

What Characterizes Personalities of Graphic Designs?

Supplemental Document

Nanxuan Zhao, Ying Cao, Rynson W.H. Lau
City University of Hong Kong

Contents

1	More Results on Model Evaluation	2
1.1	AMT Study for the Evaluation Dataset	2
1.2	Personality-based Design Ranking	2
1.3	Personality-based Design Retrieval	6
1.4	Design Feature Visualization on Posters	8
1.5	Generalization to Web Designs	8
2	More Results on the Sensitive Map	12
3	Effects of Design Factors on Personality	15
3.1	Color	15
3.2	Font	15
4	Element-level Design Suggestion	19
4.1	Candidate Value Generation and the Iterative Sampling Method	19
4.2	Details of the Evaluation Study	19
4.3	More Results	20
5	The Example-based Personality Transfer Method	25
5.1	Representation	25
5.2	Energy Terms	25
5.3	Optimizer	26
5.4	More Results	27

S 1 More Results on Model Evaluation

1.1 AMT Study for the Evaluation Dataset

Figure 1 shows a screenshot of our AMT study used to collect our evaluation dataset.

[Click here to show/close instruction](#)

Start Here!(Required)


Gender:

Age:


Graphic Design Experience:

modern

Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14	Q15	Q16	Q17	Q18	Q19	Q20	Q21	Q22	Q23
Q24	Q25	Q26	Q27	Q28	Q29	Q30	Q31	Q32	Q33	Q34	Q35	Q36	Q37	Q38	Q39	Q40	Q41	Q42	Q43	Q44		
Q45																						



Poster A



Poster B

Which Poster on the left is more modern?

☐ Poster A is more modern

☐ Poster B is more modern

☐ The image can't show! :(

☐ The choice is **obvious**

☐ The choice is **subtle**

↔

Figure 1: Screenshot of our AMT study used to collect our evaluation dataset. Workers are required to rank each pair of posters according to a given personality in 2AFC manner. Each HIT consists of 40 different comparisons for a single personality with 5 duplicate comparisons for consistency check.

1.2 Personality-based Design Ranking

Figures 2 to 4 show more results on comparisons of the top ranked and bottom ranked designs predicted by our model on different personalities.

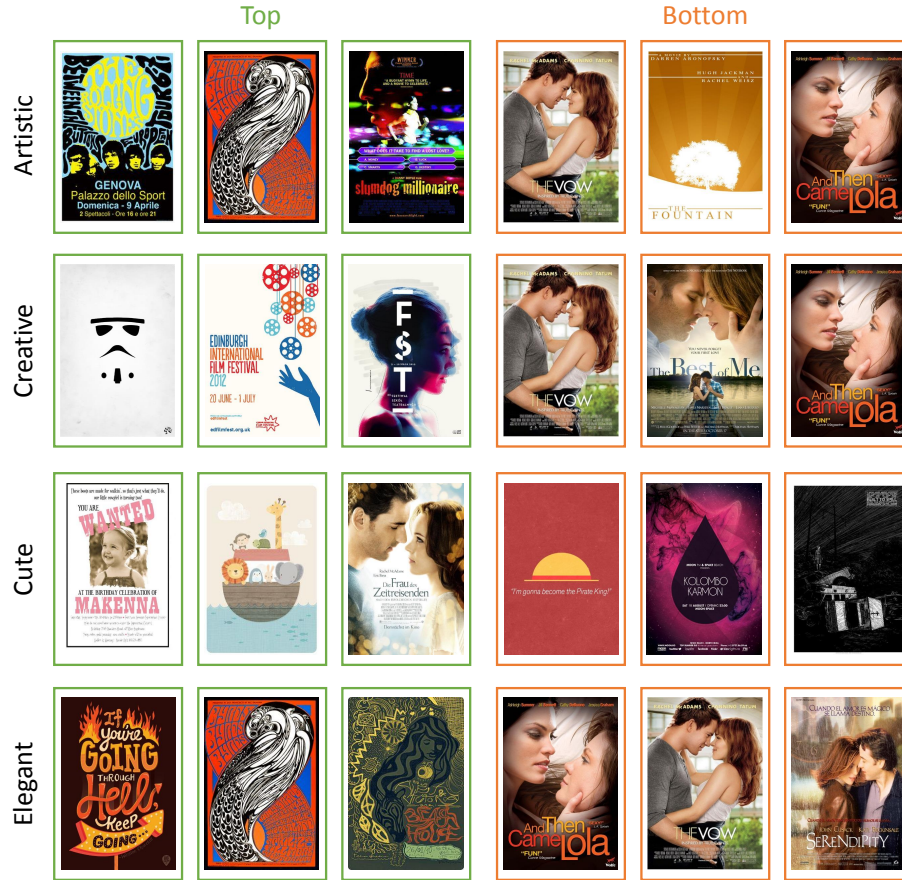


Figure 2: Comparison of the top ranked and bottom ranked designs predicted by our model on different personalities. For each personality, the top ranked designs are outlined in green boxes, while the bottom ranked designs are outlined in orange boxes.

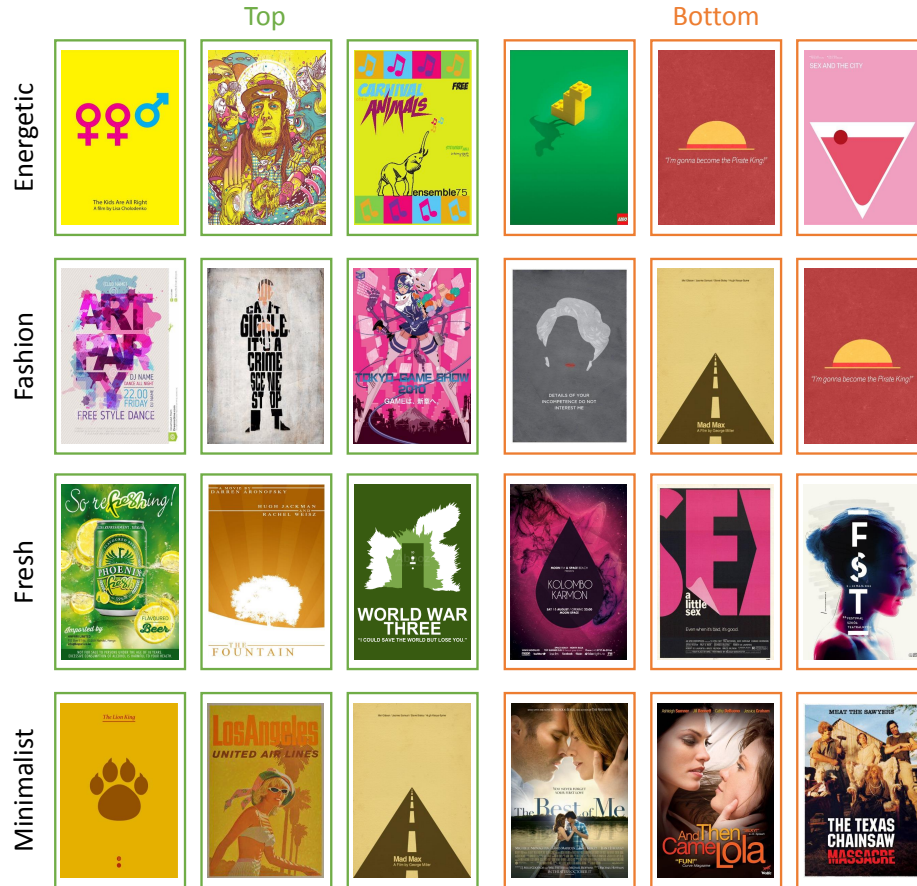


Figure 3: Comparison of the top ranked and bottom ranked designs predicted by our model on different personalities. For each personality, the top ranked designs are outlined in green boxes, while the bottom ranked designs are outlined in orange boxes.

