PropCast — Deep Inception-Convolutional LSTM Networks for Forecasting Spatio-Temporal Transitions in Property Index Heatmaps

Athanasse Zafirov
UCLA Anderson School of Business

Cory Ye
UCLA Henry Samueli School of Engineering

Abstract

Real estate property values have changed in complex and chaotic ways over time, as local and global socio-economic developments affect the prices of consumer and industrial properties in various regions around world. Forces of urbanization, gentrification, and ghettoization implicitly guide the dynamics of real estate prices depending on a variety of extraneous property characteristics, i.e. position, elevation, property type / volume, origin date, and contextual setting. To learn secular trends in the real estate market that parallel developments in sociology and predict the locations and prices of future real estate transactions, we design an inception-convolutional LSTM (ICLSTM) to forecast spatio-temporal patterns in property index time-series. To gauge the performance of the ICLSTM, we alternatively design a spatio-temporal convolutional neural net (DSTCNN) to apply imitation reinforcement learning on the time-series data in comparison. In particular, the two models predict the value and location of future property transactions relatively accurately depending on the topography of the prior property value heatmaps in the time-series.

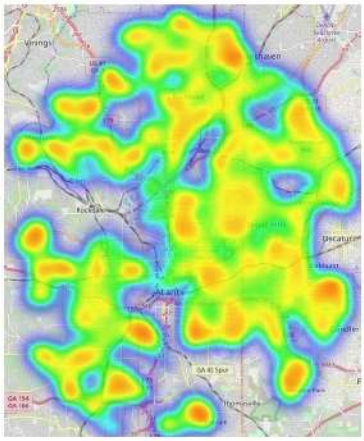

Figure 1. Property Value Heatmap of Atlanta, Georgia

1. Overview of PropCast

PropCast refers to the problem of forecasting future property values for unknown regions via training on time-series of past property feature heatmaps of known regions, i.e. data with input shape (N, T, S, S, K) and output shape (S, S), where N is the number of sample heatmap time-series, T is the number of time-steps of the time-series, S is the spatial dimension of the window of the heatmap, and K is the number of property features like price, type, etc. Consequently, we refer to both the ICLSTM and DSTCNN as algorithms to solve PropCast. Both models were built over Keras API for TensorFlow Core on Google Colab Pro. [5]

1.1. Data Retrieval, Analysis, and Processing

Data utilized in training the neural nets are sourced from ATTOM Data [4], which includes real estate transaction information separated by region and indexed in time from 2004 to 2019. Beyond spatio-temporal data that impacts the geographic price distribution of real estate, macroeconomic, demographic, and labor statistics tend to be versatile explanatory variables in real estate research literature.

Exploring county-level contextual data that augment the property transaction data, we identified informative, uncorrelated variables from the US Bureau of Labor Statistics and US Census (**Fig. 2**) via FuzzyForest [2]. Optimized toward HPI, Fuzzy Forest identified 21 significant features that explain trends in real estate prices, including data pertaining to ethnic demographics, labor statistics, and domestic migration. Concatenating property transaction data and contextual feature data produced the training data for Prop-Cast.

To process the transaction and contextual data, we extracted the desired features by transaction location and separated the input and output data by time. Subsequently, the transaction data was spatio-temporally integrated to a geographic spatial-coordinate grid with a temporal sampling window of a year, necessary to reduce the spatio-temporal sparsity of the data. Prior to training, inflated features like prices were log-normalized and/or shifted to smooth the data and prevent divergent gradients to accelerate training

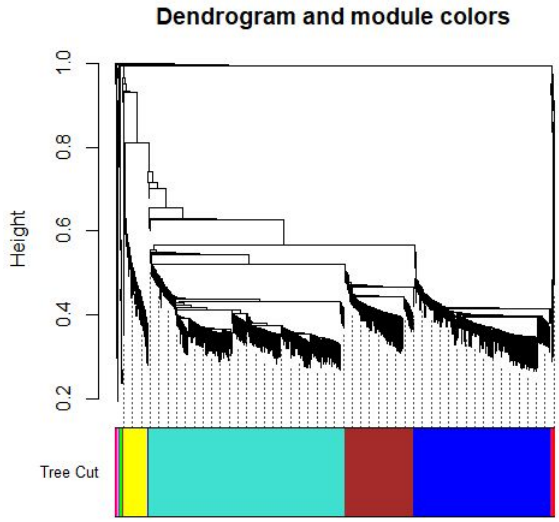

Figure 2. Fuzzy Forest for Contextual Features

and improve the quality of heatmap predictions.

