You’re Not Alone in Battle: Combat Threat Analysis Using Attention Networks and a New Open Benchmark (Appendix)

Anonymous Author(s)

A DATASET DETAILS

Dataset Statistics. In section 3, we have demonstrated that combats T_c’s from each tactic have different trajectories and feature distributions. Here, we further detail their label statistics w.r.t. each tactic. Table 1 shows the intentions class distribution shifts across different tactics, which partially explains the generally low performance of intention prediction.

Data Split. Table 1 shows that intention class 5 (Forceful Engagement) is only present in tactic 4. Thus, we do not choose tactic 4 as the test set (Recall that only the combats of one tactic serve as the test set).

Time Sampling. In sampling t_{max} timestamps, we sample the timestamps before the first attack occurs in combat. That is, if the first attack in combat T_c occurs at $t = 300$, we randomly sample $t_{max} = 20$ timestamps from $t \in [1, ..., 299]$.

Feature Semantics. Here, we provide the detailed meaning of each feature dimension. The first three dimensions are latitude(degree), longitude(degree), and altitude(meter), respectively. The 4-6th dimensions are attitudes: yaw(deg); pitch(deg); roll(deg). Speed (km) is the 7th dimension, and the 8th dimension is the force identifier, where 0s correspond to the friendly and 1s to the hostile. The last five dimensions are terrain identifiers for each entity, including Road, Forest, Open Lane, Hiding Place, and Building.

Standardization. We standardize the input features since their changes in scale are highly different across timestamps. For continuous features (coordinates, attitude, and speed), we use the overall feature dimension mean μ and standard deviation σ^2 values from the train set. Specifically, the normalized continuous features are $\frac{x-\mu}{\sigma^2}$, where x is the continuous feature element. Importantly, we do not normalize each combat T_c at an independent scale, since it may cause significant information loss. If we standardize each combat T_c independently, the coordinate feature distribution can be similar between a combat with close squads and one with distant squads. We do not standardize the binary features.

B TRAINING DETAILS

Optimization. All neural network models are optimized with Adam optmizer [2]. The number of epochs is 100, and the best model is chosen based on the train loss. A small number of epochs is chosen due to the absence of a validation set. We find a larger number of epochs generally do not contribute to performance gain, often causing overfitting to the train set. For kNN and XG-Boost, we use the default optimization algorithms provided in scikit-learn [3] and XG-boost [1] libraries.

Table 1: Label Statistics Per Tactic

	Intention	Labels					Attack Labels
Tactic	#(TE)	#(MT)	#(CR)	#(SS)	#(FE)	#(SP)	#(Attacks)
Linear Advancement	1236	1236	618	0	0	618	1916.
Sequential Progression	4	1244	1862	0	0	622	2110
Flanking Maneuver	0	0	1842	1228	0	614	2455
Direct Engagement	0	0	1866	4	1550	312	3130

• **Abbreviations**: TE = Tactical Engagement. MT = Maneuvering Techniques. CR = Coordinated Rendezvous. SS = Strategic Surprise. FE = Forceful Engagement. SP = Strategic Positioning.

Loss Reweighting. Due to class imbalance in both intention and attack prediction, we reweight the loss based on the train label distribution. Specifically, we choose the reciprocal of the label counts in the train set to reweight the loss.

Hyperparameters. We optimize the hyperparameters with a grid search for neural network models. Specifically, for each model, the dropout rate $\in \{0.0, 0.3, 0.5\}$, learning rate decay $\in \{5e-5, 1e-4, 5e-5, 1e-5, 5e-6, 1e-6\}$, the number of layers $\in \{2, 4\}$ are searched. For kNN and XG-Boost, we use default hyperparameter settings provided in [1, 3].

REFERENCES

[1] Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In ACM SIGKDD International Conference on Knowledge Discovery & Data Mining.
[2] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014).
[3] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in Python. the Journal of machine Learning research 12 (2011), 2825–2830.

, ,
© 2023 Association for Computing Machinery.
ACM ISBN . . . $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn