Freie Universität Berlin

Bachelor Thesis in Computer Science at Freie Universität Berlin

Biorobotics Working Group

Detecting honey bee trophallaxis in image data with convolutional neural networks

Mathis Hocke
Matrikelnummer: 4625905
mathis.hocke@fu-berlin.de

Supervisor: Prof. Dr. Tim Landgraf

Berlin, December 19, 2018

Abstract

Observation of the food exchange behaviour of honey bees, trophallaxis, has been laborious in the past. In this work I created a classifier to detect trophallaxis as a part the BeesBook system that aims to automate detection of bee behavior in general. Based on labeled data I created a dataset of images showing trophallaxis and trained a convolutional neural network to classify images. I show that using more than one frame to classify trophallaxis yields a better score than using a single image. The network reaches an F_1 score of 0.89 for detecting trophallaxis which is an improvement over existing methods.

Eidesstattliche Erklärung

Ich versichere hiermit an Eides Statt, dass diese Arbeit von niemand anderem als meiner Person verfasst worden ist. Alle verwendeten Hilfsmittel wie Berichte, Bücher, Internetseiten oder ähnliches sind im Literaturverzeichnis angegeben, Zitate aus fremden Arbeiten sind als solche kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungskommission vorgelegt und auch nicht veröffentlicht.

December 19, 2018

Mathis Hocke

Contents

1 Introduction

The food exchange behavior, trophallaxis, of honey bees is a focal interaction for various research. During trophallaxis, the donor bee opens the mandibles and exposes nectar to the receiver, and liquid food gets transferred from the mouth of one bee to another (Frisch, 1967). By observing trophallaxis, researchers can study the social networks of the bees and understand transmission routes of pathogens within the colony (Naug, 2008).

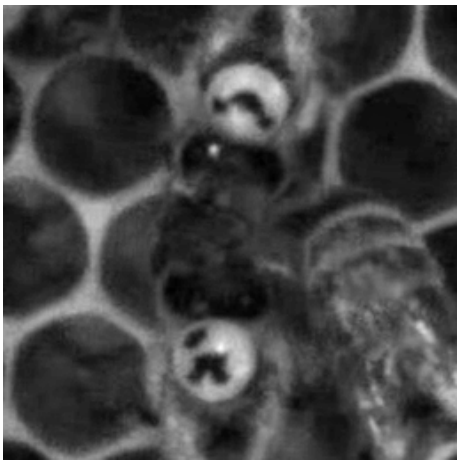


Figure 1: Two bees performing trophallaxis. The image is part of the training data but does not have any preprocessing except the first cropping step and contrast adjustment with CLAHE (see section 2.3). See figure 3 for examples of preprocessed images.

Trophallaxis is also used by the bees to exchange information about the food stores in the colony. This information gets distributed across the colony by trophallactic contacts and regulates important processes like the egg laying of the queen, and the decisions about what food needs to be collected. Information is not only in the food itself, but also in the behavior of the bees during trophallaxis. The willingness of the recipient hints at how difficult it is to store the food in the given moment. The foragers receive this signal and decrease or increase nectar foraging (Crailsheim, 1998). There is also some information in the rate with which a bee unloads nectar to a receiving bee. This rate with which a receiver unloads the food to the next receiver is positively correlated with the rate at which she received it. In this way information about the quantity of food is transmitted along with the food itself (Goyret and Farina, 2005). The transfer delay, the time until two bees start the trophallaxis is another signal that carries information about the food source. It has been shown that differences in flow rate at the food source influence transfer delay. Transfer delay also varies during the day, being shortest in the morning (Farina and Núñez, 1992).

Biologists have used manual methods for analyzing bee interactions in the past, but these are very time consuming. De Marco and Farina (2001) have studied how changes in food source profitability can influence bee behavior during trophallaxis. Due to a manual process they were only able to observe one bee per day and only 17 bees in total. By automating the process it becomes possible to observe more bees for

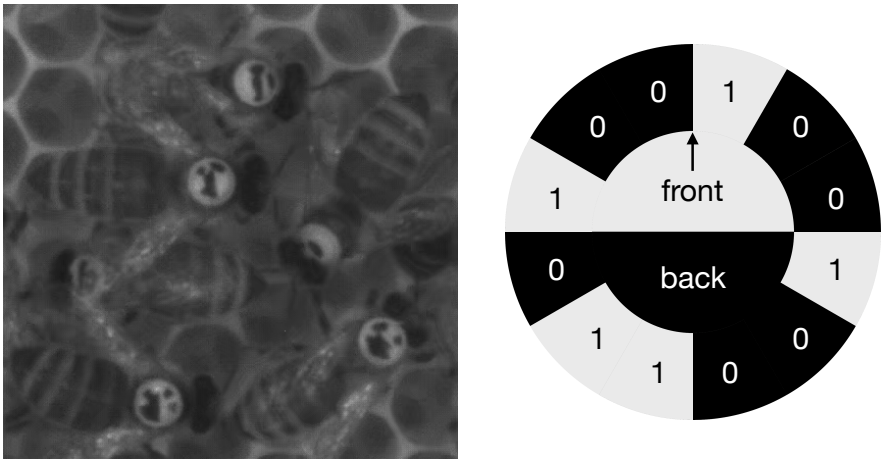


Figure 2: The image on the left shows the tags on the bees' thoraxes. Shown on the right is an example of a tag annotated with the bit values shown as numbers. A ring of 12 segments identifies the bee with a 12 bit number. The center consists of two half circles, where the white half marks the front of the bee.

a longer time.

Blut et al. (2017) describe a system which enables automated detection of bee behavior. They obtained continuous information on position and orientation by marking bees with 2D bar codes. They tracked 100 newly emerged worker bees for two days and trained a machine learning system on manually labeled behavior classes. The system used a frame rate of 4 frames per second. The training data consists of social per-frame features, which describe each individual's state in each frame in relation to its nearest nest mates, for example distance, orientation and speed towards another bee. Their classifier was used to classify four different behaviors – antennation, begging, offering and trophallaxis – as a single class. This combined class was defined by the common features of head to head rotation and antennal contact.

