Image-based classification of Pittsburgh-area buildings

Michael Shteyn

Background and significance

Estimating the value of a real estate parcel is an important challenge for prospective home buyers, municipal authorities, creditors, and real estate professionals. Though geographic features are known to have a strong influence on the real estate market, many features that determine building value are latent and challenging to catalog. The visual characteristics and visual context of a building comprise a key set of features that may have an impact on real estate value, but which are difficult to quantify using traditional survey methods. Visual characteristics and context include the exterior condition of a building, the vegetation in the lawn, the type of automobile parked in the driveway [1], the distance a home is from the street, the tidiness of its balcony, and many other factors.

To better understand the impact of visual characteristics on a building's market value, I conducted a series of deep learning experiments using neural network classifiers to predict building values based on Google Street View images of Pittsburgh-area buildings. The database consisted of 355 407 Google Street View images along with 89 corresponding semantic feature labels curated by Western Pennsylvania Regional Data Center (WPRDC) [2]. I found that both convolutional and transformer-based neural networks were effective at learning a Pittsburgh-area home's fair market value significantly above chance level, with an accuracy approaching 60% across all tested models. Further, I found that augmenting the predictive task into multitask classification space across correlated features enabled a convolutional neural network model to achieve a maximum test classification accuracy of 62.4% in predicting fair market real estate value.

Methods

A. EDA and data preparation

The original image database was scraped from the Allegheny County property assessments website [3] based on an index of 455 649 parcel identification codes provided by WPRDC. The WPRDC parcel identification codes were indexed in comma separated form along with 89 semantic features relating to each building record. The features include a building's assessed market value, the number of interior rooms, its year of construction, its total finished living area, municipality, among many others. The original dimensions of most images in the database were 352x228 pixels. The images were preprocessed to ensure universal dimensionality, and a 20 pixel area containing a black bar and parcel-identifying text at the bottom of each image was cropped. A number of images scraped were of particularly low resolution due to the lighting conditions on the day the image was captured by Google Street View. 100 242 images smaller than 10kb in size were removed, resulting in a final database of 355 407 images (***Fig. 1***).

Figure 1. Google Street View images of Pittsburgh-area buildings indexed according to fair market value, year of construction, and interior living area. Parentheticals indicates quintile class in the design that each feature is separated into five quintiles (Q1-Q5).

Exploratory data analysis involved determining the relationship between relevant semantic feature labels, data cleaning and outlier detection. Using WPRDC's database, I selected three features of interest that were likely to be correlated to the assessed market value - the number of interior rooms, year of construction, and total interior living area in square footage - for a total of four building features of interest. Extraneous features were excluded on the basis of sparsity, large quantities of null entries, or lack of obvious relationship with building appearance or assessed market value. Significant outliers in each feature category were removed, which included buildings that were anomalously over-assessed, unusually old or extremely large. To address this, data points from each feature category with a value less than or greater than three standard deviations from the mean were removed.

The data was discretized into equally distributed quintiles for each of the four features of interest to enable building classification. Discretizaiton was important for reshaping individual feature data into uniform distributions in order to facilitate strong learning. The discretization procedure also played an important role in bringing features into a standardized unit space to enable experiments which backpropogate multitask loss. An example of the living area quintiles distribution with and without outliers is shown on the ***Figure 2*** below.

