Popularity Prediction of Pet Photos using Machine Learning

CHAN Yiu Chung (20430665)
HKUST

LAM Chun Ting Jeff (12222973)
HKUST

LI Yilin (20829426)
HKUST

QIU Ruxin (20783622)
HKUST

TO Cheuk Lam (20264793)
HKUST

Abstract

The appeal of a picture can mean the difference between life and death for stray animals. In this project, we build a predictive model to precisely judge a pet photo's popularity score so as to shed light on what constitutes a good picture. Our data comes from PetFinder.my - Malaysia's top animal welfare website and contains 9,912 different pet profiles including both pet photos and metadata. Our model mainly consists of three parts: 1) a pre-trained part where we leverage transfer learning to extract features and then classical machine learning methods to predict the scores; 2) a neural network part where we concatenate features from both the photos and metadata and then passed through fully connected layers for score predictions; 3) and a stacking layer to smartly combine models from 1) and 2) for the final prediction. We hope that this model can provide reliable advice for shelters and rescuers around the world and ultimately more lives are saved and more happy families created.

1 Introduction

Every day, millions of stray animals throughout the world suffer on the streets or are euthanized in shelters. When it comes to stray animal adoption, a picture is worth a thousand words and can even mean the difference between life and death. Pets with appealing images tend to receive more attention and therefore are generally adopted more quickly. But what constitutes an excellent photograph? In this project, we would like to build a predictive model to precisely judge a pet photo's attractiveness and even recommend adjustments to give these rescued animals a better chance of finding loving homes with the help of data science.

PetFinder.my is Malaysia's most popular animal welfare website, and it currently ranks pet images using a simple Cuteness Meter. It compares the performance of hundreds of pet profiles to picture design and other parameters. We use raw photographs and metadata to forecast the "Pawpularity" of pet photos in this project. We train and test our model on PetFinder.my's millions of pet profiles in hope that it will provide reliable advice to help animals live happier lives.

2 Dataset

2.1 Data Description

The dataset comes from the Kaggle competition held by PetFinder.my. It contains 9912 raw images extracted from the profiles of pets, manually-labeled photo metadata indicating the key visual quality and composition parameters of the pet pictures, as well as the “pawpularity score”.

Each image is assigned a unique ID, which is the same as the Pet Profile ID. There are a total of 12 types of photo metadata such as “Human” and “Action”, which indicates whether humans are included and whether pets are in actions. All the photo metadata are categorical binary values, which can only be 1 or 0.

The range of the pawpularity score is from 0 to 100. It is calculated from the page view statistics of each pet profile, using an algorithm that normalizes traffic data across different pages and platforms (web & mobile). During the calculations, mistakes from duplicate clicks and crawler bot access are removed.

2.2 Exploratory Data Analysis

Pawpularity Score Distribution. The overall distribution of Pawpularity score is a right-skewed bell that focuses around the 20-50 range. However, we noticed that a substantial amount of pictures scored 100. These outliers could pose quite a challenge to our model.

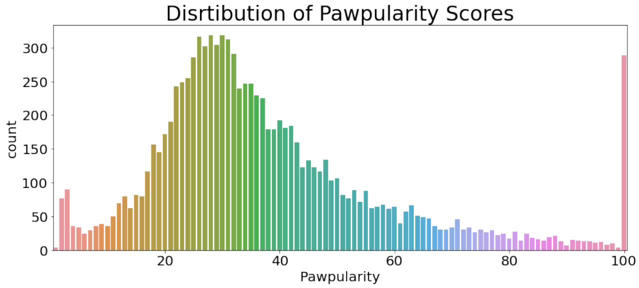

Figure 1: distribution of pawpularity scores (histogram)

We have some assumptions for the occurrence of the outliers. First, there is a particular breed of animal or a particular way of photo taking that is most desirable by the people who visit the site. In Figure 2, we sampled some pictures from different score ranges where red means lower scores and green means higher scores. It seems that there is not any particular pet type or photo taking skills in the outliers.

Another assumption is that, from the description of dataset, we can know that the Pawpularity Score is derived from each pet profile's page view statistics. It is possible there are some internet celebrities or KOL on the site so that their pet images attract more attention. And the web designer may put some pet pictures on the front page based on the images' popularity, therefore, these KOL's images further get exposed, resulting in 100 pawpularity score. Indeed, these outliers comprise around 3% of the dataset, which is in line with our assumption of KOL.