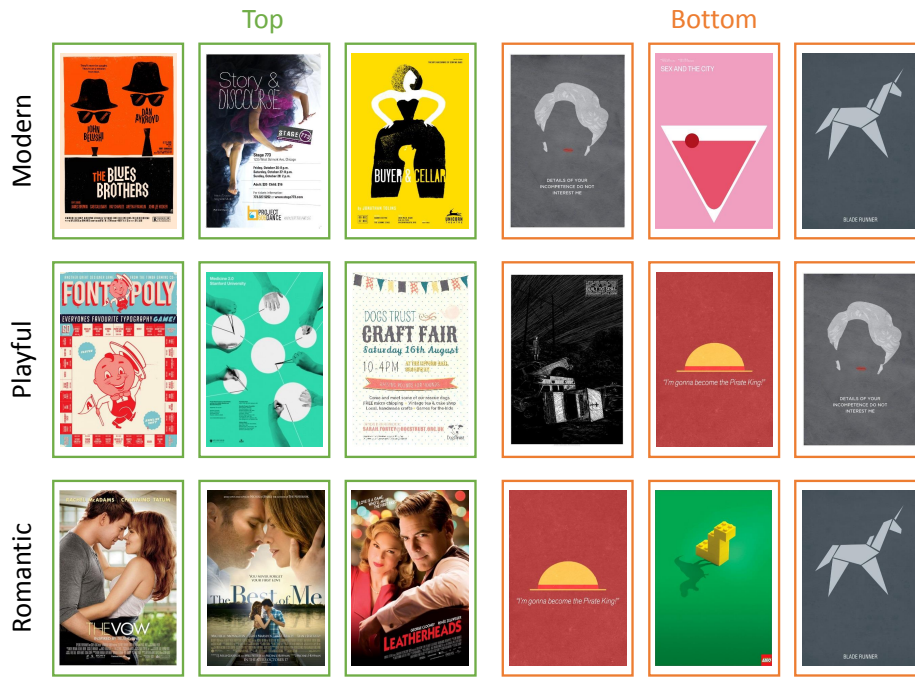


Figure 4: Comparison of the top ranked and bottom ranked designs predicted by our model on different personalities. For each personality, the top ranked designs are outlined in green boxes, while the bottom ranked designs are outlined in orange boxes.

1.3 Personality-based Design Retrieval

As our network is trained to predict personality scores, our deep features (i.e., \mathcal{F}) from our design feature network are supposed to capture personality-based semantics. To evaluate the effectiveness of our learned deep features, we perform personality-based design retrieval on our poster dataset. Given a query poster, we return a list of similar posters, measured by the cosine distance in our deep design feature space. Figure 5 shows some results. We can see that our deep features perform better than both the hand-crafted features and VGG features in retrieving posters with similar personalities.

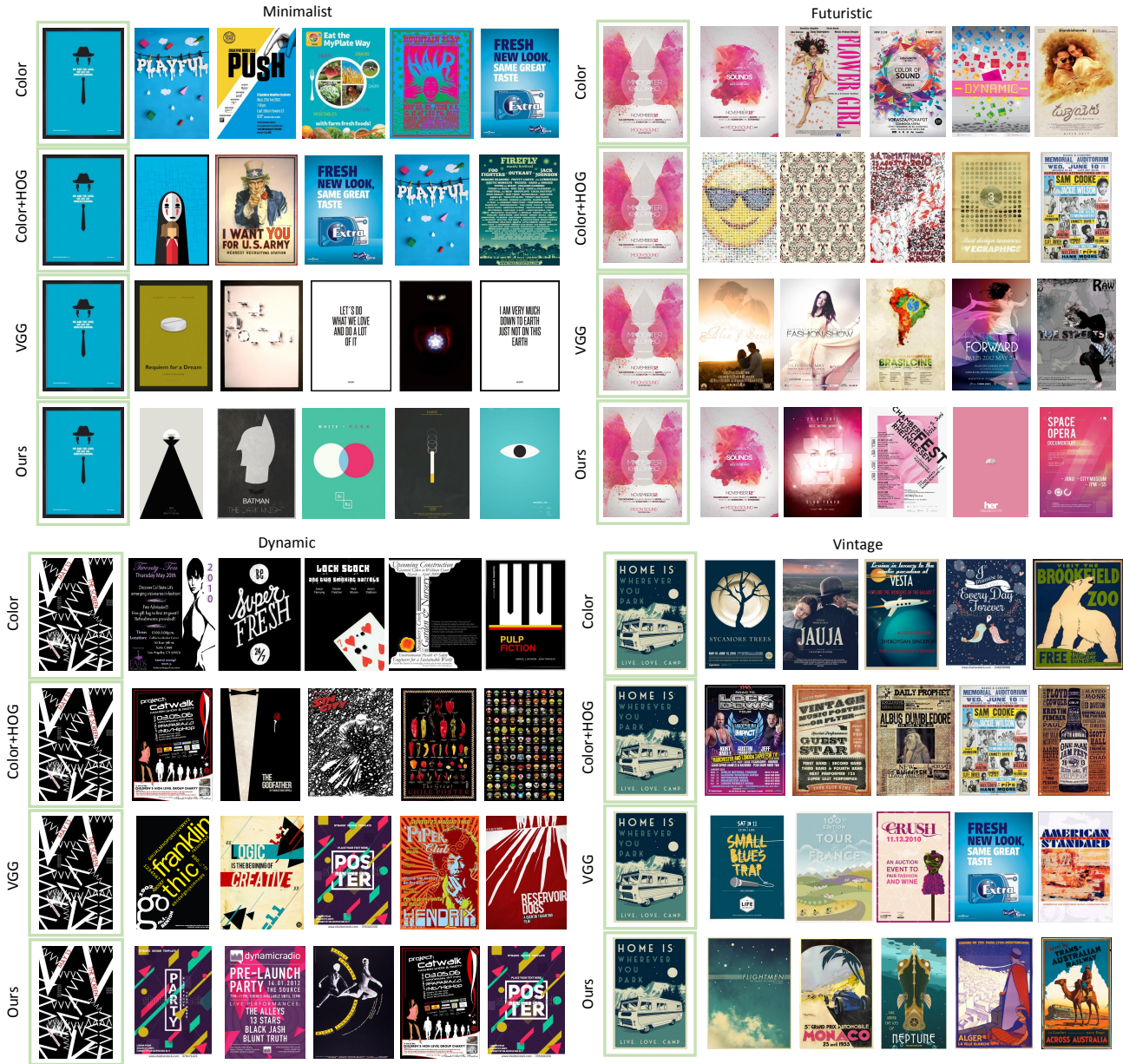


Figure 5: Personality-based design retrieval results. Given a query design (green boxes in the first column), we retrieve its nearest neighbors by computing the cosine distance using different features. Note how the results using our deep features (Ours) are more similar in personality to the queries.

1.4 Design Feature Visualization on Posters

We visualize our 256D design features using the t-SNE algorithm [MH08], mapping a high dimensional feature vector into a 2D embedding space. To reduce visual clutter, we randomly select 100 posters for each personality label and show the result by placing them on the 2D coordinates in Figure 6. Note how the distance between feature vectors in the design feature space reflects personality similarity.

