LSTM Architectures for Facade Structure Completion

This is a self-archieved version of a paper that appeared in the Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (www.grapp.visigrapp.org)
DOI: 10.5220/0010194400150024
Compiled February 23, 2021

Simon Hensel
Institute for Pattern Recognition
Niederrhein University of Applied Sciences
Reinarzstrasse 49, Krefeld, Germany
simon.hensel@hs-niederrhein.de

Steffen Goebbels
Institute for Pattern Recognition
Niederrhein University of Applied Sciences
Reinarzstrasse 49, Krefeld, Germany
steffen.geobbels@hs-niederrhein.de

Martin Kada
Institute of Geodesy and Geoinformation Science
Technical University of Berlin
Kaiserin-Augusta-Allee 104-106, Berlin, Germany
martin.kada@tu-berlin.de

Abstract

3D city models are often generated from oblique aerial images and photogrammetric point clouds. In contrast to roof surfaces, facades can not directly be reconstructed in a similar high level of quality from this data. Distortions of perspective might appear in images, due to the camera angle. Occlusions and shadowing occur as well. Objects, such as windows and doors, will have to be detected on such data if facades are to be reconstructed. Although one can use inpainting techniques to cover occluded areas, detection results are often incomplete and noisy. Formal grammars can then be used to align and add objects. However, it is difficult to find suitable rules for all types of buildings. We propose a post-processing approach based on neural networks to improve facade layouts. To this end, we applied existing Recurrent Neural Network architectures like Multi-Dimensional Long Short-term Memory Network and Quasi Recurrent Neural Network in a new context. We also propose a novel architecture, the Rotated Multi-Dimensional Long Short Term Memory. In order to deal with two-dimensional neighborhoods this architecture combines four two-dimensional Multi-Dimensional Long Short-term Memory Networks on rotated images. We could improve the quality of detection results on the Graz50 data set.

***K*eywords** Deep Learning, LSTM, Facade Reconstruction, Structure Completion

1 INTRODUCTION

In most cases, facades cannot be reconstructed in the same high quality as roofs due to their limited visibility on aerial photographs. Camera angles can lead to perspective distortions. In addition, occlusions and shadows can be caused by various objects. In order to be able to reconstruct facades, objects such as windows and doors have to be recognized. Neural networks for object detection and instance segmentation, see Section 2, search for individual object instances, but do not consider overall layout patterns. Thus, relationships between windows are not considered.

One can apply inpainting techniques on input images or use model knowledge to suggest missing objects. Inpainting is a standard computer vision technique to extrapolate visual information to fill or replace damaged, fuzzy or missing areas in images. As shown in (Bertalmio et al., 2000), inpainting is effective in removing texts or other objects present in images. Since inpainting is performed on images, it has to be used as a processing step prior to segmentation or detection of facade objects. Therefore, it does not benefit from knowledge gathered by object detection.

Techniques based on model knowledge can benefit from detection results and can be applied in a post-processing step. For building reconstruction, split grammars are described in (Wonka et al., 2003). Such grammars are a collection of rules by which object placement and orientation can be described. They allow adding missing facade objects and even to generate facades from scratch procedurally. However, grammar rules have to fit for a given building style. Unique facade styles might require individual rules that have to be provided manually. In a situation in which it is difficult to define rules, machine learning is well suited.

Instead of defining grammars, we propose to apply Recurrent Neural Networks (RNNs) with Long Short Term Memory (LSTM). They are typically used for time-dependent, one-dimensional input data. A main application is speech recognition. Our aim is to restore missing facade objects based on available, incomplete detection results. These results are represented by bounding boxes. By extending edges of these bounding boxes to lines, we obtain an irregular rectangular lattice (IRL). This is a collection of horizontal and vertical lines where the distances between the lines can vary. We merge parallel lines into a single one if lines are within a threshold distance that corresponds with the average image resolution, see Section 3.1. Areas that are bounded but not intersected by the lines are called cells. In accordance with given detection results, cells are initially labeled either as background or as belonging to a

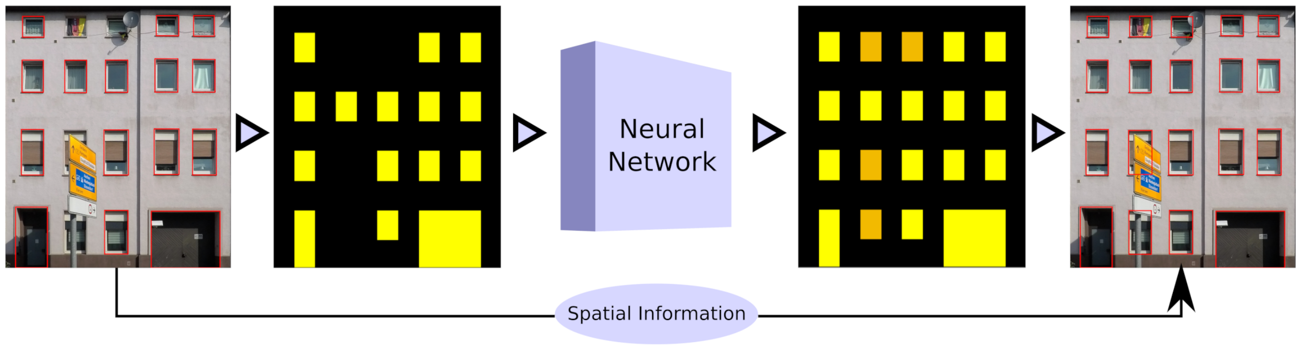


Figure 1: Workflow for facade structure completion: Due to a signpost, a satellite dish and a flag, four windows could not be detected. They were added with a LSTM.

facade object class (windows, doors). Now the task is to correct labels of cells that belong to missing objects. The proposed workflow is shown in Figure 1. To this end, we compare the proposed Rotated Multi-Dimensional LSTM (RMD LSTM, see Section 3) with Quasi Recurrent Neural Network (QRNN), see (Bradbury et al., 2016), Section 4.

