Eigenvector Grouping for Point Cloud Vessel Labeling

Patryk Rygiel PATRYK.RYGIEL@HEMOLENS.EU
Hemolens Diagnostics, Wroclaw University of Science and Technology

Maciej Zieba MACIEJ.ZIEBA@PWR.EDU.PL
Wroclaw University of Science and Technology, Tooploox

Tomasz Konopczynski TOMASZ.KONOPCZYNSKI@HEMOLENS.EU
Hemolens Diagnostics

Abstract

Segmentation of coronary arteries from Coronary Computed Tomography Angiography (CCTA) is an essential step in developing various noninvasive diagnostic methods. In this work, we tackle the task of vessel labeling on coronary artery voxel-based prediction by use of point cloud artificial neural network. We propose a novel point aggregation technique Eigenvector Grouping (EVG), tailored to the analysis of tubular-like structures. We further utilize a specifically designed post-processing technique Component-Wise Majority Point Voting (CMPV), to refine point cloud segmentation by enforcing class consistency among connected components. We show that our solution outperforms previously proposed methods in the vessel labeling task on a CCTA dataset especially, in the presence of disrupted segmentations.

Keywords: point clouds, vector grouping, medical image segmentation, coronary artery segmentation

1. Introduction

Coronary artery disease (CAD) is one of the most common types of heart disease in the European Union and the United States. Therefore, it is essential to perform computed tomography (CT) scans to support diagnosing a range of cardiovascular conditions such as CAD. To asses patient's condition noninvasively, accurate segmentation of coronary arteries (CA) from CT scans is required.

Segmentation of coronary arteries from 3D CCTA scans is often done with convolution-based 3D U-Net architecture proposed by Ronneberger et al. (2015); Cicek et al. (2016). Due to the large size of 3D CCTA scans and heavy computational cost of 3D convolutions, a single volume cannot be processed at once - volumes need to be divided into local patches. There are two main patching techniques used in literature: cube patching used by Huang et al. (2018); Chen et al. (2019); Kjerland (2017) and slice patching used by Pan et al. (2021); Cheung et al. (2021); Mirunalini et al. (2019). Even though, those approaches yield good results in the coronary arteries segmentation task, they cause some common errors. First of all, due to the lack of global context in a local patch, U-Net tends to detect large amount of false positive (FP) vessels and tubular-like structures which are not coronary vessels. Secondly, due to patching, coronary branches are prone to be disconnected from the main component - this mainly occurs in the areas where two patches meet. Disruption are also caused by common coronary artery tomography (CTA) issues such as calcified plaques, not

evenly distributed contrast and noise. Therefore, the process of further refinement of U-Net predictions is required to obtain high-quality segmentation masks.

He et al. (2020) propose to use a point cloud based neural network to perform U-Net predictions refinement with the task *vessel labeling*. It is defined as the segmentation task, where each point, predicted as a CA by U-Net, is being assigned one of the following labels: LCA (left coronary artery), RCA (right coronary artery) and FP vessel. This task has three main goals: (I) removal of FP vessels and other objects incorrectly predicted by U-Net; (II) distinction of LCA and RCA vessels; (III) detection of disconnected coronary vessel branches. He et al. (2020) use PointNet++ by Qi et al. (2017) and DGCNN by Wang et al. (2019) architectures with the proposed geometry-aware grouping (GAG) strategy to perform this task. In this work we tackle the problem of *vessel labeling* and propose a novel grouping strategy for PointNet++ architecture which we call eigenvector grouping (EVG). EVG is designed to exploit vascular tree topology by grouping points with vectors that approximate vessel direction. Such approach is robust at finding disconnected CA branches and discriminating between FP and CA vessels. To further refine point cloud segmentations to facilitate class consistency among connected components, we propose a post-processing algorithm component-wise majority point voting (CMPV). Our proposed approach is evaluated on our internal dataset composed of segmentation maps obtained from a 3D U-Net with a 3D cube patching and compared with reference methods. Moreover, we show that our method works significantly better for predictions with disrupted lumen in two additional experiments.

Our contributions: (I) We propose a new point grouping method, EVG which utilizes a prior knowledge of a vessel shape. (II) We propose a post-processing algorithm CMPV which further refines segmentation results by enforcing class consistency among connected components. (III) We report accuracy of 97.88% on CA using a skeleton metric on our internal dataset - thus surpassing current best performing approach to this task by He et al. (2020). (IV) We show that EVG has better generalization capabilities than other methods on disrupted segmentations.

2. Method

We propose a novel approach to the problem of vessel labeling described in Figure 1. Below we briefly describe each of the components of this method.

CA segmentation. During the initial segmentation stage the input 3D scan $\mathcal{I}$ is transformed to a binary segmentation mask Y, where $u(\mathcal{I}) = Y$, and $y_{i,j,k} = 1$ if the point at (i, j, k) is classified as vessel, and $y_{i,j,k} = 0$ otherwise. The operation $u(\cdot)$ can be performed by any segmentation procedure – we utilize 3D cube patching together with 3D U-Net for sake of efficiency and memory limitations.

