Automated Mouse Behavior Recognition using VGG Features and LSTM Networks

Gregory Kramida
and Yiannis Aloimonos
Department of Computer Science
University of Maryland
Colleege Park, Maryland 20742
Email: gkramida@umiacs.umd.edu

Chethan Mysore Parameshwara
and Cornelia Fermüller
Maryland Robotics Center
University of Maryland
Colleege Park, Maryland 20742
Email: cmparam9@umiacs.umd.edu

Nikolas Alejandro Francis
and Patrick Kanold
Department of Biology
University of Maryland
Colleege Park, Maryland 20742

***Abstract*—We present a mouse behavior classification method using a recurrent neural network with the long short-term memory (LSTM) model. The experimental hardware used to collect the data is a custom mouse cage with four stereo-camera pairs in each wall. Using as input the different videos, our computational method employs a so-called end-to-end learning approach: visual features from pre-trained convolutional neural networks are extracted from each image frame, and used to train a customized LSTM-based model in weakly-supervised fashion, to recognize different behaviors of the mouse in the videos. Future extensions of the system will incorporate 3D feature information from the stereo cameras and online classification functionality.**

I. Introduction

Mice and other rodents are a common animal model in biology and biomedical research. For example, mice are widely used to identify behavioral symptoms in models of human neurological diseases. Since behavior recognition is used to gauge responses of mice to external stimuli and assess readiness for procedures, it has extended applications in animal training and other research tasks, as well as pharmaceutical testing. However, manual behavior assessment of multiple rodents over extended periods of time is laborious, tedious, and error-prone even when performed by an expert. This prompts a need for better automation of behavioral assessment, robust to differences between the animal subjects.

In previous work, Jhuang et al. [1] applied background subtraction to the footage of a typical mouse cage made out of translucent plastic taken from a single vantage point. They traind an SVM to classify motion features based on optical flow. Together with position and velocity-based features, these were used to train a SVMHMM (Hidden Markov Model Support Vector Machine) to classify every frame of a video sequence into a behaviour of interest with reasonable reliability, 78.3%, on their entire dataset. The result of classification was compared with the commercially-available software HomeCageScan 2.0 (CleverSys Inc.).

Giancardo, et al. [2] present a framework for behavior recognition of multiple mice in a group setting. They extract spatiotemporal features from a position tracker, employing mouse heat signatures, to classify behavior phenotypes with random forests. Their results were recorded for different combinations of interaction types between various numbers of mice and were compared against human annotators.

More recently, Hong et al. [3] extract a set of 26 basic spatiotemporal features from each frame recorded with a single RGB-D camera positioned above the mouse cage. The features were used to train several models, each for detection of a single social behavior of the mice, using SVM, adaboost, and random forests. The latter approach yielded the best result. The technique was highly accurate on sequences longer than one second.

In this report we present a framework targeting recognition of both complex and basic mouse behaviors. We show that this framework can be used to classify behaviors which involve detailed motion as well as behaviors which only last a fraction of a second.

II. Extendable Hardware Setup

For this study, we assembled a custom cage (Figure 1) with camera nests to gather the data. Each of the four walls has a camera nest housing a custom stereo camera, to allow the system to be extended to multiple views. Multiple cameras are used to ensure the mouse is in view by at least three of the cameras at all times. The cameras have lenses with 120° horizontal angular FOV. The lenses were manually refocused

Fig. 1. Custom cage with camera nests.

to 30 centimeters, which constrained the overall depth-of-field, but allowed to keep a large part of the cage in focus. The 1920x1080 resolution of the cameras allows for acquisition of the mouse in high-enough detail regardless of its position, while the 60 hz frame rate allows to capture the mouse fast enough to gain enough information even about very short behaviour sequences. In fact, many sequences within our data span for less than one or only several seconds.

Both synchronization of video streams and analysis of behavior data are currently done in post-processing. We expect that real-time processing that is planned for future work will require different cameras, which will be streaming video to several machines synchronized to a single time server.