1.2. Supporting Inception Modules

Inception modules were utilized in the models to condense the features of the data via exploiting proximity with filter kernels of various dimension prior to spatio-temporal convolution (Conv3D) or sequential learning (CLSTM). Intuitively, the features of close properties interact with each other in non-linear and time-variant ways.

Two different inception modules (displayed in **Fig. 4**) – the elementary inception module (EICNN) and the deep inception module (DICNN) – were designed to be respectively compatible with the ICLSTM and the DSTCNN. On the one hand, the elementary inception module is inserted between layers of convolutional LSTM to replicate a generalized version of the inception LSTM in [3]. On the other hand, the deep inception module follows the design of GoogLeNet (Inception v4, [1]), and is applied to summarize comprehensive time-invariant spatial features prior to spatio-temporal convolution for imitation learning.

1.3. ICLSTM

Inception-Convolutional LSTM Neural Nets (ICLSTM) were designed to learn spatially-comprehensive time-variant or time-sequence features via injecting inception modules to the input-output pipeline of the (fully-connected) LSTM. Such adjustments generalize the model for the classical convolutional LSTM [6] that is restricted to a fixed-size convolutional filter in Keras. Another perspective of an inception-convolutional LSTM is a time-distributed pull-back of the inception-convolutional filters embedded within the memory cells of the LSTM, which derives the underlying structure of the ICLSTM for PropCast.

In particular, the architecture of the ICLSTM for PropCast (displayed on the left in **Fig. 5**) is precisely a series of batch-normalized inception-convolutional LSTM layers composed of a time-distributed elementary inception module (EICNN) and a classical convolutional LSTM (ConvLSTM2D). To merge the features of the LSTM and reduce the magnitude and back-propagated gradient of negative property values without causing neuron death in the output heatmap, the Conv2D layer and non-linear LeakyReLU activation ($\alpha = 0.1$) terminates the ICLSTM.

1.4. DSTCNN

Deep Spatio-Temporal Convolutional Neural Nets (DSTCNN) utilize spatio-temporal convolution to learn spatio-temporal features in the time-series data. Intuitively, it formulates time as a dimension of space, and applies filters that relates data proximal in time, i.e. the feature heatmap at time t depends on the heatmaps at times $t-1$ and $t+1$ in analogy to a hypothetically causal or anti-causal Markov Chain.

Analogously, a time-distributed deep inception module (DICNN) is applied to extract features from the heatmaps of the time-series, and multiple spatio-temporal convolutional layers (Conv3D) with collective temporal receptive field sufficiently greater than the length of the time-series T learn time-variant relationships and trends from the feature time-series (displayed on the right in **Fig. 5**). However, Conv3D layers are difficult to train, which necessitates feature compression via time-distributed Conv2D, while Conv2D followed by LeakyReLU terminates the DSTCNN.

2. ICLSTM and DSTCNN Performance

Overall, the ICLSTM And DSTCNN predict relatively accurate forecasts that share identifiable features with the actual property value heatmap in the future. However, limited evidence exists to explain whether the results are robust, due to the overwhelming presence of noise in the prediction. Crucially, the weakest link in the model is the training data, as $N = 42$ is insufficient data to almost surely approximate the topology of the high-dimensional latent space of heatmaps via the Law of Large Numbers.

