Scene Categorization with Shape-Based Measures

COMP 558 Final Course Project

Yik Yu Ng, Matthew Barg, Michael Lambiri

December 18, 2024

Contents

1 Abstract

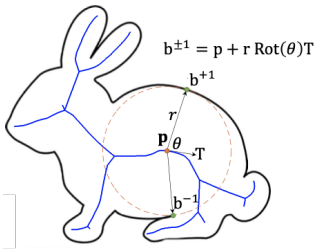
Scene categorization using deep neural networks require large datasets to perform tasks humans can do with much smaller amounts of information. In a paper by Professor Siddiqi and several other collaborators, a way to enhance traditional computer vision techniques using CNNs to perform scene classification was proposed. The paper use shape-based properties (parallelism, separation, mirror, taper) to score contoured images and enhance CNN categorization. We propose a new additional measure, convexity, in hopes to see how other common geometric properties can help determine a scene. From our results, we find that convexity may be a valid form of measuring the shape of a given scene.

2 Introduction & Background Theory

The referenced paper explores the role of shape-based perceptual cues, inspired by Gestalt grouping principles, in improving scene categorization tasks. By computing contour-based importance measures using the medial axis transform (MAT), their is a demonstration in how local geometric properties can enhance scene recognition.

2.1 Important Concepts : MAT and MAT Features

A **medial axis** is the set of centers of all disks within a bounded region D, such that the disks touch at least 2 of the boundary points on D. The **medial axis transform** is the set of centers of all disks, paired with their respective radii. An example of this is displaced below in the figure of a bunny:



Where p is the center of the disk (a medial axis point), r is the radius of the disk, and $b+1$ and $b-1$ are the boundary points which the disk touches. Thus the set of points that work just like that is the medial axis set. This concept is used to build skeletal plots by finding the distance for all points in the image to boundaries (that includes the boundaries of other objects or of the image itself). This gives a plot for distance for all pixels in the image. Then, the gradient of this plot is taken at each point, providing the rate of change of the distance to the boundaries. With this measure of gradient, the path integral over the boundaries of the images with respect to the gradient function over the image is found:

$$\text{AOF} = \frac{\int_{\partial R} \langle \dot{q}, N \rangle \, ds}{\int_{\partial R} ds}$$

Where ds is the boundary differential, $\dot{q}$ is the gradient, N is the boundary normal, and **AOF** (average outward flux) is a measure of how well $\dot{q}$ aligns with the boundary of the normal. The

maxima of this function are observably regions that are locally symmetric, which means they are equidistant from multiple boundary points. Thus, it gives us our medial axis points. Upon computing the medial axis points, we can then trace along them to generate a skeletal plot of the image based on the medial axes.

The medial axes provides contour importance scores. These scores are inspired by Gestalt principles, which describe how humans group and interpret visuals. When defining each feature, here are the main variables and concepts used:

- p : parameter in medial axis segment
- $r(p)$: medial axis radius at each point
- $\mathbf{C}(p) = \big(x(p), y(p)\big)$: coordinate of points along a particular segment
- $[\alpha, \beta]$: interval for a particular medial segment
- L : arc length of a particular medial segment
- Two curves considered in some of the features:
  - $\mathbf{\Psi} = \big(x(p), y(p), r(p)\big)$
  - $\mathbf{\Psi}' = \big(x'(p), y'(p), \frac{dr(p)}{dp}\big)$

Here are the mathematical and qualitative definitions for the scoring:

1. Separation: Measure of distance between contours in a scene. Contours that are more distant from others have higher separation scores

   $$S_{\text{separation}} = 1 - \left(\frac{\int_\alpha^\beta \frac{1}{r(p)}\, dp}{\beta - \alpha}\right)$$

2. Parallelism: Measures of parallelism in a scene (how similar two sides of the scene appear). It is calculated using the ratio between the arc length of the medial axis segment and curve $\mathbf{\Psi}$ since in scenes with high parallelism, the medial axes span the same length as the contour.

   $$S_{\text{parallelism}} = \frac{L}{L_\Psi} = \frac{\int_\alpha^\beta \big(x_p^2 + y_p^2 + r_p^2\big)^{\frac{1}{2}}\, dp}{\int_\alpha^\beta \big(x_p^2 + y_p^2 + r_p^2\big)^{\frac{1}{2}}\, dp}.$$

3. Taper: Taper measures taper symmetry, which refers to the idea that shape gradually changes its width along its length while remaining symmetric.

   $$S_{\text{Taper}} = \frac{L}{L'_\Psi} = \frac{\int_\alpha^\beta \big(x_p^2 + y_p^2 + r_p^2\big)^{\frac{1}{2}}\, dp}{\int_\alpha^\beta \big(x_p^2 + y_p^2 + (r_{pp})^2\big)^{\frac{1}{2}}\, dp}.$$

4. Mirroring: Mirroring measures how well one part of a scene or shape replicates another about part an axis. This property is inversely proportional to curvature since it is inversely related to the radius of the osculating circle, thus high curvature = low osculating circle radius = sharper changes in the curve.

$$S_{\text{Mirror}} = \frac{\int_{\alpha}^{\beta} R_{\text{curv}}(p)\,dp}{\beta-\alpha} = \frac{\int_{\alpha}^{\beta} \frac{1}{\kappa(p)}\,dp}{\beta-\alpha}$$

Where:

- $\kappa(p)$: Curvature at point p.
- R_{curv}: Radius of the osculating circle (inversely proportional to curvature).

