Real Estate Price Estimation

Mohamed S. Ghoneim
Department of Computer Science and Engineering
The American University in Cairo
Cairo, Egypt
m_ghoneim@aucegypt.edu

Eman H. Ahmed
Department of Computer Science and Engineering
The American University in Cairo
Cairo, Egypt
eman.hamed@aucegypt.edu

Ahmed Mohsen
Department of Computer Science and Engineering
The American University in Cairo
Cairo, Egypt
ahmedmohsen@aucegypt.edu

1. Abstract

The main goal of this paper is to describe a proposed system that estimates real estate prices according to images of the real estate and provided textual data such as the location and number of rooms. The system is partially implemented. It is still to be completed and tested. The system is divided into a number of modules, and the theory behind each of the modules is explained in the paper. Mainly, the flow of the system is as follows: Input images of a house are labeled according to whether they are indoors or outdoors, textual data are fed to a classifier to give a partial estimate for the real estate, and the estimated price is then further estimated by analyzing certain features on the indoor and outdoor images of the house. The labelling of the images as indoor or outdoor dictates what features should be looked for in each image.

Keywords: Real Estate, Price, Estimation, Machine Learning, Regression, SVR, Computer Vision.

2. Introduction

With the increasing number of sales in the real estate market, it has become very tedious work to estimates prices of properties based on information such as their geographical locations, local landmarks, and other price-changing factors. Companies that works in the field of real estate sales waste hours estimating properties' prices. This is becoming ever challenging as such companies attempt to sell properties in remote states because in that case research would have to be done regarding these areas in order to have a proper estimation to the value of each property. It would be of a great economic benefit to automate this process of price estimation to save resources. This paper discusses a proposed

system that would take as an input ground-truth information about properties' location, images, textual data such as number of rooms, and prices. Based on numerous such examples, the system would estimate the price of a home given its location, images, and textual data. It is feasible for a real estate company to provide the system with such ground-truth information especially if the company has a long history of sales.

3. Literature Review

3.1 Method For Combining House Price Forecasts (2003)

Some previous work to automate the real estate price evaluation process was done by Jost et al. in the U.S. Pat. No. 5,361,201. Et al proposed a neural network-based system for estimating the price of the real estate automatically. Jost et al discussed the previous efforts in this field. It discussed the traditional statistical techniques and their deficiencies such as the difficulty to capture the complexity and the changing data. Also, the difficulty to choose the proper sample size that achieves reliability of the estimates.

Jost et al still has some problems, as it didn't study combining predictive models, including statistical models, to achieve the better accuracy in predicting the real estate price. Studies showed that depending on combined results from different predictive models achieve better accuracy than each individual model. That's why the US 6609109 B1 was proposed.

The proposed system aims at estimating the price of the real estate based on many types of predictive models. It selects the best estimates and gives each of them a weight based on the calculated precision. Then, the system combines these estimates in a combines weighted estimate into a final estimate.

These are the steps for combining the different predictive models together:

1. Accessing the metadata of the real state and a plurality of predictive models
    - This step includes storing the historical data correspondent to the plurality of the predictive models in a database.
    - Also, this step includes correcting the bias using the correspondent historical data.
    - Storing the loss function of each business application to be used later

2. Forming a plurality of estimates based on different predictive models.
    - This step includes the precision measurements.
    - There will be a system that converts the estimates of plurality of the predictive models into different formats of comparison.

3. Selecting the best estimates according to the precision measurements and
    - Excluding the estimates that have inadequate precision according to a predetermined criteria.
    - Giving each estimate a weight according to its precision value.

- The loss function corresponding to each business application will be used in calculating the weight of each real-estate.

