Learning Graph Distances with Message Passing Neural Networks

Pau Riba*, Andreas Fischer†‡, Josep Lladós* and Alicia Fornés*
*Computer Vision Center, Computer Science Department, Universitat Autònoma de Barcelona, Barcelona, Spain
Email: {priba, josep, afornes}@cvc.uab.es
†Department of Informatics, DIVA Group, University of Fribourg, Fribourg, Switzerland
‡Institute of Complex Systems, University of Applied Sciences and Arts Western Switzerland, Fribourg, Switzerland
Email: andreas.fischer@unifr.ch

***Abstract*—Graph representations have been widely used in pattern recognition thanks to their powerful representation formalism and rich theoretical background. A number of error-tolerant graph matching algorithms such as graph edit distance have been proposed for computing a distance between two labelled graphs. However, they typically suffer from a high computational complexity, which makes it difficult to apply these matching algorithms in a real scenario. In this paper, we propose an efficient graph distance based on the emerging field of geometric deep learning. Our method employs a message passing neural network to capture the graph structure and learns a metric with a siamese network approach. The performance of the proposed graph distance is validated in two application cases, graph classification and graph retrieval of handwritten words, and shows a promising performance when compared with (approximate) graph edit distance benchmarks.**

I. Introduction

Graph-based representations have been widely used in different fields such as pattern recognition [1], bioinformatics [2] etc. Graphs are powerful and flexible representations able to describe shapes in terms of relationships between constituent parts or primitives. In document analysis, and in particular in handwriting recognition, graphs can perfectly capture the structure of shapes consisting in line(stroke)-based components. Graph-based representations can capture the structural invariance among writer styles or the inherent stroke variability of handwriting. Although statistical representations are still on top of the state of the art in handwritten word spotting and recognition, the new advances in graph matching have resulted in approaches whose performance is approaching the benchmark. Thus, graph-based approaches are nowadays competitive not as an alternative but complementary methods. Richer models that include structure-invariant representations of strokes allow to tackle with multiwriter scenarios avoiding specific learning for each writing style.

Since graphs are symbolic representations, they are more complex than vectorial representations such as SIFT or HOG, where simple mathematical operations can be applied. Basic operations in the vector domain have to be formally defined for graph data. When dealing with graphs, an important property is the ability to compare two graphs in terms of a similarity or distance. Lots of efforts have been made in this direction, for instance, error-tolerant or inexact graph matching algorithms have been proposed to cope with deformations between graphs. The problem consists in finding the minimum transformation cost such that an isomorphism exists between the transformed graph and the second one. Graph edit distance (GED) [3] algorithms are widely used as error-tolerant graph matching methods. The main drawback of GED is the time complexity which is exponential with respect to the number of nodes of the input graphs. Hence, GED is unfeasible in a real scenario without constraints in terms of graph size. Recently, several algorithms have been proposed to cope with this complexity [4], [5]. However, these approximate algorithms only consider very local node structures in their computation.

The success of convolutional neural networks (CNNs) [6] in different fields such as computer vision and natural language processing, has increased the interest of extending these frameworks to non-Euclidean structures, such as graphs and manifolds. These extensions are called *geometric deep learning*[1]. This emerging field has provided new learning tools when dealing with graphs [2], [7], introducing a huge range of novel applications to the field.

In this work, we propose a method to learn an enriched graph representation and a graph distance with a message passing neural network. Distance learning for graphs is achieved with a siamese architecture, inspired by earlier work in distance learning for images with siamese neural networks [8]. A siamese architecture uses the same model and weights to learn a representation where distances can be computed. The proposed approach is validated using standard graph datasets for letters and handwritten words. In this application scenario, the proposed approach based on a message passing neural network shows competitive results. Thus the use of structural information is a solid and promising complement to improve the state of the art in word recognition and spotting.

The rest of this paper is organised as follows. Section II reviews the related work. Section III describes the algorithm to learn a suitable representation for comparing pairs of graphs. Section IV presents the experimental validation. Finally, Section V draws the conclusions and future work.

