IJCCS (Indonesian Journal of Computing and Cybernetics Systems)
Vol.17, No.2, April 2023, pp. 115~126
ISSN (print): 1978-1520, ISSN (online): 2460-7258
DOI: 10.22146/ijccs.74627

Siamese-Network Based Signature Verification using Self Supervised Learning

Muhammad Fawwaz Mayda*[1], Aina Musdholifah[2]
[1]Undergraduate Program of Computer Science, FMIPA UGM, Yogyakarta, Indonesia
[2]Departement of Computer Science and Electronics, FMIPA UGM, Yogyakarta, Indonesia
e-mail: ***[1]muhammad.fawwaz.mayda@mail.ugm.ac.id**, [2]aina_m@ugm.ac.id

Abstrak

Penggunaan tanda tangan sangat sering kita jumpai dalam berbagai dokumen publik mulai dari dokumen akademik hingga dokumen bisnis yang menjadi tanda bahwa keberadaan tanda tangan sangatlah krusial dalam berbagai proses administrasi . Seringnya penggunaan tanda tangan bukan berarti sebuah prosedur tanpa celah, tetapi kita harus tetap waspada terhadap pemalsuan tanda tangan yang dilakukan dengan berbagai motif dibelakangnya. Oleh karenannya dalam penelitian ini dikembangkan sistem verifikasi tanda tangan yang bisa mencegah terjadinya pemalsuan tanda tangan dalam dokumen publik dengan menggunakan citra digital dari tanda tangan yang ada. Dalam penelitian ini digunakan jaringan syaraf tiruan dengan arsitektur berbasis jaringan kembar yang juga memberdayakan teknik self supervised learning untuk meningkatkan akurasi pada ranah data yang terbatas. Evaluasi akhir terhadap metode pembelajaran mesin yang digunakan mendapatkan akurasi maksimal sebesar 83% dan hasil ini lebih baik daripada model pembelajaran mesin yang tidak melibatkan metode self supervised learning.

Kata kunci*— pembelajaran mesin, siamese network, self supervised learning, tanda tangan*

Abstract

The use of signatures is often encountered in various public documents ranging from academic documents to business documents that are a sign that the existence of signatures is crucial in various administrative processes. The frequent use of signatures does not mean a procedure without loopholes, but we must remain vigilant against signature falsification carried out with various motives behind it. Therefore, in this study, a signature verification system was developed that could prevent the falsification of signatures in public documents by using digital imagery of existing signatures. This study used neural networks with siamese network-based architectures that also empower self-supervised learning techniques to improve accuracy in the realm of limited data. The final evaluation of the machine learning method used gets a maximum accuracy of 83% and this result is better than the machine learning model that does not involve self-supervised learning methods.

Keywords*—machine learning, siamese network, self-supervised learning, signature*

1. INTRODUCTION

In so many kinds of scenarios that confidentiality can be considered a very important aspect of it, biometrics technology deemed very useful to be used. The main purpose of biometrics technology is to provide a means of identifying an individual based on certain merit such as behavioral or physiological attributes. In many cases, this can be achieved by measuring certain aspects such as fingerprint, iris, palm, voice, facial expression, etc [1]. Amongst them, the handwritten signature also can be used to verify certain individuals and is probably still the most commonly used since the ancient times for so many things such as bank checks, forms, insurance,

Received May 13th,2022; Revised April 29th, 2022; Accepted April 30th, 2022

etc [2]. As a part of the Biometrics system, a handwritten signature can be used for two basic things: 1) Identification and 2) Verification. Identification means to be used to identify a particular person from a group of known users in the systems. Verification means to facilitate the authentication of a certain signature in other words it can be used to determine whether a certain signature is genuine or forgery [3].

In the case of handwritten signatures, we can further divide the categories into two major groups: 1) Online Signature Verification and 2) Offline Signature Verification [1], figure 1 for example. Online Signature Verification happens when the writer is writing their signature and involves a special device to record the pen activity such as pressure, speed of writing, angle, coordinate point, etc thus having more complete data available to be used for determining whether the signature is a forgery or genuine than the Offline counterpart. This method comes with certain weaknesses such as the cost of such a specialized system making real-world scenarios become less practical such as not scaled well in case we need to verify thousand of signatures and also require the person in question of their signature to be present physically. Offline Signature Verification system involved only scanned image of a person's signature [4] thus making a real-world application become more practical because in day to day scenarios the signature is already contained in paper, the practicality also extends to when we need to verify thousand of signature and also we can do the verification at any time without the presence of the person in question to giver his/her signature. In this research, the author uses the Offline Signature Verification because of the aforementioned advantages and practicality.

