On the use of GNN-based structural information to improve CNN-based semantic image segmentation

Patty Coupeau, Jean-Baptiste Fasquel, Mickael Dinomais

On the use of GNN-based structural information to improve CNN-based semantic image segmentation

Patty Coupeau[a,*], Jean-Baptiste Fasquel[a] and Mickaël Dinomais[a,b]

[a]Universite d'Angers, LARIS, SFR MATHSTIC, F-49000 Angers, France
[b]Departement de medecine physique et de readaptation, Centre Hospitalier Universitaire d'Angers, France

ARTICLE INFO

Keywords:
image segmentation
structural information
node classification
graph neural network
graph coarsening

ABSTRACT

Convolutional neural networks (CNNs) are widely used for semantic image segmentation across various fields (medicine, robotics), capturing local pixel dependencies for good results. Nevertheless, CNNs struggle to grasp global contextual representations, sometimes leading to structural inconsistencies. Recent approaches aim to broaden their scope using attention mechanisms or deep models, resulting in heavy-weight architectures. To boost CNN performance in semantic segmentation, we propose using a graph neural network (GNN) as a post-processing step. The GNN conducts node classification on appropriately coarsened graphs encoding class probabilities and structural information related to regions segmented by the CNN. The proposal, applicable to any CNN producing a segmentation map, is evaluated on several CNN architectures, using two public datasets (FASSEG and IBSR), with four graph convolution operators. Results reveal performance improvements, enhancing on average the Hausdorff distance by 24.3% on FASSEG and by 74.0% on IBSR. Furthermore, our approach demonstrates resilience to small training datasets.

1. Introduction

Semantic image segmentation is a fundamental task in computer vision for many fields [1]. It involves assigning each pixel in an image to a semantic category or class. Usually, this task is performed using modern deep learning approaches based on convolutional neural networks (CNNs) [2]. CNN apply a sliding window to the input image to capture local information and dependencies between pixels, allowing them to achieve high performance in semantic image segmentation. It has been shown that modeling the global contextual representations can provide richer local and non-local information of context-dependent objects [3], thus improving CNN-based segmentations that may have structural inconsistencies. In fact, CNNs do not explicitly model the spatial structural information available at a higher semantic level (relationships between regions or qualitative description of the scene content). This information is only partially considered by convolutional layers, depending on the size of the receptive field. Some works attempt to consider larger spatial contexts, such as those involving positional encoding (e.g. vision transformers). However, this increases the complexity of the architecture and the number of trainable parameters (requiring more training data), which is a challenging problem [4, 5, 6].

High-level structural information may include spatial relationships between different regions (e.g. distances, relative directional position [7]) or relationships between their properties (e.g. relative brightness, difference of colorimetry [8]). This high-level structural information shows promise in aiding the recognition of global scenes as complementary knowledge in fields such as document analysis [9] and robotics [10]. The high-level information is commonly represented by graphs, where nodes correspond to regions and edges carry the structural information. The challenge is to compare the relationships produced by a CNN that performs semantic segmentation with known relationships, such as those observed in an annotated dataset, to more accurately identify regions of interest. Thus, the semantic segmentation problem can be seen as an inexact graph matching problem, classically formulated as a quadratic assignment problem (QAP) [11, 12]. This approach is limited by its intrinsic highly combinatorial nature [13], although recent works have attempted to reduce the computation time of this type of method [14].

Graph neural networks (GNNs) have emerged as powerful models for capturing global dependencies by leveraging a suitable graph [15, 16]. GNNs have demonstrated very promising performance for various image-level reasoning

*Corresponding author
patty.coupeau@etud.univ-angers.fr (P. Coupeau)
ORCID(s):

tasks when combined with CNNs [17, 18]. To overcome the limitation of inexact graph matching problems, we propose to consider GNNs to learn this matching through a node classification task. The efficiency of GNN in the classification task depends partly on the type of convolution operator considered. Indeed, a wide variety of operators have been developed in the last few years [16], supporting or not n-dimensional edge attributes, capable of carrying multidimensional spatial information. Then, due to the permutation invariant nature of convolution operators during the message passing (neighborhood aggregation), working on complete graphs (resulting from using all relations between all regions) may not be appropriate for distinguishing nodes for classification purposes. Therefore, it seems interesting to integrate an appropriate coarsening strategy during the graph construction.
Besides, thanks to the proposed integration of high-level structural relationships, the proposal aims to improve the semantic segmentation of images provided by CNNs, especially when the size of the training dataset is small. Thus, our work also addresses a key limitation of deep learning: the need for a large and representative dataset for training purposes, which is often addressed by generating more training data using data augmentation [19] or by considering transfer learning [20] or few-shot learning strategies [21].
This paper extends a preliminary work on the relevance of a specific convolutional operator (ECConv) for semantic segmentation [22] and brings the following contributions:

- The proposal is based on a graph neural network working on top of a preliminary semantic segmentation map provided by a deep neural network, as well as on an adaptive coarsening strategy.
- Our method is shown experimentally to perform well on two applications for segmenting 2D and 3D data. Two types of relationships for capturing structural information are considered, illustrating the flexibility of our approach. In comparison to our previous work, the method is coupled with several recent deep learning semantic segmentation networks to show its relevance in different contexts and its advantage over more complex neural networks.
- The influence of the choice of the convolution operator (four are compared) as well as the influence of the considered coarsening level are studied.
- The robustness of our approach is evaluated as the size of the training dataset decreases, which affects the performance of the CNN used as input.

Section 2 gives an overview of some related works for semantic image segmentation. After a description of the proposal in section 3, experiments are detailed in section 4 and discussed in section 5.

2. Related works

2.1. Convolutional neural networks (CNNs)

With the development of computer vision, convolutional neural networks (CNNs), such as U-Net [23], have become the common standard network architecture for semantic image segmentation. CNNs have achieved high performance in various image analysis tasks due to their convolutional layers, which capture local information and spatial dependencies between neighboring pixels. Nevertheless, the contextual information exploited by the CNNs remains limited to the size of the receptive field used during the convolution operation [11] and the long-range pixel dependencies are mainly modeled by deeply stacked convolutional layers. Most existing pure deep CNNs based architectures [23, 24, 25] rely on broad multi-scale and multi-stream CNN frameworks to achieve higher performance. This typically requires more trainable parameters and computational resources. Independently of CNNs, Vision Transformers architectures (ViT) have emerged [26], mainly based on attention mechanisms [27]. ViT can capture larger scale relationships than CNNs, thanks to the decomposition of the image into serialized patches. Nevertheless, the smaller receptive field of CNN provides better stability and accuracy than using ViT alone. Thus, attention-based methods were introduced within the convolutional structures to strengthen the global context modeling ability of CNNs [28, 29] while taking advantage of CNN's strengths. However, these attention-based methods can be heavy-weight, despite efforts to have more light-weight architectures [30].
By combining CNN-based methods with the exploitation of large-scale structural relationships in the image represented in a graph, we aim to improve the performance of CNNs by exploiting global spatial relationships (like transformers), but with a light-weight model and little training data required.

2.2. Combination of CNNs and Graph neural networks (GNNs)

In recent years, there has been a rich line of research on graph neural networks (GNN). Inspired by the first-order graph Laplacian methods, graph convolutional networks (GCNs) have been proposed [31], achieving promising performance on graph node classification tasks in many domains [32, 33]. Research on GCNs, and in particular on their constituent operators (convolution, pooling), has multiplied. As a result, several convolutional neural network (CNN) architectures have been adapted for use with graphs, such as the Graph U-Net architecture [34].

In computer vision, GNNs have been considered for various tasks involving, for instance, point clouds [35], bounding boxes of detected objects [36] and also for semantic segmentation. Several recent methods have emerged, combining GCNs and CNNs, for image segmentation. Some proposals exploited superpixel information for graph construction, assuming additional over-segmentation algorithms, such as SLIC [37] to generate superpixels. For instance, *Ouyang et al.* [17] considered a U-shaped residual attention network to extract high-level features before applying SLIC to extract superpixels from the segmentation. A GCN was used to perform node classification on the graphs constructed from the superpixels. In the same way, *Diao et al.* [18] proposed a superpixel-based Attention Graph Neural Network (SAGNN) for semantic segmentation of high-resolution aerial images. Each node, corresponding to a superpixel obtained with SLIC, was associated with a hidden representation vector being the appearance feature extracted by a CNN. In the medical field, there has been the same tendency to combine GCNs and CNNs by considering superpixels [38]. Other recent proposals have aimed at integrating spatial relationships using GNNs in the feature space of neural networks [39, 40, 41, 42] to capture long-range spatial dependencies. Such approaches require the development of specific architectures. Other work combining CNNs and GCNs has been done for specific applications, such as *Shin et al.* [43] for a binary segmentation of retinal vessels or *Wang et al.* [44] for biomedical image segmentation considering a contour-guided graph reasoning. Compared to all these methods, our proposal focuses on a GNN-based post-processing layer that can be applied to any CNN output producing a segmentation map, for any application, without requiring the development of a specific CNN-based architecture (as in [39, 40, 41, 42]) or the integration of additional over-segmentation algorithms producing superpixels (as in [17, 18]).

2.3. Handling structural information in graphs

The structural information embedded in graphs concerns both the graph structure (edges) and the nature of the edge attributes. Regarding edge attributes, an important aspect to perform node classification is the ability to manage n-dimensional information, as illustrated by [35, 45, 46]. In their work, *Renton et al.* [46] proposed building region adjacency graphs with edge attributes that carry the relative distance between the gravity centers of two adjacent regions. Their aim was to classify symbols on binary images using the edge-conditioned convolution operator [35]. We propose extending this idea to semantic image segmentation. Independent of edge attributes, graph architecture is an important aspect that has been widely studied, for example for node clustering in graph pooling operators [47]. Some works have defined a pooling operator based on edge cuts [48]. It is this last idea of edge cutting that we explore through our graph coarsening strategy, rather than considering node pooling, in order to preserve all nodes for the node classification task in our semantic segmentation context. All these proposals demonstrate the influence of graph structure and edge attributes on node classification, which we propose to study by comparing different convolution operators and levels of graph coarsening.

3. Proposed method

Figure 1 provides an overview of the proposed method. During the training phase (Figure 1-A), a deep neural network (CNN) is trained for semantic image segmentation using an annotated dataset. A segmentation map of each training image is retrieved from the trained CNN. Complete graphs are constructed based on the set of connected components obtained from the segmentation maps of the training data (section 3.1). These complete graphs are used to define statistics describing the spatial relationships between the different regions segmented in the images (average μ and standard deviation σ). For instance, for spatial relationships regarding the distance between the barycentres of regions, we define the average distance and the standard deviation between all the segmented regions in all the images of the training dataset. These statistics are then used to define several ρ coarsening radii for the graphs (section 3.1). A GNN is trained, for each coarsening radius, to classify the graph nodes (Figure 1-A'). The annotated images in the training dataset are used to identify the actual class of the nodes in the graphs (ground truth). The optimal coarsening radius (ρ_i in Figure 1) is defined using the validation data. The associated GNN (trained on training graphs coarsening with a ρ_i radius) is then considered for the inference phase (Figure 1-B).