They found that the median duration of the trophallaxis behaviors was 8 seconds, with a minimum of 5 and a maximum of 30.5 seconds. Since the durations of the other behaviors were much shorter, they tested if the four different behaviors could be classified based on their durations. When setting a duration threshold to 5 seconds they were able to classify 100% of the labeled trophallaxis behaviors, but the false positive rate was 28%.

Gernat et al. (2018) created an automatic monitoring system for analyzing honey bee interactions and social networks. They tagged the bees with a custom matrix bar code attached to the thoraxes. Custom computer vision algorithms were used to determine position, shape and orientation of the bees. They were detecting interactions by checking if the bees' heads were connected by a shape that resembles a proboscis (tongue) or an antenna. A frame rate of one frame per second was used. As a preprocessing step they used spatiotemporal information. They used a dataset of 39,863 images, with 1,045 images showing trophallaxis to tune the parameters of their algorithms. A second dataset for evaluation was created by randomly choosing 100 triplets of successive whole-hive images, and extracting all pairs of bees, that are in possible positions for trophallaxis (Gernat et al., 2018, Supporting Information).

The data used in this work was collected in 2016 during an experiment of the BeesBook project. The BeesBook system can track all individuals of a honey bee colony by using tags. The circular, curved tags are attached to the thorax. The tags are used to identify the bees, as well as getting the current orientation, see figure 2. The system features 4 high resolution cameras with a resolution of 4000 by 3000 px each and a frame rate of 3 fps. Additionally high frequency images with lower resolution are available. (Wario et al., 2015)

Berg (2018) created a filter to find possible occurrences of trophallaxis in the data of the BeesBook project. This filter works based on the information about the bees that was extracted before, such as position and orientation of each bee. He applied his filter to the data to get pairs of bees, that may have performed trophallaxis. He then manually reviewed the resulting data to evaluate his filter, and to create the labeled dataset that has been used in this work. His filter operates on *tracks*, where a track describes a series of detections of a single bee. A track stores the bee's ID, as well as timestamps and the IDs of the corresponding frames. The filter is based on multiple criteria: minimum and maximum distance between the bees, relative rotation and duration. After generating the tracks, the filter looks at two tracks at a time and performs checks based on the criteria.

- The two tracks need to overlap for an adjustable minimum duration.
- The bees need to have a specified range of distance from each other.
- The angle of the bees has to be in a range that the bees are facing each other.

If the second and third check pass for at least the specified minimum duration, the track combination may belong to two bees performing trophallaxis.

The goal of this work is to create a component for a system for automatic detection of trophallaxis. I used the data collected by the BeesBook project that has been pre-filtered by Berg (2018) to train a convolutional neural network to detect trophallaxis in images.

2 Implementation

2.1 Generating training data

The training data is based on the labeled data generated by Berg (2018). Each entry in the dataset is an *event* that includes two bee IDs, a list of frame IDs, a label that indicates if there is trophallaxis happening in the frame, and a start and end index of the observed trophallaxis. His filter (see page 3) has few false negatives, but many false positives. That makes it ideal as a preprocessing step to make manual labeling faster, by discarding interactions that are very unlikely to be trophallaxis. By keeping the false negatives low it minimizes the bias of ignoring some kinds of trophallaxis, e.g. with unusual angles. Minimizing this bias is crucial, because the same bias would be inherited by any network that is trained on the resulting data.

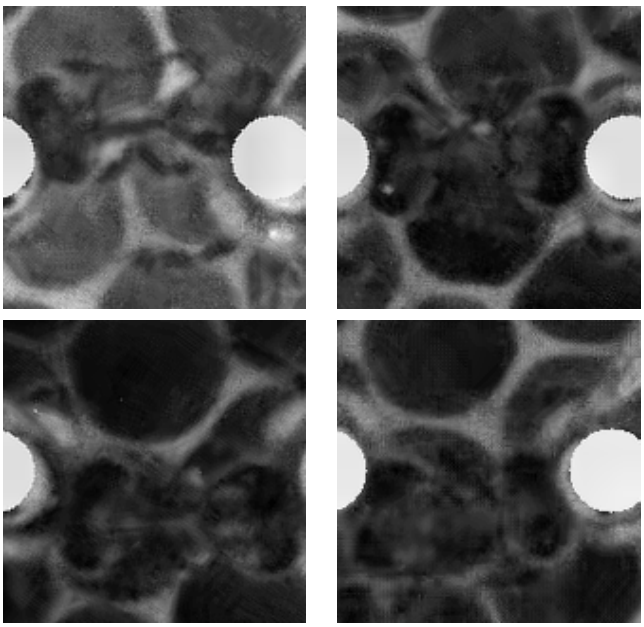


Figure 3: Shown are different head angles of bees during trophallaxis. Each image is fully preprocessed, so the left bee is rotated that its tag is directed horizontally towards the right edge. The examples show that the bees can rotate their heads and perform trophallaxis in angled positions. The crop area needs to be big enough to include all possible angles. Since the data only provides position and orientation of the tag, but not of the head, it is not easily possible to crop a smaller rectangle directly around the heads. As these are preprocessed images the tags are digitally masked to avoid overfitting. The contrast of the images has been adjusted with CLAHE (see section 2.3).

Berg (2018) applied the filter that he developed to the raw image data over a period of two hours and manually labeled the results. There are 2007 events that were labeled as human decidable. Unlabeled events and 59 events that were labeled as not human decidable were not used. Due to the amount of data collected in the BeesBook project, the image data is stored in video files for compression. The library `bb_backend`, that is part of the BeesBook system, provides an API to request images based on the frame

ID. The meta data, including the positions and orientations is stored in a PostgreSQL database. The final image size that gets fed into the network is 128 by 128 px. This size is enough to show both heads during trophallaxis in most cases. See figure 3.

I created python scripts that bring the images in the desired format. After an image is received by the API, it needs to be cropped around the two bees that are involved, to save hard disk space, by not saving the full image. Requesting all images takes multiple days to complete. A second step includes all preprocessing that is too slow to do in real time. The images get rotated and then cropped to 160 by 160 px to leave room for augmentation, to keep them as small as possible to make reading them from disk fast. The remaining processing steps are done in real time during training.

The markers of the bees are cropped out in most cases, but since they can appear at least partially in the images, they were masked with a gray circle to prevent overfitting by learning the tags. See figure 3.