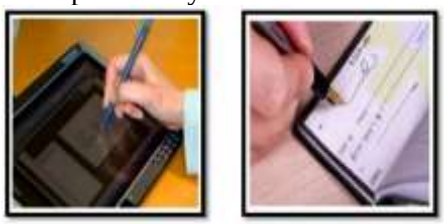


Figure 1 (a) Online Signature (b) Offline Signature

The offline Signature Verification system can again be classified into two categories: 1) Writer Dependent (WD) and 2) Writer Independent (WI) [1]. As the name implies, a Writer Dependent System can only decide whether the signature in question belongs to a specific writer that the system has been specialized to recognize. For example, if the system was able to recognize Mr.A signature, the system will give an output denoting whether the signature in question belongs to Mr.A or not, restricting the system to Mr.A signature only. This kind of system is not scaled well if for example in a banking scenario where thousands of people making transactions or opening a bank account in a certain local bank branch, then the bank will need to have way too many systems to verify each account holder signature and thus render it quite impractical for a real-world scenario. In contrast, the Writer Independent (WI) system is even more general because it can be used to verify whether a pair of signatures is belonging to a pair of genuine or forgery signatures, the system as the name suggests not assume a particular writer thus can be used to arbitrarily any new signature. The second option is more applicable to a real-world scenario in terms of practicality and scalability. This research will use the Offline Writer Independent Signature Verification system because of the aforementioned advantages and practicality.

As the current advancement in machine learning can be applied to almost any scenario to help solve so many problems, recently many successful machine learning applications in computer vision-related problems, in particular, can be traced to utilize Convolutional Neural Network or CNN[5] this is not an exception for signature verification. A handwritten signature is a unique stroke pattern drawn on paper/documents as an identification sign [6], yet very simple and easy to use which makes it very common sense in day-to-day scenarios. The actual problems emerge when someone decides to fabricate a forgery of someone else's signature, which is

considered illegal by many. Many systems developed to combat a forgery depend on the fact someone's signature is unique according to the pattern of strokes he/she performs. In order to create s successful signature verification system at least these two steps is required: 1) Extracting the signature feature, and 2) Measure the difference between the two signatures in question [7].

CNN or Convolutional Neural Network has been a primary tool for tackling machine learning problems related to computer vision [5]. The rise of CNN primarily is due to difficulties in obtaining features (feature extraction) using a hand-crafted feature extraction method, CNN offers a means to automatically learn features from raw data thus making it a very popular choice these days [8]. CNN's success can be attributed to three features that help it learn data representation automatically: 1). Local Receptive Field, 2) Shared Weights, and 3) Sampling techniques [9].

With all of those benefits, CNN does have some drawbacks, one of which is time-consuming process of training CNN to be useful for specific tasks and scenarios. To solve this gap, some researchers tried to train a pioneer CNN model on large datasets, with the goal that their CNN weights would be shared and used for other research objectives after the training has been completed. This model is called pre-trained model [10]. Since the ImageNet dataset contains 14,197,122 images categorized further into nearly 20,000 categories, it is commonly utilized to train those models [10]. A scenario of using those pre-trained model is called transfer learning, and with those principles, a researcher can reuse the pre-trained model to be applied to their specific task without the need to train the model from scratch thus this method not only save time but also improve performance.

The emerging use of transfer learning has been making a few handfuls of pre-trained models to be available for use and adoption for new tasks and scenarios. Namely, the number of a model born out of transfer learning scenarios using ImageNet Dataset has increased year by year some notable models like AlexNet, VGG-Net, GoogLeNet, and ResNet [11]. Those models differ from their respective architecture has been known to be successful in a transfer learning scenario, as such this research will utilize one of them namely ResNet. ResNet developed by [12] has been the winner of the ILSVRC challenge in 2016 by adopting a unique network architecture so-called Residual Block. ResNet has been praised for its good generalization ability despite having a quite deep layer in which before the ResNet era has been considered to have a bad generalization capability, thus the study by [13] also justifies the effectiveness of ResNet.

Metric Learning can be described as the way for machine learning to express similarity and dissimilarity between objects/data points [14]. The goal of metric learning is to learn a representation that will map certain objects/data points into some embedding space in which their "distance" should be close enough for similar samples and far enough for a dissimilar sample [14]. The importance of metric learning in terms of signature verification is undebatable because only by comparing the signature representation we can give some prediction of whether a certain signature is genuine or forgery example. Unlike humans that are capable of distinguishing an object by just looking once, (we can know which one is an apple or orange by just looking at it once) machines didn't have such a privilege to do so, instead, they rely on thousand of example to help it learn and distinguish. Just like any other machine learning application it requires a certain type of loss as a "teacher" to give guidance on how to learn correctly, in metric learning certain type of loss function is desirable to help it solve the problem of distinguishing certain objects/data points. To enable machine learning to learn similarity and dissimilarity between those data points certain types of loss function has been proposed such as 1) Contrastive Loss, and 2) Triplet Loss, these functions provided the means to increase the distance between the dissimilar object and to decrease the distance for similar object [15].

