Analyzing the Similarity Learned by a Siamese Network with Contrastive Loss on the MNIST Dataset

Antonio A. Sánchez-Ruiz[1]

[1] Department of Software Engineering and Artificial Intelligence, Instituto de Tecnologías del Conocimiento, Universidad Complutense de Madrid, Spain

Abstract

The concept of similarity plays a fundamental role in many artificial intelligence techniques. The ability to automatically learn when two instances are similar or different is especially valuable when working with unstructured datasets such as images, audio, or text. Siamese neural networks enable the automatic learning of a distance function between pairs of instances by projecting them into a latent embedding space. In this work, we analyze the the embeddings generated and the distance learned by a Siamese network trained using contrastive loss to solve various tasks using the MNIST dataset. We also identify some potential problems and propose some ideas to improve the distance learned by the network that will be researched in future works.

Keywords

Similarity, Siamese Networks, Contrastive Loss, MNIST, Embeddings, Nearest Neighbors, Clusters

1. Introduction

The concept of similarity constitutes a fundamental pillar in the development of intelligent systems [1]. Many learning techniques, both supervised and unsupervised, rely on the ability to estimate how similar two elements are, whether to classify, cluster, recommend, or detect unusual patterns. In particular, in Case-based Reasoning (CBR) systems, similarity plays a key role both in retrieving past experiences relevant to the problem at hand and in adapting those experiences to the current context [2].

In tabular datasets where information is represented symbolically, as is common in many classical machine learning applications, similarity between instances is typically computed by aggregating the individual similarities between variables. These variable-level similarities are, in turn, computed according to their nature: categorical or continuous, their scale of representation, or the variable's role in the specific context.

Computing similarity between instances becomes significantly more challenging when dealing with unstructured data, such as images, audio, or text. In such cases, the semantic properties are implicitly represented in the data, making it difficult to establish a connection between the similarity of individual features (e.g., the color of a pixel in an image) and the similarity of instances (e.g., when two images represent the same concept).

It is common to address this using neural networks that, through complex nonlinear transformations, learn to project the instances into a lower-dimensional latent vector space, generating what are known as *embeddings* [3]. These embeddings can be obtained through different approaches. For instance, autoencoder architectures adopt an unsupervised learning paradigm where the goal is to generate embeddings that capture patterns sufficient to reconstruct the original instance. In contrast, classifiers aim to produce embeddings that encode discriminative features relevant for assigning the correct label to each instance.

Siamese Neural Networks (SNNs) [4] consist of a pair of identical networks that share the same weights and biases, followed by a distance measurement layer. These twin networks process two inputs in parallel and generate embeddings that are then compared by computing a distance between them. When trained with a contrastive loss function, SNNs aim to produce embeddings in which

ICCBR BEAR'25: Workshop on Beyond Attribute-Value Case Representation at ICCBR2025, June 30 – July 3, 2025, Biarritz, France

antsanch@ucm.es (A. A. Sánchez-Ruiz)
0000-0003-0034-1254 (A. A. Sánchez-Ruiz)

instances of the same class are grouped closely together while being separated from those of other classes. Unlike traditional classification approaches, SNNs can generalize to unseen classes during training, making them particularly suitable for tasks such as few-shot learning, identity verification, and duplicate detection.

In this work, we explore the use of Convolutional Siamese Networks to automatically learn a distance function between images in the context of a classification problem. We then analyze the intra-class and inter-class similarity, identify prototypical examples for each class, and classify new images efficiently based on those prototypes. We also investigate to what extend the similarity computed by the network aligns with our intuition of similarity between images.

The rest of the paper is organized as follows. Section 2 reviews the most relevant works in which this type of network has been used in case-based reasoning systems. Section 3 presents the dataset used in the experiments. Section 4 describes the proposed network architecture and the training process. Section 5 analyzes the clusters created by the network in the latent embedding space and examines the relationship between the computed distance and the misclassified images. Finally, Section 7 summarizes the main conclusions of the work and outlines directions for future research.

2. Related work

The literature on Siamese Neural Networks is extensive; in this section, we focus specifically on recent works that apply SNNs in the context of Case-Based Reasoning systems. These works include both applications in specific domains and proposals for new architectural approaches.

One of the earliest uses of Siamese networks in a CBR system can be found in [5], which addresses the task of recognizing physical activities from time series data generated by accelerometers worn by participants. The results show that the performance of the SNN is comparable to that of a standard convolutional neural network.

