Bitmap or Vector? A study on sketch representations for deep stroke segmentation

F Hähnlein, Y Gryaditskaya, A Bousseau

Bitmap or Vector? A study on sketch representations for deep stroke segmentation

F. Hähnlein, Y. Gryaditskaya and A. Bousseau

Université Côte d'Azur, Inria

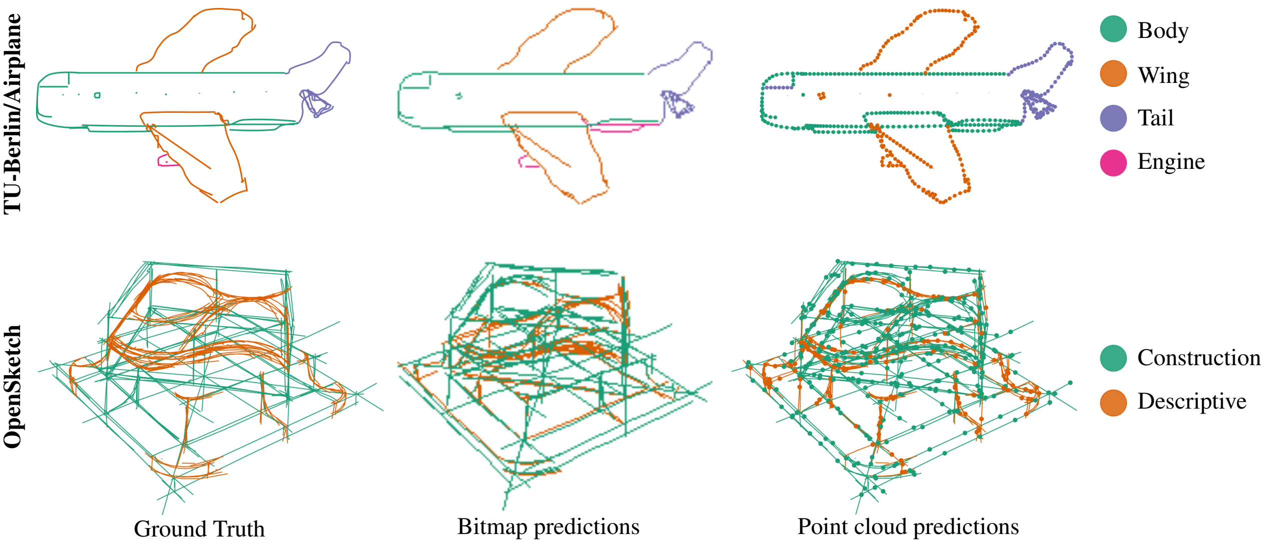

Figure 1 : *We compare bitmap and point-cloud representations on two sketch segmentation datasets. The annotated TU-Berlin dataset contains doodles made by novices (top). The OpenSketch dataset contains sketches made by professional designers (bottom).*

Abstract

Deep learning achieves impressive performances on image segmentation, which has motivated the recent development of deep neural networks for the related task of sketch segmentation, where the goal is to assign labels to the different strokes that compose a line drawing. However, while natural images are well represented as bitmaps, line drawings can also be represented as vector graphics, such as point sequences and point clouds. In addition to offering different trade-offs on resolution and storage, vector representations often come with additional information, such as stroke ordering and speed. In this paper, we evaluate three crucial design choices for sketch segmentation using deep-learning : which sketch representation to use, which information to encode in this representation, and which loss function to optimize. Our findings suggest that point clouds represent a competitive alternative to bitmaps for sketch segmentation, and that providing extra-geometric information improves performance.

Keywords : Sketch segmentation, Deep-learning, Point clouds, Vector graphics

1. Introduction and Related Work

Digital drawings can be represented either as bitmaps or vector graphics. Bitmaps are easily obtained from paper sketches or painting software. The regular structure of bitmaps also make them well suited to a plethora of image processing algorithms originally developed for natural images. However, bitmaps often do not store information about the drawing creation process, such as stroke connectivity (structural information) and stroke ordering (temporal information). In contrast, vector graphics can represent each individual stroke of a drawing, for instance acquired with a pen tablet. Yet, few algorithms exist that take advantage of this extra information.