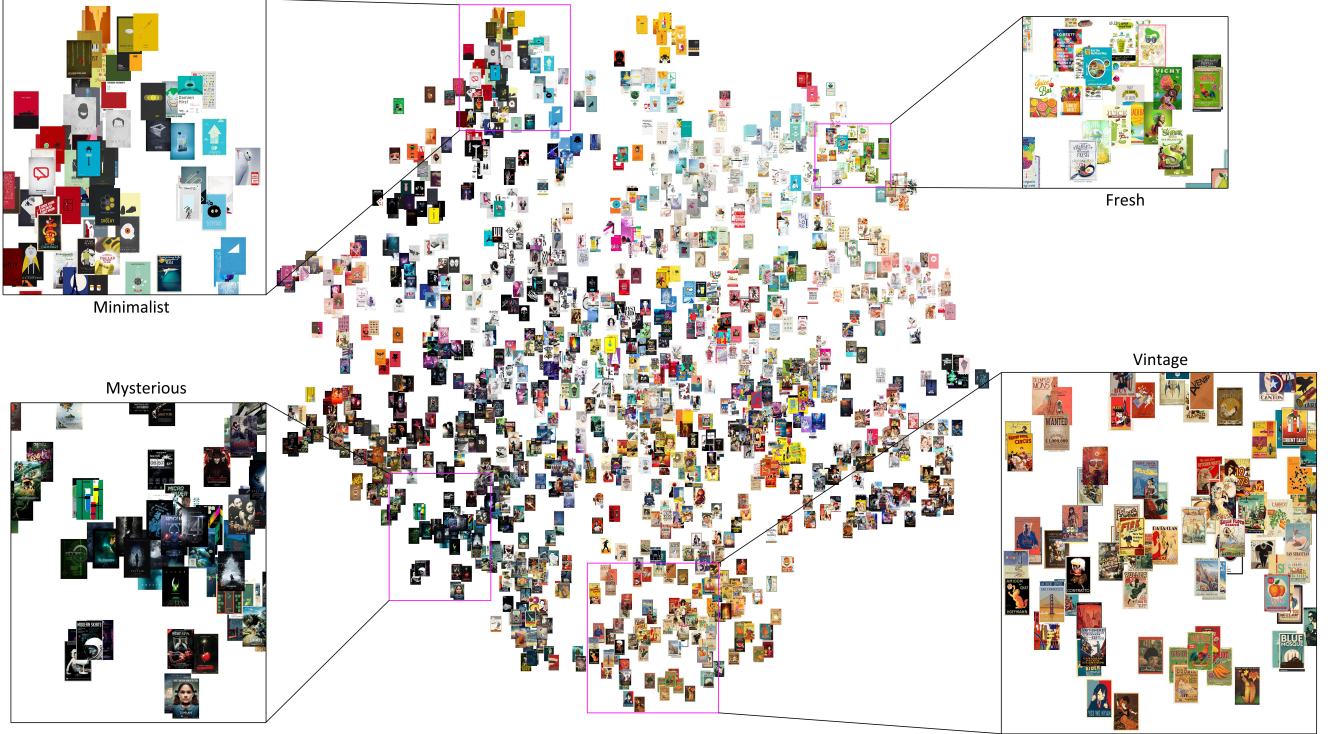


Figure 6: A 2D embedding visualization of our deep features using t-SNE, shown with 1500 selected posters (100 random selected posters for each of the 15 personality labels).

1.5 Generalization to Web Designs

To demonstrate the generality of our model to other types of graphic designs, we have also trained our network on webpages.

Statistics of the webpage dataset. We use the same set of personality labels from our poster dataset, except for two personality labels, “Dynamic” and “Mysterious”. This is because “Dynamic” in webpages may be used to indicate one website is written using a server-side scripting language, such as PHP, ASP, or JSP, while “Mysterious” is seldom used to describe webpages. We collect the webpage dataset in the same way as the poster dataset, as described in the main paper. We first use a personality label (e.g., “cute”) and “website” as keywords to search using the Google Image search engine, and only retain the images (i.e., webpages) in landscape format as they are prevalent in modern web design to fit the size of the monitors. We also filter out the low-resolution images (width is

smaller than 300 or height is smaller than 200. We end up with 2,838 webpages, whose distribution of personality labels is shown in Figure 7.

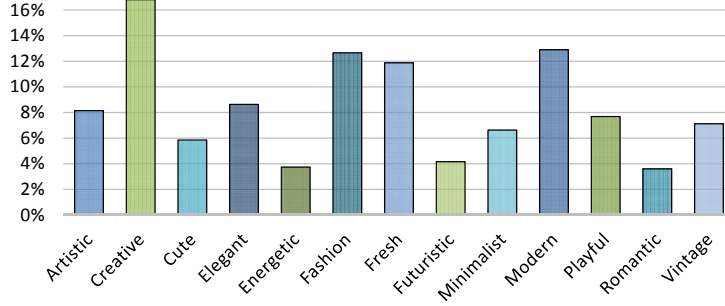


Figure 7: Distribution of personality labels in our webpage dataset.

Results. We train our semantics-aware deep ranking network in the same way as in the main paper, except that the input resolution is changed to 200×300 to fit the common aspect ratio of webpages. We perform a 5-fold cross validation on the collected data. Figure 8 shows the average accuracy of individual personalities. The average accuracy of all is 73.28%. Our model significantly outperforms the random guess baseline (50%), especially on “Cute”, “Fashion”, “Futuristic”, and “Vintage”. We have also performed personality-based design retrieval using our deep design features. As shown in Figures 9 to 10, our design features return more similar posters in terms of personality than other baseline features. These results suggest that our model can favorably deal with other types of graphic designs.

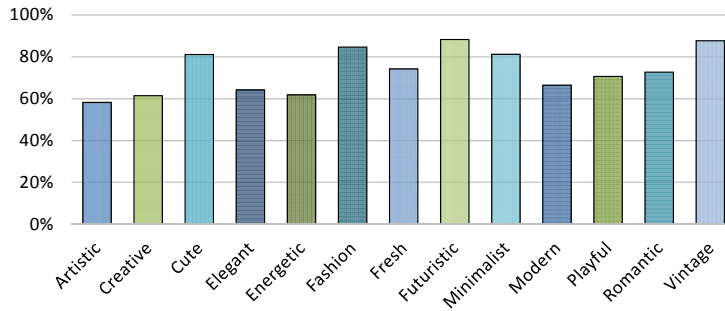


Figure 8: Accuracy of our deep ranking network on webpage dataset based on a 5-fold cross validation. Random guess gives an average accuracy of 50%.