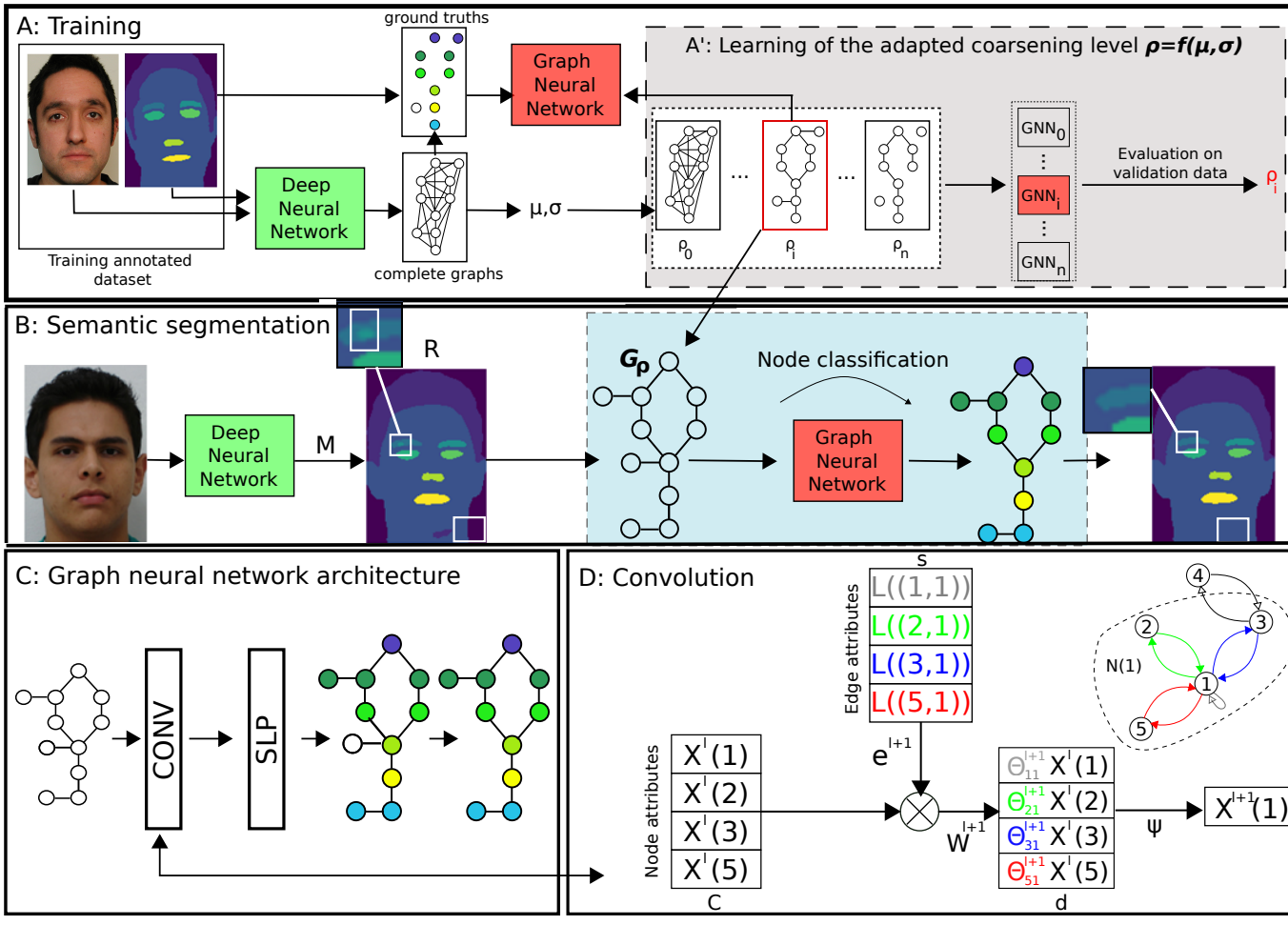

Figure 1: Overview of the proposed method. A: A deep neural network (CNN) is trained for semantic image segmentation using an annotated dataset. A segmentation map of each training image is thus retrieved. Complete graphs are constructed from the segmentation maps. The actual class (ground truth) of the nodes is identify using the annotated images. From the complete graphs of the whole training dataset, the mean (μ) and the standard deviation (σ) of the spatial relations between segmented regions (edge attributes) are defined. Several coarsening radii ρ (function of μ and σ) are considered to train the GNN to perform node classification (A'). The selection of ρ is done using the validation data. B: During inference, the image to be segmented is first passed through the trained CNN, providing a segmentation map (M) that may contain classification errors. A coarsened graph G_ρ (considering the optimal coarsening radius ρ_i) is constructed from the set of related components R. The trained GNN performs a node classification of G_ρ, from which the image regions are assigned to their new class. C: The GNN consists of a single layer perceptron (SLP) preceded by a convolutional layer (CONV). D: Convolution aims at aggregating neighborhood information (node attributes X^l and incoming edge attributes L) related to each node to update its representation.

When analyzing a new image (Figure 1-B), the trained CNN first provides a segmentation proposal, which may contain some mistakes (e.g. hair detected in the chin, eyes in the eyebrow). From this proposal, a coarsened graph is built, considering the optimized parameter ρ_i. The trained graph neural network is then used for node classification. According to this classification, regions are finally identified (each node being associated to an image region in the initial segmentation map).

Section 3.1 explains the construction of the graphs from the images. The strategy of graph coarsening is also introduced. Section 3.2 describes the architecture of the considered GNN (Figure 1-C), emphasizing the crucial role of the convolution layer (Figure 1-D).

3.1. Images and graphs

When segmenting an image, the CNN provides a segmentation map being a tensor $M \in \mathbb{R}^{P \times C}$ with P the dimensions of the image and C the total number of classes. At each pixel location p, $M(p, c) \in [0, 1]$ indicates the probability of pixel p belonging to class c. Based on probabilities, one constructs a set R of all resulting connected components (i.e. set of connected pixels a priori belonging to the same class according to $M(p, c)$, see Figure 1). From this set R, a graph $G = (V, E, X, L)$ is created, with V being the set of nodes (each $v \in V$ corresponds to a region $R_v \in R$) and E the set of edges. X is a function ($X : V \rightarrow \Re^C$) that assigns node attributes based on the average membership probability vector of the set of pixels $p \in R_v$. L is a function ($L : E \rightarrow \Re^s$) that assigns attributes to edges based on the spatial relationship (of dimension s), which is a hyperparameter of the method.

3.1.1. Training

Complete graphs (i.e., where all nodes are connected by an edge) are constructed from the segmentation maps of the training data. The set of complete training graphs $\mathcal{G}_t$ is used to define two parameters (μ and σ) representing the mean and standard deviation of the structural relationships $L((i, j))$ between the segmented regions of the images. μ and σ are defined as follows with $|.|$ denoting the set cardinality:

$$\mu = \frac{1}{|\mathcal{G}_t|} \sum_{G \in \mathcal{G}_t} \frac{1}{|E|} \sum_{i \in V} \sum_{j \in V/\{i\}} L((i, j)) \tag{1}$$

$$\sigma = \sqrt{\frac{1}{|\mathcal{G}_t|} \sum_{G \in \mathcal{G}_t} \frac{1}{|E|} \sum_{i \in V} \sum_{j \in V/\{i\}} (L((i, j)) - \mu)^2} \tag{2}$$

The parameters μ and σ define several coarsening radii $\rho = f(\mu, \sigma)$ that reduce the graph $G = (V, E, X, L)$ to a graph $G_\rho = (V, E_\rho, X, L)$ so that $E_\rho = \{e \in E \mid \|L(e)\| < \rho\}$, $\|.\|$ denoting the Euclidean norm. Coarsening plays an important role in the convolution operation (notion of message passing - see section 3.2), as it alters the number of neighbors for each node in the graph. Coarsening provides additional structural information by connecting nodes i and j only if their spatial relationship $L((i, j))$ is small enough, which enforces a focus on the local neighborhood through message passing. This means that structural information is carried by both E_ρ and L.
The optimal coarsening radius ρ_i is selected based on the node classification performance of the graph neural networks associated with each coarsening level on the validation data.

3.1.2. Inference

For each test image, a graph $G_{\rho_i} = (V, E_{\rho_i}, X, L)$ is constructed based on the level of coarsening ρ_i determined using the training dataset. The graph is generated from the set of connected components R derived from the segmentation map M provided by the CNN.

3.2. Graph neural network

As illustrated by Figure 1-C, the proposed GNN contains only two layers: the first one focuses on convolution and the second one assigns a membership probability vector to each node. The number of nodes to be classified is a priori unknown (i.e. the number of candidate regions built by a CNN) and greater than the number of classes (realistic hypothesis of over-segmentation) [8]. Therefore, graphs have arbitrary sizes. Consequently, one considers for graph convolution a spatial-based approach rather than a spectral one (spectral graph theory) [15].
The first layer consists of a convolution ($\Re^C \rightarrow \Re^d$) aimed at aggregating neighborhood information related to each node (general notion of message passing [15]). The hyperparameter d defines the dimension of node attributes at the output of the convolution (Figure 1-D). It is empirically defined in our experiments. Since graphs are non-positional and non-ordered, the aggregation of neighborhood information requires the use of a permutation-invariant function ψ (e.g. average, max, sum, etc.). Message passing can be described as mentioned in [15]:

$$X^{l+1}(i) = W^{l+1}(X^l(i), \psi(W^{l+1}X^l(j) | j \in N(i))) \tag{3}$$

$N(i) = \{j|(i,j) \in E_\rho\}$ and W^{l+1} is a learned weight matrix corresponding to a linear transformation. This formulation of message passing does not consider edge attributes. Thus, it is possible to expand Eq.3 to include edge attributes explicitly [15]:

$$X^{l+1}(i) = W^{l+1}(X^l(i), \psi(W^{l+1}e^{l+1}(L((j,i)))X^l(j)|j \in N(i))) \quad (4)$$

with $e^{l+1}(L(j,i))$ being a neural network based on edge attributes. This formulation, illustrated in Figure 1-D, can be simplified as:

$$X^{l+1}(i) = W^{l+1}(X^l(i), \psi(\Theta^{l+1}_{j,i}X^l(j)|j \in N(i))) \quad (5)$$

Coarsening (see section 3.1) plays an important role in this convolution operation, as it will modify the neighborhood $N(i)$ of each node i. This has an impact on the aggregated information (parameters $X^l(j)$ and $L((j,i))$) in Eq.5. In the example of Figure 1 - D, node 1 is not connected to node 4 due to coarsening. The new state $X^{l+1}(1)$ of node 1 is calculated based on its own state $X^l(1)$, the attributes of nodes 2, 3 and 5 (node 1 does not exploit the information carried by node 4) and the attributes of the edges (1,2), (1,3) and (1,5) (ignoring the structural relationships of edge (1,4)). By varying the coarsening level, we define a limited and local neighborhood. This way, we expect to facilitate node discrimination and therefore classification.

In the experiments, various graph convolution operators are compared (GCNConv [31], GraphConv [49], ECConv [35] and GATConv [50]). The nature of the operator, which verifies Eq.5, can be seen as a hyperparameter of the method. This work considers a single convolution layer. It is worth noting that multiple convolution layers can be cascaded. However, as underlined by [34], GCNs tend to perform better with a shallow architecture (less than 4 convolutional layers).

The second layer is a single layer perceptron ($\mathfrak{R}^d \rightarrow \mathfrak{R}^C$) that provides a class membership probability vector for each node of the graph.

The network parameters are optimized over the training dataset to maximize the node classification rate. This corresponds to minimizing the negative log-likelihood loss function: $\text{Loss}(Y, \hat{y}) = -\sum_{n=1}^{N}\sum_{c=1}^{C} Y_{n,c} \times \log(\hat{y}_{n,c})$ with N the number of nodes, C the number of classes, $Y_{n,c}$ the actual class of node n (1 if node n belongs to class c, 0 otherwise) and $\hat{y}_{n,c}$ the probability that node n belongs to class c. Based on the node classification proposed by the GNN, the pixels of the region associated with each node are set to the value of the assigned class. The tensor map M, which is the output of the CNN, is thus updated.