Analyzing the topography of the predicted heatmaps (**Fig. 7**), we observe that the ICLSTM generates forecasts with more variance and less smoothness than that of the DSTCNN. Such distinctions are perhaps a consequence of the model architecture – the ICLSTM underfits and generalizes the spatio-temporal dynamics of the training data, while the DSTCNN over-fits and imitates the spatial-temporal dynamics of the training data. In particular, inflated loss does not imply worse performance, as the DSTCNN fails to capture the density and variance of the actual transaction data, at which the ICLSTM is proficient. However, both neural nets are volatile on sparse test

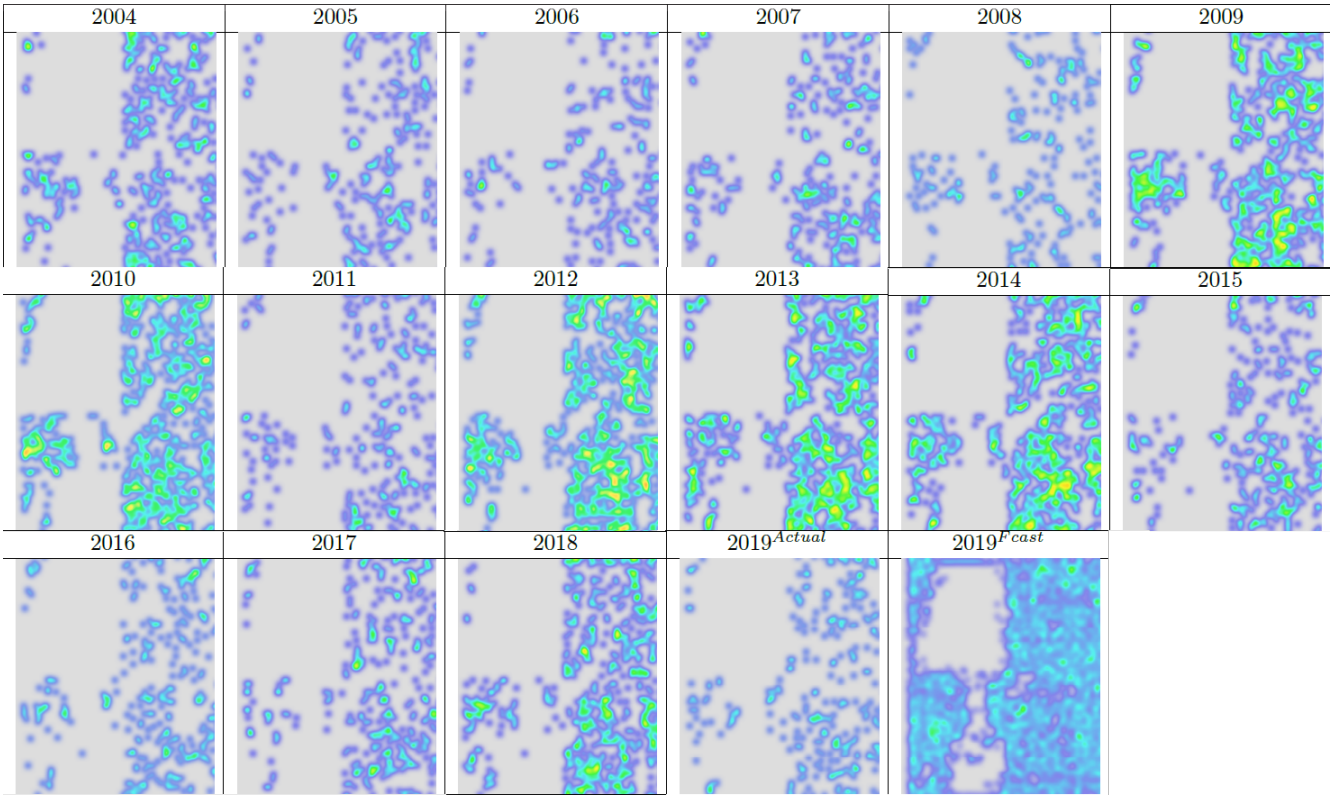

Figure 3. Property Value Heatmap Time-Series for Los Angeles County (2004-2019)

data and fail to regress and predict outliers in the test data, explaining the drastic difference in scale between the predicted and actual heatmaps. Visualizations of the heatmap time-series with both predicted and actual heatmaps for Los Angeles County is displayed in **Fig. 3**. Training loss metrics for the ICLSTM and DSTCNN are graphed in **Fig. 6**.

3. Discussion

To optimize the model, Nesterov-inertial SGD with slightly lower learning rate and momentum was optimal. Moreover, mean-squared-error loss was preferable over log-cosh loss due to quadratic penalization of sparse data and spatial regression, as for video-prediction in [3].