4. Allocating the weighted estimates in a combined model to obtain the final evaluation of the real estate price.

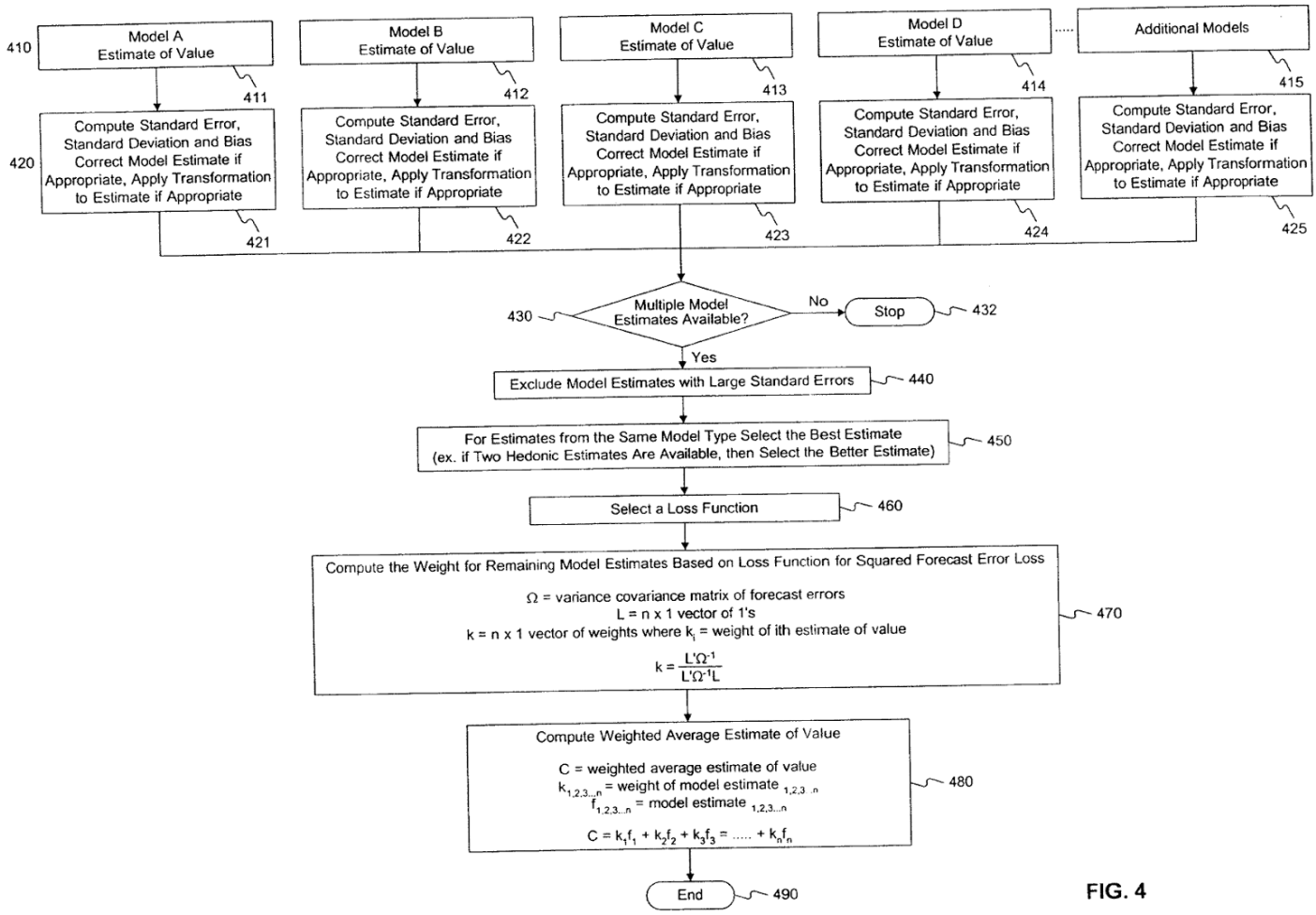

FIG. 4

Figure 1: A flowchart illustrating the combined model forecasting

3.2 Hedonic Price Model vs. Artificial Neural Network (2004)

Over the last two decades, researchers have been studying the real estate price prediction methods. In the past, the successes were in emphasizing the attributes of the property such as the property site, property quality, environment and location. However, recent studies have focused on the location attributes, transaction costs and factors affecting the future expected cost in homeownership [1].

Hedonic Price Theory:

The price of the real estate is a function of its attributes. The attributes associated with the real estate define a set of implicit prices. The marginal implicit values of the attributes are obtained by differentiating the hedonic price function with respect to each attribute [1].
The problem with this method is that it doesn't consider the differences between different properties in the same geographical area. That's why it is considered unrealistic. Flitcher et al. tried to explore the best way to estimate the property price comparing the results of aggregation and disaggregation of data. He found that the results of aggregation are more accurate. He also found that the hedonic price of some coefficients for some attributes are not stable, as they change according to the location, age and the property type. Fletcher found that the hedonic analysis can be effective while analyzing these changes. Then, the hedonic model involves regressing the properties whose prices need to be estimated against the attributes of each property. In addition, the geographical location of the property plays an important role in influencing the price of the property.

Artificial Neural Network Theory:

Neural network is an interconnected network of artificial neurons with a rule to adjust the strength or weight of the connections between the units in response to externally supplied data [1]. Neural Network Model can learn valuation patterns for "true" open market sales in the presence of some "noise" as a way of establishing a robust estimator Tay and Ho [1]. The model consists of three layers; input layer (the property attributes), the hidden layer (usually considered as a black box) and the output layer (the estimated price).

The Neural network has to be trained first for a set of data. As for a specific input, a specific output (the estimated price) is produced from the model. Then the model compares the output model (the estimated price) to the actual model (actual price). The accuracy is determined by the total mean square error. The number of the hidden layers and the number of nodes in each hidden layer can significantly affect the performance of the network.

