Assessing Deep Learning Architectures for Visualizing Maya Hieroglyphs

Edgar Roman-Rangel(✉) and Stephane Marchand-Maillet

Department of Computer Science, University of Geneva, Geneva, Switzerland
{edgar.romanrangel,stephane.marchand-maillet}@unige.ch

Abstract. This work extends the use of the non-parametric dimensionality reduction method t-SNE [11] to unseen data. Specifically, we use retrieval experiments to assess quantitatively the performance of several existing methods that enable out-of-sample t-SNE. We also propose the use of deep learning to construct a multilayer network that approximates the t-SNE mapping function, such that once trained, it can be applied to unseen data. We conducted experiments on a set of images showing Maya hieroglyphs. This dataset is specially challenging as it contains multi-label weakly annotated instances. Our results show that deep learning is suitable for this task in comparison with previous methods.

Keywords: t-SNE · Deep learning · Visualization

1 Introduction

Visualizing high-dimensional data is a challenging problem commonly addressed by: (1) feature selection [5], where the data is represented using only a subset of the original feature space, which is defined by the most relevant dimensions of the high-dimensional data according to certain criteria; or (2) feature extraction [5], where a low-dimensional representation results from a mapping function able to retain properties of interest from the high-dimensional representation of the data. Often, 2 or 3 dimensions are kept for visualization purposes.

Several methods have been proposed merely with the purpose of visualizing a fixed dataset. Among them, t-distributed Stochastic Neighbor Embedding (t-SNE) [11] stands out due to the intuitive reasoning behind it and its relative easy implementation. Another characteristic that makes t-SNE popular is that it is non-parametric, which makes possible to use it for any type of data including instances with no labels [5]. However, it is unclear how to apply it to new instances that were not seen during the learning stage. To address this issue, only a few previous works have focused on developing out-of-sample extensions of t-SNE [4–6,10], obtaining promising results that have been evaluated mainly subjectively based on their resulting low-dimensional visualization.

Since the preservation of local structure is at the heart of t-SNE, we rely on retrieval experiments to evaluate the quality of the resulting low-dimensional representations. This is, we use a quantitative criterion. We also propose the use

J.A. Carrasco-Ochoa et al. (Eds.): MCPR 2017, LNCS 10267, pp. 137–146, 2017.
DOI: 10.1007/978-3-319-59226-8_14

of deep learning methods for learning to replicate a t-SNE mapping function previously computed. By comparing their retrieval performance, we are able to assess the mapping quality of several deep learning architectures and functions. Since we consider no label information for training the deep networks, our approach remains fully unsupervised, and naturally follows t-SNE in this regard.

The experiments on this work were performed on a challenging dataset of binary images containing Maya hieroglyphs, which are very complex in visual terms. Specifically, they are compounds of individual signs whose arrangement varies arbitrarily. Often, this results in partial overlaps of classes, both at the visual and semantic spaces, i.e., an instance of a weakly annotated scenario.

Our results show that: (1) evaluating the retrieval performance is adequate for assessing dimensionality reduction techniques; and (2) deep learning can approximate t-SNE mapping function better that previous methods, such it can be applied to unseen data, while remaining fully unsupervised.

The remaining of this paper is organized as follows. Section 2 discusses related work on out-of-sample dimensionality reduction. Section 3 presents our assessment of methods for out-of-sample dimensionality reduction. Section 4 details the experimental protocol we followed for the assessment. Section 5 discusses the results. And Sect. 6 presents our conclusions.

2 Related Work

Only a few attempts have been made to enable t-SNE with out-of-sample extensions, mainly because it was designed for visualization purposes rather than learning purposes. This is, the interest is only on visualizing the underlying distribution of points from a fixed dataset. Good review papers, including other dimensionality reduction techniques besides t-SNE, are found in [1,11].