Contrastive Loss [16] has been a well-known loss function for metric learning and is probably the most intuitive. Let x_1, x_2 be the sample data points and y_1, y_2 is their respective labels. Also in condition if $y_1 = y_2$ the $C(y_1, y_2)$ will be 1 and otherwise will be 0 and α is a margin. Contrastive Loss can be formulated as equation (1):

$$\mathcal{L}_{contrastive} = C(y_1, y_2)\, D^2(x_1, x_2) \;+\; C(y_1, y_2) \max\left(0, \alpha - D^2(x_1, x_2)\right) \quad (1)$$

Within the left hand of equation (1), we will see that if the labels of $y_1 = y_2$ then the loss function will push the network to learn that the mapping distance between x1 and x2 should be close, otherwise the loss will increase. And in the right hand of equation (1), we will see that if the labels of $y_1 \neq y_2$ then the loss function will guide the network to learn the mapping to make $\alpha - D^2(x_1, x_2) < 0$ thus making the loss zero if only the mapping distance is far away and the loss will increase if the mapping distance becomes close.

The next improvement to achieve the ability to do metric learning is proposed by [17] via Triplet Loss, as the name suggests is using three samples. Let x_a, x_p, x_n (sometimes called triplets) be some samples and y_a, y_p, y_n is their respective labels in which $y_a = y_p$ and $y_a \neq y_n$, usually x_a will be named anchor, x_p will be named positive because of the same label, and x_n will be named negative because of different label. Triplet loss can be formulated as equation (2):

$$\mathcal{L}_{triplets} = \max\left(0, D^2\left(x_a, x_p\right) - D^2(x_a, x_n) + \alpha\right) \quad (2)$$

Triplet loss provided a better discrimination using both intra-class and inter-class samples simultaneously [15], show in figure 2. This discrimination characteristics comes because at the same time the distance between anchors with positive samples must be close and the distance between anchors with negative samples must be separated away. In short, triplets loss will guide the network to learn to satisfy this inequality equation (3):

$$D^2\left(x_a, x_p\right) + \alpha < D^2(x_a, x_n) \quad (3)$$

There are some cases of random triplets that will directly satisfy equation (3), these situations are deemed not effective for the training process [17], [18]. We want to find triplets that will dissatisfy the condition or $D^2(x_a, x_n) < D^2(x_a, x_p)$ (distance between anchor and negative is smaller than distance between anchor and positive) to train the network, the triplets that directly satisfy the condition is referred as easy triplets and those who don't is referred as hard triplets. Study by [18] suggests that easy triplets won't produce the best embedding result instead hard triplets will produce better embedding result and very essential if we want a successful result of using triplet loss. Another study [19] showed that hard triplets are very useful because essentially hard triplets are an error in embedding representation that needs to be updated to train a neural network in metric learning.

While the loss function for metric learning is already discussed, the architecture for metric learning also plays an important role, namely Siamese Network [20] has been proven to be useful for metric learning. In the basic form of the Siamese Network will receive a pair of images (either positive-positive or positive-negative) then the distance between those will be calculated via a loss function and then weights update will be performed, thus that process makes Siamese Network become a suitable model for maximizing and minimizing the distance between the object. Siamese networks employ shared weights that will positively affect the performance of the neural network when used to gather useful patterns of images in metric learning scenarios. It is possible to combine Siamese Network and CNN to work in tandem which included the capability to learn similarities directly from pixel, color, and texture at the same time [15]. Siamese network shown in figure 3 can use contrastive loss and triplets loss.