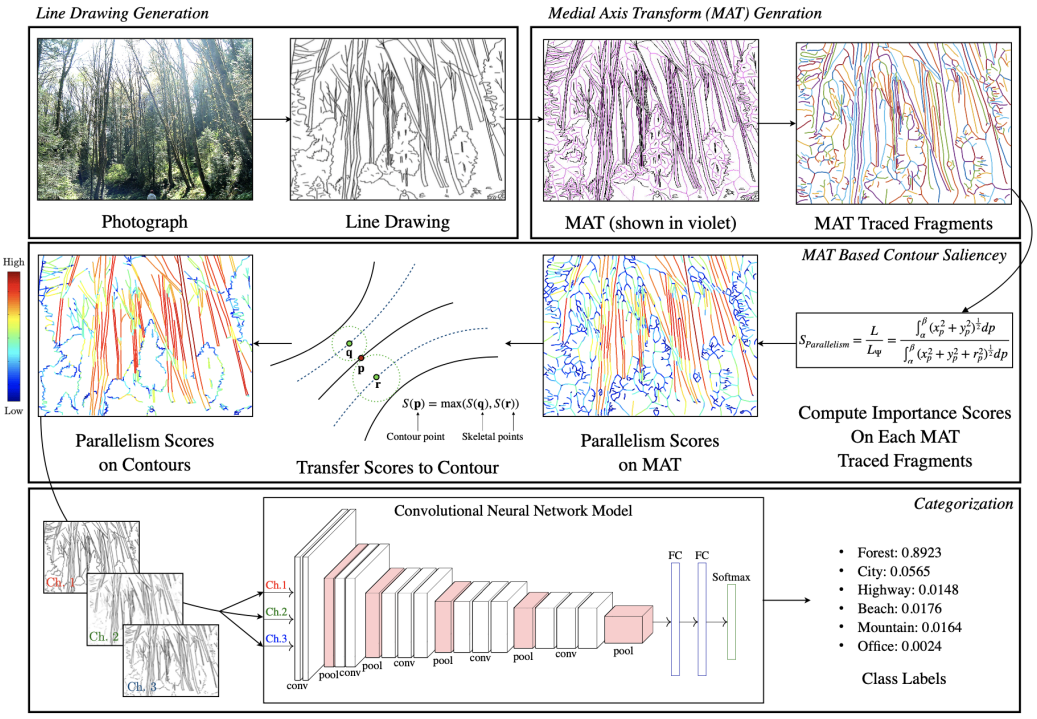
All of these measures provide key insight into the scene taken. For example, man-made objects have notoriously high parallelism scores, as they are symmetric in how they are constructed (buildings, chairs, roads, etc.).

2.2 Overall Process

The algorithm proposed in the paper works as follows:
Given an image I(x,y) of a scene:

1. Compute the line drawing of I
2. Compute the MAT skeletal plot of I & trace its fragments
3. Score each point on the MAT skeletal plot based on one of various properties (Mirror, Parallelism, Taper, and Separation)
4. Transform the scored MAT plot into a contour plot
5. Give the contour plots to a CNN, which will do the overall categorization

The paper provides an image descriptor of this algorithm below:



2.3 Additional MAT Feature - Convexity

Motivation

In computer vision and computational geometry, capturing shape properties is essential for analyzing and understanding structures, particularly in applications such as 3D modeling, object recognition, and scene analysis. Convexity is a critical geometric property that indicates how "curved outwards" a shape is. For polygons, a shape is convex if every line connecting two vertices lies entirely within the shape. However, for continuous curves and medial axis representations, defining and measuring convexity requires a more nuanced approach. Human can easily locate the curved or linear region by perception, so does machine?
We propose a *convexity measure* to quantify this geometric property locally along skeletal branches. The intuition is that convexity captures the "turning" behavior of the contour along with the medial axis, offering valuable shape information. This measure can enhance performance in machine learning models, such as convolutional neural networks (CNNs), by providing robust geometric features.

Intuition Behind the Measure

The medial axis (or skeleton) provides a concise representation of a shape, encoding the symmetry and topology of the underlying structure. Along the medial axis, the *radius function* describes the distance from each skeletal point to the contour. The key idea is:

- **Convex regions:** The radius function increases and then decreases along a segment, creating a *local maximum*.
- **Concave regions:** The radius function decreases and then increases along a segment, forming a *local minimum*.

These changes in the radius function can be quantified using its *second derivative*. A positive second derivative suggests convexity, while a negative second derivative indicates concavity.

Mathematical Formulation

To quantify convexity mathematically, consider the continuous radius function $R(s)$, where s is the arc length along the medial axis. For a segment of the medial axis between arc lengths α and β, the convexity measure C is defined as:

$$C = \frac{\int_{\alpha}^{\beta} -\left(\frac{d^2R(s)}{ds^2}\right)\, ds}{\beta - \alpha}.$$

Here:

- $\frac{d^2R(s)}{ds^2}$: The second derivative of the radius function quantifies the curvature or "sharpness" of the radius's change.
- $\beta - \alpha$: Normalizes the measure by the arc length of the segment, ensuring invariance to scale.

Normalization

1. **Length normalization:** By dividing by $\beta-\alpha$, C becomes invariant to the segment's length.
2. **Range normalization:** To map the convexity measure into $[0, 1]$, we normalize the scores using the maximum value observed over the entire medial axis in the current branch, so comparisons between different branches are fair.

