

AI-cyclopedia: A look into fascinating art and understanding why they are what they are

Shaoyang Luo
Stanford University
Palo Alto CA
sluo3@stanford.edu

Roman Puttkammer
Stanford University
Palo Alto CA
rputtkam@stanford.edu

Kana Sethu
Stanford University
Palo Alto CA
ksethu@stanford.edu

Abstract

In this paper we build an application using a Masked Autoencoder Vision Transformer (MAE-ViT) to predict various attributes of a piece of art. We use a Tree Augmented Naive Bayesian network to boost model accuracy by leveraging correlation between attributes in the training set. Finally, we attempt to use various saliency techniques to provide a cogent explanation for why those attributes were picked. In addition, we compare the results from the MAE-ViT to other approaches based on classic Convolutional Neural Network (CNNs) architectures and EfficientNET. The ViT performed best among the models though its accuracy (measured in PR-AUC and F2) was limited. The reasons likely relate to the nature of the dataset as we will discuss below. To capture the correlation among model class scores, we exploited Bayesian Network variation and adopted Tree Augmented Naive Bayes network. This additional layer of screening further improved the F2 score by 0.13. Given the similarity of styles for identifiable characteristics we were not able to generate relevant saliency maps for the test set.

We use a curated dataset provided by the Metropolitan Museum of Art and Art Institute of Chicago for the FGVCx competition which is hosted as part of the FGVC8 workshop at CVPR 2021 [1].

1. Introduction

We have a plethora of AI models that tell you what it thinks. To a layman this is nothing more than *just an opinion*. Opinions are, after all, for free. But an opinion that is based on rationality, on facts, on data, is truly priceless.

The Metropolitan Museum of Art in New York, also known as The Met, has a diverse collection of over 1.5M objects of which over 200K have been digitized with imagery. The digitized artifacts have been diligently annotated by subject matter experts and includes a wide range of data. These include, but are not limited to: multiple object clas-

sifications, artist, title, period, date, medium, culture, size, provenance, geographic location, and other related museum objects within The Met’s collection. While the annotations describe the object from an art history perspective, they can also be indirect in describing finer-grained attributes for the museum-goer’s understanding. Adding fine-grained attributes aids in the visual understanding of the museum objects and enables the ability to search for visually related objects.

We use this dataset as input to a Vision Transformer to perform multi-label classifier. One of the challenges is to ensure that the model learns useful representations of the images that correlate with the assigned attributes; shapes and colors relating to i.e. origin and era. A masked auto encoder as proposed by [9] attempts to build suitable latent representation by being trained on cropped/masked portions of the images, forcing it to learn and identify classes based on partial images. It relies on an initial, self-supervised learning step after which the decoder is discarded the the final model is build based on the encoder’s latent representations.

The number labels for each image on average is about 6 but can go as high as 20. There are a 2117 unique attributes that can be predicted. Many of these attributes are correlated. We use a Tree Augmented Naive Bayes network to assess whether the attributes are compatible with each other, using conditional probability of one attribute value given the rest.

Most importantly, this project attempts to go beyond a typical classification task of predicting attributes to adding an “explainable AI” component, by showing which parts of an image resulted in predicting a particular attribute. Adding a visual explanation of why a particular object was tagged with a certain attribute, goes a long way in explaining to an average museum-goer the general aesthetics, resources, and means of cultures from disparate time periods.

2. Related Work

We investigate how to adjust the transformer to multi-label classification, something reviewed in [11] albeit in a different setting.

Bayesian method is well adopted in the field studying conditioned inferences. Among the method variations, Naive Bayes (NB) and Tree Augmented Naive Bayes (TANB) were the top ones to consider for their implementation flexibility. Considering the fact that all attributes could be inter-correlated, we decided to experiment more on TANB even though the additional dependencies would greatly increase computing costs [15]. As we were exploring potential methods to solve attribute correlations, we also considered using factor graphs as non-directional-graph network, as opposed to a directional-graph from Bayesian Networks. However, calibrating a factor graph with Belief Propagation would take exponential run time, which is formidable when we have to calibrate potentially 2^{2117} graphs in learning. Thus this structure is left unexploited [14].

