Hybrid Feature Learning for Handwriting Verification

Mohammad Abuzar Shaikh*, Mihir Chauhan†, Jun Chu‡ and Sargur Srihari§
The Department of Computer Science and Engineering
The State University of New York at Buffalo
Buffalo, NY, USA
Email: *mshaikh2@buffalo.edu, †mihirhem@buffalo.edu, ‡jchu6@buffalo.edu, ¶srihari@cedar.buffalo.edu,

Abstract—**We propose an effective Hybrid Deep Learning (HDL) architecture for the task of determining the probability that a questioned handwritten word has been written by a known writer. HDL is an amalgamation of Auto-Learned Features (ALF) and Human-Engineered Features (HEF). To extract auto-learned features we use two methods: First, Two Channel Convolutional Neural Network (TC-CNN); Second, Two Channel Autoencoder (TC-AE). Furthermore, human-engineered features are extracted by using two methods: First, Gradient Structural Concavity (GSC); Second, Scale Invariant Feature Transform (SIFT). Experiments are performed by complementing one of the HEF methods with one ALF method on 150000 pairs of samples of the word "AND" cropped from handwritten notes written by 1500 writers. Our results indicate that HDL architecture with AE-GSC achieves 99.7% accuracy on seen writer dataset and 92.16% accuracy on shuffled writer dataset which out performs CEDAR-FOX, as for unseen writer dataset, AE-SIFT performs comparable to this sophisticated handwriting comparison tool.**

Index Terms—**Handwriting verification, Handwriting comparison, Deep Learning, Forensic Analysis, Siamese Network, AutoEncoder (AE), Gradient Structural Concavity (GSC), Hybrid Deep Learning (HDL), Scale Invariant Feature Transform (SIFT), Two Channel Convolutional Neural Network (TC-CNN), Auto-Learned Features (ALF), Human-Engineered Features (HEF) .**

I. Introduction

Handwriting comparison is a task to find the likelihood of similarity between the handwritten samples of the known and questioned writer. This comparison is based on the hypothesis that every individual has peculiar handwriting. Furthermore, the exclusive nature of individuals handwriting helps differentiate between handwritten samples of distinct individuals. State-of-the-art handwriting analysis tool, CEDAR-FOX was developed at Center of Excellence for Document Analysis and Recognition, University at Buffalo to validate the idea of writer individuality. Srihari et al. in Individuality of Handwriting [1] compares the intra-writer feature variation with inter-writer feature variations for the task of handwriting verification and identification. The features used to find these variations were conventional and computational features. Conventional features were extracted using twenty-one rules of discriminating elements of handwriting [2] [3] [4]. Computational features were computed by algorithms and comprised of macro and micro features. Eleven macro features were used to describe paragraph, line, word and character level features which closely resembles five conventional features: pen pressure, writing movements, stroke formation, slantness and height. Micro features were used to describe character and allograph level features. Gradient Structural Concavity feature (GSC) algorithm was used to compute 512 micro binary feature vector. CEDAR-FOX uses the feature variations to compute log likelihood ratio (LLR) for a given pair of input handwritten samples. The results achieved by CEDAR-FOX were state-of-the-art for verification task. Since then there has been significant efforts to improve the accuracy of the writer verification model by bridging the gap between human understanding and feature representation of an image.

The main contribution of our work is a new hybrid feature set obtained by unifying the handcrafted features from SIFT and deeply learned features from twin Auto-Encoder. In the first hybrid feature set, we have used SIFT [5] as a handcrafted feature extractor. SIFT has found tremendous application in the area of computer vision, image retrieval and recognition task, but has been underutilized in the area of document analysis as suggested by [6]. SIFT captures point level features and comprises of key-point descriptors. Note that it is not our aim to apply entire SIFT based method which mostly relies on BoW methods for generating fixed size feature vector from variable size keypoint descriptors. Instead we use nearest neighbour matching algorithm to map and compare similar key-points between the given two handwriting samples. The resulting mapped features forms one half of our feature set.

The other half is contributed by deeply learned features extracted by Convolutional Neural Networks (CNN). CNNs are often compared to human brain cortex because of hierarchical architecture and richer feature set. CNN has provided state-of-the-art performance in several vision task (e.g document recognition, image classification, object detection, etc). Deep learning networks using CNN have found application in various verification task such as signature verification (e.g SigNet [7]) and face verification [8]. For the task of handwriting comparison we have implemented a baseline architecture for feature extraction using Siamese Network [9] which is essentially a Two Channel-CNN (TC-CNN) network. Furthermore, we have implemented an advanced deep learning model, Two-Channel Auto-Encoder (TC-AE) [10] in which the latent representation forms the basis of our feature set.