Since for each image in the dataset the previous and next frames are known, it is possible to use that information during training. The network can judge each image by looking also at n frames before and after. To make it possible to include n frames before the first frame in the event, and n frames after the last frame, I created a python script that adds the missing frame IDs to the events as padding. The padding frames have no label and can not be used directly as center frames. See figure 4.

The position and orientation is not known for all frames, because the tag decoder can not decode the tags of every bee in every frame, e.g. when it is concealed. In these cases I used interpolation to fill the gaps.

2.2 Frames directly before and after trophallaxis

Each event that shows trophallaxis in some frames can have some frames that are labeled as not trophallaxis. These frames appear directly before the beginning and after the end of the interaction. This happened when the filter of Berg (2018) was detecting a trophallaxis event correctly, but was wrong about the precise start or end time. As the threshold is also difficult to set by a human observer and the images directly before and after the threshold look very similar, I was evaluating to ignore these frames and to exclude them from the training data. Excluding these frames improved performance significantly (see table 1).

2.3 CLAHE

Contrast limited adaptive histogram equalization (CLAHE) is a method to enhance contrast in images (Pizer et al., 1987).

As images of the dataset that have been processed with CLAHE are easier for humans to classify, and it normalizes contrast and lighting between the images, I applied it to the images to see if it improves the performance of the trained network.

2.4 Augmentation

Since the dataset was small it was important to augment the data to avoid overfitting. To augment the dataset three methods were used.

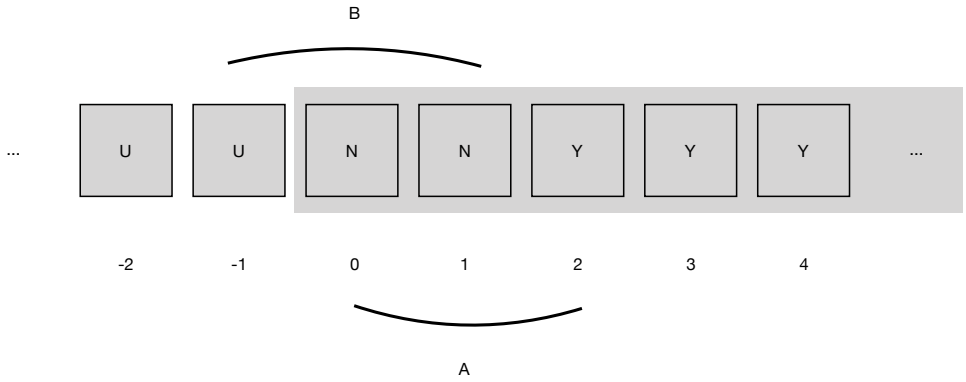


Figure 4: Each square in this figure represents a frame. The letter in each square stands for the label, that indicates if the frame shows trophallaxis. Possible values are Y for yes, N for no and U for unknown. All the frames in an event are labeled, with either yes or no. The padding frames have not been manually labeled and therefore have the unknown label. In the case *A* the frame with index 1 is the center frame. The number of channels is 3, so the frame before and after also get passed as data to the network, but only the label of the center frame is passed to the network. In case *B* the padding frames are necessary, because the frame before the center frame is not part of the original event. Since only the label of the center frame is used, it does not matter that the padding has an unknown label, but this also means, that the padding frames can not be center frames.

The images show the bees from the top, so images of trophallaxis in any rotation are possible. If the training data would show more images of trophallaxis in a certain rotation, the network may learn this and may overfit. By making the rotation consistent across the dataset this problem can be avoided. To normalize the rotation, I rotated the image in a way that one bee is always looking to the right. After that I cropped the image, placing this bee at the left side of the image. Depending on which bee is chosen to be in a fixed position, a very different looking image is created. This led to the idea to choose the bee randomly and use this as an augmentation method, see figure 5.

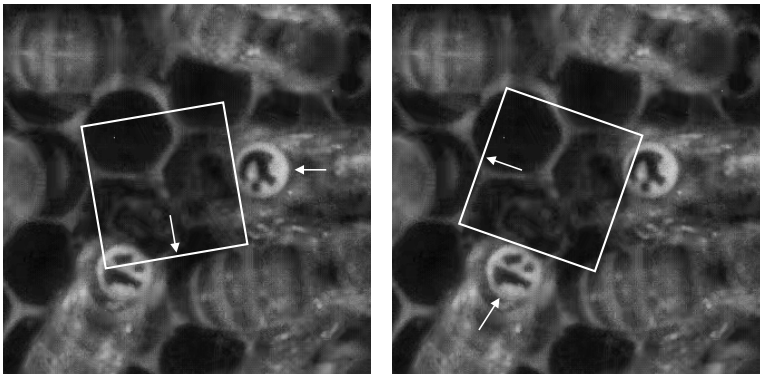


Figure 5: The figure shows two times the same image of two bees during trophallaxis. The white squares show the areas of the image, that will be used for training. The arrow inside each square indicates the side that will be the top after rotation. The arrows outside of the squares point at the tag of the bee that will then be in the fixed position at the left. Note that not only the rotation differs, but that the squares are not overlapping completely, therefore showing different parts of background, especially in the corners. The second bee, that appears in the right side of the rotated image will appear in the first case at the top right, in the second case in the bottom right.

After rotating the image to fix one bee at the left edge of the image, some padding is left around the image to leave room for further augmentation. By randomly rotating the image around it's center, overfitting can be reduced further. Another method that was used is random cropping. Instead of cropping the padding in the last step to a square around the center, the crop center is moved randomly by a few pixels.

The two rotations were performed as a preprocessing step as it takes too long to do it in real time, and it only doubles the required disk space. The random rotations and crops were done in real time during training.