III. Classification Framework

Our classification framework relies on two machine-learning mechanisms. Pre-trained VGG features are extracted from each frame of monocular video input. These features are then used as input to an LSTM recurrent neural network. Training of the LSTM model only requires the labels of the behavior in video sub-sequences.

A. VGG feature extraction

In a preprocessing step, the independent multimodal background subtraction (IMBS) algorithm[4] is used to segment out the mouse. The VGG model[5] is then applied to per-frame bounding boxes with the masked mouse to extract a 4096-entry-long vectors of features that are significant for basic vision tasks. The VGG features are applied via the widely-used Caffe deep learning framework[6].

B. LSTM setup

We use an LSTM [7] setup to classify behavior sequences. Our model is made with custom code implemented in python using the Theano library[8]. The model uses an embedding weight layer to reduce the input sequence dimensions to the LSTM layer dimensions, uses a standard LSTM with its own sets of input-unit weight, hidden-layer weight, and bias matrices, a time-propagating cell unit (the "memory" of the model), input, output, and forget gates. Within each training batch, feature vector sequences are normalized to the maximum length by employing a mask that truncates the effect of padded-on "empty" entries at the end of each sequence. Output of the LSTM is then fed into a standard linear perception layer, consisting of a weight and a bias matrix. During training, the prediction errors back-propagate through the entire model using the ADADELTA learning-rate optimizer[9] and standard L2 regularization, updating all the mentioned weight and bias matrices. The dropout techinique [10] is also used on training predictions to limit overfitting.

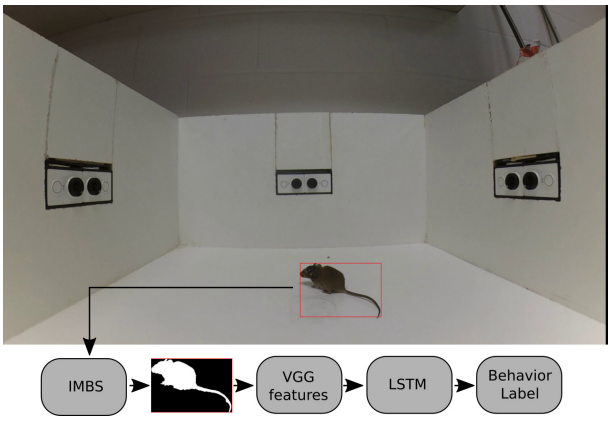

Fig. 2. Data flow used for behavior classification.

IV. Experiment Results

The current preliminary experiment was run on 40 minutes of monocular video of a single mouse. The video data consists of 270 consecutive sequences of per-frame VGG feature vectors of length 4096. Each sequence was annotated by a single trained observer with one of four behaviors: crawling, grooming, rearing, and scratching. The class distribution of sequences in the dataset was (in same order): 60%, 13.7%, 24.8%, and 1.5%. Crawling is the prevalent behavior in the data set, both in terms of sequence count and duration. Grooming is characterised by the mouse standing up on its rear paws and brushing its snout with its front paws from the ears down to the nose, and is important for identifying when the mouse is calm, for instance, to understand whether it is ready for training. Other behaviors chosen also have significant implications for various biology studies.

The set of sequences was randomized and separated into a training, validation, and test sample sets (60%, 20%, and 20% of the data, respectively). The validation portion, as in typical machine learning experiments, is used to prevent overfitting. The model obtained optimum performance on the test set after 100 epochs of training, producing errors of 3.08%, 14.81%, and 7.4% on the training, validation, and testing sets respectively. Confusion matrix for the test set classification in Figure 3 shows how well the model handles each behavior. The "scratching" behavior is currently severely underrepresented in the data, hence our model does not perform well on it.

V. Concluding Remarks and Future Work

The current result has shown that the LSTM model, in combination with learned features, can be effective in automated behavior classification of fast-moving animals.