4. Experiments

To assess the relevance of our method, we compare it to a state-of-the-art method that also aims to improve CNN segmentation performance using graphs. However, this approach is based on a quadratic assignment problem solved with an inexact graph matching approach [14]. To compare our work, we conduct our experiments using the same dataset, dealing with the semantic segmentation of 2D images of faces (FASSEG-Instances public dataset[1]) and of internal brain structures on 3D medical images (IBSR public dataset[2]). In our experiments, we consider several CNNs (hereafter detailed and similar to [14]), selected from recent works with available implementations, either specialized for 2D color images or for 3D medical images.

Section 4.1 describes the evaluation protocol, including the GNN training, the different graph convolution operators compared, the evaluation metrics, the configuration of the dataset as well as the different coarsening levels considered. Sections 4.2 and 4.3 detail experiments on the FASSEG and IBSR datasets, respectively. In each case, we first describe the dataset, the CNNs considered, the graph construction and the results obtained. Section 4.4 studies the influence of graph coarsening.

4.1. Protocol

4.1.1. Training

All experiments are performed in a Python environment using the *PyTorch Geometric*[3] library. The CNN training is detailed for each dataset (sections 4.2 and 4.3) because it varies depending on the nature of the CNN and the application.

[1] https://github.com/Jeremy-Chopin/FASSEG-instances

[2] https://www.nitrc.org/projects/ibsr

[3] https://pytorch-geometric.readthedocs.io/en/latest/

The GNN model is trained with Adam on 600 epochs, with a dropout of 0.5, considering the loss function mentioned in section 3.2. A strategy of reduction of the learning rate on plateau is used with an initial learning rate $L_{r_0} = 0.01$ and a reduction factor $\sigma = 5e^{-4}$.

4.1.2. Graph convolution operators

To investigate the impact of the graph convolution operator, we compare four state-of-the-art operators (all are spatial methods applicable to arbitrary size graphs [15]). These operators manage attributes on edges for the message passing in different ways. Two of them consider the attributes on edges as a scalar weight w (GCNConv[31] and GraphConv[49]). We set w to 1 to determine if edge attributes bring an added value in the node prediction. The ECConv[35] and GATConv[50] graph convolution operators combine node and edge attributes for the neighborhood aggregation, allowing to manage multidimensional structural information.

GCNConv [31], efficient for semi-supervised node classification, relies on node attributes and local graph structure (resulting from coarsening) without considering multidimensional edge attributes L. The general formulation of the message passing (Eq.5) can be adapted as follows:

$$X^{l+1}(i) = W^{l+1} \sum_{j \in N(i) \cup \{i\}} \frac{w_{j,i}}{\sqrt{\hat{d}_j \hat{d}_i}} X^l(j) \tag{6}$$

where $\hat{d}_i = 1 + \sum_{j \in N(i)} w_{j,i}$ with $w_{j,i}$ being the edge weight from source node j to target node i (set to 1 in our case, as explained previously). By analogy with Eq.5, ψ is the sum and $\Theta^{l+1}_{j,i} = \frac{W^{l+1}}{\sqrt{\hat{d}_j \hat{d}_i}}$.

In the same way, GraphConv [49] only considers node attributes and the local graph structure. Its difference lies in the distinction of the weight of the source node i (W^{l+1}_1):

$$X^{l+1}(i) = W^{l+1}_1 X^l(i) + W^{l+1}_2 \sum_{j \in N(i)} w_{j,i} X^l(j) \tag{7}$$

For GraphConv, ψ is a sum and $\Theta^{l+1}_{j,i} = W^{l+1}_2$ since $w(j,i) = 1$.

ECConv [35], manages both n-dimensional attributed node and edge information as shown in [22, 46]. Its message passing can be formulated as:

$$X^{l+1}(i) = \frac{1}{|N(i)|} \sum_{j \in N(i) \cup i} W^{l+1} F^{l+1}(L((j,i))) X^l(j) \tag{8}$$

F^{l+1} is a differentiable function (a multi-layer perceptron in our case) learned by training. Thus, by analogy with Eq.5, ψ is the average function and $\Theta^{l+1}_{j,i} = W^{l+1} F^{l+1}(L((j,i)))$, underlying the use of information embedded by edges.

GATConv is an attention-based architecture capable of performing node classification [50], considering both node and edge attributes in its computation:

$$X^{l+1}(i) = \sum_{j \in N(i) \cup \{i\}} \alpha^{l+1}_{i,j} W^{l+1} X^l(j) \tag{9}$$

where $\alpha^{l+1}_{i,j}$ are the attention coefficients computed as:

$$\alpha^{l+1}_{i,j} = \frac{e^{a(W^{l+1}X^l(i), W^{l+1}X^l(j), W^{l+1}_e L((j,i)))}}{\sum_{k \in N(i) \cup \{i\}} e^{a(W^{l+1}X^l(i), W^{l+1}X^l(k), W^{l+1}L((k,i)))}} \tag{10}$$

a being the attention mechanism and W^{l+1}_e a learned weight matrix specific to edge attributes. By analogy with Eq.5, ψ is a sum and $\Theta^{l+1}_{j,i} = \alpha^{l+1}_{i,j} W^{l+1}$.

For FASSEG, the output dimension d of the graph convolution operators is set to 7, while for IBSR it is set to 22.

Table 1
Size and distribution of the dataset in both experiments

	dataset size	train (validation)			test
Configuration		A	B	C	
FASSEG	70	15 (5)	10 (3)	5 (2)	50
IBSR	18	8 (4)	6 (3)	4 (2)	6

4.1.3. Evaluation metrics
To assess the segmentation quality, we compute the Dice score (DSC) [51] and Hausdorff distance (HD) [52] for each class. Compared to the Dice, the HD emphasizes the case of a segmented region that would depict small connected components that are far away from the reference annotated region. Additionally, we compare the results from the GNN output to those obtained from the CNN alone.

4.1.4. Data configurations
To assess the robustness of our method with respect to small datasets, we consider different sizes of the training dataset for both CNNs and GNNs in both applications. We work with the three configurations A, B and C presented in Table 1. For each training dataset size (configurations A, B and C), the set of images used for testing purpose is always the same. The dataset was decomposed using the approach presented in a recent work [14] to facilitate comparison.
A cross-validation is performed for each configuration. For each configuration, three random draws of training and validation data are carried out, in accordance with the distribution in Table 1, among the images in the training dataset (20 images for FASSEG, 12 images for IBSR). A CNN is trained for each random draw. The presented results are averaged over the three random draws.

4.1.5. Levels of graph coarsening
For each of the studied configurations (A, B and C), each convolution operator, and each CNN compared, six coarsening radii are considered during the optimization process (complete graphs, $\mu-0.5\sigma$, $\mu-\sigma$, $\mu-1.2\sigma$, $\mu-1.5\sigma$ and $\mu-1.8\sigma$) with μ and σ defined in Eq.1 and 2. The highest coarsening level ($\mu - 1.8\sigma$) often corresponds to a graph with no edges. It is important to note that the *torch-geometric* implementation used in experimentation does not handle this case. Therefore, we consider the GNN to be comprised of a single SLP.

4.2. Experiments on FASSEG
4.2.1. Dataset
The FASSEG public dataset[4] focuses on the multi-class semantic segmentation of the face [53]. For this study, we have considered a subset of this dataset that corresponds to a specific pose (the frontal view). This subset contains 70 images with manual semantic segmentation. The annotations were refined to ensure a unique region per label, resulting in the FASSEG-Instances public dataset. For the sake of simplicity, the term FASSEG is used throughout the rest of the paper. The dataset includes nine classes: background, hair, face, left eyebrow, right eyebrow, left eye, right eye, nose and mouth.

4.2.2. CNNs
For this dataset, four deep learning networks with a code available online are considered: U-Net [23], U-Net combined with CRF as post-processing [54], PSPNet [24] and EfficientNet [4]. Concerning the widely used U-Net, a Pytorch implementation[5] is used. The network is trained over 200 epochs with an early stopping strategy to prevent over-fitting. For the CRF-based post-processing, acting as a spatial regularization technique, a Gaussian filter of size 11 is applied. The CRF model is placed at the output of the pretrained U-Net and trained (over 50 epochs) to fine-tune the whole model.
PSPNet captures information at different scales through a pyramid pooling module, corresponding to various ranges of spatial relationships between subregions. The hyperparameters considered are those used in the original study [24], which involves using a pretrained ResNet for the initial feature map computation followed by a four-level pyramid pooling.

[4] https://github.com/massimomauro/FASSEG-repository
[5] https://doi.org/10.5281/zenodo.3522306

Table 2
FASSEG - Number of nodes for the considered deep learning architectures (average values over the 3 random selections of the training dataset) and for the different training configurations.

	A	B	C
U-Net	12.0	13.2	13.9
U-Net + CRF	12.1	12.7	14.7
EfficientNet	11.3	11.7	14.6
PSPNet	11.9	12.2	17.5

For the EfficientNet, based on a compound scaling method, we focus on the EfficientNet-B3 (pretrained on ImageNet) with the hyperparameters set as in [4]. In all cases, a median filter is applied to remove tiny artifacts from the segmentation map provided by the CNN.

4.2.3. Graph construction

A graph G_ρ is constructed as detailed in section 3.1. Connected components are extracted from the segmentation map of the CNN using 8-connectivity. Only components larger than 40 pixels are associated with a node (smaller components being assigned to the class given by the CNN) to reduce the total number of nodes. To handle the irregular shape of the regions, especially the hair whose barycentre can be located in the middle of the face (i.e., outside the hair class), spatial relationships considered for edge attributes correspond to the minimum ($d_{min}^{R_i,R_j} = \min_{a\in R_i, b\in R_j} |a-b|$) and maximum distance ($d_{max}^{R_i,R_j} = \max_{a\in R_i, b\in R_j} |a-b|$) between regions R_i and R_j normalized by the largest distance D in the image: $L((i,j)) = [\frac{d_{min}^{R_i,R_j}}{D}, \frac{d_{max}^{R_i,R_j}}{D}]$. Thus, each node has an attribute of dimension 9 (corresponding to the CNN-based probability of belonging to the 9 classes) and the edge attributes are of dimension 2.

4.2.4. Results

Table 2 reports the graph sizes involved for the different deep learning architectures and training dataset configurations. We see that we face arbitrary graph size. It is also noted that the number of nodes increases as the training dataset size decreases since the neural network is less efficient, involving more region candidates per class.

Table 3 compares the performance of segmentation with and without the proposed method on FASSEG. The results obtained using the four convolution operators are reported. It can be observed that the CNN performance degrades as the number of training data decreases (last row of Figure 2 with many segmentation errors). Nevertheless, whatever the CNN and training configuration, our proposal improves the results (for most graph convolution operators) while adding a negligible number of parameters compared to the CNN ones (Table 4). Note that the EfficientNet architecture is more complex than the U-Net, leading to a better Dice score but at the cost of more trainable parameters. PSPNet appears more efficient than U-Net and similar to EfficientNet, but with fewer parameters. This highlights the importance of capturing different pieces of information at different scales. Improvement with GNN is particularly significant in terms of HD (up to 50% according to Table 3 - PSPNet (C)), but negligible in terms of Dice score. This indicates that segmentation errors consist of small regions, relatively far away from the target region (hence significantly altering the HD). The benefit of the method is illustrated in Figure 2 where the GNN eliminates most artifacts and corrects classification errors made by the CNN. The performance of the proposal on configurations B and C demonstrates its robustness despite the lack of representativeness. In fact, our method, when combined with PSPNet, outperforms with a small training dataset (configuration B) a more complex architecture like EfficientNet trained with more data (configuration A) (HD of 19.94 vs 22.10 according to Table 3).

