matchers. For example, if one matcher runs longer than others, the base network will not account for it. To do this, we modify the training loss function by varying the desired magnitude of the fused scores:

FAR	1%	.1%	.01%
FR1	96.48	93.28	87.73
FR2	95.51	91.23	84.23
FR3	91.43	84.22	70.54
Fusion	96.59	93.63	88.50

Table 1. Performance of the three face recognition algorithms and a traditional fusion on IJB-A dataset. TAR (%) at different FAR levels are presented.

$$L = Loss_{CE}(\mathbf{s}, y) - \sum_{m=1}^{M} \lambda_m s_m^2 \tag{4}$$

Here $\mathbf{s} = \{s_1, \dots, s_M\}$ are output fused score for M stages, y is the desired output.

The reasoning for this equation and choice of parameters is following. Larger λ_m will encourage fused score s_m to have larger magnitude to minimize the cost, and, correspondingly, will make it more probable for the multistage network to make a decision to accept or reject current fusion results, rather than continue to the next stage. The optimal choice of λ_m therefore depends on the running time of subsequent matchers $m+1, \dots, M$; larger running time of these matchers leads to the need to terminate the fusion at earlier stages and to the choice of larger optimal value of λ_m. Implicitly, the choice of λ_m also depends on the performance strength of subsequent matchers; stronger matchers should lead to lower desired fusion scores and lower λ_m. Without loss of generatlity we set the last parameter of λ to zero: $\lambda_M = 0$. Since M is the last stage, we can't delay the matching decision and there is no need to perform time cost related score adjustments at this stage.

In our implementation we treat λ_m as hyperparameters during network training. Though it is preferable to have some theoretical derivation for their optimal values, the relationships between running time, matcher performance and optimal choice of λ_m seems to be too complex to allow it. Alternatively, we might wish to directly connect the optimizing loss function and choice of λ_m to the cost function of Eq. 3, but this seems to be even harder to achieve. Another complicating factor is that in our framework the operating thresholds θ_s and θ_t are obtained after network is trained, and thus increasing or decreasing λ_m might not have direct effect on the rate of accepting samples at particular stage; the relationships between different λ_m seems to be the important factor instead.

5. Experiments

For our experiments, we utilize the sets of face recognition comparison scores from three face recognition algorithms: FR1([1]), FR2([17]) and FR3([15]). All three face

FAR	1%	.1%	.01%
FR1	97.68	95.37	90.50
FR2	96.12	91.77	83.46
FR3	94.49	89.14	80.58
Fusion	97.66	95.45	90.58

Table 2. Performance of the three face recognition algorithms and a traditional fusion on IJB-C dataset. TAR (%) at different FAR levels are presented.

recognizers are based on different configurations of deep convolutional neural networks, and trained using different datasets.

The comparison scores are derived from IARPA Janus Benchmark-A (IJB-A) dataset [9] and from IARPA Janus Benchmark-C (IJB-C) [12] dataset, which is a superset of the original IJB-A. The testing protocols specify gallery and probe templates with different numbers of constituent face images and video frames (from 1 to more than 100). The template feature vectors are obtained by averaging facial features vectors of images and video frames, and the comparison scores between probe and reference templates are obtained using cosine distance. We assume that each of these algorithms takes time of 1.0 units to perform a comparison between two templates; since the templates are precalculated as 128-dimensional feature vectors, this assumption seems reasonable. In real life, the comparison procedure will probably involve feature extraction by CNN for probe templates, but we might assume that the time to do it might also be the same for these algorithms.

IJB-A dataset provides the experimental protocol of 10 splits, and for each split subject disjoint training and testing subsets are specified. In our experiments we utilized this protocol by training a multistage fusion network on training subsets and testing on test subsets for each split. IJB-C does not have such split separations, and all templates are considered to be test templates. Therefore, for our experiments, we created ten randomized splits with subject disjoint training and testing subsets similar to IJB-A protocol. In both cases, the number of genuine samples per training procedure is around 2,000, and we limited the number of impostor samples to 100 per each genuine sample to avoid instability issues during training. Half of the training set is used for network training and other half for validation.

The presented fusion task is quite challenging since these face matchers produce highly correlated comparison results, and the addition of two matchers with lower performance gives only very small improvements to the best performing one. The traditional performance results from the neural network fusion are presented in tables 1 and 2.

