Navigation Turing Test (NTT): Learning to Evaluate Human-Like Navigation

Sam Devlin [*1] **Raluca Georgescu** [*1] **Ida Momennejad** [*2] **Jaroslaw Rzepecki** [*1] **Evelyn Zuniga** [*1]
Gavin Costello [3] **Guy Leroy** [1] **Ali Shaw** [3] **Katja Hofmann** [1]

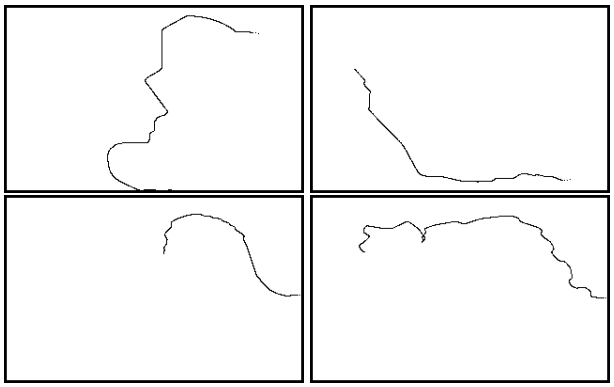

Figure 1. Inputs for the TD-CNN model. Random samples of two agent trajectories (top) and two human trajectories (bottom). Each image represents one whole trajectory/video, obtained by projecting the symbolic representation (agent position) along the "up" direction (z-coordinate).

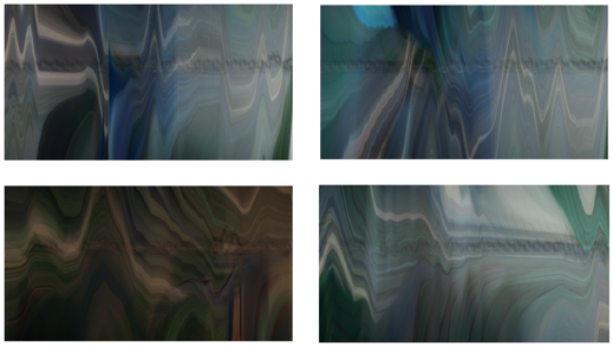

Figure 2. Inputs for the BC-CNN model. Random samples of two agent trajectories (top) and two human trajectories (bottom). Time is along the x-axis, each column in the image represents a single frame in the video where each colour channel has been separately averaged to compress the 2D frame to 1D in this representation. This allows us to represent an entire video the format expected by the VGG network: [COLOR, HEIGHT, WIDTH].

A. Appendix

A.1. Classifier Training Details

This section provides training details for our Automated Navigation Turing Test (ANTT) classifiers (described in Section 3 of the main paper).

To estimate the mean validation accuracy for hyperparameter tuning each model, we ran 5-fold cross-validation with an 80-20 split for training and validation. The training and validation sets are composed of a total of 100 episodes collected by four human players and 198 episodes of trained agents from two checkpoints. The test set is composed of the 40 videos shown in the Human Navigation Turing Test (HNTT, see Section 4 of the main paper), collected by three different human players and a different checkpoint for the trained agents (i.e., there was no overlap in players or agent checkpoints between the test and training/validation set). Human videos are selected by weighted sampling during cross-validation to account for class imbalance.

As discussed in the paper, our experiments consider different input formats to represent human and agent trajectories. To give readers a better understanding of the quality of the *top-down* (TD) and *bar-code* (BC) representations, we include additional examples in Figures 1 and 2.

For the VIS-FF, VIS-GRU, and TD-CNN models, we use a VGG network (Simonyan & Zisserman, 2014) pre-trained on the Imagenet dataset (Deng et al., 2009). The VGG's last layer is then replaced by a feedforward network (1 or 2 layers with dropout, depending on the hyperparameters) which is trained on our dataset.

Our hyperparameter tuning focused on reducing overfitting. We considered different dropout percentages (0%, 50%, 85%), hidden layer dimensions (0, 16, 32) and, for recurrent models, sequence lengths (5, 10, 20). Training efficiency was not a priority in our hyperparameter search as training was relatively fast. Each run took about 10 minutes on a single machine equipped with a Tesla V100 GPU and 6 Intel Xeon E5-2690 v4 CPUs. As such, hyperparameters such as

[*]Equal contribution [1]Microsoft Research, Cambridge, UK [2]Microsoft Research, New York, NY, USA [3]Ninja Theory, Cambridge, UK. Correspondence to: Sam Devlin <sam.devlin@microsoft.com>, Katja Hofmann <katja.hofmann@microsoft.com>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

batch size, optimizer, learning rate, and number of epochs were not explored. Our final best hyperparameter settings for each model are chosen based on their mean validation accuracy. This resulted in the following hyperparameters:

