Introduction to Machine Learning-based Art Appraisal

Final Project Report
for
Data Mining, Machine Learning, and Deep Learning (CDSCO1004E)
MSc in Business Administration and Data Science
Copenhagen Business School

Written by
Erik Konstenius (149887), Muhammad Azkaenza (142892)
and Theodore Balas (149149)

Course Coordinator	Raghava Rao Mukkamala
Number of Pages	14
Number of Characters	33,046
Semester	Spring 2022
Submission Date	24 May 2022

Abstract

Appraising art has traditionally been a manual process conducted by art experts (Bailey, 2020). In this paper, we investigate the possibility of leveraging machine learning algorithms and deep learning frameworks to appraise paintings from the website Saatchi Art. We trained three groups of models; one that was only trained on tabular data surrounding each painting, one that was trained on images of the paintings and one that was trained on both tabular and image data. Of the combined model, VGG16, VGG19, MLP, XGBoost and linear regression we found that the XGBoost regressor achieved the most promising results. The results from this paper show that the visual features in a painting do not seem to be an important element in determining the price of a painting. Moreover, the precision in the predictions shows that an art appraisal system on websites such as Saatchi Art is possible.

Keywords: convolutional neural networks, deep learning, machine learning, VGG16, XGBoost, transfer learning, art, price prediction

Table of Contents

Table of Figures

Introduction

Approximately fifty billion dollars worth of art and antiques were sold in 2020 whereby about a quarter of the total sales originate from online marketplaces (The Art Market 2022, 2022). Three of the largest marketplaces for art, Etsy, Saatchi Art and eBay, list nearly 5,000,000 paintings as of 2022. Websites such as these have enabled hobbyist painters to put their paintings up for sale and the dependence on traditional art galleries and auction houses may have decreased as new ways of selling art emerge. Traditional auction houses such as Sotheby's rely on the human appraisal of the paintings they sell. For Sotheby's, it adds up to about 50,000 appraisals per year (Bailey, 2020). Through our own research, we have not found that online marketplaces for art, such as Saatchi Art, currently offer appraisals of paintings.

Our paper aims to investigate if one could leverage deep learning to appraise art with the motivation to democratize the art market further by offering art appraisals to painters and hobbyists that sell art on websites such as Saatchi Art. A successful model could also help art buyers find undervalued art. In between, we have marketplaces such as Saatchi Art that could benefit from revenue gains as increased transparency has shown to lead to higher liquidity (Bailey, 2020). The model would appraise - or rather recommend a price - of paintings. We have restricted the scope of the appraisal system to exclude fine art. The choice to exclude fine art is motivated by the inherent difficulty in art valuations, an intention to differentiate from the focus of previous research and a will to utilize the large publicly available datasets found on these websites to better understand if art valuations can be applied at larger scales on websites such as Saatchi Art. These points will be discussed throughout this paper.

Traditional art appraisal

Research has shown that appraisers take into account factors such as the country of origin, age of the painting, dimensions, rarity, signature, materials, subject, and condition, to name a few, in their valuation. Some argue that fine art is particularly difficult to appraise as external factors such as prestige and the image of the painter play a significant role in the price - e.g. Banky's "A girl with a balloon" that according to some estimates doubled in value from 1,400,000 USD after being shredded at auction (Bailey, 2020). Another example is the banana that was taped to a wall and sold for 150,000 USD (Pogrebin, 2019). Although, traditional appraisals by auction houses like Sotheby's are conducted by art experts, studies have shown that the presale low and high estimates of final hammer prices are more often wrong than right (Bailey, 2020).

Machine learning-based art appraisal

Data scientists at Sotheby's have indicated that they are researching how they could alleviate some of the work that is currently carried out by art experts by utilizing artificial intelligence (Bailey, 2020). Yet, attempts to train machine learning models solely on the paintings

themselves are expected to perform poorly if the prices are in reality mainly determined by the reputation surrounding the painters and their paintings or by other external factors. This has been shown by researchers in previous papers (Ayub, Orban and Mukund, 2017). However, efforts to combine tabular data surrounding the painting with a ConvNet of visual features of the painting itself have achieved mixed results. In an article published on Artsy.net, researchers created a model that achieved an average difference between the actual price and the forecasted price of 5.5% on a dataset of paintings originating from one painter. The analysis of feature importance showed that the size, coloring and if it was painted on paper or canvas all played an important role. The most important factor however was the number of billionaires in the world. Similar results are observed in a paper by Henry Nho and Haijin Park. They found that combining image and tabular data from auction houses outperforms models that only deal with one type of data. The model with the most promising results is a concatenation of a VGG16 architecture and an MLP with two dense layers at the end. Worth noting is that deep ConvNets like a ResNet50 using pre-trained weights performed among the worst of all models reaffirming previously mentioned research that the visual features are not instrumental for the valuation of paintings (Nho and Park, 2019).

Research Methodology

Data retrieval

As with (almost) every data-science project, there is not much science to be produced without available data. Since no accessible dataset fulfilled the prerequisites[1] for the project, web scraping was deemed necessary to retrieve the required data. After considering most of the reputable art selling websites, Saatchi Art was chosen. Among the reasons that lead to that decision were:

i. the wide array of available art pieces from all over the world, being one of the largest online art galleries
ii. the selling of art pieces that that are:
    - original; allowing more insight into the pricing of the artists as well as avoiding pollution in the dataset from copies/reprints etc.
    - affordable; compatible with the widely- -available-art goal set in the research question
    - contemporary; greatly reducing the importance of time as a parameter thus, allowing the problem to be considered static
iii. almost the entirety of the images comprise of solely the art piece and have no background or frame that would further complicate the image pre-processing by requiring a cropping of only the painting area or, if inputted as are, undermine the model

The first attempted method to acquire the data was the creation of a custom-built python web

[1] those being: a. sizeable enough to produce results while implementing deep learning, b. the inclusion of both the images of paintings and the required tabular data

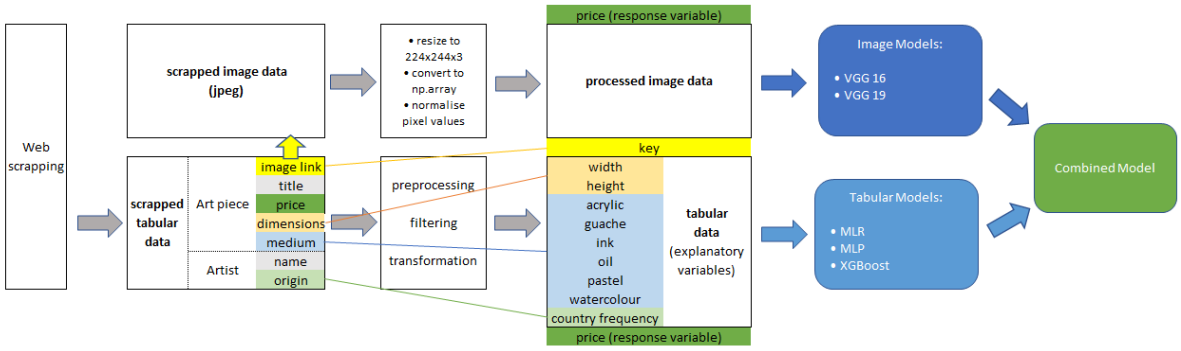


Figure 1 – Workflow

scraper using Selenium and Beautiful Soup, however, the scraping was blocked by the website that prohibited the use of the aforementioned libraries. Therefore, it became apparent that a more refined tool had to be utilised (adhering of course to the terms of service while also, creating as little load to the target servers as possible). Among those available, Octoparse was chosen given its seamless compatibility with Saatchi Art. After defining the elements to be extracted and optimizing the parameters, the scraped data was downloaded, saved, and then exported to an excel file. The scraped database consisted of 136043 rows of data with the following columns:

1. a link to an image file;
2. the title of the painting;
3. the price of the painting;
4. its dimensions;
5. its medium*;
6. the artist name; and
7. country of origin

Note that during the scraping only the search page was accessed, together with the accompanying information available there, and not the individual art pieces' pages. Since the medium could not be scraped from the search page, a work around was to use the webpage's filtering option. Consequently, six different scraping iterations with different filtering parameters (mediums) occurred.

