
A Reproducibility Report on the CLVSA Model

Charles Ran

Sky Yun

Abstract

1 Reproducibility is a critical component of rigorous machine learning research.
2 In this work, we attempt to reproduce the core experimental results of Wang et
3 al. 2019 through an independent implementation. We provide an open-source
4 PyTorch implementation of CLVSA and systematically evaluate its empirical
5 performance. In addition, we conduct ablation studies to assess the contribution
6 of key model components, including the variational component, and examine
7 whether the reported improvements can be reproduced. We further explore small
8 architectural extensions to investigate potential performance improvements.
9 Our results do not reproduce the performance gains reported in the original work,
10 and ablation experiments suggest that certain components contribute less than
11 expected under our implementation. We identify several potential sources of
12 discrepancy, including underspecified implementation details and sensitivity to
13 training setup. Finally, we discuss the strengths and limitations of the proposed
14 approach.

15 1 Introduction

16 Reproducibility is essential for validating empirical claims in machine learning. However, prior
17 work has shown that incomplete reporting of implementation details, hyperparameters, and data
18 preprocessing steps can lead to discrepancies between reported and reproduced results (Semmelrock
19 et al. 2025). Reproducibility studies therefore play a critical role in assessing the reliability and
20 robustness of proposed methods.

21 Prior work by Wang et al. (2019) proposes a Convolutional LSTM-based Variational Seq2Seq model
22 with Attention (CLVSA), which incorporates a variational latent space to explicitly model uncertainty
23 in sequential financial data. The authors report improved predictive performance and generalization
24 relative to deterministic baselines, attributing these gains to the stochastic representation learned
25 through variational inference.

26 In this work, we conduct a reproducibility study of Wang et al. (2019), with the goal of indepen-
27 dently validating their core claims. Beyond evaluating the proposed model, we perform ablation
28 studies to examine the contribution of the model’s main architectural components, including the
29 variational component. We also evaluate small extensions to the original design, such as modified
30 feature transformations and scaled attention, to assess whether these refinements improve empirical
31 performance.

32 Our contributions are threefold: (1) we provide an independent PyTorch implementation of CLVSA,
33 (2) we systematically evaluate the contribution of its key components, including the variational
34 component, to assess whether the reported performance gains can be reproduced, and (3) we explore
35 small architectural extensions to determine whether further improvements can be achieved.

36 2 Background and Similar Works

37 Financial time-series forecasting remains a fundamentally challenging problem due to a high degree
38 of noise and the presence of latent stochastic factors that are difficult to model explicitly. As such,

39 several deep-learning approaches have been utilized to predict financial trends and prices despite the
 40 inherent complexity of the problem.

41 Work by Pawar, Jalem, and Tiwari (2019) showed that a vanilla LSTM recurrent neural network can
 42 predict stock trends more accurately compared to traditional machine learning algorithms. From this,
 43 the authors in Md et al. (2023) proposed a multi-layered sequential LSTM to predict stock prices
 44 through context-specific dependencies. Another similar work by Su et al. (2024) used attention-based
 45 convolutional networks to predict trends in financial markets.

46 While the aforementioned approaches have demonstrated empirical success, Wang et al. (2019)
 47 asserts that their deterministic nature may limit their ability to capture the inherent uncertainty in
 48 financial markets. Thus, Wang et al. (2019) proposes incorporating stochastic latent states.

49 3 Model Implementation

50 We implement the proposed CLVSA model and other baseline models as described in the original
 51 paper in PyTorch (Wang et al. 2019). Our implementation is available at <https://github.com/rancharles/clvsa>.