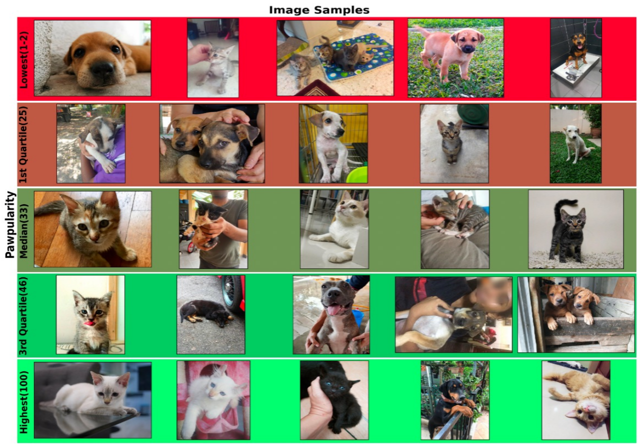

Figure 2: Examples images of some pawpularity scores

Correlation Matrix. To gain a rough idea of the metadata and their correlation we plotted the correlation matrix heat map. The correlated features groups such as eyes and face, human, and occlusion make common sense but provide little information. We can also see that there is no obvious correlation between the pawpularity score and the single feature of the image.

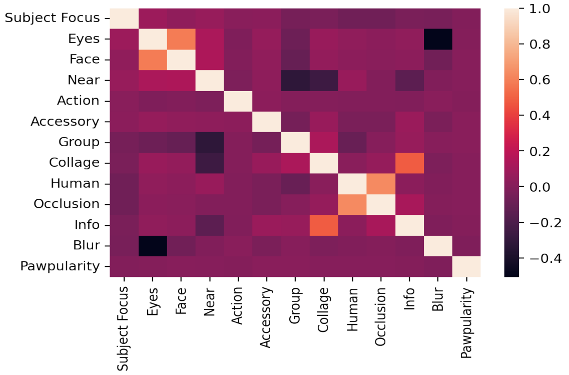

Figure 3: Metadata Correlation Matrix

3 Methodology

There are mainly three parts in our model.

Transfer Learning + Classical Machine Learning. Transfer learning is a popular machine learning method in deep learning, where we use a pre-trained model on a specific task as the starting point for other tasks. In the computer vision community, a pre-trained model often have learnt to extract meaningful features from the images based on training on large-scale image-classification tasks like ImageNet 1K.

For our project, we begin with transfer learning to exploit the already captured knowledge in different pre-trained models. We simply use the pre-trained models as feature extractors to convert the image data to a feature vector, which can save vast compute and time resources in contrast to finetuning the models one by one, and then classical machine learning methods are applied to fit the data. We exploit the timm library to get those pre-trained models based on ImageNet. Besides extracting features from the original image, a horizontally-flipped and slightly-cropped version of the image is exploited to extract an extra feature vector.

Another kind of pre-trained model, CLIP [5] , is also adopted to extract features from the images. Compared with those ImageNet-based models, CLIP is based on **C**ontrastive **L**anguage–**I**mage **P**re-training, with an attempt to find the underlying linkage of image and text. It tries to construct a latent space where images and text can be aligned. For example, an image of a cat should have a high similarity with the sentence "a photo of cat" in CLIP. Therefore, the features extracted by the image encoder in CLIP would give another view of the image.

For the classical ML techniques, support vector regression (SVR), random forest (RF), Catboost, LightGBM (LGB) are used to fit the data. These machine learning methods are

Pre-training Type	Family	Detail Models
Pre-trained On ImageNet 1K	Residual neural network	resnetv2_152, resnext_101*
	Distilled data-efficient Image Transformer (DeiT)	deit_base*
	Efficientnet	efficientnet_l2*
	Class-Attention in Image Transformers (cait)	cait_m36 *
Pre-trained On Image-Text Pair	CLIP	RN50x4, RN50x16, RN50x64, ViT-B/32, ViT-B/16, ViT-L/14

* means that besides extracting with normal image, flipped version of the images are also exploited to extract an extra feature vector

Table 1: Transfer Learning models

robust and powerful compared with other methods, which should be able to handle most cases.