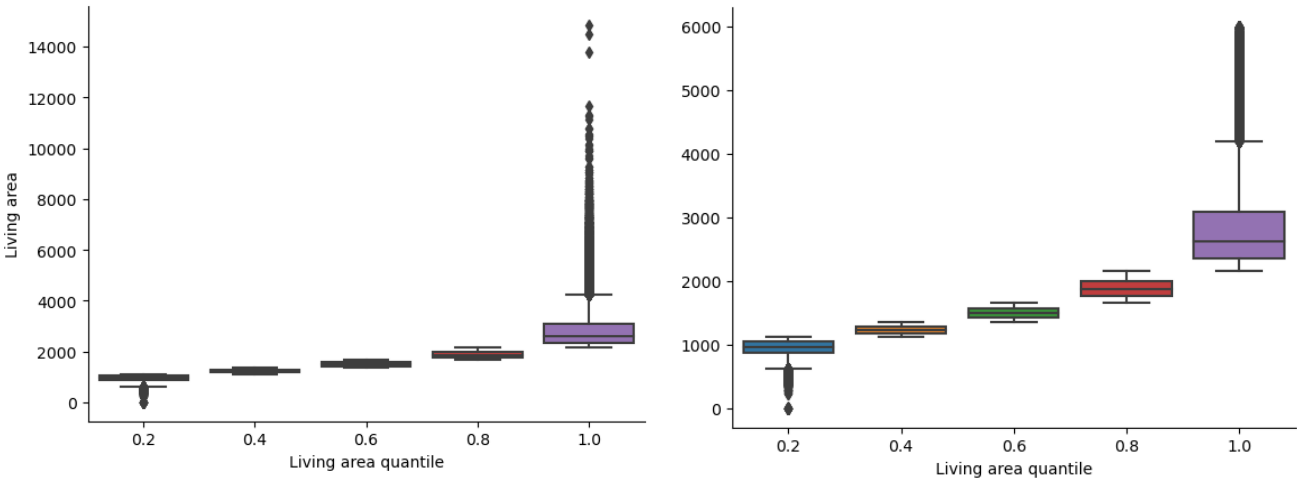

Figure 2. The finished living area quintiles distribution with (A) and without (B) outliers.

The year of construction and interior living space area distributions were linearly correlated with assessed market value suggesting newer and bigger houses are usually more expensive (***Figure 3***). Though th linear relationship between these features was clear, a significant number of exceptions to this rule enabled us to consider whether reframing the training procedure as a multitask classification problem would enhance predictive accuracy.

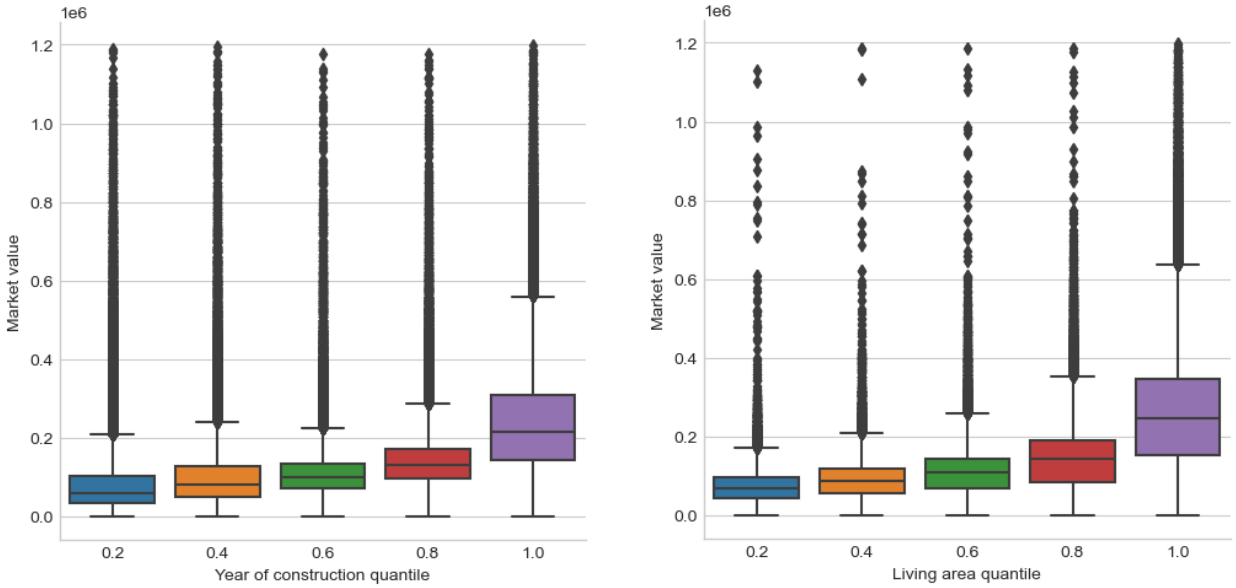

Figure 3. The finished living area and the year of construction quintiles linearly correlates with the market value.