The four convolution operators improve segmentation in terms of HD. However, the GNN based on GCNConv is less efficient and tends to degrade the Dice. The first line of Figure 2 illustrates errors that can occur with this operator. This may be explained by the ignorance of edge attributes and by a too small number of parameters (only 142 according to Table 4) to learn all the variations between faces. The performance of the three other operators is equivalent, although GATConv provides the smallest HD on average (first two lines of Figure 2) despite having a few parameters. The higher complexity of ECConv (Table 4) allows it to better correct spatial inconsistencies when the number of regions is the highest (Configuration C - PSPNet according to Table 2), as illustrated in Figure 2-C. Overall, as the number of nodes

Table 3
Comparison of segmentation on FASSEG using CNN only and the proposed combination of CNN and GNN. Results with the 4 graph convolution operators are reported (EC for ECConv, GCN for GCNConv, Graph for GraphConv and GAT for GATConv). DSC: Dice score, HD: Hausdorff distance.

CNN	Configuration A									
	DSC					HD (improvement in percentage)				
	CNN	EC	GCN	Graph	GAT	CNN	EC	GCN	Graph	GAT
U-Net	0.70	0.70	0.70	0.70	0.70	37.40	30.79 (18%)	28.80 (23%)	29.88 (20%)	**28.78** (23%)
U-Net+CRF	0.71	0.71	0.71	0.71	0.71	35.39	27.29 (23%)	**26.24** (26%)	27.96 (21%)	**26.24** (26%)
EfficientNet	0.84	0.84	0.82	0.84	0.84	22.10	19.01 (14%)	20.05 (9%)	18.29 (17%)	**16.75** (24%)
PSPNet	0.83	0.83	0.83	0.83	0.83	24.18	20.16 (17%)	19.62 (19%)	19.49 (19%)	**18.30** (24%)
Average	0.77	0.77	0.76	0.77	0.77	29.76	24.31 (18%)	23.68 (20%)	23.90 (20%)	**22.52 (24%)**
CNN	**Configuration B**									
	DSC					**HD (improvement in percentage)**				
	CNN	EC	GCN	Graph	GAT	CNN	EC	GCN	Graph	GAT
U-Net	0.72	**0.73**	0.71	0.72	0.72	43.02	**29.23** (32%)	32.87 (24%)	30.71 (29%)	30.72 (29%)
U-Net+CRF	0.70	0.70	0.68	0.70	0.70	39.88	30.92 (22%)	36.53 (8%)	31.00 (22%)	**30.60** (23%)
EfficientNet	0.83	0.83	0.82	0.83	0.83	26.39	19.93 (24%)	22.19 (16%)	21.11 (20%)	**18.77** (29%)
PSPNet	0.82	0.82	0.81	**0.83**	0.82	28.05	20.86 (26%)	21.17 (24%)	**19.94** (29%)	20.16 (28%)
Average	0.77	0.77	0.75	0.77	0.77	34.33	25.23 (26%)	28.19 (18%)	25.69 (25%)	**25.06 (27%)**
CNN	**Configuration C**									
	DSC					**HD (improvement in percentage)**				
	CNN	EC	GCN	Graph	GAT	CNN	EC	GCN	Graph	GAT
U-Net	0.71	0.71	0.69	0.71	0.71	54.25	41.12 (24%)	42.98 (21%)	41.57 (23%)	**36.61** (32%)
U-Net+CRF	0.69	0.68	0.64	0.69	0.67	60.00	45.26 (24%)	55.44 (8%)	42.71 (29%)	**41.88** (30%)
EfficientNet	0.81	0.81	0.74	0.81	0.81	49.80	28.54 (43%)	43.30 (13%)	27.90 (44%)	**26.39** (47%)
PSPNet	0.75	**0.78**	0.72	0.76	0.76	81.51	**40.66** (50%)	55.47 (32%)	48.52 (40%)	45.77 (44%)
Average	0.74	**0.75**	0.70	0.74	0.74	61.39	38.90 (37%)	49.30 (20%)	40.17 (34%)	**37.66 (39%)**

Table 4
Number of trainable parameters for each architecture used for the FASSEG dataset. The parameters of the GNN architecture also encompass those of SLP.

CNN Architecture	# parameters	GNN Architecture	# parameters
U-Net	7 865 529	ECConv	387
U-Net + CRF	7 865 532	GCNConv	142
EfficientNet	17 779 713	GraphConv	241
PSPNet	3 849 420	GATConv	184

becomes larger (Configuration C), convolution operators that consider both node and edge attributes (i.e., ECConv and GATConv) seem to be more adapted to correct the segmentation errors (Table 3).
Table 5 compares the segmentation results obtained using our method for the different CNNs and training set configurations to the recent analogous work based on inexact graph matching [14]. Our method improves the Hausdorff distance in all cases except for PSP-Net in configuration C. However, our approach does not show any significant improvement in terms of Dice, except for PSP-Net in configuration C where it improves by 2%.

4.3. Experiments on IBSR

4.3.1. Dataset

The IBSR public dataset provides 18 3D MRI scans of the brain, with manual segmentation of 32 regions. In our experiments, similarly to [55], only 15 classes of the annotated dataset are considered: background, thalamus, caudate, putamen, pallidum, hippocampus, amygdala and accumbens (left and right for all structures).

4.3.2. CNNs

For this dataset, we consider three neural network architectures for segmentation proposal : a 3D U-Net neural network [56], a 3D U-Net combined with CRF [54], and the recent UNETr based on transformers [29]. A specificity of this

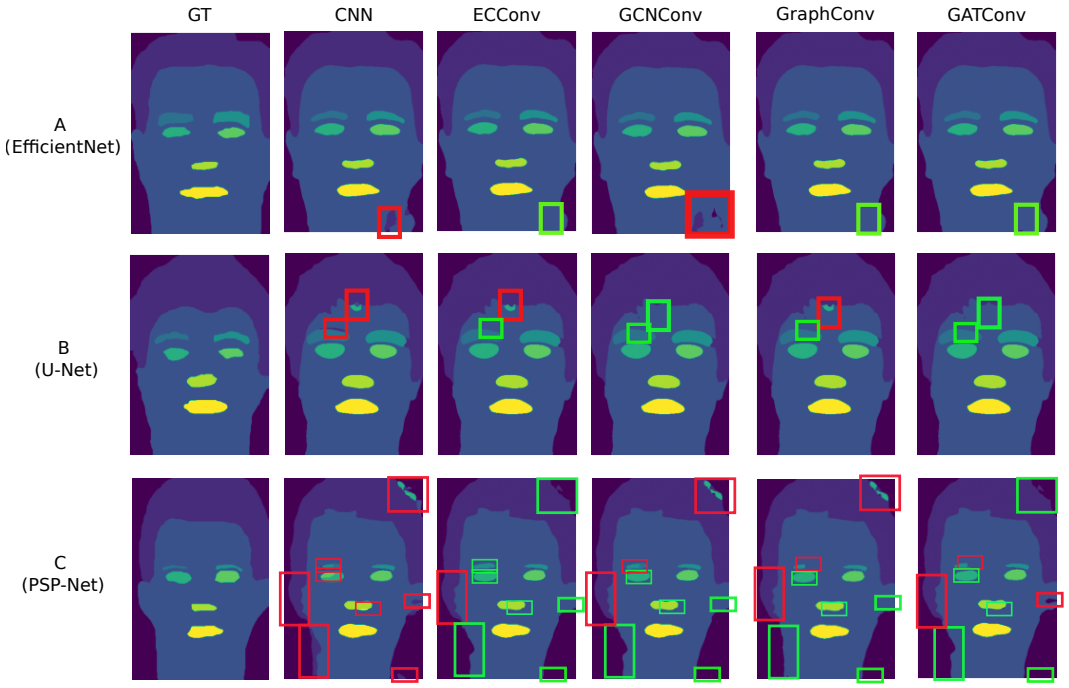

Figure 2: Examples of segmentation on FASSEG with and without GNN. Comparison of the 4 convolution operators is provided. Bounding boxes highlight regions with significant improvements. A, B and C correspond to the considered training configurations.

Table 5
Comparison of the proposal on FASSEG with SOTA method [14] for the different CNNs and configurations considered. Results with the best graph convolution operator are presented.

CNN	Configuration A				Configuration B				Configuration C			
	[14]		Proposal		[14]		Proposal		[14]		Proposal	
	Dice	HD	Dice	HD	Dice	HD	Dice	HD	Dice	HD	Dice	HD
U-Net	0.7	35.79	0.7	**28.78**	0.73	32.59	0.73	**29.23**	**0.72**	37.55	0.71	**36.61**
U-Net + CRF	0.71	34.07	0.71	**26.24**	**0.71**	34.48	0.70	**30.60**	0.69	43.99	0.69	**42.71**
EfficientNet	0.84	20.48	0.84	**16.75**	0.83	22.64	0.83	**18.77**	0.81	29.13	0.81	**26.39**
PSPNet	0.83	20.44	0.83	**18.30**	0.82	20.96	**0.83**	**19.94**	0.76	**33.07**	**0.78**	40.66

dataset is the size of the medical images and the highly unbalanced classes, which require specific strategies such as the use of patches [57]. The 3D U-Net neural network is based on the generic implementation considered for FASSEG, including a comparison with a CRF-based post-processing (using the same Gaussian filter size as for FASSEG). The network is trained over 60 epochs and an early stopping strategy is applied to prevent over-fitting. A 3D patch-based approach [58] is considered due to the high level of class imbalance (i.e. small size of target regions with respect to other brain tissues and background). Patches of 32^3 voxels in size are extracted around the centroid of each label (random selection) using the Torchio library[6]. For each MRI image, 64 patches are selected with a frequency proportional to the inverse prior probability of the corresponding class.

We also consider the recent UNETr designed for medical image segmentation. This network, based on the attention mechanism, models long-range (spatial) dependencies and captures global context, which is not the case with standard CNNs that are limited to localized receptive fields. In our experiments, the hyperparameters have been set as in [29] with patches of size 16^3.

[6]https://torchio.readthedocs.io/

Table 6
IBSR - Number of nodes for the deep learning architectures considered (average values over the 3 random selections of the training dataset) and for the different training configurations.

	A	B	C
U-Net	20.6	23.9	23.6
U-Net + CRF	22.4	22.8	28.3
UNETr	15.4	15.6	21.3

4.3.3. Graph construction

A graph G_ρ is constructed as detailed in section 3.1. Connected components are extracted from the segmentation map of the CNN using 26-connectivity. Only components larger than 40 voxels are associated to a node (smaller connected components being assigned to the class given by the CNN) to reduce the number of nodes. Spatial relationships considered for edge attributes correspond to the relative positions of the barycentres of connected regions R_i and R_j over the 3 dimensions: $L((i,j)) = [d_x^{R_i,R_j}, d_y^{R_i,R_j}, d_z^{R_i,R_j}]$. Thus, each node has an attribute of dimension 15 (corresponding to the 15 classes) and the edge attributes are of dimension 3.

4.3.4. Results

Table 6 reports involved graph sizes for the various considered neural network architectures and configurations of the training dataset. As for FASSEG, we face arbitrary graph size and the number of nodes increases as the training dataset size decreases.

Table 7 compares the segmentation results with and without the proposed method on IBSR. The performance of deep learning-based segmentation degrades with the reduction of the dataset size. It is noteworthy that for configuration C using UNETr, the results are much lower than those obtained by considering a larger dataset or a classical U-Net (Dice of 0.67 vs 0.77 with U-Net+CRF). This highlights the need for a larger training dataset for transformer-based networks compared to traditional CNNs, due to the significantly larger number of trainable parameters (Table 8). However, for configurations A and B, the attention mechanism of the UNETr, which considers partly spatial information to guide segmentation, yields better results in terms of Hausdorff distance than the classic U-Net (HD divided by 2). This is also illustrated in Figure 3 (detailed later) where we observe that U-Net produces more artifacts (surrounding nodes - first line) than UNETr (second line).