Initially, the uncompressed training data expended tons of memory (approximately 64 GB) to train the spatio-temporal convolutional net and inception-convolutional LSTM. To reduce expenditure of RAM, we reduced spatial resolution, reduced numerical precision (to single-precision or 32 bit), and optimized memory efficiency for the time-distributed EICNN and DICNN.

In general, LSTMs are complicated to tune and difficult to train, and inception-convolutional LSTMs exacerbate the drawbacks. Training loss descends rather monotonically, but validation loss curves are sometimes unstable and chaotic. In particular, norm-limited gradients mysteriously accelerate training for deep convolutional LSTMs.

References

[1] V. V. A. A. Christian Szegedy, Sergey Ioffe. Inception-v4, inception-resnet and the impact of residual connections on learning, 2016. Source on arXiv: https://arxiv.org/abs/1602.07261.

[2] G. L. C. R. Daniel Conn, Tuck Ngun. Fuzzy forests: Extending random forests for correlated, high-dimensional data. *UCLA Bio-Statistics Research Reports*, 2015. Source: https://escholarship.org/uc/item/55h4h0w7.

[3] M. H. G. R. Matin Hosseini, A.S. Maida. Inception-inspired lstm for next-frame video prediction, 2019. Source on arXiv: https://arxiv.org/abs/1909.05622.

[4] A. D. Solutions. Attom data property index database, 2020. Database API Documentation: https://api.developer.attomdata.com/docs.

[5] TensorFlow. Keras api: The python deep learning library, 2020. Keras Documentation: https://keras.io/.

[6] H. W. D.-Y. Y. W.-K. W. W.-c. W. Xingjian Shi, Zhourong Chen. Convolutional lstm network: A machine learning approach for precipitation nowcasting, 2015. Source on arXiv: https://arxiv.org/abs/1506.04214v1.

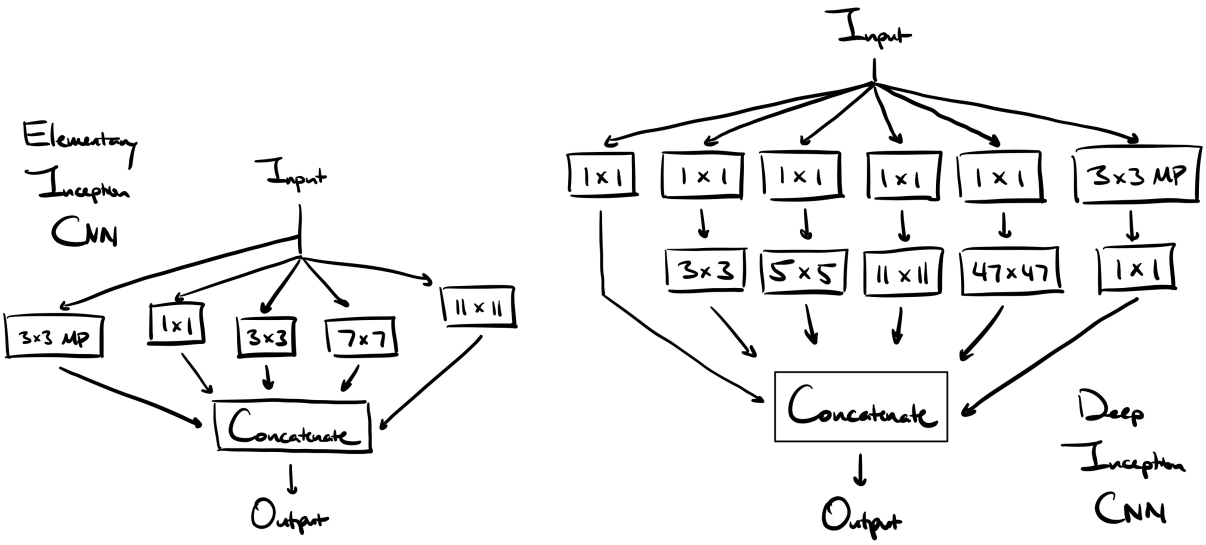

Figure 4. Elementary and Deep / Wide Inception-Convolutional Neural Net Architectures