In the domain of textual CBR systems, [6] explores the combination of SNNs and autoencoders with word embeddings, showing how this approach can enhance textual CBR systems by establishing stronger relationships across cases and measuring similarities with minimal input from domain experts.

In the context of Process-Oriented Case-Based Reasoning, where cases are typically represented using semantic graphs to model workflows, [7] proposes the use of Siamese Graph Neural Networks to approximate the similarity between semantic graphs.

For the domain of fault detection and prediction in industrial environments, where numerous sensors and actuators are involved, [8] proposes an SNN architecture that combines 2D convolutions with graph convolutions to extract both temporal and spatial features. They also demonstrate that incorporating expert knowledge can significantly reduce the number of learnable parameters required.

In [9], a more theoretical study is presented, where the authors propose a framework to classify different similarity learning approaches based on whether the feature extraction and similarity computation are modeled or learned. They show that using a classifier as the basis for a similarity measure can achieve results comparable to state-of-the-art methods, and further improvements are possible by learning the similarity function between embeddings.

In [10], the authors present a novel approach in which they train different class-to-class Siamese networks to learn the patterns of both similarity and difference between pairs of classes. They then use these patterns for classification, explanation, and prototypical case identification. Although the results are promising, the proposed approach requires training a quadratic number of networks with respect to the number of classes.

Finally, [11] introduces a novel architecture that incorporates self-attention mechanisms into the traditional Siamese network structure. By weighting the features of the embeddings before computing the distance, the proposed method improves classification accuracy across several datasets from the UCI repository.

As we can observe, Siamese networks have been successfully used in various CBR systems to retrieve similar cases or propose prototypical cases across different domains. However, we believe that a deeper investigation is needed into the type of distance these networks learn and its characteristics, in order to better understand how they can be effectively used in this context. In particular, in this work we analyze the clusters formed by the cases, how prototypical cases can be used to efficiently retrieve similar ones, and how the distance computed by the network aligns with our intuitive notion of similarity between cases.

3. MNIST dataset

The MNIST dataset [12] is a very popular benchmark used in machine learning to evaluate image classification algorithms. It consists of 70,000 grayscale images of handwritten digits ranging from 0 to 9, each normalized and centered in a 28x28 pixel grid. The dataset is divided into 60,000 training samples and 10,000 test samples and each image is labeled with the corresponding digit class. Figure 2 shows some example images.

Despite its apparent simplicity, achieving near-perfect accuracy on MNIST was once considered a significant milestone, and the dataset continues to be a valuable resource for educational and benchmarking purposes. The images of the digits written by different persons, introduces considerable intra-class variability in terms of handwriting style, stroke thickness, orientation, and alignment. This variability makes the dataset particularly useful for evaluating a model's generalization capacity and robustness to variations in input appearance.

Another reason for choosing this dataset is that it is relatively easy for a human to judge whether the handwriting of two digits looks more or less similar, and we aim to compare this notion of similarity to the distance learned by a Siamese neural network. Finally, although the dataset contains many examples, they are small enough to allow for rapid experimentation.

4. The Convolutional Siamese Network

A typical Siamese neural network [4] consists of *two or more identical subnetworks* that share the same weights and map the input instances to fixed-length vectors (embeddings) in a latent space. Then, the last part of the network compares those vectors using a distance function, such as the Euclidean distance or the cosine similarity. The normalized distance between the instances is the output of the network, that can also be fed into additional layers for classification or regression tasks.

Figure 1 shows the network architecture used in this work, which is inspired by the network proposed in [13]. The network receives two input images of size 28×28 pixels with a single color channel, denoted as x_1 and x_2, and processes them through a shared encoder $f(\cdot)$. This encoder module produces two embeddings, $f(x_1)$ and $f(x_2)$, each represented as a 10-dimensional vector. The network then computes the Euclidean distance between the vectors and normalizes it using a sigmoid function:

$$D(f(x_1), f(x_2)) = \|f(x_1) - f(x_2)\|_2$$

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

$$\mathcal{N}_\theta(x_1, x_2) = \sigma(f(x_1), f(x_2))$$

The network is trained on pairs of instances labeled as *similar* (same digit) or *dissimilar* (different digits), with the objective of minimizing the embedding distance for similar pairs and maximizing it (up to a margin) for dissimilar ones. We use the *contrastive loss* [14], defined as:

$$L = (1 - y)D^2 + y \max(0, m - D)^2$$

where y indicates similarity (0 for similar, 1 for dissimilar), D is the distance between embeddings, and m is a margin that enforces a minimum separation between dissimilar pairs ($m = 1$).