Image specific class saliency methods were first introduced in [21], [18] (SmoothGrad) and [16] (Gradient*Input). Guided backprop was first introduced by [17] which suggested using the concept of an ‘imputed’ gradient rather than the true gradient to amplify the ability to discern pixel importance. We had planned to use the standardized method for comparing saliency maps proposed in [10]. The ROAR metric is based on the notion if a saliency map is an accurate representation of what is important in an image, then if we block out salient regions, network performance should degrade. Conversely, if we block out non-salient regions, we should see little degradation in performance. The metric measures these degradations explicitly. [13] propose a method of generating explainable AI saliency maps which claims to be more efficient than the standard methods such as Guided Backprop and other class activation methods. Efficiency is important to us because we envision our application being available via a museum’s iPhone app. Gradient methods hypothetically require a single backward pass but in reality require a lot more to give an accurate rendering. While this is feasible in a lab setting with unlimited access to GPUs, but not quite on an iPhone in real time. The method measures the information at the end of each network scale and combines it into *one* single Saliency map by using a technique called Saliency Map Order Equivalence (SMOE) which takes advantage of the fact that informative activations have a sparse pattern with strong peaks.

Aside of saliency methods used for convolutional networks, different methods have been proposed for convolutional networks (see [12]) something that maybe looked at in future work.

3. Dataset and Features

The IMET2021FGVC8 dataset contains a train set of 145K images and a test set of 53K images. The dataset defines 3475 different *attributes* which has the format "*AttributeCategory::AttributeCategoryType*". *AttributeCategory* fall under five broad sections [Figure 1a]:

- Country: includes 100 different countries
- Culture: includes 961 types of culture
- Dimension: includes 5 different dimensions: *large, medium, small, tiny, very large*
- Medium: 1920 types of mediums, varying from simple ones like *clay* and *wool* to exotic ones like *vitreous paint* and *silver stain*
- Tags: this includes 768 descriptive terms like *bible, trumpets, politics*, etc.

Images are of varying sizes. Each image is assigned at least one attribute [Figure 1b]. Most images have between 1-6 attributes [Figure 1c], although a few images have 20+ attributes!

Data Preprocessing Each image was processed in the same way as proposed by [2] and used in their model, with the same pipeline being leveraged across all models. For training, images are randomly cropped (20% or larger) and resized to 224x224, followed by a random horizontal flip, a random gaussian blur with a 5x5 kernel and sigma in the range of 0.1 to 1. Finally images are normalized using mean/stddev of 0.5/0.5. For the validation/test set the images are center-cropped to 224x224 and normalized in the same way. The blur was added to the pipeline to reduce the likelihood that specific background and lightning patterns would bias the classifier. Artifacts with similar origin and type may have been photographed in the same location and with the same background and lightning. While this would benefit a pure classification task that uses a subset of the same data as test set, it would reduce the likelihood of the model to actually pick up on image attributes that a human observer would focus on, such as patterns, colors and textures.

4. Method

We’ve broken up our project into three parts: multi-label classification, Bayesian network to improve prediction accuracy and saliency.

4.1. Model Evaluation Metrics

Of the multitude of metrics available to evaluate multi-label models, we decided the F2 score - although selected by the kaggle competition - as most suitable. F2 is a better

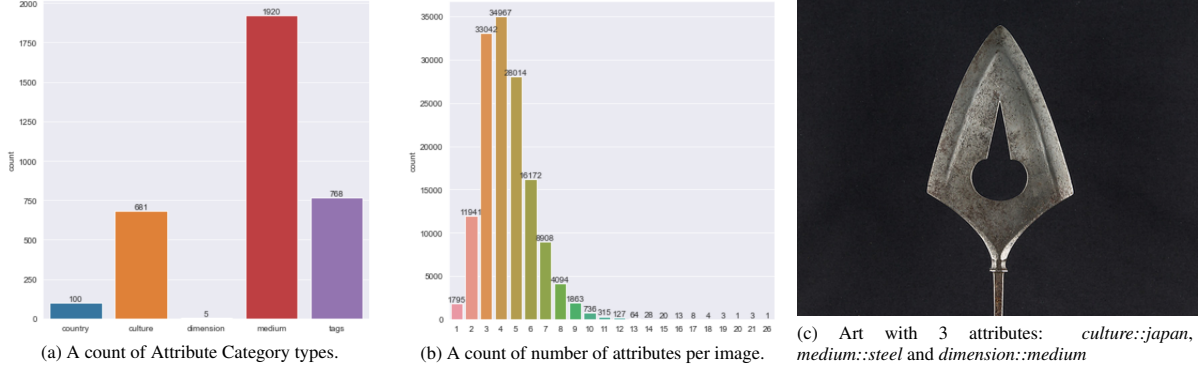


Figure 1. Data set features

score for highly imbalanced data sets as plain accuracy and F1 will place too much emphasis on false positives, leading to few if any positive labels being identified. However in addition to not being differentiable, the F2 score further depends on the selection of the decision threshold, making it a less than ideal choice for evaluating a model’s overall performance. A better choice is the precision/recall AUC which provides a metric that measures performance across all thresholds. Like F2, PR-AUC is not differentiable and cannot directly be used in a model relying on gradient descent.