The only notable missing characteristic is the *subject* of the painting. Although it plays an integral part in determining its price (Garay, 2017), it proves extremely hard to define it "objectively". Although the self-definition by the painting also plays a part, the subjective classification might lower the robustness of the model. For instance, please observe the paintings below


Both categorized as "Nature" in Saatchi's database. The inclusion of the subject could only be possible if it has been validated (e.g. in auction house databases). A standardization of the painting subjects, by human or machine, would be especially helpful in relevant future projects.

The excel file was then imported to a python environment as a dataframe using pandas. Note

that the images were downloaded from the link column using a VBA script. For more information on the preprocessing of the images please look at the section below as the following discussion concerns the tabular data.

Data preprocessing and description

Duplicate removal - Due to some paintings appearing in multiple scraping iterations or multiple times within the same iteration, duplicates rows, i.e. those sharing the shame unique image link, were removed (~ 16000 rows).

Null Value Removal - After removing duplicates, rows with any null values present were also removed; relatively few in number (~ 6300 rows in total, most in the country origin column and a few in the price column). Note that, besides the nulls, the validity of the existing values was also checked after their structure was finalized.

From the scraped database above, the title of the painting and artist names were only maintained for the purpose of completeness and did not contribute to the project in any way other than a point of future reference. Therefore, the remaining columns were:

1. link to an image file (string)
2. the price of the painting (numerical)
3. its dimensions (numerical)
4. its medium* (categorical)
5. the country of origin (categorical)

Value extraction and transformation

Link to an image file → image name - From the link column the image name is extracted which acts as a unique key in synchronizing tabular and image dataset. After extracting the image name, a few broken links were removed.

Dimensions → height + width - From the dimension's column, height and width were extracted. Depth also was dropped since it is a characteristic of the frame and not the painting itself (while the frame contains artistic value, narrowing the scope of the problem is also important).

Outlier removal

Having the numerical values in a useable format, the price and dimensions outliers were studied and removed.

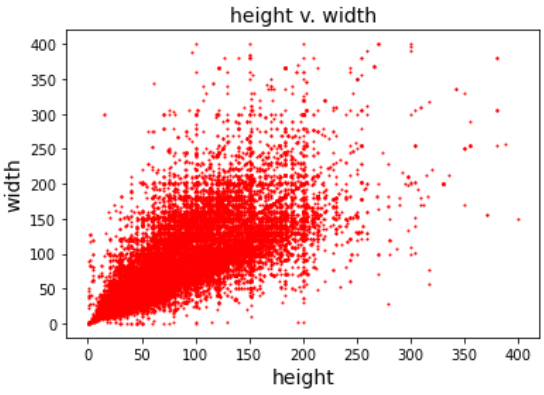


Figure 2 – Dimension

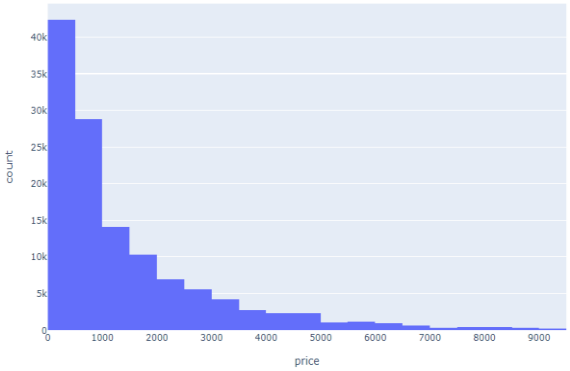


Figure 3 – Price Distribution

Height and width - By removing all art pieces with any dimension greater than 200 cm (around 1,5% of the total dataset), the model robustness was increased while retaining most of the information from the dataset.

Price - Figure 3 shows the price distribution of paintings in the dataset. By restricting the maximum price to 9000 EUR we greatly reduced the number of outliers in the dataset. Yet, we still allow plenty of deviation from the mean to ensure proper representation. Also note that besides the statistical implications, the decision to limit the price was also affected by the emphasis on affordable art set out in the research question.

Encoding

Normally the encoding takes place before the train/test/validation split dataset in order to avoid data-leakage that would make the model evaluation overly optimistic. But in this case, since the distribution of the characteristics was dense and the numerical data encoded using MinMax, only the maxima and minima would affect the result. After drawing 10%+10% of the database to create the test and validation sets, the minima and maxima in every set is almost identical, therefore the affect is miniscule.

OneHot (Medium) - Medium was encoded using OneHot since the unique number of values was only six and the added dimensionality appeared to outweigh the risk of being perceived as an interval feature as would be the case with most encoding methods.

Frequency (country origin) - As the number of unique countries was considerable, OneHot was not an option due to the Curse of Dimensionality. Since the number of paintings per country does have an importance in price determination (Garay, 2019), the country column was transformed into a number representing as the frequency of this country's paintings in the dataset. The main downside of the method being the loss of information if multiple countries share the same frequency which is not applicable in this case.

MinMax (width, height, price) - Finally, as a method of standardising the numerical features, MinMax was utilised to avoid over representation by larger numbers while also maintaining the distribution of the features. Since the latter was not known, the data was not normalised.

Finally, due to the processing capacity limitations regarding the images, the data frame was further limited to 60000 rows by dropping

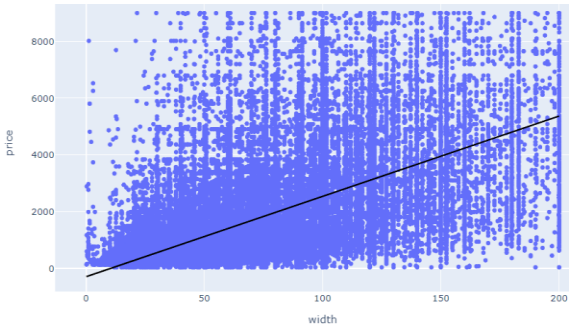


Figure 4 – Relationship between Width and Price

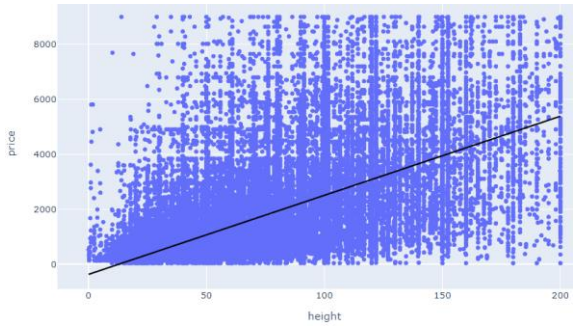


Figure 5 – Relationship between Height and Price

countries to reduce the noise in country origin frequency and taking an (almost) equal representation of the six mediums.