We implement each fusion block as fully connected network with 2 hidden layers consisting of 10 nodes each. Such network configuration seems to be sufficient to capture the distribution of comparison scores, while avoiding overfitting issues. Implementation was based on Caffe network library [8].

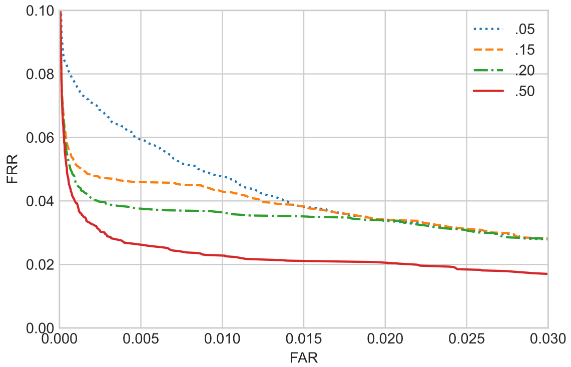

Figure 3. Sample ROC curves for different time thresholds t of the same multistage fusion system.

The implemented multistage fusion algorithms are compared to two baseline algorithms:

1. The best algorithm performing alone; the running time of this single algorithm is 1.0.

2. All three algorithms performing comparison and their results are fused by traditional network (fully connected with 2 hidden 10-node layers); the running time of this method is 3.0.

The ‘Multistage’ fusion algorithm in tables 3 and 4 refers to the base implementation of the multistage fusion network described in the beginning of section 4. The ‘Optimization’ refers to the optimized algorithm described in section 4.1. For both datasets, the hyperparameters λ_m of Eq. 4 were chosen to be $(\lambda_1, \lambda_2, \lambda_3) = (.3, .3, 0)$.

The performance values for different cost functions, e.g. $1FRR + 10FAR + .01T$, and for each algorithm are given in the tables. Note, that we optimize the costs using Eq. 3, so in each case potentially different thresholds θ_s and θ_t are found, minimizing corresponding cost value. The components of the resulting optimized cost, i.e. $FAR(\theta_s, \theta_t)$, $FRR(\theta_s, \theta_t)$ and $T(\theta_s, \theta_t)$, are given as well. Note, that the summation of the components with corresponding weights sums to the costs, and all numbers are the averages of experiments using 10 splits of the datasets.

In all cases, we see that the multistage fusion does indeed result in the reduction of the total cost of system. Most of the reduction seems to be caused by the transitioning to multistage/two threshold system framework, and optimization procedure of section 4.1 has relatively limited effect. It is possible that the optimization will be more useful when

Algorithm	Performance measures	1FRR+10FAR +.01T	1FRR+100FAR +.01T	1FRR+1000FAR +.01T	1FRR+10000FAR +.01T
Baseline 1	Cost	.0807	.1266	.2295	.5242
	FRR,FAR(%)	5.40,.168	8.78, .029	13.2,.0087	35.9,.0016
	T	1.0	1.0	1.0	1.0
Baseline 2	Cost	.0981	.1423	.2410	.5233
	FRR,FAR(%)	5.49,.132	8.29,.029	12.34,.0088	34.81,.0015
	T	3.0	3.0	3.0	3.0
Multistage	Cost	.0784	.1214	.2187	.5047
	FRR,FAR(%)	5.48,.132	8.52,.026	12.54, .0083	33.08, .0016
	T	1.043	1.011	1.031	1.230
Optimization	Cost	.0779	.1155	.1526	.2313
	FRR,FAR(%)	5.47,.129	8.55,.020	12.77, .0015	17.17, .0005
	T	1.032	1.015	1.039	1.065

Table 3. Multistage fusion algorithm performance on IJB-A dataset.

Algorithm	Performance measures	1FRR+10FAR +.01T	1FRR+100FAR +.01T	1FRR+1000FAR +.01T	1FRR+10000FAR +.01T
Baseline 1	Cost	.0654	.1060	.1706	.2832
	FRR,FAR(%)	4.27,.126	7.23, .024	12.51,.0035	19.49,.0008
	T	1.0	1.0	1.0	1.0
Baseline 2	Cost	.0843	.1247	.1901	.3074
	FRR,FAR(%)	4.07,.136	7.20,.023	12.66,.0034	19.57,.0008
	T	3.0	3.0	3.0	3.0
Multistage	Cost	.0644	.1044	.1690	.2798
	FRR,FAR(%)	4.07,.135	7.23,.022	12.33, .0036	19.72, .0007
	T	1.007	1.0008	1.003	1.007
Optimization	Cost	.0639	.1037	.1692	.2782
	FRR,FAR(%)	4.08,.130	7.14,.022	12.29,.0036	19.62,.0007
	T	1.002	1.002	1.007	1.008

Table 4. Multistage fusion algorithm performance on IJB-C dataset.

working with more diverse recognizers, both in accuracy and time performance characteristics.

