Teaching Economics to the Machines

Hui Chen, Yuhan Cheng, Yanchu Liu, Ke Tang

Present by Li Ziming

Motivation

- Is theory dead?
  - Simply applying AI without considering economic implications can produce powerful predictive ability.
  - Theory provides **motivations** to identify features and **restrictions** for forecasting.
- Economic models and ML models have comparative advantages.
  - Structural economic models can convey appealing economic insights but suffer from poor fit with data.
  - Non-structural ML models offer rich flexibility but have over-fitting issues.

Research Questions

- (1) How to combine economic models with machine learning?
- (2) Do transfer learning approach outperform?
- (3) Why is transfer learning better?

Contribution

- Contribute to literature of applying ML to finance and economics.
  - Prior literature: solve the pricing equations (Chen et al., 2022) and Bellman equations (Kong et al., 2020) implied by structural models.
  - Extend: bring economic restrictions from structural models into ML model.
- Contribute to literature on derivative pricing.
  - Prior literature: stochastic differential models outperform best ML (Jang and Lee, 2019); knowledge in finance modify performance of NN (Garcia and Gençay, 2000).
  - Extend: transfer learning approach better than deep learning and traditional stochastic differential models in terms of pricing power and hedging power.

Transfer learning approach

- Predict $y \in Y$ using a function of potential features $f(x)$ for $x \in X$.
  - Search for function $\hat{f}(\cdot)$ from given function family H (set of neural networks).
  - Minimizes the loss function $L(f(x), y)$ over a given training set S.
- Bayesian framework: use theoretical restrictions to form informative prior for function $f(\cdot)$ and fine-tune it with real data.
  - If theoretical restrictions between x and y valid, $\mathrm{Q}_{(y|x)}$ consistent with $P_{(y|x)}$.
  - Misspecification also useful: increase biases but restrictions reduce variance.

Transfer learning approach

- Source domain
  - Randomly generate training samples within X and calculate theoretical model output $g(X_i)$.
  - A sufficiently deep neural network can approximate arbitrary functions.
  - True model without any noise, train on source domain with numerous epochs and large learning rates.

A neural network with L layers: $F(L; \sigma_1, \sigma_2, \sigma_3, \ldots, \sigma_L; W_1, W_2, W_3, \ldots, W_L)(X_i)$

$$\widehat{W_1}, \widehat{W_2}, \widehat{W_3}, \ldots, \widehat{W_L} = argmin \lambda_0 L_0 + \sum_{j=1}^{\# X_i} \lambda_j L_j$$

Transfer learning approach

- Target domain
  - Fine-tune original form with the last few layers initialized according to real data, choose K and replace the layers K+1 to L.
  - Make the model not seriously deviate from economics theory and can flexibly modify within certain range.
  - Noisy in empirical data, train on target domain with low learning rate and epochs.

$$\widetilde{W_1}, \widetilde{W_2}, \widetilde{W_3}, \dots, \widetilde{W_L}$$
$$= argmin \sum_{i=1}^{N} |F(L; \sigma_1, \dots, \sigma_K, \widehat{\sigma_{K+1}}, \dots, \widehat{\sigma_L}; W_1, W_2, W_3, \dots, W_L)(X_i) - y_i| w_i$$

Final pricing model: $F(L; \sigma_1, \dots, \sigma_K, \widetilde{\sigma_{K+1}}, \dots, \widetilde{\sigma_L}; \widetilde{W_1}, \widetilde{W_2}, \widetilde{W_3}, \dots, \widetilde{W_L})(X_i)$