This way, the Siamese network learns a mapping from inputs to an *embedding space*, where similar instances are close together and dissimilar ones are far apart according to a chosen distance metric.

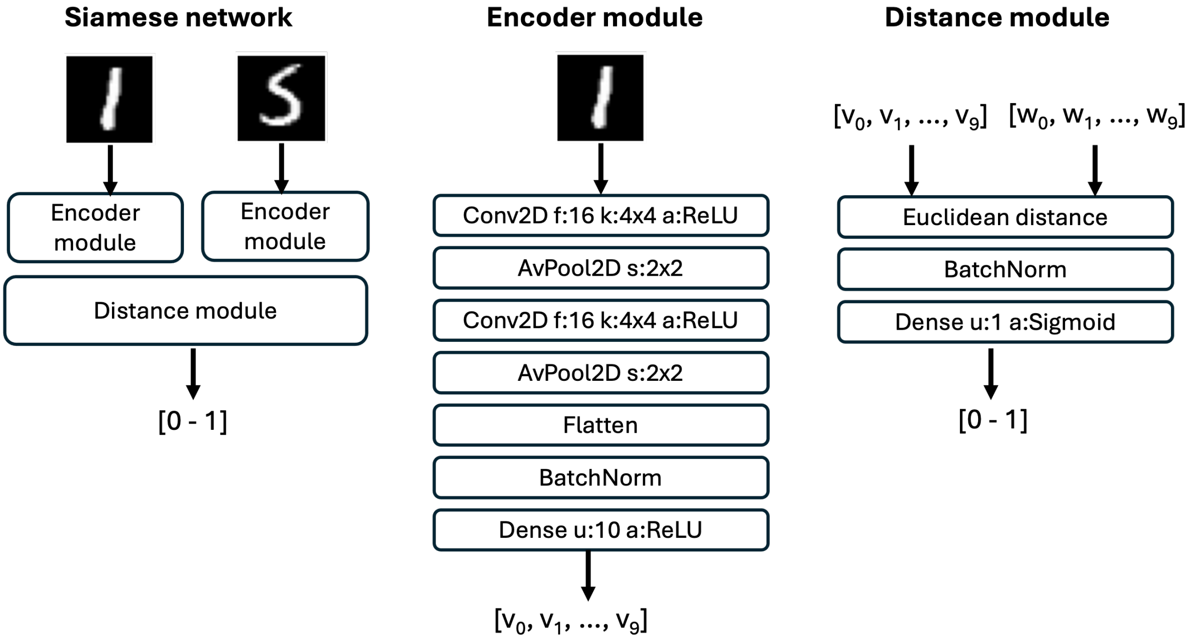

Figure 1: Architecture of the Convolutional Siamese Network and each of its subnetworks.

Class	Train		Test	
	Mean	Std	Mean	Std
0	0.02	0.11	0.02	0.08
1	0.02	0.11	0.02	0.10
2	0.07	0.19	0.06	0.18
3	0.14	0.28	0.12	0.25
4	0.05	0.15	0.05	0.15
5	0.06	0.16	0.06	0.18
6	0.02	0.10	0.04	0.15
7	0.04	0.14	0.05	0.15
8	0.07	0.17	0.07	0.17
9	0.06	0.17	0.06	0.19

Table 1
Mean distance of each train image to similarity scores for each class (alternative configuration).

The network was created using *Keras* over *JAX* and, although the architecture has several layers, most of them are convolutions, so the resulting network is very small and only has 7,470 trainable parameters. It was trained on 200,000 random pairs of images using the *RMSprop* optimizer for 5 epochs and 10,000 random pairs of images for validation. The complete training process took less than 3 minutes on a nvidia RTX 4070 with 12 GB. We obtained an accuracy score of 0.9876 on the test dataset, which means it can determine whether two images represent the same digit 98.76% of the time.

5. Clusters and similarity

Once the network has been trained, we can study the clustering of images by category performed by the network's encoder module. For example, we can compute the embedding corresponding to a prototypical image of each class by calculating the average embedding of the images belonging to each class in the training set.