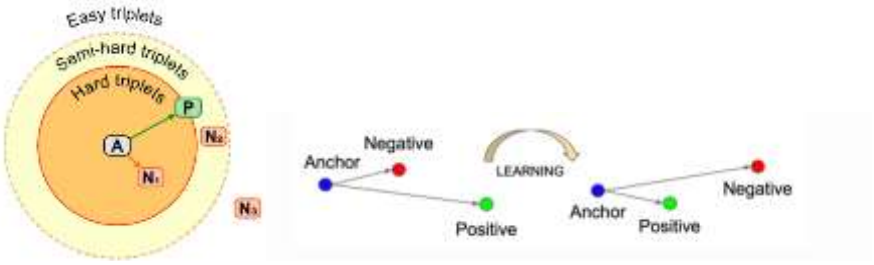


Figure 2 Triplet Loss

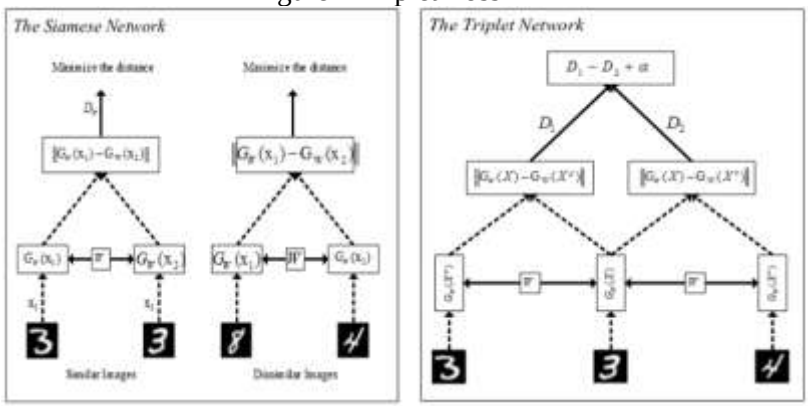


Figure 3 Siamese Network with contrastive and triplets loss

Many of the above techniques are considerably used by many studies to create a signature verification system. A study by [21] proposed a signature verification that used two-phased training: a writer independent feature extractor learning followed by writer dependent classifier system however, this study is different from the initial intention to train writer independent signature verification system. A study by [2] introduces the use of Siamese Network to train signature verification system using contrastive loss as a loss function alongside CNN from scratch as a feature extractor, this study sparks so much interest in using Siamese Network for offline signature verification. Another study by [22] shows some interest in using transfer learning for offline signature verification using some architecture like GoogLeNet and ResNet, while a later study by [23] also introduces the use of active learning for signature verification in the hope of gaining more accuracy. Finally, a study by [24] introduces the usage of triplets loss for online signature verification by combining it with Siamese Network this also differs from the initial intention to train offline signature verification.

Within the growing application of machine learning especially supervised learning which requires the presence of data samples and labels to effectively learn [5] making its way into many practical applications. While the machine learning application is growing the need for data also growing because ultimately data is just like a "fuel" for machine learning system, even if in the era of big data with the abundance of data their need for labels in supervised learning scenarios can limit its practical use and limit the performance. Many recent advancements try to mitigate these problems by introducing techniques like transfer learning, data augmentation, active learning, etc, one such recent example is self-supervised learning techniques which have the capabilities to learn representation from the unlabeled dataset [25]. One key difference from self-supervised learning is the usage of two-phased tasks namely: 1) Pretext task, and 2) Downstream task. The pretext task is the first phase and acts as an important strategy to learn a meaningful representation of using pseudo labels from data, usually, pseudo labels can be generated automatically. The result of the Pretext task can be used in the various downstream task such as classification, segmentation, detection, etc where the availability of labeled data is limited and with the expectation of improving the performance after learning a useful data representation [26] and also performance evaluation of pretext task often neglected [27].

To give a simple analogy of how self-supervised learning will work consider this example of playing badminton. Usually, people before playing badminton will do some warm-ups like running around the field before. We can think of the pretext task like a warm-up activity that aims to prepare physically before playing badminton and therefore the main goal is to play badminton then how many times we manage to run around the field is not so important, the important thing is that we are more physically prepared to play badminton compared to us directly playing badminton without warming up. This badminton is an example of the downstream task that we will do even so the activity of running around the field itself (pretext task) is not only useful for playing badminton and even playing football will be better if we warm up first.

Similarly, what happens in self-supervised learning with the existence of a pretext task will provide an opportunity for machine learning models or CNN models to be able to learn data representations first so that they can be used better in the downstream task that will be done. Because it has successfully learned data representation, it is likely that when doing the downstream task will not need too much data because the model has learned good data representation. The pretext task will utilize unlabeled data while the downstream task will utilize labeled data.

SimCLR is a self-supervised learning framework developed by [28], pseudocode availabe in figure 4. This framework uses contrastive learning as the pretext task to learn a meaningful representation of data, and utilizes some data augmentation techniques to create more examples for learning. The augmentation will be applied twice to get a different point of view from the same image and then will be used to train the feature extractor to learn in a contrastive learning fashion. After the step is completed the feature extractor/feature encoder component can be used for another downstream task.