B. Model architectures

Transfer learning was applied to several state-of-the-art artificial neural networks belonging to two distinct architectural families: *(A)* convolutional neural networks (CNNs) (GoogLeNet [5], ResNet50 [6]), and *(B)* transformer-based architectures (Vit-L [7], Vit-B [7] and Swin-transformer [8]).

All of the models were trained using the PyTorch library in the Conda virtual environment utilizing GPU resources on the high performance Bridges-2 cluster located at the Pittsburgh Supercomputing Center.

Results

A. Model selection

The model selection procedure consisted of two phases. The first phase was exploratory and included testing a range of CNNs and transformers with standardized hyperparameters - the *Adam* optimizer [3] with learning rate 0.001, batch size 64, and three fully connected layers applied to a single task: assessed market value (discretized into 5 quintiles). Each exploratory experiment consisted of a single 10 hour training session on one GPU. The second phase involved choosing the two highest performing models from each architecture class based on the exploratory results, after which hyperparameters (optimizer class, learning rate, number of fully

connected layers) were tuned to enhance the classification accuracy. The discretization method (number of quintiles for each feature category) and expanding the classification problem into multi-task learning were also explored for the top performing models.

I found that each image classification network architecture tested in the exploratory round achieved significantly better performance than chance level (five quintiles: 20% chance). In the interest of exploring whether hyperparameter tuning may have a stronger impact on the classification performance of one network architecture over the other, I selected one representative CNN and one transformer model for further testing.

The Swin-transformer (test accuracy: 58.6%) and ResNet50 (test accuracy: 59.6%) models were selected for deeper examination based on achieving the maximum classification accuracies among models tested (***Table 1***). The second phase of experiments consisted of a single 16 hour transfer-learning period on one GPU. Each model was applied using as a starting point weights that were originally pre-trained to achieve high classification accuracy on the ImageNet database and used the sum of cross-entropy as the loss function.

GoogLeNet	**ResNet50**	**ViT-B**	**Vit-L**	**Swin**
59.0%	**59.6%**	58.3%	58.1 %	**58.6%**

Table 1. A comparison of maximum held-out test accuracy among a variety of image classification models tested in the exploratory phase. ResNet50 and Swin were selected as the exemplar models of their class.

B. Hyperparameter tuning and second order tests

I. Optimizer and learning rate

The first hyperparameter of interest was the optimization method. Although *Adam* is the state of the art optimizer for many deep learning applications, I were interested in testing the *Lion* optimizer, recently proposed to train faster and achieve higher classification accuracy by Google researchers [4]. The *Lion* optimizer achieved strong performance and was faster at the suggested learning rate (0.0001) than the Adam optimizer (learning rate 0.001) when applied to the Swin transformer. When applied to ResNet50, the *Lion* optimizer was outcompeted by *Adam* at the standard learning rate (0.001). This is consistent with published data suggesting *Lion* best augments the performance of transformer models [4].

Because of the large scale and training time of the dataset, tuning the learning rate hyperparameter was limited to just a few variations. ResNet50 performed best with the Adam optimizer, so I focused on tuning the Adam learning rate, testing three variations: 0.002, 0.001, 0.0005. I also tested three Lion learning rates for the Swin transformer: 0.0001, 0.0002, 0.00005. The smallest tested learning rate (*Lion*: 0.00005, *Adam*: 0.0005) trained best for both models.

II. Multi-task learning

I compared both the Swin-transformer (*Lion*, learning rate = 0.00005) and ResNet50 (*Adam*, learning rate = 0.0005) models on their performance in predicting the fair market value

in a single-task as well as multi-task learning setting. The multi-task test included an experiment in which, in addition to fair market value, the network predicted two additional features ("year of construction", "finished living area") with five levels each, as well an experiment in which it predicted three additional features ("year of construction", "finished living area", "number of interior rooms") with seven levels each.