[1] http://geometricdeeplearning.com/

II. Related Work

A. Graph edit distance

Graph edit distance (GED) [3] evaluates the similarity of two graphs in terms of edit operations. The computation of the distance is inspired by the string edit distance formulation. Thus, the main idea is to compute the minimum cost transformation from the source graph to the target one in terms of a sequence of edit operations. The typical edit operations are node and edge insertion, deletion and substitution. Each edit operation has an associated that is added to the final edit distance at every step. Among all the possible edit sequences, the distance is formulated in terms of the minimum cost edit sequence. GED algorithm finds an optimal edit sequence, but its computational complexity is exponential in the number of nodes of the involved graphs. Formally, GED is defined as

$$\mathrm{d}(G_1, G_2) = \min_{(e_1,\ldots,e_k)\in\Upsilon(G_1,G_2)} \sum_{i=1}^{k} \mathrm{c}(e_i),$$

where $\Upsilon(G_1, G_2)$ denotes the set of edit paths transforming G_1 into G_2, and c is the cost function of the edit path e_i. Due to the computational complexity, several approximations to GED have been proposed. *Bipartite graph matching* (BP) algorithm, proposed by Riesen *et al.* [4], is based on the assignment problem solution using a cost matrix with the edit operations costs. It provides an upper bound of order $\mathcal{O}((n_1 + n_2)^3)$ of the original GED. Another approximation called *Hausdorff Edit Distance* (HED) was proposed by Fischer *et al.* [5]. It is based on the *Hausdorff matching* and provides a lower bound of order $\mathcal{O}(n_1 \cdot n_2)$ of the original graph edit distance. Many other proposals have been put forward on how to obtain an efficient and accurate approximation of GED [9].

A typical drawback of the approximation algorithms is that they rely only on local edge structures rather than global information. Some efforts have been made to improve the performance by increasing the node context at matching time [10]. However, obtaining a better knowledge on the relation of each node within the graph is still an open issue. In this work, we propose to learn a metric between graphs able to distinguish two classes rather than compute an exact edit distance.

B. Graph-based Keyword Spotting

Keyword Spotting (KWS) is defined as the task of retrieving instances of a given keyword in a document without explicitly transcribing it. Graph representations for KWS have gained popularity in the literature. Wang *et al.* [11] represented handwritten words using characteristic skeleton points *i.e.* starting/ending, high-curved and branch points, labelled with shape context features. In [12] a convexity-based representation is proposed nodes represent convexities labelled with shape descriptors. These works faced the problem of segmentation based keyword spotting. More recently Riba *et al.* [13] proposed a segmentation free approach based on graph indexation. Stauffer *et al.* [14] proposed a pure structural representation, where the nodes are only labelled with their image coordinates. Although BP is the usually used algorithm, it has the drawback that is not able to learn adapting the costs to the particularities of handwriting.

C. Geometric deep learning

Geometric deep learning (GDL) generalises deep neural models to non-Euclidean data, *i.e.*graphs and manifolds [15]. In this work, we are focusing on the graph domain. Geometric deep learning provides tools able to learn representations at graph or node level providing information on its topology. Several recent works have proposed different architectures. In general, these learning architectures on graphs can be divided in two groups, spatial and spectral domain respectively. *Spatial domain* methods extend the idea of CNN at the image domain by moving a filter across nodes and applying a set of operations involving the local neighbourhood to compute a new representation [16], [17]. *Spectral domain* architectures take advantage of the spectral graph theory [18] in order to generalize the convolution operation to arbitrary graphs by means of the graph Laplacian [19], [20].

Recently, Gilmer *et al.* [2] proposed an approach named *Message Passing Neural Networks* (MPNNs) as a general supervised learning framework for graphs. This approach is able to generalize the aforementioned methodologies to have a common pipeline. Therefore, they redefine the previous spatial and spectral architectures using the MPNN framework by means of two phases: *message passing* and *readout*.