2.5 Convolutional Neural Network

The convolutional neural network consists of 5 convolutional layers. After each convolutional layer, max pooling is applied. After the second, third and forth layer, batch norm is applied as well. See figure 6. ReLU is used as the activation function (see Glorot et al., 2011). The network was implemented in PyTorch. The gray scale images

get passed to the network in the same way as multiple channels of a color image. The number of channels is adjustable, to test the effect of the number of channels on the performance of the classifier. Initially I evaluated using existing models that are implemented as part of PyTorch. I trained the PyTorch implementation of ResNet (see He et al., 2015), with a small change to make it work with the 128 by 128 input size. Training with different network sizes showed, that the smallest network performed best. Since smaller networks are also faster to train and thus allow faster experimentation, I wanted to find the smallest network that still performs as good or better as the smallest ResNet. Experimentation showed, that networks with less parameters performed better. Initially I tried using a fully connected layer as the last layer, but this was again increasing the number of parameters. As a result, the F_1 score on the training set was close to 1, but the score on the test set was much lower. Removing the fully connected layer resulted in a better test score, a lower training score and faster training.

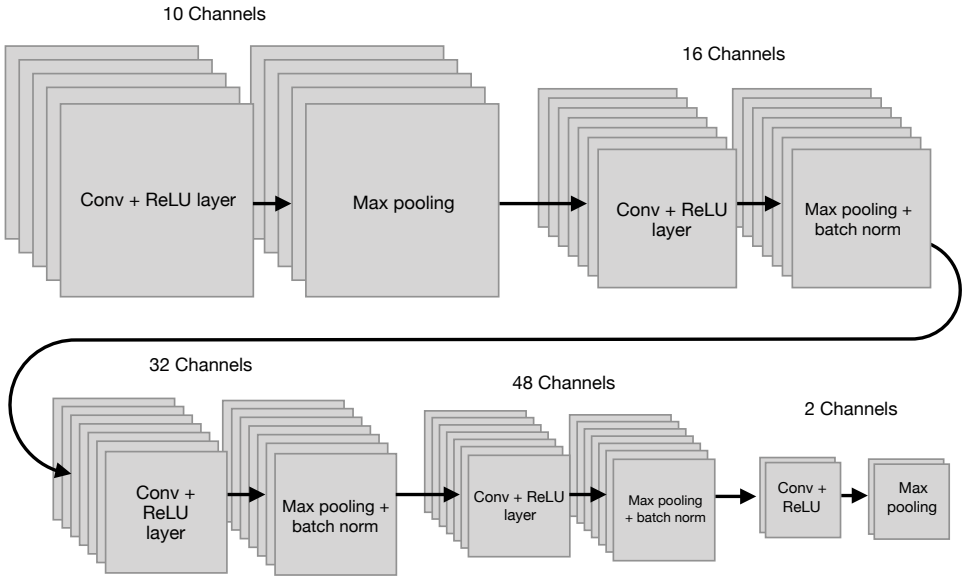


Figure 6: **Network architecture.** The convolutional neural network uses five convolutional layers, batch norm, and max pooling. ReLU is used as activation function.

2.6 Training

For training the network, the data was split into a training set and a test set, keeping all frames of each event in either the training data or the test data, to avoid having very similar frames in both sets. The events were split randomly with 80% of the events going into the training data. Cross validation was used with ten different splits. Each training was run for 50 epochs. Cross entropy loss was used as loss function and Adam (see Kingma and Ba, 2014) was used as the optimizer. Training took around 18 minutes on average per training run on an Nvidia TITAN Xp, when using 3 channels. Increasing the number of channels increases the training time.

3 Evaluation

3.1 Results

As only 7% of the events are labeled as trophallaxis, judging the result by the accuracy is not useful, as the accuracy can be very high, even if all frames were classified as not trophallaxis. Because of this the F_1 score was used as a metric, with $F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$ (see table 5 for accuracy and other performance metrics). The network was repeatedly trained in cross validation runs of 10 trainings each.

Table 1 shows the effect of using the negative frames around a trophallaxis event. Using the frames before and after trophallaxis clearly impacts the performance negatively. Because of this these frames were not used. In table 2 the positive effect of the two rotations (see figure 5) is shown. The effect of contrast normalization with CLAHE is shown in table 3. CLAHE did not improve performance when using 17 channels, but did improve the score when using only 3 channels.

Figure 7 shows how increasing the number of channels affects the performance of the network. Increasing the number of channels improves the performance significantly, but comes at a cost of increased training time. The most significant improvement is obtained when increasing the number of channels from one channel to three. When increasing the number of channels beyond 3 channels, the score slowly increases further.

Figure 8 and figure 9 show the effects of random crops and random rotations. The effect of random rotations and random crops is so small, that it is not noticeable because of the random variations of the score between training runs. In table 4 the effect of combining the augmentation methods with varying parameters is shown. When combining the most promising values for rotations and crops there is a small improvement for 3 channels, but the score for 17 channels is worse with the same values. This is likely due to variation between runs and the effect of random crops and random rotations is negligible. Table 5 shows the best results that were achieved.

drop frames	F_1 score	standard deviation
no	0.7651	0.0693
yes	0.8284	0.0397

Table 1: Average F_1 scores and standard deviations of cross validated training runs. Augmentation was used, with random rotations at 20 degrees and random crops at 8 pixels, 3 channels were used.

selected bee	F_1 score	standard deviation
always first bee	0.8028	0.0410
random	0.8284	0.0397

Table 2: Selecting the bee that is rotated to the left randomly compared to always selecting the same bee. See figure 5 for details. Average F_1 scores and standard deviations of cross validated training runs. Augmentation was used, with random rotations at 20 degrees and random crops at 8 pixels, 3 channels were used.

CLAHE	channels	F_1 score	standard deviation
no	3	0.8284	0.0397
yes	3	0.8463	0.0346
no	17	0.8852	0.0159
yes	17	0.8793	0.0252

Table 3: Effect of contrast normalization with CLAHE on the F_1 score. Augmentation was used, with random rotations at 20 degrees and random crops at 8 pixels, 3 channels were used.