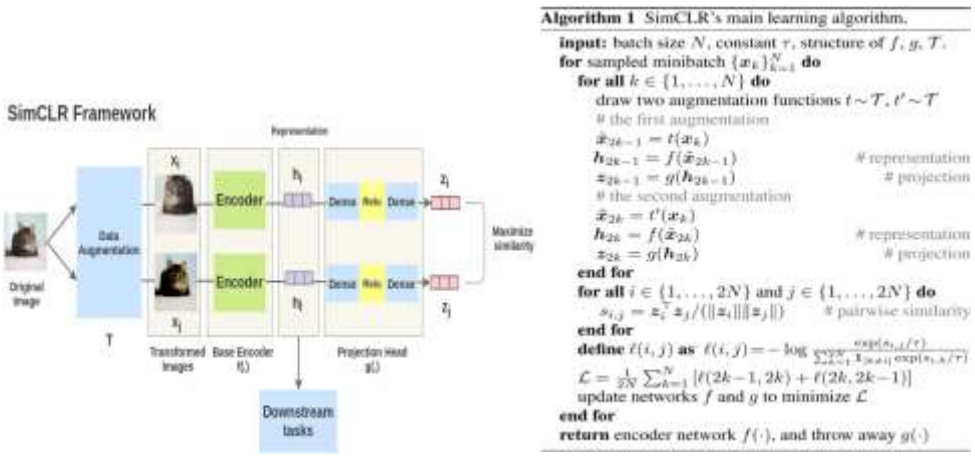


Figure 4 Ilustrasted overview of SimCLR

2. METHODS

This study will propose a writer independent offline signature verification that consists of the above method. Using a self-supervised learning scenario thus there will be pretext tasks and downstream tasks. The pretext task will be based on a framework created by [28] named SimCLR (will be discussed after) and the downstream task will use the Siamese network in tandem with triplets loss. T The last stage incorporates verifying if the signature is a genuine signature pair or a genuine-forgery signature pair using learnt representation. Those steps will assume the use of ResNet 50 as the feature. For all stages, apply Adam optimizer with learning rate 0.01 for 50 epoch with early halting (5 patience). General overview is shown in figure 5.

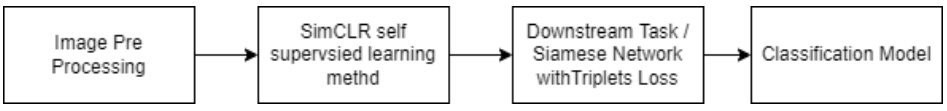


Figure 5 General Overview

2.1 Dataset

This study will leverage the available online dataset from CEDAR [29] which contains signature images from 55 writers consisting of 24 genuine signatures and 24 forgery signatures in a grayscale fashion. Figure 6 shows some sample.

(a) (b)

Figure 6 Signature samples a) genuine signature (b) forgery signature

2.2 Image Pre Processing

Signature images will go through these preprocessing steps: 1) Thresholding with Otsu, 2) Mean Filtering, 3) Thinning and, 4) Resizing. Threshold will provide a clear separation between the signature and the background of paper then mean filtering will be performed to clean up any remaining noise. Thinning will make the signature strokes to be in uniform / equal width (in certain cases signature line width will different because of different pen sizes) and the last step is to resize the images to a certain size which is 224 x 224 (size of ResNet input). At the end of the step, the images will be stored in a specific folder to save time so that subsequent step of training will only need to read images from the specific folder itself.

2.3 Feature Extractor

Matched with the previously mentioned fact that the signature verification system will need a feature extractor component for capturing the representation of signature images, the author will use ResNet 50 as the feature extractor/feature encoder. The reason for choosing ResNet is primarily because of suggestion from [13] and also because the next step of training is leveraging self-supervised learning using SimCLR method, [28] suggests that SimCLR may benefit from a deep network as such ResNet with their respective variants, thus making it become a prevalent choice.

2.4 Self Supervised Learning

SimCLR will use as the self-supervised learning framework and after this step finish, there will be ready to use feature encoder to be used in the downstream task (triplets training). Also, the author will try to compare the performance when this step is performed and when isn't.

2.5 Triplets Training

In this step, the author will utilize Siamese Network paired with Triplets Loss to learn the embedding to discriminate signature images. Triplets to form anchor, positive, and negative will be selected from the genuine signatures of certain writers for both anchor and positive samples and the forgery of certain writers for negative. While in this stage the author will also compare the different results of applying hard mining (using hard triplets only) and easy mining (using all available triplets) strategies.