In the message passing phase, the hidden state h_v^t of each node v is updated by a node update function U_t which receives a message m_v^{t+1} collected from the neighbouring nodes. This phase is repeated during T time steps and is defined as:

$$m_v^{t+1} = \sum_{w\in\mathcal{N}(v)} M_t(h_v^t, h_w^t, e_{vw}), \tag{1}$$

where h_w^t and h_v^t are the hidden states of nodes v and w at iteration t and $\mathcal{N}(v)$ denotes the neighbours of v in graph G. Afterwards, the hidden state of node v is updated according to the message m_v^{t+1}.

$$h_v^{t+1} = U_t(h_v^t, m_v^{t+1}) \tag{2}$$

The message passing phase gathers structural information of the graph and embeds this information as node labels. The readout phase computes a feature vector for the whole graph based on the set of hidden states of the nodes. Hence, the readout function R must be permutation invariant,

$$\hat{y} = R(\{h_v^T | v \in G\}) \tag{3}$$

The three functions are learned and must be differentiable. In the proposed model, the readout function will be omitted in order to have a graph metric space.

III. Architecture based on a Siamese MPNN model

Following the idea of a siamese neural network, a twin MPNN with shared weights is applied to the input graphs. MPNNs have the ability to update the hidden state of a particular node v with the information of its neighbourhood sent

through a particular edge e. Through the different time steps, the system is gathering structural information of the local context of the node. Thus, the MPNN approach is learning an enriched representation of the original graph. The readout function R that is applied to this enriched representation is able to map a graph G to a vector space. The consideration of this readout function raises two drawbacks: first, there is no node correspondence between the graphs; and second, individual properties for nodes and edges are not taken into account.

To avoid these drawbacks, we discard the readout phase and directly compute a distance between enriched graph representations provided by the message passing phase. Therefore, we propose to follow a GED idea to obtain a similarity metric between graphs. Figure 1 shows the proposed architecture.

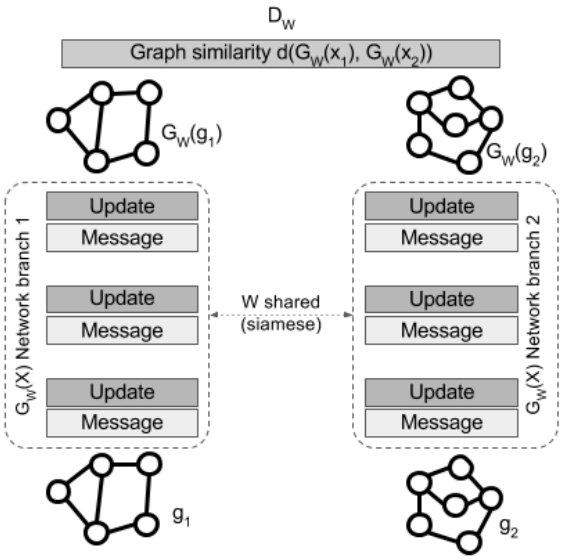

Fig. 1. Architecture for the proposed siamese MPNN model. Pairs of message and update functions are iterated a predefined set of time steps T.

A. MPNN Model

Our architecture is based on the model proposed by Li *et al.* [17]. It assumes discrete edge types. The message function is formulated as $M(h_v, h_w, e_{vw}) = A_{e_{vw}} h_w$, where $A_{e_{vw}}$ is a learned matrix for each possible edge label. To overcome the restriction imposed by this function, Gilmer *et al.* [2] modified the message function as $M(h_v, h_w, e_{vw}) = A(e_{vw})h_w$ to allow vectorial data as edge attributes, where $A(e_{vw})$ is a neural network which maps the edge vector to a matrix. The update function is defined as $U(h_v, m_v) = \mathrm{GRU}(h_v, m_v)$, where GRU is the Gated Recurrent Unit [21]. For comparison reasons, the readout function has been defined as

$$R = \sum_{v \in V} \sigma(i(h_v^{(T)}, h_v^0)) \odot (j(h_v^T)),$$

where i and j are neural networks and σ is the Sigmoid activation function.