Therefore we decided on following a standard approach:

1. Train the model on binary cross entropy loss.
2. Select best performing model using the PR-AUC metric, on the validation set.
3. Optimize performance by choosing a per-label decision threshold that maximizes the F2 score on the train and validation set.

We consider three ways to calculate PR-AUC and F scores; **macro**, **micro** and **weighted**. For the macro score, individual label scores are calculated independently and averaged. For the weighted score, label scores are also independently calculated but then averaged with weights relative to the positive support for each label. Finally for the micro scores the overall scores are calculated overall with each sample and label being considered independently. Weighted and micro scores are probably best representative of our learning approach. Interestingly, while weighted and micro scores are better suited for imbalanced problems, a macro score was chosen for the kaggle competition. (Maybe the competition hosts chose a balanced test set which would address this issue.)

4.2. Classification using Convolutional Network Architectures

In the first part we study the classification accuracy of three different models:

Model	Parameters	Pretraining
ViT	87’426’629	ImageNet-21k
Simple CNN	13’431’589	n/a
EfficientNET	6’719’425	ImageNet

Table 1. Test set results with optimal threshold

1. Simple CNN
2. EfficientNET
3. Transformer (ViTMAE)

Table 1 shows the configuration of the different models along with the number of trainable parameters.

Architecture 1: Simple CNN We chose a simple MLP with convolutional, relu and max batch layers - not unlike the one suggested in the CS231 HW - to serve as baseline. Contrary to the other two models that were pretrained, this CNN needed to be trained from scratch. The expectation was also that a simple architecture relying primarily on convolutions and without residual paths may make it easier to perform inference on what image attributes correlate with specific labels, through i.e. saliency and similar methods. Results were mixed, although the network performed comparatively well, especially considering the lack of pretraining.

Architecture 2: EfficientNET We further chose the efficientNET architecture (base b0 model, see [19]) given the good performance on ImageNET in spite of it’s relatively compact size with less than 7m trainable parameters. Its main building block is mobile inverted bottleneck MBConv and squeeze-and-excitation blocks. For performance reasons we chose the base version rather than one of the scaled networks aimed to further improve performance. Through the torch library we had access to a model pretrained on ImageNET. We applied transfer learning by replacing the classification head and by fine tuning the model on the training set.

Architecture 3: Vision Transformer (ViTMAE) In recent years the *transformer architecture* has been successfully used for vision tasks, both in conjunction but also as replacement for convolutional networks (as proposed by [6].) We'll examine the performance of the transformer both for the classification task and to analyze correlations and differences across artifact images and their attributes. A number of different changes to the transformer were proposed to improve learning and scalability, including [20], [3] and [4]. We're specifically looking at [9], a masked autoencoder that we hope may be able to provide suitable latent representations.

ViTMAE is with 87M parameters the largest model we used. There are two training methods possible, either from scratch or by transfer learning from an existing model: (I) unsupervised pre-training of the auto-encoder (or encoder/decoder pair) on the training images followed by supervised fine-tuning against the labels. After spending significant time on the pre-training, we decided to forgo that step and focus on supervised fine-tuning only. Even fine-tuning alone provided us with better performance compared to both other models.

Model Loss & Classification Head Both pre-trained architectures needed to be adjusted to the new data set for multilabel classification with 2117 labels. This was done by dropping the existing classification head (if applicable) and adding a new fully connected layer with 2117 output before training/fine-tuning. As loss function we used the standard binary cross entropy directly on the logits. Each score is converted into a per-label probability using the sigmoid function.

Addressing Class Imbalance during Training There are primarily two methods available to address class imbalance during multilabel training: I Oversampling of underrepresented classes and II adding positive weights in the loss function. With multilabel classification applying oversampling isn't straightforward since oversampling any record will increase occurrence for multiple labels. We therefore decided to choose adding positive, per-label weights to the loss function that accounted for the overall label imbalance based on the training set.