Regardless of the training configuration and the CNN, there is at least one graph convolution operator that improves the results obtained from the segmentation proposal. As for FASSEG, the improvement is significant in terms of Hausdorff distance, with an average reduction of almost 75% for all configurations according to Table 7 (e.g., from 20.08 to 5.09 with configuration B). This underlines the relevance of our approach in correcting structural errors. Even with a transformer-based network like UNETr, which provides fewer artifacts due to its attention mechanism (Figure 3), our proposal significantly reduces the HD from 11.01 to 5.27 (reduction of 52%) for the training configuration A, from 13.77 to 5.82 (reduction of 58%) for the training configuration B and from 25.75 to 9.69 (reduction of 62%) for the training configuration C. This demonstrates our method's ability to compensate for the lack of representativity in small training datasets: relationships observed and learned during training can correct structural mistakes achieved by a CNN. In this way, our method achieves better results with less data than a complex deep learning network like UNETr trained with more data, this being a challenging issue [6]. Our proposal, which adds at most 1995 parameters, combined with U-Net, reaches better performance with a small dataset (configuration C) than UNETr alone trained with more data (configuration A) (HD of 5.04 vs 11.01, Dice of 0.78 vs 0.75 according to Table 7). This illustrates the importance of considering high-level relationships on the top of a CNN-based segmentation map, not only in terms of segmentation efficiency but also in terms of the number of trainable parameters.

ECConv and GraphConv are the best operators, able to significantly improve the CNN segmentation in all configurations (in terms of Dice and HD - Table 7). This can be explained by their larger number of trainable parameters (Table 9), which provides them with greater degrees of freedom. The efficiency of GraphConv shows that multi-dimensional edge attributes are not necessary for this case (unlike FASSEG). For this application, the spatial information provided by coarsening (Section 4.4) seems sufficient. Note that, in comparison to a recent study on this dataset [55], but involving

Table 7
Comparison of segmentation on IBSR with CNN only and with the proposed combination of CNN and GNN. Results with the 4 graph convolution operators are reported (EC for ECConv, GCN for GCNConv, Graph for GraphConv and GAT for GATConv). DSC: Dice score, HD: Hausdorff distance

CNN	Configuration A									
	DSC					HD (improvement in percentage)				
	CNN	EC	GCN	Graph	GAT	CNN	EC	GCN	Graph	GAT
U-Net	0.80	**0.81**	0.64	**0.81**	0.79	22.51	**5.73** (74%)	57.36 (-155%)	5.74 (74%)	8.83 (61%)
U-Net+CRF	0.81	**0.82**	0.74	**0.82**	0.80	25.77	6.60 (74%)	29.06 (-13%)	**4.40** (83%)	11.74 (54%)
UNETr	0.75	**0.76**	0.57	**0.76**	0.73	11.01	**5.27** (52%)	62.95 (-472%)	**5.27** (52%)	13.55 (-23%)
Average	0.79	**0.80**	0.65	**0.80**	0.77	19.76	5.87 (70%)	49.79 (-152%)	**5.14 (74%)**	11.37 (42%)
CNN	**Configuration B**									
	DSC					**HD (improvement in percentage)**				
	CNN	EC	GCN	Graph	GAT	CNN	EC	GCN	Graph	GAT
U-Net	0.79	**0.81**	0.70	**0.81**	0.76	22.99	**4.72** (79%)	41.08 (-79%)	**4.72** (79%)	17.08 (26%)
U-Net+CRF	0.79	**0.81**	0.74	**0.81**	0.77	23.49	5.69 (76%)	22.84 (3%)	**4.53** (81%)	18.88 (20%)
UNETr	0.73	0.73	0.58	0.73	0.73	13.77	**5.82** (58%)	59.75 (-334%)	6.03 (56%)	7.19 (48%)
Average	0.77	**0.78**	0.67	**0.78**	0.75	20.08	5.41 (73%)	41.22 (-105%)	**5.09 (75%)**	14.38 (28%)
CNN	**Configuration C**									
	DSC					**HD (improvement in percentage)**				
	CNN	EC	GCN	Graph	GAT	CNN	EC	GCN	Graph	GAT
U-Net	0.76	**0.78**	0.70	**0.78**	0.71	26.94	6.76 (75%)	26.88 (0%)	**5.04** (81%)	28.78 (-7%)
U-Net+CRF	0.77	0.79	0.72	**0.80**	0.69	28.50	8.47 (70%)	24.59 (14%)	**5.15** (82%)	33.33 (-17%)
UNETr	0.67	0.67	0.47	0.67	0.62	25.75	**9.69** (62%)	76.89 (-199%)	12.26 (52%)	20.65 (20%)
Average	0.73	**0.75**	0.63	**0.75**	0.67	27.06	8.31 (69%)	42.79 (-58%)	**7.49 (72%)**	27.59 (-2%)

Table 8
Number of trainable parameters for each CNN architecture used for IBSR.

CNN Arch.	# parameters
U-Net	15 711 887
U-Net + CRF	15 711 891
UNETr	94 197 951

Table 9
Number of trainable parameters for each GNN architecture used fo

GNN Arch.	# parameters
ECConv	1995
GCNConv	697
GraphConv	1027
GATConv	829

an atlas, our approach results in slightly lower Dice score (0.82 versus 0.84 in [55]) but slightly better HD (4.40 versus 4.49 in [55]).

Figure 3 allows a clearer visualization of the corrections provided by the GNN. Thus, in the first example (first line), the GNN correctly classifies the artifacts produced by U-Net (boxed in red) as background (in black). It is possible that these artifacts are discarded because the corresponding nodes are not connected to any other node of the graph due to coarsening. In the second example, UNETr made some classification errors (boxed in red). The GNN corrects the pink node in cyan and removes the incorrect red structure on the right. It also detects an anomaly related to the cyan node on the right side, but misclassifies it as background. This illustrates the ability of our method to identify and correct structural anomalies, although errors may persist in some cases (e.g. the green node at the bottom right).

On Figure 4-3rd row, we see that all operators manage to correct the artifacts provided by U-Net with CRF. ECConv and GraphConv succeed in correcting the misclassifications made by the CNN in all configurations. Nevertheless, on the first two rows, we can see that when a class is not detected by the CNN (such as the left hippocampus boxed in red), GNN is not able to recover it (one red box remains with ECConv and GraphConv). In contrast to the application on FASSEG, GCNConv and GATConv seem to have more difficulty in correctly classifying the nodes. This can be explained by the larger size of the graphs involved (more than 20 nodes on average according to Table 6 compared to less than 15 for FASSEG (Table 2)) and their smaller number of parameters (Table 9). Indeed, they tend to lose information leading to spatial inconsistencies and make classes disappear altogether (Figure 4 e.g. GCNConv replaces the putamen with the pallidum - 1st and 2nd row, GATConv replaces the pallidum with the hippocampus - 3rd row), which greatly degrades the results as reported in Table 7.

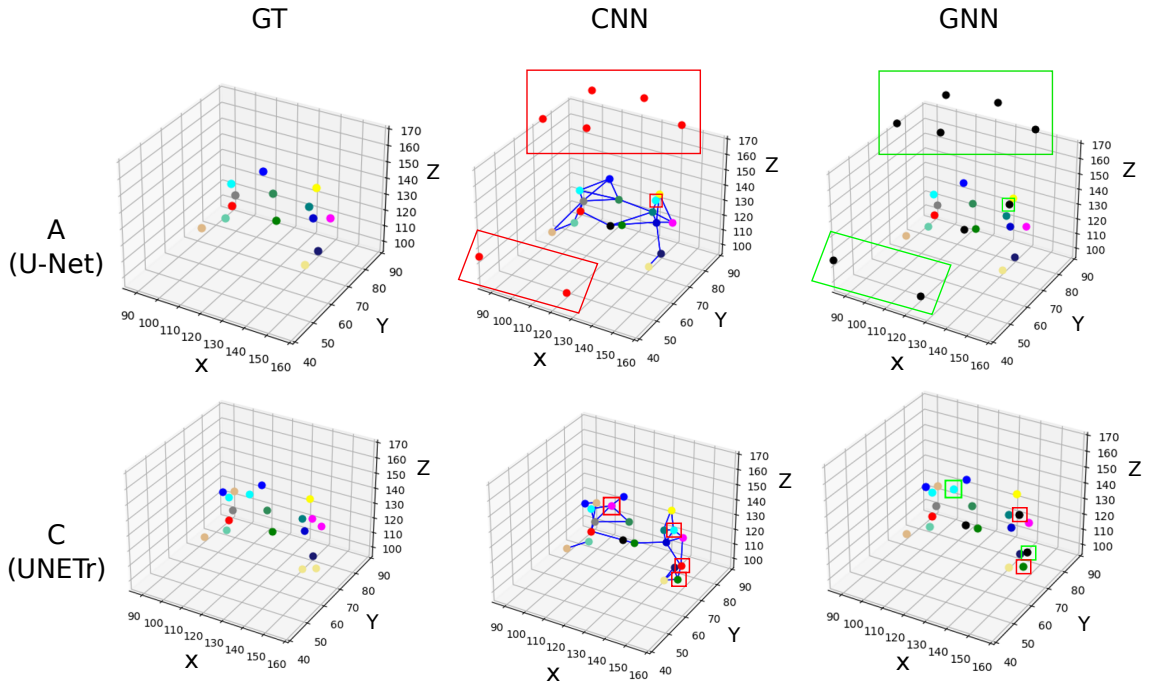

Figure 3: Nodes of the graphs built from the segmentations on IBSR (3D position of related region barycentres). From left to right: real class of the nodes (ground truth - GT), classes assigned at the output of the CNN and edges taken into account for the GNN (graph coarsening), classes assigned at the output of the GNN. Two examples of two configurations are given. Each color corresponds to a brain structure, black nodes correspond to the background. A and C correspond to the considered training configurations. X: right-left axis, Y: anterior-posterior axis, Z: inferior-superior axis of the brain.

Table 10
Comparison of the proposal on IBSR with SOTA method [14] for the different CNNs and configurations considered. Results with the best graph convolution operator are presented.

CNN	Configuration A				Configuration B				Configuration C			
	[14]		Proposal		[14]		Proposal		[14]		Proposal	
	Dice	HD	Dice	HD	Dice	HD	Dice	HD	Dice	HD	Dice	HD
U-Net	**0.82**	**4.61**	0.81	5.73	0.81	4.74	0.81	**4.53**	0.78	5.19	0.78	**5.04**
U-Net + CRF	0.82	4.61	0.82	**4.40**	0.81	4.73	0.81	**4.53**	0.80	5.22	0.80	**5.15**
UNETr	0.75	5.40	**0.76**	**5.27**	0.73	**5.52**	0.73	5.82	**0.69**	**6.42**	0.67	9.69

Table 10 compares the segmentation results obtained with our method for the different CNNs and configurations of the training set with the recent analogous work based on inexact graph matching [14]. Both approaches yield equivalent results in most configurations, with our method showing a slight improvement in Hausdorff distance. Our proposal exhibits weaker results only for UNETr in configuration C.

4.4. Influence of coarsening

Depending on the coarsening radius, the number of edges in the graph varies significantly (Figure 5 and 6).
Figure 5 represents the average number of edges on the test dataset as a function of the coarsening radius. Only four configurations are reported for clarity: two with FASSEG (Efficient Net and U-Net - both with configuration C) and two with IBSR (U-Net - configuration C and UNETr - configuration B). It can be seen that the number of edges evolves similarly for all configurations. Typically, a radius of $\mu - 1.5\sigma$ divides the number of edges by about 20 compared to complete graphs. The higher number of edges for U-Net (C) on IBSR can be explained by the higher number of nodes (Table 6).