2 RELATED WORK

For the 3D reconstruction of facades their components have to be recognized. If images of the facades are available (e.g. from oblique aerial images), this can be done using neural networks. During the last decade, Convolutional Neural Networks (CNNs) have become a standard tool for image-based object detection and segmentation. For image segmentation on low-performance hardware YOLO (Bochkovskiy et al., 2020) is widely used. By reducing the number of weights through adding connections between layers and outputs, ResNET (He et al., 2016) was a milestone in deep learning. In the field of object detection is Mask R-CNN (He et al., 2017) an enhancement of a regional convolutional network with added segmentation capabilities. RetinaNet (Lin et al., 2017) introduced the concept of focal loss to distinguish between fore- and background. These image segmentation or object detection networks can be used within a 3D building reconstruction framework. In this regard, CityGML is the standard open data model for semantic 3D city models, see (Gröger et al., 2007).

We use bounding boxes of detected objects to reconstruct facades. To this end, two problems have to be solved: Boxes might not be aligned properly and boxes of occluded objects might be missing. The alignment problem can be seen as a combinatorial optimization problem, see (Hensel et al., 2019) and (Hu et al., 2020). In this paper, we deal with adding missing boxes. Image inpainting techniques as well as the application of formal grammars can be applied to estimating such missing information. The given paper discusses a third deep-learning-based method that can be applied to facade layouts.

Two inpainting methods that were used early are diffusion-based or example-based, cf. (Guillemot and Le Meur, 2014). Diffusion-based inpainting with smoothing priors is capable of repairing sparsely distributed small holes, but fails in the event of major disturbances. Example-based inpainting is able to extend textures into larger areas that need to be filled. However, it does not preserve the edges, which are an important feature of facade objects. Both methods try to keep simple textures instead of detecting more complex structures. Therefore, they are not suitable for facades. This issue is addressed in (Dai et al., 2013). The authors use a Random Forest-based approach to obtain a semantic segmentation. Edges of segment borders are used to define an IRL. Corresponding cells are initially labeled based on semantic segmentation similar to our approach that is based on bounding boxes, see Section 3.1. The IRL is interpreted as an undirected graph to define a graph labeling problem. The labeling is optimized by minimizing an energy function. This function measures the image and structural consistency. In contrast to the measurement of structural consistency, facade objects are clustered in our approach.

The article (Huang et al., 2014) deals with another inpainting algorithm that is applied in connection with the reconstruction of facade structures. This approach is based on line segments of edges. Corresponding lines might intersect in vanishing points. The algorithm detects all vanishing points and classifies line segments according to their vanishing point. Areas that are covered by line segments belonging to two vanishing points might be part of a 3D plane. The knowledge of planes is then used to continue textures. The method is implemented as a random search algorithm supported by various cost functions based on appearance, guidance, orthogonal direction and proximity.

The introduction of Generative Adversarial Networks (GANs), see (Goodfellow et al., 2014) is a milestone in inpainting techniques, see for example the application of Wasserstein GAN (Arjovsky et al., 2017) in (Yu et al., 2018). A more recent example of image inpainting with GANs is EdgeConnect (Nazeri et al., 2019). Here, edge images serve as GAN inputs.

Typically, it is necessary to specify the area that is to be filled by an inpainting algorithm. This additional complexity does not occur with model-based techniques. GANs tend to be unstable and are therefore harder to train, see (Arjovsky and Bottou, 2017). That is one reason why we use more robust LSTM architectures.

In contrast to data-based approaches like inpainting, model based methods consider typical facade layouts that can be either learned (as proposed here) or explicitly given in terms of rules by formal grammars. The first used formal grammars for