4.3. Tree Augmented Naive Bayesian network

Each of these networks generates scores for the 2117 different attributes, among which the correlation is not leveraged. Thus we will pass the top-scored attributes into a Bayesian network to assess whether the attributes are compatible with each other, using conditional probability of one attribute value given the rest.

This implementation would adopt Tree Augmented Naive Bayesian network (TANB) [8] where each attribute is only dependent on upto one attribute other than root attribute. Based on deep net's predic-

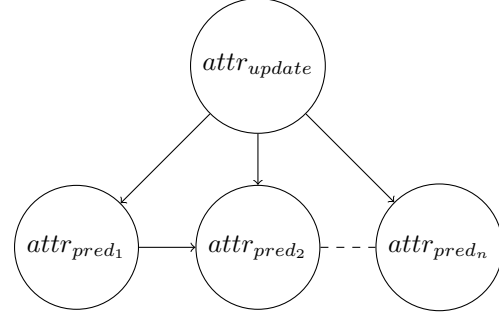


Figure 2. TANB

tion as the initialization state (binary r.v to indicate whether each prediction attribute should stay $\{attr_update, attr_pred_1, attr_pred_1, attr_pred_2, \dots, attr_pred_n\}$). We will optimize deep net's predictions with Markov Chain Monte Carlo (MCMC): For each rotated $attr_update$, we update whether the attribute should be kept based on the most likely posterior conditioned on all other selected attributes until convergence. The converged combination would then serve as the final attribute predictions. And from there we could assess the accuracies.

4.4. Model Ensemble

There are four variant ensembles to forge final predictions we experimented.

1. Vanilla approach: Picking out N attributes of top scores.
2. TopN approach: Picking out N attributes of top scores and apply TANB assessment to drop unlikely attributes
3. Threshold approach: Picking out attributes that have meet their corresponding optimal cutoffs that were trained to maximize f2 scores
4. ThresholdCapN approach: Picking out N attributes of top scores that also have met their optimal cutoffs. If there are M attributes that have met the cutoffs and $M < N$, then pick M attributes to go through TANB assessment.

4.5. Explainable AI

What is an explanation? An explanation is an interpretable description of behavior. The complex internals of Deep Neural Nets (DNN), with millions of parameters and non-linear elements, make it very difficult to explain why and how a DNN maps a particular input to an output. There are a variety of innovative methods to help improve DNN interpretability. Saliency maps are a popular class of post-hoc explanation methods aimed at achieving this goal for DNNs operating on image data. These methods, which are a subset of feature importance methods, produce estimates

of the relevance of each pixel to the classification output score, which can be displayed as a saliency map that highlights important pixels.

One of the most common ways to calculate saliency maps is to compute the gradient of the score corresponding to the correct class with respect to the pixels of the image.

$$\nabla_x F_i(x) \in \mathbb{R}^d \quad (1)$$

where the input image $x \in \mathbb{R}^d$ and F_i are the logits of the i th pixel. We implemented this in class. [Figure 3a] shows 5 images from Imagenet and [Figure 3b] shows the saliency map calculated using this technique. This technique is very simple and computational cheap. The gradients sometimes be hard to interpret if the input is noisy.

SmoothGrad overcomes issues with noisy inputs by averaging the input gradients. This is a simple but effective modification. Gaussian noise ϵ is added to N copies of the input image x . The average of the saliency of the N 'noisy' images is shown in [Figure 3c]. The saliency map generated with this method tends to be more coherent: Note that this method picks up the third bale of hay in the first image which gets lost in the vanilla gradient method.

$$\frac{1}{N} \sum_i^N \nabla_{(x+\epsilon)} F_i(x + \epsilon) \in \mathbb{R}^d \quad (2)$$

We also looked another minor modification of the vanilla gradient method, Gradient*Input, which multiplies the gradients by the input image. While his technique is mathematically equivalent to the previous two it is visually striking and easiest to interpret [Figure 3d].

$$\nabla_x F(x) \odot x \in \mathbb{R}^d \quad (3)$$

The techniques we have discussed thus far are simple, easily interpretable and, most importantly for us, computationally cheap. These techniques can be run on a phone in real-time.

Another popular class of saliency map methods involves calculating feature relevance by modifying the back propagation layers. Guided back propagation is an example of this technique. The core idea behind guided propagation is very simple: positive gradients contribute/support a prediction while negative gradients suppress a prediction. In guided back propagation we *modify* the backprop layers using different rules. Guided backprop is a popular method of generating saliency maps but it can sometimes require more than one backward pass and performs better with a GPU.

