Recurrent Neural Networks for Identifying Phases of the Honeybee Waggle Dance

Griffin Holt, C S 474 Deep Learning

I. The Problem

In the 1920s, Karl von Frisch conducted a series of experiments regarding the behavior of honeybees [1]. One of Frisch's most significant discoveries was that of the meaning and interpretation of the honeybee "waggle dance": forager honeybees use geometry through a cyclic dance to communicate information about located food sources [2]–[4]. In this report, I describe my attempts to utilize Recurrent Neural Networks (RNNs) and their variants to identify two different portions of this waggle dance: the run and the return phase.

A. Explanation of the Waggle Dance

Upon discovering a food source that is further than 100 meters from its resident hive, a forager honeybee will begin a cyclic dance–termed the "waggle" dance by entomologists–that follows the following form [2]–[4]: (see Fig. 1)

1) The forager bee walks in one direction along a linear path, "waggling" its abdomen from side to side as it progresses along this path. (This portion of the dance is referred to in the literature as the *run* or *waggle run*.)
2) The forager then turns either to the left or to the right, circling around to return to the start of the waggle run. (This portion of the dance is referred to as the *return phase*.)
3) The forager repeats the waggle run with the next return phase being to the opposite side.

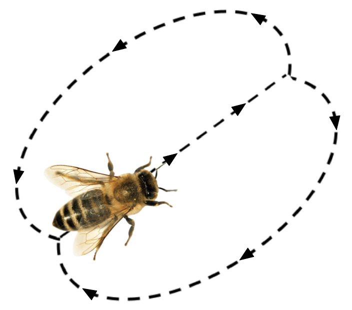

Fig. 1: The path of the honeybee waggle dance, including both the "waggle run" and two return phases.

B. Problem Description

In my personal research out of Information and Decision Algorithm Laboratories, I have been working for almost a year to construct a real-time translator of the honeybee waggle dance. A key requirement of this project will be the development of an algorithm that can properly classify a location point, in (x, y)-coordinates, of a dancing honeybee as being either on *the waggle run* or *the return phase*. There are currently no other algorithms or approaches to this problem in the existing literature.

This problem is a classification problem and my approach is supervised.

II. Dataset

My research group in IDeA Labs hand-annotated 28 videos of waggle dances pulled from YouTube. Each frame of each video was marked with the (x, y)-coordinate of the thorax of the honeybee performing the dance and with a label: 0 for the point being on the return phase, and 1 for the point being on the waggle run. Across all 28 videos, there are thus 9330 data points, complete with an x-coordinate, y-coordinate, and correct label (0 or 1).

A. Data Exploration

As can be observed in Fig. 2, the waggle run is much more active than the return phases. Along the run, the thorax of the bee oscillates very quickly as the bee walks; along the return phases, the thorax does not oscillate and follows the smooth arc of the bee's path. Also, the waggle run is closer to the centroid of all the data points, whereas points on the return phases will be farther away.

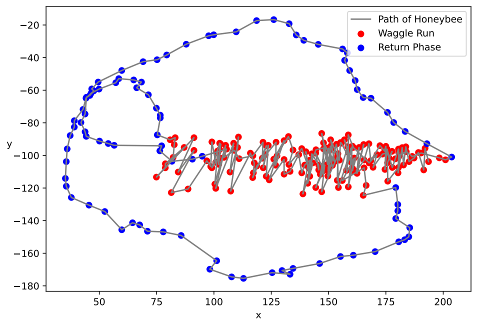

Fig. 2: A sample annotated honeybee waggle dance

B. Data Augmentation & Preparation

An input u_n to the RNN model at time step $t = n$ is defined by

$$u_n = \begin{bmatrix} x_n & y_n & \theta_n & d_n \end{bmatrix}^T, \quad (1)$$

where x_n is the horizontal coordinate of the bee at time $t = n$; y_n is the vertical coordinate of the bee at time $t = n$; θ_n is the angle at time $t = n$ between the previous location (x_{n-1}, y_{n-1}) and the next location (x_{n+1}, y_{n+1}) with the current location (x_n, y_n) at the center; and d_n is the Euclidean distance between the location (x_n, y_n) at time $t = n$ and the current centroid of the data $\left(\frac{1}{n}\sum_{i=1}^{n} x_i, \frac{1}{n}\sum_{i=1}^{n} y_i\right)$.

I added the latter two data points to augment the dataset, since they can assist the RNN in describing the oscillation along the waggle run and the distance of the return phase points from the center of the location data.

In addition, I split each of dances into sequences of complete cycles, thereby spreading out the data points across more sequences to be fed to the RNN models.

III. Methodology

I experimented with three different types of models: standard Recurrent Neural Networks (RNNs), Long Short-Term Memory Networks (LSTMs), and Gated Recurrent Units (GRUs).

Each model consists of a recurrent model with l recurrent layers and a single fully-connected linear layer mapping the final recurrent layer to 2 outputs. Each model was also kept to less than 200 parameters.