Results:

Comparing the results of the hedonic model versus the neural network model, the neural network outperforms the hedonic model. The lack of information in the hedonic model may be the cause of the poor performance. However, there are some limitations in the neural network model, as the estimated price is not the actual price but it is close to the real one. This is because of the difficulty in obtaining the real data from the marker. Also, the time effect plays an important role in the estimation process that the neural network cannot handle automatically. Then implies that the property price is affected by many other economic factors that are hard to be included in the estimation process.

3.3 Repeat Sales Indices (2008)

Repeat Sales is a way of calculating changes in the sales price of the same piece of real estate over time. Housing market analysts use repeat sales to estimate changes in home prices over a period of months or years. Various housing price indexes use the repeat-sales method to provide information about the housing market to homebuyers and sellers, housing market investors, and those working in the housing and housing finance industries.

3.4 Using Machine Learning Algorithms for House Price Prediction (2014)

In this paper [2], the author is trying to answer this question, will the house be sold in a higher or lower price than the listing price. He compared 4 algorithms in his paper Decision Tree (C4.5), Naive Bayes, Adaboost, and RIPPER. Achieved This is the list of his attributes:

Table 1
List of physical features variables selected.

Category	Name of attributes	Descriptions	Original data type
Physical features (16)	BasementTypeValue	Type of basement (fully finished; partially finished; full; partial; walkout)	Nominal
	Bathsfull	Number of bathroom having a toilet, wash basin and bathing facilities	Ratio
	Bathshalf	Number of bathroom having a toilet and wash basin	Ratio
	Bedrooms	Number of bedrooms	Ratio
	ExteriorTypeValue	Type of exterior (brick; aluminum siding; vinyl siding; wood/cedar; composition; brick front; stone; stucco; concrete)	Nominal
	ExteriorFeaturesTypeValue	Type of exterior features (deck; bump; fenced-fully; fenced-partial; fenced-rear)	Nominal
	CoolingTypeValue	Type of cooling system (central a/c; heat pump; ceiling fan; attic fan)	Nominal
	Fireplaces	Number of fireplaces	Ratio
	TotalSquare	Square feet of living area	Ratio
	GarageSpaces	The number of cars parked inside garage	Ratio
	HeatingTypeValue	Type of heating system (baseboard; electric air filter; forced air; forced air; heat pump; radiator)	Nominal
	HeatingFuelTypeValue	Type of heating fuel (bottled gas/propane; central; electric; natural gas)	Nominal
	HotWaterTypeValue	Type of fuel for hot water (bottled gas/propane; electric; natural gas)	Nominal
	StyleTypeValue	Style of the property (colonial; contemporary; split foyer; split level; etc.)	Nominal
	LotSqft	Square feet of lot size	Ratio
	ParkingType	Type of parking (garage; covered parking; driveway; unassigned; assigned street; street)	Nominal

Table 2
List of public school rating and mortgage rate variables selected.

Category	Name of attributes	Descriptions	Original data type
Public school ratings (3)	ElementarySchoolRate	Quality of elementary school where the property is located (1–10; 10 is the highest)	Ordinal
	MiddleSchoolRate	Quality of middle school where the property is located (1–10; 10 is the highest)	Ordinal
	HighSchoolRate	Quality of high school where the property is located (1–10; 10 is the highest)	Ordinal
Mortgage contract rate and others (8)	Listmonth	Month when the property was listed	Interval
	Listprice (USD)	Price a seller asks	Ratio
	FMR	Fixed mortgage rates	Ratio
	AMR	Adjusted mortgage rates	Ratio
	City	City name	Nominal
	Zip5	Zip code	Nominal
	YearBuilt	Year the property was built	Interval
	DaysOnMarket	Number of days on market	Ratio
Dependent variable	HighOrLow	High: closing price >= listing price Low: closing price < listing price	Nominal

Figure 2

His Conclusion:

In this study, several machine learning algorithms are used to develop a prediction model for housing prices. We test for the performance of these techniques by measuring how accurately a technique can predict whether the closing price is greater than or less than the listing price. Four different machine learning algorithms including C4.5, RIPPER, Naïve

Bayesian, and AdaBoost are selected, and tested for which algorithm produces the highest rate of the accuracy.

We find that the performance of RIPPER is superior to that of the C4.5, Naïve Bayesian, and AdaBoost models. In all the tests, RIPPER outperforms the other housing price prediction models. Previous studies pertinent to housing price predictions have focused on hedonic-based methods which are conventional statistical approaches having some limitations of assumptions and estimations. More recent research tried to compare conventional ways with machine learning approaches such as neural network and SVM. However, this study compares the performance of various classifiers in machine learning algorithms, and finds the best classifier for a better housing price prediction. Thus, our study shows that a machine learning algorithm can enhance the predictability of housing prices and significantly contribute to the correct evaluation of real estate price. In practical applications, mortgage lenders and financial institutions can employ a machine learning based housing price prediction model for better real estate property appraisal, risk analysis, and lending decisions. The potential benefits of using this model include reducing the cost of real estate property analysis and enabling faster mortgage loan decisions.