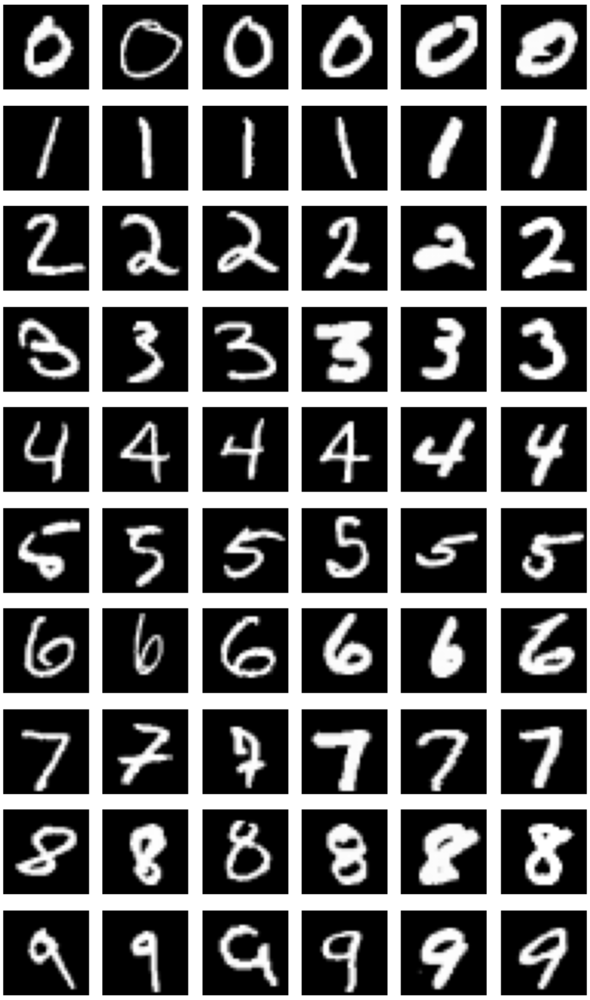

Figure 2: Sample images from the MNIST dataset. The first column shows the train image closest to the centroid of each class (a prototype) and the following columns show the train images closest to the quintiles when they are ordered according to their distance to the centroid.

$$p_i = \frac{1}{n_i} \sum_{j=1}^{n_i} f(x_j)$$

where n_i is the number of images belonging to class i.

It is interesting to note that these prototypical embeddings do not correspond to any image from the training set, and we cannot reconstruct their associated images because the network does not have a *decoder* module. However, we can compute the closest images to each of them, which are shown in the first column of Figure 2. We can observe that, they correspond to clearly distinguishable digits, although they do not always represent the most standard handwriting of each digit.

Although we cannot exactly visualize the images associated with these centroids, we can use them to compute distances using the network's distance module. In this way, Figure 3 shows the distances between the centroids of each cluster. The minimum distance between two centroids from different classes is 0.998, which indicates that the network has successfully separated the different classes. However, the distance between centroids does not seem to be related to the notion of similarity between digits. For example, intuitively, we would say that the handwriting of a 1 is closer to a 7 than to a 5, but this is not reflected in the learned distance.

On the other hand, Table 1 shows the average distances of each image to the centroid of its class. We can observe that the digit 3 exhibits the highest variability, and that the values in both the training and

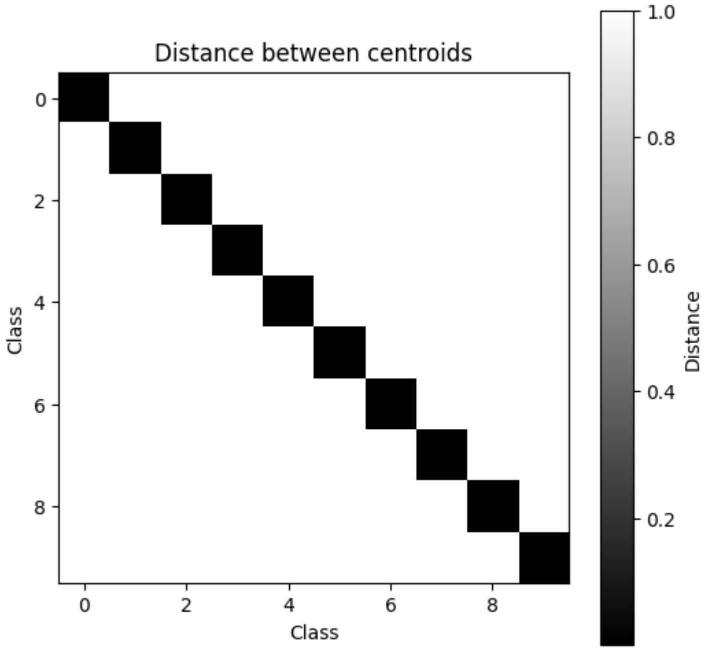

Figure 3: Distance matrix between the centroids of each class.

test sets are similar. Based on this, we conclude that the clusters are fairly compact and well separated from each other. We can confirm our intuition in Figure 4, which shows a projection of the embeddings onto the plane using the TSNE algorithm.