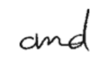
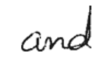
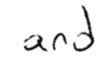
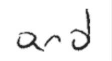
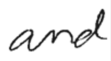
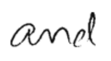
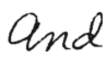
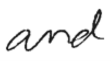
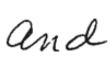
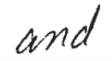
										
Sample ID [XXXXy_numZ]	0001a_num1	0001a_num2	0002a_num1	0002a_num2	0005b_num1	0005b_num2	1121a_num1	1121a_num2	1160a_num1	1160a_num2
Writer Number [XXXX]	Writer 0001	Writer 0001	Writer 0002	Writer 0002	Writer 0005	Writer 0005	Writer 1121	Writer 1121	Writer 1160	Writer 1160
Page Number [y]	Page 1	Page 1	Page 1	Page 1	Page 2	Page 2	Page 1	Page 1	Page 1	Page 1
Sample Number [Z]	Sample 1	Sample 2	Sample 1	Sample 2	Sample 1	Sample 2	Sample 1	Sample 2	Sample 1	Sample 2

Fig. 1: Dataset Description

The deeply learned features extracted by using TC-CNN/TC-AE are appended to the handcrafted features extracted using SIFT in the first hybrid setting and to the rule based GSC features in the second hybrid setting. Complementary CNN and SIFT [11] shows that in general, CNN complements SIFT. Furthermore, [12] shows that although CNN achieves decent performance, we cannot always infer that CNN will outperform SIFT. Hence, the hybrid deep learning (HDL) model was inspired to capture richer features by combining deep learning with handcrafted features as done by [13] [14]. Experiments are performed to compare the accuracy of SIFT, GSC, TC-CNN, TC-AE and HDL on CEDAR letter dataset. Furthermore, we have validated our results using batch verification functionality of CEDAR-FOX.

II. Dataset

Our dataset comprises of "AND" images extracted from CEDAR Letter dataset [1]. The CEDAR dataset consists of handwritten letter manuscripts written by 1567 writers. Each of the writer has copied a source document thrice. Hence, there are a total of 4701 handwritten letter manuscripts. Each letter has vocabulary size of 156 words (32 duplicate words, 124 unique words) and has one to five occurrences of the word "AND" (cursive and hand printed). Image snippets of the word "AND" were extracted from each of the manuscript using transcript-mapping function of CEDAR-FOX [15]. The total number of "AND" image fragments available after extraction are 15,518. Figure 1. shows examples of the "AND" image fragments.

A. Choice of Dataset

Our motivation for using the word "AND" for the handwriting comparison task is based on two premises:

- Abundance in English Language: The conjunction "AND" ranks 4th in the list of most frequently used words by Corpus of Contemporary American English (COCA).
- Availability in CEDAR dataset: On an average there are 10 samples of the word "AND" from each of the writer. These samples are good enough for comparing the intra-class variations within the same writer and the inter-class variation between different writers.

B. Data Inconsistency

Improper transcript-mapping of "AND" fragments from manuscript would lead to inconsistencies in the dataset. For example, handwritten symbols extracted by the transcript-mapping tool may produce outliers as "AND" words which would lead to reduced accuracy of the overall system. To avoid data inconsistency, outlier symbols are removed at the data validation step by rejecting incorrect input symbols.

C. Data Preprocessing

Every handwritten image is padded uniformly on all the four edges so as to have consistent size corresponding to the maximum width and height (384x384) across all the samples. We then downscale the image by a factor of six resulting in a square image of size 64x64.