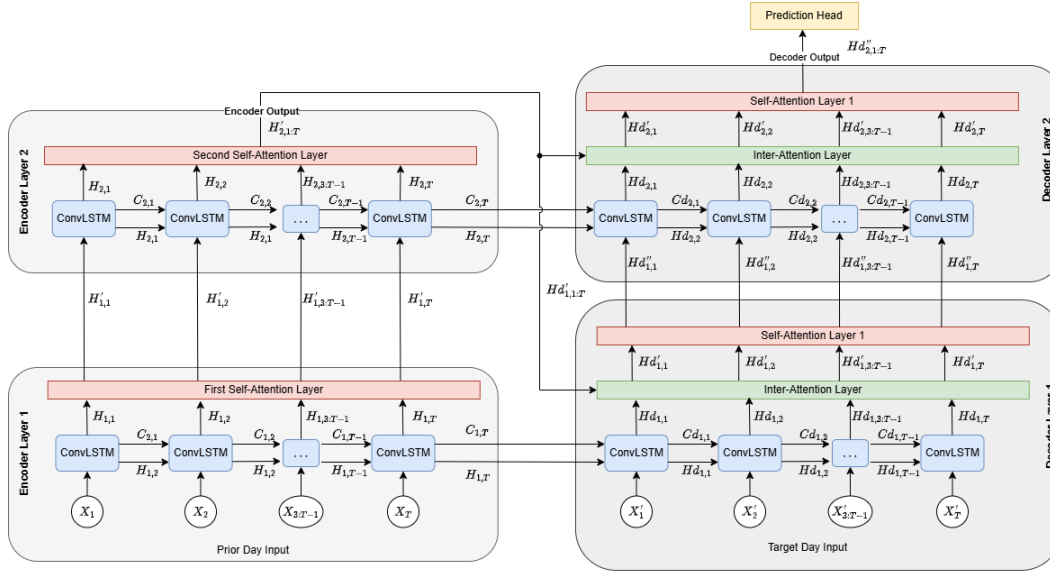


Figure 1: Interpretation of the CLSA architecture with ConvLSTM cells, self-attention, and inter-attention. CLVSA extends this model with a variational backward decoder.

52

53 The model is based on a sequence-to-sequence architecture with convolutional LSTM units, self-
 54 attention, inter-attention, and a variational backward decoder. The encoder and decoder each receive
 55 2-D financial data frames corresponding to two consecutive trading days. The encoder processes
 56 the first day, while the decoder models the second day using the encoder representation. CLVSA
 57 further introduces a backward decoder over the reversed second segment to construct an approximate
 58 posterior distribution and generate a KL-divergence regularizer (Wang et al. 2019).

59 3.1 Convolutional LSTM Seq2Seq Design

60 On a high level, the model uses a sequence-to-sequence encoder-decoder architecture (Sutskever,
 61 Vinyals, and Le 2014). The recurrent component is based on convolutional LSTM (convLSTM)
 62 units. Unlike a standard LSTM, whose gates are parameterized by fully connected transformations
 63 (Hochreiter and Schmidhuber 1997), a ConvLSTM replaces these transformations with convolutional
 64 kernels (Shi et al. 2015). Conceptually, this allows the recurrent cell to first extract local temporal
 65 patterns from input frames. Each input $X_t \in \mathbb{R}^{5 \times 6}$ represents a short window of market data, where

66 the rows correspond to the five OHLCV features (open, high, low, close, volume), and the columns
 67 correspond to six consecutive 5-minute time intervals.

68 Following Wang et al. (2019), we use 1×3 convolutional kernels that only move horizontally across
 69 columns. We use 32 output channels as in the original design, where each channel shares its kernel
 70 across all feature rows. In addition, we experiment with a row-specific variant where each channel
 71 has its own convolutional kernel. The motivation is that the different OHLCV features may exhibit
 72 different pattern dynamics.

73 3.2 Attention Mechanisms

74 The model incorporates both inter-attention and self-attention to capture dependencies across and
 75 within trading segments (Bahdanau, Cho, and Bengio 2014; Cheng, Dong, and Lapata 2016; Wang
 76 et al. 2019). Inter-attention connects the encoder and decoder by allowing each decoder state to attend
 77 over the encoder’s hidden states, enabling the model to condition predictions on relevant information
 78 from the previous segment. Self-attention is applied within each sequence to emphasize informative
 79 time steps.

80 In contrast to the proposed model, which uses unscaled dot-product attention, we adopt scaled
 81 dot-product attention. In our experiments, this scaling prevents the attention distribution from being
 82 overly sharp and leads to improved performance.

83 3.3 Variational Backward Decoder

84 The key extension from CLSA to CLVSA is the variational backward decoder. The motivation is that
 85 supervised labels such as *up*, *flat*, and *down* only provide a discretized summary of market movement.
 86 They do not fully preserve the variability contained in the original OHLCV sequence. To alleviate
 87 this bias, CLVSA introduces stochasticity into the latent states and regularizes the model using
 88 a KL-divergence term inspired by variational autoencoders (Kingma and Welling 2013; Rezende,
 89 Mohamed, and Wierstra 2014; Wang et al. 2019).