Implementation Details

Algorithm Steps:

1. **Preprocessing:** Compute the medial axis of the shape and extract the radius function $R(s)$ along the axis.
2. **Compute derivatives:** Estimate the first and second derivatives of $R(s)$ numerically for discrete points.
3. **Integrate locally:** Evaluate the integral of $\left(\frac{d^2R(s)}{ds^2}\right)$ over small intervals $[\alpha, \beta]$.
4. **Normalize:** Normalize the result for length invariance and range mapping.

Algorithm 1: Convexity Measure Algorithm

Input: Medial axis M, radius function $R(s)$, window size K
Output: Normalized convexity measure $C(s)$ for each point on M

1 **Preprocessing:** Smooth the first derivative of the radius function $R(s)$:
- Compute the first derivative $dR(s)$ and smooth it.
- Compute the second derivative $ddR(s)$ from $dR(s)$ and smooth it.

Convexity computation:
foreach *point* s_i *along the medial axis* M **do**
  Determine the effective window size $eK = \min(\min(i-1, N-i), K)$;
  Compute local convexity $C(s_i) = -\text{mean}(ddR(s_{i-eK:i+eK}))$;
end
Normalization:
Normalize $C(s)$ to the range $[0, 1]$ using:

$$C(s) = \frac{C(s) - \min(C)}{\max(C) - \min(C)}$$

Visualization

The convexity measure is visualized as a heatmap along the medial axis:

- High scores correspond to turning points (e.g., sharp corners).
- Low scores appear on straight segments where $\frac{d^2R(s)}{ds^2} \approx 0$.

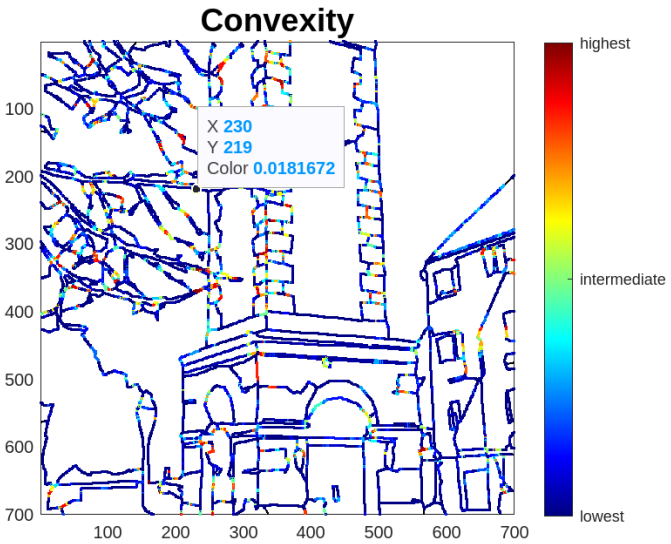


Figure 1: Heatmap Visualization of Convexity along the Medial Axis (High convexity scores highlight areas with pronounced curvature)

Short Summary:

1. **Shape analysis:** Detecting and characterizing regions with high curvature.
2. **Input to ML models:** Providing geometric descriptors to improve performance in object recognition or scene understanding.

2.4 CNN Structure

To effectively classify the shape properties of medial axis-based features, we utilize the well-established VGG16 convolutional neural network architecture. VGG16 is a deep network pre-trained on ImageNet, known for its ability to capture hierarchical features in images. This powerful architecture serves as the backbone of our framework.
In our approach:

- **Modified Fully Connected Layers:** The original fully connected layers of VGG16 are replaced with two new fully connected layers, followed by a softmax layer tailored for our datasets.
- **Pre-trained Weights:** The convolutional layers retain weights pre-trained on ImageNet, leveraging its rich feature extraction capabilities for our task.
- **Output Processing:** The final softmax layer produces a classification vector, where the highest score determines the predicted label.

This setup enables the network to integrate medial axis-based features, such as the convexity measure, with other geometric properties, enhancing its ability to learn subtle shape variations. By fine-tuning the network on our datasets, this architecture captures the rich geometric features derived from the medial axis and supports robust classification across various categories.

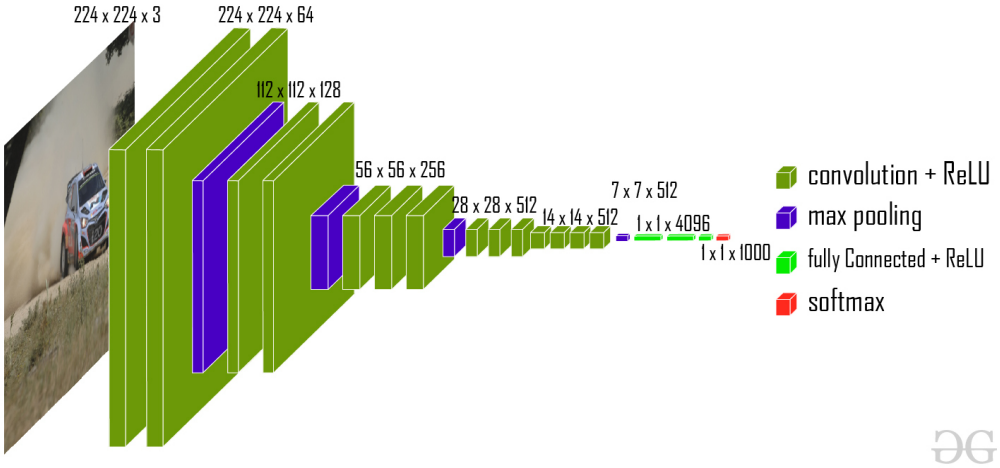


Figure 2: Highly Similar Modified VGG16 Network Structure: Pre-trained convolutional layers extract hierarchical features, while custom fully connected layers perform dataset-specific classification. (From GeeksforGeeks)

3 Methods

3.1 MLV Toolbox Codebase & Additions

The Bernhardt-Walther Lab at the University of Toronto has created a MLV ToolBox GitHub repository. This toolbox provided us with the ability to analyze medial axes and perceptual organization cues without having to produce all of the matlab logic. The process for generating results for a single test image was as follows.