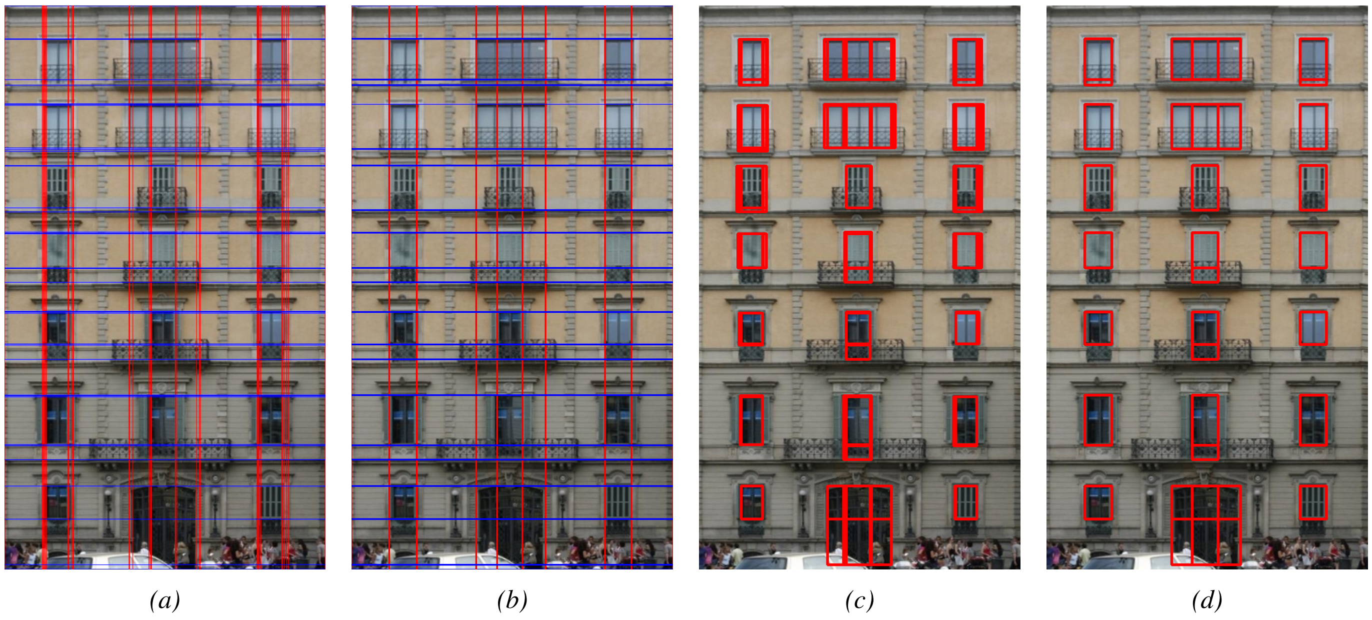


Figure 2: Images *(a)* and *(b)* show an IRL representation of the facade from the CMP data set. Images *(c)* and *(d)* show corresponding bounding boxes. IRL in *(b)* was generated by merging lines of the IRL in *(a)*.

facade understanding and reconstruction were split grammars (Wonka et al., 2003). In the article (Teboul et al., 2011) shape grammars to generate probabilities for facade object classes are described. The initial probability map for terminal symbols is provided by a discriminative model. To further optimize labels, the model is used in a Markov Decision Process. Reinforced learning is applied to optimize the model. A system consisting of immediate and cumulative rewards is used to obtain facade segmentations which are close to reality. A greedy-type algorithm computes final results.

Also, formal grammars are not limited to be used on 2D images, see (Dehbi et al., 2016). Utilizing point cloud data, values along horizontal and vertical lines are cumulated with a Kernel Density Estimation. Lower values correspond with the occurrence of doors and windows. On those values, weighted attribute context-free grammars are applied to refine a facade model. In a weighted attributed context-free grammar, attributes define semantic rules and these rules are weighted with probabilities.

Since we do not want to define grammar rules manually, we apply deep learning. The essence of RNN and LSTM architectures is to provide a neural network with memory and previous predictions, cf. (Sherstinsky, 2020). These networks are mostly used for recognition of speech, text or generally one-dimensional data, cf. (Salehinejad et al., 2017) and (Mtibaa et al., 2020). The ability to process one-dimensional data with memory gives them high capabilities over one-dimensional CNNs, see (Zhang and Wang, 2016).

Extensions of LSTMs are Multi-Dimensional LSTM (Graves et al., 2007), on which the new Rotated Multi-Dimensional LSTM is built upon, Quasi RNN (Bradbury et al., 2016) and Grid LSTM (Kalchbrenner et al., 2015). To our knowledge, LSTMs have not been applied to the problem of facade reconstruction so far. Due to memory connections and a two-dimensional input serialized to one dimension, RNN and LSTM architectures have a high demand for hardware resources. Especially for Grid LSTM this causes problems. In Section 3.1 we explain how we reduce the size of data so that it can be processed by an RNN or LSTM.

3 RECURRENT NEURAL NETWORKS FOR PATTERN COMPLETION

We consider the ability of RNNs to utilize predictions from the past (processed spatial regions) to learn facade layouts. Our contribution consists of

- a workflow for refining facade object detection
- the Rotated Multi-Dimensional LSTM for facade completion and object recommendation.
- a comparison of outcomes of Quasi RNN, RMD LSTM (and MD LSTM).

3.1 Data Preparation

With the CMP facade data set (Tyleček and Šára, 2013) and the Graz50 data set (Riemenschneider et al., 2012) we used two different data sets to train and evaluate the RNN architectures. The CMP data set provides 606 facade images with ground truth information for object detection and segmentation. The Graz50 data set consists of 50 images with corresponding annotations for segmentation. The larger CMP data set was used for training. Images of one data set belong to a different facade style than images of the second set do. Thus, we avoided effects of overfitting, and we were able to demonstrate the ability to complete facade layouts of arbitrary facade types. Since we wanted to improve already detected facade layouts, we did not work with the RGB facade images of both data sets but we only considered corresponding ground truth information. We focused on windows and doors and ignored other classes of facade objects. Either ground truth of these data sets or detection results define position and size of windows and doors. This information can be interpreted as an IRL, see Section 1. Then each cell of the IRL is assigned to an entry of a matrix M. For example, the facade in Figure 1 is represented by

$$M = \begin{bmatrix} 0&0&0&0&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&1&0&1&0\\ 0&0&0&0&0&0&0&0&0&0&0\\ 0&1&0&1&0&1&0&1&0&1&0\\ 0&0&0&0&0&0&0&0&0&0&0\\ 0&1&0&0&0&1&0&1&0&1&0\\ 0&0&0&0&0&0&0&0&0&0&0\\ 0&1&0&0&0&1&0&1&1&1&0\\ 0&1&0&0&0&0&0&1&1&1&0 \end{bmatrix}. \qquad (1)$$