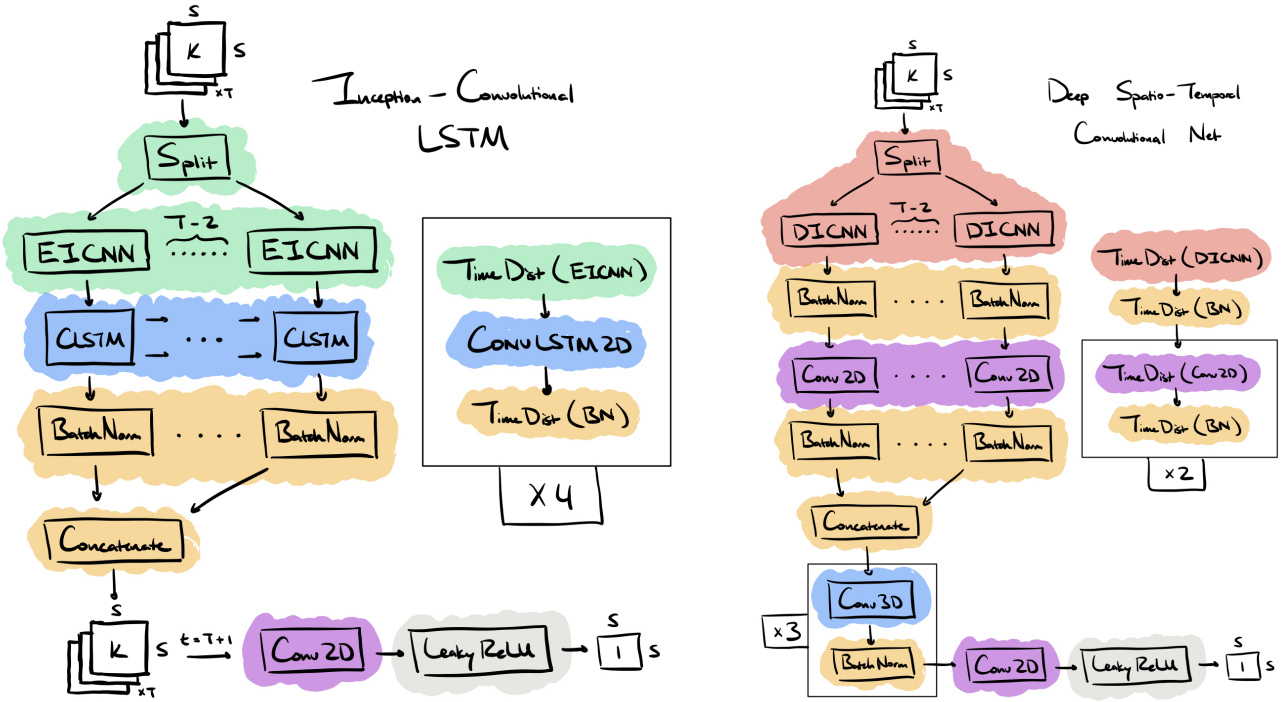

Figure 5. Inception-Convolutional Long Short-Term Memory and Deep Spatio-Temporal Convolutional Neural Net Architecture