Conversion to a point cloud representation. To perform task of *vessel labeling* to remove artifact and keep disconnected CA vessels from the segmentation results we apply further processing using efficient point cloud representation. We create the set of 3D coordinates (point cloud representation) $\mathcal{X} = \{(i, j, k) : y_{i,j,k} = 1\}$ for which the segmentation mask returned positive values.

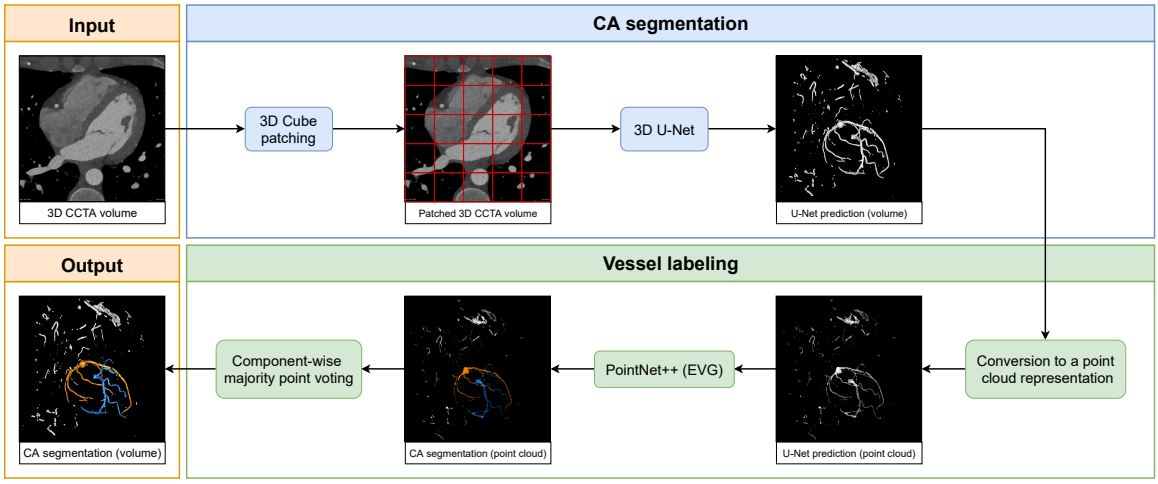
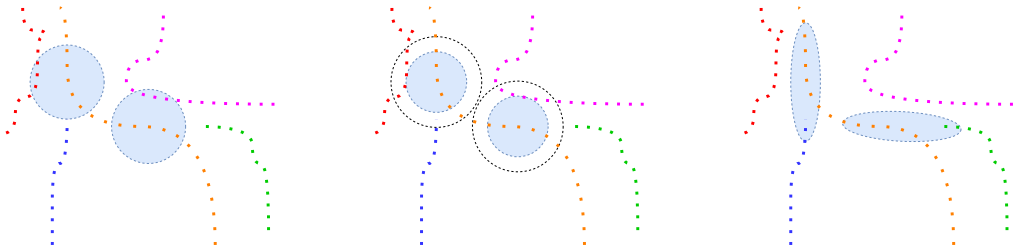


Figure 1: Method diagram. The *CA segmentation* step utilizes a 3D U-Net architecture with 3D cube patching. The next step *vessel labeling*, on which we focus in this work, performs point cloud segmentation with a PointNet++ and proposed EVG grouping strategy. Results are later refined with CMPV algorithm and converted back to the volumetric representation.

Point cloud processing. Following the procedure described in He et al. (2020) we process each point $\mathbf{x}_n$ from $\mathcal{X}$ using a PointNet++ architecture. The encoder block of PointNet++ architecture extracts set of representatives $\mathcal{Z} \subseteq \mathcal{X}$ with `FPS` algorithm by Eldar et al. (1997). For each $\mathbf{z}_m \in \mathcal{Z}$ grouping operation $g(\cdot)$ is performed to extract a set of representative's neighbours $\mathcal{Q}_{\mathbf{z}_m}$. Neighbourhoods extracted in such manner are then processed with a PointNet to perform feature extraction. Features are later aggregated onto the set of representative points $\mathcal{Z}$ to reduce data dimensionality. As a grouping operation $g(\cdot)$, PointNet++ utilizes multi-scale grouping (MSG) which is based on computing Euclidean distance and then querying points with the KNN or the ball query defined in Qi et al. (2017). A multi-scale feature is a process of defining more than one grouping area per representative point $\mathbf{z}_m$ and can be used for any grouping strategy we discuss in this work. He et al. (2020) propose geometry-aware grouping (GAG) which facilitates class consistency among vessels by being more likely to group points belonging to the same connected component. The Euclidean distance metric is modified by multiplying it by a parameter λ (set to 0.25 by the authors) when two points are from the same connected component and by $(1 - \lambda)$ otherwise. Visual comparison between these approaches is showcased at Figures 2(a), 2(b). Neither of these approaches leverages topological relationships between disconnected CA vessels and the fact that direction of the vessel provides meaningful information. Disconnections are most often caused by the occurrence of plaques. Since only the plaque neighbouring segmentation is missing, the disconnected vessel part can be easily traced by following the main vessel direction. Moreover, GAG in its design inherently works against finding disconnected vessels, since points belonging to different components are less likely to