First, a vectorized line drawing was created from the RGB image using function *traceLineDrawingFromRGB*. The vectorized line drawing was taken and a MAT was computed (using *computeMAT*). Within computing the MAT, the distance map, average outward flux (AOF) and the skeleton is all calculated. For the sake of testing, we only cared about the skeleton produced. Two illustrations were produced following this; The vectorized line drawing was visualized into an image using *renderLinedrawing* and the MAT skeletoned was overlayed on the line drawing to create a second image.

We were now able to analyze the contour properties (using *computeALLMATfromVecLD*). This step required us to add to the codebase. Initially, a very small adjustment had to be made for the default settings to include the taper feature (as this was left out in the original code). More importantly, we had to write additional code to one of the helper functions to include our new feature we added, "convexity". Once all properties were computed, visuals for each property were created (using *drawMATproperty*). In the paper, 50-50 splits of the contour scenes were created. The 50-50 splits analyzed in the paper were used to evaluate how feature scores are distributed across a particular scene and identified the most informative regions for each feature. The process divides pixels into top 50% (high scores) and bottom 50% (low scores) based on the magnitude of the feature scores. These gave key insights in how features contributed to the overall structure of the scene. To reproduce this analysis, the line drawings of each MAT property were split into two new images, the top 50% of values and the bottom 50% of values being separated. This took the scores of each property (including our additional property) and produced 3 new images for each (a total of 15): a line drawing with only the bottom 50% values visible (surrounded with a blue border), a line drawing with only the top 50% values visible (surrounded with a red border) and a intact line drawing with all values (bottom values colored blue, top values colored red). An example of these images are seen in figure 8. To produce these 15 images, we had to write a new function. We called it *drawSplitMATproperty* and it did exactly as described above. After creating the 50-50 splits, our group wanted to experiment with different splits such as 30-70 and 20-80. This required the function to be modified to be more flexible.

3.2 CNN

When running the CNN, we concatenated the results from 3 different scored contours to create a new scored contour with the R, G, and B channels each representing a different score. We did this by gray-scaling the scored contour images with their R and G channels switched. This is because Matlab's gray-scale function calculates the gray-scale intensity I (approximately) as $I \approx 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B$. This is however problematic, since if we were to do gray-scaling like this, then areas with high scores (drawn red) and areas of intermediate scores (drawn green) would be drawn incorrectly. Thus before gray-scaling the image, we flip the weights of R and G to ensure that areas of high scores on the contour map to high gray-scale intensities.

Below is an example of one such plot for convexity/taper/separation (convexity = R, taper = G, separation = B) and the respective individual convexity, taper, and separation plots:

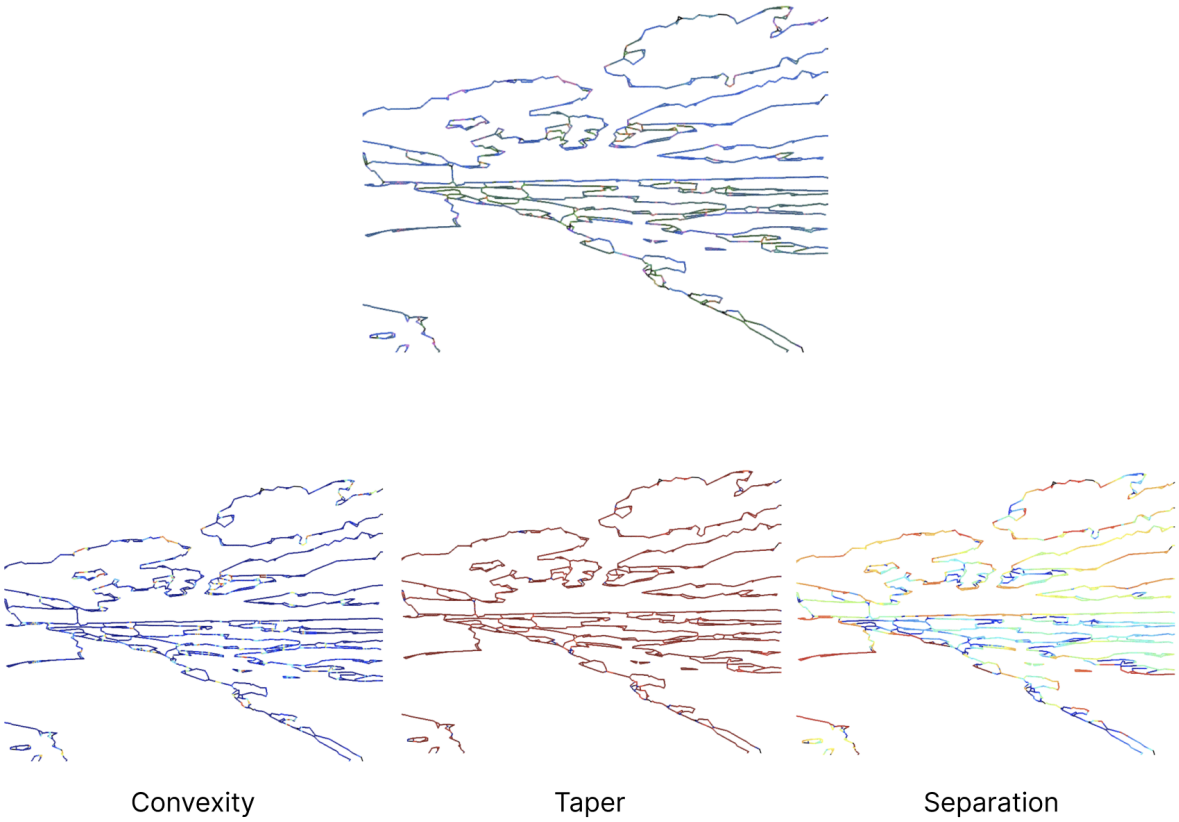


Figure 3: CNN Process - Input Construction

The taper score is universally high over the image, (mostly red) and thus the concatenated image is also mostly green. The convexity score is quite small across the image, only increasing at select corners through the image, which is reflected in the concatenated image by small pink spots. Lastly, the separation score is very high in the clouds and in select areas along the ground, leading to the concatenated image being very blue in areas that the separation is high (clouds, select areas of the ground), and being less present in areas closer to the horizon, which are visibly more green and red in the concatenated image.

4 Results

4.1 MAT & Features Generation

Multiple images throughout our experimenting phase were tested on but we chose to display two examples. In one example, the focus is a building while in the other image the focus is a dog. The building image was taken from the MLV Toolbox database while the image of the dog was a personal image. We chose these photos as they are quite distinct in categories and we were interested in how the contour scores and 50-50 splits of contour scenes would change. Figures 4 and 5 showcased the process of creating the line drawing image followed by the MAT skeleton from the original images. All of the MAT contour importance scores were visually summarized through heat maps in Figures 6 and 7. Lastly, the results of the splits of the contour scenes were shown in Figures 8 and 9.