An application to option pricing

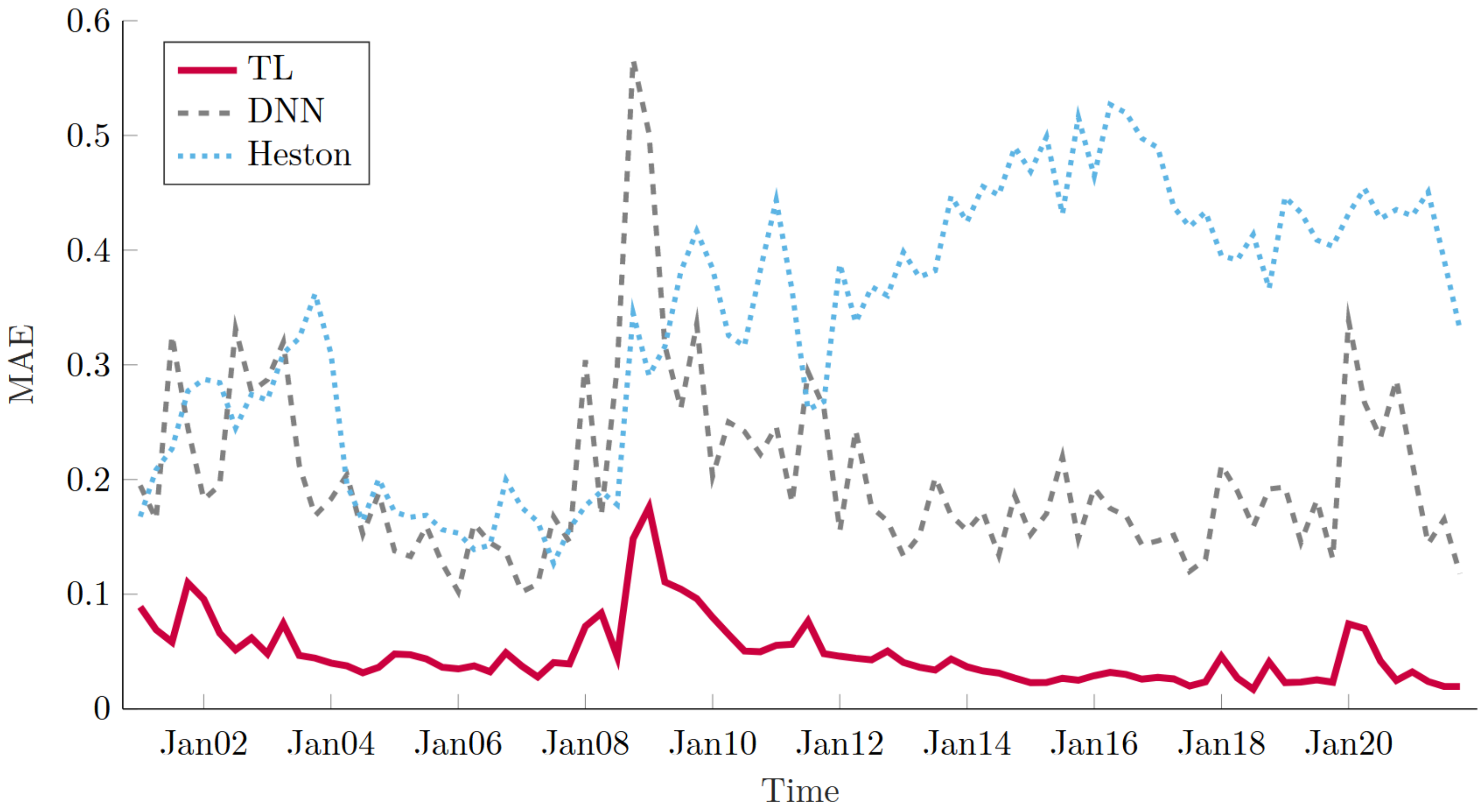
- Source domain
  - Input: random generated $X_i = (S_i, K_i, T_i, Vol_i, d_i, r_i, Z_i)$
  - Economic restriction: stochastic volatility model (SBAR、CEV)
  - Objective function: $\widehat{W_1}, \widehat{W_2}, \widehat{W_3}, \ldots, \widehat{W_L} = argmin \lambda_1 L_1 + \lambda_2 L_2 + \lambda_3 L_3$
  - $L_1 = \sum_{i=1}^{N} \left| \frac{1}{|\delta_1| + \epsilon_c} \left(F(L; \sigma_1, \sigma_2, \sigma_3, \ldots, \sigma_L; W_1, W_2, W_3, \ldots, W_L)(X_i) - g(X_i) \right) \right|$
  - Delta: $L_2 = \sum_{i=1}^{N} \left| \frac{\partial F(X_i)}{\partial S} - \frac{\partial g(X_i)}{\partial S} \right|$
  - Vega: $L_2 = \sum_{i=1}^{N} \left| \frac{\partial F(X_i)}{\partial vol} - \frac{\partial g(X_i)}{\partial vol} \right|$