B. Similarity

As explained in Section II, GED is a traditional metric to measure the similarity between graphs. To use this similarity metric in the proposed siamese approach, we have two main restrictions. First the approach must be computationally fast. Second it must be differentiable, able to backpropagate the gradients and update the MPNN weights. We propose a simple but effective metric based on the Hausdorff distance. The Hausdorff distance of two sets A and B on a metric space, with the metric $d(a, b)$ where $a \in A$ and $b \in B$ is defined as:

$$\mathrm{H}(A, B) = \max\big(\sup_{a \in A} \inf_{b \in B} d(a, b), \sup_{b \in B} \inf_{a \in A} d(a, b)\big)$$

For finite sets A,B the Hausdorff distance is defined as:

$$\mathrm{H}(A, B) = \max\big(\max_{a \in A} \inf_{b \in B} d(a, b), \max_{b \in B} \inf_{a \in A} d(a, b)\big)$$

By definition, Hausdorff distance is very sensitive to outliers. Hence, as proposed in [5], replacing the maximum operator with the sum, forces the distance to take into account all nearest neighbour distances and becomes more robust to noise than the original one. Thus, we can define the distance as:

$$\hat{\mathrm{H}}(A, B) = \sum_{a \in A} \inf_{b \in B} d(a, b) + \sum_{b \in B} \inf_{a \in A} d(a, b)$$

Therefore, we define the distance between two graphs $g_1 = (V_1, E_1, \mu_1, \nu_1)$ and $g_2 = (V_2, E_2, \mu_2, \nu_2)$ as:

$$d(g_1, g_2) = \frac{\hat{\mathrm{H}}(V_1, V_2)}{|V_1| + |V_2|}$$

The distance d can be seen as a specific case of HED where all nodes must be substituted and there are no insertions nor deletions. Moreover, the edges are not taken into account because the local structure exploited by HED has been embedded during the message passing phase.

C. Training

The proposed approach is trained in a supervised manner knowing whether or not a pair of graphs belong to the same class. All the models were trained using the *Stochastic Gradient Descent* optimiser with Momentum and weight decay *i.e.* L_2 regularization. The proposed objective function that is minimised is a *Contrastive loss function* based on the proposed distance, and it is defined as:

$$l(D_W) = Y\frac{1}{2}(D_W)^2 + (1 - Y)\frac{1}{2}\{\max(0, m - D_W)\}^2$$

where $D_W = d(G_W(X_1), G_W(X_2))$ is a distance defined between the outputs of the message phase of our model G_W with shared weights W; $Y \in \{0, 1\}$ is a label indicating positive or negative pairs of graphs, *i.e.* they belong to the same class; and m is a margin. In this work, m is set to 1.0.

IV. Experimental Validation

For validating the proposed approach, two application cases have been considered, graph classification and graph retrieval. All the code is available at github.com/priba/siamese_ged.

A. Datasets

The experimental evaluation uses benchmarks for graph classification *viz. Letters* and *Histograph* datasets whereas graph retrieval has been evaluated on a keyword spotting dataset *viz. George Washington* (GW) dataset.

Letters: The *Letters* graph database is a part of the IAM graph repository[2] [22] and consists of 6750 graphs representing 15 different capital letters of the Roman alphabet with different levels of noise generated from a prototype graph. The dataset is split into train, validation and test sets, where each set contains 750 graphs uniformly distributed among the classes. The goal is to predict the class of the graphs representing one of the 15 different letters. The nodes of the graphs are labelled with a two-dimensional attribute giving its position, and edges are unlabelled.

HistoGraph: The *HistoGraph* dataset[3] [23] consists of graphs representing words from the communicating letters written by the first US president, George Washington. It consists of 293 graphs generated from 30 different words. Given a word, the goal is to predict its class, choosing among the 30 words. Nodes are only labelled with their position in the image. Furthermore, this dataset used 6 different graph representation paradigms for delineating a single word into a graph, which results in 6 different subsets of graphs. The entire dataset is divided into 90, 60 and 143 graphs respectively for train, validation and test. Here, we will use the most promising subsets of graphs according to [23]: *Keypoint* and *Projection*.

The summary of both datasets is presented in Table I and some example graphs are shown in Figure 2.