be grouped together. We propose to use an alternative, novel strategy named Eigenvector grouping (EVG) which is designed to exploit vessel direction.



(*a*) MSG (*b*) GAG (*c*) EVG

Figure 2: Visual comparison of the points grouping strategies with a ball query. Points belonging to the same components are marked with the same color. (a) MSG: The grouping is performed with a ball of a given radius around the root point; (b) GAG: The smaller ball groups all the points present in it and the larger one only those which are of the same component as the root point; (c) EVG: The grouping areas stretch along the vessel direction grouping points of disconnected vessel branches while omitting artefact points.

Eigenvector Grouping (EVG). Our proposed solution is based on finding the first eigenvector (one with the largest eigenvalue) of local point neighbourhood and performing point aggregation with it. The top eigenvector defines direction of the most variance in a local point cloud, which corresponds to the vessel's direction. First, for each representative $\mathbf{z}_m \in \mathcal{Z}$, we identify K nearest neighbours in the input point cloud set $\mathcal{X}$. The identified nearest neighbours are stored in rows of matrix $\mathbf{N}_{\mathbf{z}_m}$ of size $K \times 3$. The mean point $\boldsymbol{\mu}_{\mathbf{z}_m}$ is further calculated, and each of the points stored in $\mathbf{N}_{\mathbf{z}_m}$ is centred by subtracting the calcualted mean value. In the next step a covariance matrix $\mathbf{C}_{\mathbf{z_m}}$ of $\mathbf{N}_{\mathbf{z}_m}$ is calculated:

$$\mathbf{C}_{\mathbf{z_m}} = \frac{1}{K} \cdot \mathbf{N}_{\mathbf{z}_m}^{\mathrm{T}} \mathbf{N}_{\mathbf{z}_m} \tag{1}$$

To calculate the top eigenvector of the given matrix $\mathbf{C}_{\mathbf{z_m}}$ many different methods can be used. Common eigendecomposition algorithms: Francis (1961); Jacobi (1846); Cuppen (1981) compute all eigenpairs and thus are too time-consuming to be used repeatedly in a training loop. Since our approach requires only an eigenvector with the top eigenvalue we utilize a power iteration algorithm by Müntz (1913) which is an efficient choice for approximating the top eigenvector. The top eigenvector is computed in a following iterative manner:

$$\mathbf{b}_{\mathbf{z}_m}^{(i+1)} = \frac{\mathbf{C}_{\mathbf{z_m}} \mathbf{b}_{\mathbf{z}_m}^{(i)}}{||\mathbf{C}_{\mathbf{z_m}} \mathbf{b}_{\mathbf{z}_m}^{(i)}||_2} \tag{2}$$

where $\mathbf{b}_{\mathbf{z}_m}^{(0)}$ is a random vector with a norm of 1. With each iteration vector $\mathbf{b}_{\mathbf{z}_m}^{(i)}$ better approximates the top eigenvector since $\mathbf{b}_{\mathbf{z}_m}^{(i)} \xrightarrow[i\to\infty]{} \mathbf{e}_{\mathbf{z}_m}$. For our purposes 20 iterations were enough to obtain desired approximation of $\mathbf{e}_{\mathbf{z}_m}$.

For each $\mathbf{z}_n \in \mathcal{Z}$ we construct a grouping line segment $\mathbf{s}_{\mathbf{z}_m}$ which direction is defined by the computed eigenvector $\mathbf{e}_{\mathbf{z}_m}$. The line segment is centered at the root point $\mathbf{z}_n$ and its length is specified by the method hyperparameter λ. For each $\mathbf{x}_n \in \mathcal{X}$ the Euclidean distance to the line segment $\mathbf{s}_{\mathbf{z}_m}$ is calculated. The set of neighbours $\mathcal{Q}_{\mathbf{s}_m}$ of the line segment $\mathbf{s}_{\mathbf{z}_m}$ can then be queried via the KNN or the ball query which defines distance threshold (radius) for grouping. The queried set $\mathcal{Q}_{\mathbf{s}_m}$ is later processed with a PointNet.

Our proposed EVG strategy is more likely to group points along the vessel direction to robustly discriminate between disconnected CA vessel and FP ones (see Figure 2(*c*)).