- Target domain
  - Input: real world option data
  - Objective function: $\widetilde{W_1}, \widetilde{W_2}, \widetilde{W_3}, \dots, \widetilde{W_L} =$

$$argmin \sum_{i=1}^{N} |F(L; \sigma_1, \dots, \sigma_K, \widetilde{\sigma_{K+1}}, \dots, \widetilde{\sigma_L}; W_1, W_2, W_3, \dots, W_L)(X_i) - y_i| w_i$$

  - Hyper-parameters and NN structure tuned based on grid search and K-fold validation.
- Residual learning method
  - Avoid VGP to make NN deep enough.
  - $f(\cdot)$ learns to fit residual term $\sigma_i(w_i I_i + b_i) = f(w_i I_i + b_i) + I_i$
- Performance metric
  - pricing discrepancy: $\widetilde{\epsilon_{it}} = \left|\sigma(P_{it}; K_i, T_{it}, r_i, S_t, d_i) - \sigma\left(\widehat{P_{it}}; K_i, T_{it}, r_i, S_t, d_i\right)\right|$
  - MAE weight **in-the-money contracts** more than out-of-the-money.

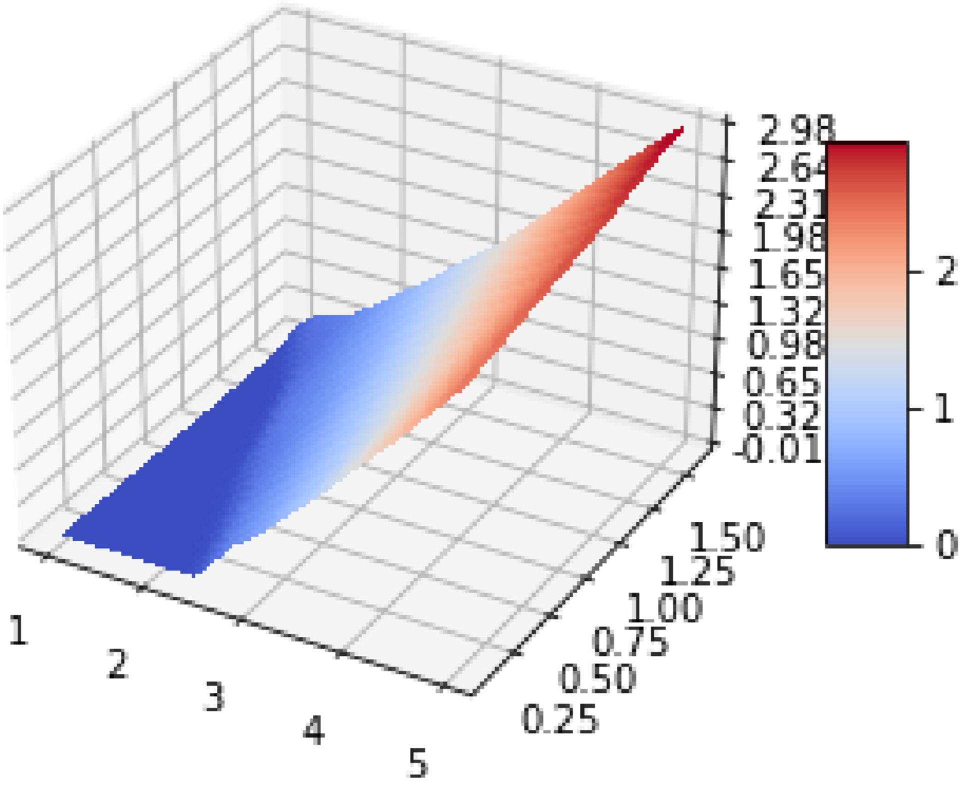
- Compare implied volatility MAE of option pricing models
  - Representative deep learning method: DNN
  - Well-established parametric pricing model: Heston