We are continuing this line of research by incorporating data from multiple (four) views around the mouse into the classification, devising a real-time classification scheme which uses a stabilization threshold to estimate the correct behavior category at an arbitrary point in time, and, eventually, integrating the 3D data stream from stereo to classify more fine-grained behaviors. We will also attempt to estimate skeletal poses of the mice over time.

The dataset will be expanded to include more subjects (different mice, multiple mice at once), labeled by multiple human observers, and made publicly available to other researchers. We will also publish the code of our LSTM model for other researchers' convenience as an open-source package.

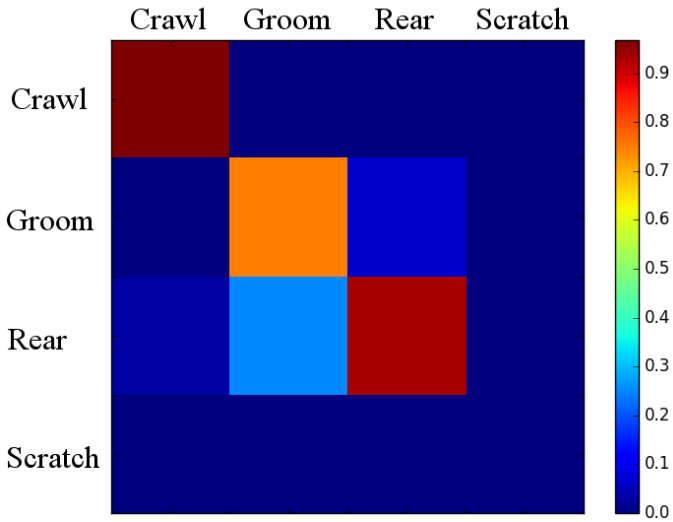

Fig. 3. Confusion matrix for test classification results.

REFERENCES

[1] H. Jhuang, E. Garrote, J. Mutch, X. Yu, V. Khilnani, T. Poggio, A. D. Steele, and T. Serre, "Automated home-cage behavioural phenotyping of mice." *Nature communications*, vol. 1, no. 5, p. 68, 2010. [Online]. Available: http://dx.doi.org/10.1038/ncomms1064

[2] L. Giancardo, D. Sona, H. Huang, S. Sannino, F. Managò, D. Scheggia, F. Papaleo, and V. Murino, "Automatic Visual Tracking and Social Behaviour Analysis with Multiple Mice," *PLoS ONE*, vol. 8, no. 9, 2013.

[3] W. Hong, A. Kennedy, X. P. Burgos-Artizzu, M. Zelikowsky, S. G. Navonne, P. Perona, and D. J. Anderson, "Automated measurement of mouse social behaviors using depth sensing, video tracking, and machine learning," *Proceedings of the National Academy of Sciences*, vol. 112, no. 38, pp. E5351–E5360, 2015. [Online]. Available: http://www.pnas.org/content/112/38/E5351.abstract

[4] D. Bloisi and L. Iocchi, "Independent multimodal background subtraction." in *CompIMAGE*, 2012, pp. 39–44.

[5] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014. [Online]. Available: http://arxiv.org/abs/1409.1556

[6] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22Nd ACM International Conference on Multimedia*, ser. MM '14. New York, NY, USA: ACM, 2014, pp. 675–678. [Online]. Available: http://doi.acm.org/10.1145/2647868.2654889

[7] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997. [Online]. Available: http://dx.doi.org/10.1162/neco.1997.9.8.1735

[8] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, "Theano: A CPU and GPU Math Compiler in Python," in *Proceedings of the 9th Python in Science Conference*, S. van der Walt and J. Millman, Eds., 2010, pp. 3–10.

[9] M. D. Zeiler, "ADADELTA: an adaptive learning rate method," *CoRR*, vol. abs/1212.5701, 2012. [Online]. Available: http://arxiv.org/abs/1212.5701

[10] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014. [Online]. Available: http://jmlr.org/papers/v15/srivastava14a.html