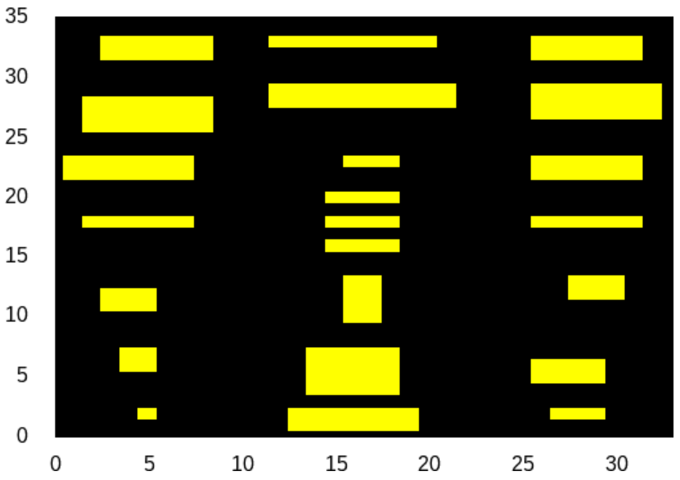
However, these derived IRLs result in matrices that are by far too large to be used directly as input of RNNs. Therefore, we further simplify IRLs by merging grid lines. We iterate through all lines, determine for each line all other lines of the same orientation within a distance of at most 8 pixels and then replace them with a mean line. Figure 2 *(b)* shows an IRL and the effect of merging close lines in comparison with Figure 2 *(a)*. Figures 2 *(c)* and 2 *(d)* represent the corresponding bounding box representation. Merging close lines improves simplicity and size of generated matrices, see Figure 3, without changing the facade layout significantly. Note that the IRLs Figure 2 fit with matrices in Figure 3. Figure 3 also shows that merging can lead to patterns and symmetry. Using the simplified IRL we generate a two-dimensional matrix consisting of zeroes (cell belongs to background) and ones (cell covered by a facade object). Spatial information for restoring bounding boxes, i.e. the IRL, is stored separately. The combination of matrix and IRL allows us to reconstruct bounding boxes. Depending on the complexity of the facade we are able to reduce a label image to a matrix with a size between 10×10 and 100×100 entries. We restrict ourselves to network inputs of size 25×25. To fit matrices into this format, we experimented with scaling. To obtain good results however, we only worked with facades that lead to matrices which did not exceed input size. This are 359 facades of CMP and 46 facades of Graz50 data. We applied zero padding. Whereas MD LSTM and RMD LSTM allow 2D matrix input, matrices have to be serialized for QRNN.

3.2 *Workflow*

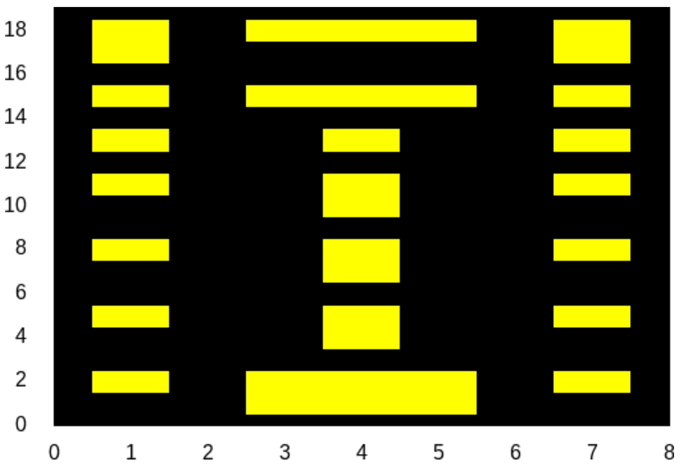
Merging lines of the IRL leads to aligned bounding boxes. However, both in an additional pre- and post-processing step, one can apply the algorithms in (Hensel et al., 2019) or (Hu et al., 2020) to further improve alignments. Between these steps, facade structures have to be completed by adding missing facade objects. For building , training and testing of neural networks we use Tensorflow (Abadi et al., 2015) and provide source code [1]. To this end, we propose the following workflow.

Training. At first, we process the object detections into a 2D matrix, like explained in Section 3.1. Randomly generated data

[1]Source code available to the general public at `https://github.com/SimonHensel/LSTM-Facade-Completion`



(a) Matrix representation of IRL before merging of lines within threshold distance.



(b) Resulting Matrix with merged horizontal and vertical lines.

Figure 3: Impact of merging horizontal and vertical grid lines in the IRL of Figure 2*(a)*. Note that axes are scaled differently, such that the resulting matrix of the simplified IRL is much smaller.

showed great potential in other work, see (Tobin et al., 2017), and also should minimize effects of overfitting. Hence, we select matrices randomly to generate training batches on the fly. Then we randomly select a matrix entry that belongs to a facade object. Using flood filling, we reset all entries of the chosen object to represent background. We repeat this step until 20% of objects are re-labeled. With this we trained networks using a batch size of 16 and 20,000 batches. We trained on a total of 320,000 randomly generated matrices. Networks are trained so that missing facade objects are added to the input. Therefore, the ground truth is the original matrix. To further improve results, we define clusters such that a number of the cluster serves as a refined label. Different classes (windows, doors) belong to different clusters. Objects of one class belong to the same cluster if they are positioned in exactly the same rows and if all objects possess the same number of columns. As seen in Figure 5*(d)*-*(f)*, this results in a horizontal cluster-