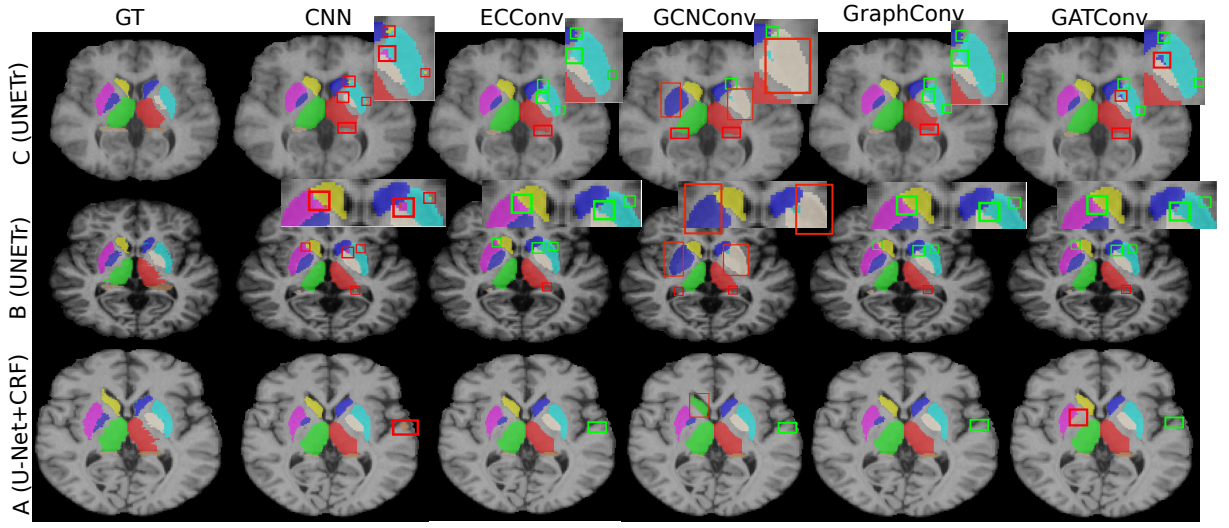

Figure 4: Examples of segmentation on IBSR with and without GNN. Comparison of the 4 graph convolution operators is provided. Bounding boxes highlight regions with significant improvements. A, B and C correspond to the considered training configurations.

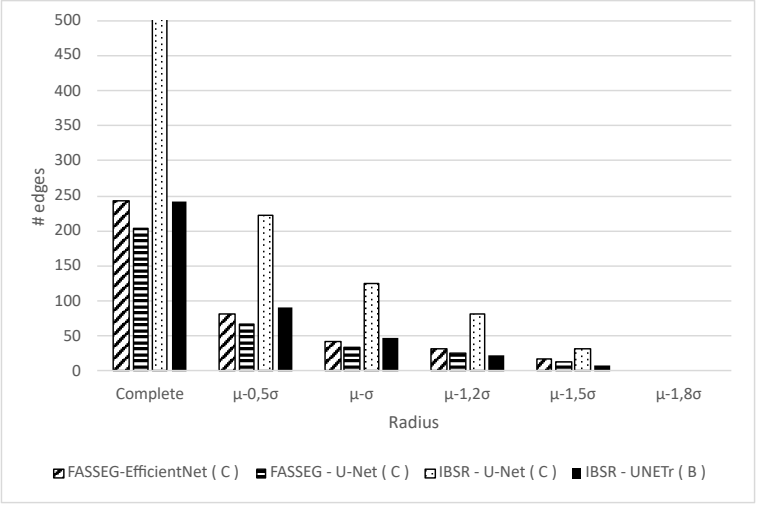

Figure 5: Average number of edges in the test dataset as a function of the coarsening radius considered. Two studies on FASSEG (EfficientNet-C, U-Net-C) and IBSR (U-Net-C, UNETr-B) are illustrated.

Figure 6 shows that coarsening provides structural information to the GNN. Only nodes corresponding to spatially connected structures are linked together, while isolated nodes that are too far from other structures are considered artifacts.

Figure 7 presents the evolution of the Hausdorff distance on the training data as a function of the coarsening radius in the four configurations considered. According to the CNN outputs, the number of nodes (and thus of edges) is different, resulting in a different optimal coarsening radius for each case. Therefore, it is necessary to set the most suitable radius for each configuration, based on the training dataset. On the two graphs in the upper line, corresponding to the FASSEG dataset, we observe similar behaviors for EfficientNet and U-Net. In both configurations, GCNConv and GATConv are less efficient (sometimes even worse than the CNN alone) when the radius becomes too large ($\rho \geq \mu - 1.2\sigma$), i.e., when the number of edges increases (higher than approximately 40, according to Figure 5). This can be explained by their insufficient number of parameters (Table 4) to manage such a large number of edges and related embedded structural information. GCNConv even achieves its best performance for $\rho = \mu - 1.8\sigma$, i.e. when the GNN is reduced to a SLP,

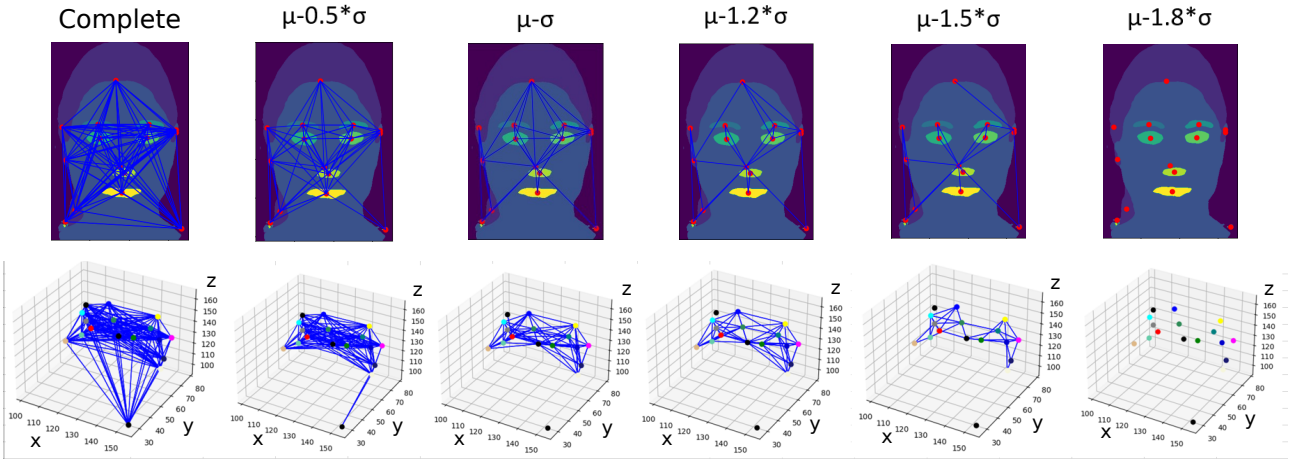

Figure 6: Evolution of the number of edges as a function of the coarsening radius. The first line shows an example on FASSEG (EfficientNet-C) and the second one on IBSR (U-Net - C). x: right-left axis, y: anterior-posterior axis, z: inferior-superior axis of the brain.

which raises questions about its effectiveness. GraphConv and ECConv are more adapted to different coarsening radii. However, ECConv is the only operator that consistently works well with complete graphs in all cases (FASSEG - U-Net on Figure 7). This may be due to its consideration of edge attributes and its higher number of trainable parameters (Table 4). The third graph (IBSR - U-Net) presents similar information regarding the decrease in efficiency of GCNConv and GATConv as the number of edges increases. It also highlights the impact of the graph convolution operator with the Hausdorff distance re-augmentation in the case $\rho = \mu - 1.8\sigma$ going from about 4 to 28 for ECConv and GraphConv. This indicates that convolution adds value beyond the simple use of a SLP. For UNETr, which corresponds to a small number of nodes (Table 6), all coarsening radii seem equivalent. In any case, the segmentation produced by the GNN is improved by coarsening compared to the use of complete graphs. From the training dataset, it seems interesting to work with a radius between $\mu - 1.2\sigma$ and $\mu - 1.5\sigma$ to maximize performance.
Thus, the same study is performed on test images considering the most suitable coarsening radii, which were determined based on the training data, namely $\mu - 1.2\sigma$ and $\mu - 1.5\sigma$. Each value is applied to both the training and testing datasets, and the results reported in Figure 8 only pertain to the test dataset. To evaluate the effectiveness of graph convolution operators compared to a unique SLP, we also study the results with the coarsening radius $\rho = \mu - 1.8\sigma$. For FASSEG, it is observed that the use of a convolution operator improves the results compared to a unique SLP (i.e. $\rho = \mu - 1.8\sigma$), except for GCNConv combined with EfficientNet. This observation confirms the importance of convolution in the node classification task. For the IBSR dataset, the results confirms that GCNConv and GATConv are not appropriate for this overly complex problem (refer to Table 7). Indeed, they produce worse results than a single SLP. When using U-Net, ECConv and GraphConv improve the node predictions, resulting in a significant reduction in HD. For UNETr, none of the convolutional operators provide any added value. A possible reason for this is that the average number of nodes (15.6 according to Table 6) is almost identical to the number of classes (15), which simplifies the classification task. In such cases, using only a SLP to exploit node attributes is sufficient. However, this is not representative of the majority of cases.

5. Discussion

This work demonstrates the effectiveness of a graph convolutional network in enhancing CNN-based semantic image segmentation by exploiting CNN prediction and spatial relationships between segmented regions. Our proposal improves the performance of various CNNs by an average of 24% for FASSEG (Table 3) and 74% for IBSR (Table 7) when trained with a complete training dataset (configuration A). The results presented in section 4.2, for the segmentation of faces on 2D images, show the relevance of using convolutional operators that consider multidimensional edge attributes (L function) in the message passing strategy (ECConv, GATConv) for the node classification task. However, as the number of nodes and of classes (i.e. the dimension of node attributes) increases, the number of trainable parameters of graph convolution operators becomes a fundamental factor in correctly classifying