The final dataset contained data on the width & height (float between zero and one), medium (six categories, encoded), country frequency encoding (float between zero and one) and price float (dependent variable, between zero and one).

Note that paintings using oil and acrylic appear to be valued higher than paintings using gauche and watercolor (Figure 6). The width and height of the painting appear to have positive correlation with the price (Figure 4 and 5). The relationships do however contain some noise.

Images retrieval and preprocessing

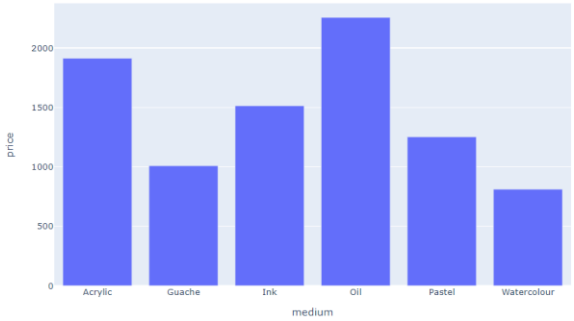


Figure 6 – Average Price of Painting by Medium

The image dataset was created by sequentially reading a list of URLs of the images of the paintings that we got from the web-scraped tabular dataset. We applied a simple VBA script that opened each link and saved the corresponding image to a folder on our local system. 136 007 images added up to about 3.25 GB of space. This step took approximately 20 hours. After the dataset was split into train, validation, and test sets, the relevant images were moved to three different folders according to a split ratio of 80:10:10. The last part of the image link, which is also the name of the image, was used as a unique key in moving the images to their respective folders. By default, the images were sorted based on their name, thus we had to ensure that the images and prices in the split dataset also followed the same order.

The images were further preprocessed to meet the input requirement of VGG16 and VGG19 architectures. This means that we resized the images and converted the pixel values into a range between zero and one. Afterwards, we converted all images into one NumPy array. This last step was memory intensive and not possible without a good understanding of memory allocation in Python. Without attempts to optimize the performance of the model, it would require over 100 GB of memory to convert the images to an array. We used Fast AI's batch processing as well as cloud computers from Ucloud with 400 GBs of memory to finish the task. We also reduced our dataset to 60 000 paintings. This reduction is sub-optimal but it made it considerably easier to work during the preprocessing and training.

Modeling

The modeling of the supervised regression problem of appraising art was split into three strategies; first strategy that was only trained on tabular data relating to the paintings, second strategy was only trained on images of the paintings and third strategy was trained on both tabular and image data (Figure 7).

Figure 7 – Models used Based on the Type of Dataset

By taking previous research into consideration, the first step was to investigate the predictive power of the tabular data.

MLR - We trained a simple linear regression without regularization. A regression model has the benefit of high interpretability, does not take very long to train, and does not have many parameters to tune - especially if we do not need to regularize it.

MLP - Furthermore, we constructed an MLP based on what is commonly perceived as good hyperparameters and architectures in MLPs, by the particular task at hand and by the dataset we have available (Géron, 2019). The initial model uses four dense layers and two batch normalisation layers that standardize the output of a layer (Géron, 2019).

XGBoost - Lastly, we trained an XGBoost regressor to see how an algorithm from the family of gradient boosting algorithms would perform. Research has shown that XGBoost in particular is often an important component of winning entries in machine learning competitions (Géron, 2019). Tree ensemble models (such as XGBoost) are usually the go-to method for classification and regression problems with tabular data. Still, several deep learning models for tabular data have been proposed during the recent years, that claim to outperform XGBoost for some use cases. A study conducted by Shwartz-Ziv et al. (2021) reports that XGBoost outperforms these deep models across the datasets, including the datasets used in the papers that proposed the deep models. It also demonstrated that XGBoost requires much less tuning, although an ensemble of deep models and XGBoost performs better on these datasets than XGBoost alone.

ConvNets - ConvNets were used to train on the image data. Traditional machine learning and deep learning models are in many cases not able to deal with the large number of features that are found in images. ConvNets solve this problem by using partially connected layers and weight sharing (Géron, 2019). Through effective use of filters in convolutional layers and pooling kernels, deep ConvNet architectures have shown to repeatedly be able to pick up complex patterns in images, and are today one of the default choices when making predictions on image data. These models are commonly applied in fields such as object detection, image captioning and autonomous driving (Géron, 2019). The chosen base model follows a VGG16 architecture meaning it has five sections of convolutional layers that each end with a max-pooling layer. The filters in the convolutional layers extract important features in each image into feature maps and the pooling layers help to shrink the input. The pooling layers thereby help alleviate some of the computational load as well as reduce the risk of overfitting (Géron, 2019). In order to make

predictions, the output of the last pooling layer was flattened and fed into two dense layers. Between these two layers, we placed a dropout layer that would set input neurons to zero for half of the neurons. This is motivated by reducing the risk of overfitting. Note that the neurons are not dropped during testing. The output layer uses the linear activation function that simply returns the weighted sum of its inputs - and for our purpose returns a price estimate. The weights in the ConvNet were initialized using the weights from a VGG16 model that was trained on the ImageNet dataset. Furthermore, we froze the first five layers that pick up the most simple shapes in the images. Results from previous research on the effect of transfer learning from ImageNet on images of paintings show that weight initialization and freezing result in better performance than training a model from scratch (Hentschel, Wiradarma and Sack, 2016). We estimated that freezing additional layers would make the model underfit the dataset due to differences between the paintings in our dataset and the images in ImageNet.

Lastly, we constructed a model that concatenated the output from a custom-built ConvNet and a custom-built MLP to make predictions that incorporates both image and tabular data. The input to the concatenated part of the model was further fed into a final MLP. The combined architecture can be seen in Figure 8.

The initial goal was to use the ConvNet that performed the best on the image data and combine it with the MLP that performed the best on the tabular data. Yet, exploding gradients made the model too unstable for training. We ended up creating a custom-built model that was robust enough for training. The following section will dwell deeper into the performance of our first model and how we changed the hyperparameter and model architecture to create a more stable model.