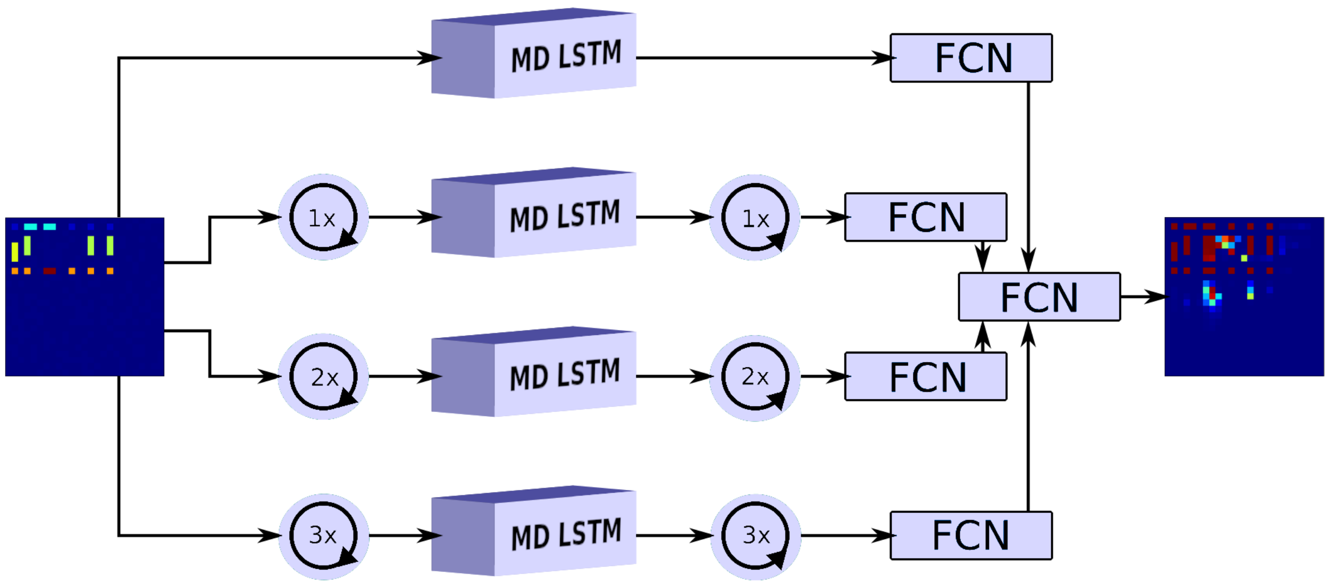


Figure 4: Rotated Multi-Dimensional LSTM

ing. We enumerate clusters and use the numbers as new object labels, i.e., as new matrix entries. Clustering is only applied to input data, ground truth and network output consist of probability values. To avoid overfitting, we add noise to background entries of the input. For comparison we tested RMD LSTM against MD LSTM and QRNN. MD LSTM and RMD LSTM take about a full day to finish training on an NVIDIA P6000 GPU. The QRNN, on the other hand, is much faster, the training was finished after 3 hours. All networks were trained with the same number of iterations using the Adam optimizer and a Mean Squared Error loss function. After training we tested on matrices derived from the Graz50 data set (Riemenschneider et al., 2012).

Multi-Dimensional LSTM. MD LSTM is a RNN architecture that allows for multi-dimensional input by using separate memory connections for each dimension. In the two-dimensional case for example, a cell (x,y) is connected with $(x-1,y)$ and $(x,y-1)$. Thus, a directed spatial context is established. The proposed Rotational MD LSTM is based on this network.

Quasi Recurrent Neural Network. The QRNN is a CNN that emulates memory connections with an embedded pooling layer. For inputs x_1 to x_T three vectors at timestep t are the outputs of the convolutional layers. A candidate vector z_t, a forget vector f_t and an output vector o_t are calculated as follows

$$\begin{aligned} z_t &= \tanh(conv_{W_z}(x_t,\dots,x_{t-k+1})) \\ f_t &= \sigma(conv_{W_f}(x_t,\dots,x_{t-k+1})) \\ o_t &= \sigma(conv_{W_o}(x_t,\dots,x_{t-k+1}))\ . \end{aligned}$$

The convolution *conv* uses a filter size of k and the weight vectors are represented by W_z, W_f and W_o. The immediate outputs z_t, f_t and o_t are then used to calculate the hidden states c_t in the pooling layers. In this step

$$\begin{aligned} c_t &= f_t \odot c_{t-1} + (1-f_t) \odot z_t \\ h_t &= o_t \odot c_t\ , \end{aligned}$$

where h_t is the network output for timestep t. c_0 and h_0 are initialized with 0. Operator $\odot$ denotes element-wise multiplication. An advantage of the QRNN is that input can be processed in parallel, while regular RNNs compute intermediate outputs sequentially. Network operations including pooling are cheap to perform such that the network is fast and memory efficient. A sequential processing only takes place in the pooling layer.

Rotated Multi-Dimensional LSTM. RNNs are mainly used for time-dependent problems where information from the past have to be taken into account. To solve spatial problems, often data of a complete spatial neighborhood is relevant. However, when MD LSTM for two-dimensional input deals with cell (x,y), information of "future" cells $(x+1,y)$ and $(x,y+1)$ needs to be considered but has not been computed. Especially when applying MD LSTM to facade completion, missing facade objects in the upper left image region can't be added. To overcome this problem, we combine four MD LSTMs, working on facade inputs that are rotated by $k \times 90°$, $k \in \{0,1,2,3\}$. Thus, each single MD LSTM starts in a different corner of the facade. The outputs of the MD LSTMs are rotated back to the original facade orientation. These results are combined with fully connected layers and sigmoid activation function, see Figure 4. We also experimented with maximum pooling instead of fully connected layers, which only resulted in a small increase by an absolute values of 0.05 for IoU.