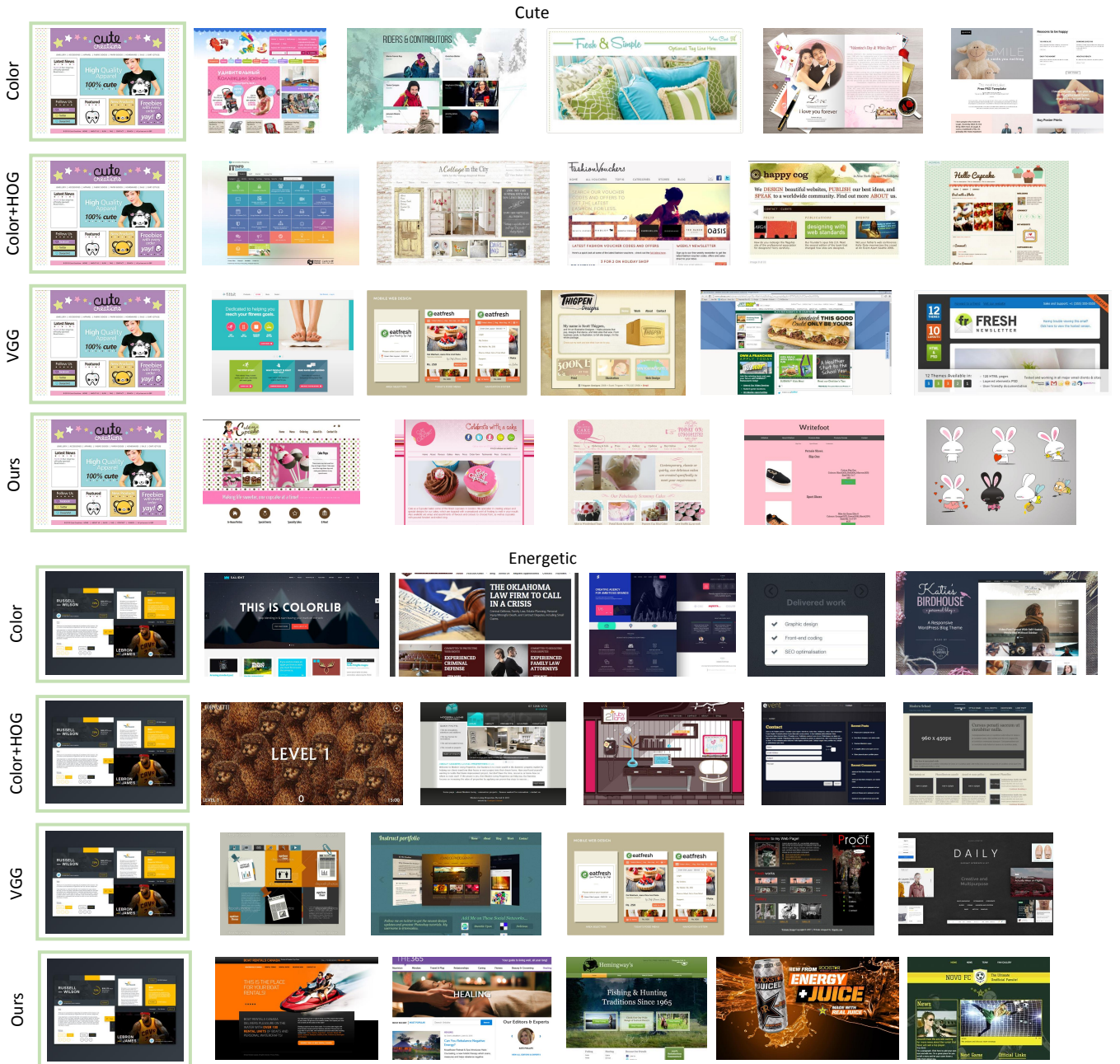


Figure 9: Personality-based design retrieval results on the webpage dataset. Given a query design (green boxes in the first column) from our webpage dataset, we retrieve the nearest neighbors by computing the cosine distance using different features. Note how our results (Ours) are more similar in personality to the queries.

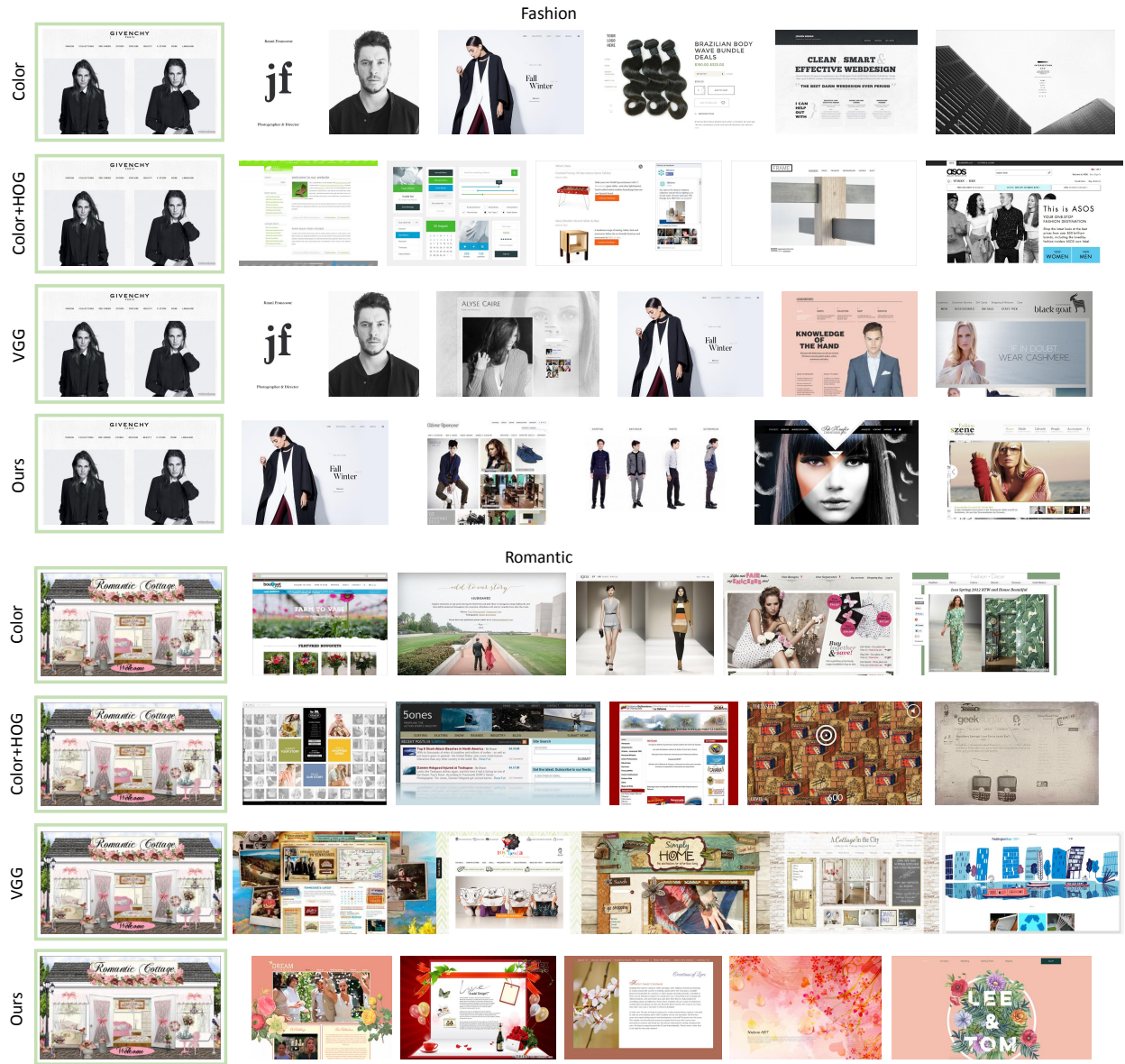


Figure 10: Personality-based design retrieval results on the webpage dataset. Given a query design (green boxes in the first column) from our webpage dataset, we retrieve the nearest neighbors by computing the cosine distance on different features. Note how our results (fourth row of each set) are more similar in personality to the queries.

S 2 More Results on the Sensitive Map

Figures 11 to 13 show more results on the sensitive map and discriminative patches.