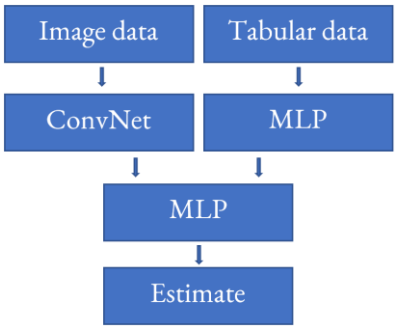


Figure 8 – Architecture for the Combined Model

Training and Hyperparameter Tuning

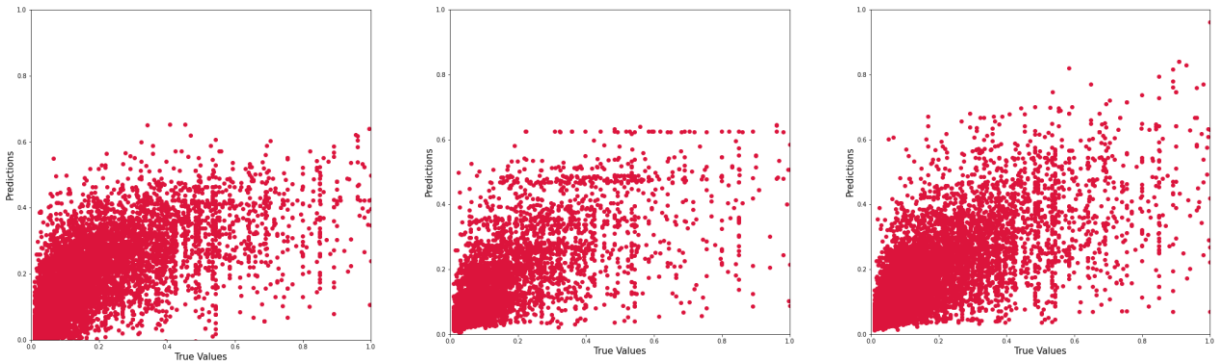
MLP - The results from the initial MLP on the validation set were on par with previous research with an R2 of approximately 0.5 (Nho and Park, 2019). We constructed a second MLP through KerasTuner's implementation of random search to see if we can improve upon the results of the initial model. The tool searches randomly through combinations of parameters in a user-defined hyperparameter space. We restricted the search space by what is commonly perceived as good hyperparameters and architectures in MLPs, by the particular task at hand and by the dataset we have available (Géron, 2019). The search space consisted of testing between one and eight hidden layers, between ten and a hundred neurons per layer and a learning rate of 0.01, 0.001 and 0.0001. It improved the R2 by about 0.3 percentage

points. We could also have tried bayesian optimization to create an optimizer that learns from previous trials to get a better performing model.

XGBoost - Three different XGBoost regressors were created. The first one was trained using the default parameters. The second was trained after optimizing the hyperparameters using grid search. Finally for the third, the model was trained using handpicked parameters derived from a custom methodical hyperparameter tuner. The third one performed the best among them, explaining 5% more of the variance in price compared to the second best (grid search). It needs to be stressed that it also had the best performance overall compared to all other models used in this paper. XGBoost should not perceived solely as an algorithm. It is an entire open-source library, designed as an optimized implementation of the Gradient Boosting framework focusing on flexibility, speed as well as model performances. Its strength is not derived purely algorithmically, but also from all the underlying system optimization (hardware optimization, parallelization, caching), features we did not utilize and could possibly further boost its performance.

ConvNet - One limitation with ConvNets is that all computations during the forward pass need to be preserved in memory for the reverse pass in the backpropagation step (Géron, 2019). In our case, this requirement added up to about 150 GB of RAM when training the model. Yet, better parameterization and memory usage could likely have reduced the required memory significantly. Some examples of potential tweaks to reduce memory usage are smaller batch sizes, larger stride, more pooling layers, constructing a simpler model or removing data. We did however not need to compromise for better memory usage as we had access to about 400 GB of memory.

The initial VGG16 model performed poorly on the validation set as shown by the fact that the loss is not decreasing noticeably after each epoch and that the R2 is negative. The model may have got stuck in a flat section in the loss function. A horizontal line through the trend would create better predictions as observed from the negative R2. The model appeared to underfit and fails to capture the structure in the data. We can create a deeper model to combat a model that is underfitting. We can also increase the number of epochs and thereby let it train for



MLR MLP XGBoost

Figure 9 – Prediction vs True Price

a longer period. Below are the steps we took to improve our model.

First, we changed the architecture to a VGG19 to try to capture more complex patterns in the data. This will mean that the model will have three more convolutional layers. The model is otherwise identical to the previous VGG16. We justify adding more layers because it appears to be underfitting rather than overfitting. This is expected to increase the training time.

Second, we removed the dropout layer as overfitting did not seem to be a concern. It could be that the network did not have time to correct the errors caused by the dropped neuron before the weights got updated. Additionally, if we do not train until convergence we might not benefit from the dropout layer. A drawback with removing the dropout layer is that training times may increase.

Third, we increased the learning rate. It appears as if the VGG16 model gets stuck at a certain point in the loss function. This could be a saddle point or a local minimum where the model does not know how to update the weights properly to escape because of a too low learning rate. By increasing the learning rate we allow the optimizer to make larger jumps and hopefully not get stuck as easily. We also added a batch normalisation layer before the dense layers. Batch normalisation is usually used to combat overfitting but we use it to allow a higher learning rate.

After applying the above-mentioned steps we reran the model in thirty-minute intervals to get a better understanding of expected performance and training time. From these quick runs, we decided to decrease the number of neurons in each dense layer by a factor of two and froze a total of five convolutional layers and decreased the number of epochs to ten. These steps were mostly carried out to reduce training time. We understand these final modifications are possibly not ideal if a model is underfitting the data.

ConvNet & MLP - The combined model was initially designed as a combination of the previously designed VGG19 and MLP, but it appeared unstable during training and experienced exploding gradients. It was not too unexpected as deep networks run the risk of experiencing exploding gradients when propagating through the network. To counter this behaviour, we used Xavier initialization, gradient clipping and the ReLU activation function as research suggests to reduce the danger of exploding gradients at the beginning of training. The gradient clipping puts a user-defined threshold on the gradients in the backpropagation (Géron, 2019). We also added batch normalization in both the tabular MLP and the concatenated MLP to greatly reduce the number of training steps. It increases our training time but research shows that it leads to faster convergence (Géron, 2019). Lastly, we decreased the learning rate to combat the exploding gradient. We removed three convolutional layers and reduced the number of filters in each convolutional layer by a factor of two. The combined model, after the aforementioned steps, consisted of approx. 3.8 million parameters. This means that the total

Model	Metrics			
	MSE	R2	Training time	Memory
Tabular				
MLR	0.0140	0.5028	seconds	insignificant
MLP	0.0136	0.5097	seconds	insignificant
XGBoost	0.0124	0.5549	seconds	insignificant
Images				
VGG16	0.03	0.0588	9.5 h	150 GB+
VGG19	0.02	-0.0104	12.5 h	150 GB+
Tabular + Images				
MLP + ConvNet	-1.91	-67.786	10.3 h	150 GB+

Figure 10 – Model Performances

Model	MSE	R2
MLR	0.014581	0.50902
Initial MLP	0.014330	0.52608
Random search MLP	0.014238	0.52911
Default XGB	0.013850	0.54193
Grid search XGB	0.013558	0.54193
Methodical XGB	0.013387	0.55726

Figure 11 – Model Performance of All Tabular Models

number of parameters is about 20 % of that of the VGG19 model.