90 The forward decoder defines a prior distribution over latent variables,

$$p_{\theta}(z_t | x_{1:t}, z_{1:t-1}),$$

91 while the backward decoder defines an approximate posterior,

$$p_{\phi}(z_t | x_{T:t}, y'_t) = \mathcal{N}(\mu_t, \text{diag}(\sigma_t^2)).$$

92 The latent variable is sampled using the reparameterization trick:

$$z_t = \mu_t + \sigma_t \circ \epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, I).$$

93 z_t is sampled from the approximate posterior during training, and from the learned prior at evaluation
 94 and test time. (Wang et al. 2019).

95 3.4 Objective Function

96 The CLVSA objective combines three terms: the forward decoder prediction loss, the backward
 97 decoder prediction loss, and a KL-divergence penalty between the approximate posterior and the
 98 learned prior. We write the minimization objective as

$$\mathcal{J} = \mathcal{L}^{\text{forward}} + \alpha \mathcal{L}^{\text{backward}} + \beta D_{\text{KL}}(p_{\phi}(z | x, y) \| p_{\theta}(z | x))$$

99 Here, $\mathcal{L}^{\text{forward}}$ is the main forward decoder classification loss,

$$\mathcal{L}^{\text{forward}} = -\frac{1}{T} \sum_{t=1}^T \mathbb{E}_{p_{\phi}(z_{1:t}|x_{T:t},y'_t)} [\log p_{\theta}(y_t | x_{1:t}, z_{1:t})],$$

100 which in implementation is approximated by sampling z and then computing the usual cross-entropy
 101 over the decoder outputs. Similarly, $\mathcal{L}^{\text{backward}}$ is an auxiliary cross-entropy loss from the backward
 102 decoder over the reversed target sequence, controlled by a constant $\alpha = 2.5 \times 10^{-4}$ as in Wang et al.
 103 (2019). The KL term regularizes the approximate posterior $p_{\phi}(z | x, y)$ toward the learned prior
 104 $p_{\theta}(z | x)$ at each decoder time step, with β used as a KL annealing weight (Bowman et al. 2015;
 105 Wang et al. 2019). The original CLVSA objective also includes an explicit L_2 regularization term.
 106 In our implementation, we omit this term and instead use weight decay through AdamW, so weight
 107 regularization is handled by the optimizer rather than directly added to the loss.

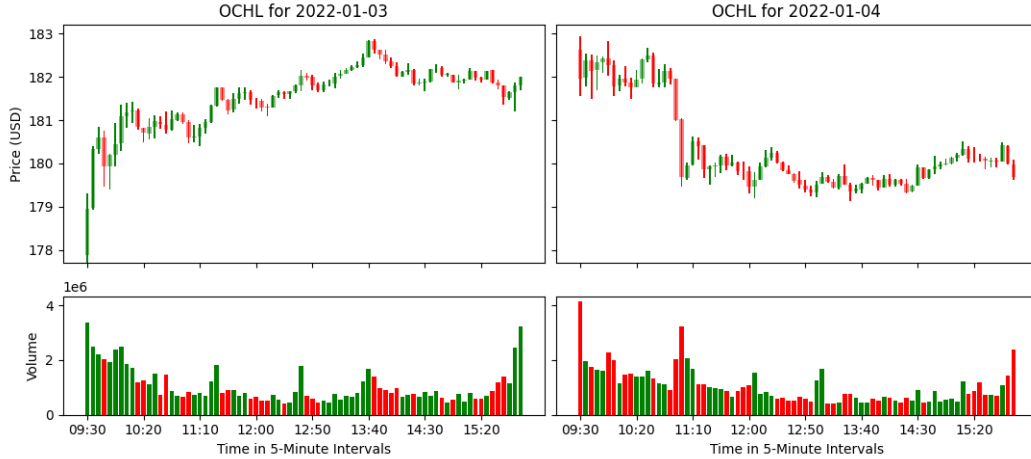


Figure 2: The OCHLV of two-consecutive days of AAPL stock starting from January 3rd, 2022

108 4 Methodology

109 4.1 Experimental Objectives

110 The primary objective of this study is to reproduce the key findings of Wang et al. (2019), specifically
 111 the reported performance improvements of CLVSA over common baseline models such as LSTM
 112 for financial time-series prediction, as well as the contribution of the variational component. Our
 113 secondary objective is to evaluate small extensions of CLVSA to assess whether further performance
 114 improvements can be achieved.

