CSC 180-01 Intelligent Systems (Fall 2024)

Title: - House Price Prediction using Functional API

Due at 10:30 am – Wednesday, October 23, 2024

Name	**Student ID**
Taekjin Jung	303293432
Illya Gordyy	302682939
Jenil Shingala	302796429
Danny Phan	301698774

1. Problem Statement

The goal of this project is to build a deep learning model to estimate house prices by integrating textual data (such as area, number of rooms) with visual features (images of the house). This project aims to improve prediction accuracy by including visual data.

2. Methodology

Data Preparation:

- The dataset for this project was sourced from GitHub, which includes both visual and textual data for house price prediction.
    - The dataset consists of: 2140 images, representing 535 houses, with each house having four images (bedroom, bathroom, kitchen, and a frontal image).

    - A text file containing data for each house, including the number of bedrooms, bathrooms, house area, zip code, and the target price.
- Removed outliers, only houses priced using z-score
- Split training/testing data randomly for both textual and image data, then split the textual data by price

Model Development:

- Implemented two types of neural networks, CNN for image and FCNN for textual data, and merge the layers
- Applied transfer learning by loading VGG16 for our first 10 layers of CNN
- Used EarlyStopping with 5 patience during training.
- Set up the input variables for each different layer (image, text)
- Hyperparameters
    - Activation: Relu (for the first hidden layer)
    - Optimizer: adam

Evaluation Metrics:

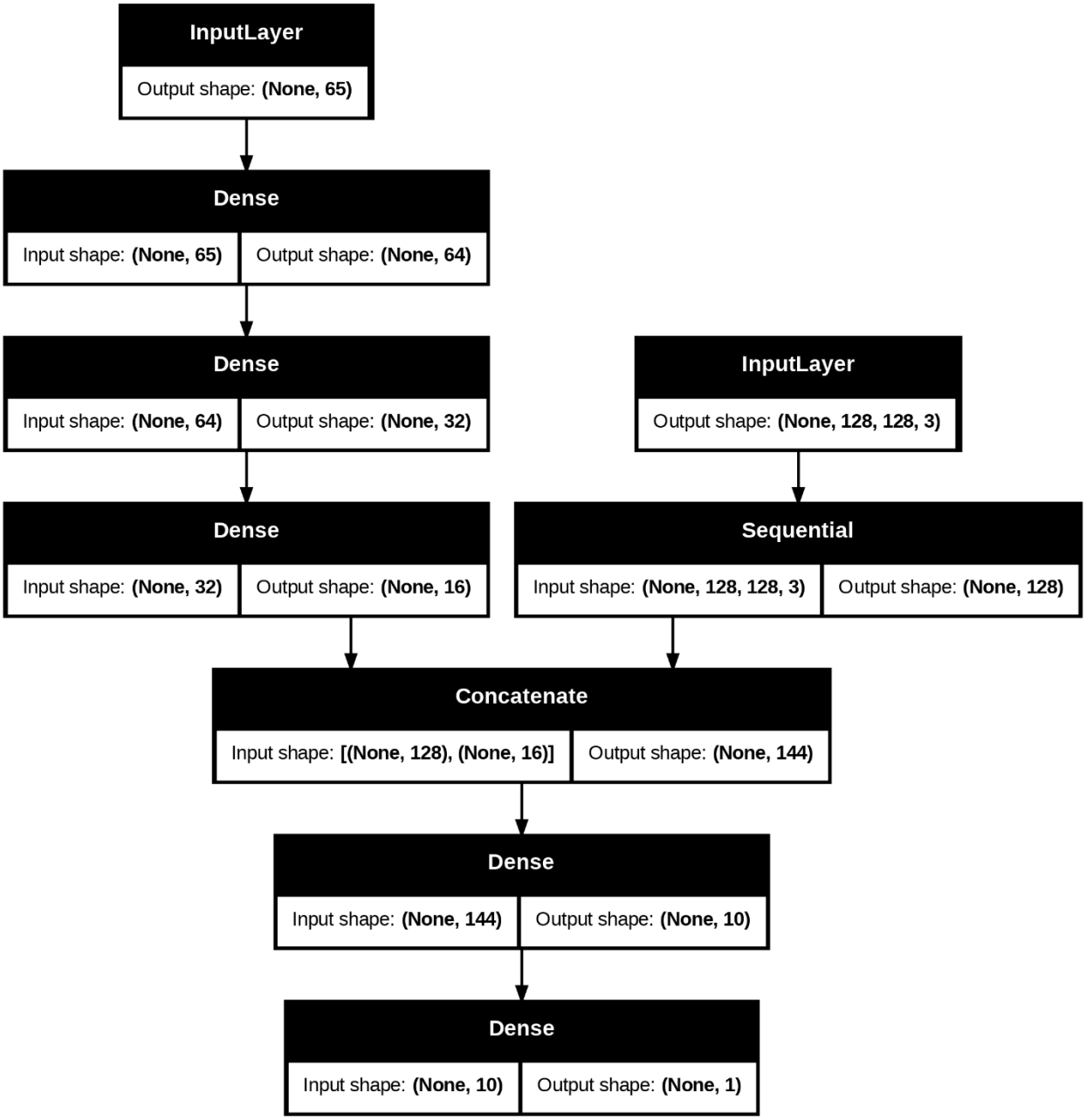
- RMSE for the accuracy of the price prediction
- Regression graph