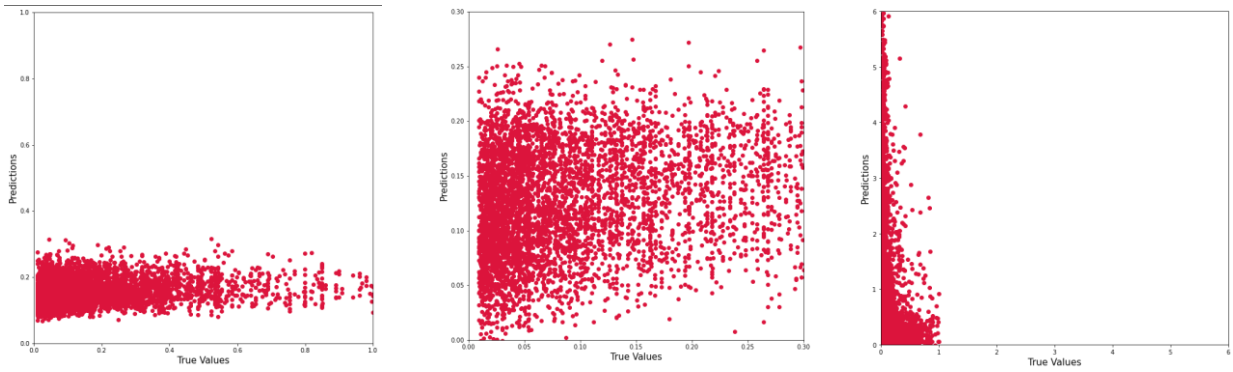
Results

From the summary statistics in Figure 10, it can be observed that the models that were solely trained on the tabular data outperformed the other two groups of models in terms of MSE and R2. The model that used both tabular and image data performed the worst.

Although the difference between the models that were trained on tabular data is small, XGBoost achieved the best performance with an R2 of 0.55726. A summary of the performance of all tabular models can be seen in the table below. The table highlights the performance gains from the hyperparameter tuning of each model. Each model did perform better after hyperparameter tuning.

The predictions the MLR, best MLP and best XGBoost regressor made can be observed in Figure A1-A6. All three tabular models appear to be able to extract important features in the data that help determine the price. The results from the models that were trained on tabular data are similar to previous research that looked at fine art at art auctions (Nho and Park, 2019). The dataset used in this paper did however not include features such as the presale estimate by an art appraisal, the economic environment, which auction house it was sold at as well as during which years the painter lived. This indicates that the models used on the tabular dataset in this paper performed surprisingly well considering the rudimentary dataset at hand.

The VGG16 model showed some better results compared to the VGG19. VGG16 achieved a R2 of Yet, both models fail at their task of predicting prices with any meaningful precision. The decisions the models made can be observed in the Figure 12. Both models appear to fail at picking up features that affect the value of paintings seen in figure 12 the negative R2 on the predictions, seen in Figure 10. The results from the two models are however similar to previous research (Nho and Park, 2019).



VGG16 VGG19 Combined

Figure 12 – Prediction vs True Price

The combined model achieved the worst performance among all of the seven trained models. Although the fluctuations in the gradient were considerably better after initial tunings, the model does not seem to be able to generalize well on unseen data. Further papers could investigate how to make the model stable while retaining high precision in the estimates as well as a high degree of generalizability. Figure 13 shows a table consist of the training history and how the model fails to generalize on the validation data. And decisions made by combined model can be observed in the Figure 12. It appears to fail at picking up features that affect the value of paintings.

Epoch	MSE	R2	Val MSE	Val R2
1	0.0171	0.36	3.2978	-160.32
2	0.0165	0.38	0.3851	-165.17
3	0.0163	0.39	12742.9690	-615671.94
4	0.0176	0.33	57.8969	-1570.00
5	0.0163	0.40	0.0149	0.43
6	0.0163	0.39	2.0463	-96.52

Figure 13 – Training History of the Combined Model shows signs of overfitting

Discussion

The results from this paper have shown the inherent difficulty of predicting the price of paintings using machine learning algorithms and deep learning - especially if the models are only trained on the paintings themselves. The XGBoost achieved the best results on our metrics among all of the seven models that were trained and would through an overall assessment be the model we would recommend for implementation into production if it would be used as a live art appraisal system on Saatchi Art. The model was able to produce relatively accurate predictions on a simple dataset of solely tabular data. Considering that the standard practice for art appraisals is to produce a low and high estimate and not an exact valuation could mean that end-users would be more lenient if the model happened to make inaccurate estimates (Bailey, 2020). The XGBoost model is furthermore fast to train and requires significantly less computational resources compared to the models that incorporate image data. This enables a degree of flexibility for fine-tuning and further developments that the deep learning models

using image data do not offer to the same degree.

The results from this paper reaffirm previous research that the visual features in paintings are not necessarily the most important feature in art appraisals. While tasks such as detecting a pedestrian on a sidewalk, adding a caption to an image or face recognition are easy tasks for us humans, predicting the price of a painting by only looking at an image of the painting is to most people significantly more difficult. Marketing and brand management play a vital role in every business, and without a proper marketing strategy can even excellent products experience low demand. We cannot see how it would be any different for paintings. With this in mind, we argue that it would be unexpected to see a ConvNet that has only been trained on images of paintings outperform models that have used tabular data surrounding the painting.

A limiting factor in our paper is that we used the listing prices picked by the painter. We would ideally use the price of sold paintings to train our models. We were not able to find such a dataset that was sufficient for our task.

Conclusion

In this paper, we trained seven machine learning models using three different datasets that relate to listed paintings on the website SaatchiArt.com in order to see how well machine learning algorithms can predict the price of paintings. First, we trained a linear regression model, an MLP model and an XGBoost model on tabular data surrounding the painting. Second, we trained a VGG16 model as well as a VGG19 model on images of paintings. Third, we trained a model that consists of a custom-built ConvNet, an MLP and an MLP that takes a concatenation of the two mentioned models. This third model takes both image data and tabular data. The results from our paper show that visual features in paintings are not what primarily drive the price. XGBoost achieved the best result with an R2 of approximately 0.5572 on the test set.

Future papers could further investigate feature importance in art valuation. The tabular dataset used in this paper contained the very basics surrounding paintings. Given that previous research found that the number of billionaires was the most important factor for the valuation of fine art, it is worth investigating what other external factors could determine the price. One could also look at other ConvNet architectures and how well they perform together with an MLP. Future papers can also see how the ConvNets perform if they are trained for longer periods.

References

Art Basel. 2022. *The Art Market 2022*. [online] Available at: <https://www.artbasel.com/about/initiatives/the-art-market> [Accessed 19 May 2022].

Ayub, R., Orban, C. and Mukund, V., 2017. *Art Appraisal Using Convolutional Neural Networks*. [online] Available at: <http://cs229.stanford.edu/proj2017/final-reports/5229686.pdf> [Accessed 19 May 2022].

Bailey, J., 2020. *Can Machine Learning Predict the Price of Art at Auction?*. [online] HDSR. Available at: <https://hdsr.mitpress.mit.edu/pub/1vdc2z91/release/3> [Accessed 22 May 2022].

Chen, T. & Guestrin, C. 2016. *XGBoost: A Scalable Tree Boosting System.* [online] Available at: https://www.kdd.org/kdd2016/papers/files/rfp0697-chenAemb.pdf

Garay, Urbi. (2019). Determinants of art prices and performance by movements: Long-run evidence from an emerging market. Journal of Business Research. 127. 10.1016/j.jbusres.2019.03.057.

Géron, A., 2019. *Hands-on machine learning with Scikit-Learn and TensorFlow*. 2nd ed.

Hentschel, C., Wiradarma, T. and Sack, H., 2016. Fine tuning CNNs with scarce training data–adapting ImageNet to art epoch classification. IEEE, pp.3693–3697.