Component-wise majority point voting (CMPV). To further refine segmentation results we design a post-processing algorithm CMPV, which enforces class consistency among components by assigning one class to all points belonging to the same connected component. The set of unique components $\mathcal{C} = \{c_1, c_2, ..., c_M\}$ is obtained from segmentation mask Y from prior segmentation procedure $u(\cdot)$, by application of *connected components labeling* algorithm (CCL) by Park et al. (2000) (CCL operates on voxels thus we utilize a prior voxel segmentation mask). For each component $c_m \in \mathcal{C}$ we identify the points $\mathcal{X}_m \subseteq \mathcal{X}$, that belong to c_m and calculate their corresponding class labels $\hat{\mathcal{Y}}_m$ returned by PointNet++. The most frequently observed class in $\hat{\mathcal{Y}}_m$ identified and all members of $\mathcal{X}_m$ are re-labelled with this majority class. This procedure is applied to all unique components.

3. Experiments & Results

Our dataset consists of 100 CCTA volumes gathered from five different medical centers. It consists of 42 volumes of female and 58 volumes of male patients who are on average 70.41 years old. Images are acquired from six kinds of scanners: Siemens SOMATOM Force, Siemens Sensation Cardiac 64, Canon Medical Systems Aquilion ONE, and GE Medical Systems Discovery CT750 HD. Volumes have a size of $512 \times 512 \times (130 - 420)$, with spacing $0.4 - 0.6$ mm and a typical pixel size of around 0.33^2 mm^2. The dataset example is showcased in Figure 1, more examples are included in the supplementary Appx. A. For the task of *vessel labeling* we split the dataset into 80 training and 20 test samples. To avoid data leakage between the two training modules, we use the same data split during training for both U-Net and PointNet++.

For a fair comparison between grouping methods we use the same backbone for each task. We train a standard 3D U-Net on the CCTA dataset with the `DiceLoss`. Each point cloud $\mathcal{X}$ consists of $200,000$ to $400,000$ points. During training we downsample them randomly to $20,000$ points. PointNet++ is trained for 200 epochs with batch size of 4. As an optimizer we use `Adam` and for learning rate (lr) scheduler `OneCycleLR` by Smith (2018) with a max lr set to 0.0025. As a loss function we use classic `CrossEntropy` and for data augmentation small rotations up to 30 degrees in each axis, translations and jittering. Architectures details and their running times are provided in the supplementary Appx. B. During inference we adopt scheme proposed by Balsiger et al. (2019). The

point cloud is split into chunks of 20,000 and inference is run for each chunk seperately to obtain segmentation result for each point in the original point cloud. This procedure is repeated five times with random chunks. The segmentation scores are computed via majority voting. Implementation is done with use of `PyTorch` 1.8.1 Paszke et al. (2019), `pytorch-lightning` Falcon and `MONAI` Consortium (2020) libraries. For the PointNet++ architecture we use the `Pointnet++ PyTorch` Wijmans library and extend it with GAG and EVG grouping strategies implementation.

Table 1: Qualitative results using the skeleton metric - mean (± std). Results for MSG, GAG and EVG grouping strategies with different query methods. Each method is further evaluated with CMPV.

Grouping	**Query + post-processing**	**Metrics (%)**			
		Accuracy	**F1-score**	**Precision**	**Recall**
MSG	knn	96.08 (± 4.60)	92.53 (± 10.12)	93.18 (± 10.50)	92.78 (± 9.18)
	knn + CMPV	96.80 (± 3.58)	93.55 (± 9.56)	94.46 (± 8.99)	93.49 (± 10.41)
	radius	97.36 (± 2.11)	95.20 (± 4.12)	95.20 (± 5.67)	95.59 (± 4.45)
	radius + CMPV	97.73 (± 1.79)	95.75 (± 3.52)	95.89 (± 5.27)	**96.09 (± 3.62)**
GAG	knn	96.63 (± 3.27)	93.16 (± 9.08)	94.89 (± 5.99)	92.56 (± 12.02)
	knn + CMPV	96.72 (± 3.32)	93.06 (± 10.12)	94.15 (± 9.86)	92.68 (± 11.84)
	radius	97.55 (± 1.82)	95.48 (± 3.59)	96.01 (± 4.25)	95.38 (± 3.93)
	radius + CMPV	97.64 (± 1.94)	95.66 (± 3.49)	96.35 (± 3.87)	95.53 (± 4.13)
EVG	knn	96.18 (± 3.49)	92.33 (± 9.59)	93.08 (± 9.44)	92.19 (± 10.97)
	knn + CPMV	97.14 (± 3.02)	93.99 (± 9.04)	94.86 (± 8.57)	93.71 (± 10.26)
	radius	97.73 (± 1.62)	95.90 (± 3.05)	96.35 (± 3.63)	95.85 (± 3.24)
	radius + CMPV	**97.88 (± 1.76)**	**96.10 (± 3.20)**	**96.77 (± 3.31)**	96.00 (± 3.19)

We compare our proposed approach EVG with MSG by Qi et al. (2017) and GAG by He et al. (2020) grouping strategies. For all methods, we experiment with both the ball query (denoted "radius") and k-nearest-neighbours (KNN) as point query methods. Models are tested with and without our post-processing algorithm CMPV. We evaluate the performance either by sampling from the entire region of an artery (denoted "dense metric") or from a skeleton of an artery (denoted "skeleton metric"). We consider the skeleton metric to be more meaningful as it better conveys correct cardiovascular tree topology as vessel width.

