TUG-AT

Authors: Christian Payer, Darko Štern, Marlies Feiner, Horst Bischof, Martin Urschler

Email: christian.payer@icg.tugraz.at

Platform: Linux

Prerequisites: Python >= 3.4 (see requirements.txt for python package requirements)

TUG-AT: SUMMARY

Our approach uses a recurrent fully convolutional network architecture for tracking cell instance segmentations over time. The network architecture incorporates convolutional gated recurrent units (ConvGRU [1]) into a stacked hourglass network [2] to utilize temporal video information. Furthermore, the network is trained with an embedding loss based on cosine similarities, such that the network predicts unique embeddings for every instance throughout videos. For datasets with large input size, we split the input image into multiple overlapping tiles and predict embeddings for each of the overlapping tile independently. For each overlapping tile, instances are clustered for subsequent frame pairs with the algorithm mean shift [3]. Afterwards, the instances of the individual overlapping tiles are merged again to generate instances for the whole input image sequence. More detailed explanations can be found in our MICCAI 2018 paper [4]. The framework for training and testing the networks is available under https://github.com/christianpayer.

TUG-AT: PREPROCESSING

As the approach combines segmentation and tracking and is trained with sequences of consecutive frames, the segmentation labels of the training data were modified such that the labels are consistent over the whole video. The intensity values of the images were normalized to be between -1 and 1. As microscopy images may contain outliers in terms of minimum and maximum values, we calculate the minimum value as the median of i_{min}% of all intensity values of an image, and the maximum as the median of i_{max}%. Furthermore, to reduce the amount of noise in the images, they were filtered with a Gaussian function σ. For datasets with large input images, we resample the input image to a dataset dependent size s_{input} and then split the input image into multiple tiles of size s_{network} that overlap by s_{overlap}. The network then generates predictions for each of the overlapping tiles independently.

TUG-AT: SEGMENTATION AND TRACKING

Network architecture: We modify the stacked hourglass architecture [2] by integrating ConvGRU [1] to propagate temporal information. As proposed by [2], we also stack two hourglasses in a row to improve network predictions. Therefore, we concatenate the output of the first hourglass with the input image to use it as input for the second hourglass. We apply the cosine embedding loss function on the outputs of both hourglasses, while we only use the outputs of the second hourglass for the clustering of embeddings. We use networks with an input and output image size s_{network}.

Cosine Embedding Loss: We let the network predict a d-dimensional embedding vector for each pixel p of the image. To separate cell instances, firstly, embeddings of pixels $p \in S^{(i)}$ belonging to the same instance i need to be similar, and secondly, embeddings of $S^{(i)}$ need to be dissimilar to embeddings of pixels $p \in S^{(j)}$ of other instances $j \neq i$. We compare two embeddings with the cosine similarity

$$\cos(\mathrm{e}_1, \mathrm{e}_2) = \frac{\mathrm{e}_1 \cdot \mathrm{e}_2}{\|\mathrm{e}_1\| \|\mathrm{e}_2\|},$$

which ranges from −1 to 1, while −1 indicates the vectors have the opposite, 0 orthogonal, and 1 the same direction.

Clustering of Embeddings: To get the instance segmentations from the predicted embeddings of each tile, individual groups of embeddings that describe different instances need to be identified. As the number of instances is not known, we perform this grouping with the clustering algorithm mean shift [3] adapted for cosine similarities, which estimates the number of clusters automatically. Mean shift uses a single parameter called bandwidth b, which is connected to the distance between clusters. To simplify clustering and still be able to detect splitting of instances, we cluster only overlapping pairs of consecutive frames at a time. Since our embedding loss allows same embeddings for different instances that are far apart, we use both image coordinates (multiplied with a factor *c*) and the value of the embeddings as data points for the clustering algorithm. After identifying the embedding clusters with mean shift and filtering clusters that are smaller than t_{size}, the segmented instances are obtained.

Merging of Overlapping Tiles: To merge the segmented instances of overlapping tiles, we remove instances that are touching a tile border that is not an outer border of the image. In case an instance is predicted by multiple overlapping tiles, the instance with the larger area is taken.

Propagating Instances over Time: For merging the segmented instances of subsequent frame pairs that overlap in time, we identify same instances by the highest intersection over union (IoU) between each segmented instance in the overlapping frame. The resulting segmentations are then upsampled back to the original image size, generating the final segmented and tracked instances.

TUG-AT: POST-PROCESSING

Instances that are outside of the defined valid region for each dataset are removed. The parent instance IDs are detected by dilating the instance segmentation by d_{parent} pixels and searching for the instance with the largest IoU within the previous f_{parent} frames. We fix wrong mitosis events, where an instance is split and then merged again within t_{merge} frames. Furthermore, the detected parent and child instances are postprocessed such that they conform with the evaluation protocol of the challenge, i.e., a parent instance may not exist at the same time or after its child.

REFERENCES

1. Ballas N, Yao L, Pal C, Courville A. Delving deeper into convolutional networks for learning video representations. In *Proceedings of the 4th International Conference on Learning Representations*, abs:1511.06432 (2016).
2. Newell A, Yang K, Deng J. Stacked hourglass networks for human pose estimation. In *Proceedings of the 14th European Conference on Computer Vision*, 483-499 (2016).
3. Comaniciu D, Meer P. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **24**, 603-619 (2002).
4. Payer C, Štern D, Neff T, Bischof H, Urschler M. Instance segmentation and tracking with cosine embeddings and recurrent hourglass networks. In *Proceedings of the 21st Medical Image Computing and Computer Assisted Intervention*, 3-11 (2018).