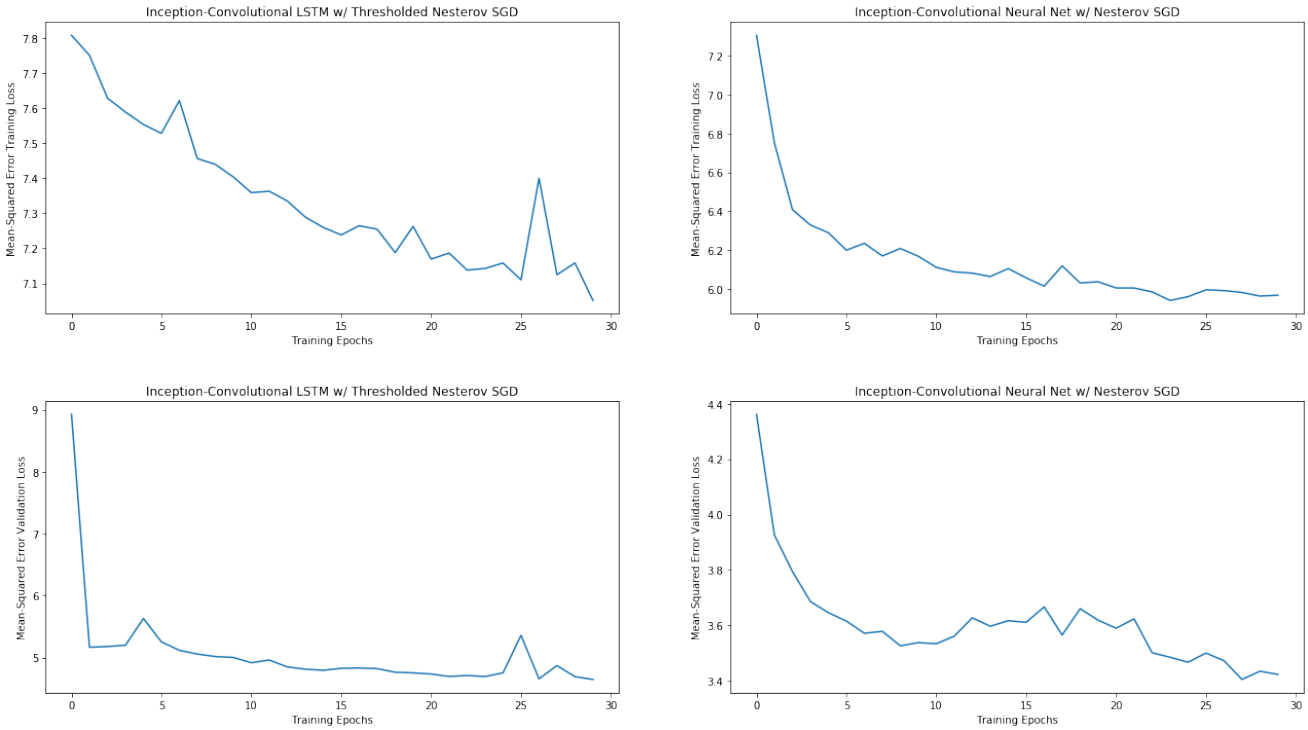

Figure 6. Sample Training and Validation Loss for ICLSTM and DSTCNN for Los Angeles County

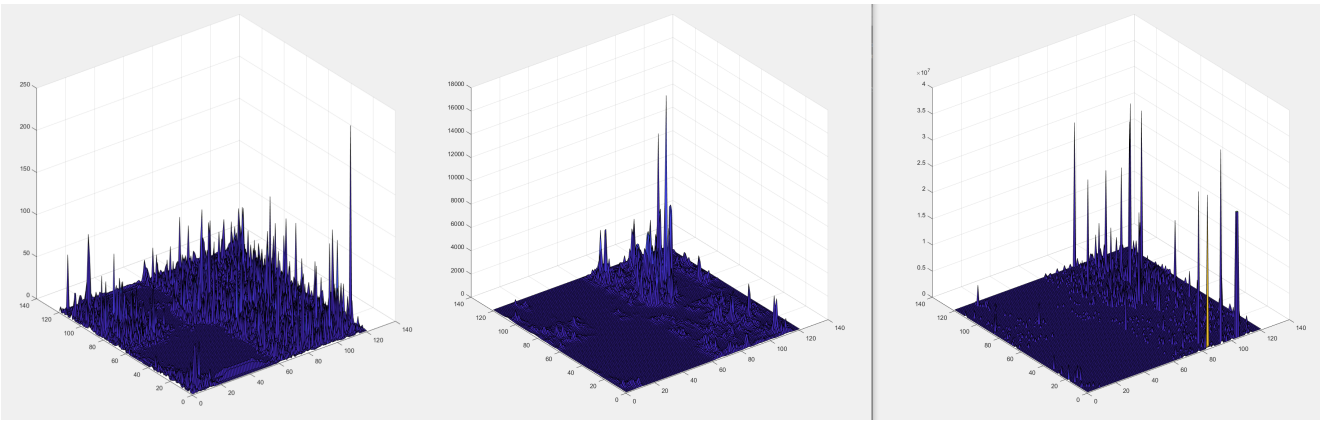

Figure 7. Predicted Heatmap Topography for ICLSTM (left) and DSTCNN (center) with Actual (right) for Los Angeles County