I designed the following progression of trials in order to find the recurrent network that best fits the data:

1) Perform a Grid Search on the RNN model, the LSTM model, and the GRU model. The Grid Search is performed across 3 or 4 different network topologies (defined by the number of layers and the size of the hidden layer); two different sequence overlap lengths ($w = 10, 20$); and three different learning rates ($\mu = 0.0005, 0.001, 0.005$).
2) Perform the same Grid Search, now with the order of the input sequences shuffled each epoch.
3) Perform the same Grid Search, now with loss normalization implemented. This is repeated once more with shuffled sequences.
4) Perform the same Grid Search, now with both loss normalization and gradient clipping implemented. Different maximum gradient norms were added to the Grid Search. This is repeated once more with shuffled sequences.

Models were trained on all of the dances except one, "Waggle Dance 18", which was used as the test set–using the Adam optimizer and Cross-Entropy Loss for either 300, 400, 500, or 600 epochs, depending on the type of model (LSTMs and GRUs tended to require more epochs for training).

Each Grid Search would return the best fit model for each set of hyperparameters. These best fit models were then evaluated on the test set according to three measures: Cross-Entropy Loss, Classification Accuracy, and Levenshtein Edit Distance. For the accuracy and edit distance measures, the *argmax* of the output probabilities was used as the output.

IV. Analysis of the Results

42 of the 130 best models reported from the Grid Searches reported a test classification accuracy above 75%. 11 of the best models reported a test classification accuracy above 80% (displayed in Table II.) The baselines for comparison are threefold: first, the baseline formed from guessing only 0's; second, the baseline formed from guessing only 1's; and third, the baseline formed from sampling 0 or 1 from a uniform distribution. The All-Zeroes Baseline has the highest scores for this test set–since there are more points on the return phase than the waggle run in our test dance, Waggle Dance 18.

The best model–the Recurrent Neural Network with 2 recurrent layers and a hidden size of 6, trained over 500 shuffled epochs with loss normalization and the gradient clipped to a value of 30–achieved 82.7% classification accuracy and a Levenshtein edit distance of 67. (Note, however, that the second-best model–the LSTM with 2 layers and hidden size 3, trained over 500 shuffled epochs with normalized loss and no gradient clipping–had the lowest Cross-Entropy Loss, 0.448). Thus, the model improved significantly over the All-Zeros Baseline. The best model also had a True Return Phase Rate of 84.1% and a True Waggle Run Rate of 80.3%, suggesting that it performed slightly better at classifying return phase points than waggle run points.

	Predicted: 0	Predicted: 1	
Actual: 0	253	48	301
Actual: 1	34	139	173
	287	187	474

TABLE I: Confusion Matrix for the Best Model

I am certain that no over-fitting occured: the test dance was from a completely different YouTube video and a completely different beehive than the other videos; thus, the test dance truly represents novel data never seen before by the algorithm.

It is also important to note that the best model was not one of my first models: in fact, it was one of the last models trained (on the final iteration of improving models using Gradient Clipping).

V. Future Work

The final classification accuracy of the best model, 82.7%, will serve as an excellent baseline moving forward for my research group. Our next steps will be to attempt to beat this RNN accuracy by utilizing unsupervised learning techniques, such as the Fourier Transform, to distinguish between the points on the waggle run and return phases.

References

[1] T. Munz, *The Dancing Bees: Karl von Frisch and the Discovery of the Honeybee Language*. The University of Chicago Press, 2016.

[2] K. R. von Frisch, *The Dancing Bees: An Account of the Life and Senses of the Honey Bee*. Methuen and Co Ltd, 1966.

[3] ——, *The Dance Language and Orientation of Bees*. Belknap Press of Harvard University Press, 1967.

[4] ——, *Bees: Their Vision, Chemical Senses, and Language*, revised ed. Cornell University Press, 1971.

Model	**Hidden Size**	**# Recurrent Layers**	**Total # Parameters**	**Training Epochs**	**Nmlzd. Loss**	**Grad. Clip**	**Shuffled**	**Test CE Loss**	**Test Accuracy**	**Test Levenshtein Edit Dist.**
All-Zeros Baseline	-	-	-	-	-	-	-	**0.678**	**63.5**	**173**
All-Ones Baseline	-	-	-	-	-	-	-	0.9483	36.5	301
Random Baseline	-	-	-	-	-	-	-	0.6931	50.0	189 - 239
RNN	6	2	170	500	Yes	30	Yes	0.463	**82.7**	**67**
LSTM	2	3	166	500	Yes	None	Yes	**0.448**	82.3	74
GRU	2	5	198	600	Yes	2	Yes	0.481	81.6	72
LSTM	3	2	166	500	Yes	None	Yes	0.504	81.6	76
RNN	5	2	127	500	Yes	10	No	0.450	81.2	75
GRU	3	2	161	600	Yes	1	Yes	0.461	81.0	75
GRU	2	3	126	600	Yes	None	No	0.470	80.8	75
RNN	6	2	170	300	No	None	No	0.467	80.4	80
RNN	5	2	127	500	No	None	Yes	0.458	80.4	76
GRU	2	3	126	600	Yes	1	Yes	0.498	80.4	78
RNN	5	3	187	500	Yes	30	No	0.467	80.4	79