In this work, we compare bitmap and vector representations of sketches for the particular task of *sketch segmentation*, where the goal is to assign different labels to strokes in a drawing, for instance to recognize parts of objects [HFL14b], to edit the drawing [NSS*12], or to perform geometric analysis [IBB15]. We focus our evaluation on deep learning algorithms, which achieve state-of-the art performances on similar segmentation tasks for natural images and 3D shapes, yet have no established representation for sketch processing. We also evaluate the impact of feeding the deep networks with additional information like stroke opacity, time and speed.

Inspired by the large body of work on natural image segmentation [RFB15, IZZE16], a number of sketch-segmentation algorithms rely on deep convolutional networks to process line drawings in a bitmap form [HFL14a, LFT18]. In particular, we include the recent method of Li et al. [LFT18] in our study, which complements an encoder-decoder convolutional network with a post-processing graph-cut optimization to favor large segments.

As an alternative to bitmaps, Ha et al. [HE17] proposed to process a line drawing as a sequence of points to account for the order in which the strokes have been drawn. They employ a recurrent neural network architecture, as is common in related domains like natural language processing. While the original work was targeting sketch generation, Wu et al. [WQLY18] extends it to perform segmentation. However, vanilla recurrent neural networks have been shown to have limited capacity [CVMBB14], which might prevent this approach to scale to more complex sketches than simple doodles. For this reason, we didn't include this family of methods in our study, and keep their evaluation for future work.

Finally, a few authors have proposed to represent vector line drawings as point clouds [WCZ18, LBS*18, WLW*19]. Such methods build on deep network architectures originally developed by the geometry processing community [BBL*17]. In our study, we adopt the architecture of Wang et al. [WSL*18], which includes a dynamic convolutional operator to account for both global and local features of the point cloud.

Most of the methods cited above rely on a standard cross-entropy loss for training. However, this loss is known to suffer from class imbalance, which is common in segmentation. In our study, we compare performances obtained with different losses, including the cross-entropy loss, the Dice loss [SLV*17] and Mean False Error [WLW*16], of which the two latter have been designed to cope with class imbalance.

In summary, we evaluate three key aspects of deep learning-based methods for sketch segmentation :

— Firstly, we compare between different representations for sketches, namely, between bitmaps and point clouds.
— Secondly, we compare the added value of different input features exceeding mere geometry, such as stroke opacity, time and speed. As this information becomes increasingly available in digital sketches, it is important to assess its usefulness for the development of novel methods and architectures.
— Thirdly, we test the ability of different loss functions to cope with imbalanced datasets, a common scenario for sketch segmentation.

2. Experiments

Our goal is to compare different sketch representations and to evaluate the added value of extra-geometric information in the context of semantic segmentation. We first describe the sketch segmentation datasets we used for this study, before detailing the deep learning methods we trained on those datasets – one based on a bitmap representation and one based on a point cloud representation.

2.1. Datasets

TU-Berlin Airplanes. Recent sketch datasets such as TU-Berlin [EHA12] and QuickDraw ! [HE17] contain sketches in vector format, which makes them good candidates for comparing methods using different representations since vector images can be easily rasterized. In particular, we adopt the TU-Berlin [EHA12] dataset, which has been partially annotated for stroke-based semantic segmentation [LFT18]. Each sketch represents an object of a specific class, drawn by a novice using a web-based interface. The semantic stroke labels are based on the object components, as shown in Figure 1(top) where the airplane is decomposed into body, wings, tail, engines. However, this dataset only contains 80 sketches per object class, which is insufficient for training a deep neural network. Li et al. [LZZ*18] addressed this challenge by collecting 3D models segmented with the same labels as the TU-Berlin classes, which they rendered from multiple viewpoints to create synthetic sketches for training. We followed the same approach and used their 3D models of the *Airplane* class to build our training dataset. However, while Li et al. used OpenGL to render bitmap sketches, we used Blenders Freestyle [GTDS10] to export the contour renderings as SVG files. In total, our training dataset contains 5000 contour renderings. Our test dataset contains the 80 sketches of the *Airplane* class from the annotated TU-Berlin dataset.