One of such works [10] consists in learning a parametric model constructed upon stacked autoencoders, with a latter fine tuning step that takes into account the divergence between the distribution of point in both the high- and the low-dimensional representations. This approach is in fact, designed to improve the use of traditional autoencoders for dimensionality reduction [6,12], where the optimization was only driven by sparsity constraints. Both of these methods are of easy implementation and have attained decent results in the MNIST dataset. However, they seem to fail when dealing with multi-instance images.

Specific works focusing on out-of-sample extension of t-SNE [4,5] rely on regression approaches, either linear [4] or kernel-based [2]. These methods are simple and can obtain competitive results when trained with supervision, e.g., supervised t-SNE [7]. However, we are interested in an unsupervised scenario.

Therefore, we took a step further and combined intuitions behind these two types of approaches. Specifically, we use regression function embedded in neural networks to enable out-of-sample extensions to unsupervised t-SNE. The details of our approach are presented in Sect. 3.

3 Our Approach

This section presents our methodology. First, it provides a brief description of the t-SNE computation, and then it presents our proposed deep learning approach.

T-SNE [11] estimates a mapping function that provides low-dimensional representations, such that, for a given point of interest x_i, the probability p_{ij} of its neighbors $(x_j, \forall j \neq i)$ in the high-dimensional space is the same as the probability q_{ij} of its neighbors in the low-dimensional space. This mapping is obtained by minimizing the Kullback-Leibler divergence $D_{KL}(p\|q)$ between the probabilities across both spaces.

Here, probabilities in the original high-dimensional space are estimated by,

$$p_{ij} = \frac{p_{i|j} + p_{j|i}}{2n}, \tag{1}$$

where, n is the number of points in the full set, and

$$p_{j|i} = \frac{\exp\left(-0.5\|x_i - x_j\|^2/\sigma_i^2\right)}{\sum_{k\neq i} \exp\left(-0.5\|x_i - x_k\|^2/\sigma_i^2\right)}, \tag{2}$$

where, x_i denotes the i-th high-dimensional vector, and σ_i^2 is the variance of a Gaussian centered at x_i.

Complementary, the probabilities in the low-dimensional space are estimated using the Student t-distribution,

$$q_{ij} = \frac{\left(1 + \|y_i - y_j\|^2\right)^{-1}}{\sum_{k\neq i} \left(1 + \|y_i - y_k\|^2\right)^{-1}}, \tag{3}$$

where, y_i is the expected low-dimensional representation of point x_i.

As previously mentioned, this approach provides low-dimensional representations distributed similarly to their high-dimensional counterparts, thus it is suitable for unlabeled data. However, it is not suitable for replicating such transformation for unseen data.

Deep Learning approaches are able to approximate arbitrary complex transformation functions with high degree of precision [8]. The reason for their success in such complex tasks results from their multilayer arrange that process multiple non-linear transformations, which properly designed and trained, might achieve high levels of abstraction.

Here, we propose to exploit this characteristic to replicate t-SNE mapping function for unseen data. Mathematically, we optimize a loss function $\mathcal{L}\left(f_\Omega(x), y\right)$, where the loss $\mathcal{L}$ could refer to the mean square error (mse), or any other error, between the high-dimensional input vector x and the low-dimensional expected vector y; and the function $f_\Omega(\cdot)$ corresponds to a deep neural network, such that,

$$f_\Omega : x \mapsto y, \tag{4}$$

and it is parametrized by a set of weights Ω.

In Sect. 4 we detail several neural networks evaluated as the function f_Ω.

4 Experiments

This section details the procedure followed to validate our approach. First, it introduces the dataset used for experiments, and then the experimental protocol.

4.1 Dataset

The data used for our experiments is a collection of binary images containing Maya hieroglyphs. Specifically, Maya glyph-blocks, which correspond to compounds of individual glyph-signs visually arranged together to express full sentences. These images were generated by our team by applying binarization to a subset of glyph-blocks from the collection of the Maaya project [3][1]. Currently, we are working towards the release of this collection for research purposes.