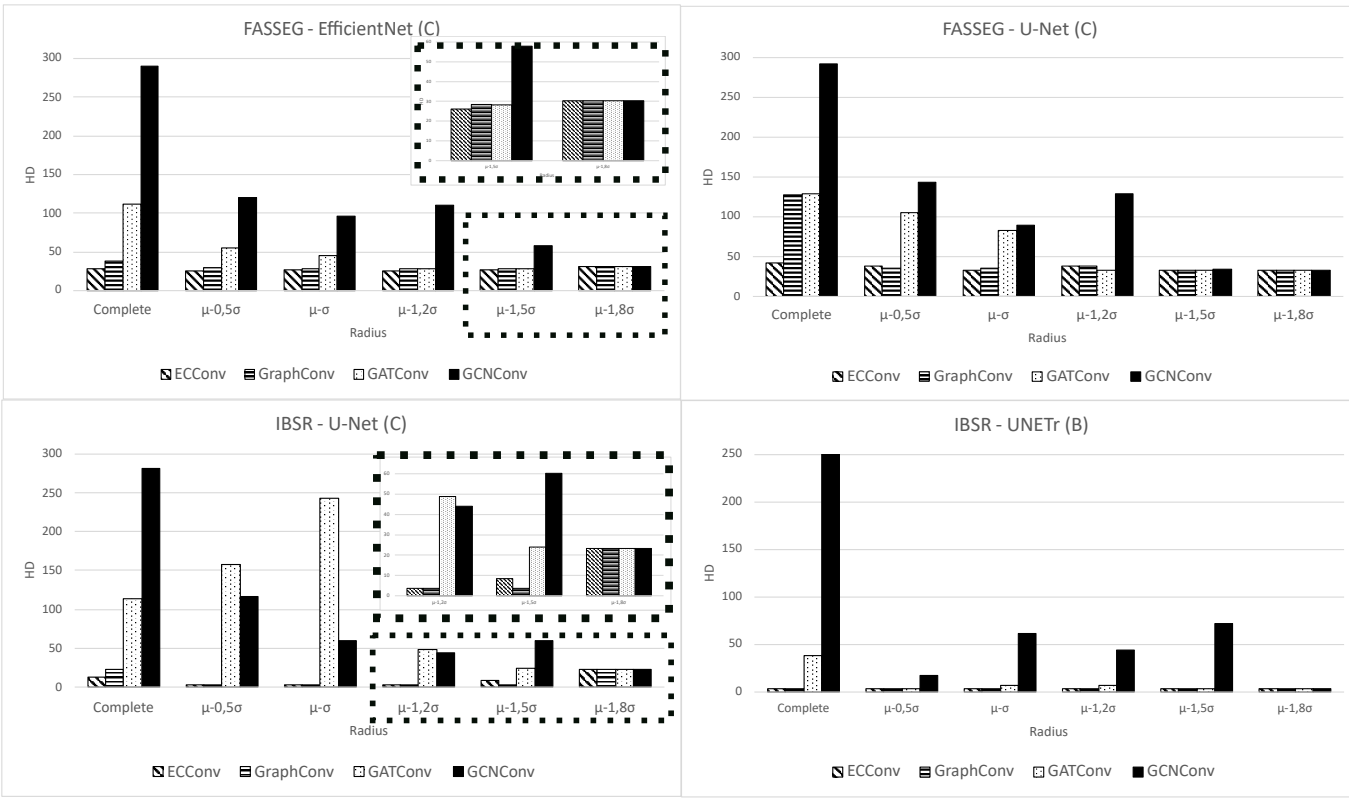

Figure 7: Evolution of the Hausdorff distance of the segmentations as a function of the coarsening radius considered (study on the training dataset). Two studies on FASSEG (EfficientNet-C and U-Net-C) and IBSR (U-Net-C and UNETr-B) are illustrated.

nodes. Therefore, in the case of IBSR (Section 4.3), more complex operators like ECConv and GraphConv (Table 9) outperform simpler ones like GCNConv or GATConv. The efficiency of GraphConv, despite ignoring attributes on edges (more precisely, being limited to a scalar value set to 1 instead of a full vector), demonstrates that the spatial information provided by the graph architecture (coarsening) is sufficient, in this case, to correct structural inconsistencies. In section 4.4, we showed that GCNConv is not well adapted to the problem of semantic image segmentation.

The proposed method is a generic framework, that can be applied to any CNN producing a segmentation map. It does not require the design of a specific architecture such as [40, 41, 42], nor does it require additional over-segmentation algorithms to produce superpixels such as [17, 18]. The structural information presented in each application (normalized minimum and maximum distances for FASSEG, relative positions of the barycentres for IBSR) can be extended to other relationships. It is important to note that while ad-hoc segmentation strategies (e.g., facial landmark detection methods for FASSEG) may improve results, they may also reduce genericity. The simplicity of the proposed network, with only two layers, enhances its comprehensibility, speed, and reduces the 'black box' effect that some GNNs may have [15]. The inference time (including graph construction and node classification) is, on average, of 3 seconds for FASSEG and of 10 seconds for IBSR (Nvidia Quadro RTX 3000 GPU), compared to more than 3 minutes in [14]. Additionally, our proposal is resilient to small training datasets, which often result in larger graphs due to imperfect training of the CNN, leading to more connected components. By combining our method with a light-weight architecture such as U-Net (compared to UNETr, for instance) trained with few data, we achieve comparable or even superior results to complex deep learning networks such as UNETr and EfficientNet, trained on larger datasets.

The efficiency of the method significantly depends on graph coarsening (section 4.4). Coarsening provides additional spatial information by reducing the number of edges in the graph (Figure 5), resulting in a richer and finer graph structure. This decreases the number of neighbors involved in message passing for convolution operators, thereby reducing graph complexity. We shew that a radius ρ between $\mu - 1.2\sigma$ and $\mu - 1.5\sigma$ yields optimal performance in all

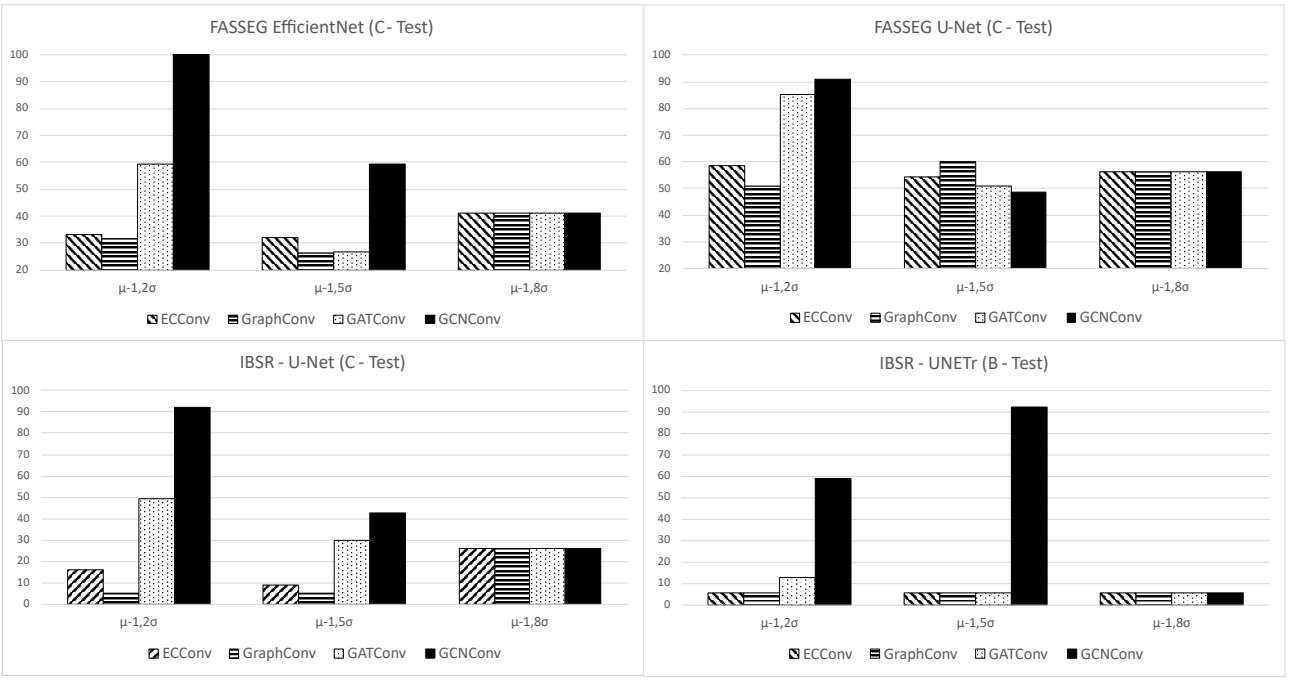

Figure 8: Evolution of the Hausdorff distance of the segmentations as a function of the coarsening radius considered (study done on the test dataset). Two studies on FASSEG (EfficientNet-C and U-Net-C) and IBSR (U-Net-C and UNETr-B) are illustrated.

configurations, while also dividing the number of edges in the graph by about 20. In Figures 7 and 8, we observed that a convolution operator like ECConv, with more trainable parameters and managing multidimensional edge attributes, handles complete graphs much better than the three other operators studied. Maybe the larger number of parameters to be optimized of ECConv better compensates for the effect of the permutation invariant function during the message passing (average, sum, etc.) which can partly eliminate the discriminating character of each edge.

Our proposal suffers from some limitations that we point out hereafter, together with future works to be performed.

First, it would be interesting to incorporate the learning of the coarsening level (parameter ρ) into the GNN training process, like the edge contraction used in some pooling operators [48] based on an edge score (depending in our case on the considered spatial relationship). One would therefore benefit from a homogeneous end-to-end training procedure, allowing to simultaneously learn both the coarsening level and the node classification task.

Second, in some cases, different coarsening radii can provide complementary information for node classification. Thus, in Figure 7, the radii $\mu - \sigma$ and $\mu - 1.5\sigma$ with ECConv and GraphConv for instance, are working well and could enrich the predictions for each node by being combined. We attempted to combine two convolution inputs, which resulted in a slight improvement in performance. This approach was not pursued in this work due to the increased complexity it introduced to the architecture.

Third, the efficiency of our method requires that each structure is detected and possibly over-segmented by the CNN. It is assumed that the CNN produces a kind of relevant superpixels, which is a reasonable hypothesis based on experiments. Our proposal acts as a GNN-based post processing aiming at ensuring the overall spatial coherence. This clearly appears in the segmentation results (Tables 3 and 7) showing that the Hausdorff distance improvement is significant, particularly for IBSR. These enhancements reflect the effective correction of artifacts, as well as the accurate correction of classification errors (Figures 2 and 4). Regarding the Dice coefficient, the improvement is minimal, especially for FASSEG. This is expected since the Dice is not significantly affected by small (compared to the target region) and distant artifacts. It would be worthwhile to assess whether the subdivision of the regions segmented by the CNN would have an impact on the Dice. The subdivisions could be determined by considering the uncertainty of the CNN [59].

This paper focuses on proposing the method and evaluating it on two datasets using various CNNs and graph convolution operators, with the same experimental methodology recently considered in [14]. As a future prospect, we aim to evaluate our proposal on additional datasets (ADE20K [5], RUGD [60] or MSD [29]).

6. Conclusion

We propose a framework based on a graph neural network (GNN) performing a node classification task to improve CNN-based semantic image segmentation. Our approach utilizes the vector probability from the CNN output as node attributes and the spatial relationships between segmented regions as edge attributes. We investigate the impact of structural information through the edge attributes (comparison of four graph convolutional operators), but also through the graph architecture (coarsening level). Our method is validated through experiments on two applications (2D image and 3D volumetric data), using various CNNs. The results demonstrate the robustness of our proposal even in the absence of large and representative training datasets, outperforming more complex deep learning methods (EfficientNet, UNETr) trained with larger datasets. The proposal leads to a generic framework for semantic image segmentation, that can be extended to any type of structural information and additional properties describing regions (e.g., combining class membership probabilities and region size in node attributes). Future work will focus on improving the GNN by incorporating the learning of the most appropriate coarsening level, considering CNN uncertainty for region subdivision prior to graph construction, and applying the approach to other relevant applications.

References

[1] S. Asgari Taghanaki, K. Abhishek, J. Cohen, J. Cohen-Adad, G. Hamarneh, Deep semantic segmentation of natural and medical images: a review, Artificial Intelligence Review 54 (2021) 137–178.

[2] A. Garcia-Garcia, S. Orts-Escolano, S. Oprea, V. Villena-Martinez, P. Martinez-Gonzalez, J. Garcia-Rodriguez, A survey on deep learning techniques for image and video semantic segmentation, Applied Soft Computing 70 (2018) 41–65.

[3] H. Zhang, K. Dana, J. Shi, Z. Zhang, X. Wang, A. Tyagi, A. Agrawal, Context encoding for semantic segmentation, in: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, 2018, pp. 7151–7160. doi:10.1109/CVPR.2018.00747.