D. Data Partitioning

We have compared three approaches for partitioning the dataset for training and testing. Each of these three approaches described below would have an impact on the training and testing accuracy as shown in the result section:

- Unseen Writer Partitioning: In this method there exists no writer which is present in both the training (Tr) and testing (Ts) writer set simultaneously. Hence, any test writer would not be a part of training set and vice-versa.

$$T_r \bigcap T_s = \emptyset \tag{1}$$

- Shuffled Writer Partitioning: In this method, entire dataset is first shuffled. Hence, there are X writers which are concurrent in both the training (Tr) and testing (Ts) writer set. Hence, given a test writer may or may not be present in the training set.

$$T_r \bigcap T_s = X \tag{2}$$

- Seen Writer Partitioning: In this method, we train over 80% of each writers samples and test over the remaining 20% samples of each writer.

$$T_s = \bigcup_{j=1}^{N} 0.2 * S_j \tag{3}$$

$$T_r \bigcup T_s = S \tag{4}$$

III. Deep Learning

Our approach is based on using CNN as a feature extractor. Recently, many popular CNN architectures have been employed for feature extraction (e.g VGG [16], Alexnet [17], Resnet [18], Inception [19]). Since our goal is to focus on testing baseline models, we have designed a simplistic five layer CNN model for high level feature extraction. At the same time, since the AND images are in grayscale with minimal pixel, hence, we remove the max pooling layer so that we do not lose any important datapoint. However, we use a valid padding in CNN with larger receptor size to reduce the dimensionality. Furthermore we expand our networks to learn the features of two input images simultaneously.

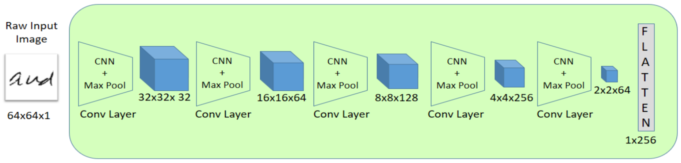


Fig. 2: CNN Architecture

A. *Two channel CNN (TC-CNN)*

Most commonly a two channel CNN network, is called as Siamese Network named after the work done by Bromley, et al [9], where, they use an energy function to compute their loss. We have a similar setting where the two networks share weights. Essentially the same network is responsible to generate features for the two different images (I_i, I_j) in a parallel setting. We do not use energy function to calculate our loss, but we experiment on multiple functions to compare the features (F_i, F_j) from I_i and I_j. In the first setup we tie (F_i, F_j) by concatenating them $(F_x\_conc = F_i \bigcup F_j)$, and then train F_x_conc over Fully Connected layers. In the second setup we find the difference (F_x_diff) between (F_i, F_j). This feature difference vector is then used as an input and trained over Fully Connected layers. Finally we obtain the output in the form of a two class classifier and hence categorical cross entropy (CCE) loss function after the last softmax layer becomes an obvious choice. We design a simplistic model containing five Convolution layers each followed by a max pooling layer as displayed in Figure 2. Two gray scale images of height h=64 and width w=64 are input to the first convolution layer containing 32 kernels k of size 3x3. Max pool layers of stride 2x2 are introduced to reduce the h to h/2 and w to w/2 after each convolution. We double the size of k until the fourth conv layer. For the fifth conv layer k=64 so that post flatten the total number of neurons output from one channel is 256.

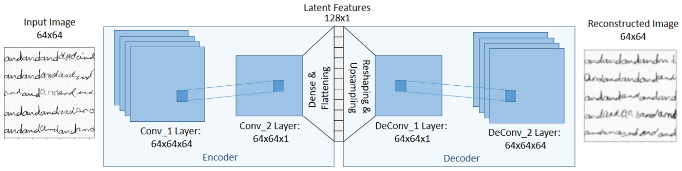


Fig. 3: Auto Encoder Architecture

B. *Two channel Autoencoder (TC-AE)*

Although CNN is an excellent feature extractor, it maps the images to only the class they belong, which in this case would be great if our task was a writer identification task. In CNN the difference between an output class and the actual class is propagated back and the weights are updated to only have a better mapping between F_i, F_j and class C_{ij} (Where C_{ij} belongs to 0,1). However, since the "and" images are in grayscale, we need more robust features that describe the image, not necessarily mapping any class. We do this by reconstructing the image using an AutoEncoder, and by adding the reconstruction Mean Absolute Error (MAE) between the logistic outputs and the image pixels as a regularizer. Furthermore, we do the same setup, as displayed in TC-CNN, considering the latent variables L_i, L_j of I_i, I_j as F_i, F_j. We train the AE for a batch b = 512 of images that appear in each channel one by one. The encoder architecture contains two Convolution layers with k=64 and k=1 respectively, followed by three fully connected (FC) layers with neurons n=4096, n=1024, n=128 respectively, as shown in Figure 3. In the decoder architecture the encoder is simply reversed to maintain balance between encoder and decoder, however, we apply upsampling with cubic interpolation (CI) after the Convolution layers to resize the 2D arrays to size 64x64. The FC layer in encoder with n = 128 acts as latent representation of the input. In one training iteration the weights AE are updated, then encoder is duplicated to generate two channels. Each channel outputs feature vectors F_i and F_j of length 128. Finally, the same operations are performed over F_i and F_j as performed in the two setups of TC-CNN.