OpenSketch. The TU-Berlin dataset does not contain any information about stroke opacity, time and speed. This limitation motivated us to complement it with the recently pusblished OpenSketch dataset [GSH*19], which contains around 200 product design sketches made by professionals and design students. The sketches were collected with a pen tablet and contain individual stroke trajectories with per-point pressure and time information. Each stroke is labeled according to a hierarchical taxonomy of drawing techniques used by product designers. While this taxonomy defines several dozen classes, we simplify the task by focusing on the two classes defined by the coarsest level of the hierarchy – construction and descriptive lines [GSH*19]. We then split

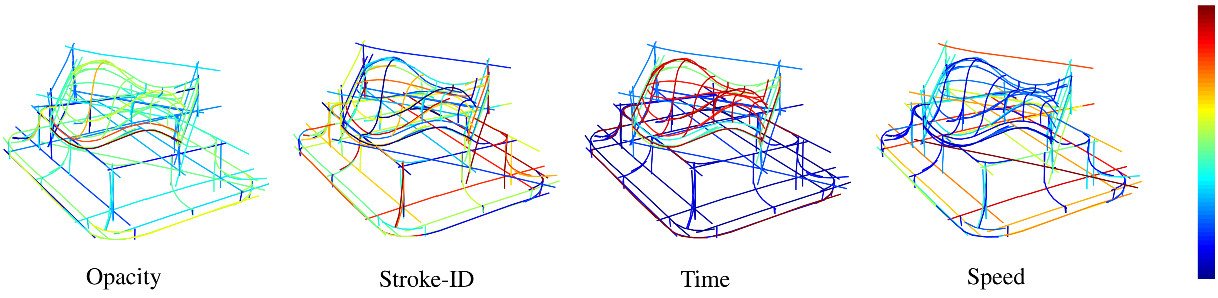

Figure 2 : *Visualization of the additional information we consider : opacity, stroke-ID, time, speed.*

the dataset into complementary subsets of designers and objects to form our training and test sets. The sketches of the test set come from the object categories *bumps*, *tubes* and *wobble surface* and were made by the designers *student2*, *student5*, *student9*, *Professional3* and *Professional5*. The sketches of the training set come from different object categories (*flange, hairdryer, house, mixer, mouse, potato-chip, shampoo bottle, vacuum cleaner* and *waffle iron*) and were made by different designers (*Professional1, Professional2, Professional4, Professional6, student1, student3, student4, student6, student7* and *student8*). Given this setting, neither the depicted objects, nor the designers overlap between the training and the test data. For data augmentation, we apply local and global sketch deformation by applying moving least squares [SMW06] to stroke endpoints and to the convex hull respectively, inspired by [YYL*17]. In total, the training set contains around 1100 sketches and the test set around 300 sketches.

Rasterization and point sampling. To use these two datasets as bitmaps and as point clouds, we rasterize and sample them respectively. The rasterization is conform to [LFT18] at a resolution of 256×256 pixels. In addition, we also rasterize channels featuring the stroke-id and, in the case of OpenSketch, opacity, time and speed, as shown in Figure 2. The purpose of the stroke-id is to provide cues about the stroke based structure of the sketch, i.e., which points belong to which stroke. To distinguish this structural information from stroke ordering, we randomly shuffle the stroke-ids within a sketch. To generate point clouds, we uniformly sample a fixed number of points (500) along the 1D curve described by appending all strokes of the sketch. Each point corresponds to a multi-dimensional vector containing the same information as the bitmap channels.