As an additional illustration to the proposed multistage fusion method and performance measures, in Fig. 3 we draw ROC curves obtained by fixing a time related threshold θ_t and letting score threshold parameter θ_s to change. As we discussed in section 3, these curves are one dimensional sections of two-dimensional surface in three dimensional performance value space: $(FAR(\theta_s, \theta_t)$, $FRR(\theta_s, \theta_t)$, $T(\theta_s, \theta_t))$. The ROC curve corresponding to the threshold value .50 has best performance, since in this case almost none of fused scores at earlier stages is getting accepted, and the ROC curve consists mostly from the scores obtained at the last fusion stage. In contrast, the ROC curve corresponding to the threshold .05 has most scores obtained from in the first stage, and has worse performance.

Fig. 3 shows that it might be difficult to compare the performance of multistage fusion systems using fixed value of FAR, as it is done in many papers on biometric recognition. Instead, we have to look at all three performance values, and comparing their weighted sum in cost function seems to be an adequate solution. Note also, that in tables 3 and 4 the reduction of total cost from one algorithm to another does not necessarily implies the reduction of a particular performance value, e.g. FAR, but of the combination of all three performance numbers.

From another point of view, suppose we want to achieve a smaller FAR value for a given system. Intuitively, one can think that this can be achieved by increasing the system running time and by running later stage recognizers more frequently. The experiments show that this might not be always true. The comparison of $10FAR$ and $100FAR$ columns in table 3 shows that the reduction in FAR did occur in spite of decrease in average running time (from 1.043 to 1.011 for multistage, and from 1.032 to 1.015 for optimized versions). This apparently strange behavior is explained by large increases in FRR. This example shows that the use of two thresholds for optimizing the perfor-

Algorithm	1FRR+10FAR +.01T	1FRR+100FAR +.01T	1FRR+1000FAR +.01T	1FRR+10000FAR +.01T
Baseline 1	.0848	.1283	.2108	.3908
Baseline 2	.0764	.1145	.1751	.2877
Multistage	.0623	.0983	.1558	.2705
Optimization	.0619	.0983	.1571	.2665

Table 5. Multi-sample fusion algorithm performance on IJB-C dataset, face recognizer 1.

Algorithm	1FRR+10FAR +.01T	1FRR+100FAR +.01T	1FRR+1000FAR +.01T	1FRR+10000FAR +.01T
Baseline 1	.1291	.2105	.3400	.5734
Baseline 2	.1019	.1615	.2556	.4098
Multistage	.0903	.1481	.2420	.3940
Optimization	.0891	.1480	.2411	.3933

Table 6. Multi-sample fusion algorithm performance on IJB-C dataset, face recognizer 2.

mance of multi-stage system might lead to non-intuitive FAR and FRR performance settings, and that relying on a single total cost performance is a preferable approach.

6. Multi-sample score fusion

The multistage fusion architecture can also be utilized in the problem of fusing multi-sample comparison scores, where the scores are produced by the single matching algorithm for the multiple observations of the same person. In our experimental IJB-A and IJB-C, the facial templates are created from the multiple images of the same person. We can assume a scenario where these images are obtained in a sequence, and there is a cost associated with the acquisition of each subsequent image. The goal of the fusion algorithm will be to not only fuse the corresponding comparison scores in a most efficient way, but to also decide if it is beneficial to terminate the matching process given already processed images, or continue acquiring and recognizing new images. This scenario can also be in the settings of continuous authentication and multi-frame fusion for video authentication [3].

Note that the traditional multi-sample fusion [22, 16] tries to achieve the best recognition accuracy by implicitly weighing the samples and agglomerating corresponding feature vectors according to calculated weights. It might be possible that during such fusion some samples would be assigned a weight of zero, and be omitted from the fused template. Thus it might appear that such fusion is similar to the multi-stage fusion task considered in the current paper. But above papers still process all available samples to achieve best performance and do not consider the time costs of acquiring and processing multiple sample. In contrast, our proposed multi-stage fusion architecture takes into account the time costs, and can be used as a complementary step in such systems.