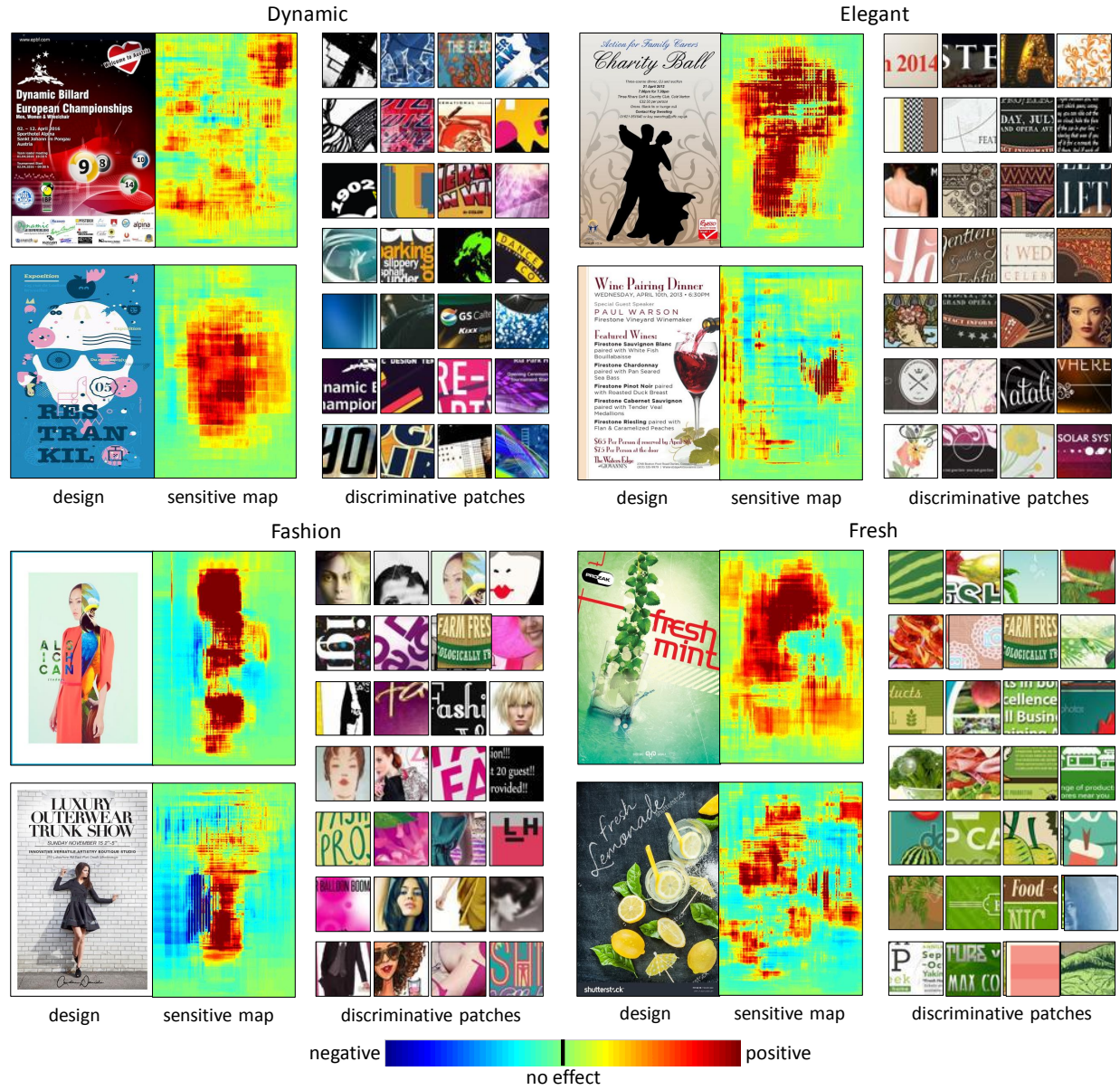


Figure 11: Sensitive maps and discriminative patches for various personalities. For each personality, input designs and their sensitive maps are shown on the left. A sensitive map is used to show the locations on a design with positive (red) or negative (blue) impacts upon a given personality. The discriminative patches are shown on the right, which are image patches that play important roles in characterizing a given personality.

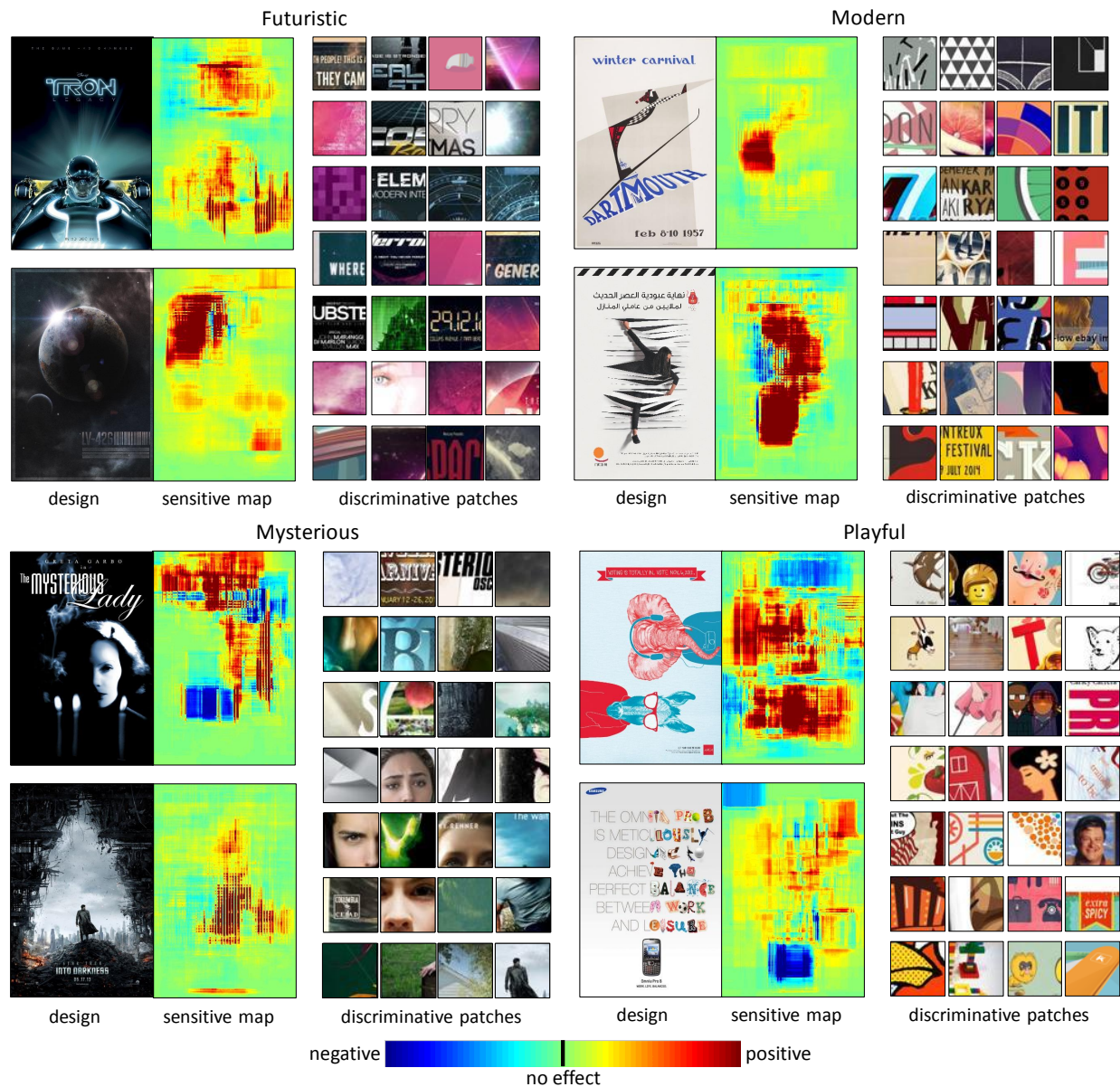


Figure 12: Sensitive maps and discriminative patches for various personalities.

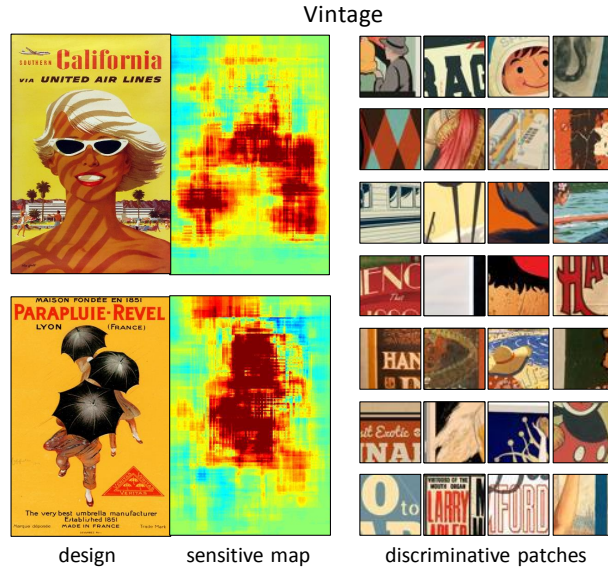


Figure 13: Sensitive maps and discriminative patches for “Vintage”.

S 3 Effects of Design Factors on Personality

3.1 Color

Figures 14 to 16 show more results on the correlation between HSV colors and graphic design personalities.

3.2 Font

Figure 17 shows more results on the correlation between font properties and graphic design personalities.