Neural Network. In this part, neural networks are trained based on the image and the categorical features. We use three state-of-the-art neural networks in computer vision, Swin Transformer[3], ConvNeXt[4] and BEiT[1]. Among these chosen architectures, Swin Transformer and BEiT are transformer-based while ConvNeXt only exploits the conventional convolution layers. Swin Transformer is a powerful Vision Transformer that can construct hierarchical feature maps with **S**hifted **win**dows. In Swin Transformer, the self-attention process is carried out on non-overlapping local windows and a shifted windowing scheme is leveraged to allow for cross-window connection. BEiT, stands for **B**idirectional **E**ncoder representation from **I**mage **T**ransformers is a computer vision version of BERT, which is a famous model natural language processing area. It is pre-trained on a masked image modeling task, and after pre-training, it can be fine-tuned for different downstream tasks. ConvNeXt is a "modernized" version of the convolutional neural network, which modifies the ResNet architecture based on the recent advances in deep learning architecture like vision transformer. It is a pure convolutional neural network but is competitive with transformer-based models in multiple tasks such as classification and segmentation.

To include the categorical features, an embedding layer is set up for each feature. For each categorical value, we represent it with a 4-dimension embedding vector. Those embedding features are concatenated with the image features extracted by the backbone and then pass through fully-connected layers for prediction.

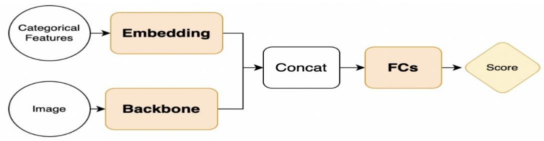

Figure 4: neural network structure

Stacking. Stacking is a well-known ensemble machine learning algorithm, which exploits a meta-learning algorithm, i.e., ridge regression, to learn the best combinations of the predictions from multiple base machine learning algorithms. In our task, stacking is adopted in order to learn how to best combine the predictions from the transfer learning and conventional machine learning methods, and the neural networks. We choose the ridge regression as the meta-learning algorithm to stack the predictions together. Since we have a number of predictions, i.e., more than 60, rather than using all of them for stacking, we use hill climbing based on the cross-validation score to select the best subset of predictions for stacking.

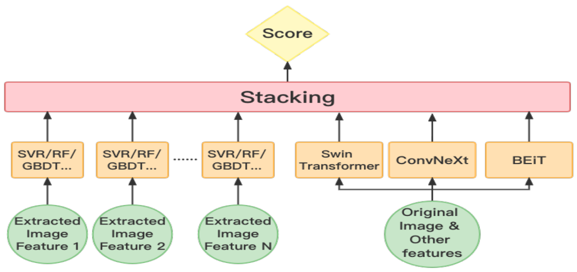
Score
Stacking
SVR/RF/ GBDT...
SVR/RF/ GBDT...
......
SVR/RF/ GBDT...
Swin Transformer
ConvNeXt
BEiT
Extracted Image Feature 1
Extracted Image Feature 2
Extracted Image Feature N
Original Image & Other features

Figure 5: model structure

4 Experiments

To test our models, we create a split of 90%/10% of the data to train/test. A stratified 10-fold cross-validation is performed based on the pawpularity scores such that in every fold, the distribution of the scores is similar. During training, the out-of-fold prediction is used for validation.

4.1 Evaluation Metric

Root Mean Square Error (RMSE) is used as the evaluation metric for this regression problem. The lower, the better.

4.2 Transfer Learning + Classical Machine Learning

Setting. After extracting the features from the images, the feature vectors are fed as the input for the traditional machine learning model, and output with the pawpularity score. We

Model	SVR		RF		Catboost		LightGBM	
	val	test	val	test	val	test	val	test
ResNetV2[2]	18.247	18.051	18.243	18.051	18.249	18.058	18.244	18.056
ResNext[9]+Flipped	17.791	17.672	17.792	17.678	17.796	17.672	17.798	17.671
Efficientnet[6]+Flipped	17.729	17.613	17.728	17.616	17.723	17.614	17.72	17.612
DeiT[8]	17.879	17.655	17.873	17.658	17.877	17.652	17.872	17.652
cait[7]	17.755	17.705	17.756	17.707	17.756	17.701	17.756	17.703
CLIP(RN50x64)	17.643	17.558	17.646	17.554	17.646	17.553	17.646	17.558

Table 2: Best combination result of each family model

have 15 types of feature vectors from the pre-trained models and 4 types of machine learning methods. Overall, there are 60 combinations.