TABLE I
DETAILS ON LETTERS AND HISTOGRAPH DATASETS

Datasets	Subset	#Graphs	#Class	Avg.($\|V\|$, $\|E\|$)	Labels
Letters	LOW MED HIGH	2250 (750, 750, 750)	15	(4.676, 3.132) (4.675, 3.206) (4.670, 4.500)	(x,y) position
HistoGraph	Keypoint Projection	293 (90, 60, 143)	30	(73, 67) (44, 41)	(x,y) position

George Washington: The *George Washington* dataset introduced in [24] is based on handwritten letters in English. It consists of twenty pages with a total of $4,894$ handwritten words. The same experimental setup from [25] has been used. This dataset consists of 105 different keywords, with $2,447$, $1,000$ and $1,224$ words for train, validation and test sets. This setting is known as the segmentation-based query-by-example task where all the words have been previously cropped. From the word images, the *Keypoint* protocol presented in the *Histograph* dataset is used to extract the graphs.

Graph Representation: All the datasets presented in this work only contain spatial (node labels) and structural (unlabelled edges) information. However, in order to exploit the power of message passing, edges are enriched by adding the spatial relation between nodes, *i.e.* l_2-distance and angle of the edge. Thus, the graphs are made directed. This is done in the specific domain of pattern recognition where the spatial relation within the parts is important.

[2]Available at http://www.fki.inf.unibe.ch/databases/iam-graph-database
[3]Available at http://www.histograph.ch/

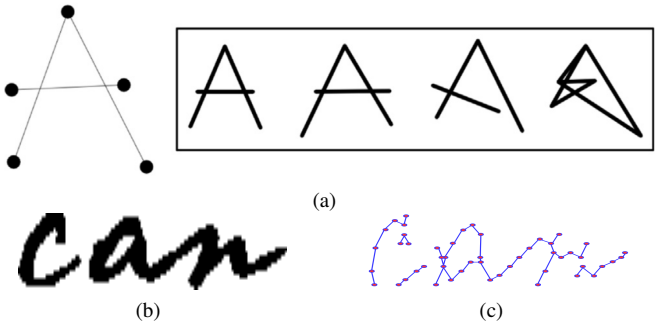

Fig. 2. (a) A graph representation of a capital letter at different distortion levels from the Letters dataset (extracted from [5]), (b) a handwritten word belonging to the George Washington dataset, (c) a graph representation of the handwritten word in (b).

B. Experimental Setup

The experiments presented in this paper make use of the following hyper-parameters. T is set to 3 time steps. Regarding the message passing function, A is a neural network which maps the edge features 2 to 64 by a 64 matrix making use of 4 linear layers and ReLU activation function. Hence, the message has a size of 64. The update function uses a hidden state of size 64 for the GRU network. Regarding the readout function, i and j are 2 layer neural networks with the ReLU activation function which maps 66 and 64 vectors to a vector of size corresponding to the number of classes at each problem. The model has been trained using 400 epochs for the Letters dataset and 200 epochs for the other experiments.

Two application cases have been proposed to evaluate our approach, classification and retrieval of graphs.

Classification: The first application scenario consists in classifying a graph among a set of different classes. Following the setup proposed by [5], a k-nearest neighbour classifier (KNN) is used based on graph distances. The system is trained as a siamese approach selecting pairs in the training set and validated on pairs within the validation set with a simple threshold. In order to balance the number of positive and negative samples in training time, a Weighted Random Sampler with replacement has been used. $M = 2Nn(n-1)$ graphs have been used to train and validate each epoch where, N is the number of classes and n is the average number of elements per class. Finally, for testing the approach, nearest neighbours from the test set are selected in the training set to measure the final system performance which is comparable to classical approaches. The proposed approach has been run 5 times in order to know the effect of the random initialisation.

Retrieval: Graph retrieval extracts samples from a graph dataset given a query graph. This problem is usually tackled by computing a ranking of similarity measures between elements. As a retrieval scenario, we evaluated our approach in *keyword spotting* (KWS) problem. Similarly to the classification problem, pairs are selected in the training and validation sets.