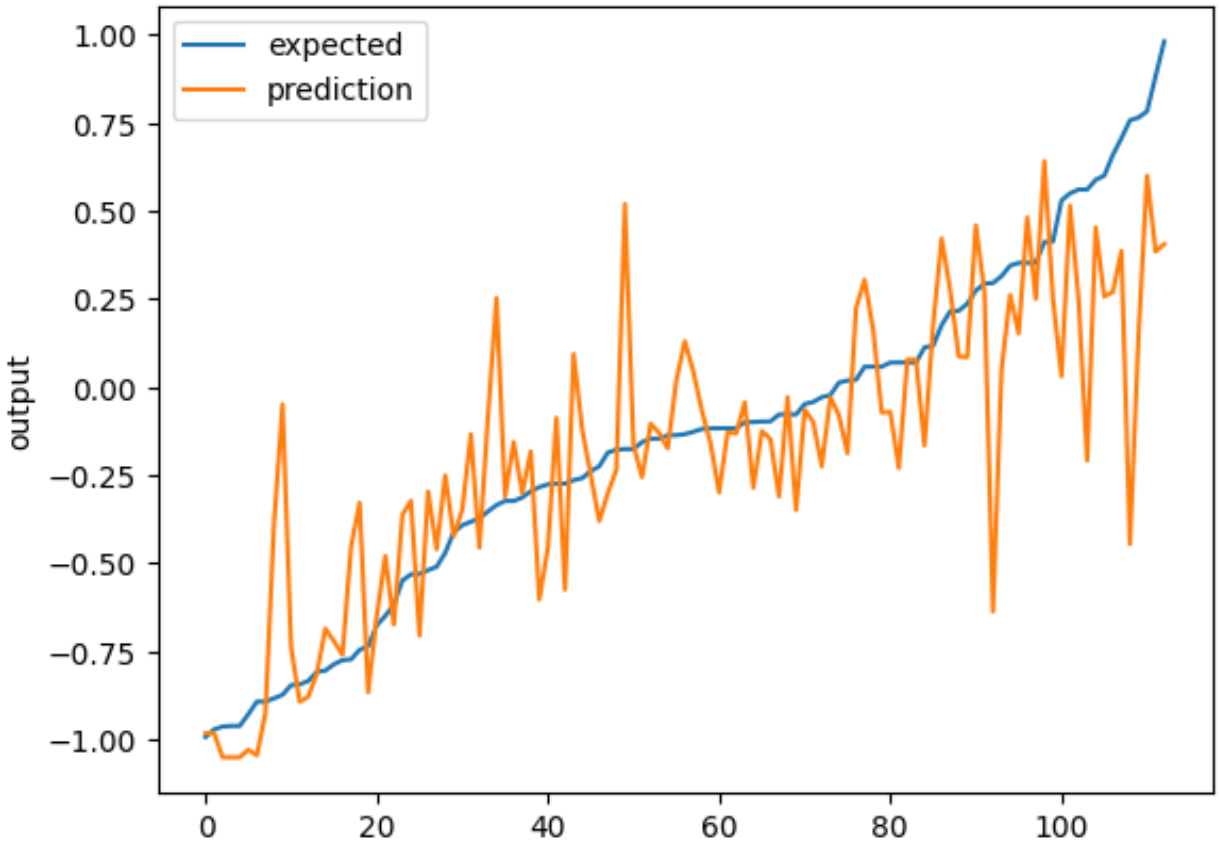
3. Experimental Results and Analysis

Model: "functional_87"

Layer (type)	Output Shape	Param #	Connected to
input_layer_8 (InputLayer)	(None, 65)	0	–
dense_21 (Dense)	(None, 64)	4,224	input_layer_8[0][0]
input_layer_7 (InputLayer)	(None, 128, 128, 3)	0	–
dense_22 (Dense)	(None, 32)	2,080	dense_21[0][0]
sequential_3 (Sequential)	(None, 128)	16,844,992	input_layer_7[0][0]
dense_23 (Dense)	(None, 16)	528	dense_22[0][0]
concatenate_3 (Concatenate)	(None, 144)	0	sequential_3[0][0], dense_23[0][0]
dense_24 (Dense)	(None, 10)	1,450	concatenate_3[0][0]
dense_26 (Dense)	(None, 1)	11	dense_24[0][0]

Total params: 16,853,285 (64.29 MB)
Trainable params: 2,138,597 (8.16 MB)
Non-trainable params: 14,714,688 (56.13 MB)
None


InputLayer
Output shape: (None, 65)
Dense
Input shape: (None, 65)
Output shape: (None, 64)
Dense
Input shape: (None, 64)
Output shape: (None, 32)
Dense
Input shape: (None, 32)
Output shape: (None, 16)
InputLayer
Output shape: (None, 128, 128, 3)
Sequential
Input shape: (None, 128, 128, 3)
Output shape: (None, 128)
Concatenate
Input shape: [(None, 128), (None, 16)]
Output shape: (None, 144)
Dense
Input shape: (None, 144)
Output shape: (None, 10)
Dense
Input shape: (None, 10)
Output shape: (None, 1)



As a result we got a final RMSE of .2787 with a model with layers shown above and Adam as the optimizer. The third figure is the regression chart.

4. Task Division

Taekjin Jung	Data Encoding/Splitting, Functional API, Additional Feature (Transfer Learning)
Illya Gordy	Data Preprocessing, FCNN, CNN
Jenil Shingala	Data Visualization, Support of Additional Feature
Danny Phan	Report, Support of Data Encoding

Challenges

- Took a lot of time using CPU
- Connecting two different layers
- Deleting outliers of the data
- Applying transfer learning
    - Problem was because we froze all the transferred data (didn't get our data)
    - We solved the problem by freezing the first 10 layers of transferred model

Learning Outcome

- How to manage functional API
- How to deal with multiple images
- How to combine two different layers (Functional API)
- How to load another model and merge into ours (Transfer Learning)