Our study has the following limitations which future research could examine further. This study focuses on a specific region, Fairfax County and on a specific type of residential properties, town- houses. First, location is one of the most important factors in buying and selling real estate because real estate markets have important regional differences[2]. Other geographic regions might require different attributes. Second, residential property includes single family houses, townhouses, and condominiums. Results might be different based on the type of residential property. Third, the performance evaluation is based only on classifiers. Performance comparison of other machine learning algorithms should be considered. In future works, this study can be extended in several ways. First, it could be desirable to investigate other problem domains (real estate market prediction, interest rate forecasting, economic growth rate forecasting, oil price forecasting, and stock price index forecasting) to generalize the results of this study. Second, a future study must establish housing price prediction model that enables forecast of multiclass or continuous dependent variables. Lastly, the housing market can be influenced by macro-economic variables. Future research should consider macro-economic and environmental amenities variables for housing price prediction model inputs. For this purpose, more data sources are needed. For example, property tax and appraised value of a property, and primary residence can be achieved from tax authorities and real estate agency websites.

4. Design Overview

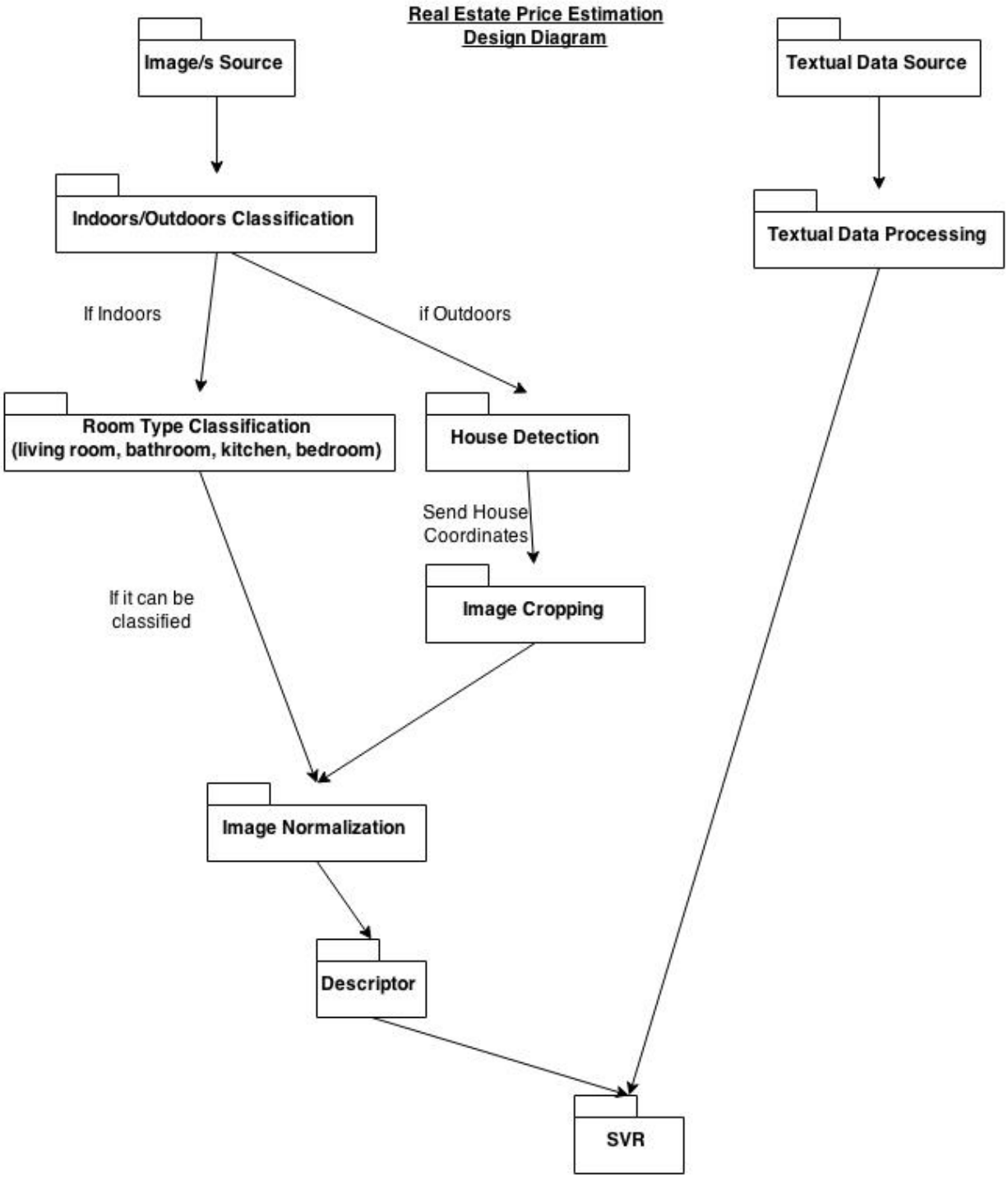

Figure 3: System Design Diagram

The system design (Figure 3) suggested in this paper has 2 main sources, images and text, each of them has his own unique processing path until both reach the regression engine