Finally, Figure 2 also shows different training images from each cluster, ordered by their distance to the centroid. The first column displays the image closest to the centroid (the most prototypical image of the class), and the following five columns show the images in each of the quintiles. In general, the distance to the centroid does not appear to have a clear relationship with the notion of difference between handwriting variations within the same digit.

We can conclude that the internal representation of the images learned by the network effectively clusters the digits. However, the resulting image-to-image distance does not appear to support an intuitive ordering based on the visual similarity of the handwritten digit shapes—neither within the same class nor across different classes.

6. Classification based on the nearest centroid

One of the common uses of Siamese networks is to retrieve the nearest neighbors to classify new instances. This approach has a computational cost that is linear with respect to the number of instances in the training set. However, since the clusters generated by the network are compact and well separated, we can also compute distances to their centroids, thus reducing the classification complexity to be linear with respect to the number of classes (which can be considered constant in practice).

Table 2 shows the results of this approach broken down by class. The classifier's accuracy is 98.20%, which is quite good considering that we are using a very small neural network and only computing distances to 10 points each time. Precision and recall scores are very high for all digits, although classifying fives, sevens, and nines appears to be slightly more challenging.

Figure 5 shows the distribution of distances from each test image to its nearest centroid. We observe that nearly all correct classifications occur when the distance to the centroid is below 0.1. In contrast, incorrect classifications are more uniformly distributed across the full range of distances, which is a somewhat surprising finding.

We can delve a bit deeper by visualizing some of the misclassified images. Figure 6 shows the