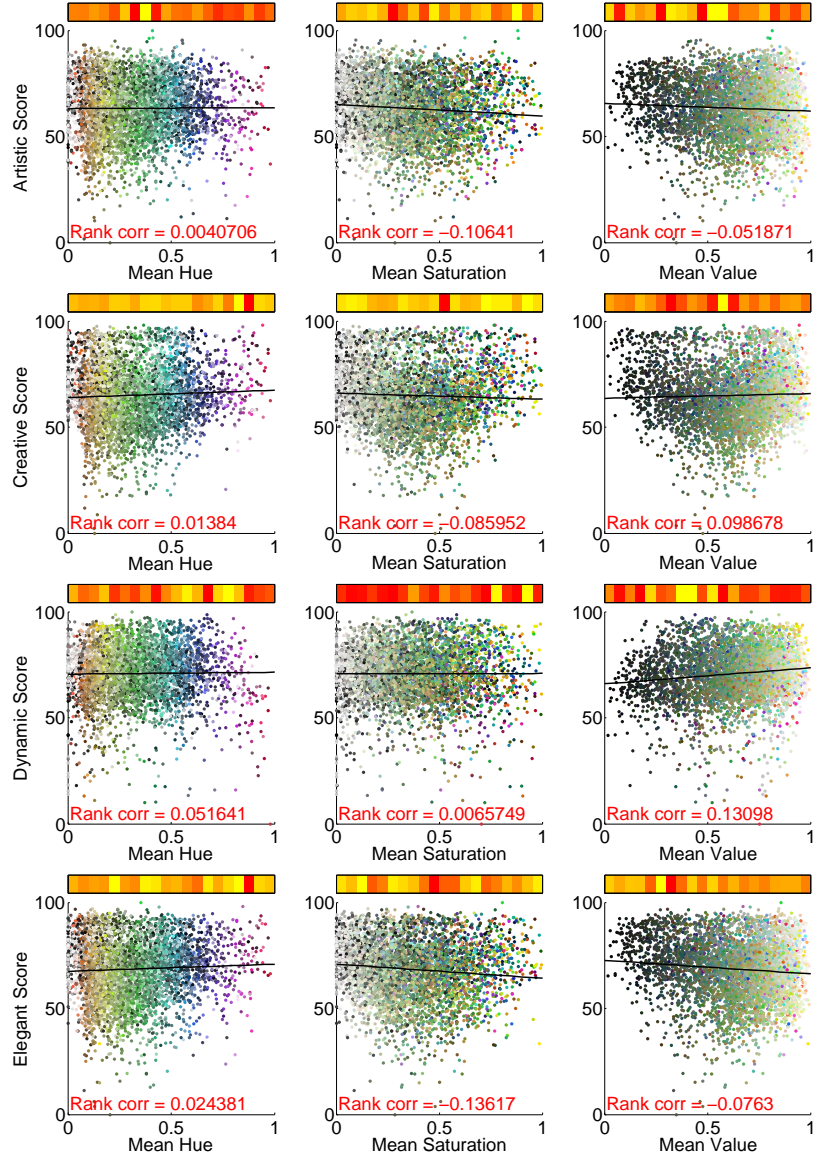


Figure 14: Correlation between HSV colors and personalities. In each diagram, a dot corresponds to one design represented by its mean color. A linear least-square fitting line (in black) is also shown, along with the Spearman rank correlation at the bottom-left corner. The weights of the linear RankSVM model are shown as a stripe above the diagram, where a block being more yellow (or red) indicates that the corresponding value range on the x-axis has a more positive (or negative) effect upon a specific personality.

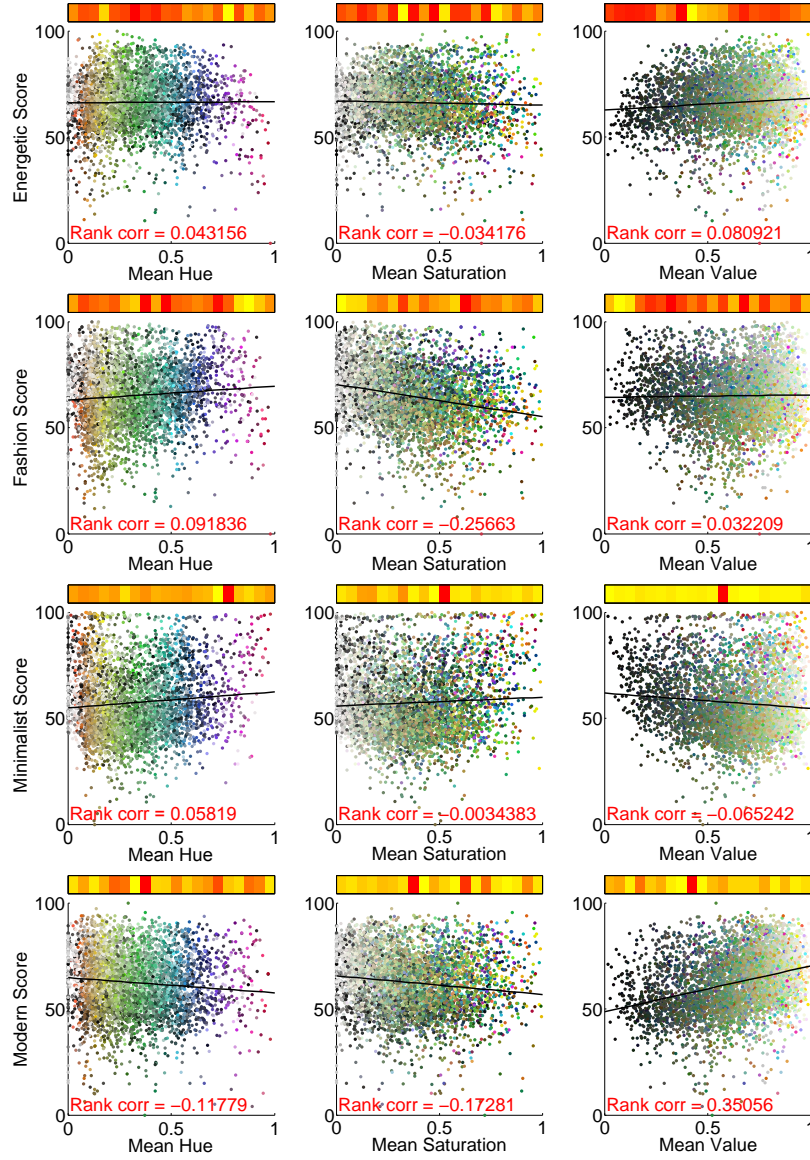


Figure 15: Correlation between HSV colors and personalities.

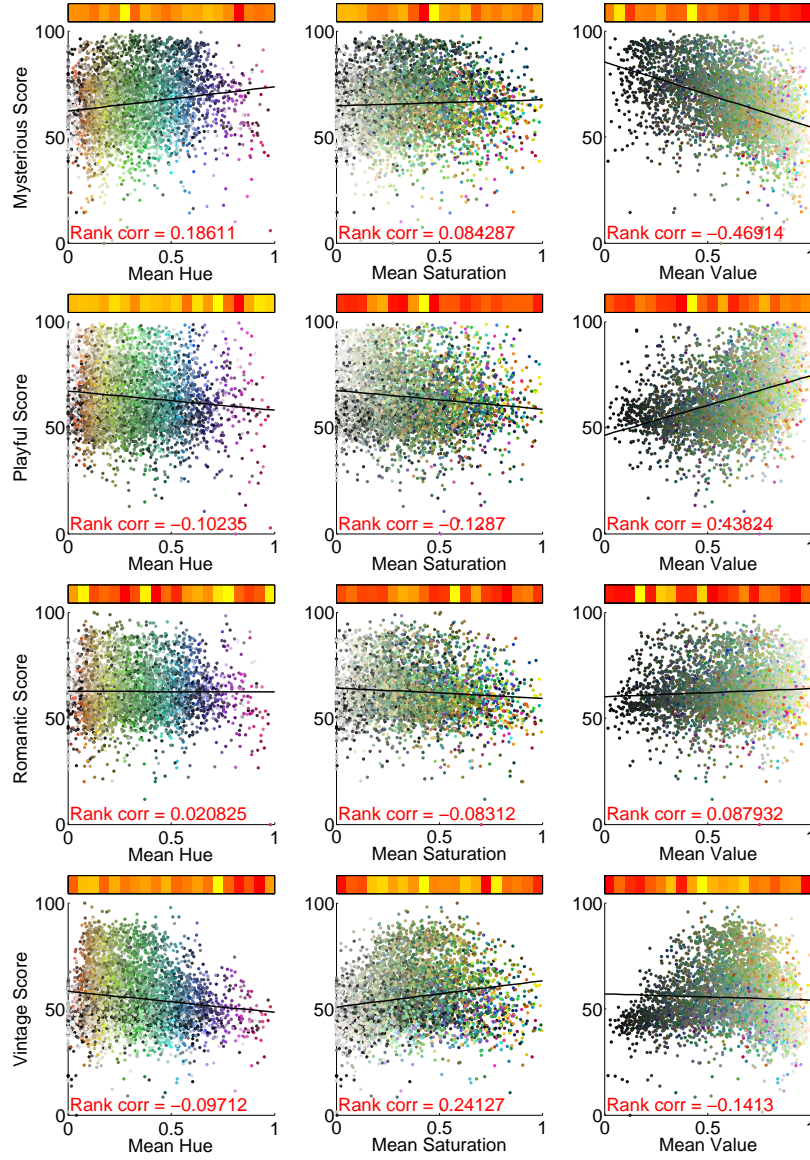


Figure 16: Correlation between HSV colors and personalities.