Figure 4: Building Focused - MAT Process

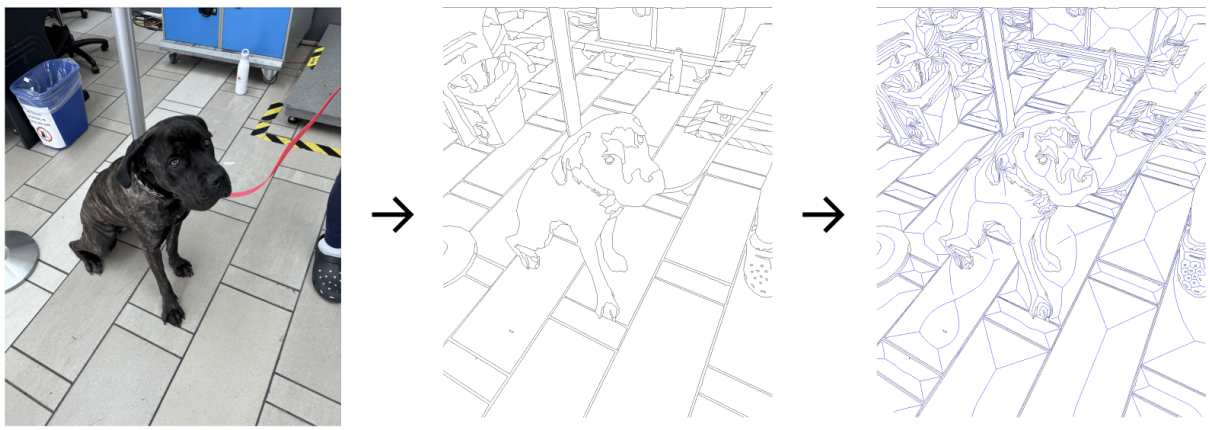


Figure 5: Animal Focused - MAT Process

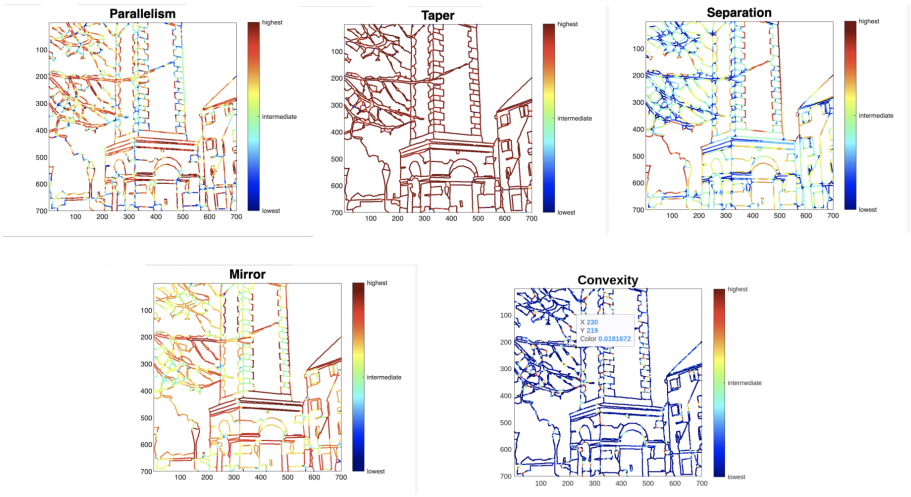


Figure 6: Building Focused - Heatmap Visualization of Scene Contour Scores

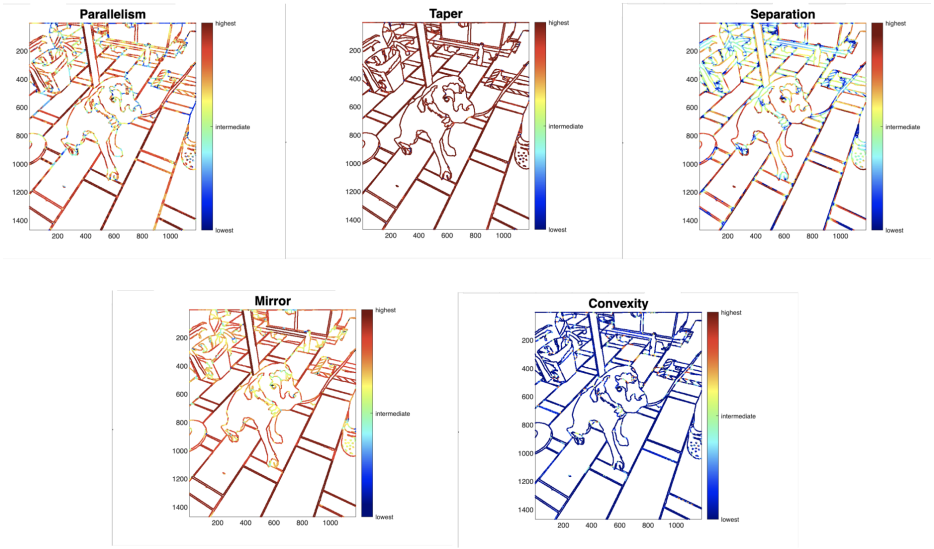


Figure 7: Animal Focused - Heatmap Visualization of Scene Contour Scores

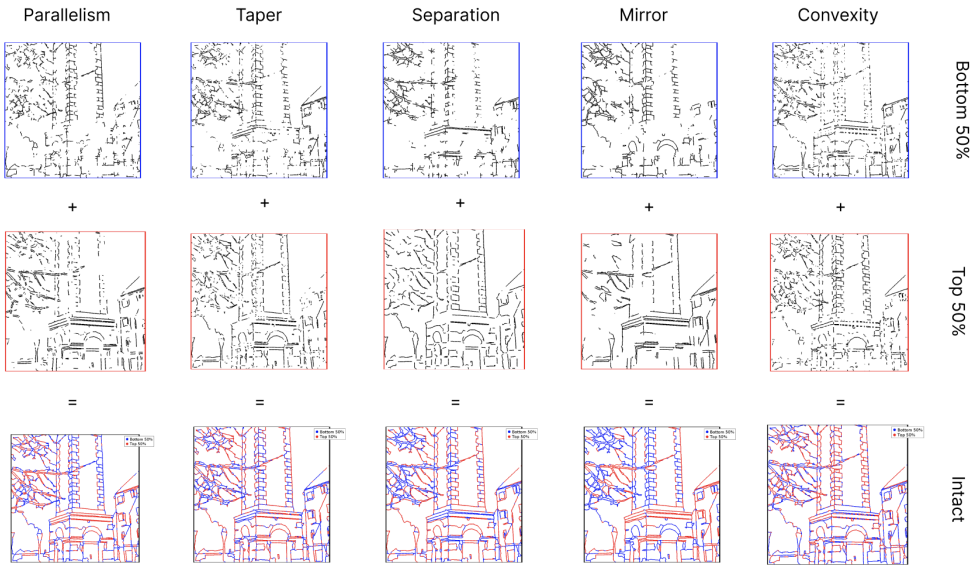


Figure 8: Building Focused - 50-50 Splits of Contour Scenes

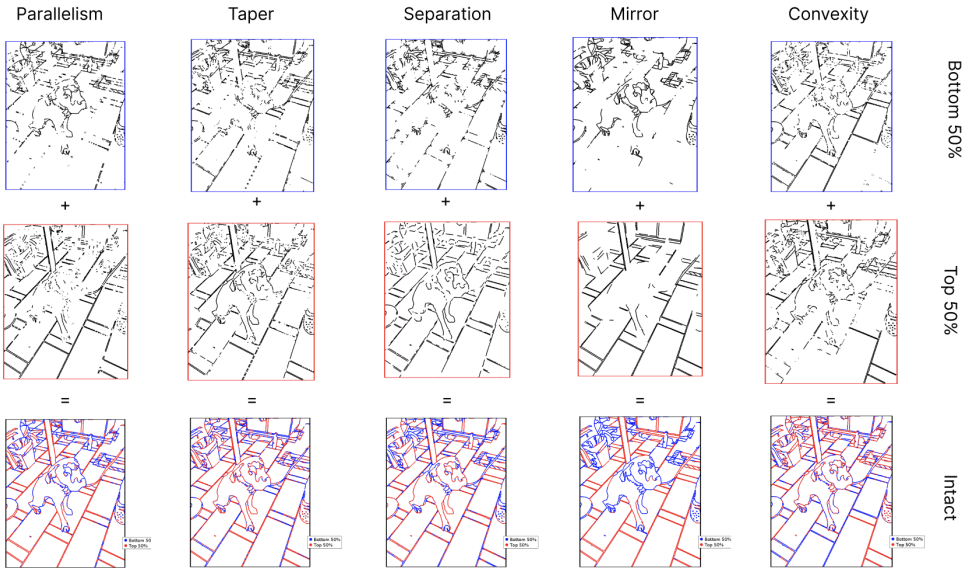


Figure 9: Animal Focused - 50-50 Splits of Contour Scenes

4.2 CNN

Table 1 summarizes the results from training the CNN. Each case was ran with 50 epochs, using 50/50 testing and using 90 images (15/16 for each of 6 different types). It is important to note that "Contour" in this context means a non-scored contour. We aimed to replicate the results found in the study while observing how our new added feature, convexity, performed.