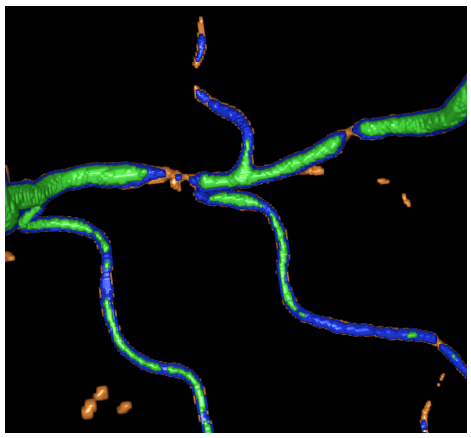
Evaluation on CCTA Table 1 showcases results on our internal CCTA dataset that represents good prior U-Net performance. For all grouping strategies, methods using the ball query outperform the KNN query. Our proposed approach achieves the highest results with both the post-processing and without it for the accuracy, F1-score and the precision metric. The highest result for the recall metric is obtained by MSG. CMPV enforces the class consistency among components. Methods utilizing it achieve slightly better results. Additional experiment using the dense metric is included in the supplementary Appx. C.

Undersegmented predictions We evaluate robustness of methods toward undersegmented predictions by reducing the threshold values of U-net predictions. Table 3(a) showcases methods performance for 5 different threshold values (Fig 3(b) shows U-Net segmen-

tation for different thresholds). With higher thresholds CA disruptions are more prevalent and larger. For higher thresholds performance of MSG and GAG degrades while EVG maintains mean F1-score results of > 0.9. Figure 4 shows a qualitative example in which the superiority of EVG for highly undersegmented vessels is clearly visible. Additional statistical significance tests are in the Appx. C.

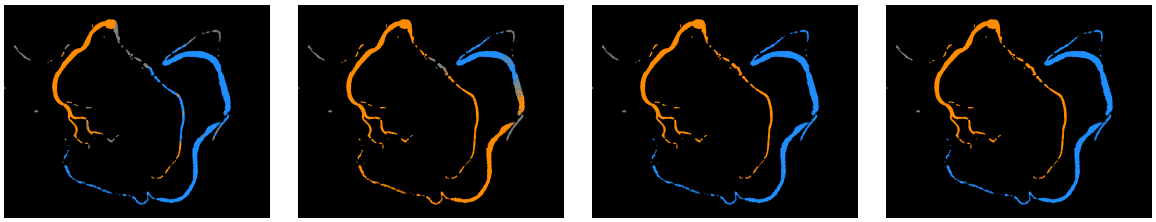
Threshold	**F1-score (%)**		
	MSG	**GAG**	**EVG**
0.5	95.20 (± 4.12)	95.48 (± 3.59)	95.90 (± 3.05)
0.8	95.16 (± 3.56)	94.76 (± 2.76)	95.90 (± 3.23)
0.9	94.62 (± 4.03)	93.23 (± 7.77)	95.42 (± 4.40)
0.95	90.06 (± 9.12)	83.65 (± 21.51)	94.20 (± 6.38)
0.97	76.53 (± 20.24)	67.88 (± 27.40)	90.88 (± 9.88)

(*a*) Undersegmentation results (radius methods) - mean (± std). Evaluated on point clouds created from prior U-Net segmentation with different binarization thresholds.



(*b*) U-Net segmentations with different binarization thresholds: orange - 0.5, blue - 0.8, green - 0.97

Figure 3: Undersegmantation benchmark



(*a*) MSG (radius) (*b*) GAG (radius) (*c*) EVG (radius) (*d*) GT

Figure 4: Qualitative example on undersegmented vessel point clouds with threshold 0.97: RCA (blue), LCA (orange), FP artefacts (grey).

Artificially disrupted vessels To further evaluate EVG robustness and generalization capabilities in more comprehensive benchmark we construct a dataset with artificially disrupted vessels that simulate disruptions caused by pathologies such as calcification or coronary artery stenosis. The datasets are created by generating discontinuities along arteries by choosing at random root points from artery skeletons and discarding all points in the specified radius around the point. Points are discarded only from CA vessels and for each

artery, there is the same number of disruptions generated. To control the process of generating disruptions, two parameters were introduced: radius specifying the size of disruption and number of disruptions. Point clouds are normalized to the interval $[-0.5, 0.5]$ and radii of disruptions are given as the values in the same space.