I found that both the ResNet50 and the Swin-transformer models achieved a higher accuracy and a lower loss when training to predict three features separated into five levels each as opposed to four features separated into seven levels each (***Fig. 4***). The ResNet transformer achieved a maximum classification accuracy of 63.2% on the three-feature task, while the ResNet50 network achieved a maximum classification accuracy of 60.7%, compared to 51.2% and 49.7% on the four feature tasks respectively.

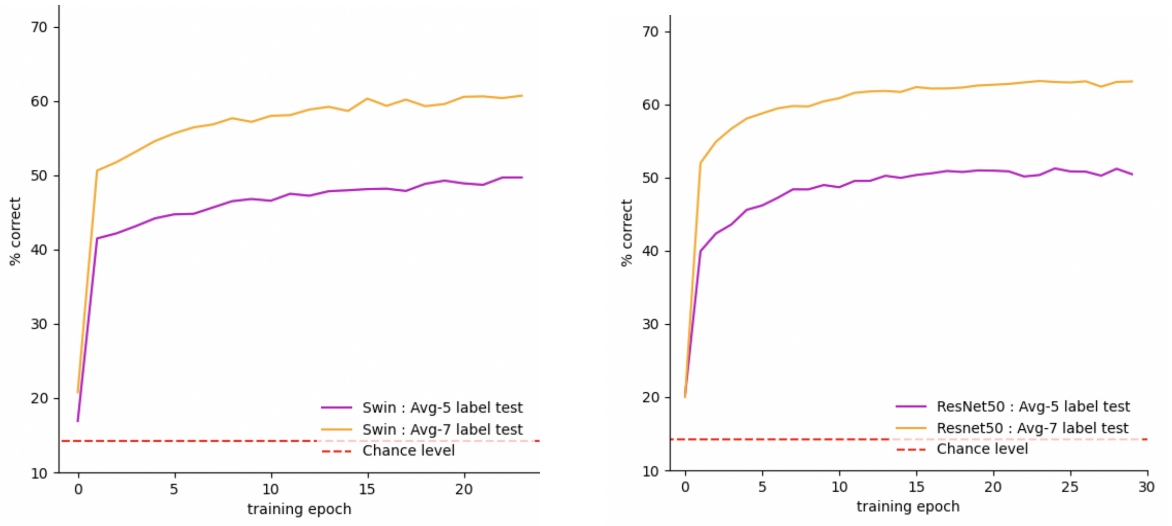

Figure 4. Comparison between multi-task learning on three feature categories with five levels vs. four feature categories with seven levels each for the Swin transformer (left) and ResNet50 (right). Plotted is the cross-entropy loss computed from the sum loss across all tasks.

Interestingly, both the Swin-transformer and ResNet50 models achieved a lower loss and a higher test accuracy in predicting fair market value in multi-task transfer learning than in single-task transfer learning (***Fig. 5***). In the case of both model architectures, the multi-task networks were initially slower to train and were outperformed by their respective single-task counterparts during the first few training epochs. Ultimately, both multi-task networks achieved stronger performance than their respective single task variants beginning after approximately five (***Fig. 5*****: top,** ResNet50) and ten (***Fig. 5*****: bottom,** Swin) epochs of training. The maximum test accuracy for the multi-task trained ResNet50 model was 62.4%, compared to 59.9% for the single-task trained model. Accordingly, the multi-task trained Swin-transformer model outperformed the single task model 60.6% to 58.9% when trained for the same 16 hour duration.