[4] T. Mingxing, V. L. Quoc, EfficientNet: Rethinking model scaling for convolutional neural networks, in: International Conference on Machine Learning (ICML), 2019, pp. 6105–6114. doi:10.48550/arXiv.1905.11946.

[5] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, P. Luo, SegFormer: Simple and Efficient Design for Semantic Segmentation with Transformers, Advances in Neural Information Processing Systems 34 (2021) 12077–12090.

[6] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, S. Xie, A ConvNet for the 2020s, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022, pp. 11976–11986. doi:10.48550/arXiv.2201.03545.

[7] I. Bloch, Fuzzy sets for image processing and understanding, Fuzzy Sets and Systems 281 (2015) 280–291.

[8] J.-B. Fasquel, N. Delanoue, A graph based image interpretation method using a priori qualitative inclusion and photometric relationships, IEEE Transactions on Pattern Analysis and Machine Intelligence 41 (2019) 1043–1055.

[9] A. Delaye, E. Anquetil, Fuzzy relative positioning templates for symbol recognition, in: Internation Conference on Document Analysis and Recognition, 2011, pp. 1220–1224. doi:10.1109/ICDAR.2011.246.

[10] L. Kunze, C. Burbridge, M. Alberti, A. Thippur, J. Folkesson, V. Jensfelt, N. Hawes, Combining top-down spatial reasoning and bottom-up object class recognition for scene understanding, in: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2014, pp. 2910–2915. doi:10.1109/IROS.2014.6942963.

[11] J. Chopin, J.-B. Fasquel, H. Mouchere, R. Dahyot, I. Bloch, Semantic image segmentation based on spatial relationships and inexact graph matching, in: Tenth International Conference on Image Processing Theory, Tools and Applications (IPTA), 2020, pp. 1–6. doi:10.1109/IPTA50016.2020.9286611.

[12] J. Maciel, J.-P. Costeira, A global solution to sparse correspondence problems, IEEE Transactions on Pattern Analysis and Machine Intelligence 25 (2003) 187–199.

[13] A. Zanfir, C. Sminchisescu, Deep learning of graph matching, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 2684–2693. doi:10.1109/CVPR.2018.00284.

[14] J. Chopin, J.-B. Fasquel, H. Mouchère, R. Dahyot, I. Bloch, Model-based inexact graph matching on top of dnns for semantic scene understanding, Computer Vision and Image Understanding (2023) 103744.

[15] D. Bacciu, F. Errica, A. Micheli, M. Podda, A gentle introduction to deep learning for graphs, Neural Networks 129 (2020) 203–221.

[16] Z. Zhang, P. Cui, W. Zhu, Deep learning on graphs: A survey, IEEE Transactions on Knowledge and Data Engineering 34 (2020) 249–270.

[17] S. Ouyang, Y. Li, Combining deep semantic segmentation network and graph convolutional neural network for semantic segmentation of remote sensing imagery, Remote Sensing 13 (2021).

[18] Q. Diao, Y. Dai, C. Zhang, Y. Wu, X. Feng, F. Pan, Superpixel-based attention graph neural network for semantic segmentation in aerial images, Remote Sensing 14 (2022) 305.

[19] C. Shorten, T. M. Khoshgoftaar, A survey on image data augmentation for deep learning, Journal of Big Data 6 (2019).

[20] K. Weiss, T. Khoshgoftaar, D. Wang, A survey of transfer learning, Journal of Big Data 3 (2016).

[21] J. Ding, N. Xue, G. Xia, D. Dai, Decoupling zero-shot semantic segmentation, in: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022, pp. 11583–11592. doi:10.1109/CVPR52688.2022.01129.

[22] P. Coupeau, J.-B. Fasquel, M. Dinomais, On the relevance of edge-conditioned convolution for GNN-based semantic image segmentation using spatial relationships, in: Eleventh International Conference on Image Processing Theory, Tools and Applications (IPTA), 2022, pp. 1–6. doi:10.1109/IPTA54936.2022.9784143.

[23] O. Ronneberger, P. Fischer, T. Brox, U-Net: Convolutional Networks for Biomedical Image Segmentation, Medical Image Computing and Computer-Assisted Intervention 9351 (2015) 234–241.

[24] H. Zhao, J. Shi, X. Qi, X. Wang, J. Jia, Pyramid scene parsing network, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 6230–6239. doi:10.1109/CVPR.2017.660.

[25] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, A. Yuille, Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs, IEEE transactions on pattern analysis and machine intelligence 40 (2017) 834–848.

[26] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, T. Unterthiner, X. Zhai, An image is worth 16x16 words: Transformers for image recognition at scale, arXiv preprint arXiv:2010.11929 (2020).

[27] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017).

[28] J. Fu, J. Liu, H. Tian, Y. Li, Y. Bao, Z. Fang, H. Lu, Dual attention network for scene segmentation, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019, pp. 3146–3154. doi:10.1109/CVPR.2019.00326.

[29] A. Hatamizadeh, Y. Tang, V. Nath, D. Yang, A. Myronenko, B. Landman, H. R. Roth, D. Xu, UNETR: Transformers for 3D medical image segmentation, in: IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), 2022, pp. 1748–1758. doi:10.1109/WACV51458.2022.00181.

[30] J.-W. Zhang, Y. Sun, Y. Yang, W. Chen, Feature-proxy transformer for few-shot segmentation, Advances in Neural Information Processing Systems 35 (2022) 6575–6588.

[31] T. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, International Conference on Learning Representations (ICLR) (2017).

[32] C. O. Laura, S. Wesarg, G. Sakas, Graph matching survey for medical imaging: On the way to deep learning, Methods 202 (2022) 3–13.

[33] A. Mohamed, K. Qian, M. Elhoseiny, C. Claudel, Social-STGCNN: A Social Spatio-Temporal Graph Convolutional Neural Network for Human Trajectory Prediction, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 14412–14420. doi:10.1109/CVPR42600.2020.01443.

[34] H. Gao, S. Ji, Graph U-Nets, IEEE Transactions on Pattern Analysis and Machine Intelligence 44 (2022) 4948–4960.

[35] M. Simonovsky, N. Komodakis, Dynamic edge-conditioned filters in convolutional neural networks on graphs, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 3693–3702. doi:hal-01576919.

[36] A. S. Nassar, S. D'Aronco, S. Lefevre, J. D. Wegner, Geograph: Graph-based multi-view object detection with geometric cues end-to-end, in: European Conference on Computer Vision (ECCV), 2020, pp. 488–504. doi:10.1007/978-3-030-58571-6_29.

[37] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, S. Süsstrunk, Slic superpixels compared to state-of-the-art superpixel methods, IEEE Transactions on Pattern Analysis and Machine Intelligence 34 (2012) 2274–2282.

[38] J. Zhang, Z. Hua, K. Yan, K. Tian, J. Yao, E. Liu, M. Liu, X. Han, Joint fully convolutional and graph convolutional networks for weakly-supervised segmentation of pathology images, Medical Image Analysis 73 (2021).

[39] R. Xu, Y. Li, C. Wang, S. Xu, W. Meng, X. Zhang, Instance segmentation of biological images using graph convolutional network, Engineering Applications of Artificial Intelligence 110 (2022) 104739.

[40] X. Li, Y. Yang, Q. Zhao, T. Shen, Z. Lin, H. Liu, Spatial pyramid based graph reasoning for semantic segmentation, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 8947–8956. doi:10.1109/CVPR42600.2020.00897.

[41] Y. Chen, M. Rohrbach, Z. Yan, S. Yan, J. Feng, Y. Kalantidis, Graph-based global reasoning networks, 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2019) 433–442.

[42] Q. Liu, M. Kampffmeyer, R. Jenssen, A.-B. Salberg, Scg-net: Self-constructing graph neural networks for semantic segmentation, 2020. URL: https://arxiv.org/abs/2009.01599. doi:10.48550/ARXIV.2009.01599.

[43] S. Y. Shin, S. Lee, I. D. Yun, K. M. Lee, Deep vessel segmentation by learning graphical connectivity, Medical Image Analysis 58 (2019) 101556.

[44] K. Wang, X. Zhang, Y. Lu, X. Zhang, W. Zhang, CGRNet: Contour-guided graph reasoning network for ambiguous biomedical image segmentation, Biomedical Signal Processing and Control 75 (2022) 103621.

[45] Y. Li, D. S. H. Tam, S. Xie, X. Liu, Q. Ying, W. Lau, D. Chiu, S. Z. Chen, Gtea: Representation learning for temporal interaction graphs via edge aggregation, ArXiv abs/2009.05266 (2020).

[46] G. Renton, M. Balcilar, P. Héroux, B. Gaüzère, P. Honeine, S. Adam, Symbols detection and classification using graph neural networks, Pattern Recognition Letters 152 (2021) 391–397.

[47] Z. Ying, J. You, C. Morris, X. Ren, W. Hamilton, J. Leskovec, Hierarchical graph representation learning with differentiable pooling, Advances in Neural Information Processing Systems 31 (2018) 4800–4810.

[48] F. Diehl, Edge contraction pooling for graph neural networks, ArXiv arXiv:1905.10990 (2019).

[49] C. Morris, M. Ritzert, M. Fey, W. L. Hamilton, J. E. Lenssen, G. Rattan, M. Grohe, Weisfeiler and leman go neural: Higher-order graph neural networks, in: Conference on Artificial Intelligence (AAAI), 2019, pp. 4602–4609. doi:10.1609/aaai.v33i01.33014602.

[50] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, Y. Bengio, Graph Attention Networks, International Conference on Learning Representations (ICLR) (2018).

[51] A. P. Zijdenbos, B. M. Dawant, R. A. Margolin, A. C. Palmer, Morphometric analysis of white matter lesions in MR images: method and validation, IEEE Transactions on Medical Imaging 13 (1994) 716–724.

[52] M. Beauchemin, K. P. B. Thomson, G. Edwards, On the hausdorff distance used for the evaluation of segmentation results, Canadian Journal of Remote Sensing 24 (1998) 3–8.

[53] K. Khan, M. Mauro, R. Leonardi, Multi-class semantic segmentation of faces, in: IEEE International Conference on Image Processing (ICIP), 2015, pp. 827–831. doi:10.1109/ICIP.2015.7350915.
[54] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, P. H. S. Torr, Conditional random fields as recurrent neural networks, in: IEEE International Conference on Computer Vision (ICCV), 2015, pp. 1529–1537. doi:10.1109/ICCV.2015.179.
[55] K. Kushibar, S. Valverde, S. González-Villà, J. Bernal, M. Cabezas, A. Oliver, X. Lladó, Automated sub-cortical brain structure segmentation combining spatial and deep convolutional features, Medical Image Analysis 48 (2018) 177–186.
[56] O. Çiçek, A. Abdulkadir, S. Lienkamp, T. Brox, O. Ronneberger, 3D U-Net: Learning dense volumetric segmentation from sparse annotation, Lecture Notes in Computer Science (2016) 424–432.
[57] P. Coupeau, J.-B. Fasquel, E. Mazerand, P. Menei, C. N. Montero-Menei, M. Dinomais, Patch-based 3D U-Net and transfer learning for longitudinal piglet brain segmentation on MRI, Computer Methods and Programs in Biomedicine 214 (2022).
[58] B. Lee, N. Yamanakkanavar, J. Y. Choi, Automatic segmentation of brain MRI using a novel patch-wise U-Net deep architecture, PLOS ONE 15 (2020) 1–20.
[59] A. A. Abdullah, M. M. Hassan, Y. T. Mustafa, A review on bayesian deep learning in healthcare: Applications and challenges, IEEE Access 10 (2022) 36538–36562.
[60] Y. Jin, D. Han, H. Ko, Trseg: Transformer for semantic segmentation, Pattern Recognition Letters 148 (2021) 29–35.