In particular, our dataset consists of 155 classes, which correspond to 155 different types of glyph-blocks. More precisely, each class is defined by the sequence of names of its individual glyph-signs. Here the Thompson catalog [9] is used as naming convention, e.g., T0759b-T0025-T0181 refers to a glyph-block with three individual signs: 759b, 25, and 181; and the prefix 'T', for Thompson, is always used. These annotations were defined by expert epigraphers following a reading convention for Maya hieroglyphs. Note that following this definition of a class, two classes might partially overlap, i.e., two glyph-blocks might contain the same individual signs one or several times, or they might have the same set of signs but on different arrangement.

The 155 selected classes correspond to those glyph-blocks with at least 20 instances in the original Maaya corpus, from which we randomly picked exactly 20. Later, we generated 20 more instances by applying erosion, and 20 more with dilation, both convolving with filter of size fixed to 9×9 pixels. Therefore, our datasets contains 9300 instances, thus 60 per class. Furthermore, the 40 synthetically generated instances are always used as *training set* in our experiments, and the original 20 instances as *test set*. Figure 1 shows some examples of glyph-blocks.

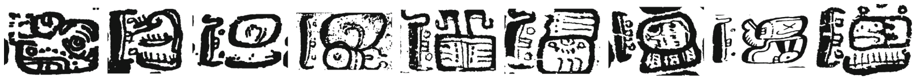


Fig. 1. Examples of Maya glyph-blocks. Each image exemplifies a specific class.

Representation. We used the VGGnet [8] as feature extractor, which has been previously trained on the ImageNet dataset. More precisely, we fed the images of the glyph-blocks to the VGGnet, and recorded the output of the last convolutional layer (conv5). This representations is a 4096-dimensional vector, which for our specific dataset of binary images, happens to be a very sparse

[1] www.idiap.ch/project/maaya/.

vector as only those units processing edge information are activated through the network. We use this representation as the input for all of our experiments.

Note that this corresponds to an instance of a transfer learning scenario. Or in other words, the pipeline for feature extraction is fully unsupervised since the VGGnet was trained on a totally different dataset.

4.2 Protocol

We used 6200 instances of glyph-blocks (40 per class) as training set, and the remaining 3100 instances (20 per class) for testing. Specifically, we computed an initial t-SNE dimensionality reduction on the training set, and then used the proposed approach (or the baseline methods) to estimate the corresponding mapping function that could replicate the initial t-SNE reduction. Finally, we evaluated its performance on unseen data, i.e., test set.

The Matlab implementation of t-SNE [11] that we used outputs 2-dimensional representations whose values are zero-centered and between -150 and 150. We scaled these representations to be between -1 and 1 before feeding them to the neural network. And we used hyperbolic tangent and rectified linear transfer functions as the units of the hidden layers in the network.

Table 1 shows the list of neural architectures and transfer functions that we evaluated for the replication of the t-SNE dimensionality reduction.

Table 1. Architectures and transfer functions evaluated for replicating t-SNE dimensionality reduction. Logistic and ReLu transfer functions where used with different combinations of hidden layers and units.

Network	Number of units in the hidden layers		
A	200 tanh	100 tanh	50 tanh
B	200 tanh	100 tanh	100 tanh
C	100 tanh	100 tanh	100 tanh
D	200 tanh	100 tanh	–
E	200 relu	100 relu	50 relu
F	200 relu	100 relu	100 relu
G	200 relu	100 relu	–
H	200 tanh	100 relu	100 tanh

We used gradient descent for all of our experiments and mean square error (mse) as loss function. Also, we used the following fixed hyperparameters: learning rate equals to 0.01; momentum rate of 0.9. We ran our experiments for 1000 epochs, and relied on early stopping if the optimization reached 10 iterations without improvement, or if the gradient reached a value of 10^{-10}.

We compared our results using retrieval experiments as evaluation of performance. We report the mean Average Precision (mAP) obtained for the first 10 and the first 20 retrieved elements, mAP@10 and mAP@20 respectively.