In this example for a *RELU* layer we created a guided back propagation rule to zero out negative gradients.

Activation

$$f_i^{l+1} = \text{relu}(f_i^l = \max(f_i^l, 0))$$

Backpropagation

$$R_i^l = (f_i^l > 0) \cdot R_i^{l+1}, \text{ where } R_i^{l+1} = \frac{\partial f^{out}}{\partial f_i^{l+1}}$$

Guided backpropagation

$$R_i^l = (f_i^l > 0) \cdot (R_i^{l+1} > 0) \cdot R_i^{l+1}$$

The code for this is pretty simple. We define a function that implements this rule and the register this function to be called the *RELU* layers during back propagation in the model:

Listing 1. BackProp RELU hook

```
def relu_hook_function(module, grad_in, grad_out):
    if isinstance(module, torch.nn.ReLU):
        return (torch.clamp(grad_in[0], min=0.)),
```

Listing 2. Register BackProp hook

```
for i, module in enumerate(model.modules()):
    if isinstance(module, torch.nn.ReLU):
        module.register_backward_hook(relu_hook_function)
```

Layer Relevance Propagation (LRP) is another type of modified back prop that specifies rules in every layer for how the gradients are backpropagated.

5. Results / Evaluation

5.1. Classification using Convolutional Network Architectures

The data set proved to be quite challenging; labels are extremely imbalanced. Also, labels may not be consistently applied across the data set which would make reliable classification likely much harder. For instance dimension tags apply only to 72% of the data set. As noted earlier, the models were trained with a weighted BCE multi-label loss with PR-AUC being evaluated after each epoch. Once the best model was identified, the per-label decision thresholds were selected to maximize the F2 score.

The loss curves for all models suggest reasonable learning performance. The validation and test set metrics indicate that there was no significant overfitting. The PR-AUC in table 2 summarizes overall results. Micro/weighted scores are higher as expected.

Figure 4 shows the loss curves for the three models. ViT demonstrated the best training performance and final results, reaching its best PR-AUC score. In spite of not performing pre-training with the autoencoder, this seems to demonstrate the effectiveness of the transformer architecture for vision problems. The simple CNN and the efficientNET both showed similar performance with the simple CNN slightly above the efficientNET, something that maybe explained by the simple CNN having twice the parameters of the EfficientNET. It also suggests that the task is not well suited for transfer learning, at least from an ImageNET source. This supports our findings as well as we