TABLE II: Models with $> 80\%$ Test Accuracy

Date	Time Spent	Category	Details
Tue, 11/2/21	6:05	Prep Work	- Prepared my research journal - Prepared the Jupyter notebook for my initial RNN experiments - Started writing the pipeline to get the data from .CSV format to be ready for PyTorch (split into training and test) - Computed a few additional input data points for the RNN - Fixed some mistakes (outliers) in the data & fixed the data prep algorithm as well - Wrote up my Final Dataset Project report (due Nov. 20)
Tue, 11/9/21	2:22	Research	- Read through Ch. 10 of the Deep Learning textbook to become more familiar with RNNs (specifically about how to combat vanishing/exploding gradient issues) - Read through Ch. 15 of Hands-On Machine Learning to learn more about RNNs, LTSMs, & GRUs (esp. how to use them in practice) - Reread The Unreasonable Effectiveness of Recurrent Neural Networks - Reread Understanding LSTMs - Researched how to do Peephole Connections for LSTMs in PyTorch - Researched how to do Gradient Clipping in PyTorch - Researched 1D Convolutional Networks for Time Series - Explored Levenshtein Distance as a possible accuracy measurement for final models - Based on my notes from my research, I began to plan out and design my experiment in using RNNs
Mon, 11/15/21	2:50	Prep Work	- Reworked my data pipeline (included only 0s and 1s) - Removed all the ”travels” from the data - Split up dances into cycled sequences so as to spread out the data - THE DATA IS NOW READY
Mon, 11/15/21	4:30	Model Work	- Started experimenting with the Standard RNN - Ran different combinations of Learning Rates, Hidden State sizes, and Numbers of Layers - Achieved some initially promising results. I think the LTSM and GRU will definitely be better.
Tue, 11/16/21	0:20	Research	- Met with Eric (the TA) to discuss different baselines to use for comparison for the RNN models
Tue, 11/16/21	3:30	Model Work	- Planned out & wrote the code for the Grid Search for the Standard RNN - Fixed a few things in the data pipeline - Ran the Grid Search for the Standard RNN - Received 4 top Standard RNN results. Recorded their hyperparameters to refine all 4 models & choose the best one
Wed, 11/17/21	2:20	Model Work	- Ran the fine tuning on the top 4 Standard RNNs - Started the Grid Search for the Standard LSTM - Fine-tuned 2 of the LSTM models - Fine-tuned another of the LSTM models.
Thu, 11/18/21	2:00	Model Work	- Met with Eric (TA) & discussed including another data point in my input (the classification of the previous point) - We arrived at the conclusion that without that data point, my RNNs are not taking advantage of recurrence and memory (and are little more than a fancy linear network). - Trained the new RNNs using the additional column - Wrote the training code & discovered that my old RNNs are still doing really well (despite my previous thoughts on the subject)
Thu, 11/18/21	0:10	Prep Work	- Added the additional column of the previous movement classification to the input data
Fri, 11/19/21	5:00	Model Work	- Ran tests with the additional column of the previous movement classification to the input data (The results of these tests suggest that this new data point is too highly correlated with the output, and for such a small dataset, leads simply to the memorization of the dataset without gaining any insight into the structure of the sequences) - Reverted data back to original 4 inputs - Fixed a bug in my training function (the test data was chosen incorrectly) - Wrote the testing function and separated the test data - Started my official experiments: Standard RNN w/ Grid Search for various topologies; similarly, Standard LSTM
Sat, 11/20/21	5:00	Model Work	- Continued my experiments from the previous day: this time, with Grid search across LSTM models and GRU models - The LSTM models and GRU models have not been as effective, generally, as the RNNs at this task
Mon, 11/22/21	1:10	Model Work	- Worked more on training the RNNs, specifically with loss normalization & shuffling (able to achieve 81% accuracy so far) - Planned out next steps to refine the models further - Started writing the code for Gradient Clipping
Tue, 11/23/21	1:00	Model Work	- Ran the code for Gradient Clipping on the RNN models (took 10 hours to run)
Wed, 11/24/21	1:00	Model Work	- Ran the code for Gradient Clipping on the LSTM models (took 24 hours to run)
Thu, 11/25/21	1:00	Model Work	- Ran the code for Gradient Clipping on the GRU models (took 24 hours to run)
Mon, 12/6/21	4:00	Model Work	- Wrote and tested an ensemble approach: didn't do any better than the individual models (wasn't worth adding it to the report)
Mon, 12/6/21	4:00	Model Work	- Compiled the finals results & wrote the final report

Research	Prep Work	Model Work	Total Hours
2:42	9:05	34:30	46:17

TABLE III: Project Log