115 To this end, we train, evaluate, and compare CLVSA and its variants on two financial datasets similar
 116 to those used in the original work.

117 4.2 Datasets

118 Because the original datasets are not publicly released, we use publicly accessible Hugging Face
 119 OHLCV datasets. For futures data we use BTCUSDT and for equities we use AAPL stock, both from
 120 2021–2024. Figure 3 shows one particular example of AAPL OHLCV information.

121 Following Wang et al. (2019), we first aggregate the data into 30-minute frames. We then deviate from
 122 the original setup by transforming the raw OHLCV values into stationary features before constructing
 123 model inputs. For each frame t , the open, high, low, and close prices are expressed relative to the
 124 previous close price:

$$\tilde{o}_t = \log\left(\frac{o_t}{c_{t-1}}\right), \quad \tilde{h}_t = \log\left(\frac{h_t}{c_{t-1}}\right), \quad \tilde{l}_t = \log\left(\frac{l_t}{c_{t-1}}\right), \quad \tilde{c}_t = \log\left(\frac{c_t}{c_{t-1}}\right).$$

125 For volume, we use a log transform,

$$\tilde{v}_t = \log(1 + v_t).$$

126 These transformed values form the five input features used by the model. We found that these
 127 stationary transformations are essential for reducing regime drift, as they remove dependence on the
 128 absolute price levels.

129 Labels are constructed following Wang et al. (2019) by thresholding the next-frame close-price log
 130 return \tilde{c}_{t+1} . Observations above $b_{\text{up}} = \log((\mu_c + \lambda)/\mu_c)$ are labeled *up*, observations below $b_{\text{down}} =$
 131 $\log((\mu_c - \lambda)/\mu_c)$ are labeled *down*, and the rest are labeled *flat*. We choose λ to approximately
 132 balance the three classes.

133 4.3 Training Procedure

134 Each model is trained to classify the trend of the next 30-minute interval as *up*, *flat*, or *down*, given
135 all preceding 30-minute frames. Each frame contains normalized OHLCV features as described in
136 the previous subsection.

137 For each training run, we use a contiguous 3-year window starting from January 1 of the first year
138 to December 31 of the last year. The final 200 days are reserved for validation and testing, split
139 into consecutive 100-day validation and test sets. The remaining data is used for training. For each
140 training run, we select the threshold parameter λ to approximately balance the class labels in the
141 training set. Each training example consists of two consecutive days. Each example is a contiguous
142 two-day window: the first day forms the encoder source sequence, and the second day forms the
143 decoder target sequence.

144 We employ early stopping based on validation balanced accuracy (BACC), with a minimum of 8
145 epochs and a patience of 6 epochs. Models are trained using AdamW with a learning rate of 3×10^{-4}
146 and weight decay of 1×10^{-4} . The final model is selected based on the epoch achieving the highest
147 validation BACC.

148 Following Wang et al. (2019), we report mean average precision (MAP) as the primary evaluation
149 metric. In practice, we find that BACC provides a more stable signal for model selection during
150 training. All experiments are conducted on an RTX 3060 Ti GPU.

151 4.4 Model Variants

152 We train a plain LSTM baseline for comparison with CLSA and CLVSA. We also evaluate ablations
153 of CLSA to isolate the effects of its convolutional and attention components:

- 154 • **Attention:** We remove inter-attention, self-attention, or both attention mechanisms.
- 155 • **Convolutions:** We replace ConvLSTM cells with standard LSTM cells. We also evaluate a
156 row-specific ConvLSTM variant, where each OHLCV feature row has its own convolutional
157 kernels.