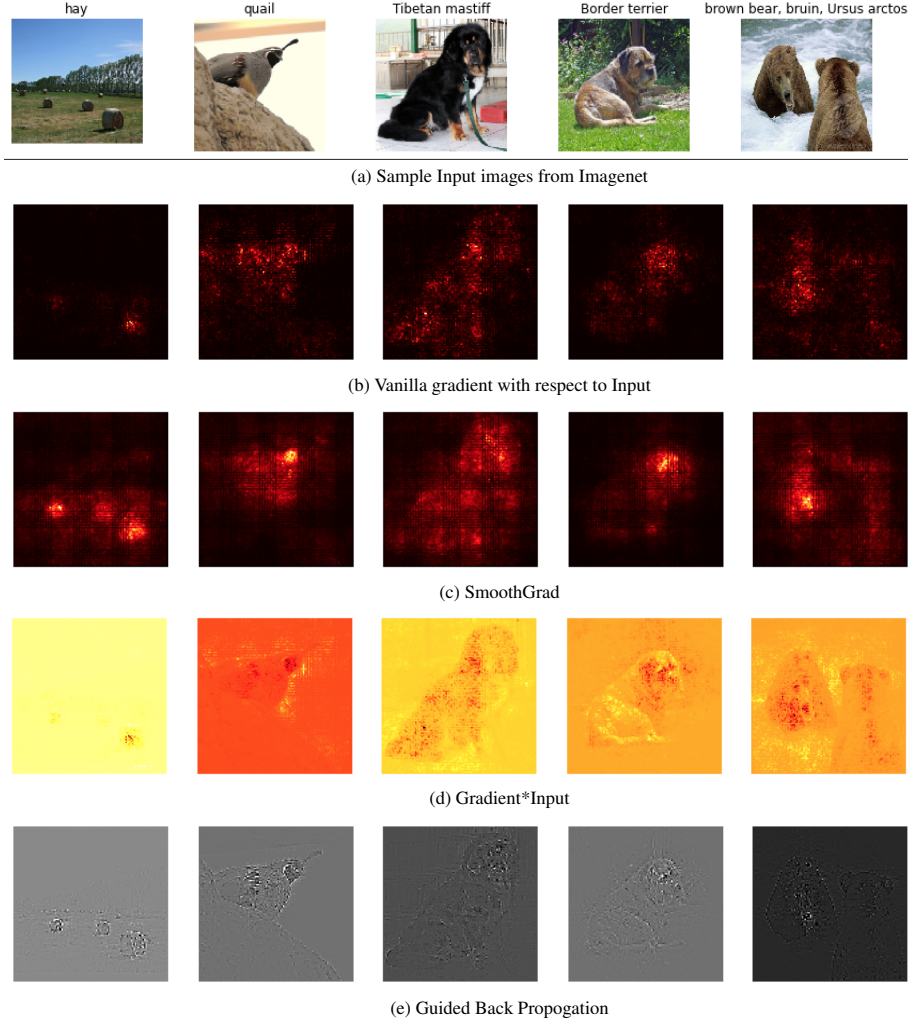


Figure 3. Different methods of calculating saliency maps, based on SqueezeNET used in HW2 [7]

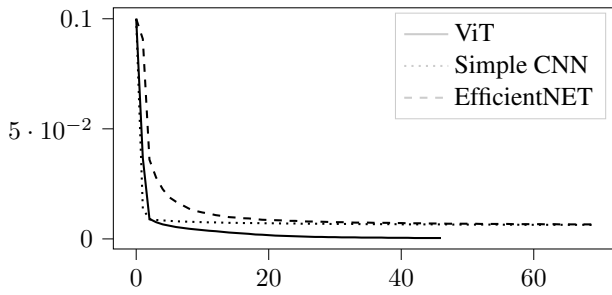


Figure 4. Training loss

evaluated saliency and related metrics - rather than on specific attributes the model seems to have used background and image styles rather than distinct objects, colors and shapes to build the classifier. We suspect this for a number of reasons:

	Train	Test	Validation
Model	Weighted PR-AUC		
ViT	.95	.59	.59
Simple CNN	.40	.40	.39
EfficientNET	.41	.37	.37
	Micro PR-AUC		
ViT	.96	.60	.60
Simple CNN	.44	.41	.41
EfficientNET	.45	.40	.39
	Macro PR-AUC		
ViT	.92	.31	.30
Simple CNN	.33	.17	.16
EfficientNET	.40	.16	.15

Table 2. PR-AUC

Model	F2	F2
	Threshold=0.5	Threshold=OPT
ViT	.23	.28
Simple CNN	.14	.15
EfficientNET	.11	.16

Table 3. Test set macro F2 score with threshold 0.5 vs optimal

Model	Precision	Recall	FPR	FNR
ViT	.45	.31	.001	.69
Simple CNN	.36	.21	.004	.79
EfficientNET	.32	.22	.004	.78

Table 4. Test set metrics with optimal threshold

- The number of positive samples for each label is relatively small with a mean/median of 180/16. That may not be sufficient for the model to learn specific shapes/-colors and other attributes correlated with the object class.
- Image labels are not consistently applied; see above example re. dimension tag applied to 72% images only, which could result in the model selecting other image attributes (background, lightning, etc) to associate with individual labels.
- Some labels relate to images features that are very localized, i.e. an image of an hourglass in a larger scene and spanning only a few percent of the image area. Notably the current preprocessing (random crop) cannot ensure each feature is retained for each sample during training.
- Some labels seem to be closely associated with specific types of artifacts - i.e. see the example of the hourglass later; if the labels were created & applied independently to separate types of artifacts, that may introduced significant bias on the training set.

The results discussed further below around saliency will further illustrate the nature of the data set and challenges that arise. The next section discusses the results from the TANB model which captures conditional probabilities. If indeed the labels in the data set were created and applied separately to different groups of artifacts, then one may expect that analyzing the conditional probabilities of the labels across images could boost the model significantly.