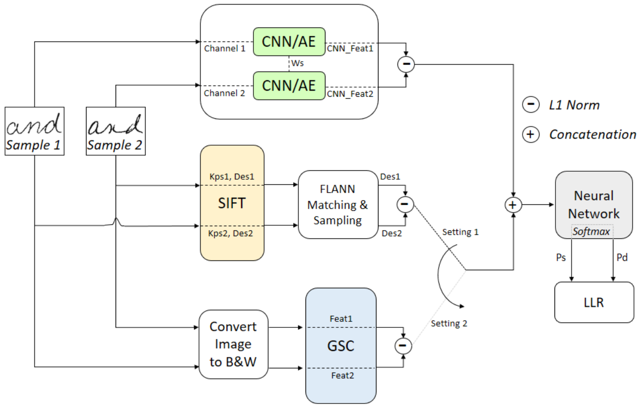


Fig. 4: Hybrid Deep Learning Architecture

IV. Hybrid Deep Learning

In this section, we describe in detail the two hybrid deep learning techniques to solve for handwriting comparison task:

A. *TC-AE/TC-CNN with SIFT*

In the first hybrid setting we append the handcrafted features from SIFT to deeply learned features from TC-CNN/TC-AE as shown in Figure 4. SIFT outputs variable number of keypoints with descriptors for a given input image sample. Each keypoint is a result of a stable maxima produced across all the differences of blurred scales in an octave. The blur is produced by convolving the image with a gaussian kernel.

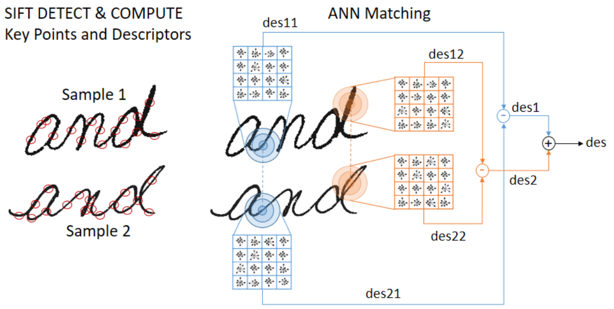


Fig. 5: SIFT Feature Extractor with FLANN Feature Matching

The outliers from the resulting keypoints are removed using low contrast detector and Harris corner detector. Associated with each input sample are variable number of keypoints. For each keypoint SIFT creates scale, rotational and orientational invariant image descriptors. Length of each descriptor is 128. We use nearest neighbour mapping algorithm FLANN [20] to match n keypoints for given pair of input image samples as shown in Figure 5. The matching process is similar to forensic document examination wherein the handwriting samples are matched on the basis of: strokes (connecting, beginning, ending), slantness, flourishments and embellishments. After the matching process, we take the L1 difference between the n matched descriptors. The resulting feature vector is of fixed size n*128 which represents the differences between the features of the two handwriting samples. This difference feature vector of SIFT is appended to the deeply learned features of TC-CNN/TC-AE. Combined features are then fed as an input to a two class neural network classifier. We use the softmax layer of the neural network classifier to provide a degree of similarity (LLR) between the pair of input samples. LLR is computed by taking the logarithm of the ratio of the probability of input pairs being similar to the probability of the pair being dissimilar. x_1 and x_2 represent the features of image sample 1 and 2 respectively while c_0 and c_1 represent similar and dissimlar classes. Hence, the LLR equation is computed as follows:

$$\begin{aligned} LLR &= \frac{P(x_1, x_2|c_0)}{P(x_1, x_2|c_1)} \\ &= \frac{\frac{P(c_0|x_1,x_2)*P(x_1,x_2)}{P(c_0)}}{\frac{P(c_1|x_1,x_2)*P(x_1,x_2)}{P(c_1)}} \\ &= \frac{P(c_0|x_1, x_2) * P(c_1)}{P(c_1|x_1, x_2) * P(c_0)} \end{aligned} \tag{5}$$

Under the fair assumption that the probability of input samples being similar and dissimilar is the same i.e $P(c_0) = P(c_1)$. We can safely conclude that:

$$LLR = \frac{P(c_0|x_1, x_2)}{P(c_1|x_1, x_2)} \tag{6}$$

Hence, for computing LLR we use logarithmic ratio of the resultant softmax probabilities for each class within the neural network classifier.