2.6 Classification Training

The classification model is trained from the learnt representation of the preceding triplets training procedure as the final step in the training process. Simply put, the model from this stage will generate a binary classification of 0 for genuine-forgery signature pairs and 1 for genuine signature pairs. The assessment will be based on cross-validation, using 5 fold validation (80% for training and 20% for testing). Because the goal was to create independent authors, the split will be based on the number of writers; 44 will be utilized for training and the remaining 11 for testing; the model will never see the testing writer beforehand. Figure 7 depicts the many comparisons that will be done.

Figure 7 General Comparison

3. RESULTS AND DISCUSSION

Conclusion of the result supported the main hypothesis that self-supervised learning will improve the overall accuracy when used. Table 1 shows more detailed of the result achieved.

Table 1 Comparison Between Models

No	Name	Fold	TP	TN	FP	FN	Accuracy	FAR	FRR
1	Pre train + hardmining	1	446	320	208	60	0,741	0,394	0,119
2	Pre train + hardmining	2	448	392	136	58	0,812	0,258	0,115
3	Pre train + hardmining	3	442	318	210	64	0,735	0,398	0,126
4	Pre train + hardmining	4	314	404	124	192	0,694	0,235	0,379
5	Pre train + hardmining	5	442	418	110	64	0,832	0,208	0,126
	AVERAGE						**0,763**	**0,298**	**0,173**
6	Pre train + easymining	1	416	372	156	90	0,762	0,295	0,178
7	Pre train + easymining	2	448	402	126	58	0,822	0,239	0,115
8	Pre train + easymining	3	366	368	160	140	0,710	0,303	0,277
9	Pre train + easymining	4	194	414	114	312	0,588	0,216	0,617
10	Pre train + easymining	5	292	480	48	214	0,747	0,091	0,423
	AVERAGE						**0,726**	**0,229**	**0,322**
11	Hardmining	1	422	256	272	84	0,656	0,515	0,166
12	Hardmining	2	360	416	112	146	0,750	0,212	0,289
13	Hardmining	3	390	340	188	116	0,706	0,356	0,229

14	Hardmining	4	302	302	226	204	0,584	0,428	0,403
15	Hardmining	5	346	446	82	160	0,766	0,155	0,316
	AVERAGE						**0,692**	**0,333**	**0,281**
16	Easymining	1	302	362	166	204	0,642	0,314	0,403
17	Easymining	2	366	432	96	140	0,772	0,182	0,277
18	Easymining	3	0	528	0	506	0,511	0,000	1,000
19	Easymining	4	184	390	138	322	0,555	0,261	0,636
20	Easymining	5	458	406	122	48	0,836	0,231	0,095
	AVERAGE						**0,663**	**0,198**	**0,482**

Here is a visualization using tSNE [30] from a model that involves self-supervised pre-training stages and hard mining triplets on the fifth fold (the latter). Here we can see that the original and fake signatures are quite separated both where there is a considerable distance between the two. If we look more deeply at writer number 50 his original and fake signatures the distance is not too far. In the interest of visualization tSNE also the author does not include all signature samples because it will make the analysis less good when visualized.

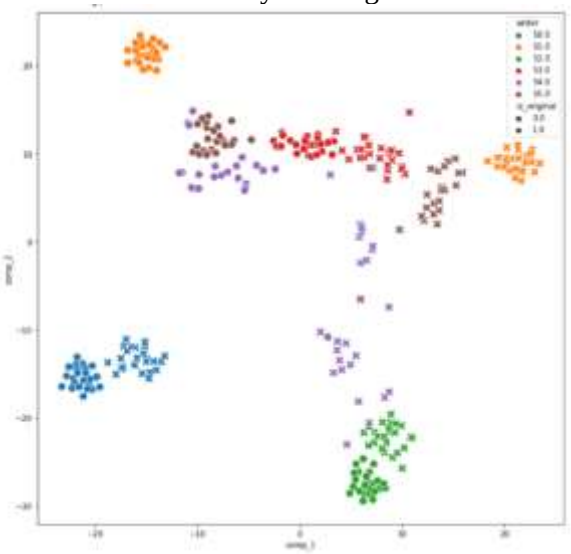
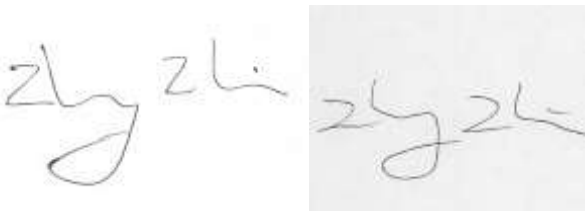


Figure 8 tSNE Visualization

After a visual inspection, the author found that the signature pattern of the 50th writer is quite easy to fake so it is possible if the two are more or less in close proximity. Both look similar and according to the author will be quite easy to fake. Here's an example:



(a) (b)

Figure 9 Example of writer no 50 signatures

Here the author concludes that the distance separator that exists between the cluster from the original signature and the fake signature will be the determinant of success in classifying existing signatures. This ability is certainly obtained by going through the main triplet loss training stage, but the addition of the self-supervised pre-training stage makes the process of triplet loss training better because there is already learning of representations that can be used in distinguishing real and fake signatures.