5 Results

Our first experiment, which later we use as baseline, is an initial comparison of the retrieval performance achieved by the output of the VGGnet before and after applying t-SNE in the traditional sense. Table 2 shows the mean Average Precision (mAP) obtained for the first 10 and the first 20 retrieved elements, mAP@10 and mAP@20 respectively.

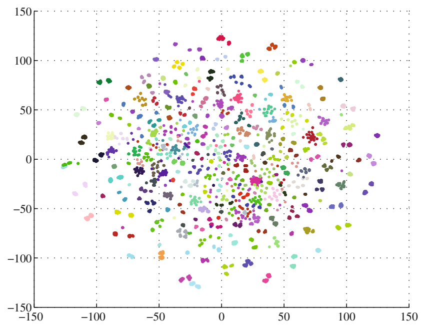
Table 2. mAP obtained on the training set alone and the training set combined with the test set. For both cases using the 4096-D vector provided by the VGGnet, and a 2-D vector after applying t-SNE.

Set	Dimensionality	mAP	
		@10	@20
Training	4096-D	0.9591	0.9258
Training	(t-SNE) 2-D	0.9509	0.9311
Training & Test	4096-D	0.9578	0.9245
Training & Test	(t-SNE) 2-D	0.9455	0.9287

The first observation is that squeezing the output of 4096 dimensions of the VGGnet into only 2 dimensions, results in a slight decrease in the retrieval performance (i.e., discriminative potential) for the first 10 retrieved elements. This behavior was already expected since, intuitively, 2 dimensions might be insufficient for discriminative representations of the very complex shapes in our dataset. However, it is interesting to notice that this drop is almost negligible. Furthermore, the retrieval performance shows a slight increment for the first 20 retrieved elements, i.e., from 0.9258 to 0.9311 when using only the training set, and from 0.9245 to 0.9287 when the training and test sets are merged. Another observation is that the retrieval performance decreases by adding the test set in the experiments. This is a consequence of the addition of noisy instances.

Since a new mapping function is obtained every time t-SNE is computed, results obtained using only the training set are, with high probability, different from those obtained using both the training and test sets together. This is to say, the retrieval performance is barely impacted when the test set is added to the training set, as t-SNE effectively estimates a mapping function considering all the elements in its input. However, such mapping function is different from that estimated only for the test set. Figure 2 shows two different point-clouds of 2-D coordinates for the training set only and the training and test sets together. It is clear that the location of colored clusters, which indicate different classes, is different in the two images.

In Table 3, we show the retrieval performance obtained with the several architectures described in Table 1. Compared to the baseline presented in Table 2, all of these approaches result in certain amount of loss of the discriminative capabilities of the vectors. Concretely, the best performing approach (B in Table 3)


(a) Training set only.

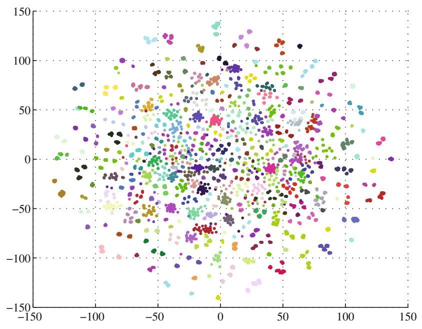

(b) Training and test set together.

Fig. 2. Point-clouds showing different 2-D representations obtained after dimensionality reduction of the VGGnet outputs using t-SNE.

shows about 23% lower retrieval performance. This is the result of compressing 4096 dimensions into only 2 dimensions, which corresponds to a very *aggressive* transformation. Although we notice that using only 12 dimensions is still enough to achieve results above 90% in retrieval precision, in this work we are interested in generating representations of only 2 dimensions for visualization purposes.