B. *TC-AE/TC-CNN with GSC*

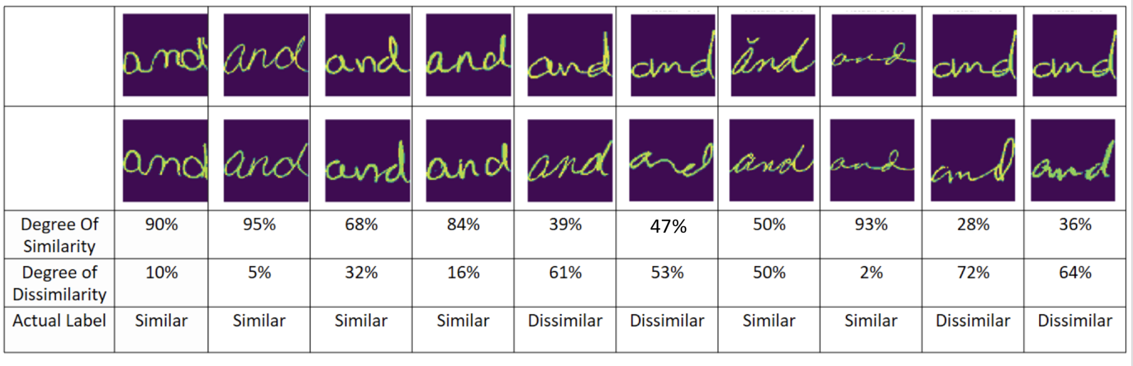
In the second hybrid setting we append the rule based features obtained from GSC to deeply learned features obtained from TC-CNN/TC-AE. Since GSC algorithm requires binarized images as input, we use thresholding technique to binarize the images. First step of GSC computation is to uniformly subsample the input image into 4x4 parts. Each part comprises of three features:

- Gradient Features are computed by first convolving 3x3 sobel filter [21] with each subsample to obtain gradient angle at each pixel with respect to all its adjacent pixels. The gradient angle would vary between 0 to 2π radians in polar coordinate system. The polar coordinate system is now discretized into 12 bins, each of size $2\pi/12$ radians. We now find the frequency of gradient angles occuring in each bin within each subsample. If the frequency of a bin is greater than the set threshold then the value of the binary gradient feature vector for this bin is 1. If the frequency is lower than the threshold than the value for the bin is set to 0. Hence, each subsample would contribute to a binary gradient feature vector of size 12. The entire image would have 12*4x4=192 binary gradient features.
- Structural Features are computed by applying set of 12 rules to each pixel with respect to its adjacent eight neighbours within each subsample. We use standard 12 rules as used by John T. Fatava and Geetha Srikantan [22] for finding structural features which detects horizontal line, vertical line, diagonal rising, diagonal falling and four corners. Each rule is considered a bin. Hence, structural feature vector has 12 bins. Similar to gradient feature bins, thresholding structural feature bins would result in a binary vector. The entire image would have 12*4x4=192 binary structural features.
- Concavity Feature are of size 128 bits composed of three subfeatures:
  - Pixel Density subfeatures are the number of black pixels within each subsample. Thresholding the pixel density would result into a binary value. Entire image would have 1*4x4=16 binary pixel density subfeatures.
  - Large-Stroke features are computed by finding the largest continuous sequence of black pixels along horizontal and vertical directions. Thresholding the sequence would result in a binary value. Entire image would have 2*4x4 large stroke subfeatures.
  - Concavity features capture up/down/left/right pointing concavities by convolving with starlike operator [22]. The algorithm to compute the star operator is beyond the scope of this paper. Rules are associated to detect the concavities. Entire image would have 5*4x4 = 80 concavity subfeatures.

We use L1 difference between the pairs of GSC features of handwritten samples. The difference feature vector is then appended to the deeply learnt features of TC-CNN/TC-AE.