Reconstruction. To use the neural network in reconstruction tasks, the detected objects have to be prepared as described in Section 3.1. The resulting matrix is then processed by the chosen neural network to obtain object recommendations, see Figure 1. Recommended objects are equipped with bounding boxes by considering cell coordinates of the IRL. However, this might lead to multiple boxes covering a single object. Such boxes can be merged in a post-processing step.

4 EXPERIMENTS

All evaluations were performed on the Graz50 data set. To be able to compare results, fixed evaluation data with 10000 sets was generated beforehand by removing 20% of facade objects using the same method as mentioned in Section 3.2. We start with quantitative results and discuss qualitative results later. Tables 1 and 2 show a comparison of results with and without pre-clustering. The output of the neural networks consist of a probability map. Object probabilities below 0.5 were considered as background. For calculating scores, we counted every classified matrix entry in one of four sums: true positive (tp), false positive (fp), true negative (tn) and false negative (fn). Tables 1 and 2 show average values of

$$\begin{aligned} \text{Accuracy (Acc.)} &= \frac{\text{tp}+\text{tn}}{\text{tp}+\text{tn}+\text{fp}+\text{fn}} \\ \text{Precision (Prec.)} &= \frac{\text{tp}}{\text{tp}+\text{fp}} \\ \text{Recall (Rec.)} &= \frac{\text{tp}}{\text{tp}+\text{fn}} \\ \text{IoU} &= \frac{\text{tp}}{\text{tp}+\text{fp}+\text{fn}} . \end{aligned}$$

We focused on Intersection over Union (IoU) as the most meaningful evaluation score since it does not consider background. All scores were computed separately on single facades. Then arithmetic means over all facades of the evaluation data set were taken. To determine whether the networks really improved the facade layouts, we also calculated the values for the unchanged network input. Here, precision is 1.0 because annotated facade objects also belong to the ground truth. Missing facade objects contribute to fn but not to accuracy. As it can be seen in Table 2, the proposed RMD LSTM produces better results in terms of accuracy, recall and IoU.

Figure 5 shows how QRNN and RMDLST make recommendations for missing objects. Although we trained the networks on the different CMP data set, RMD LSTM was able to also complete Graz50 layouts. QRNN somewhat failed to achieve similar results. Furthermore, Figure 6 shows a variation of examples of RMD LSTM recognition completion on the data set used for evaluation. It can be seen that it completes missing objects on facades like in Figure 6 *(a)*, *(b)*, *(h)* and *(i)* in a good manner. But also that there are some difficulties with objects that are composed of more than one cell. It also adds door-shaped objects in some places which the network considers useful but which are not present in the original images.Besides testing with graz50 data set, we also calculated scores on training data. Here, QRNN performed better than RMD LSTM. QRNN brought an absolute increase of the IoU value by 0.18. If the style of facades is known and if one can train networks on facades of the given style then QRNN might be superior to RMD LSTM.

Furthermore, experiments were also conducted with GridLSTM (Kalchbrenner et al., 2015), as it performed better in language and text translation compared to LSTM networks. It has an advantage over them due to its grid of multi-way interactions. The overhead generated by this grid is a disadvantage, because it increases the memory requirements. However, the memory requirement became too high for the problem of facade structure completion discussed in Section 1. Thus, we had to reduce batch size and the number of hidden units, with these restrictions the trained network classified all objects as background.

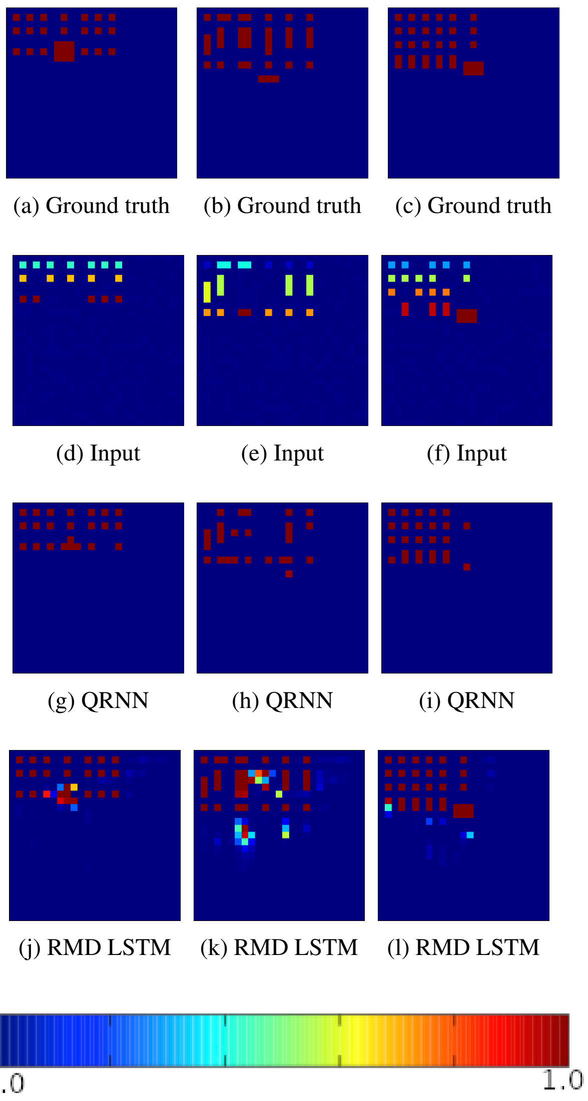


Figure 5: Network output consisting of 25×25 probability values (0 = blue, 1 = red) for three facades of the Graz50 data set: images *(a)* to *(c)* represent the ground truth, images *(d)* to *(f)* show the input, images *(g)* to *(i)* present the output of QRNN, and images *(j)* to *(l)* show the results of RMD LSTM.