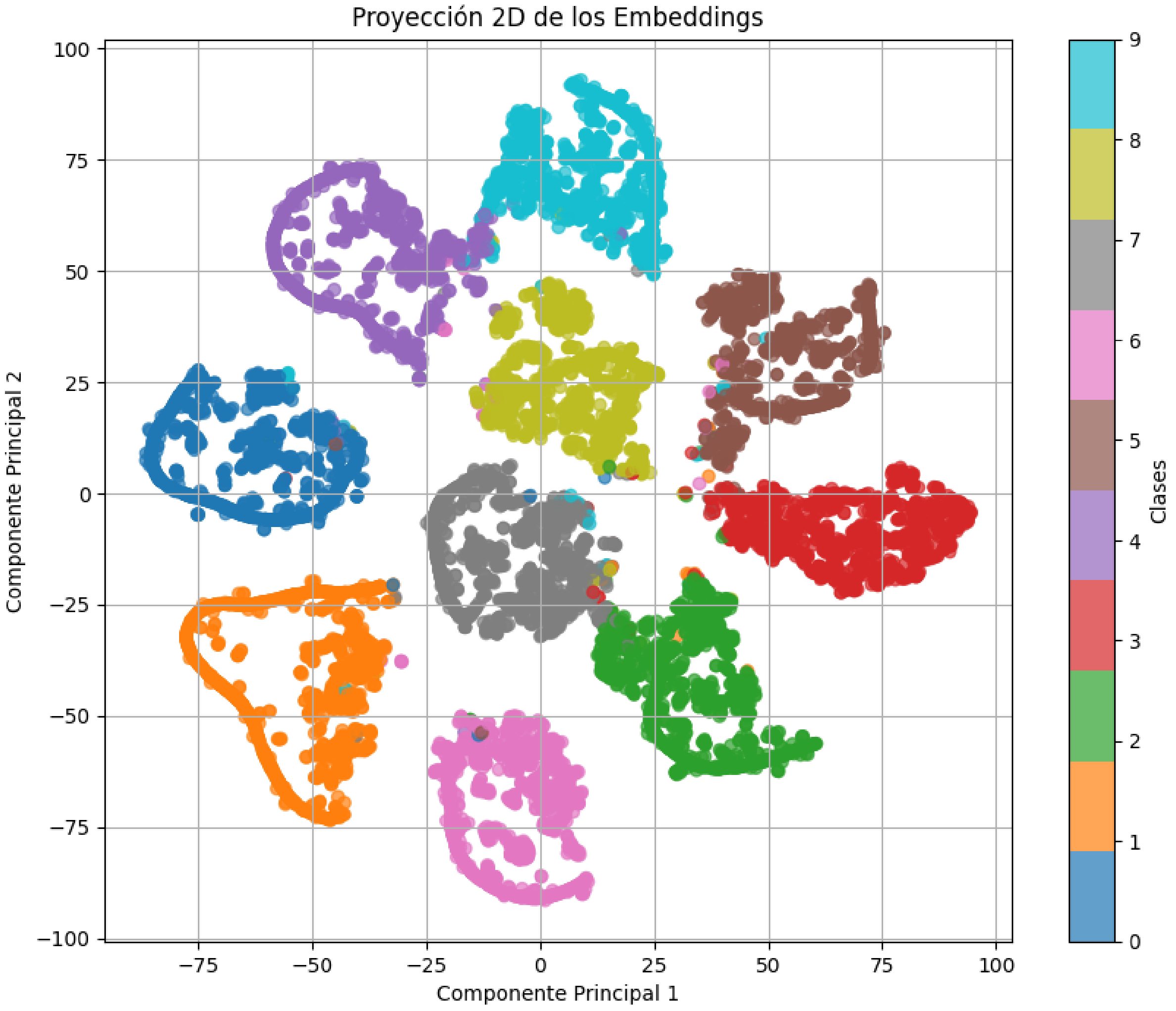

Figure 4: Embeddings of the test images projected to 2D using TSNE.

incorrectly classified images that are farthest from any of the centroids. These images correspond to embeddings with no close neighbors, and therefore the classifier does not know how to label them. We can observe that the images on the left correspond to poorly defined digits, and it is possible that there are not many similar examples in the training set. More surprising are the images on the right, where the digits are perfectly recognizable, yet the distance computed by the network suggests they are not close to the prototypical digits. In any case, these situations are not too problematic, as the system could choose to indicate uncertainty rather than provide an incorrect answer.

Figure 7 illustrates the opposite situation, where the neural network indicates that the images are very similar to one of the prototypical digits, which is, however, not the correct one. Some of these images may cause confusion, as their handwriting appears to be somewhere between multiple digits. Nevertheless, a person would still be able to correctly recognize the digits in most cases and some of them are in fact quite recognizable.

7. Conclusions and Future Work

Siamese networks appear to be highly effective at determining whether two images are similar (i.e., belong to the same class) or not (i.e., belong to different classes) in the MNIST dataset. Training through the optimization of the discriminative loss function results in very compact embedding clusters that are well separated from each other. This makes Siamese networks a promising approach for classification

Class	Precision	Recall	F1-score	Support
0	0.9839	0.9959	0.9899	980
1	0.9834	0.9938	0.9886	1135
2	0.9807	0.9835	0.9821	1032
3	0.9910	0.9772	0.9840	1010
4	0.9857	0.9796	0.9826	982
5	0.9680	0.9821	0.9750	892
6	0.9843	0.9791	0.9817	958
7	0.9702	0.9815	0.9758	1028
8	0.9906	0.9784	0.9845	974
9	0.9819	0.9673	0.9745	1009
Accuracy			**0.9820**	10000
Macro avg	0.9820	0.9819	0.9819	10000
Weighted avg	0.9821	0.9820	0.9820	10000

Table 2
Classification performance metrics per digit class on the MNIST test set using the closest centroid.

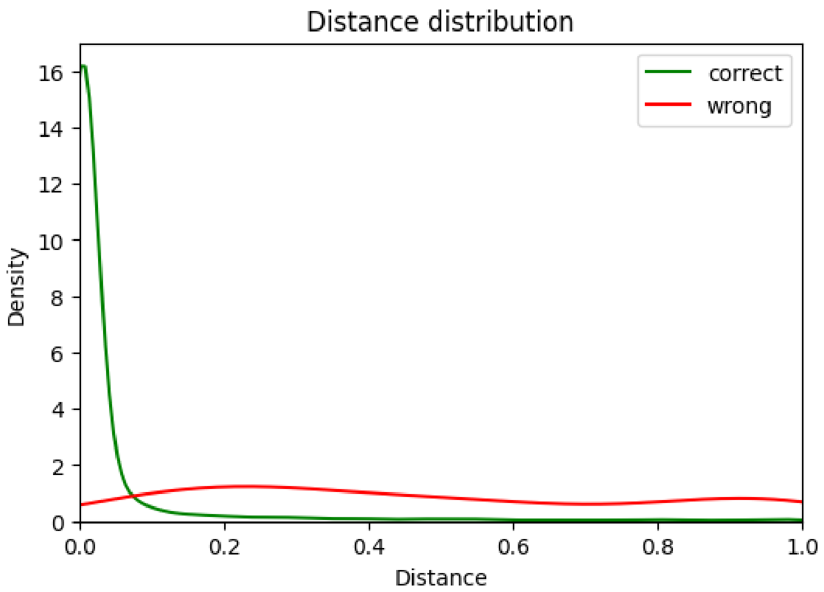

Figure 5: Distance distribution between the test images and the closest centroid.