Nakkiran, et. al. 2019. *Deep double descent: Where bigger models and more data hurt.* https://arxiv.org/abs/1912.02292

Nho, H. and Park, H., 2019. *The Art of predicting Art Auction Price*. [online] Available at: <https://cs230.stanford.edu/projects_fall_2019/reports/26261328.pdf> [Accessed 19 May 2022].

Pogrebin, R., 2019. *Banana Splits: Spoiled by Its Own Success, the $120,000 Fruit Is Gone (Published 2019)*. [online] Nytimes.com. Available at: <https://www.nytimes.com/2019/12/08/arts/design/banana-removed-art-basel.html> [Accessed 22 May 2022].

Shwartz-Ziv, R. & Armon, A. 2021. *Tabulardata: Deep learning Is not all you need.* https://doi.org/10.48550/arXiv.2106.03253

Woodham, D. and Liu, D., 2019. *Using AI to Predict How Much Rothko Paintings Could Sell for at Auction*. [online] Artsy. Available at: <https://www.artsy.net/article/artsy-editorial-ai-predict-rothko-paintings-auction-prices> [Accessed 19 May 2022].

Appendix

The datasets and the scripts that have been used in this paper can be accessed by the following link: https://drive.google.com/drive/folders/1-76Gq_GDkrKm21L2YYANiCtGDnm0Rroe?usp=sharing