158 5 Experimental Results

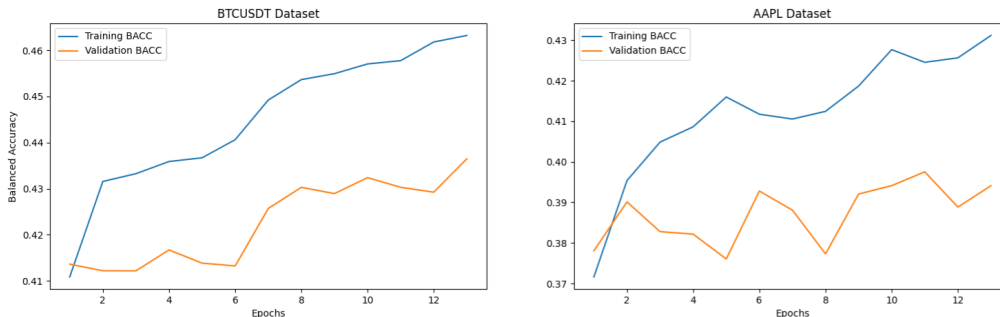


Figure 3: Validation balanced accuracy over training epochs for CLVSA on the BTCUSDT and AAPL datasets from 2022–2024.

Table 1: Experimental results of the testing MAP (%) corresponding to each trained model and dataset. CLS and LSA refer to the CLSA model without attention entirely and the convolutional part of LSTM cell, respectively.

Dataset	Model				
	CLVSA	CLSA	LSTM	CLS	LSA
BTCUSDT (Futures)	43.3	43.2	42.9	43.2	43.0
AAPL (Equity)	41.2	40.6	40.0	40.6	40.2

159 The results of the ablation studies suggest that both inter-attention and self-attention provide only
 160 small gains from 0.1 to 0.2 MAP. The ConvLSTM cell gives a modest improvement over standard
 161 LSTM cells of approximately 0.4 MAP. Our row-specific convolutional kernel extension did not
 162 improve performance, suggesting that shared kernels across OHLCV rows are sufficient.

163 Overall, our experimental results do not reproduce the large performance gains reported for CLSA and
 164 CLVSA over basic baselines such as LSTM. Table 1 reports the test MAP for each model and dataset.
 165 CLSA achieves only modest improvements over the LSTM baseline, and adding the variational
 166 component provides limited additional improvement around 0.5 MAP. In particular, we are unable
 167 to reproduce the consistent 2.0 MAP gains of CLVSA from CLSA nor reach the 45–50 MAP range
 168 reported by Wang et al. (2019).

169 However, we do observe that the variational regularizer improves training consistency, producing
 170 more stable validation performance across epochs. We also find that CLSA and CLVSA improve
 171 balanced accuracy (BACC) substantially. The LSTM baseline achieves a BACC (%) of 36.3 on
 172 the AAPL dataset. CLSA improves BACC by 2.9 over LSTM, and CLVSA provides an additional
 173 0.7 BACC improvement over CLSA. This is consistent with our use of validation BACC for early
 174 stopping, which also produced the best test MAP results in our experiments.

175 6 Conclusion

176 The aim of this paper is to reproduce the performance improvements reported for the CLSA and
 177 CLVSA models in Wang et al. (2019). We evaluate CLSA, CLVSA, and several variants on AAPL
 178 and BTCUSDT datasets.

179 Reproducing the results of Wang et al. (2019) proved challenging, partly because several implemen-
 180 tation and training details are underspecified. For example, the paper specifies that both attention
 181 mechanisms are used, but does not fully determine whether self-attention is applied before or after
 182 inter-attention, which affects the hidden states passed through the decoder.

183 Our results show that the vanilla LSTM baseline performs competitively in terms of test MAP,
 184 with CLSA and CLVSA producing only marginal improvements. This differs from Wang et al.
 185 (2019), where CLVSA substantially outperforms the baseline models. The discrepancy may be due
 186 to differences in dataset construction, training procedure, or interpretation of architectural details.
 187 Access to the original datasets and implementation would make it easier to isolate these factors and
 188 more directly reproduce the reported results.

189 Although we did not reproduce the large reported gains, several small extensions improved perfor-
 190 mance in our experiments. In particular, using log-relative scaling for stationary OHLCV features
 191 greatly improved performance by reducing regime shifts. Replacing unscaled dot-product attention
 192 with scaled dot-product attention further led to small improvements in empirical performance.