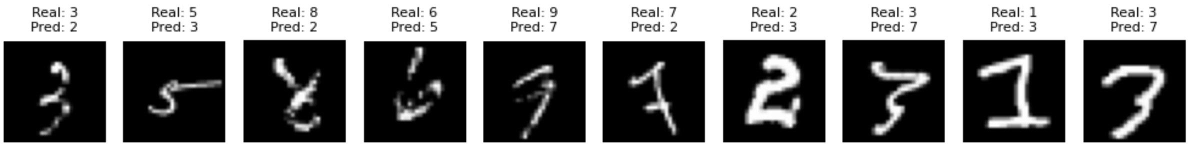

Figure 6: Images that were classified incorrectly and the classifier was quite unsure of their class. They are far from any centroid and ordered from left to right by decreasing distance.

problems that rely on nearest-neighbor retrieval. In fact, we can achieve fairly good results by simply computing the distance to the cluster centroids, which significantly reduces the computational cost of such algorithms.

However, the distance computed by the network does not seem to exhibit other desirable properties of a similarity metric. In particular, it does not appear to effectively reflect the degree of similarity between two instances, at least in the problem we have addressed in this work. Images from the same

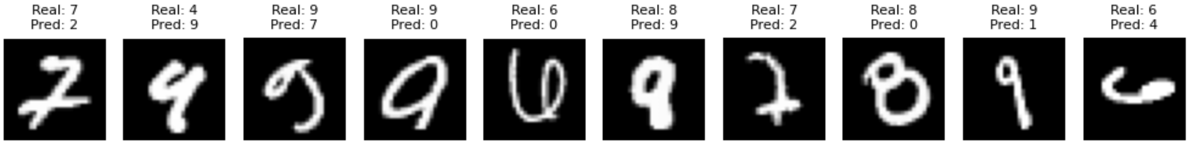

Figure 7: Images that were classified incorrectly but the classifier was quite sure of their class. They are close to some centroid and ordered from left to right by increasing distance.

class all are very similar to one another, and visually there does not seem to be a clear relationship between handwriting style and the distance computed by the network. A similar situation occurs with images from different classes, where the network simply indicates that they are very different, without effectively capturing that some digits are more visually like others.

Although the distance learned by the network can be useful in tasks where it is only necessary to determine whether two entities are similar or different, such as binary classification or identity verification, it may not be suitable for problems that require generating a similarity-based ranking.

For instance, in CBR systems, it is common to retrieve the most similar cases and then assess their contribution to the proposed solution based on their distance to the query. It is also standard practice to apply a threshold beyond which retrieved cases are no longer considered relevant. Both practices become more difficult when the learned distances are either too homogeneous (i.e., all cases are very similar or very dissimilar) or do not follow a smooth, interpretable distribution.

Furthermore, in recommender systems, where a ranked list of relevant items is typically provided, the ordering generated using this type of learned similarity may lack meaningful differentiation. The same challenge arises in distance-based explainability methods, where examples and counterexamples may not appear intuitive to users simply because the distance learned by the network does not reflect the types of features or relationships users expect to define similarity.

How could we encourage the network to generate better embeddings, where the relative distance captures more than just class membership? One possible approach is to incorporate into the loss function a component that does not rely on labels, but rather on the intrinsic characteristics of the images. In this regard, a *decoder* module could be added to attempt to reconstruct the original image, thereby forcing the network to encode the most significant visual patterns into the generated embeddings.

On the other hand, the network seems to make classification errors fairly uniformly across the entire range of distances, which is somewhat surprising. This could be due to the test set containing images that are very different from those in the training set, or it may suggest that the network is not able to adequately capture the patterns that distinguish different digits. By selecting training pairs more carefully, we might overrepresent the harder-to-classify images, thus encouraging the network to better learn their differences from other classes.

As part of future work, we plan to address these issues and analyze the behavior of these networks across different datasets. The ability to automatically learn a similarity measure between images is very promising, but we must develop mechanisms that align the learned similarity with our human notion of similarity.

Acknowledgments

This work has been partially funded by the Ministry of Science, Innovation and Universities (PID2021-123368OB-I00) and the Complutense University of Madrid (Group 921330).

Declaration on Generative AI

The author did not use any generative AI during the preparation of this work.

References

[1] E. L. Rissland, AI and similarity, IEEE Intell. Syst. 21 (2006) 39–49. URL: https://doi.org/10.1109/MIS.2006.38. doi:10.1109/MIS.2006.38.