5 CONCLUSIONS

Our experiments with RMD LSTM showed that LSTMs are a suitable means to fill gaps in facade layouts. RMD LSTM performed better than the original MD LSTM and QRNN with an increase by 14% in IoU compared to input data. The advantage of such deep learning methods over grammar-based algorithms is that no rules have to be defined explicitly.

Results shown in Figure 6 indicate that there is still room for improvements. The main issue are objects that are represented by more than one matrix entry. This occurs in rare cases and is therefore troublesome for learning. Examples are shown in Figure 6 *(a)*, *(d)* and *(j)*, where doors are incomplete or mistaken for a window. Other problems that can occur are that objects are added by mistake, existing objects are expanded, or a com-

Table 1: Comparison of network results on binary input matrices (without cluster labels).

	Acc.	Prec.	Rec.	**IoU**
START	0.938	1.000	0.682	0.682
MD LSTM	0.980	0.913	0.787	0.732
QRNN	0.973	0.888	0.720	0.664
RMD LSTM	0.979	0.907	0.785	0.726

Table 2: Comparison of QRNN and RMD LSTM on input data labeled by clustering.

	Acc.	Prec.	Rec.	**IoU**
START	0.938	1.000	0.682	0.682
QRNN	0.969	0.901	0.641	0.604
RMD LSTM	**0.984**	**0.925**	**0.832**	**0.779**

bination of both, resulting in an unwanted contiguous cluster of objects.

So far, we have limited the training of neural networks to windows and doors. However, other facade objects can be treated in a similar way. A major limitation in the application of LSTMs is currently the high memory usage. MD LSTM and RMD LSTM required between 21 and 23 GB of GPU memory of the graphics card, depending on the number of hidden units used. Future work should deal with reducing memory consumption.

REFERENCES

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. *arXiv:1603.04467*.

Arjovsky, M. and Bottou, L. (2017). Towards principled methods for training generative adversarial networks. *arXiv:1701.04862*.

Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein GAN. *arXiv:1701.07875*.

Bertalmio, M., Sapiro, G., Caselles, V., and Ballester, C. (2000). Image inpainting. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 417–424.

Bochkovskiy, A., Wang, C.-Y., and Liao, H.-Y. M. (2020). YOLOv4: Optimal speed and accuracy of object detection. *arXiv:2004.10934*.

Bradbury, J., Merity, S., Xiong, C., and Socher, R. (2016). Quasi-recurrent neural networks. *arXiv arXiv:1611.01576*.

Dai, D., Riemenschneider, H., Schmitt, G., and Van Gool, L. (2013). Example-based facade texture synthesis. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1065–1072.

Dehbi, Y., Staat, C., Mandtler, L., Pl, L., et al. (2016). Incremental refinement of facade models with attribute grammar from 3D point clouds. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 3:311.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc.

Graves, A., Fernández, S., and Schmidhuber, J. (2007). Multi-dimensional recurrent neural networks. *CoRR*, abs/0705.2011.

Gröger, G., Kolbe, T. H., and Czerwinski, A. (2007). OpenGIS CityGML Implementation Specification (City Geography Markup Language). *Open Geospatial Consortium Inc, OGC*.

Guillemot, C. and Le Meur, O. (2014). Image inpainting: Overview and recent advances. *Signal Processing Magazine, IEEE*, 31:127–144.

He, K., Gkioxari, G., Dollar, P., and Girshick, R. (2017). Mask R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2961–2969.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

Hensel, S., Goebbels, S., and Kada, M. (2019). Facade reconstruction for textured LoD2 citygml models based on deep learning and mixed integer linear programming. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, IV-2/W5:37–44.

Hu, H., Wang, L., Zhang, M., Ding, Y., and Zhu, Q. (2020). Fast and regularized reconstruction of building facades from street-view images using binary integer programming. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, V-2-2020:365–371.

Huang, J.-B., Kang, S. B., Ahuja, N., and Kopf, J. (2014). Image completion using planar structure guidance. *ACM Transactions on graphics (TOG)*, 33(4):1–10.

Kalchbrenner, N., Danihelka, I., and Graves, A. (2015). Grid long short-term memory. *arXiv:1507.01526*.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollar, P. (2017). Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988.

Mtibaa, F., Nguyen, K.-K., Azam, M., Papachristou, A., Venne, J.-S., and Cheriet, M. (2020). LSTM-based indoor air temperature prediction framework for hvac systems in smart buildings. *Neural Computing and Applications*, pages 1–17.

Nazeri, K., Ng, E., Joseph, T., Qureshi, F. Z., and Ebrahimi, M. (2019). Edgeconnect: Generative image inpainting with adversarial edge learning. *arXiv:1901.00212*.

Riemenschneider, H., Krispel, U., Thaller, W., Donoser, M., Havemann, S., Fellner, D., and Bischof, H. (2012). Irregular lattices for complex shape grammar facade parsing. In *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1640–1647.

Salehinejad, H., Sankar, S., Barfett, J., Colak, E., and Valaee, S. (2017). Recent advances in recurrent neural networks. *arXiv:1801.01078*.