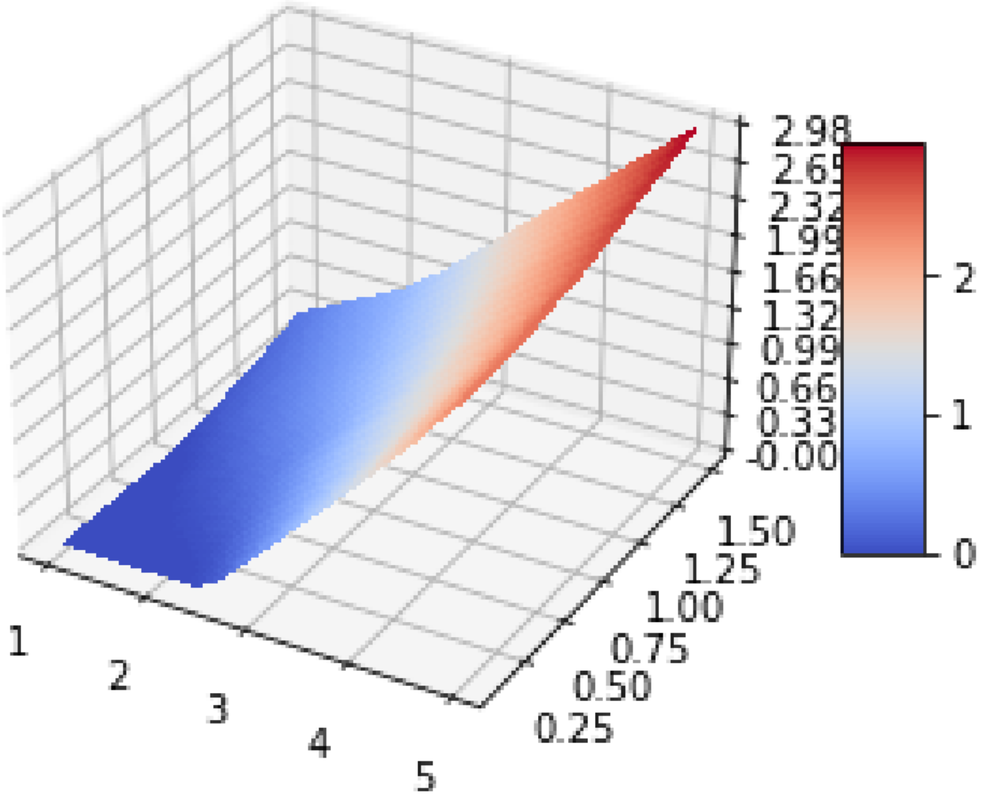
- Feature importance
  - Incremental rise in loss function on training dataset when specific feature omitted.

Feature	Mean	Median	Std	$Q_{25\%}$	$Q_{75\%}$
S	0.080802	0.065811	0.053689	0.046090	0.102313
K	0.254765	0.281755	0.125342	0.152587	0.338298
T	0.008463	0.008559	0.006065	0.005045	0.011809
r	0.163964	0.189252	0.067358	0.108908	0.221215
d	0.000165	-0.000035	0.001459	-0.000547	0.000336
IV1	0.015858	0.010775	0.015948	0.004021	0.022601
mom1	0.000232	0.000170	0.000491	-0.000098	0.000386
mom4	0.001008	0.000723	0.001664	-0.000079	0.001711
hv1	0.002691	0.000421	0.007501	-0.000077	0.001748
hv9	0.000717	0.000350	0.002214	-0.000284	0.001217
volume1	0.000003	0.000020	0.000195	-0.000064	0.000073
volume5	0.000080	0.000124	0.000267	-0.000005	0.000233
Put-Call Ratio	0.003640	0.001666	0.007027	0.000419	0.003318
Earnings-Price Ratio	-0.000080	0.000025	0.000776	-0.000248	0.000297
spmom1	0.000114	0.000064	0.000399	-0.000108	0.000275
spmom4	0.000489	0.000240	0.001012	-0.000001	0.000539

- Function smoothness
  - the surface associated with transfer learning demonstrates a enhanced smoothness compared to the more jagged surface produced by deep learning.