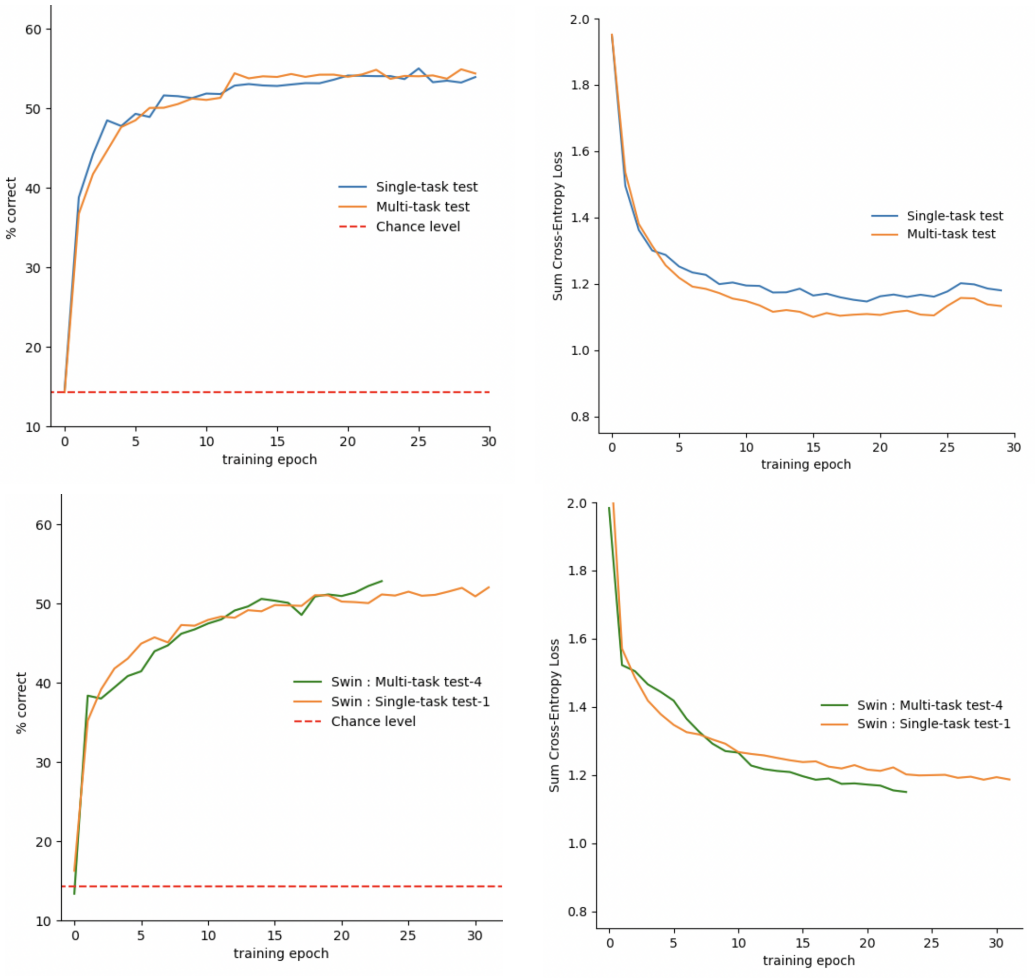

Figure 4. Comparison between single-task and multi-task learning. Minimum multi-task test loss is lower and maximum test accuracy is higher in predicting fair market value across seven quintile levels for both ResNet50 (Top) and the Swin-transformer

Error analysis

As a result of the high dimensionality of the image database and the summing procedure used to compute cross-entropy loss across multiple tasks, I wondered whether the multi-task model was developing a strategic bias toward automatically learning the representation of some data labels while strategically neglecting to learn a representation of other data labels to minimize loss during training. To this end, I analyzed the ResNet50 model's error pattern as its weights evolved during transfer-learning. I found that, on average, errors were reduced across all categories, providing little evidence of a training bias (***Fig. 6***). One exception to this general trend was for the estimate of finished living area. Here, the network increasingly misclassified homes within the third quantile of this feature as training increased. More investigation is necessary to better understand this behavior.

Over training, the network achieved the lowest classification errors for data in the tails of the distribution for each feature. This is expected behavior given the tail ends contain the greatest variance and therefore the least similarity in feature space to adjacent class categories. However, over the course of training, the distribution of the network's errors became substantially more

uniform suggesting the network indeed learned a sophisticated representation of the building image dataset.