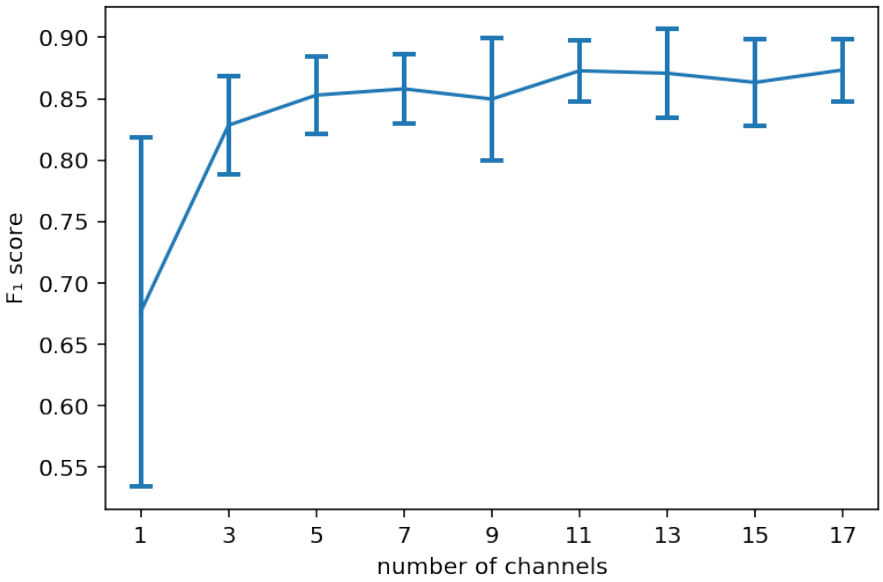


Figure 7: Effect of the number of channels on the F_1 score. Each score is the average of 10 cross validation runs. Augmentation was used, with random rotations at 20 degrees and random crops at 8 pixels.

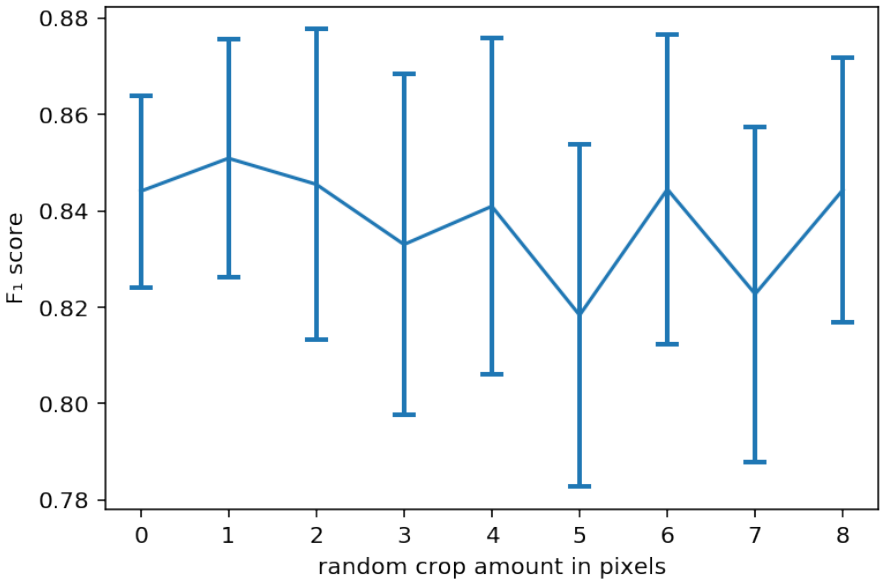


Figure 8: Effect of the random crops on the F_1 score. Each score is the average of 10 cross validation runs. Random rotations were turned off. The center of the crop area was randomly moved by a few pixels in any direction.

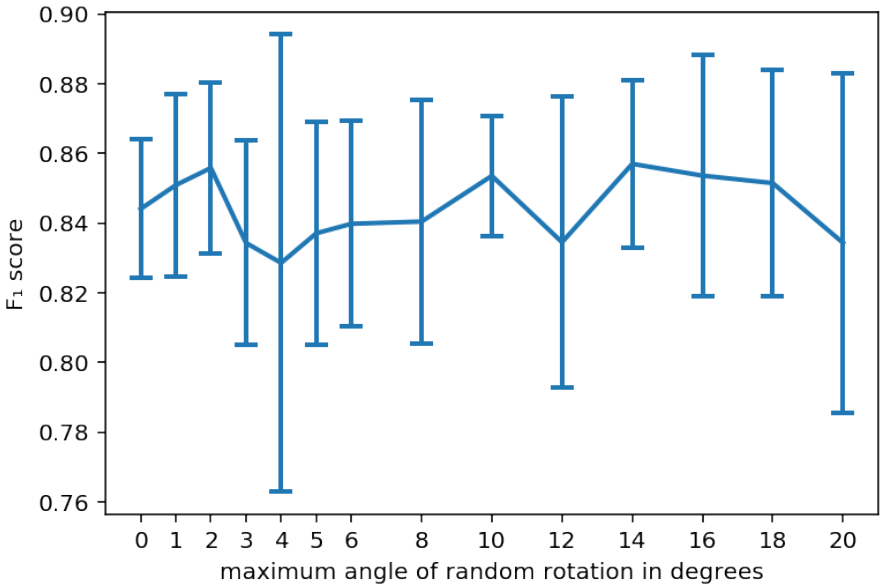


Figure 9: Effect of the random rotations on the F_1 score. Each score is the average of 10 cross validation runs. Random crops were turned off. The image was randomly rotated with increasing maximum angle.

random rotation angle	random crop pixels	channels	F_1 score	standard deviation
20	8	3	0.8284	0.0397
0	0	3	0.8441	0.0199
2	1	3	0.8595	0.0264
0	0	17	0.8852	0.0159
2	1	17	0.8697	0.0342

Table 4: Effect of the augmentation on the F_1 score. Each score is the average of 10 cross validation runs.

performance metric	3 channels	17 channels
F_1 score	0.8441	0.8852
Accuracy	0.9705	0.9775
Sensitivity	0.8999	0.9129
Specificity	0.9771	0.9840
Positive predictive value	0.7968	0.8611
Negative predictive value	0.9902	0.9910

Table 5: Performance of the final network with 3 channels (faster training) and 17 channels (better performance). Random crops and random rotations were not used. CLAHE was applied to the images when training with 3 channels. Frames before and after trophallaxis were dropped.

3.2 Failure cases

Figure 10 shows frames from the test set, that get incorrectly classified as *not trophallaxis* after training the network with 3 channels. For each example only the center frame is shown, without the frame before and after. In frames *d* and *e*, the bees performing trophallaxis are not visible due to another bee walking over them. These frames should be removed from the data set, as it is not possible to know if there are bees performing trophallaxis or not.