In our experiments on multi-sample fusion we consider each of three face recognizers separately. The gallery templates are obtained by averaging the feature vectors of all images in the template as before, but for probe templates we assume the sequential accumulation of features: given first k probe template images, we construct k-th probe template $F_{p,k}$ by averaging feature vectors of these first k images. The k-th sequential comparison score is obtained by matching gallery template F_g with this k-th probe template: $s_k = Matcher_m(F_g, F_{p,k})$. Next, since we limited our experiments to 3-stage fusion architecture and the number of images in templates varies, for each gallery and probe template pair we select at random 3 scores s_k to be used as inputs to multi-stage fusion network. If probe template has less than three images, we exclude it from experiments. It appears that this experiment arrangement directly corresponds to realistic operating scenario in continuous authentication system: the subsequent frames are processed and feature vectors are agglomerated, and at random intervals the template comparison, score fusion and decision to continue is performed.

The results of the multistage fusion experiments for such operating scenario are presented in tables 5, 6 and 7 for corresponding face recognizers of FR1([1]), FR2([17]) and FR3([15]). IJB-C dataset was used for template calculation, and 10 random person-separated splits into training and testing subsets were generated as before. For clarity, we omitted the corresponding FAR, FRR and average running time performance numbers.

The time cost optimization of multi-stage network proposed in Eq. 4 had less effect on performance improvement in these experiments. The tables show the optimized performance achieved with relatively small parameter values $(\lambda_1, \lambda_2, \lambda_3) = (.05, .05, 0)$. The larger values of $(\lambda_1, \lambda_2$ led

Algorithm	1FRR+10FAR +.01T	1FRR+100FAR +.01T	1FRR+1000FAR +.01T	1FRR+10000FAR +.01T
Baseline 1	.1495	.2299	.3329	.4623
Baseline 2	.1196	.1887	.2778	.3945
Multistage	.1078	.1754	.2622	.3756
Optimization	.1071	.1751	.2626	.3773

Table 7. Multi-sample fusion algorithm performance on IJB-C dataset, face recognizer 3.

to rather mixed results. It is possible that the training of non-optimized version of multi-stage network results in more balanced output scores, whose values in different stages better correlate with the time costs. Thus, more confident score (having larger absolute value) reflects not only additional image data used to obtain it, but the proportional time cost as well. The situation might be different for the task of fusing different recognizers of previous section, where additional algorithms lead to smaller improvements while requiring significant running costs.

7. Conclusion

In this work we analyzed the problem of serial or multistage fusion and its relationship to the traditional parallel fusion problem. We emphasized the need to incorporate the trade-off between recognition accuracy error rates and time costs into the evaluation of multistage fusion algorithms. In our implementation of multistage fusion we separated the optimization of the operating threshold parameters from the fusion algorithm, and speculated on the need to have two thresholds, and, correspondingly, two parameters controlling the system execution. Our fusion is implemented as a single neural network where the learning of different stages of fusion is defined by the loss function hyperparameters. The experiments with two possible scenarios of utilizing face recognizer, the fusion of recognition algorithms and the fusion of scores at different stages of continuous authentication by the single recognizer, show good performance of the proposed method.

As we mentioned in section 4.1 the hyperparameter solution to connect different stages of the fusion system might not be the best one. In the future work, it would be desirable to remove this limitation, and perform better optimized training procedure. Note, that in this case, the running times of each algorithm should be supplied as input parameters to the network. It would also be desirable to better reflect the final system cost given by Eq. 3 during the training of fusion system. As noted in [11], the particular trade-off between error rates might favor the use of one or the other recognizer; in our case, the choice of cost function of Eq. 3 could lead to particular choice of hyperparameters λ_m of Eq. 4.

Acknowledgement

This material is based upon work supported by the Center for Identification Technology Research and the National Science Foundation under Grant # 1822190.