TABLE I: Experimental results of comparison between Deep Learning, Handcrafted and HDL methods. Here notation ”f1 - f2” means the results belong to setup when network was trained on the vector difference of the features obtained, from the two input samples, using the methods in the third row of column header. And “$f1 \bigcup f2$” means the results belong to setup when network was trained on the union of the features obtained, from the two input samples, using the methods in the third row of column header. The first rowc code of column header tells the category of feature extraction.

Method	**Deep Learning**		**Handcrafted**		**Hybrid Deep Learning**				**Baseline**
					Seen dataset Partitioning				
Setups	**CNN**	**AE**	**SIFT**	**GSC**	**CNN_SIFT**	**AE_SIFT**	**CNN_GSC**	**AE_GSC**	**CEDAR-FOX**
f1 - f2	93.47	93.21	71.49	87.94	84.9	86.47	95.35	**96.11**	81.36
f1 $\bigcup f2$	98.78	99.56	76	97.38	78.5	82.39	98.95	**99.78**	
					Shuffled dataset Partitioning				
f1 - f2	84.21	87.14	70.15	89.65	72.41	76.17	89.05	**91.43**	82.53
f1 $\bigcup f2$	86.8	91.23	71.1	89.91	70.21	70.21	90.29	**92.16**	
					Unseen dataset Partitioning				
f1 - f2	53.48	55.96	59.21	52.69	61.9	**64.24**	58.32	58.31	**80.92**
f1 $\bigcup f2$	51.24	56.9	58.55	51.89	62.21	**63.12**	55.89	59.91	



Degree Of Similarity	90%	95%	68%	84%	39%	47%	50%	93%	28%	36%
Degree of Dissimilarity	10%	5%	32%	16%	61%	53%	50%	2%	72%	64%
Actual Label	Similar	Similar	Similar	Similar	Dissimilar	Dissimilar	Similar	Similar	Dissimilar	Dissimilar

Fig. 6: Sample Input and Output using HDL

Similar to SIFT the combined features are then fed as an input to a two class neural network classifier. The classifier outputs a degree of similarity between pair of input handwritten samples as described in Equation 5.

V. Experiment and Results

Our experimental setup comprises of 3 parts: Setting up SIFT extractor; setting up GSC extractor; setting up HDL models. For feature extraction using SIFT/GSC; we pre-extract the matching features for each pair of images in Tr_set and Ts_set of each dataset and store it on file system (FS) in CSV formats with corresponding name. For HDL we train and test our model using four 11GB NVIDIA GTX 1080 Ti GPUs and a TensorFlow backend. We consider CEDAR-FOX (CF) software as our baseline for evaluating our architectures. We input pairwise combination of image files to CFs batch verification tool. CF generates an excel with negative and positive scores of LLR, where the higher the positive value means a higher similarity and lower the negative value indicates a higher dissimilarity. This process is followed for ts_set of each dataset results are compared to the results obtained from our models. We train our model for over 4000 epochs. The results are presented in table1.

The code for GSC feature extraction, SIFT feature extraction using OpenCV and the code for HDL training are available on github at the following link: https://github.com/mshaikh2/HDL_Forensics.git

The accuracy of a model is defined as the ability of system to generate a positive LLR if the ground truth images (GT) are labeled similar i.e 1 and negative if the GT images are labelled dissimilar i.e 0. The results in table 1 represent the percentage accuracy obtained using each architecture with the mathematical setups, f_1 + f_2 and f_1 - f_2. Furthermore there are many measures for finding similarity between features like euclidean, hamming, chi square distance, however, our focus in this paper is to compare the similarity based features with unified features. Hence, we choose to compare simple difference based features with concatenated based features. The results on unseen writer dataset are relatively worse as the writing in the testing set was totally different from what the model saw during training, however results on seen and shuffled dataset seems to outperform and are comparable to the baseline CEDAR-FOX results. Specifically the HDL architecture using AE and GSC under the concatenated setting performs the best in shuffled dataset, where as, the HDL model using AE and SIFT performed best in unseen writer dataset. Overall, the performance of concatenation of features performs better than similarity based features as we can argue the loss of information in the latter, which is clearly evident by our results.