Table 1: Top 1 & Top 5 Performances in 3-Channel Configuration

Channel 1	**Channel 2**	**Channel 3**	**Top 1**	**Top 5**	**f1-Score**
Convexity	Mirror	Taper	50.53%	91.58%	0.47
Convexity	Mirror	Parallelism	54.74%	94.74%	0.49
Convexity	Taper	Parallelism	51.58%	95.79%	0.45
Convexity	Taper	Separation	58.95%	96.84%	0.53
Convexity	Parallelism	Separation	50.53%	95.79%	0.45
Parallelism	Taper	Separation	60.00%	95.79%	0.53
Separation	Mirror	Parallelism	53.68%	93.68%	0.48
Taper	Mirror	Parallelism	53.68%	96.84%	0.45
Contour	Contour	Contour	60.00%	93.68%	0.55

- Top 1 score: The percent of test images such that the model's highest confidence prediction matches true label
- Top 5 score: The percent of time the correct label appears in the model's top 5 predictions
- f1 score: A score = 2(Precision)(Recall)/(Precision + Recall) where 0 <= Precision, Recall <= 1, Precision = % of how many instances the model labelled as positive and Recall = % of how many instances meant to belong to be in the positive set that are identified correctly. It can also be calculated by taking a weighted sum over all the points in the image (as was implemented)

5 Discussion

5.1 MAT Features, 50-50 Analysis

The heatmaps in figure 6 & 7 represent the intensity of each specific feature analyzed. The visualizations help highlight which parts of the image strongly exhibit these features. The two images chosen were used to show how two drastically different categories (artificial building and animal) provided some differences in results. For example, in the parallelism maps, high values are seen in areas with straight edges or aligned structures. These described characteristics are more prominent in the buildings in image 1 whereas in image 2, the background tile floor is where the high scores are seen and lower scores for the dog (the main focus of the image). Looking at the colors alone of these heat maps, it is hard to draw strong conclusions. Combining these results with the 50-50 splits in figures 8 & 9 strengthen our understanding of how the different features contribute to the overall categorization of a scene.

In image 1, observing the results of the top 50% pixels of each feature, the main building in the image can be made out quite well in every image. When comparing to the bottom 50%, the top values are noticeably better other than in mirror and convexity. These 2 features show a more even result between the top 50% and bottom 50% containing key components of the image. The building's consistent structure and curvature produce evenly distributed convexity and mirror scores across both top and bottom 50%. These scores are more global compared to the other features, for example, mirror evaluates how well one part of the scene or shape replicates another part. This property considers symmetry across relatively large regions of the image. For the other features, it makes sense the top 50% performs well since these features prioritize specific regions with high geometric regularity. Some different results are seen in image 2. Mirror's top 50% image almost eliminates the dog entirely, other than some of the body outline. This occurs because the dog's body lacks strong bilateral symmetry, especially at the angle the dog is positioned in the photo. The parallelism bottom 50% arguably performs better in identifying the dog than the top 50%. This result can be explained with similar reasoning as in the mirror results. As seen in the building image, the convexity appears to be rather even in performance.

5.1.1 Analyzing other Percentages - Top 30%, 20%

After observing the 50-50 splits, our group wondered how different splits would perform. To analyze if lower percentiles of the top values could capture enough of the scene, we produced splits of 30-70, 20-80. The following figures display the results. Only the top split images are shown since we are not concerned with the 70% & 80% results. For image 1, the top 30% results do a decent job of preserving the main building. The individual differences between features are similar to what was seen previously in the 50-50 split images. While you can still roughly identify the building, the top 20% results don't give enough detail overall. In image 2, again, the earlier observations in the 50-50 splits are seen but more drastically now. The dog has been removed from the mirror images, and almost entirely from the parallelism ones as well. Overall, one could hypothesize for most images it is reasonable to believe that 30% of the top scores is enough for a human to estimate the scene in a similar level to seeing the top 50%, while 20% is not enough. These estimations are much easier on consistent structures like the building in image 1, and more challenging on less consistent subjects like the dog in image 2.