which uses Support Vector regression SVR. The textual data path is to first get the data from the source then pass it through a data processing engine which responsible for many tasks such as normalizing the data, and data cleansing. The image data path is to first get the images, then send it to an indoors/outdoors classification engine. If the data is outdoors, send to a house detection engine to detect the exact coordinate of the house, which is then sent to be cropped and then normalized and described then is sent to the SVR. The other path would be if the image is indoors, then it will pass through a room type engine (living room, bathroom, bedroom, and kitchen), then if it was correctly classified to one of these types, it will be normalized then described and then sent to the SVR. Check the following sections for more details about each box.

4.1 SVR Box

The whole design is based on a regression model where all the training data are fed to the system and a model is learnt using these data. This model can later be used for responding to different “new” queries. A query response will typically be a price expressed in the currency unit used. For the purpose of this project we will be using the library LIBSVM provided by Chih-Chung Chang and Chih-Jen Lin [3]. Training an SVM or SVR solution is not an easy task, especially if the number of features is not small. There are some good practices anyone who is using LIBSVM should follow [4]:

1. Transform data to the format of an SVM package
2. Conduct simple scaling on the data
3. Consider the RBF kernel
4. Use cross-validation to find the best parameter C and γ
5. Use the best parameter C and γ to train the whole training set5
6. Test

4.2 Textual Data Source Box

This box will be simply the input source of the training, validation and testing textual data.

4.3 Textual Data Processing Box

As mentioned before the literature review, for this problem there are lots of different textual features to be used, nominal such as Basement Type (fully furnished, partially furnished, full, partial, walkout), and Parking Type (garage, covered parking, driveway, unassigned, assigned street, street), or ratio such as number of bedrooms, number of bathrooms. This textual data box will be responsible for converting the textual data into numeric values that can be fed to the SVR box. The textual data that are ratios can be easily converted into the range -1 to 1. This range -1 to 1 is suggested by [4].which describes a technique for training the SVR in order to achieve better results. For nominal features, it works by converting each type for example into a specific value in the range from -1 to 1, but how to select these values? One way is for each feature to manually assign it a certain value based on what prior knowledge for example, giving the types that are usually associated with high prices a higher value than others.

The textual data box will also be responsible for converting the zip code, the neighborhood data or the address into relevant information that can help the SVR such as getting the traffic, elementary school rating, high school rating, quality of services in this area, and how luxurious is it considered to live in such area. This will help make all the input the SVR somehow easily converted from nominal to specific values, unlike dealing with the zip code itself for which it will be so difficult to find specific values to represent this code.

4.4 Indoors/Outdoors Detection Box

The indoors/outdoors detection module of the system aims to separate the input images into two classes where the first class is the images of houses from outside whereas the second class is the images of the houses from inside. Those are the two types of images that are expected as an input to the system. In order to build this classifier, a dataset is needed to extract distinct features that classify those two classes. In a paper, written by Paul Fitzpatrik, that discusses indoor/outdoor classification the following list of features were suggested as a starting point to build the classifier:

- Amount of “green”
- Amount of “blue”
- Degree of vertical change in brightness
- Degree of vertical orientation
- Degree of local homogeneity
- Degree of horizontal symmetry.

Fitzpatrik claimed that using those features could lead to an accuracy of about 86% [5]. The most important and effective three features were chosen from this list with a slight modification. The features that are actually extracted from the dataset are as follows:

- Amount of green in the lower third of the image
- Amount of blue in the upper third of the image
- Degree of vertical change in brightness

The slight changes of localizing the features in the image is very wise because in the specific case of differentiating between images of houses from outside and images of houses from inside the green color is expected only in the lower third of the image and the blue color is expected only in the higher third of the image in case of an outdoor house. The vertical change in brightness is calculated by subtracting the average brightness in the lower third of an image from the average brightness in the upper third of the image. Those three features were fed in all possible combinations to a Gaussian Bayesian classifier and it turned out that the best accuracy obtained is 79.6% when using one feature which is the amount of green in the lower third of the images. The Gaussian Bayesian classifier is chosen because the histograms of all three features show a normal distribution. A dataset of 704 images of houses from the outside and a dataset of 975 images of houses from inside were used to train the classifier where the testing set is one-fifth of each of the datasets and the remaining is used for training. The datasets are obtained from ImageNet.

4.5 House Detection Box

After recognizing that the image is for outdoor scene, the exact position of the house needs to be detected. Detecting the house position is needed for normalizing all the images

that contain houses. The normalized images will be the input of the SVR afterwards. To detect the house in the image, the Adaboost algorithm is used.