Sherstinsky, A. (2020). Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network. *Physica D: Nonlinear Phenomena*, 404:132306.

Teboul, O., Kokkinos, I., Simon, L., Koutsourakis, P., and Paragios, N. (2011). Shape grammar parsing via reinforcement learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2273–2280. IEEE.

Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., and Abbeel, P. (2017). Domain randomization for transferring deep neural networks from simulation to the real world. In *Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 23–30.

Tyleček, R. and Šára, R. (2013). Spatial pattern templates for recognition of objects with regular structure. In Weickert, J., Hein, M., and Schiele, B., editors, *Pattern Recognition*, pages 364–374, Berlin, Heidelberg. Springer.

Wonka, P., Wimmer, M., Sillion, F., and Ribarsky, W. (2003). Instant architecture. *ACM Transactions on Graphics (TOG)*, 22(3):669–677.

Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X., and Huang, T. S. (2018). Generative image inpainting with contextual attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5505–5514.

Zhang, D. and Wang, D. (2016). Relation classification: CNN or RNN? In Lin, C.-Y., Xue, N., Zhao, D., Huang, X., and Feng, Y., editors, *Natural Language Understanding and Intelligent Applications. ICCPOL 2016, NLPCC 2016. Lecture Notes in Computer Science*, volume 10102, pages 665–675, Cham. Springer International Publishing.

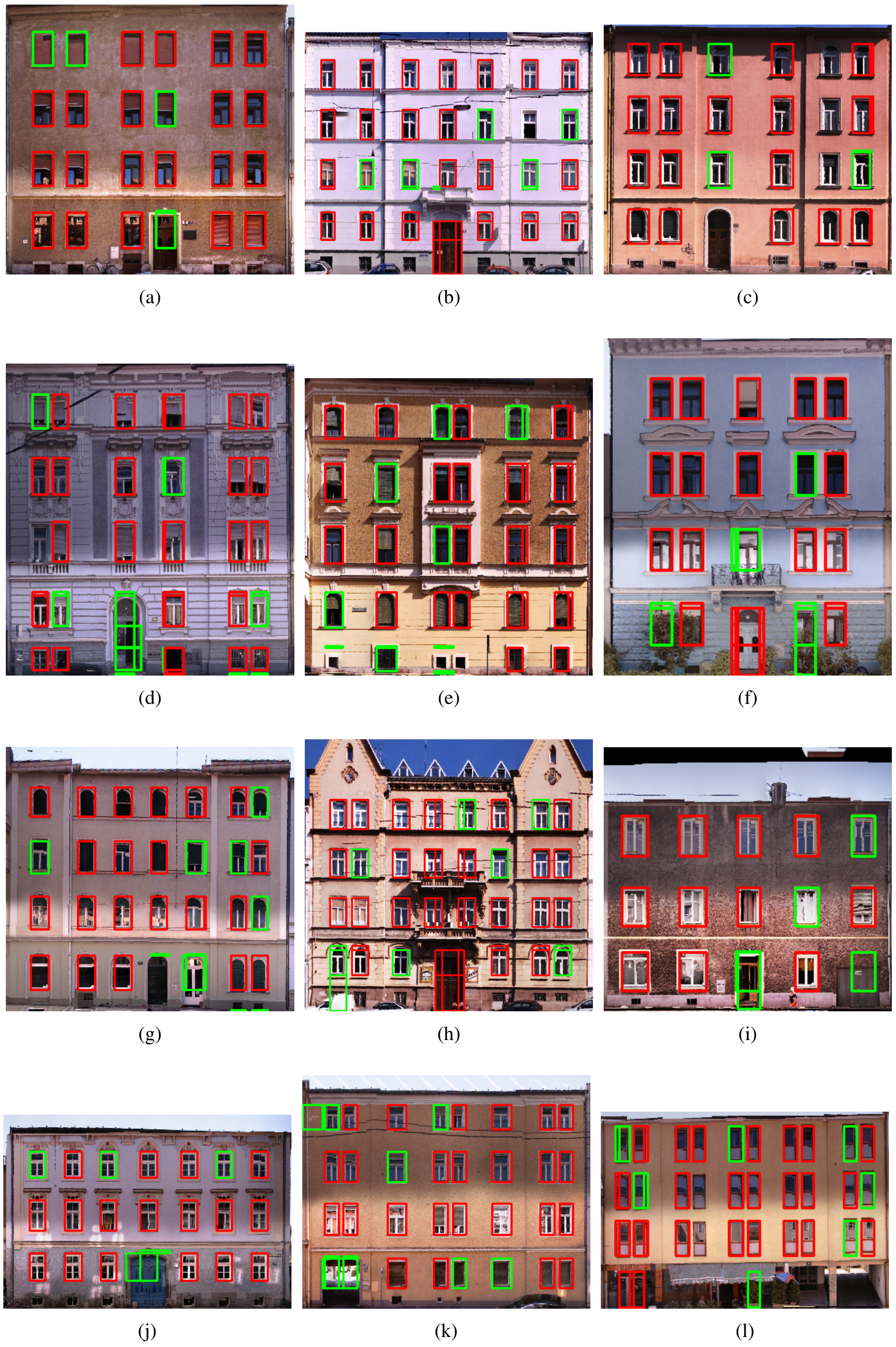


Figure 6: Objects were randomly removed from the ground truth of the Graz50 dataset. Remaining objects are marked with red boxes. The RMD LSTM network added most of the missing objects. Added objects are annotated with green boxes.