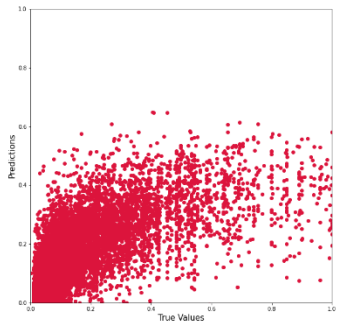


Figure A1 – Predictions vs true price of the MLR

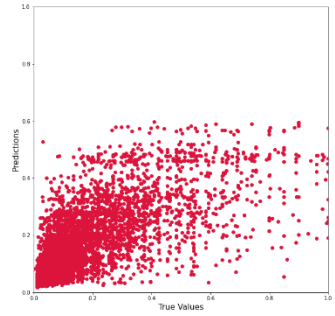


Figure A2 – Predictions vs true price of the initial MLP

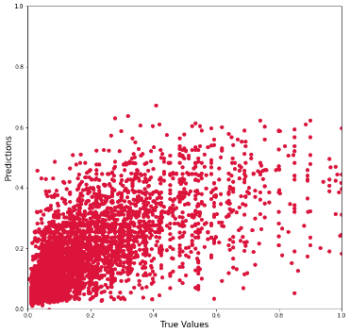


Figure A3 – Predictions vs true price of the grid search MLP

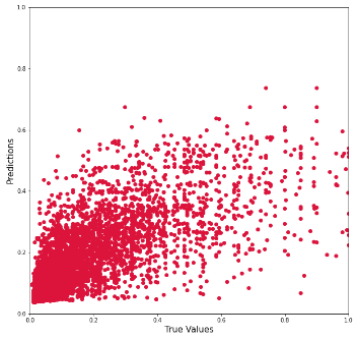


Figure A4 – Predictions vs true price of the default XGBoost

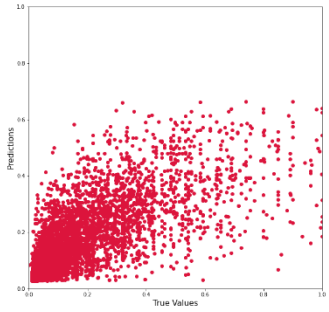


Figure A5– Predictions vs true price of the grid search XGBoost

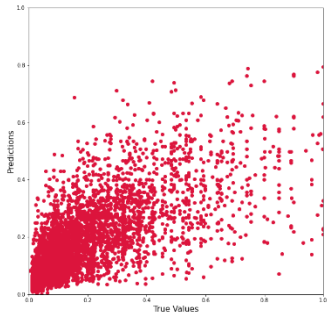


Figure A6 – Predictions vs true price of the methodical XGBoost

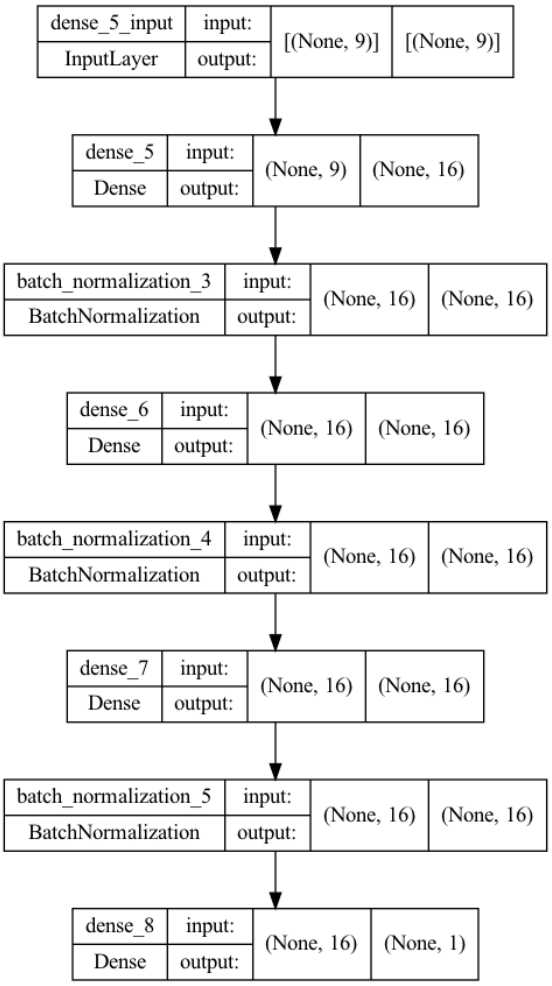


Figure A7 – Initial MLP architecture

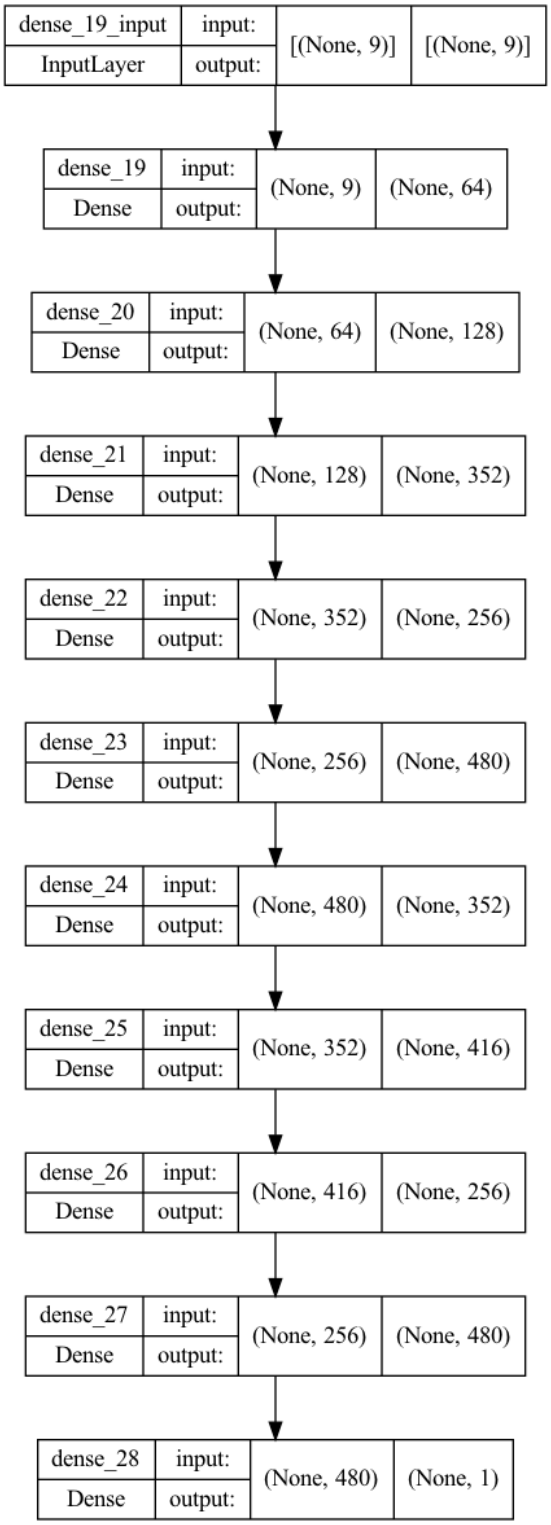


Figure A8 – Tuned MLP architecture

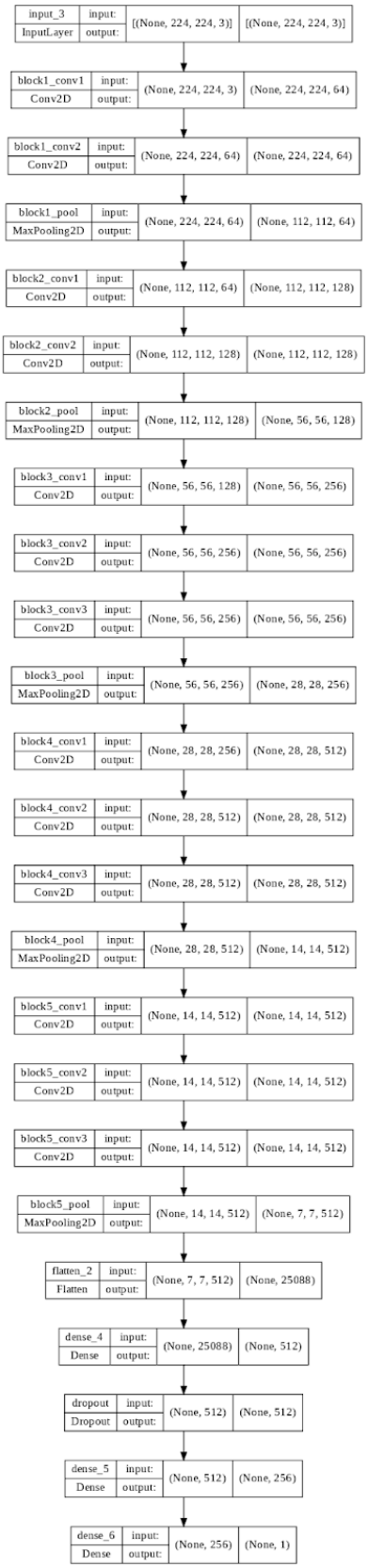


Figure A9 – VGG16 architecture