- SYM-FF: dropout 0%, hidden layer size 32, 50 epochs, batch size 256, Adam optimizer with learning rate 10^{-3}.
- SYM-GRU: dropout 0%, hidden layer size 32, sequence length 5, 50 epochs, batch size 256, Adam optimizer with learning rate 10^{-3}.
- VIS-FF: dropout 50%, hidden layer size 32, 10 epochs, batch size 8, Adam optimizer with learning rate 10^{-4}.
- VIS-GRU: dropout 50%, hidden layer size 32, sequence length 20, 10 epochs, batch size 8, Adam optimizer with learning rate 10^{-4}.
- TD-CNN: dropout 50%, no hidden layer, 10 epochs, batch size 32, SGD optimizer with learning rate 5×10^{-3} and momentum 0.9.
- BC-CNN: dropout 0%, hidden layer size 32, 10 epochs, batch size 8, Adam optimizer with learning rate 10^{-4}.

A.2. Human Navigation Turing Test - Procedure

This section provides additional details about the behavioral study (Section 4.2 of the main paper) used to collect human ground truth data for our Human Navigation Turing Test (HNTT).

Survey. Two HNTT studies were administered as anonymous surveys structured with Introduction, Background, and Task components as follows. The Introduction included an IRB-approved consent form and was followed by a Background page, which included a short description of Third Person Action Games, the game used in this study, and the research and task descriptions. Participants were asked to rank on a 5-point Likert scale the answer to: "How familiar are you with Third Person Action video games?" and "How familiar are you with the video game [title]?" (Figure 3).

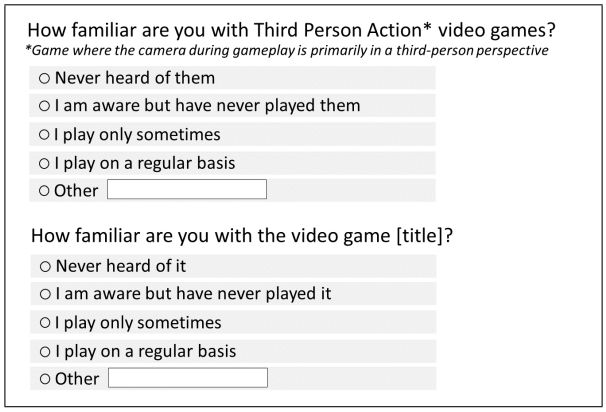

Figure 3. HNTT familiarity questions.

Task. The Task component of the survey consisted of ten Human Navigation Turing Test trials, in each of which participants watched two side-by-side videos (Video A and Video B), and were asked three questions about the videos. The first HNTT question, a two alternative forced choice (2AFC), was: "which video is more likely to be human?", to which the participant could respond by choosing "Video A is more likely to be human" or "Video B is more likely to be human". Participants followed by giving a free-form response to the question "why do you think this is the case? Please provide as much detail as possible". Finally they were asked to indicate on a 5-point Likert scale: "How certain are you of your choice?". See Figure 6 in the main paper for a screenshot of the HNTT trial.

A.3. Navigation Agent Training Details

This section provides details on our training procedure for the reinforcement learning agents (Section 4.3 in the main paper).

Agent architectures Our *symbolic* and *hybrid* agent architectures are referenced in Figure 4 in the main paper, containing hidden layer sizes for the fully connected layers. For the convolutional layers of the hybrid model, we used the following hyperparameters: and

The output of the convolutional layers was flattened and passed through a Dense layer of size 128 with ReLU activations (Zeiler et al., 2013).

Hyperparameter	**Value**
Batch size	2048
Dropout rate	0.1
Learning rate	3e-4
Optimizer	Adam
Gamma	0.996
Lambda	0.95
Clip range	0.2
Gradient norm clipping coefficient	0.5
Entropy coefficient	0.0
Value function coefficient	0.5
Minibatches per update	4
Training epochs per update	4

Table 1. Hyperparameters for training the symbolic and the hybrid agent models using PPO (Schulman et al., 2017).

Classifier	Identity Accuracy	Human-Agent Accuracy	Human-Agent Rank	Hybrid-Symbolic Accuracy	Hybrid-Symbolic Rank
SYM-FF	0.850 (0.062)	0.850 (0.062)	0.364 (0.043)	0.475 (0.166)	−0.244 (0.252)
SYM-GRU	0.850 (0.082)	0.850 (0.082)	0.173 (0.049)	0.400 (0.200)	−0.249 (0.210)
VIS-FF	0.633 (0.041)	0.633 (0.041)	−0.041 (0.160)	0.225 (0.050)	−0.165 (0.286)
VIS-GRU	0.767 (0.097)	0.767 (0.097)	0.220 (0.267)	0.425 (0.127)	−0.056 (0.331)
TD-CNN	0.583 (0.075)	0.583 (0.075)	0.222 (0.059)	0.525 (0.094)	−0.093 (0.149)
BC-CNN	0.717 (0.145)	0.717 (0.145)	−0.009 (0.131)	0.475 (0.050)	−0.095 (0.412)

Table 2. Classifier accuracy and rank compared to human judgments on held-out test data. All results are the mean (and standard deviation) from 5 repeats of training the classifier with hyperparameter settings chosen by their average validation accuracy in 5-fold cross-validation.