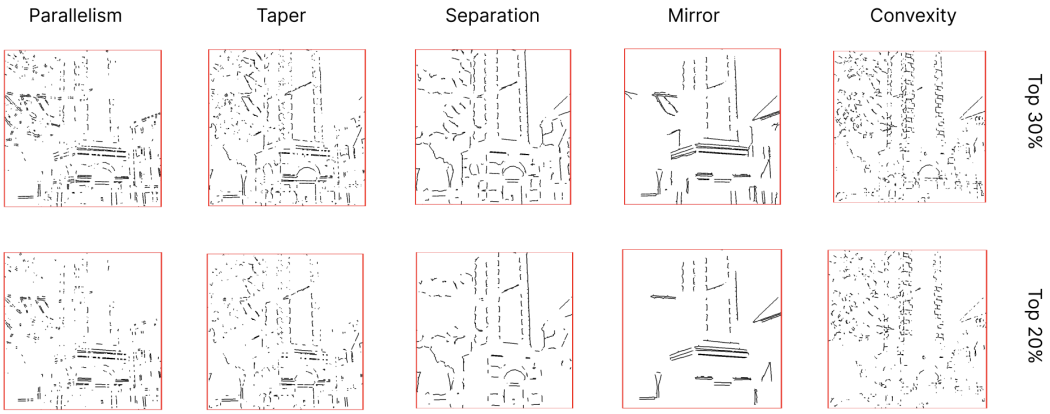


Figure 10: Building Focused - 30%, 20% Splits

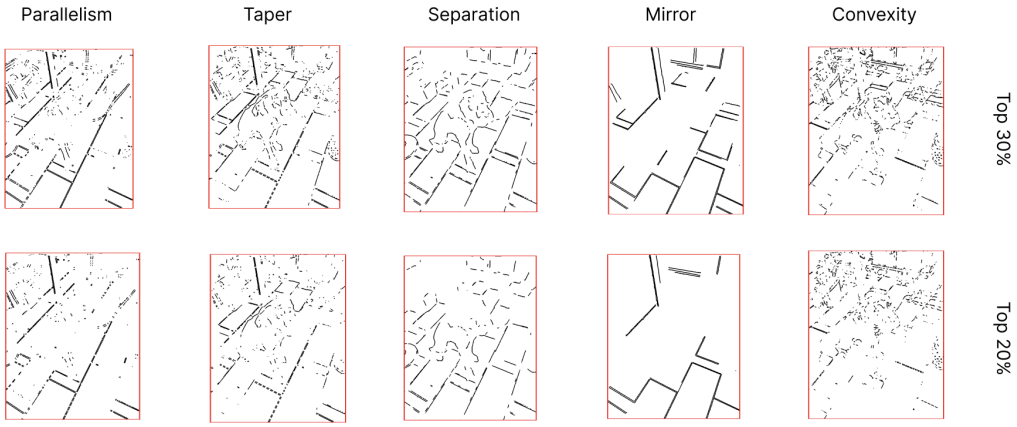


Figure 11: Animal Focused - 30%, 20% Splits

5.2 CNN

We were getting very high top1 and f1 values across the board when comparing to the original paper. We were surprised to see the non-scored contour plot is tied for the highest top 1 score and has the highest f1 score. Thus the non-scored contour plots have the greatest accuracy for high confidence predictions. We expected that the scored contour plots would have greater insight into the scene.

As for using convexity as a shape based measure of the scene, some interesting results are seen. Notably, convexity has much higher top 5 scores than either the top 1 or f1. Particularly the combination of convexity/ parallelism/separation and convexity/taper/parallelism had also some of the lowest top 1 and the lowest f1 score (0.45). It is possible that the other features play a role in these results. The low f1 score indicates that the precision and recall metrics are low, meaning that either few instances are labelled as positive, or many of the instances being labelled as positive are being incorrectly labelled (or both).

However, we did see some very interesting results when pairing convexity/separation/taper met-

rics. Not only does it have one of the highest top 1 scores (3rd) and f1 scores (2nd), but it has the highest top 5 score at 96.84%. Thus, this shows that convexity in combination with separation and taper provides accurate insight into the geometry of the scene and help categorize.

5.2.1 Improvements / Future Exploration

Possible further analysis can be done to look for experiments that could be run with the non-scored contour plot to see why it may have been particularly effective. For example, trying with a larger dataset. Testing with more of a variety of different scenes, so the model would have a better generalization power.

As we looked at different split percentages in section 5.1, this can also be applied to CNN. Our implementation only tested with 50-50 intensity splits where the results might change with different values. It is unlikely that reducing the number of values would increase the categorization results however, it would be interesting to see how much percentage in reduction there would be if at all.

The paper utilizes the VGG16 model, which is designed to accept three input channels at a time. However, this limitation is not inherent, and the model can be adapted to handle a different number of channels. An interesting direction for further experimentation could involve exploring the trade-off between increasing the number of geometric measures used as input channels and maintaining computational efficiency.

6 Conclusion

The purpose of our work was to explore the original study and try and reproduce some of the results. In addition to this, our goal was to attempt to expand upon it. The added feature of convexity was an effort to show a new category that offered additional insights to human and CNN scene categorization. Along with analyzing the top 50% splits of contour scores, exploring how the top 20% and 30% was meant to create further analysis on how much information in a scene is needed for confident identification by humans. These insights tried to offer some additional views and compliment the original report.

7 Work Distribution

Each group member participated in multiple group discussions. We collaborated on strategy and planned the overall structure as we progressed through the assignment. Below is a summary of the work distribution of the final product:

Yik Yu Ng - Creating additional MAT feature (Convexity), CNN code, paper sections 2.3, 2.4, 5.2.1, review

Matthew Barg - MLV Toolbox implementation and splits of contour scenes code, paper sections 3.1, 4.1, 5.1, 6, intro to 2, paper formatting and review

Michael Lambiri - Running CNN tests, paper sections 1, 2.1, 2.2, 3.2, 4.2, 5.2

8 References

1. Rezanejad M, Wilder J, Walther DB, Jepson AD, Dickinson S, Siddiqi K. Shape-Based Measures Improve Scene Categorization. *IEEE Trans Pattern Anal Mach Intell.* 2024 Apr;46(4):2041-2053. doi: 10.1109/TPAMI.2023.3333352. Epub 2024 Mar 6. PMID: 38039177.

2. bwlabToronto. *MLV Toolbox*. Available at: `https://github.com/bwlabToronto/MLV_toolbox`.

3. Dirk Warther's Lab. *Contour/Line Drawing Dataset*. OSF Repository. Available at: `https://osf.io/9squn/`.

4. PyTorch Team. *PyTorch Examples - ImageNet*. GitHub Repository. Available at: `https://github.com/pytorch/examples/tree/main/imagenet`.