2.2. Bitmap architecture

We evaluate the performance of bitmap-based sketch segmentation methods using the convolutional network recently proposed by Li et al. [LFT18]. The network is composed of an encoder and a decoder linked by skip connections, similar to the popular U-Net originally proposed for image segmentation [RFB15]. The predicted segmentation map is subsequently processed by a graph-cut optimization procedure to promote spatial smoothness.

We used the network implementation provided by the authors, but reimplemented the training procedure, which might be a reason for slight differences in the final performance. Otherwise, we keep the hyperparameter settings from the original paper.

2.3. Point cloud architecture

To the best of our knowledge, there is no specialized point cloud architecture for sketch segmentation. We therefore select a recent *3D* point cloud segmentation network, namely Dynamic Graph CNN [WSL*18]. Extending the seminal PointNet [QSMG17], Dynamic Graph CNN dynamically constructs k-nearest neighbor graphs to compute convolutions on unstructured point clouds. We use the original implementation and hyperparameter settings.

2.4. Loss function

The *Airplane* and the OpenSketch datasets described in Section 2.1 have an imbalance ratio of 2.1 and 1.42 respectively, where the imbalance ratio is defined as the ratio of the number of samples of the majority class over the number of samples of the minority class. Whereas no loss function completely remedies class imbalance, some are more prone to overfitting than others. In particular, the popular weighted cross-entropy loss used by previous sketch segmentation networks can lead to bias in favor of the majority class.

We tested the ability of three different loss functions to cope with class imbalance : weighted cross-entropy loss, weighted cross-entropy with Dice loss [SLV*17] and Mean False Error [WLW*16]. We only trained Dyncamic Graph CNN on OpenSketch for this experiment, and used the best-performing loss for other experiments.

3. Results

For performance measures we use the percentage of correctly predicted points, i.e., the model segmentation accuracy, also called the *P-metric* [HFL14a, LFT18]. Since we want to compare the results of two different representations of the data, bitmaps and point clouds, we sample the predictions from bitmaps at the point cloud coordinates.

The results of our experiment on the TU Berlin and OpenSketch datasets are shown in Table 1 and in Table 2 respectively. First, we use only geometric data, i.e., the rasterized binary sketch for bitmaps and point coordinates for

Architecture	G.	G.SID.
[LFT18]	65%	68%
DGCNN [WSL*18]	72%	70%

Table 1 : *Results for the Airplanes dataset and different input features : Geometric (G) and Stroke-ID (SID).*

Architecture	G.	G.SID.	G.SID.O.	G.SID.O.T.	G.SID.O.T.S.	SID.O.T.
Wang et al. [LFT18]	65%	67%	71%	70%	70%	-
DGCNN [WSL*18]	57%	71%	71%	73%	72%	-
SVM [GSH*19]	-	-	-	-	-	76%

Table 2 : *Results for the OpenSketch dataset and different input features : Geometric (G), Stroke-ID (SID), Opacity (O), Time (T) and Speed (S).*

point clouds. The results are shown in the column entitled *Geometric*. Starting from there, we add progressively more input information : the stroke-ids, opacity, time and speed. We add the last three features only for OpenSketch, since the dataset from [LZZ*18] does not include this kind of information.

Additionally, we report the results obtained by the SVM classifier used by Gryaditskaya et al. [GSH*19] for stroke classification (Table 2). We re-trained the classifier on the binary classification task described in Section 2.1. Their handcrafted stroke representations include the stroke-id, opacity and time. No geometric information is taken into account.

Impact of representation. The two architecture we considered do not perform equally well on the different tasks. When using only geometry, the bitmap architecture performs better than the point cloud architecture on the *OpenSketch* dataset (Table 2), but worse on the *Airplanes* dataset (Table 1). However, the performance gap between the two representations reduces once additional information is provided, including stroke-id for the *Airplanes* dataset and various combinations of opacity, time and speed for the *OpenSketch* dataset.

Surprisingly, both deep learning architectures are outperformed by the SVM classifier on the *OpenSketch* dataset (Table 2, last column), despite the fact that this classifier ignores spatial information. This high performance of SVM might be due to the specific nature of the task, since the usage of construction and descriptive lines in product design sketches is strongly correlated with non-geometric features like time, as discussed by Gryaditskaya et al. [GSH*19].