Pricing surfaces for deep learning model



Pricing surfaces for transfer learning model

- Attribution Analysis
  - Regress pricing errors for DL and TL on explanatory variables.
  - $\widetilde{\epsilon_{it}}^{DL} - \widetilde{\epsilon_{it}}^{TL} = x_{it-1}\beta + u_{it}$

	(1)	(2)	(3)	(4)	(5)	(6)
T	0.0222***	0.116***	0.123***			
	(112.86)	(191.94)	(169.05)			
0-7				0.0332***	0.0222***	0.0202***
				(121.22)	(80.22)	(71.96)
7-14				0.0159***	0.0101***	0.00972***
				(72.82)	(45.52)	(43.91)
90+				0.0224***	0.0416***	0.0367***
				(121.44)	(145.59)	(127.98)
marketIV	0.149***	0.345***	0.368***	0.154***	0.257***	0.243***
	(150.35)	(252.29)	(254.94)	(155.14)	(191.32)	(182.81)
BAspread	-0.00056***	-0.00057***	-0.00039***	-0.000154***	0.000182***	0.000178***
	(-19.73)	(-18.93)	(-14.19)	(-6.58)	(7.51)	(7.20)
distance	0.000843***	0.000435***	0.000366***	0.000746***	0.000525***	0.000523***
	(82.72)	(42.13)	(35.97)	(74.49)	(50.79)	(50.78)
vol20	-0.139***	-0.137***	-0.134***	-0.164***	-0.177***	-0.162***
	(-65.73)	(-65.12)	(-63.96)	(-78.01)	(-82.90)	(-75.25)
left_tail_vol	0.00271***	0.00317***	0.00317***	0.00264***	0.00297***	0.00295***
	(85.61)	(101.13)	(101.14)	(83.49)	(95.15)	(94.38)

OTM	0.00103***	0.132***	0.133***	-0.00439***	0.119***	0.118***
	(4.28)	(251.35)	(254.45)	(-17.78)	(221.35)	(219.36)
DOTM	0.0705***	0.99***	0.967***	0.057***	0.977***	0.926***
	(161.79)	(159.80)	(152.26)	(127.17)	(158.25)	(148.42)
ITM	-0.0462***	-0.0524***	-0.053***	-0.052***	-0.0666***	-0.0683***
	(-82.92)	(-64.82)	(-65.77)	(-93.73)	(-81.98)	(-84.56)
DITM	-0.0595***	-0.00969***	-0.0135***	-0.0644***	-0.0145***	-0.0276***
	(-72.30)	(-8.88)	(-12.22)	(-78.18)	(-13.19)	(-24.83)
IV_ITM		-0.00484	-0.00918***		0.0359***	0.0771***
		(-1.82)	(-3.34)		(13.20)	(28.16)
IV_DITM		-0.171***	-0.147***		-0.173***	-0.0543***
		(-52.70)	(-40.89)		(-52.22)	(-15.12)
IV_DOTM		-1.85***	-1.8***		-1.81***	-1.67***
		(-156.45)	(-147.36)		(-153.97)	(-139.54)
IV_OTM		-0.378***	-0.394***		-0.345***	-0.316***
		(-233.35)	(-233.93)		(-208.03)	(-187.47)
IV_T		-0.394***	-0.404***		-0.135***	-0.0765***
		(-176.27)	(-130.53)		(-115.47)	(-55.51)
IV_T_OTM			0.0484***			-0.0713***
			(26.38)			(-50.50)
IV_T_ITM			0.00943			-0.166***
			(1.21)			(-21.78)
IV_T_DOTM			-0.0496***			-0.2***
			(-13.46)			(-59.77)
IV_T_DITM			-0.0593***			-0.291***
			(-8.94)			(-46.24)
const	0.0894***	0.035***	0.035***	0.0871***	0.0628***	0.0648***
	(321.07)	(103.06)	(100.15)	(311.28)	(197.30)	(204.45)
R^2	0.02265	0.04427	0.04454	0.02400	0.03613	0.03807

Conclusion

- Introduce transfer learning that incorporate economic model into ML.
  - Transfer learning approach yields **lower pricing and hedging errors** compared to stochastic volatility models and direct use of deep learning.
- Transfer learning overcome inherent drawbacks of data-driven methods.
  - Suitable for **high-volatility** market environments characterized by **small-sample**.
  - More robust to various **shocks**.
- Economic models created by human help AI to overcome inherent flaws.
  - Even with considerable effort in feature engineering, its improvement is hard to match the help of theoretical models in the source domain for neural networks.

New ideas

- Apply transfer learning approach to prediction tasks in other contexts.
  - Macroeconomic forecast (total output, inflation, labor supply, consumption)
  - Other pricing model (stocks, bonds, futures)