5.2. TANB results

The TANB model learns from the conditional probabilities from the binary sparse matrix of attributes. For every set of attribute combinations that goes under TANB assessment, the conditional probabilities needs to be re-learned because the underlying independency map changes from at-

tribute set to attribute set. Since the learning involves deriving mutual information and creating the max-spanning tree, the calculation is expensive and thus we limited number attributes going into TANB under 6 and metrics. After filtering attributes with TANB on a small subset of validation samples, it turns out TopN=4 attributes fed into TANB would provide the results of best F2 scores 0.74. The fact that TANB works better with N=4 than with N=5 or 6 is an indication of not perfect rank ordering of scores on the attributes, and a Bayesian inference becomes more important screening off false positives.

TopN Attribute	F2 Score
4	0.74
5	0.72
6	0.70

Table 5. Ensemble test set scores by TopN

Using this as the start point, we experimented Vanilla approach, TopN=4 approach, Threshold Approach, and ThresholdCapN=4 approach for deeper collaboration of deep learning scores with TANB. TopN=4 and ThresholdCapN=4 are the winning approaches. Next to that, applying optimized thresholds with respect to attribute scores would also improve identification of true attributes. Although TANB brought big lifts to the F2 scores, the pre-application of thresholds didn't further improve performance as the attribute likelihoods could already be captured by Bayesian inferences.

Approach	F2 Score
Vanilla	0.67
TopN=4	0.80
Threshold	0.73
ThresholdCapN=4	0.80

Table 6. Ensemble test scores by approach

5.3. Saliency results

We decided to start by reviewing the saliency maps using the vanilla gradient method before settling on a particular method for our application. Note that the dataset is highly imbalanced and [Figure 1a] shows that attributes are dominated by *mediums*. A quick look through the test set showed three images predicted with attribute *medium::glaze*. [Figure 5] shows these images with saliency map for this attribute. The saliency map indicates that the most relevant pixels are from background. We thought this was okay given that one typically does identify medium based the background.

The test set also showed an image, [Figure 6], has an hourglass correctly predicted with the attribute

tags::hourglasses (Hourglass visible near the top left corner of the image). However the saliency maps once again predicted the background pixels next the hourglass are more salient than pixels belonging to the hourglass. This was a bit puzzling but a quick look through the training set showed only 10 images with attribute *tag:hourglasses*. [Figure 7] shows 6 of these images. Note that all the images do have hourglasses but they are all the same style of art: black sketches on a sepia toned medium. The model, it seems, predicted attribute *tags::hourglasses* for the test image because the style was similar to training images with attribute *tags::hourglasses*.

Searches for attributes that seem to indicate distinctive shapes like *tags::sleighs* and *tags::daisies* showed that the training images looked very similar in style more than content. Generating saliency or any sort explainable features seemed futile. If we had more time we'd would have liked to take time to balance our datasets; make sure that there are diverse images for each of attributes and that all attributes are represented in a balanced manner.

6. Future Work

Considering the performance of the transformer, the choice of architecture seems to be promising in terms of classification. It would make sense to re-train the network as a masked auto-encoder to learn better representation of the image feature; this may further improve the classification with the model itself. It would also make sense to customize the pre-processing of the images as cropping of a small section carries a risk of excluding all features related to a specific label.

Another area that could be promising is *self-supervised contrastive learning*; it equally relies on self-supervised learning but compares records in latent space. A simple framework has recently been proposed in [5].

Aside of saliency methods used for convolutional networks, different *saliency methods have been proposed specifically for transformers* in [12]. These could possibly yield improved results.

There is also room to *improve the use of Bayesian Networks*; (I) More sophisticated TANB updates: Current TANB only drops attributes, and the number of attributes is severely limited by the starting number of attributes passed in. The updates would be more robust if we are not removing weak attributes, but instead replace with more appropriate attributes. The number of ending attributes would then become much more flexible. And (II) improved Bayesian learning: The current learning is based on the overall training set. But there's no optimization done to improve the training set to be a more balanced set, and could further empower TANB capabilities.