C. Results and Discussion

In our evaluation we compare with classical graph edit distance algorithms such as BP and HED.

The *Letters* dataset has been used in a graph classification problem as a sanity-check of our proposed approach. This simple dataset is also used to check that the model is able to learn, and to test and to evaluate parameters in a controlled scenario. Since it is a small dataset in terms of nodes, almost the whole graph can be covered with a single jump, and the learning may be redundant and noisy. The state of the art on this dataset is almost 100% which which shows the simplicity of it. Table II shows the accuracies obtained using a 5-NN classifier as in [5]. MPNN directly performs a classification using the readout function combined with a *Softmax* and *Cross-entropy loss*. Siamese MPNN corresponds to the performance with the model trained and tested using the proposed distance. The performance of BP and HED was also tested using the previous trained system.

TABLE II
RESULTS FOR THE LETTERS DATASET. MPNN APPROACHES PROVIDE ACCURACY ± STANDARD DEVIATION FOR 5 RUNS.

	LOW	**MED**	**HIGH**
BP [4]	99.73	94.27	89.87
HED [5]	97.87	86.93	79.2
Embedding[4][26]	99.80	94.90	92.90
MPNN	95.04 ±0.7224	83.20 ±1.2189	72.27 ±2.0060
Siamese MPNN	98.08 ±0.1068	89.0136 ±0.1808	74.77 ±6.4505
Test BP	98.19 ±0.1361	88.37 ±0.41	79.65 ±6.4345
Test HED	98.00 ±0.1461	89.79 ±0.3110	77.07 ±5.6106

Firstly, we observe that a siamese MPNN is more stable and obtains better accuracy than the classification through the readout function on both subsets LOW and MED. Also, it has better accuracy in the subset HIGH even though it is not as stable as in the other subsets. A more elaborated distance in test time BP or HED only improves in the HIGH subset. We assume that the importance of the edges is emphasized using our approach. Therefore, high distortions in the graphs introduce a level of noise that cannot be dealt with our proposed system. Compared to the state-of-the art, our approach improves the HED which is the reference used to build our distance. However, the use of a more elaborated distance such as BP or an Embedding is clearly superior to our approach.

The *Histograph* dataset provides an insight on our approach when dealing with bigger graphs, also in a classification scenario. We observe that this is a rather small dataset, only 3 samples per class are seen at training time. However, our approach can extract useful information able to improve the accuracy using either embeddings or a graph edit distance algorithm. Table III presents the results on two subsets and two reference systems. For direct comparison with the BP results presented in [23], we show the results using a 5-NN classifier although our experiments show that it is better to use a smaller k such as 3. This observation makes sense taking into account that only 3 examples per class appear in the training set.

TABLE III
CLASSIFICATION ACCURACY FOR THE HISTOGRAPH DATASET.

			Siamese MPNN	
Subset	**BP** [23]	**PSGE**[4][27]	3-NN	5-NN
Keypoint	77.62	80.42	**85.31 ± 1.6552**	82.80 ± 0.5600
Projection	81.82	80.42	**73.15 ± 2.6014**	69.65 ± 1.5064

We also observe that the *Projection* subset leads to worse results than *Keypoint*. We assume that MPNN trusts a lot on the structure, which is better represented on the Keypoint. The query time required for classification is around 0.5s per query in the Letters dataset (750 comparisons) and 3s per query in Histograph using the Keypoints data (90 comparisons). Note that the time increases a lot with the number of nodes.

Finally, the *George Washington* dataset has been used in a retrieval scenario *viz. Keyword Spotting*. A retrieval problem evaluates the capacity of our system to provide a ranking of retrieved graphs. Word spotting is usually evaluated in terms of the *mean average precision* (mAP) metric. Firstly, let us define Average Precision (AP) as

$$\mathrm{AP} = \frac{\sum_{n=1}^{|\mathrm{ret}|} P@n \times r(n)}{|\mathrm{rel}|}$$

where $P@n$ is the precision at n and $r(n)$ is a binary function on the relevance of the *n*-th item in the returned ranked list. Taking the AP definition, the mAP is defined as

$$\mathrm{mAP} = \frac{\sum_{q=1}^{Q} \mathrm{AP}(q)}{Q},$$

where Q is the number of queries.