Frame *l* shows an unusual angle as the right bee is rotated in a way, that it is visible from the side. This is a case that will hopefully be improved by labeling more data, as there is currently no other event in the training data that shows a rotation like this. More data will hopefully make the training data more diverse.

Figure 11 shows the 16 false positives with the highest confidence. In the majority of the false positives the bees are in a position in which trophallaxis is possible. A human observer may classify some of them as trophallaxis, if the frames before and after are not known. The reason that these frames are not labeled as trophallaxis is most likely that the contact was not long enough. These cases may get correctly classified by increasing the number of channels, which would explain the better score when using more input images (see figure 7).

Frames *a* and *j* in figure 11 are from the same event and show the bees in a position, in which trophallaxis is clearly not possible, as the right bee has stuck its head inside the comb. Only judging by the position and orientation of the bees' tags, trophallaxis

may be possible, which explains why the frame was considered trophallaxis by the spatiotemporal filter of Berg (2018). The network could have learned the position of the tags, but it is unlikely that this influences the classification significantly, as the false negatives in figure 10 show tags in very similar positions. Another possibility is that the network learned the reflection that is present in both images as a white dot. These reflections may have appeared in trophallaxis frames. If this is really the problem in these frames, it would be solved as well by labeling more data, as this would introduce more frames with this kind of reflections into the data set, both showing and not showing trophallaxis. This way the network can no longer overfit on these random details.

3.3 Comparison with prior classifiers

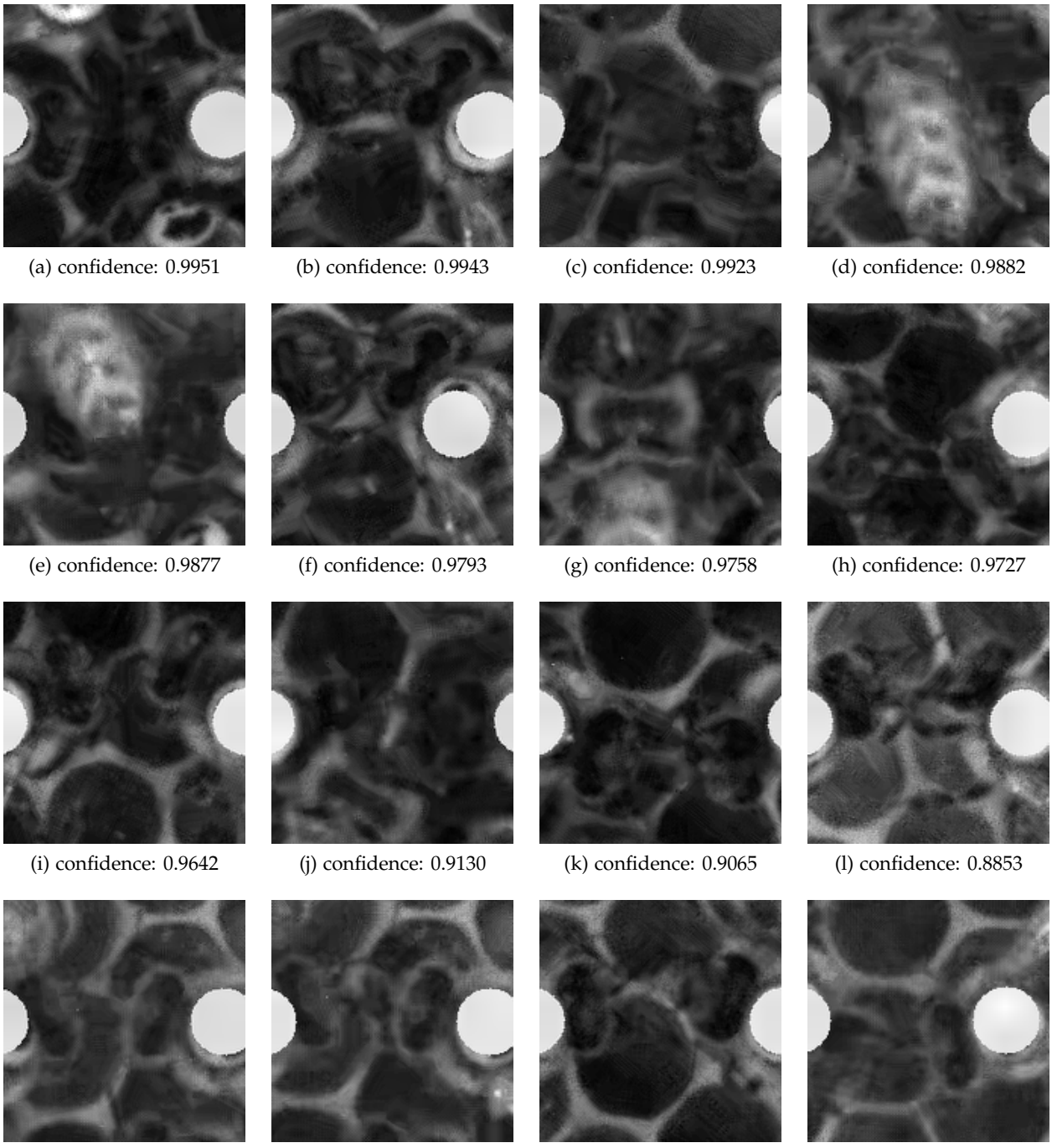
Berg (2018) trained a random forest based on his data reaching an F_1 score of 0.4038. Blut et al. (2017) did not try to detect trophallaxis directly, but did detection of encounter behaviors, including antennation, begging, offering and trophallaxis combined as one class. They achieved an F_1 score of 0.7593 and showed that they can filter out non-trophallaxis encounters with a filter based on duration. When setting the threshold on ≥ 5 seconds, no trophallaxis encounters in their data get filtered out, but the false positive rate is 28%. Combining their detection method with this filter could be used to automatically detect trophallaxis, but the F_1 score will be lower than 0.7593. Gernat et al. (2018) used spatiotemporal information as a filter in a similar way as the filter of Berg (2018) was used for this work. Interestingly the score of Berg (2018) and the score of the classifier, that uses only spatiotemporal information by Gernat et al. (2018) are very similar. They both used a similar approach but on different data. Table 6 shows that my results are an improvement over existing methods.