Impact of input features. In the case of the *Airplanes* dataset, adding the stroke-id improves a bit the results of the bitmap architecture, but degrades slightly the ones of the point cloud architecture. This effect is possibly due to the domain gap between the training set and the test set, since the human-made sketches from the TU-Berlin dataset tend to have different stroke partitions than the synthetic sketches, where each continuous contour is represented by a single stroke.

On the other hand, on the *OpenSketch* dataset, the performance of both architectures improves when structural information (stroke-id) and information about the drawing process (opacity, time and speed) is available. The point cloud architecture even outperforms the bitmap architecture when all these features are provided.

Impact of losses. The results shown in Table 3 illustrate the effectiveness of different loss functions to cope with class imbalance. Whereas all losses lead to comparable *global accuracies*, they result in different *per-class accuracies*.

In our experiment, the weighted cross-entropy loss leads to biased results in favor of the majority class. While the weighted cross-entropy loss with Dice [SLV*17] compensates in part for this behavior, it still performs worse than the Mean False Error loss [WLW*16]. These results reemphasize the importance of choosing the right metric for training a neural network, especially when training on imbalanced data, as it is often the case in sketch segmentation.

4. Conclusion

Sketch segmentation is a challenging task. Architectures developed for doodles do not necessarily perform equally well for more elaborate sketches. Depending on the application, different input features might be available and the design decision for the segmentation method should be adapted accordingly.

In this paper, we have retrained and compared two different deep-learning based sketch segmentation methods on two different datasets. We have evaluated the usefulness of additional features which inform about the structure of the sketch (stroke-id) and about the drawing process itself (opacity, time and speed). If available, this information is easy to integrate into existing architectures by augmenting the number of input channels. For future work, it would be interesting to investigate if sketch classification and recognition tasks also benefit from those features.

An important insight to take away from this paper is that an bitmap-based method does not automatically outperform its point cloud counterpart. While bitmaps benefit from established deep convolutional architectures, they suffer from finite resolution. With the recent and rapid development of Geometric Deep-Learning, novel convolution operations and architectures developed for point clouds should be considered for sketch processing. Finally, additional information has proven to be useful for data-driven sketch analysis and we hope that it will be increasingly present in future sketch datasets.

Loss	Minority Class Acc.	Majority Class Acc.	Global Acc.
Cross-Entropy	51%	85%	71%
Cross-Entropy with Dice	61%	80%	72%
Mean False Error	70%	70%	70%

Table 3 : *We train DGCNN [WSL*18] on OpenSketch using three different loss functions and report per-class and global accuracies.*

Références

[BBL*17] Bronstein M. M., Bruna J., LeCun Y., Szlam A., Vandergheynst P. : Geometric deep learning : going beyond euclidean data. *IEEE Signal Processing Magazine. Vol. 34*, Num. 4 (2017), 18–42.

[CVMBB14] Cho K., Van Merriënboer B., Bahdanau D., Bengio Y. : On the properties of neural machine translation : Encoder-decoder approaches. *arXiv preprint arXiv :1409.1259* (2014).

[EHA12] Eitz M., Hays J., Alexa M. : How do humans sketch objects ? *ACM Trans. Graph.. Vol. 31*, Num. 4 (2012), 44–1.

[GSH*19] Gryaditskaya Y., Sypesteyn M., Hoftijzer J. W., Pont S., Durand F., Bousseau A. : Opensketch : A richly-annotated dataset of product design sketches.

[GTDS10] Grabli S., Turquin E., Durand F., Sillion F. X. : Programmable rendering of line drawing from 3d scenes. *ACM Transactions on Graphics (TOG). Vol. 29*, Num. 2 (2010), 18.

[HE17] Ha D., Eck D. : A neural representation of sketch drawings. *arXiv preprint arXiv :1704.03477* (2017).