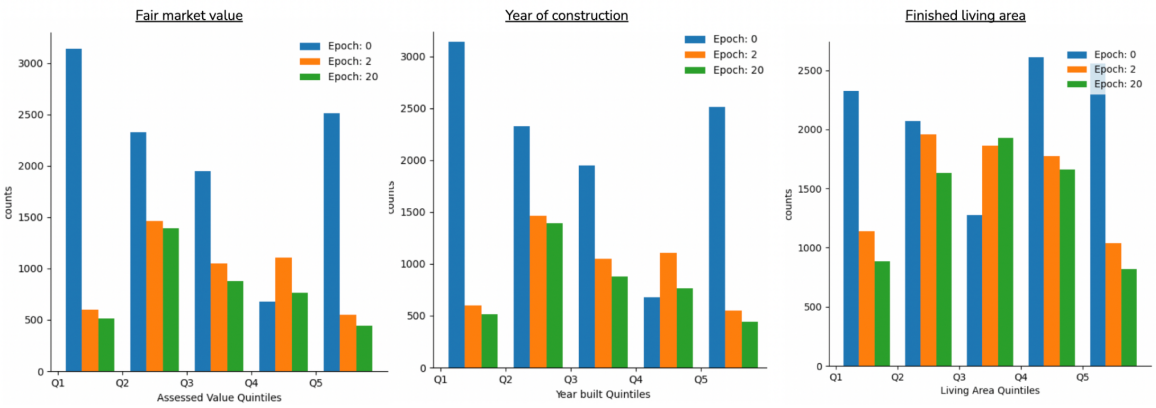

Figure 6. Incorrectly classified labels for each feature across quintile classes over the course of transfer learning. Generally, errors decreased uniformly across classes for each feature.

Discussion and future directions

I found that both convolutional and transformer-based neural network architectures were effective at classifying Pittsburgh-area buildings based on Google Street View image data alone. With training time held equal across all experiments, the most effective classification model was ResNet50, achieving a **62.4%** held-out test accuracy in predicting the fair market value of a real estate parcel across five quintiles. ResNet50 was most effective in predicting fair market value when trained on a multi-task classification that also included predicting the year of construction and finished living area of a building, generating a multi-task error that was back-propagated to enhance training on the fair market value prediction task. There was some suggestion that, given more training time, the Swin-transformer may achieve a higher test accuracy than the ResNet50 model. This suggestion comes from the observation that test accuracy was still monotonically increasing and test loss monotonically decreasing at the time Swim-transformer training was terminated, which was not the case for ResNet50. Because computing resources were limited, I were unable to continue training the Swin-transformer until it reached maximum classification accuracy. This would be a subject for further examination.

I also note that although errors decreased across all features and across nearly all classes as training increased, there was at least one notable exception to this trend. In the estimate of finished living area, ResNet50 increasingly misclassified homes in the third quintile. This suggests the network may have been developing a strategy to classify all other labels with high accuracy, thereby minimizing its classification error for other classes, while maximizing its classification error for third quintile category members. While this may be a successful strategy in minimizing total loss, it is a suboptimal strategy for learning the relationship between building images and finished living area. More study will be necessary to better understand the source of this error trend. Taken together, I demonstrate that contemporary deep learning tools can enable estimating building value from image data alone.

References.

(1) Gebru, Timnit, et al. "Using deep learning and Google Street View to estimate the demographic makeup of neighborhoods across the United States." Proceedings of the National Academy of Sciences 114.50 (2017): 13108-13113.
(2) Allegheny County Office of Property Assessments, Department of Administrative Services . *Property assessment parcel data (as of July, 14, 2015)* [Data set]. Licensed under CC BY. Retrieved from the Western Pennsylvania Regional Data Center on August 12, 2015 http://data.wprdc.org/dataset/property-assessment
(3) Kingma, Diederik P., and Jimmy Ba. "Adam: A method for stochastic optimization." arXiv preprint arXiv:1412.6980 (2014).
(4) Mirjalili, Seyedali. "The ant lion optimizer." Advances in engineering software 83 (2015): 80-98.
(5) Szegedy, Christian, et al. "Going deeper with convolutions." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015
(6) He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.
(7) Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An
image is worth 16x16 words: Transformers for image recognition at scale. In: International Conference on Learning Representations (2021)
(8) Liu, Ze, et al. "Swin transformer: Hierarchical vision transformer using shifted windows." Proceedings of the IEEE/CVF international conference on computer vision. 2021