Classification method	F_1 score
Random forest on trajectories (Berg, 2018)	0.4038
Only spatiotemporal information (Gernat et al., 2018)	0.4048
Spatiotemporal information, computer vision (Gernat et al., 2018)	0.6407
(Blut et al., 2017)	0.7593
Convolutional neural network, image data, 3 channels (mine)	0.8441
Convolutional neural network, image data, 17 channels (mine)	0.8852

Table 6: Comparison to previous classifiers. The F_1 score of Blut et al. was calculated by using the confusion matrix in (Blut et al., 2017, table 2). The F_1 score of Gernat et al. was calculated by using the performance measures in (Gernat et al., 2018, Supporting Information, table S3).

3.4 Application on other data

From the unlabeled data collected by the BeesBook project, a separate dataset was created without the filter described by Berg (2018). A more relaxed filter was used, that only filtered by proximity. That way many frames that show two bees in angles where trophallaxis is not possible were included, and therefore much more false positives.



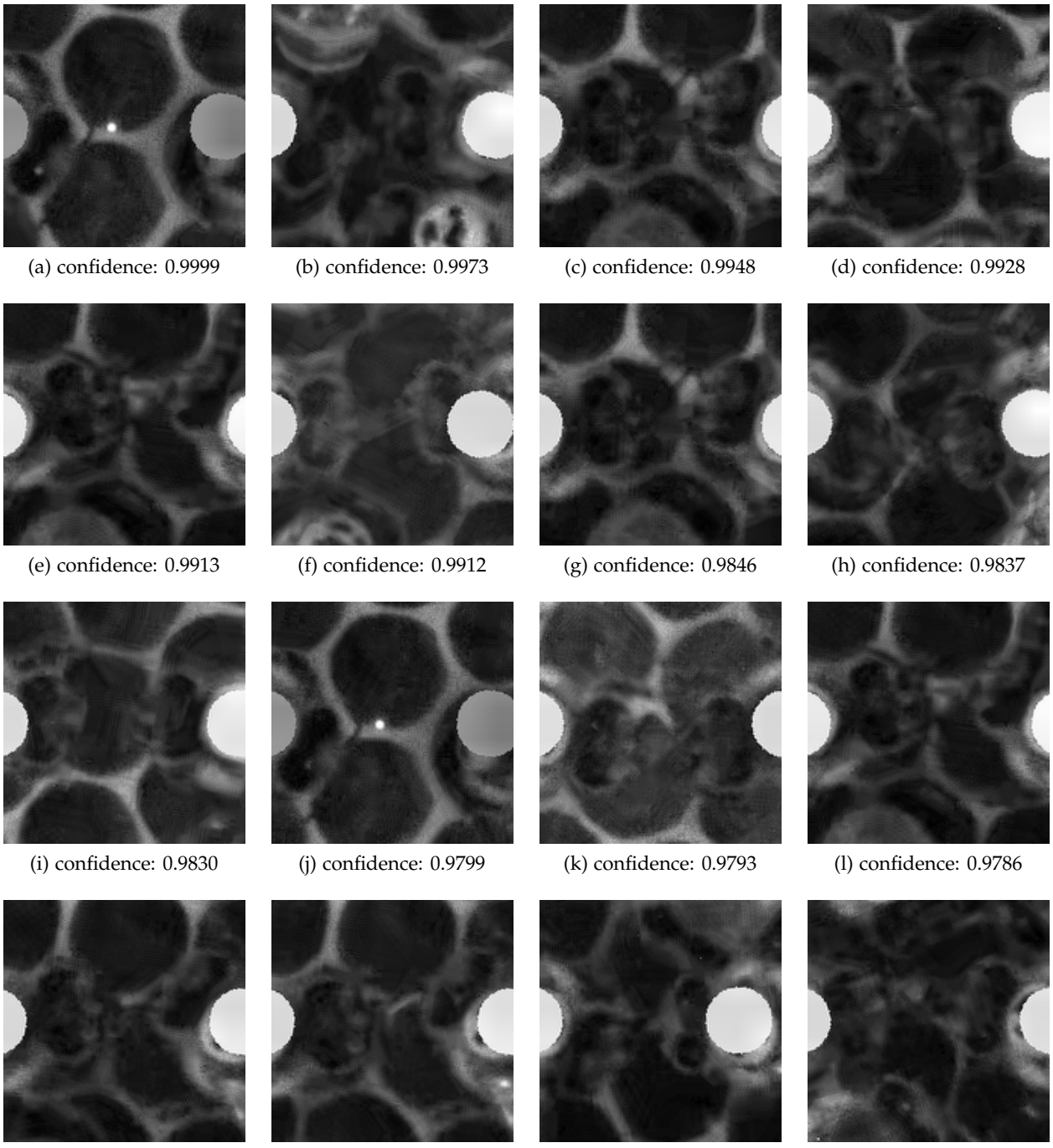
(a) confidence: 0.9951 (b) confidence: 0.9943 (c) confidence: 0.9923 (d) confidence: 0.9882

(e) confidence: 0.9877 (f) confidence: 0.9793 (g) confidence: 0.9758 (h) confidence: 0.9727

(i) confidence: 0.9642 (j) confidence: 0.9130 (k) confidence: 0.9065 (l) confidence: 0.8853

(m) confidence: 0.8775 (n) confidence: 0.8528 (o) confidence: 0.8444 (p) confidence: 0.8077

Figure 10: False negatives with highest confidence. Confidence was calculated by applying the softmax function to the output of the network and then using the value of the *not trophallaxis* class. Total number of false negatives was 27. The network was trained with 3 channels, random rotations at 20 degrees and random crops at 8 pixels



(a) confidence: 0.9999 (b) confidence: 0.9973 (c) confidence: 0.9948 (d) confidence: 0.9928

(e) confidence: 0.9913 (f) confidence: 0.9912 (g) confidence: 0.9846 (h) confidence: 0.9837

(i) confidence: 0.9830 (j) confidence: 0.9799 (k) confidence: 0.9793 (l) confidence: 0.9786