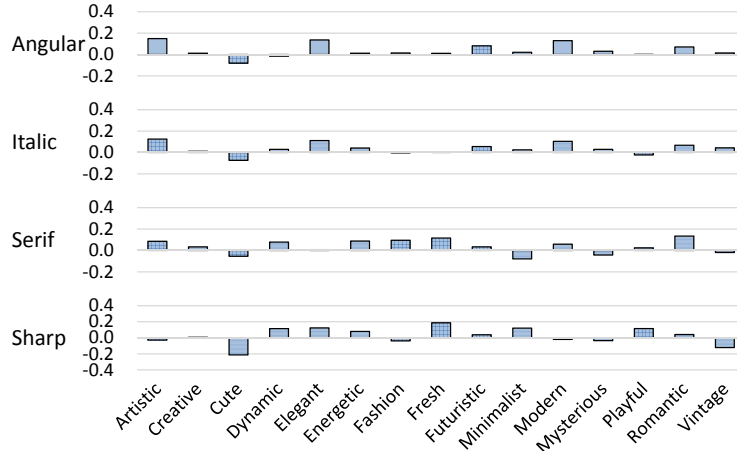


Figure 17: Correlation between font properties and graphic design personalities. For each font property (“Angular”, “Italic”, “Serif”, or “Sharp”), the bars denote the average rank correlation coefficient between the property and all personalities.

S 4 Element-level Design Suggestion

4.1 Candidate Value Generation and the Iterative Sampling Method

For text color selection, we sample 16 values along each of the RGB channels uniformly. For image enhancement, we sample 20^4 values in a 4-dimensional space spanned by saturation, brightness, contrast and sharpness, with 20 values sampled uniformly along each dimension. For image cropping task, the input image is first resized so that its smaller dimension matches the cropping region. We then rescale the image from 100% to 150% with an interval of 5%, and slide the cropping region across the image with a stride of 5 pixels at each scale to produce the candidate cropped regions. However, exhaustively evaluating all the values in a candidate set is still expensive. For computational efficiency, we adopt the iterative sampling approach as follows. We start with a complete candidate set. At each iteration, we uniformly sample 200 values from the candidate set, and calculate the personality scores of the sampled values, which are then removed from the candidate set. Our method stops when the average score of the top 10 already-sampled values is smaller than a threshold (0.5) or when the maximum number of iterations (20) is reached. The top 10 already-sampled values will then be ranked and returned.

4.2 Details of the Evaluation Study

Comparisons to baselines. For each comparison, two results were displayed side-by-side in random order. The participants were asked to select one that would better manifest a given personality. Each HIT contains 36 comparisons with 5 duplicated to test worker consistency. Each comparison was evaluated by 6 different workers. Note that it is possible for two or three of the methods to suggest the same element design, e.g., same font, for a design. In this case, the choices of workers are shared among the different methods.

4.3 More Results

Figures 18 to 20 show more comparison results on personality-based element-wise design suggestions generated by Manual (novices without prior training on graphic design) and our method. Figures 21 to 22 show more comparison results on personality-based element-wise design suggestions generated by professional designers and our method.

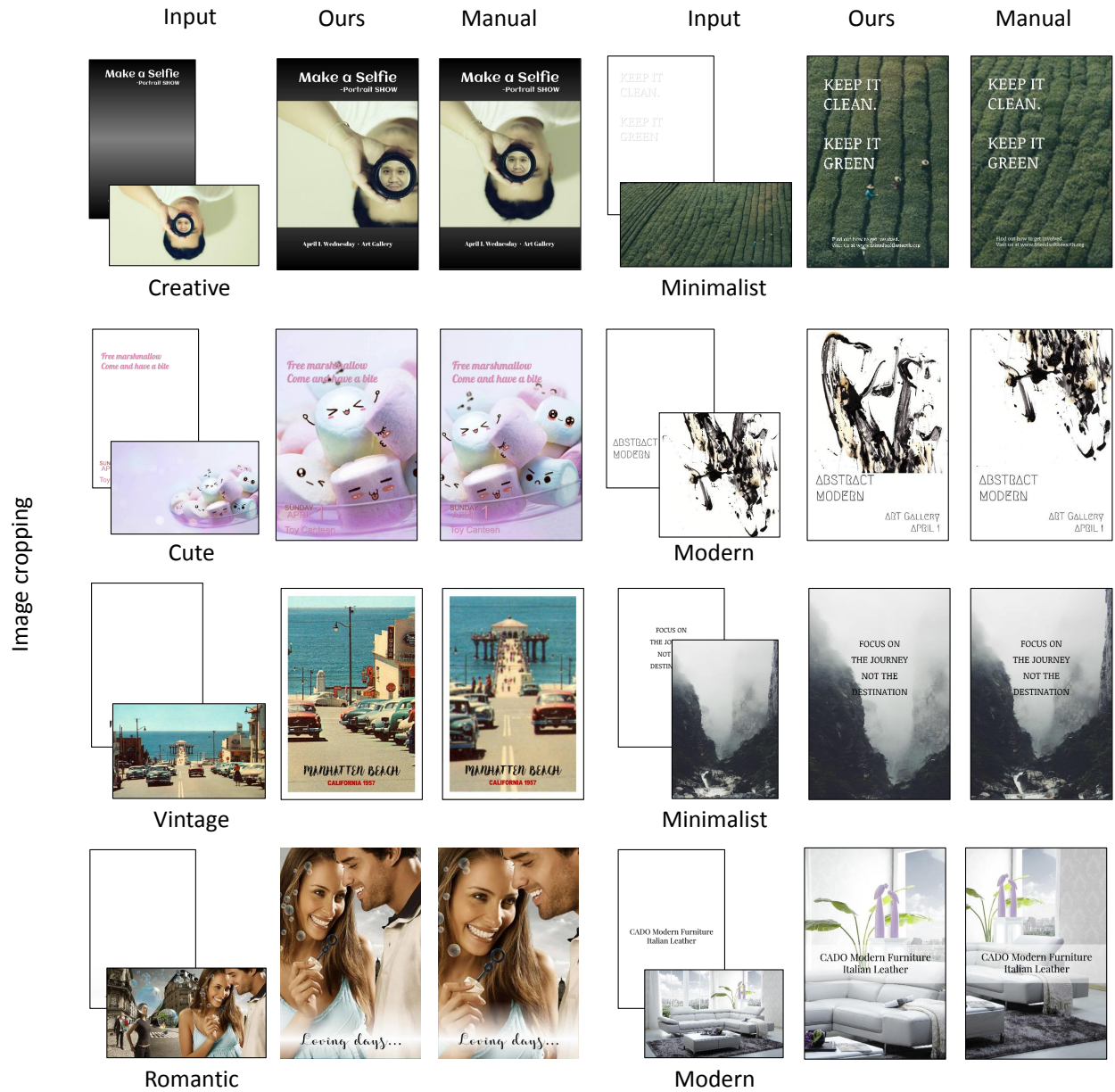


Figure 18: Element-level design suggestion results on image cropping. For each design case, we show the input design, image and personality on the left. Top results by our method (Ours) and results by novices (Manual) are shown on the right.