Result. Table 2 summarizes the performance of all combinations. We have several findings. First, the combination of CLIP model (RN50x64), a vision-language model, with any traditional machine learning method outperformed all other combinations. Regarding the performance of RMSE on pre-trained models based on ImageNet1K classification, the best model is Efficientnet. Also, the result shows that among different traditional machine learning methods, there is no big difference in final RMSE. Rather we notice that pre-trained models for feature extraction contribute a large proportion to the final result.

What is a good feature vector? Often, when a pre-trained classification model is used as a feature extractor, the output on the internal embedding layer (the layer before the linear classification head) will be used as the feature vector. However, in our task, when exploiting the pre-trained models based on 1000 classes Imagenet images, we found that extracting the features from the top 1k linear classification head performs better, which can decrease the RMSE by around 0.1. We hypothesize that since Imagenet 1k has some classes of breeds of dogs and cats, the probabilities of classifying the images as dogs and cats contain more high-level information such as semantic information about whether an image is a good photo of animals. We have done more investigation on the classifier layer. In Figure 6, we show the number of images in each classes of ImageNet1K. We can see that the pet images are always classified to some kinds of cats and dogs. And from Figure 7, we can see some classes like golden retriever have a higher average pawpularity score.

We think this interesting finding somehow explain why CLIP outperforms others. CLIP extracts more high-level and robust features than the classification models since it links an image to text description, which are more similar to human perception. For example, if we are given a cat image, the feature vector from CLIP may tell us this image is highly related to a text description, " an image of a siamese cat", while the feature vector from classification models only tell us whether there are ears and eyes in the image.

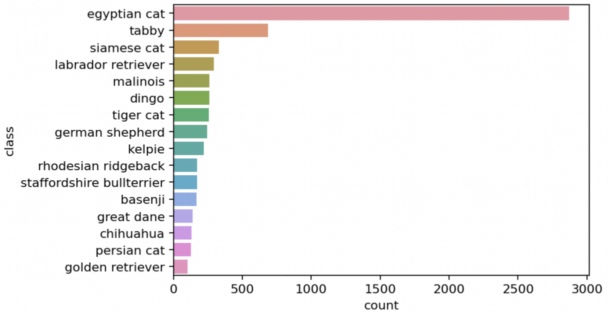

Figure 6: Number of images in each classes of ImageNet1K by Resnet. Only the top 16 classes are shown here.

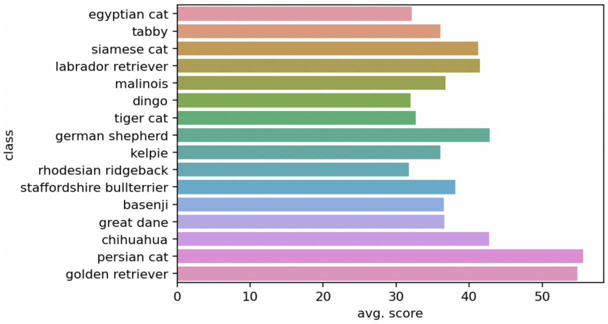

Figure 7: Average pawpularity score in each classes of ImageNet1K.

4.3 Neural Network

Setting. Overall, we finetune 3 models for the prediction, Swin Transformer, BEiT, and ConvNeXt. We use the AdamW optimizer with a training rate of 4e-5 to train the networks for 10 epochs. For the learning rate scheduling, we leverage a linear warmup with linear decay. Early stopping is adopted based on the validation loss to prevent overfitting. In addition, we perform some data augmentations such as color jittering, and random horizontal and vertical flips.