(m) confidence: 0.9782 (n) confidence: 0.9769 (o) confidence: 0.9737 (p) confidence: 0.9701

Figure 11: False positives with highest confidence. Confidence was calculated by applying the softmax function to the output of the network and then using the value of the trophallaxis class. Total number of false positives was 106. The network was trained with 3 channels, random rotations at 20 degrees and random crops at 8 pixels

The network was applied to around 17000 frames of this dataset, but only 19 frames were classified as trophallaxis by the network, and only 2 of them were true positives. This shows that the network that was trained on data that was filtered with the method of Berg (2018), only works well on data that has been filtered in the same way. When sorting the images by the confidence with which the network classified them as *not trophallaxis*, it is clear that the images with low confidence have more similarity with trophallaxis than the ones with high confidence. It is not known how many cases of trophallaxis are in this dataset. The network that was used was not the final model and it is possible that the final version would perform better. Only 3 channels were used. An increased number of channels could increase performance, but training and generating the images would take longer.

4 Discussion

It was shown that it is possible to get reasonable results for detection of trophallaxis with little labeled data. Using sequences of images instead of using single images improved the performance significantly. The results are clearly an improvement over the previous automatic classification method based on trajectories, as well as over existing solutions using custom computer vision algorithms. However it seems that more data would further improve the results. As there is a lot of data available from the BeesBook project, that can be filtered easily with the filter of Berg (2018), it is possible to label more data in the future. When more labeled data is available it may be possible to increase the network size, and likely improve the results further. To create more labeled data it may be useful to use active learning, specifically uncertainty sampling, which is useful in situations, where unlabeled data is abundant and labeling is time-consuming (see Settles, 2009). For this the trained network gets applied to unlabeled data and then only the images with the lowest confidence get labeled. Since it was shown that using image sequences can improve the performance over using single frames, it may be interesting to explore network architectures that are better suited for video classification by using e.g. 3d convolutions.

As the labeled data by Berg (2018) includes information about which bees is donor and which is receiver, it may be possible to train a network to predict this, reusing the same image dataset and augmentation methods of this work.

Bibliography

A. Berg. Detecting honey bee trophallaxis in trajectory data. *Bachelor's Thesis, Freie Universität Berlin*, Jan. 2018.

C. Blut, A. Crespi, D. Mersch, L. Keller, L. Zhao, M. Kollmann, B. Schellscheidt, C. Fülber, and M. Beye. Automated computer-based detection of encounter behaviours in groups of honeybees. *Scientific Reports*, 7(1):17663, Dec. 2017. ISSN 2045-2322. doi: 10.1038/s41598-017-17863-4. URL https://www.nature.com/articles/s41598-017-17863-4.

K. Crailsheim. Trophallactic interactions in the adult honeybee (Apis mellifera L.). *https://dx.doi.org/10.1051/apido:19980106*, 29, Jan. 1998. doi: 10.1051/apido:19980106.

R. De Marco and W. Farina. Changes in food source profitability affect the trophallactic and dance behavior of forager honeybees (Apis mellifera L.). *Behavioral Ecology and Sociobiology*, 50(5):441–449, Oct. 2001. ISSN 0340-5443, 1432-0762. doi: 10.1007/s002650100382. URL https://link.springer.com/article/10.1007/s002650100382. 00055.

W. M. Farina and J. A. Núñez. Trophallaxis in honey bees: transfer delay and daily modulation. *Animal Behaviour*, 45(6):1227–1231, 1992. ISSN 0003-3472. URL https://www.academia.edu/32891470/Trophallaxis_in_honey_bees_transfer_delay_and_daily_modulation.

K. v. Frisch. The dance language and orientation of bees. *Harvard UniversityPress, Cambridge, Massachusetts*, 1967.

T. Gernat, V. D. Rao, M. Middendorf, H. Dankowicz, N. Goldenfeld, and G. E. Robinson. Automated monitoring of behavior reveals bursty interaction patterns and rapid spreading dynamics in honeybee social networks. *Proceedings of the National Academy of Sciences*, 115(7):1433–1438, Feb. 2018. ISSN 0027-8424, 1091-6490. doi: 10.1073/pnas.1713568115. URL http://www.pnas.org/lookup/doi/10.1073/pnas.1713568115.

X. Glorot, A. Bordes, and Y. Bengio. Deep sparse rectifier neural networks. In G. Gordon, D. Dunson, and M. Dudík, editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 315–323, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR. URL http://proceedings.mlr.press/v15/glorot11a.html.

J. Goyret and W. M. Farina. Trophallactic chains in honeybees: a quantitative approach of the nectar circulation amongst workers. *Apidologie*, 36(4):595–600, 2005. ISSN 0044-8435. URL https://doi.org/10.1051/APIDO:2005050.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. URL https://arxiv.org/abs/1512.03385.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. URL http://arxiv.org/abs/1412.6980.

D. Naug. Structure of the social network and its influence on transmission dynamics in a honeybee colony. *Behavioral Ecology and Sociobiology*, 62(11):1719–1725, Sept. 2008. ISSN 0340-5443, 1432-0762. doi: 10.1007/s00265-008-0600-x. URL `https://link.springer.com/10.1007/s00265-008-0600-x`.

S. Pizer, E. Amburn, J. Austin, R. Cromartie, A. Geselowitz, T. Greer, B. ter Haar Romeny, J. Zimmerman, and K. Zuiderveld. Adaptive histogram equalization and its variations. *Computer Vision, Graphics, and Image Processing*, 39(3):355–368, 1 1987. ISSN 0734-189X. doi: 10.1016/S0734-189X(87)80186-X.

B. Settles. Active learning literature survey. *Computer Sciences Technical Report 1648, University of Wisconsin-Madison*, 2009. URL `http://burrsettles.com/pub/settles.activelearning.pdf`.

F. Wario, B. Wild, M. J. Couvillon, R. Rojas, and T. Landgraf. Automatic methods for long-term tracking and the detection and decoding of communication dances in honeybees. *Behavioral and Evolutionary Ecology*, page 103, 2015. doi: 10.3389/fevo.2015.00103. URL `https://journal.frontiersin.org/article/10.3389/fevo.2015.00103/full`.