References

[1] J. C. Chen, V. M. Patel, and R. Chellappa. Unconstrained face verification using deep CNN features. In *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–9, 2016. 4, 7

[2] Luigi Pietro Cordella, Alessandro Limongiello, and Carlo Sansone. Network intrusion detection by a multi-stage classification system. Multiple Classifier Systems, pages 324–333, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg. 2

[3] D. Crouse, H. Han, D. Chandra, B. Barbello, and A. K. Jain. Continuous authentication of mobile user: Fusion of face image and inertial measurement unit data. In *2015 International Conference on Biometrics (ICB)*, pages 135–142. 7

[4] H. El-Shishiny, M. S. Abdel-Mottaleb, M. El-Raey, and A. Shoukry. A multistage algorithm for fast classification of patterns. *Pattern Recognition Letters*, 10(4):211–215, 1989. 2

[5] Julian Fierrez, Aythami Morales, Ruben Vera-Rodriguez, and David Camacho. Multiple classifiers in biometrics. part 1: Fundamentals and review. *Information Fusion*, 44:57–64, 2018. 1

[6] Yoav Freund and Robert E. Schapire. Experiments with a new boosting algorithm. In *International Conference on Machine Learning*, pages 148–156, 1996. 2

[7] Lin Hong and Anil Jain. Integrating faces and fingerprints for personal identification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12):1295–1307, 1998. 2

[8] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014. 5

[9] B. F. Klare, B. Klein, E. Taborsky, A. Blanton, J. Cheney, K. Allen, P. Grother, A. Mah, M. Burge, and A. K. Jain. Pushing the frontiers of unconstrained face detection and recognition: IARPA Janus Benchmark A. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1931–1939, 2015. 5

[10] M. Last, H. Bunke, and A. Kandel. A feature-based serial approach to classifier combination. *Pattern Analysis and Applications*, 5(4):385–398, 2002. 2

[11] Gian Luca Marcialis, Fabio Roli, and Luca Didaci. Personal identity verification by serial fusion of fingerprint and face matchers. *Pattern Recognition*, 42(11):2807–2817, 2009. 2, 8

[12] Brianna Maze, Nathan D Kalka, James Duncan, Anil Jain, Jocelyn Adams, Tyler Niggel, Patrick Grother, Tim Miller, Jordan Cheney, and Charles Otto. Iarpa janus benchmark – c: Face dataset and protocol. In *International Conference on Biometrics*, 2018. 5

[13] N. Poh, T. Bourlai, J. Kittler, L. Allano, F. Alonso-Fernandez, O. Ambekar, J. Baker, B. Dorizzi, O. Fatukasi, J. Fierrez, H. Ganster, J. Ortega-Garcia, D. Maurer, A. A. Salah, T. Scheidat, and C. Vielhauer. Benchmarking quality-dependent and cost-sensitive score-level multimodal biometric fusion algorithms. *IEEE Transactions on Information Forensics and Security*, 4(4):849–866, 2009. 1

[14] P. Pudil, J. Novovicova, S. Blaha, and J. Kittler. Multistage pattern recognition with reject option. In *Proceedings., 11th IAPR International Conference on Pattern Recognition. Vol.II. Conference B: Pattern Recognition Methodology and Systems*, pages 92–95, 1992. 2

[15] Rajeev Ranjan, Swami Sankar, Carlos D. Castillo, and Rama Chellappa. An all-in-one convolutional neural network for face analysis. In *Automatic Face & Gesture Recognition (FG 2017), 2017 12th IEEE International Conference on*, pages 17–24, 2017. 4, 7

[16] Nishant Sankaran, Sergey Tulyakov, Srirangaraj Setlur, and Venu Govindaraju. Metadata-based feature aggregation network for face recognition. In *International Conference on Biometrics (ICB 2018)*, 2018. 7

[17] Swami Sankaranarayanan, Azadeh Alavi, Carlos D. Castillo, and Rama Chellappa. Triplet probabilistic embedding for face verification and clustering. In *2016 IEEE 8th International Conference on Biometrics Theory, Applications and Systems (BTAS)*, 2016. 4, 7

[18] Maneet Singh, Richa Singh, and Arun Ross. A comprehensive overview of biometric fusion. *Information Fusion*, 52:187–205, 2019. 1

[19] S. Theodoridis and Koutroumbas K. *Pattern Recognition*. Academic Press, 1999. 3

[20] Kirill Trapeznikov, Venkatesh Saligrama, and David Castañón. Multi-stage classifier design, 2012. Steven C. H. Hoi Wray Buntine 459–474 25. 2

[21] Andreas Uhl and Peter Wild. Parallel versus serial classifier combination for multibiometric hand-based identification. In *International Conference on Biometrics*, pages 950–959, 2018. 1

[22] J. Yang, P. Ren, D. Zhang, D. Chen, F. Wen, H. Li, and G. Hua. Neural aggregation network for video face recognition. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5216–5225, 2017. 7