Result. Table 3 summarizes the performance of the neural network. Among the three neural networks, Swin Transformer is the best in terms of validation RMSE, while BEiT is the

Without features With features	Val RMSE	Test RMSE
Swin Transformer	17.87 17.76	17.54 17.45
BEiT	17.95 17.87	17.56 17.43
ConvNeXt	18.22 18.12	17.79 17.70

Table 3: Result of neural net

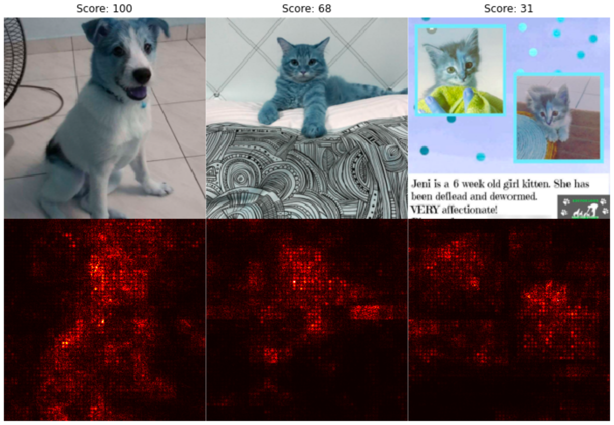
Score: 100
Score: 68
Score: 31
Jeni is a 6 week old girl kitten. She has been deflead and dewormed. VERY affectionate!

Figure 8: Examples of Saliency Map of Swin Transformer. The neural network generates the prediction mainly based on the pet but has put some attention to the background.

best in terms of test RMSE. Also, we found that, regarding the validation RMSE, the result of the neural networks is surprisingly worse than CLIP in transfer learning + ML. One possible reason is that the neural network suffered from some kind of overfitting. Since if we look at the test RMSE, Swin Transformer and BEiT get lower errors than all methods in transfer learning, which indicates the capabilities of deep learning. Another finding is that, the CNN-based ConvNeXt performs worse than the other two transformer-based networks, its performance is even worse than some methods in transfer learning + ML. This is quite weird since we expect it should be competitive with the other two models according to the original paper of ConvNeXt [4]. Since neural networks work like a black box, we do not have any good idea. Our guess is that the attention mechanism in transformer-based models can better capture the global information of the image than CNN.

Are the categorical features useful? In our neural network architecture, we add an additional embedding layer to include those manually labeled features into the neural network but it is questionable whether these features help boost the performance since those features, such as whether there is occlusion and whether the pet faces are clear, all can be figured out from the image. Theoretically, if the neural networks are powerful enough, they should be able to learn to extract these features from the image for prediction.

In Table 3, we study the effect of including those features. Without the categorical feature, the test RMSE slightly increases around 0.1 for all models. We believe that, even though the neural networks are powerful, they still cannot directly learn to extract and exploit these specific features, and thus the manually labeled features are useful, they may work like a hint for the network. Also, the embedding layer only slightly increases the time in training and inference, it is not a big issue to retain it in our network architecture.

Is test time augmentation useful? Test time augmentation is a popular technique to boost performance. During inference, we can also apply image data augmentation to the test dataset, so that we allow the model to make predictions based on different views of the test image data. The multiple predictions of the augmented image are eventually averaged, which can enhance the predictive performance. After finishing the training of the neural network, we have tried to perform test time augmentation, i.e. averaging the predicted scores of the original image, the horizontally-flipped image, and the vertically-flipped image. This method improves the RMSE of the models by around 0.007. Since the improvement is negligible and it requires extra time to carry out, we abandon it.

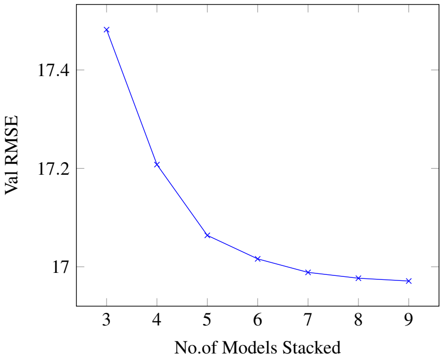

Figure 9: Performance along the hill climbing process

4.4 Stacking

The main results are presented in Table 4. When we simply stack all the predictions together with ridge regression, the val RMSE is 17.05 and the test RMSE is 16.98. We also carry out hill climbing to select the best combination of model predictions. Our final combination includes nine models, BEiT, ConvNeXt, Swin Transformer, CLIP(RN50x6) + SVR, Efficientnet + Catboost, CLIP(ViT-L/14) + SVR, ResNeXt + LGB, CLIP(ViT-B/32) + SVR, ResNeXt + LGB. In comparison to stacking all predictions, the val RMSE and test RMSE are decreased to 16.97 and 16.89. It is believed that the model selection process helps to alleviate overfitting by preventing the combinations to be too complicated. This is proved in Figure 9, we begin the search with the three predictions from the neural network. After combining nine models, we observe that validation RMSE cannot be further decreased and starts to rise.