Also, can be concluded that the use of triplets loss will greatly benefit from good triplets selection as per experiment show hard triplets (obtaining using hard mining) perform better than their counterpart which use easy mining in both scenarios with and without self-supervised pre-training. Within the use of the self-supervised pre-training method, we can see improvement in the model's ability to capture meaningful representation, in experiment shows that models with self-supervised + easy mining perform better than the model with hard mining only. Thus this finding aligns with a study by [31] which stated that self-supervised methods can provide good results in the realm of cases that have limited data because the self-supervised pre-training stage provides good regularization to prevent overfitting of the model for the next downstream task stage. So that it becomes a good alternative solution in the case of signatures because the signature data is considered privacy so the availability is scarce.

4. CONCLUSION

Combining triplets loss with good triplets selection algorithms, such as only training the network from hard triplets (whose violated the triplet inequality), can be used in signature verification. To compensate for the lack of signature imagery, we may choose self-supervised learning to improve accuracy of the model and generalization capability.

REFERENCES

[1] N. Sharma, S. Gupta, and P. Mehta, "A Comprehensive Study on Offline Signature Verification," in *Journal of Physics: Conference Series*, 2021, vol. 1969, no. 1, p. 012044, doi: 10.1088/1742-6596/1969/1/012044.

[2] S. Dey, A. Dutta, J. I. Toledo, S. K. Ghosh, J. Llados, and U. Pal, "SigNet: Convolutional Siamese Network for Writer Independent Offline Signature Verification," Jul. 2017, Accessed: Aug. 29, 2021. [Online]. Available: www.elsevier.com.

[3] N. H. Al-banhawy, H. Mohsen, and N. Ghali, "SIGNATURE IDENTIFICATION AND VERIFICATION SYSTEMS: A COMPARATIVE STUDY ON THE ONLINE AND OFFLINE TECHNIQUES," *Futur. Comput. Informatics J.*, vol. 5, no. 1, pp. 28–45, 2020, doi: 10.54623/fue.fcij.5.1.3.

[4] S. Jerome Gideon, A. Kandulna, A. A. Kujur, A. Diana, and K. Raimond, "Handwritten signature forgery detection using convolutional neural networks," *Procedia Comput. Sci.*, vol. 143, pp. 978–987, 2018, doi: 10.1016/j.procs.2018.10.336.

[5] A. Geron, *Hands–On Machine Learning with Scikit–Learn and TensorFlow 2nd edition*. 2019.

[6] J. Poddar, V. Parikh, and S. K. Bharti, "Offline Signature Recognition and Forgery Detection using Deep Learning," *Procedia Comput. Sci.*, vol. 170, no. 2019, pp. 610–617, 2020, doi: 10.1016/j.procs.2020.03.133.

[7] S. N. Srihari, H. Srinivasan, S. Chen, and M. J. Beal, "Machine Learning for Signature Verification," in *Studies in Computational Intelligence*, vol. 90, 2008, pp. 387–408.

[8] E. Parcham, M. Ilbeygi, and M. Amini, "CBCapsNet: A novel writer-independent offline signature verification model using a CNN-based architecture and capsule neural

networks," *Expert Syst. Appl.*, vol. 185, no. February, p. 115649, 2021, doi: 10.1016/j.eswa.2021.115649.

[9] R. Takahashi, T. Matsubara, and K. Uehara, "A novel weight-shared multi-stage CNN for scale robustness," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 4, pp. 1090–1101, 2019, doi: 10.1109/TCSVT.2018.2822773.

[10] A. Foroozandeh, A. Askari Hemmat, and H. Rabbani, "Offline Handwritten Signature Verification and Recognition Based on Deep Transfer Learning," *Iran. Conf. Mach. Vis. Image Process. MVIP*, vol. 2020-Janua, 2020, doi: 10.1109/MVIP49855.2020.9187481.

[11] M. M. Hameed, R. Ahmad, M. L. M. Kiah, and G. Murtaza, "Machine learning-based offline signature verification systems: A systematic review," *Signal Process. Image Commun.*, vol. 93, no. October 2020, p. 116139, 2021, doi: 10.1016/j.image.2021.116139.