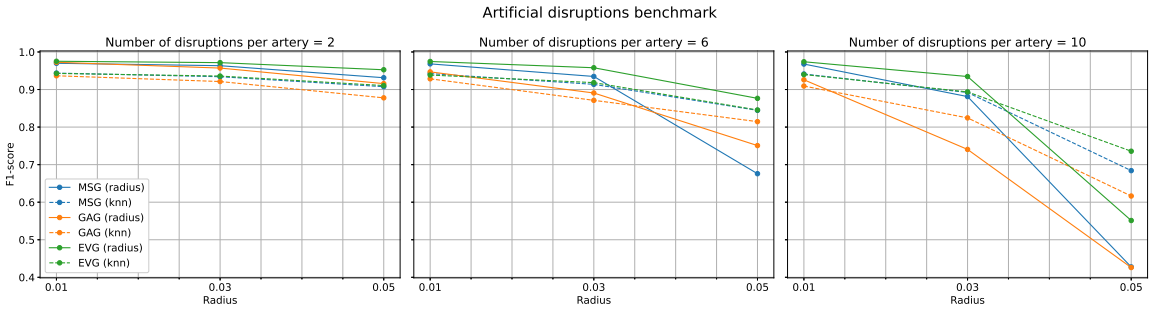


Figure 5: Artifical disruptions benchmark. F1-score is computed on skeletonized point clouds from datasets with varying values of radii and number of disruptions.

Figure 5 showcases how F1-score (skeleton metric) for each grouping strategy changes for different radii and numbers of disruptions. For the smallest number of disruptions, all methods achieve comparable results, although we can see that with larger radii EVG tends to behave better - ball query methods are better than KNN ones. With the number of disruptions set to 6, EVG with the ball query outperforms all other methods with F1-score not dropping below 87%. Other grouping strategies using the ball query decrease in performance drastically while the radius gets larger and drop to 75% F1-score or even lower. For the max radius, the KNN query for both MSG and GAG performs better, while still being worse than EVG with the ball query. For the largest disruption size and the most amount of them, methods using the KNN query perform better. Among them, EVG is the best method achieving over 72% F1-score. With more and larger disruptions the sparseness of a point cloud increases, thus the KNN, which adapts to such changes inherently, is more robust. GAG is outperformed by both MSG and EVG for almost all experiments. It can be caused by the fact that GAG is designed to facilitate class consistency among components while inherently being less robust towards finding disconnected artery components.

4. Summary

In this work, we tackle the post-processing task in CA segmentation called *vessel labeling*. We propose a novel grouping strategy on point clouds EVG, designed to perform point aggregation along vessels' direction. To further refine results and facilitate class consistency among connected components we propose to use a novel post-processing algorithm CMPV. Our approach outperforms previously proposed GAG and native PointNet++ MSG grouping strategies by achieving 97.88% accuracy on CA skeletons on our CTA dataset. Moreover, the proposed method shows greater generalization capabilities in the undersegmentation and artificial disruptions benchmarks.

Acknowledgements

The work conducted by Maciej Zieba was supported by the National Centre of Science (Poland) Grant No. 2020/37/B/ST6/03463. For the purpose of Open Access, the author has applied a CC-BY public copyright licence to any Author Accepted Manuscript (AAM) version arising from this submission.

References

Fabian Balsiger, Yannick Soom, Olivier Scheidegger, and Mauricio Reyes. Learning shape representation on sparse point clouds for volumetric image segmentation. *CoRR*, abs/1906.02281, 2019.

Yo-Chuan Chen, Yi-Chen Lin, Ching-Ping Wang, Chia-Yen Lee, Wen-Jeng Lee, Tzung-Dau Wang, and Chung-Ming Chen. Coronary artery segmentation in cardiac ct angiography using 3d multi-channel u-net, 2019.

Wing Keung Wing Keung Cheung Cheung, Robert Robert Bell Bell, Arjun Arjun Nair Nair, Leon J. Leon J. Menezes Menezes, Riyaz Riyaz Patel Patel, Simon Simon Wan Wan, Kacy Kacy Chou Chou, Jia Chen Chen, Ryo Ryo Torii Torii, Rhodri H. Rhodri H. Davies Davies, James C. James C. Moon Moon, Daniel C. Daniel C. Alexander Alexander, and Joseph Joseph Jacob Jacob. A computationally efficient approach to segmentation of the aorta and coronary arteries using deep learning. *Ieee Access*, 9:108873 – 108888, 2021.

Ozgun Cicek, Ahmed Abdulkadir, Soeren S. Lienkamp, Thomas Brox, and Olaf Ronneberger. 3d u-net: Learning dense volumetric segmentation from sparse annotation, 2016.