	Val RMSE	Test RMSE
Stack All	17.05	16.98
Stack with Hill Climbing	16.97	16.89
Stack only transfer learning	17.43	17.32
Stack only neural nets	17.48	17.21

Table 4: Result of Stacking

Ablation Studies. We have carried out ablation studies to investigate the necessities of different components in our entire architecture. We have tried stacking either without the transfer learning + ML or without the neural network. The result is summarized in Table 4. Either stacking the transfer learning + ML predictions or the neural network predictions only, the val RMSE and test RMSE are both higher than the hill climbing model, which implies that both the transfer learning part and the neural network part are essential. In the hill climbing model, we can see that three CLIP model predictions are included, therefore, the pre-trained models used in transfer learning, especially CLIP that are pre-trained on image-text pairs, are able to figure out some features in the images that are not captured by the tuned neural networks.

4.5 Outliers

As we know from Figure 1, there are a substantial amount of pictures with 100 scores which makes the data very imbalanced. If we exclude those outliers, RMSE becomes 15.57. Since time is limited, we did not retrain our model without outliers and then test the updated RMSE. One intuitive idea is that we can first train a classifier to separate these outliers from the rest of the data. This would be an interesting topic to explore in the future.

5 Future Work

In our current approaches, the model only tells the appealing of the pet image but cannot explicitly tell the user how to improve the image taking. Also, since we include neural networks, more work can be done in the future on explainable AI, so that we can know which factors contribute most to the final output.

6 Conclusion

Through this project, a machine learning predicting approach is designed, implemented, and applied to give credible predictions to estimate the pawpularity of the pet images. The result is comparable to the top 5 score in the Kaggle public leader board. We do not submit our method since the competition has strict restriction to internet access and running time.

We hope that with help of this predictive model, shelters and rescuers around the globe can easily boost the appeal of their pet profiles by automatically improving photo quality and advising composition changes. In turn, many precious lives could be saved and more happy families can welcome their beloved family members.

Acknowledgements

Chan, Lam, and Li worked on most coding and part of report. Qiu and To worked on some interesting experiments, report and presentation.

References

[1] Hangbo Bao, Li Dong, and Furu Wei. “BEiT: BERT Pre-Training of Image Transformers”. In: *arXiv e-prints*, arXiv:2106.08254 (June 2021), arXiv:2106.08254. arXiv: 2106.08254 [cs.CV].

[2] Kaiming He et al. “Identity Mappings in Deep Residual Networks”. In: *CoRR* abs/1603.05027 (2016). arXiv: 1603.05027. URL: http://arxiv.org/abs/1603.05027.

[3] Ze Liu et al. “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows”. In: *arXiv e-prints*, arXiv:2103.14030 (Mar. 2021), arXiv:2103.14030. arXiv: 2103.14030 [cs.CV].

[4] Zhuang Liu et al. “A ConvNet for the 2020s”. In: *arXiv e-prints*, arXiv:2201.03545 (Jan. 2022), arXiv:2201.03545. arXiv: 2201.03545 [cs.CV].

[5] Alec Radford et al. “Learning Transferable Visual Models From Natural Language Supervision”. In: *CoRR* abs/2103.00020 (2021).

[6] Mingxing Tan and Quoc V. Le. “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks”. In: *CoRR* abs/1905.11946 (2019).

[7] Hugo Touvron et al. “Going deeper with Image Transformers”. In: *CoRR* abs/2103.17239 (2021).

[8] Hugo Touvron et al. “Training data-efficient image transformers & distillation through attention”. In: *CoRR* abs/2012.12877 (2020).

[9] Saining Xie et al. “Aggregated Residual Transformations for Deep Neural Networks”. In: *CoRR* abs/1611.05431 (2016). arXiv: 1611.05431. URL: http://arxiv.org/abs/1611.05431.