[2] R. L. de Mántaras, E. Plaza, Case-based reasoning: An overview, AI Commun. 10 (1997) 21–29. URL: http://content.iospress.com/articles/ai-communications/aic106.

[3] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016. http://www.deeplearningbook.org.

[4] D. Chicco, Siamese neural networks: An overview, in: H. M. Cartwright (Ed.), Artificial Neural Networks - Third Edition, volume 2190 of *Methods in Molecular Biology*, Springer, 2021, pp. 73–94. URL: https://doi.org/10.1007/978-1-0716-0826-5_3. doi:10.1007/978-1-0716-0826-5_3.

[5] K. Martin, N. Wiratunga, S. Sani, S. Massie, J. Clos, A convolutional siamese network for developing similarity knowledge in the selfback dataset, in: A. A. Sánchez-Ruiz, A. Kofod-Petersen (Eds.), Proceedings of ICCBR 2017 Workshops (CAW, CBRDL, PO-CBR), Doctoral Consortium, and Competitions co-located with the 25th International Conference on Case-Based Reasoning (ICCBR 2017), Trondheim, Norway, June 26-28, 2017, volume 2028 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2017, pp. 85–94. URL: https://ceur-ws.org/Vol-2028/paper8.pdf.

[6] K. Amin, Cases without borders: Automating knowledge acquisition approach using deep autoencoders and siamese networks in case-based reasoning, in: 31st IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2019, Portland, OR, USA, November 4-6, 2019, IEEE, 2019, pp. 133–140. URL: https://doi.org/10.1109/ICTAI.2019.00027. doi:10.1109/ICTAI.2019.00027.

[7] M. Hoffmann, L. Malburg, P. Klein, R. Bergmann, Using siamese graph neural networks for similarity-based retrieval in process-oriented case-based reasoning, in: I. Watson, R. O. Weber (Eds.), Case-Based Reasoning Research and Development - 28th International Conference, ICCBR 2020, Salamanca, Spain, June 8-12, 2020, Proceedings, volume 12311 of *Lecture Notes in Computer Science*, Springer, 2020, pp. 229–244. URL: https://doi.org/10.1007/978-3-030-58342-2_15. doi:10.1007/978-3-030-58342-2_15.

[8] P. Klein, N. Weingarz, R. Bergmann, Using expert knowledge for masking irrelevant data streams in siamese networks for the detection and prediction of faults, in: International Joint Conference on Neural Networks, IJCNN 2021, Shenzhen, China, July 18-22, 2021, IEEE, 2021, pp. 1–10. URL: https://doi.org/10.1109/IJCNN52387.2021.9533544. doi:10.1109/IJCNN52387.2021.9533544.

[9] B. M. Mathisen, A. Aamodt, K. Bach, H. Langseth, Learning similarity measures from data, Prog. Artif. Intell. 9 (2020) 129–143. URL: https://doi.org/10.1007/s13748-019-00201-2. doi:10.1007/S13748-019-00201-2.

[10] X. Ye, D. Leake, W. Huibregtse, M. M. Dalkilic, Applying class-to-class siamese networks to explain classifications with supportive and contrastive cases, in: I. Watson, R. O. Weber (Eds.), Case-Based Reasoning Research and Development - 28th International Conference, ICCBR 2020, Salamanca, Spain, June 8-12, 2020, Proceedings, volume 12311 of *Lecture Notes in Computer Science*, Springer, 2020, pp. 245–260. URL: https://doi.org/10.1007/978-3-030-58342-2_16. doi:10.1007/978-3-030-58342-2_16.

[11] Z. Cheng, A. Yan, A case weighted similarity deep measurement method based on a self-attention siamese neural network, Ind. Artif. Intell. 1 (2023). URL: https://doi.org/10.1007/s44244-022-00002-y. doi:10.1007/S44244-022-00002-Y.

[12] L. Deng, The mnist database of handwritten digit images for machine learning research, IEEE Signal Processing Magazine 29 (2012) 141–142.

[13] Mehdi, Image similarity estimation using a siamese network with a contrastive loss, https://keras.io/examples/vision/siamese_contrastive/, 2021. Accessed: (2025-04-21).

[14] S. Chopra, R. Hadsell, Y. LeCun, Learning a similarity metric discriminatively, with application to face verification, in: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005), 20-26 June 2005, San Diego, CA, USA, IEEE Computer Society, 2005, pp. 539–546. URL: https://doi.org/10.1109/CVPR.2005.202. doi:10.1109/CVPR.2005.202.