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016, vol. 2016-Decem, pp. 770–778, doi: 10.1109/CVPR.2016.90.

[13] K. Huang, Y. Wang, M. Tao, and T. Zhao, "Why do deep residual networks generalize better than deep feedforward networks? — A neural tangent kernel perspective," *Adv. Neural Inf. Process. Syst.*, vol. 2020-Decem, 2020.

[14] Y. Sun, K. Fu, Z. Wang, C. Zhang, and J. Ye, "Road network metric learning for estimated time of arrival," *Proc. - Int. Conf. Pattern Recognit.*, pp. 1820–1827, Jun. 2020, doi: 10.1109/ICPR48806.2021.9412145.

[15] Mahmut Kaya and Hasan Sakir Bilge, "Deep Metric Learning : A Survey," *Symmetry (Basel).*, vol. 11.9, p. 1066, 2019.

[16] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*, 2005, vol. I, pp. 539–546, doi: 10.1109/CVPR.2005.202.

[17] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015, vol. 07-12-June, pp. 815–823, doi: 10.1109/CVPR.2015.7298682.

[18] A. Hermans, L. Beyer, and B. Leibe, "In Defense of the Triplet Loss for Person Re-Identification," 2017, [Online]. Available: http://arxiv.org/abs/1703.07737.

[19] H. Xuan, A. Stylianou, X. Liu, and R. Pless, "Hard Negative Examples are Hard, but Useful," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 12359 LNCS, pp. 126–142, 2020, doi: 10.1007/978-3-030-58568-6_8.

[20] J. Bromley *et al.*, "Signature Verification using a 'Siamese' Time Delay Neural Network," 1994.

[21] L. G. Hafemann, R. Sabourin, and L. S. Oliveira, "Learning features for offline handwritten signature verification using deep convolutional neural networks," *Pattern Recognit.*, vol. 70, pp. 163–176, 2017, doi: 10.1016/j.patcog.2017.05.012.

[22] Jahandad, S. M. Sam, K. Kamardin, N. N. Amir Sjarif, and N. Mohamed, "Offline signature verification using deep learning convolutional Neural network (CNN) architectures GoogLeNet inception-v1 and inception-v3," *Procedia Comput. Sci.*, vol. 161, pp. 475–483, 2019, doi: 10.1016/j.procs.2019.11.147.

[23] T. Younesian, S. Masoudnia, R. Hosseini, and B. N. Araabi, "Active Transfer Learning for Persian Offline Signature Verification," in *4th International Conference on Pattern Recognition and Image Analysis, IPRIA 2019*, 2019, vol. 2018-Janua, pp. 234–239, doi: 10.1109/PRIA.2019.8786013.

[24] M. Kurowski, A. Sroczyński, G. Bogdanis, and A. Czyżewski, "An automated method for biometric handwritten signature authentication employing neural networks," *Electron.*, vol. 10, no. 4, pp. 1–19, 2021, doi: 10.3390/electronics10040456.

[25] A. Jaiswal, A. R. Babu, M. Z. Zadeh, D. Banerjee, and F. Makedon, "A Survey on Contrastive Self-Supervised Learning," *Technologies*, vol. 9, no. 1, p. 2, Dec. 2020, doi: 10.3390/technologies9010002.

[26] L. Jing and Y. Tian, "Self-Supervised Visual Feature Learning with Deep Neural Networks: A Survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 11, pp. 4037–4058, 2021, doi: 10.1109/TPAMI.2020.2992393.

[27] L. Weng, "Self-Supervised Representation Learning," 2019. https://lilianweng.github.io/lil-log/2019/11/10/self-supervised-learning.html (accessed Nov. 11, 2021).

[28] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *37th International Conference on Machine Learning, ICML 2020*, 2020, vol. PartF16814, pp. 1575–1585, Accessed: Sep. 10, 2021. [Online]. Available: https://github.com/google-research/simclr.

[29] M. K. Kalera, S. Srihari, and A. Xu, "Offline signature verification and identification using distance statistics," in *International Journal of Pattern Recognition and Artificial Intelligence*, 2004, vol. 18, no. 7, pp. 1339–1360, doi: 10.1142/S0218001404003630.

[30] L. Van Der Maaten and G. Hinton, "Visualizing Data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, 2008.

[31] A. Newell and J. Deng, "How useful is self-supervised pretraining for visual tasks?," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2020, pp. 7343–7352, doi: 10.1109/CVPR42600.2020.00737.