The MONAI Consortium. Project monai, December 2020. URL https://doi.org/10.5281/zenodo.4323059.

J.J.M. Cuppen. A divide and conquer method for the symmetric tridiagonal eigenproblem. *Numerische Mathematik. 36*, page 177–195, 1981.

Y. Eldar, M. Lindenbaum, M. Porat, and Y.Y. Zeevi. The farthest point strategy for progressive image sampling. *IEEE Transactions on Image Processing*, 6(9):1305–1315, 1997. doi: 10.1109/83.623193.

WA et al. Falcon. Pytorch lightning. URL https://github.com/PyTorchLightning/pytorch-lightning.

J.G.F Francis. The qr transformation, i. *The Computer Journal, 4(3)*, pages 265–271, 1961. doi: 10.1093/comjnl/4.3.265.

Jiafa He, Chengwei Pan, Can Yang, Ming Zhang, Yang Wang, Xiaowei Zhou, and Yizhou Yu. Learning hybrid representations for automatic 3d vessel centerline extraction. *Lecture Notes in Computer Science*, page 24–34, 2020. ISSN 1611-3349. doi: 10.1007/978-3-030-59725-2_3. URL http://dx.doi.org/10.1007/978-3-030-59725-2_3.

Weimin Huang, Lu Huang, Zhiping Lin, Su Huang, Yanling Chi, Jiayin Zhou, Jun-Mei Zhang, Ru San Tan, and Liang Zhong. Coronary artery segmentation by deep learning neural networks on computed tomographic coronary angiographic images. volume 2018, pages 608–611, 07 2018. doi: 10.1109/EMBC.2018.8512328.

C.G.J. Jacobi. Über ein leichtes verfahren, die in der theorie der säkularstörungen vorkommenden gleichungen numerisch aufzulösen. *Crelle's Journal (30)*, pages 51–94, 1846. doi: 10.1515/crll.1846.30.5.

Øyvind Kjerland. Segmentation of coronary arteries from ct-scans of the heart using deep learning. 2017.

Palaniappan Mirunalini, Chandrabose Aravindan, A. Thamizh Nambi, S. Poorvaja, and V Pooja Priya. Segmentation of coronary arteries from cta axial slices using deep learning techniques. *TENCON 2019 - 2019 IEEE Region 10 Conference (TENCON)*, pages 2074–2080, 2019.

Herman Müntz. Solution directe de l'équation séculaire et de quelques problèmes analogues transcendants. *Compte Rendu Acad, Paris*, pages 43–46, 1913.

LS. Pan, CW. Li, and SF. Su. Coronary artery segmentation under class imbalance using a u-net based architecture on computed tomography angiography images. *Sci Rep 11, 14493*, 2021. doi: https://doi.org/10.1038/s41598-021-93889-z.

Jungme Park, Carl Looney, and Hui-Chuan Chen. Fast connected component labeling algorithm using a divide and conquer technique. 01 2000.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32:8026–8037, 2019.

Charles R. Qi, Li Yi, Hao Su, and Leonidas J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space, 2017.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

Leslie N. Smith. A disciplined approach to neural network hyper-parameters: Part 1 - learning rate, batch size, momentum, and weight decay. *CoRR*, abs/1803.09820, 2018.

Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. Dynamic graph cnn for learning on point clouds, 2019.

Erik Wijmans. Pointnet++ pytorch. URL https://github.com/erikwijmans/Pointnet2_PyTorch.

Appendix A. Data

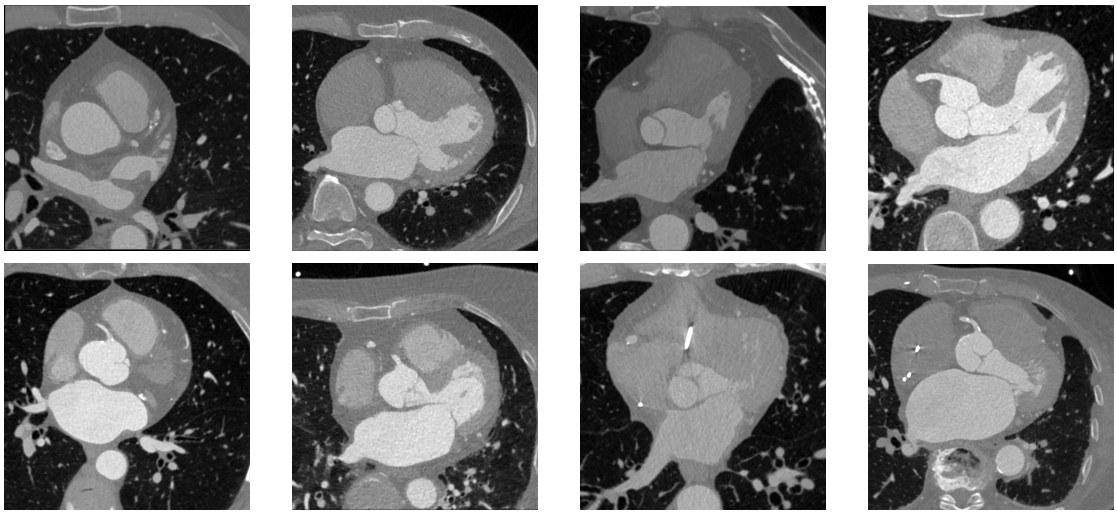


Figure 6: Examples of axials slices from the CCTA dataset.