In order to generate decent saliency maps we will need to significantly augment the training set. We noticed that vi-

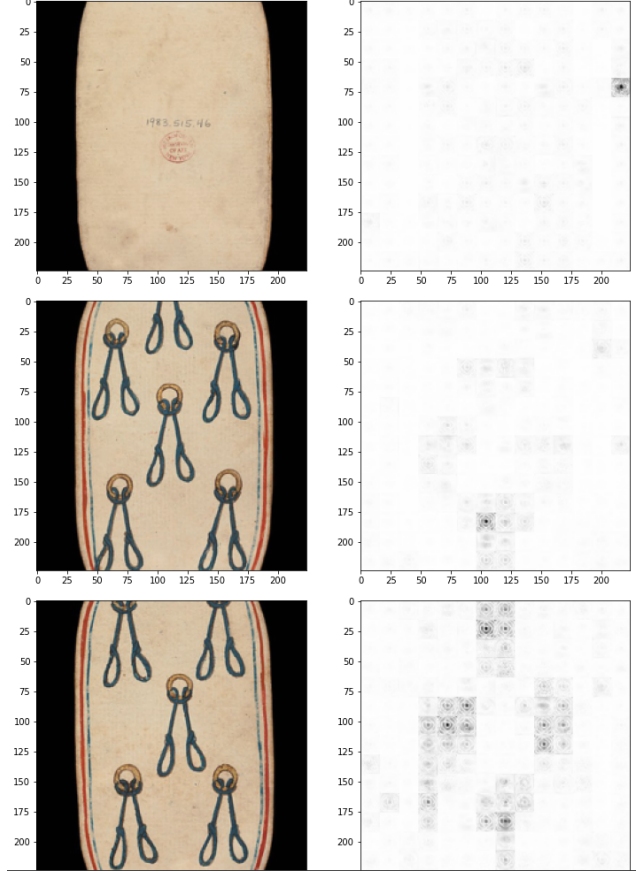


Figure 5. Saliency Maps for attribute *medium::glaze*

ually distinct attributes, like "daisies", "hourglasses" and such occurred in images of the same style. We could augment the dataset by cropping the distinct features into images of vastly differently styles e.g. paste hourglasses into images by Van Gogh or Vermeer. This will take a lot of extra effort but might be worth it. Another novel approach might use a page-segmentation or instance segmentation model to identify these visually distinct attributes, crop the relevant areas and generate augmented data for the project.

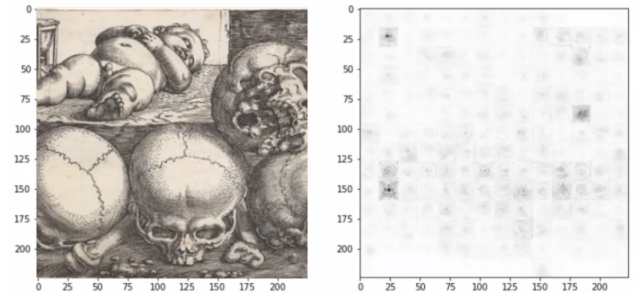


Figure 6. Saliency Maps for test image with predicted attribute *tags::hourglasses*



Figure 7. Training images with attribute *tags::hourglasses*. (There is indeed an hourglass in each of these images.)

References

- [1] iMET data set at FGVC8 competition CVPR 2021. <https://sites.google.com/view/fgvc8/competitions/imet2021>. Accessed: 2022-05-08. 1
- [2] Masked Auto-Encoder/Vision Transformer at HuggingFace (ViTMAE). https://huggingface.co/docs/transformers/model_doc/vit_mae. 2
- [3] Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers, 2021. 4
- [4] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers, 2021. 4
- [5] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations, 2020. 8
- [6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2020. 4
- [7] Kana Sethu et al. CS231n Spring 2022, Homework 2, Network Visualizations, 2022. 6
- [8] Geiger Friedman and Goldszmidt. 4
- [9] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners, 2021. 1, 4
- [10] Sara Hooker, Dumitru Erhan, Pieter-Jan Kindermans, and Been Kim. A benchmark for interpretability methods in deep neural networks, 2018. 2
- [11] Jack Lanchantin, Tianlu Wang, Vicente Ordonez, and Yanjun Qi. General multi-label image classification with transformers, 2020. 2
- [12] Nian Liu, Ni Zhang, Kaiyuan Wan, Ling Shao, and Junwei Han. Visual saliency transformer, 2021. 2, 8
- [13] T. Nathan Mundhenk, Barry Y. Chen, and Gerald Friedland. Efficient saliency maps for explainable ai, 2019. 2
- [14] Marc Mézard and Andrea Montanari. Information, physics, and computation, 2009. 2
- [15] Harini Padmanaban. Comparative analysis of naive bayes and tree augmented naive bayes models (master’s projects), 2014. 2
- [16] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps, 2013. 2
- [17] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps, 2013. 2
- [18] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net, 2014. 2
- [19] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. 2019. 3
- [20] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention, 2020. 4
- [21] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks, 2013. 2