VI. Conclusion

Overall for handwriting comparison task, HDL proves to be a promising architecture and provides decent accuracy even on unseen writer dataset, based on evaluation of results against CEDAR-FOX. However, extracting features from SIFT and GSC is a time consuming task and it is worth noting that there is a scope to learn these features automatically. It is also evident that human engineered feature extractors like SIFT and GSC complement the deeply learned features extracted by a CNN. However, feature extraction using SIFT and GSC add about 70% to the training time, and create features from different sources, which makes it inconvenient to train the model end to end. We open a new realm of research in the field of Handwriting Comparison for bridging the gap between these feature extraction methods as a future scope, which can lead to increase in overall accuracy of the system.

VII. Acknowledgement

We are extremely grateful to CEDAR for providing access to CEDAR-FOX for validating the results obtained by hybrid deep learning models. We would also like to thank Dr. Mingchen Gao for providing GPU compute environment. Finally, we thank our team mates for proof reading our paper.

References

[1] "Individuality of handwriting: A validation study," in *Proceedings of the Sixth International Conference on Document Analysis and Recognition*, ICDAR '01, (Washington, DC, USA), pp. 106–, IEEE Computer Society, 2001.

[2] R. A. Huber and A. Headrick, "Handwriting identification: facts and fundamentals,"

[3] S.N. Srihari, Feature extraction for address block location, From Pixels to Features, J.C. Simon, editor, North Holland, Elsevier, Amsterdam, 1989, 261-274.

[4] Srihari SN. Recognition of handwritten and machine-printed text for postal address interpretation. Pattern Recognition Letters 1993;14: 291303.

[5] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, pp. 91–110, Nov. 2004.

[6] J. Rodrguez-Serrano and F. Perronnin., "Local gradient histogram features for word spotting in unconstrained handwritten documents.," ICFHR, Montreal, 2008.

[7] S. Dey, A. Dutta, J. I. Toledo, S. K. Ghosh, J. Lladós, and U. Pal, "Signet: Convolutional siamese network for writer independent offline signature verification," *CoRR*, vol. abs/1707.02131, 2017.

[8] A. Lebedev, V. Khryashchev, A. Priorov, and O. Stepanova, "Face verification based on convolutional neural network and deep learning," in *2017 IEEE East-West Design Test Symposium (EWDTS)*, pp. 1–5, Sept 2017.

[9] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a "siamese" time delay neural network," in *Proceedings of the 6th International Conference on Neural Information Processing Systems*, NIPS'93, (San Francisco, CA, USA), pp. 737–744, Morgan Kaufmann Publishers Inc., 1993.

[10] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1," ch. Learning Internal Representations by Error Propagation, pp. 318–362, Cambridge, MA, USA: MIT Press, 1986.

[11] K. Yan, Y. Wang, D. Liang, T. Huang, and Y. Tian, "Cnn vs. sift for image retrieval: Alternative or complementary?," in *Proceedings of the 2016 ACM on Multimedia Conference*, MM '16, (New York, NY, USA), pp. 407–411, ACM, 2016.

[12] V. Chandrasekhar, J. Lin, O. Morère, H. Goh, and A. Veillard, "A practical guide to cnns and fisher vectors for image instance retrieval," *CoRR*, vol. abs/1508.02496, 2015.

[13] G. Zhang, Z. Zeng, S. Zhang, Y. Zhang, and W. Wu, "Sift matching with cnn evidences for particular object retrieval," vol. 238, 02 2017.

[14] L. Zheng, Y. Yang, and Q. Tian, "SIFT meets CNN: A decade survey of instance retrieval," *CoRR*, vol. abs/1608.01807, 2016.

[15] C. Huang and S. N. Srihari, "Mapping Transcripts to Handwritten Text," in *Tenth International Workshop on Frontiers in Handwriting Recognition* (G. Lorette, ed.), (La Baule (France)), Université de Rennes 1, Suvisoft, Oct. 2006. http://www.suvisoft.com.

[16] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25* (F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, eds.), pp. 1097–1105, Curran Associates, Inc., 2012.

[18] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015.

[19] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," *CoRR*, vol. abs/1512.00567, 2015.

[20] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration," in *In VISAPP International Conference on Computer Vision Theory and Applications*, pp. 331–340, 2009.

[21] N. Kanopoulos, N. Vasanthavada, and R. L. Baker, "Design of an image edge detection filter using the sobel operator," *IEEE Journal of Solid-State Circuits*, vol. 23, pp. 358–367, Apr 1988.

[22] J. T. Favata and G. Srikantan, "A multiple feature/resolution approach to handprinted digit and character recognition," vol. 7, pp. 304 – 311, 12 1998.