Appendix B. Architecture details

Table 2: PointNet++ architecture that is used for all grouping methods. All groupings utilize the multi-scale grouping approach by defining more than one grouping area per block (`radii`, `nsamples`). Parameter `radii` defines radii of ball queries and `nsamples` numbers of neighbours to aggregate with KNN.

Architecture	**Parameters**	**Encoder**				
		Block 1	**Block 2**	**Block 3**	**Block 4**	**Block 5**
PointNet++ (radius)	npoint	2048	1024	256	64	32
	radii	[0.05, 0.1]	[0.1, 0.2]	[0.2, 0.4]	[0.4, 0.6]	[0.4, 0.8]
PointNet++ (knn)	npoint	2048	1024	256	64	32
	nsamples	[256, 512]	[64, 128]	[32, 64]	[16, 32]	[8, 16]

Table 3: EVG parameters. The parameter `prior nsamples` defines how many neighbours are aggregated with an initial KNN to estimate the eigenvector. The parameter `vector lengths` defines the length of grouping vectors.

Parameters	**Encoder**				
	Block 1	**Block 2**	**Block 3**	**Block 4**	**Block 5**
prior nsamples	[64, 64]	[16, 16]	[8, 8]	[8, 8]	[4, 4]
vector lengths	[0.01, 0.02]	[0.01, 0.02]	[0.01, 0.02]	[0.01, 0.02]	[0.01, 0.02]

Table 4: Training (inference) time comparison between methods. All models were trained on the NVIDIA GeForce RTX 3060 12 GB (require 11.5 GB of RAM).

Query	**MSG**	**GAG**	**EVG**
knn	2h 41min	3h 15min	5h 14min
radius	1h 7min (4.93s)	1h 10min (4.94s)	2h 25min (9.65s)

Appendix C. Additional results

Table 5: Dense metrics - mean (± std) for the evaluation on the CCTA dataset. Dense metrics are computed on point clouds sampled from the entire region of an artery.

Grouping	**Query + post-processing**	**Metrics (%)**			
		Accuracy	**F1-score**	**Precision**	**Recall**
MSG	knn	98.07 (± 3.37)	94.78 (± 10.13)	95.72 (± 9.62)	94.79 (± 9.20)
	knn + CMPV	98.54 (± 2.41)	95.57 (± 9.82)	96.59 (± 8.39)	95.14 (± 10.79)
	radius	99.10 (± 0.90)	97.56 (± 2.95)	97.52 (± 4.01)	97.72 (± 2.51)
	radius + CMPV	99.24 (± 0.86)	98.00 (± 2.24)	98.30 (± 2.70)	**97.86 (± 2.38)**
GAG	knn	98.55 (± 2.02)	95.56 (± 8.08)	97.28 (± 4.43)	94.79 (± 11.00)
	knn + CMPV	98.53 (± 2.22)	95.15 (± 10.07)	96.24 (± 9.22)	94.64 (± 11.65)
	radius	99.20 (± 0.80)	97.74 (± 2.73)	98.23 (± 2.65)	97.47 (± 3.12)
	radius + CMPV	99.22 (± 0.82)	97.88 (± 2.61)	98.57 (± 2.10)	97.48 (± 3.26)
EVG	knn	98.30 (± 2.14)	94.74 (± 9.22)	95.78 (± 7.94)	94.34 (± 11.03)
	knn + CPMV	98.78 (± 1.98)	95.95 (± 9.13)	97.13 (± 7.20)	95.27 (± 10.75)
	radius	99.24 (± 0.74)	97.98 (± 2.34)	98.45 (± 2.14)	97.70 (± 2.65)
	radius + CMPV	**99.30 (± 0.78)**	**98.05 (± 2.54)**	**98.75 (± 1.85)**	97.67 (± 2.87)

Table 6: Wilcoxon signed-rank test on F1-score of GAG vs EVG and MSG vs EVG for the experiment on the undersegmented predictions. The results are presented in p-values for the ball query (radius) methods. The higher threshold, the higher significance.

	F1-score p-value				
Threshold	**0.5**	**0.8**	**0.9**	**0.95**	**0.97**
EVG vs MSG	0.116	0.013	5.7e-4	1.2e-6	3e-7
EVG vs GAG	0.144	0.263	0.017	0.003	1.7e-5