Adaboost algorithm is a method that combines weak classifiers to make strong classifier. Weak classifiers correspond to image features. The features that the adaboost algorithm chooses to detect the house is very critical to the success of the algorithm. The set of features used for house detection by the Viola-Johns algorithm uses Haar-like features on the integral image of the original image. The size and position of the five Haar-like patterns (figure 4) can vary but the white and black rectangles should still have the same dimensions. Hence, for the same Haar-like pattern, multiple features can result from the shift and the size change. The computed features are assumed to have all the information required to detect the house in the image.

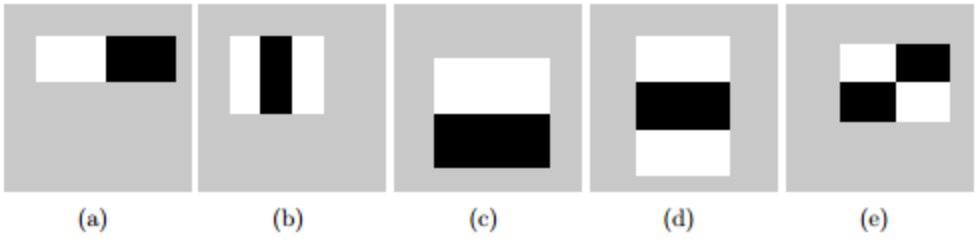

Figure 4

Then, a weak classifier is built from these features by computing the probability distribution. As, a good feature will have two probability distributions; either to be a house or not to be a house. Using many weak classifiers, a strong classifier can be built where the classifier maps an observation to a labeled value. For house detection, it assumes the form of $f : R^d 7\rightarrow \{-1, 1\}$, where 1 means that there is a house in the image and −1 the contrary and d is the number of Haar-like features extracted from an image. Then, given an observation $x \in R^d$, a decision stump predicts its label, whether a house or not, using the following rule [6].

$h(x) = (1_{\pi f x} \geq t - 1_{\pi f x<t})T = (1_{\pi f x \geq t} - 1_{\pi f x<t})1_T=1 + (1_{\pi f x<t} - 1_{\pi f x\geq t)}1_T=-1 \in \{-1, 1\}$

Where πf_x is the feature vector's f-th coordinate.

Using a trained cascade, the detected house can be determined by a window. To achieve accurate results, more layers should be trained because the more the layers, the less false positive rate.

Applying Adaboost gives very high performance with low false positives and false negatives. When testing 20,000,000 image windows, taken from 35 image, the total false positives were over 118 and the false negatives were 27 [7]. This shows how accurate the Adaboost algorithm is.

4.6 Room Type Classification Box

After classifying the image as an indoor image, we need to classify the indoor scene into categories. The categories we worked on were kitchen, living room,

bathroom and bedroom. We used the “visual bag of words” along with a linear SVM as a classification method. Also, the “tiny image features” and the k- nearest neighbors classifiers were implemented.

The first phase was building the visual vocabulary. It was built from the training data. The vocabulary word represented a feature. The vocabulary word can be considered as a SIFT descriptor of a specific scene like a kitchen. In order to build the vocabulary, SIFT descriptors are built from the training images. These descriptors are then clustered with kmeans [8].

The second phase is building the bag of SIFT features. After having the vocabulary words, the test images to be classified can be compared against them. Then, the SIFT features should be extracted from the test images. This implies that there will be many descriptors for each image. Each descriptor will be compared to the vocabulary to find the closest vocab word to the descriptor. Then, a histogram of the matched vocab word will be created. This process will happen for all the images, so each image will have its own histogram created and then all the histograms will be normalized to ensure that all histograms will have the same range of values. The normalized histogram is considered the image descriptor that will be passed to the classifier.

The third phase is building the linear the SVM classifiers. There were four linear SVMs created. Each SVM was trained as a one-vs-all classifier for a specific category. For example, the “living room” SVM was trained to recognize whether the image is a living room or not a living room. After implementing the three phases, the classifiers are applied to test the images and figure out which type of the indoors scene the image belongs to. The accuracy varied from 0.64 to 0.68 [8].

4.7 Normalization Box

The main goal of the normalization box is to unify the dimensions of the input images to make it easy for the descriptor box, which will be shortly explained, to perform well. The input images include the ones of the house from outside and inside. This module is yet to be implemented.

4.8 Descriptor Box

The descriptor box takes the normalized images as an input and finds SIFT descriptors for each image. This step is done so that the SIFT descriptor vectors would be fed to the SVR Box. This is the step where the input image is translated to a set of numbers that represent the image and the SVR Box would perform it’s prediction accordingly. This module is yet to be implemented.