[HFL14a] Huang Z., Fu H., Lau R. W. : Data-driven segmentation and labeling of freehand sketches. *ACM Transactions on Graphics (TOG). Vol. 33*, Num. 6 (2014), 175.

[HFL14b] Huang Z., Fu H., Lau R. W. H. : Data-driven segmentation and labeling of freehand sketches. *ACM Trans. Graph.. Vol. 33*, Num. 6 (novembre 2014), 175 :1–175 :10.

[IBB15] Iarussi E., Bommes D., Bousseau A. : Bendfields : Regularized curvature fields from rough concept sketches. *ACM Transactions on Graphics* (2015).

[IZZE16] Isola P., Zhu J.-Y., Zhou T., Efros A. A. : Image-to-image translation with conditional adversarial networks. *arxiv* (2016).

[LBS*18] Li Y., Bu R., Sun M., Wu W., Di X., Chen B. : Pointcnn : Convolution on x-transformed points. In *Advances in Neural Information Processing Systems* (2018), pp. 820–830.

[LFT18] Li L., Fu H., Tai C.-L. : Fast sketch segmentation and labeling with deep learning. *IEEE computer graphics and applications. Vol. 39*, Num. 2 (2018), 38–51.

[LZZ*18] Li L., Zou C., Zheng Y., Su Q., Fu H., Tai C.-L. : Sketch-r2cnn : An attentive network for vector sketch recognition. *arXiv preprint arXiv :1811.08170* (2018).

[NSS*12] Noris G., Sỳkora D., Shamir A., Coros S., Whited B., Simmons M., Hornung A., Gross M., Sumner R. : Smart scribbles for sketch segmentation. In *Computer Graphics Forum* (2012), vol. 31, Wiley Online Library, pp. 2516–2527.

[QSMG17] Qi C. R., Su H., Mo K., Guibas L. J. : Pointnet : Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2017), pp. 652–660.

[RFB15] Ronneberger O., Fischer P., Brox T. : U-net : Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention* (2015), Springer, pp. 234–241.

[SLV*17] Sudre C. H., Li W., Vercauteren T., Ourselin S., Cardoso M. J. : Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations. In *Deep learning in medical image analysis and multimodal learning for clinical decision support*. Springer, 2017, pp. 240–248.

[SMW06] Schaefer S., McPhail T., Warren J. : Image deformation using moving least squares. In *ACM transactions on graphics (TOG)* (2006), vol. 25, ACM, pp. 533–540.

[WCZ18] Wang X., Chen X., Zha Z. : Sketchpointnet : A compact network for robust sketch recognition. In *2018 25th IEEE International Conference on Image Processing (ICIP)* (2018), IEEE, pp. 2994–2998.

[WLW*16] Wang S., Liu W., Wu J., Cao L., Meng Q., Kennedy P. J. : Training deep neural networks on imbalanced data sets. In *2016 international joint conference on neural networks (IJCNN)* (2016), IEEE, pp. 4368–4374.

[WLW*19] Wang F., Lin S., Wu H., Li H., Wang R., Luo X., He X. : Spfusionnet : Sketch segmentation using multi-modal data fusion. In *2019 IEEE International Conference on Multimedia and Expo (ICME)* (2019), IEEE, pp. 1654–1659.

[WQLY18] Wu X., Qi Y., Liu J., Yang J. : Sketchsegnet : A rnn model for labeling sketch strokes. In *2018 IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP)* (2018), IEEE, pp. 1–6.

[WSL*18] Wang Y., Sun Y., Liu Z., Sarma S. E., Bronstein M. M., Solomon J. M. : Dynamic graph cnn for learning on point clouds. *arXiv preprint arXiv :1801.07829* (2018).

[YYL*17] YU Q., YANG Y., LIU F., SONG Y.-Z., XIANG T., HOSPEDALES T. M. : Sketch-a-net : A deep neural network that beats humans. *International journal of computer vision. Vol. 122*, Num. 3 (2017), 411–425.