As this dataset is an extended version of *Histograph*, the same training protocol has been used. Table IV shows a comparison with other techniques. Special interest comes from *Mean Ensemble BP* which is another graph-based technique. Observe that our approach largely outperforms (more than 5 points) their performance thanks to the enriched graph representation. As stated before, graph-based representations are gaining popularity in word spotting thanks to their ability to model the deformations present on handwritten words. They offer complementary properties with respect to statistical techniques. Moreover, some of the statistical methods shown in the Table require the transcription of words to be trained. Contrary, our approach only requires to know if the graphs belong to the same class or not at training time. Moreover, we are able to compute graph distances with out-of-vocabulary words (i.e. words never seen at training).

[4]Makes use of an SVM classifier.
[5]Segmentation-free, query-by-string
[6]Learning-free

TABLE IV
MAP FROM DIFFERENT APPROACHES ON GW DATASET.

Method		mAP
PHOC [28][5]		64.00
BOF HMM [29][5]		80.00
DTW[6]	DTW'01 [30]	42.26
	DTW'08 [31]	63.39
	DTW'09 [32]	64.80
	DTW'16 [33]	68.64
Mean Ensemble BP [25][6]		69.16
Siamese MPNN		75.85±3.64

V. CONCLUSIONS AND FUTURE WORK

In this paper we have proposed a siamese MPNN architecture for computing graph distances. Our architecture is able to learn node features based on structural information of nodes local contexts. These learned features lead to an enriched graph where we propose to compute a graph distance. We have validated our proposed architecture on two application scenarios: classification and retrieval in handwriting. We have observed that our methodology emphasises the structure, so it suffers from extreme distorted graphs in terms of their edges. However, in a real application scenario, where the distortions are more realistic, our methodology is able to deal with them.

Future work will be devoted to explore the possibility to modify the graph similarity module to be capable to compute a Graph Edit Distance approximation taking into account insertions and deletions of nodes, and also, insertion, deletion and substitution of edges. Moreover, it would be interesting to allow the system to learn the cost functions. Also, virtual edges will be studied to deal with very distorted structures.

ACKNOWLEDGMENT

Work supported by Spanish projects and fellowships: TIN2015-70924-C2-2-R, RYC-2014-16831, FPU15/06264, the CERCA Program/Generalitat de Catalunya and RecerCaixa (XARXES, 2016ACUP-00008), a research program from Obra Social "La Caixa" with the collaboration of the ACUP. We thank NVIDIA for the donation of a Titan Xp GPU.

REFERENCES

[1] D. Conte, P. Foggia, C. Sansone, and M. Vento, "Thirty years of graph matching in pattern recognition," *IJPRAI*, vol. 18, no. 3, pp. 265–298, 2004.

[2] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *ICML*, 2017.

[3] X. Gao, B. Xiao, D. Tao, and X. Li, "A survey of graph edit distance," *PAA*, vol. 13, no. 1, pp. 113–129, 2010.

[4] K. Riesen and H. Bunke, "Approximate graph edit distance computation by means of bipartite graph matching," *IVC*, vol. 27, no. 7, pp. 950 – 959, 2009.

[5] A. Fischer, C. Y. Suen, V. Frinken, K. Riesen, and H. Bunke, "Approximation of graph edit distance based on Hausdorff matching," *PR*, vol. 48, no. 2, pp. 331 – 343, 2015.

[6] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[7] F. Monti, D. Boscaini, J. Masci, E. Rodolà, J. Svoboda, and M. M. Bronstein, "Geometric deep learning on graphs and manifolds using mixture model cnns," *CVPR*, 2017.

[8] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a "siamese" time delay neural network," in *NIPS*, 1994, pp. 737–744.

[9] Z. Abu-Aisheh, B. Gazere, S. Bougleux, J.-Y. Ramel, L. Brun, R. Raveaux, P. Héroux, and S. Adam, "Graph edit distance contest: Results and future challenges," *PRL*, vol. 100, pp. 96 – 103, 2017.