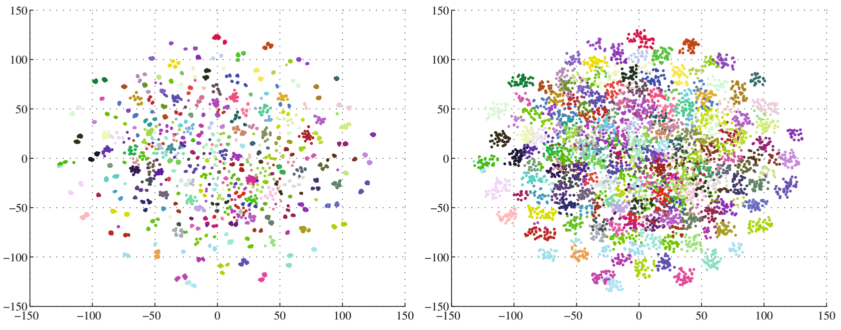
Table 3. *mAP* obtained for different architectures of deep neural networks.

mAP	A	B	C	D	E	F	G	H
@10	0.674	**0.719**	0.552	0.718	0.692	0.702	0.326	0.706
@20	0.655	**0.706**	0.551	0.700	0.690	0.689	0.211	0.702

From Table 3, it is clear that approach B produces results whose performance is above all the others. In general terms, the use of the hyperbolic tangent function (*tanh*) attained better results in comparison to the rectified linear function (*relu*). Comparing approach B with the other approaches that also use the hyperbolic tangent function (i.e., A, C, D), one can notice that decreasing the number of units in the first layer, as done in approach C, has a larger impact on the retrieval performance than decreasing it in the last layer, as done in approach A. Surprisingly, removing one layer, as in approach D, has little effect as long as the number of units in the last layer remains the same. Intuitively, this happens because the number of parameters decreases, thus making the model easier to train although slightly less robust.

This behavior is consistent with the results obtained using the rectified linear function, where approach F, which is similar to B in the number of layers and units, achieves higher performance with respect to other approaches using relu transfer functions (i.e., E, G). Finally, combining tanh and relu function has no effect in the performance, as seen by the fact that approach F and H provide almost the same retrieval performance.

Figure 3 shows the point-clouds obtained after using the dimensionality reduction approach B. To facilitate reading, Fig. 3a shows again the point-cloud resulting from applying traditional t-SNE on the training set (i.e., this is the same point-cloud as shown in Fig. 2a). However, different from Fig. 2b, which shows a full t-SNE dimensionality reduction applied on both the training and test sets, and which is very different from the point-cloud in Fig. 2a, the point-cloud presented in Fig. 3b shows the 2-D reduction of the training and test sets together after using approach B. Note that in this case, the color structure is conserved, i.e., the rough location of each class remains constant when applied to unseen data, with a regular distribution spread around their original locations. This is, the proposed method is able to generalize the use of t-SNE for unseen data up to certain degree.



(a) Training set only. (b) Training and test set together.

Fig. 3. Point-clouds showing the 2-D representations resulting from applying the proposed dimensionality reduction approach, i.e., approach B.

Specifically, the spreading in the location of the 2-dimensional representations, which results from using the proposed approach, and which is shown in Fig. 3b, is the reason for the drop in retrieval performance reported in Table 3.

For further validation of the proposed approach, we compared it against other methods previously reported in the literature. Table 4 shows the retrieval performance obtained from this comparison. In particular, the two autoencoder-based methods, autoencoder and parametric t-SNE, both obtained retrieval performance slightly below 70%. The ideas behind autoencoders and parametric t-SNE are very similar to each other, with the main difference that the fine tunning step of parametric t-SNE focuses on the optimization of the Kullback-Leibler divergence, whereas autoencoders are only constrained by a sparsity prior. However, this difference seems to be negligible, as suggested by the very similar performance obtained by these two methods.

From Table 4, one can also see that it is possible to achieve more than 50% retrieval precision by solving a simple linear regression formulation. Furthermore,

Table 4. mAP obtained for the best performing architecture according to Table 3, and other approaches for regression.