193 Overall, our findings highlight the difficulty of reproducing complex architectures from descriptions
 194 alone, and emphasize the importance of releasing precise implementation details, datasets, and code
 195 alongside proposed model designs.

196 Acknowledgments

197 We acknowledge the use of ChatGPT and Claude to search the literature and aid with the implemen-
 198 tation.

199 **References**

- 200 Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio (2014). *Neural Machine Translation by*
201 *Jointly Learning to Align and Translate*. arXiv: 1409.0473 [cs.CL].
- 202 Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Jozefowicz, and Samy Bengio
203 (2015). *Generating Sentences from a Continuous Space*. arXiv: 1511.06349 [cs.LG].
- 204 Jianpeng Cheng, Li Dong, and Mirella Lapata (2016). *Long Short-Term Memory-Networks for*
205 *Machine Reading*. arXiv: 1601.06733 [cs.CL].
- 206 Sepp Hochreiter and Jürgen Schmidhuber (1997). “Long Short-Term Memory.” *Neural Computation*
207 9.8, pp. 1735–1780. DOI: 10.1162/neco.1997.9.8.1735.
- 208 Diederik P. Kingma and Max Welling (2013). *Auto-Encoding Variational Bayes*. arXiv: 1312.6114
209 [stat.ML].
- 210 Abdul Quadir Md, Sanjit Kapoor, Chris Junni A.V., Arun Kumar Sivaraman, Kong Fah Tee, Sabireen
211 H., and Janakiraman N. (2023). “Novel optimization approach for stock price forecasting using
212 multi-layered sequential LSTM.” *Applied Soft Computing* 134, p. 109830. ISSN: 1568-4946. DOI:
213 <https://doi.org/10.1016/j.asoc.2022.109830>. URL: <https://www.sciencedirect.com/science/article/pii/S1568494622008791>.
- 214 Kriti Pawar, Raj Srujan Jalem, and Vivek Tiwari (2019). “Stock Market Price Prediction Using
215 LSTM RNN.” *Emerging Trends in Expert Applications and Security*. Ed. by Vijay Singh Rathore,
216 Marcel Worrying, Durgesh Kumar Mishra, Amit Joshi, and Shikha Maheshwari. Singapore: Springer
217 Singapore, pp. 493–503. ISBN: 978-981-13-2285-3.
- 218 Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra (2014). *Stochastic Backpropagation*
219 *and Approximate Inference in Deep Generative Models*. arXiv: 1401.4082 [stat.ML].
- 220 Harald Semmelrock, Tony Ross-Hellauer, Simone Kopeinik, Dieter Theiler, Armin Haberl, Stefan
221 Thalmann, and Dominik Kowald (2025). “Reproducibility in machine-learning-based research:
222 Overview, barriers, and drivers.” *AI Magazine*, p. 7000. DOI: <https://doi.org/10.1002/aaai.70002>.
- 223 Xingjian Shi, Zhoung Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo
224 (2015). “Convolutional LSTM Network: A Machine Learning Approach for Precipitation Now-
225 casting.” *Advances in Neural Information Processing Systems*, pp. 802–810.
- 226 Hongyang Su, Xiaolong Wang, Yang Qin, and Qingcai Chen (2024). “Attention based adaptive
227 spatial-temporal hypergraph convolutional networks for stock price trend prediction.” *Expert*
228 *Systems with Applications* 238, p. 121899. ISSN: 0957-4174. DOI: <https://doi.org/10.1016/j.eswa.2023.121899>. URL: <https://www.sciencedirect.com/science/article/pii/S0957417423024016>.
- 229 Ilya Sutskever, Oriol Vinyals, and Quoc V. Le (2014). “Sequence to Sequence Learning with Neural
230 Networks.” *Advances in Neural Information Processing Systems*, pp. 3104–3112.
- 231 Jia Wang, Tong Sun, Benyuan Liu, Yu Cao, and Hongwei Zhu (2019). “CLVSA: A Convolutional
232 LSTM Based Variational Sequence-to-Sequence Model with Attention for Predicting Trends of
233 Financial Markets.” *Proceedings of the Twenty-Eighth International Joint Conference on Artificial*
234 *Intelligence (IJCAI-19)*, pp. 3705–3711. DOI: 10.24963/ijcai.2019/514.