[10] A. Fischer, S. Uchida, V. Frinken, K. Riesen, and H. Bunke, "Improving hausdorff edit distance using structural node context," in *GbRPR*. Springer, 2015, pp. 148–157.

[11] P. Wang, V. Eglin, C. Garcia, C. Largeron, J. Lladós, and A. Fornes, "A novel learning-free word spotting approach based on graph representation," *DAS*, pp. 207–211, 2014.

[12] P. Riba, J. Lladós, and A. Fornés, "Handwritten word spotting by inexact matching of grapheme graphs," in *ICDAR*, 2015, pp. 781–785.

[13] P. Riba, J. Lladós, A. Fornés, and A. Dutta, "Large-scale graph indexing using binary embeddings of node contexts for information spotting in document image databases," *PRL*, vol. 87, pp. 203–211, 2017.

[14] M. Stauffer, A. Fischer, and K. Riesen, "Graph-based keyword spotting in historical handwritten documents," in *S+SSPR*, 2016, pp. 564–573.

[15] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: going beyond euclidean data," *IEEE SPM*, vol. 34, no. 4, pp. 18–42, 2017.

[16] D. K. Duvenaud, D. Maclaurin, J. Iparraguirre, R. Bombarell, T. Hirzel, A. Aspuru-Guzik, and R. P. Adams, "Convolutional networks on graphs for learning molecular fingerprints," in *NIPS*, 2015, pp. 2224–2232.

[17] Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel, "Gated graph sequence neural networks," *ICLR*, 2016.

[18] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE SPM*, vol. 30, no. 3, pp. 83–98, 2013.

[19] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *NIPS*, 2016, pp. 3844–3852.

[20] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *ICLR*, 2017.

[21] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," in *SSST*, 2014.

[22] K. Riesen and H. Bunke, "Iam graph database repository for graph based pattern recognition and machine learning," in *S+SSPR*, 2008, pp. 287–297.

[23] M. Stauffer, A. Fischer, and K. Riesen, "A novel graph database for handwritten word images," in *S+SSPR*, 2016, pp. 553–563.

[24] A. Fischer, A. Keller, V. Frinken, and H. Bunke, "Lexicon-free handwritten word spotting using character hmms," *PRL*, vol. 33, no. 7, pp. 934–942, 2012.

[25] M. Stauffer, A. Fischer, and K. Riesen, "Ensembles for graph-based keyword spotting in historical handwritten documents," in *ICDAR*, 2017, pp. 714–720.

[26] J. Gibert, E. Valveny, and H. Bunke, "Graph embedding in vector spaces by node attribute statistics," *PR*, vol. 45, no. 9, pp. 3072 – 3083, 2012.

[27] A. Dutta, P. Riba, J. Lladós, and A. Fornés, "Pyramidal stochastic graphlet embedding for document pattern classification," in *ICDAR*, 2017, pp. 33–38.

[28] S. K. Ghosh and E. Valveny, "Query by string word spotting based on character bi-gram indexing," in *ICDAR*. IEEE, 2015, pp. 881–885.

[29] L. Rothacker and G. A. Fink, "Segmentation-free query-by-string word spotting with bag-of-features hmms," in *ICDAR*, 2015, pp. 661–665.

[30] U.-V. Marti and H. Bunke, "Using a statistical language model to improve the performance of an hmm-based cursive handwriting recognition system," *IJPRAI*, vol. 15, no. 01, pp. 65–90, 2001.

[31] J. A. Rodriguez and F. Perronnin, "Local gradient histogram features for word spotting in unconstrained handwritten documents," *ICFHR*, pp. 7–12, 2008.

[32] K. Terasawa and Y. Tanaka, "Slit style hog feature for document image word spotting," in *ICDAR*, 2009, pp. 116–120.

[33] B. Wicht, A. Fischer, and J. Hennebert, "Deep learning features for handwritten keyword spotting," in *ICPR*, 2016, pp. 3434–3439.