5. Development Phases

5.1 Phase 0: SVR Only

In this phase we will get used to the SVR and practicing the techniques used to train the model in order to achieve the best possible result. We are using the LIBSVM (matlab version)[3]. In this phase we implemented 2 simple regressors:

1. A straight line:

```
% GENERATING THE DATA
% Numbers in the range -1 to 1 that differs by 0.1
Y = -1:0.1:1;
% Numbers in the range -1 to 1 that differs by 0.1
X = -1:0.1:1;
% get the transpose of the vector it will 1xN
Y = Y';
% get the transpose of the vector it will 1xN
X = X';

% svm options
svmopts=['-s 4 -t 0 -c 2 -g 1'];

% train SVM
model=svmtrain(Y, X, svmopts);

% test SVM on training data
[Yout, Acc, Yext]=svmpredict(Y,X,model,'');

tY = ones(1,1);
tX = [0.123];
% test SVM on test data
[tYout, tAcc, tYext]=svmpredict(tY,tX,model,'');

% print the result of the prediction.
tYout
```

2. A U curve generated using this formula Y = X^2

```
% GENERATING THE DATA
X = -1:0.1:1;
Y = X.^2;
Y = Y';
X = X';
```

```
% svm options
svmopts=['-s 4 -c 2 -g 1'];

% train SVM
model=svmtrain(Y, X, svmopts);

% test SVM on training data
[Yout, Acc, Yext]=svmpredict(Y,X,model,'');

tY = ones(1,1);
tX = [.25];
% test SVM on test data
[tYout, tAcc, tYext]=svmpredict(tY,tX,model,'');
tYout
```

5.2 Phase 1: SVR and Textual Data

In this phase we will start our testing on the textual data, and here is a simple pseudo code:

```
% GENERATING THE DATA
[Y, X] = getTrainingData();

% svm options
svmopts=['-s 4 -t 0 -c 2 -g 1'];

% train SVM
model=svmtrain(Y, X, svmopts);

% run SVM on training data and get the model
[Yout, Acc, Yext]=svmpredict(Y,X,model,'');

[tY, tX] = getTestingData();

% test SVM on test data
[tYout, tAcc, tYext]=svmpredict(tY,tX,model,'');

% print the result of the prediction.
tYout
```

5.3 Phase 2: SVR, Textual Data, and Raw Pixel Data (without any optimizations)

In this phase we will work with the textual data, and raw pixel data. It will be the same exact code as phase 1, but the difference will be in the getTrainingData, and getTestingData functions.

```
% GENERATING THE DATA
function [Y, X] = getTrainingData()
```

```
        [Y, X] = getTextualData();
        for x in X:
                x.append(getImages(x.id))
        endfor;
end;
```

5.4 Phase 3: SVR, Textual Data, and Indoors/Outdoors detection

Same as phase 1, but the difference will be in the getTrainingData, getTestingData functions.

```
% GENERATING THE DATA
function [Y, X] = getTrainingData()
        [Y, X] = getTextualData();
        for x in X:
                images = getImages(x.id);
                for img in images:
                        io = indoorsOutdoors(img);
                        if io == indoors:
                                x.indoors.append(describe(img))
                        else if io == outdoors:
                                x.outdors.append(describe(img))
                        endif
                endfor
        endfor;
end;
```

5.5 Phase 4: SVR, Textual Data, Indoors/Outdoors detection, House Detection, and Room Type Classification.

Same as phase 1, but the difference will be in the getTrainingData, getTestingData functions.

```
% GENERATING THE DATA
function [Y, X] = getTrainingData()
        [Y, X] = getTextualData();
        for x in X:
                images = getImages(x.id);
                for img in images
                        io = indoorsOutdoors(img);
                        if io == indoors:
                                roomType = getroomType(img);
                                if roomType == livingroom:
                                        x.livingroom.append(describe(img));
                                else if roomType == kitchen:
                                        x.kitchen.append(describe(img));
                                else if roomType == bathroom:
                                        x.bathroom.append(describe(img));
                                else if roomType == bedroom:
```

```
                        x.bedroom.append(describe(img));
                    endif;
                else if io == outdoors:
                    [dimensions, houseExists] = detectHouse(img);
                    if houseExists:
                        img = crop(img, dimensions);
                        x.house.append(describe(img));
                    endif;
                endif;
            endfor;
        endfor;
end;
```

6. Expected Problems and Suggested Solutions

6.1 Handling Missing Data

1. Ignoring the missing data: This is not easy, because the input of the SVR should be well formatted meaning all the fields must be present.
2. Filling the missing data with zeros.
3. Filling the missing data with the average of this column.
4. Generating all possible of combinations of features and for each combination generate a new SVR model. This solution is not practical if the number of features is not small.
5. If the missing data is in the training data, just ignore the whole row. If it was in the testing data let the network abstain.
6. Based on the available features, get the closest data point (euclidean distance) and use the missing data from this data point.