The size of the models' logit output is equivalent to the agents' discretized action space of size 8, which corresponds to the following valid actions: none, forward, left/right (by 30, 45, 90 degrees).

We used a fixed set of hyperparameters throughout all agent training, as shown in Table 1. These were found to perform best on preliminary experiments.

Training framework Both symbolic and hybrid agents were trained using the OpenAI Baselines PPO2 implementation (Dhariwal et al., 2017) running on Tensorflow 2.3 (Abadi et al., 2015), on top of a custom library for asynchronous data sampling.

Training infrastructure The symbolic model was trained on a CPU-only machine, with 64 Intel Xeon E5-2673 v4 2.3 GHz cores. The hybrid model made use of 1 GPU for training, an Nvidia Tesla K80 and 24 Intel Xeon E5-2690 v3 CPUs. The samples were collected from 60 parallel game instances, running in an Azure virtual scale set of 20 virtual machines (VMs). Each VM ran 3 separate game instances. Each simulation VM had one half of an Nvidia Tesla M60 GPU and 6 Intel Xeon E5-2690 v3 (Haswell) CPUs.

A.4. Evaluation Details

This section provides details for Section 5.2 in the main paper. All evaluation in this section is only on the held out test data set composed of the videos shown in the behavioral study. These videos were collected by different human players and a different checkpoint for the trained agents than those included in the training and validation data set.

To compare our models (that are trained to classify a single trajectory as either human or agent) and the responses from the behavioral study (where participants chose which of a pair of videos was more likely to be human) we must define a method for the models to pick the most human-like video from a pair. For models that classify a single full trajectory (TD-CNN and BC-CNN) we choose the video the model predicts to have the highest likelihood of being human. For models that classify sub-sequences from a video (SYM-FF, SYM-GRU, VIS-FF and VIS-GRU) we predict the class of every non-overlapping sub-sequence, then pick the video with the highest percentage of human sub-sequence classifications (i.e., we aggregate by the robust method of majority voting). This process gives us model responses to each question in the behavioral study that can then be compared to the participant responses.

For questions in the behavioral study that compared one human and one agent video (questions 1 to 6 in both studies, but note that questions were presented to participants in randomized order) we calculate:

Identity Accuracy: the accuracy of the model compared to the known origin of the video/trajectory (i.e., whether the trajectory was truly generated by a human player).

Human-Agent Accuracy: the accuracy of the models compared to the majority of study participants (i.e. we aggregate participant responses per question by majority vote.)

Human-Agent Rank: the Spearman rank correlation coefficient (Croux & Dehon, 2010) between two lists, each with one entry per question in the behavioral study. The first is ranked by the percentage of participants that agreed with the participants' aggregated majority vote choice. The second is ranked by either the likelihood (for TD-CNN and BC-CNN) or percentage of sub-sequences classified as human (for SYM-FF, SYM-GRU, VIS-FF and VIS-GRU) for the video chosen by the model as most likely to be human.

For questions in the behavioral study that included two agents (questions 7 to 10 in both studies) we calculate **Hybrid-Symbolic Accuracy** and **Hybrid-Symbolic Rank** which are equivalent to the corresponding metric for the human-agent questions. For these questions, there is no equivalent metric to Identity Accuracy as both videos are from agents and so the only comparison possible is to the human ground truth data obtained through out HNTT.

For each evaluation metric that we report in Figures 9 and 10 of the main paper, the mean (and standard deviation) by averaging the value measured for each of the five trained instances of a model on the five folds of the training data with the best hyperparameters obtained by 5-fold cross validation, as detailed in Section A.1. None of these models were trained on data from the test data set or other data from the same human players and agent checkpoints. For completeness, we include the raw data used to generate Figures 9 and 10 in Table 2.

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. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015.

Croux, C. and Dehon, C. Influence functions of the spearman and kendall correlation measures. *Statistical methods & applications*, 19(4):497–515, 2010.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Dhariwal, P., Hesse, C., Klimov, O., Nichol, A., Plappert, M., Radford, A., Schulman, J., Sidor, S., Wu, Y., and Zhokhov, P. Openai baselines. https://github.com/openai/baselines, 2017.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint:1707.06347*, 2017.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICLR)*, 2014.

Zeiler, M. D., Ranzato, M., Monga, R., Mao, M., Yang, K., Le, Q. V., Nguyen, P., Senior, A., Vanhoucke, V., Dean, J., et al. On rectified linear units for speech processing. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 3517–3521. IEEE, 2013.