Method	mAP@10	mAP@20
[6] Autoencoder	0.668	0.695
[10] Parametric t-SNE	0.677	0.667
Linear regression	0.569	0.545
[4] MSI	0.615	0.609
[4] Kernel t-SNE	0.654	0.648
Our approach	**0.719**	**0.706**

the two methods presented in [4], which are kernel-based variations of linear regression, and that are referred to as MSI and kernel t-SNE, both obtained slightly improved retrieval precision with respect to the use of simple linear regression. However, all these three methods performs slightly worse than the autoencoder-based methods.

Finally, the proposed approach based upon the use of deep networks achieves about 72% of retrieval precision, which corresponds to a rough 23% of drop in performance when compared to traditional t-SNE. However, this high performance of t-SNE remains true as long as it is computed using the test set as part of the training set. In contrast, our approach enables the ability to generalize for unseen data within a range of acceptable loss in the discriminative potential of the resulting 2-dimensional representations.

6 Conclusions

We proposed the use of retrieval experiments for assessing the quality of unsupervised and non-parametric dimensionality reduction techniques. Specifically, for methods that replicate the mapping function of t-SNE for unseen data. We conducted experiments on a dataset of Maya hieroglyphs, whose main challenge lies on the visual complexity of its instances and the fact that they are weakly annotated. We also proposed an unsupervised method for estimating 2-D representations of very complex shapes, which can be readily applied to unseen data. More specifically, a deep network architecture able to approximate t-SNE mapping functions up to certain degree.

Our results show that using retrieval experiments is an adequate criteria for evaluation of non-parametric dimensionality reduction methods, and that the proposed approach performs better than previous methods. Furthermore, 2-dimensional representation is adequate for visualization purposes of complex shapes at the cost of lower retrieval performance.

For future work, more complex architectures and combinations of transfer functions, which could yield improved results should be evaluated. Also, it is expected that the evaluation of our approach on a supervised scenario will result in improved performance.

Acknowledgments. This work was supported by the Swiss-NSF MAAYA project (SNSF-144238).

References

1. Bengio, Y., Courville, A., Vincent, P.: Representation learning: a review and new perspectives. IEEE Trans. Pattern Anal. Mach. Intell. **35**(8), 1798–1828 (2013)
2. Bunte, K., Biehl, M., Hammer, B.: A general framework for dimensionality-reducing data visualization mapping. Neural Comput. **24**(3), 771–804 (2012)
3. Gatica-Perez, D., Gayol, C.P., Marchand-Maillet, S., Odobez, J.-M., Roman-Rangel, E., Krempel, G., Grube, N.: The MAAYA project: multimedia analysis and access for documentation and decipherment of maya epigraphy. In: Workshop DH (2014)
4. Gisbrecht, A., Lueks, W., Mokbel, B., Hammer, B.: Out-of-sample kernel extensions for nonparametric dimensionality reduction. In: Proceedings of the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (2012)
5. Gisbrecht, A., Schulz, A., Hammer, B.: Parametric nonlinear dimensionality reduction using kernel t-SNE. Neurocomputing **147**, 71–82 (2015)
6. Hinton, G., Salakhutdinov, R.R.: Reducing the dimensionality of data with neural networks. Science **313**(5786), 504–507 (2006)
7. Kim, H., Choo, J., Reddy, C.K., Park, H.: Doubly supervised embedding based on class labels and intrinsic clusters for high-dimensional data visualization. Neurocomputing **150**, 570–582 (2015)
8. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. CoRR abs/1409.1556 (2014)
9. Thompson, J.: A Catalog of Maya Hieroglyphs. University of Oklahoma, Norman (1962)
10. van der Maaten, L.: Learning a parametric embedding by preserving local structure. In: Proceedings of the International Conference on Artificial Intelligence and Statistics (2009)
11. Maaten, L., Hinton, G.: Visualizing data using t-SNE. J. Mach. Learn. Res. **9**, 2579–2605 (2008)
12. Wang, W., Huang, Y., Wang, Y., Wang, L.: Generalized autoencoder: a neural network framework for dimensionality reduction. In: Proceedings of IEEE CVPR (2014)