6.2 Handling Real Time Data Changes

One of the main challenges that face the system is that it does not react well to unusual changes in environmental/economical factors. For example, the estimated prices of real estate would not be affected by economic crisis neither would it be affected by natural disasters such as hurricanes or earthquakes that might happen in an area. This is the case because the price estimation is based on data that have collected in the past. Similarly, the system would not be responsive to price hikes that might happen in a certain residential area. The data that the system uses to estimate the price might become outdated in case of any such changes. To counter this kind of challenges, a server would constantly collect recent information about real estate prices and compare these prices with the ones used to train the system. Once there appears to be a pattern of a price increase or decrease in a certain area compared to the previously used prices, the system would be updated with the recent prices. In this way the system would be able to handle sudden unexpected price changes.

6.3 Data anomalies

Sometimes, the ground truth data has problems or anomalies. That make it inaccurate or corrupted to use. Anomalies can be classified into many categories; syntactical, semantic, and coverage anomalies. Syntactical anomalies are related to the format and the values used for representing the data. Semantic anomalies hinder the dataset from being comprehensive and non-redundant. Coverage anomalies decrease the entities represented in the dataset which gives inaccurate information as a ground truth data [9].

Syntactical anomalies have many forms. One form is lexical errors in which there is a contradiction between the structure of the data items and the specified format. For example, when the number of the values are unexpected whether low or high such as expecting four attributes each in a column and receiving a three columns only. Another form of syntactical anomalies is irregularities, which is concerned with the non-uniform use of values, units or abbreviations.

Also, semantic anomalies have many forms. One form is the integrity constraint violation. This means that the provided data violates one or more of the constraints determined for this attribute. For example, specifying the size of a house the has two bedrooms as zero in the dataset. Also, duplicated are another form of semantic errors. This means that two or more attributes describe the same thing but have different values. In addition, contradictions are one of the most popular forms of the semantic errors. This can happen when there is a dependency between two entities and the information provided in one of them contradicts the other one. For example, if the price of one meter is $1000 and the size of the house is 100 meters, then the house price should be $100,000, assuming that the price just depends on the size. If the house price provided in the data set was different, this is a contradiction.

Coverage anomalies also can cause the data to be inaccurate or corrupted. Moreover, these anomalies have many forms. One form is missing values, as some data will be provided as NULL while it should have a value. On the other hand, there might be missing attributes. For example, in a case where the number of bedrooms is always provided in the dataset but for a specific set of images, this piece of information is not provided.

Anomalies affect the data quality and cause inaccuracy in estimations. Also, sometimes, these anomalies push the experts to exclude some of this data to avoid depending on inaccurate data which results in poor performance and low efficiency. That's why some new techniques are working towards detecting and correcting these errors.

7. Future Work

The following work is yet to be done in order to add on to the system, and further improve the performance of the modules that are already implemented:

- Indoor/outdoor classification module: Try different classifiers, such as LDA and SVM, and compare their accuracy with that of the Bayesian Classifier and choose the one that gives the best results.
- The indoors scene classification : Try different feature and classification techniques such as, GIST features and Fisher encoding. Then compare the performance of each method and choose the one that gives the best results.

8. Conclusion

Even though that system is not complete yet, and so cannot be properly tested, the individual testing that has been done on each unit separately show acceptable results which is positive indication. Based on research that has been done regarding the system and the results obtained so far, it is safe to assume that the system could perform very well in estimating real estate prices. The proposed system would save a lot of hours when estimating the prices. The next steps that are to be taken to complete the system would be starting to implement the missing modules, then start facing the challenges that are mentioned earlier under the Expected Problems and Suggested Solutions section by implementing the proposed solutions.

References

[1] V. Limsombunchai, "House Price Prediction: Hedonic Price Model vs. Artificial Neural Network," Canterbury, 2004.

[2] B. Park and J. K. Bae, "Using machine learning algorithms for housing price prediction: The case of Fairfax County, Virginia housing data," Expert Systems With Application, pp. 2928-2934, 26 November 2014.

[3] C.-C. Chang and C.-J. Lin, "LIBSVM -- A Library for Support Vector Machines".

[4] C.-W. Hsu, C.-C. Chang and C.-J. Lin, "A Practical Guide to Support Vector Classification," Taipei, 2010.

[5] P. Fitzpatrik, "Indoor/outdoor scene classification project," MIT, Massachusetts,
2003.

[6] Y. Wang, 'An Analysis of the Viola-Jones Face Detection Algorithm', *Image Processing On Line*, vol. 4, pp. 128-148, 2014.

[7] X. Chen and A. Yuille, "AdaBoost Learning for Detecting and Recognizing Text," Los Angeles, 2004.

[8] Cs.brown.edu, 2015. [Online]. Available: http://cs.brown.edu/courses/cs143/results/proj3/dkocoj/. [Accessed: 10- Apr- 2015].

[9] H. Muller and J.-C. Freytag, "Problems, Methods, and Challenges in Comprehensive Data Cleansing," Berlin.