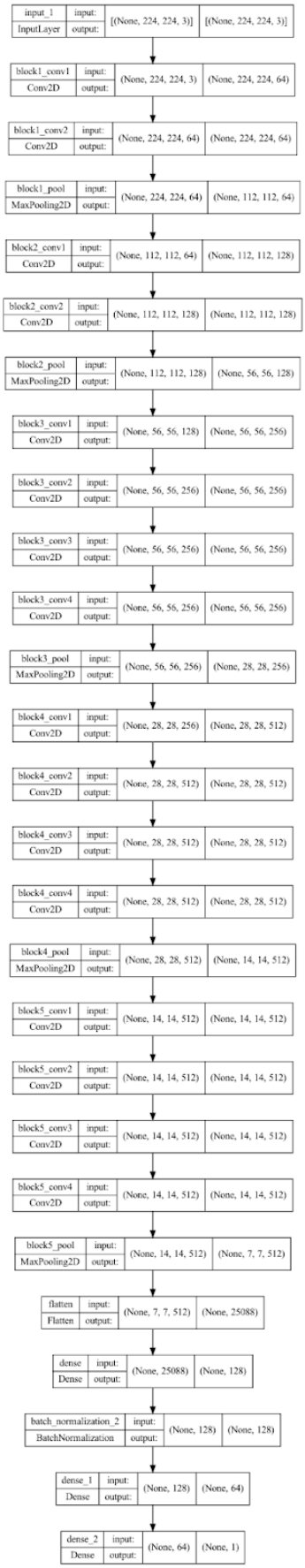


Figure A10 – VGG19 architecture

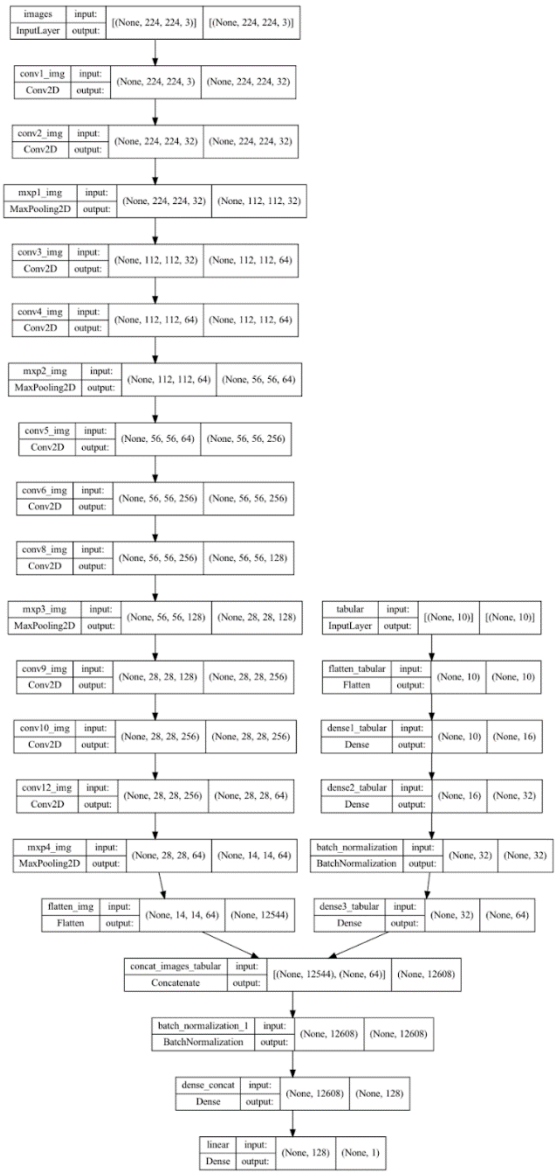


Figure A11 – Combined architecture

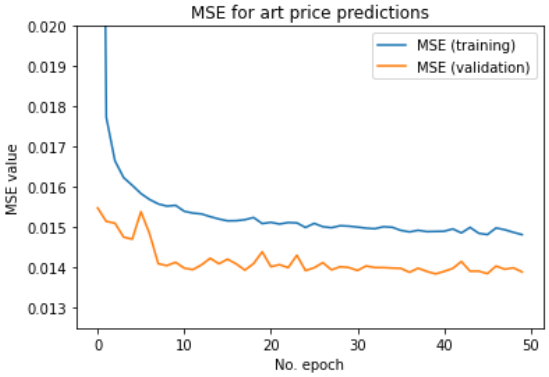


Figure A12 – MSE Initial MLP

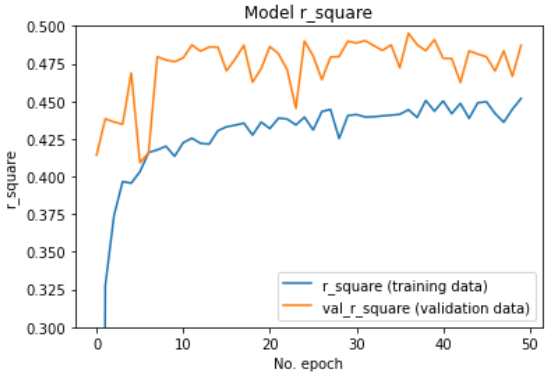


Figure A13 – MSE Initial MLP

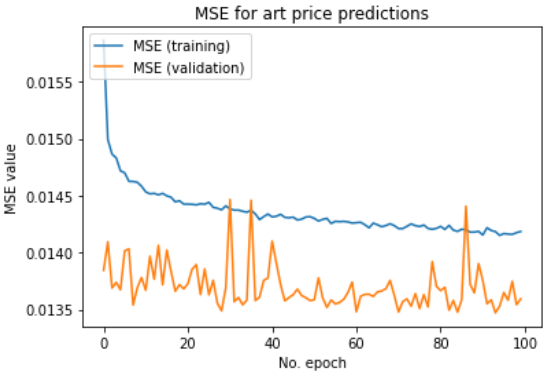


Figure A14 – MSE Grid search MLP

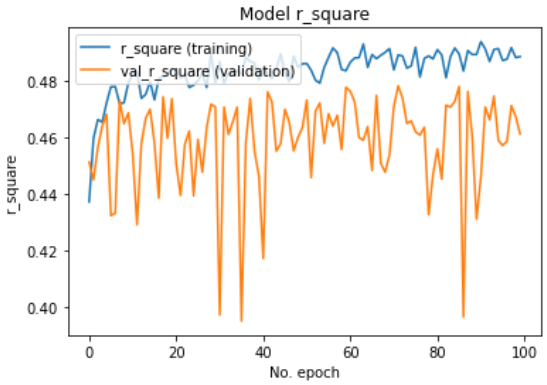


Figure A15 – R2 MLP

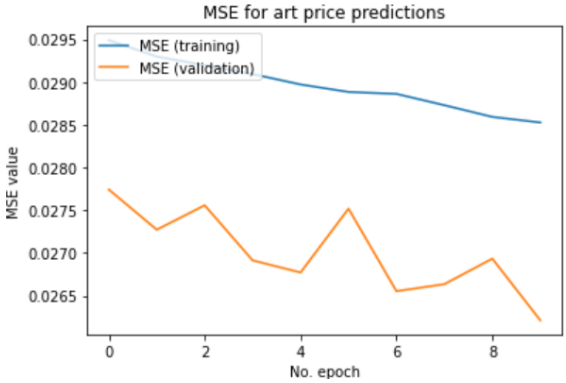


Figure A16 – MSE VGG16

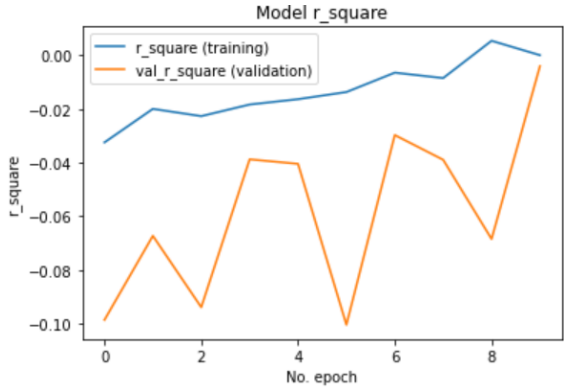


Figure A17 – R2 VGG16

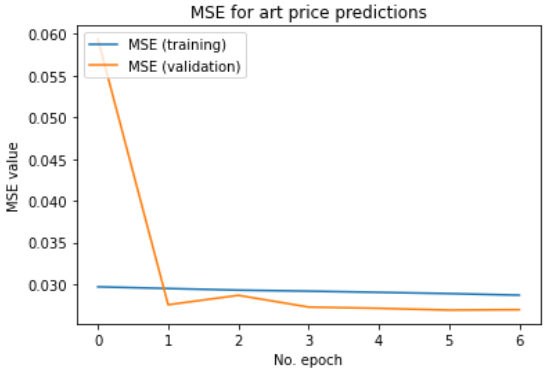


Figure A18 – MSE VGG19

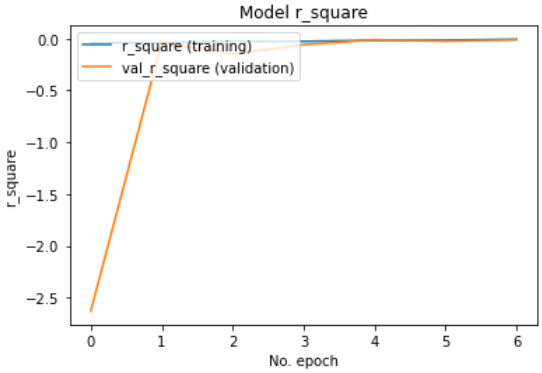


Figure A19 – R2 VGG19

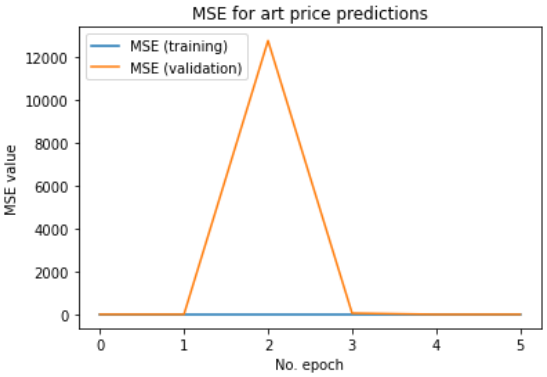


Figure A20 – MSE ConvNet & MLP

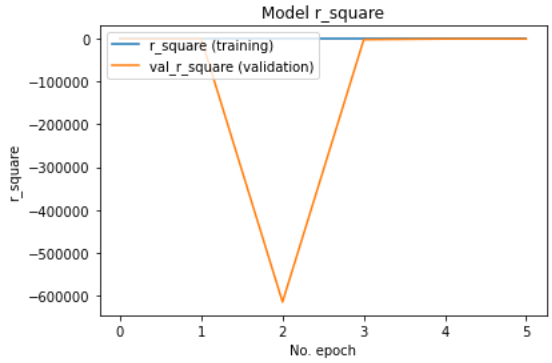


Figure A21 – R2 ConvNet & MLP