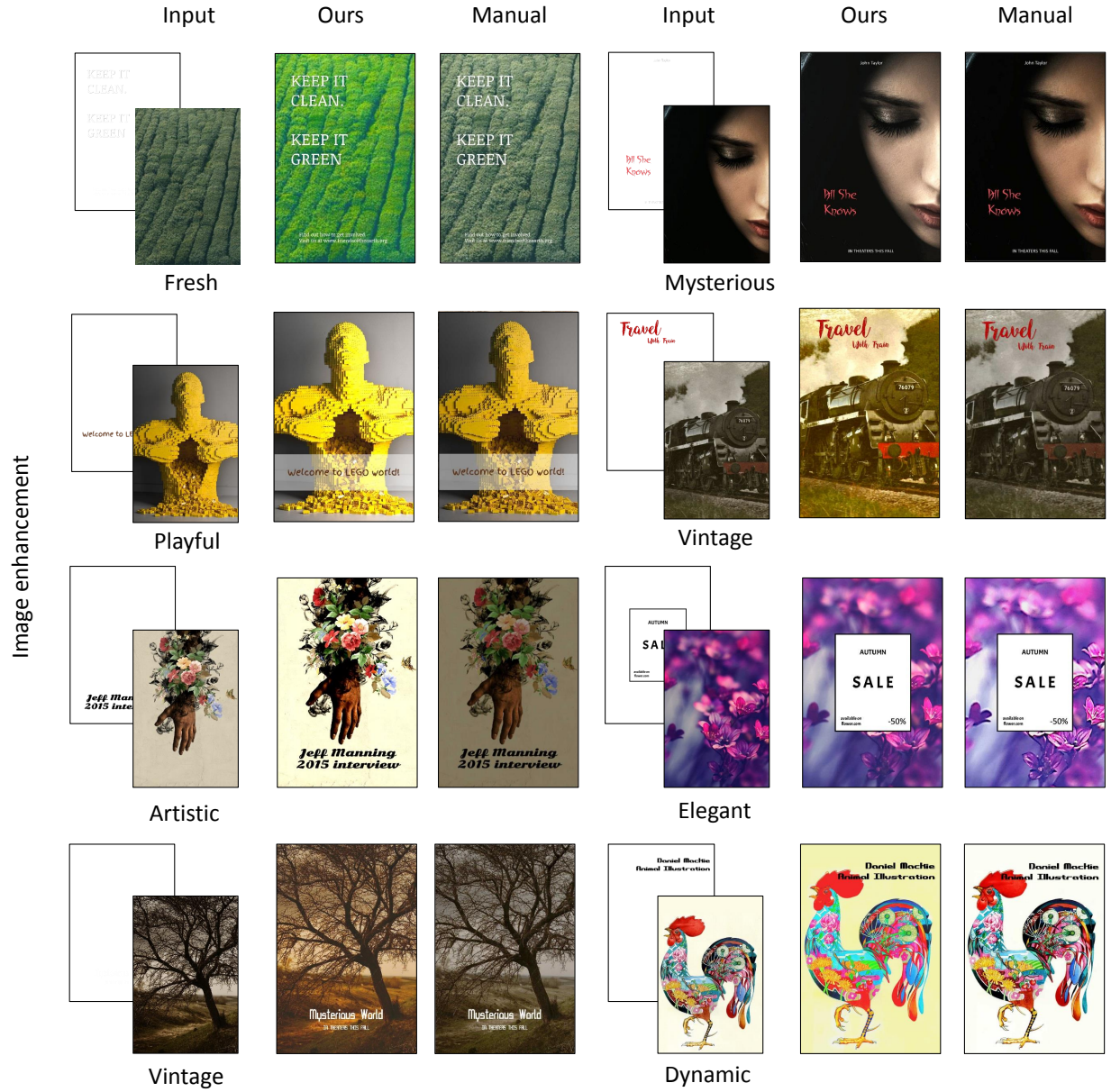


Figure 19: Element-level design suggestion results on image enhancement. For each design case, we show the input design, image and personality on the left. Top results by our method (Ours) and results by novices (Manual) are shown on the right.

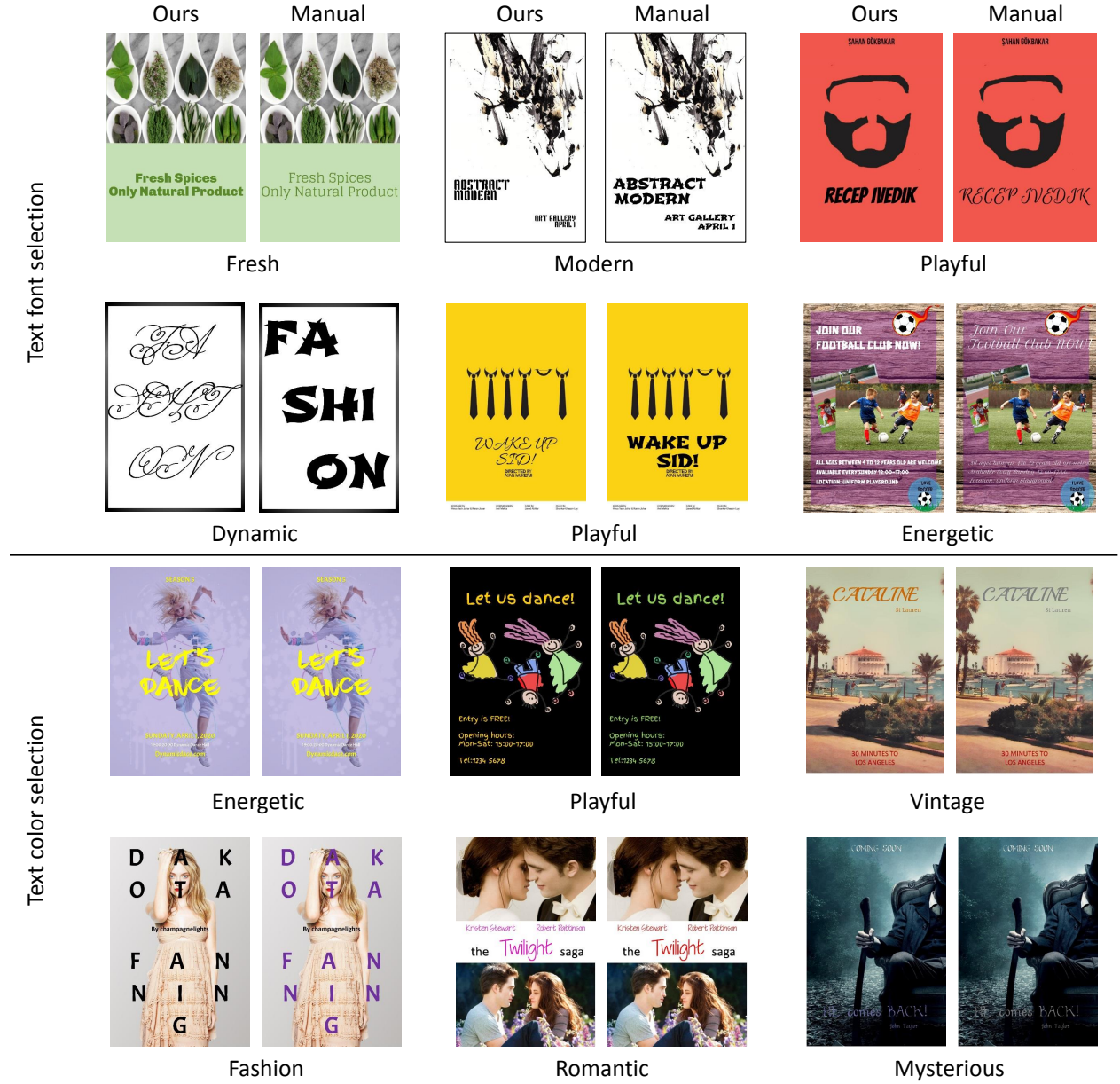


Figure 20: Element-level design suggestion results on text font and color selection. For each design case, we show the top results by our method (Ours) and results by novices (Manual).



Figure 21: Element-level design suggestion results on image cropping and image enhancement. For each design case, we show the input design, image and personality on the left. Top results by our method (Ours) and results by professional designers (Artist) are shown on the right.

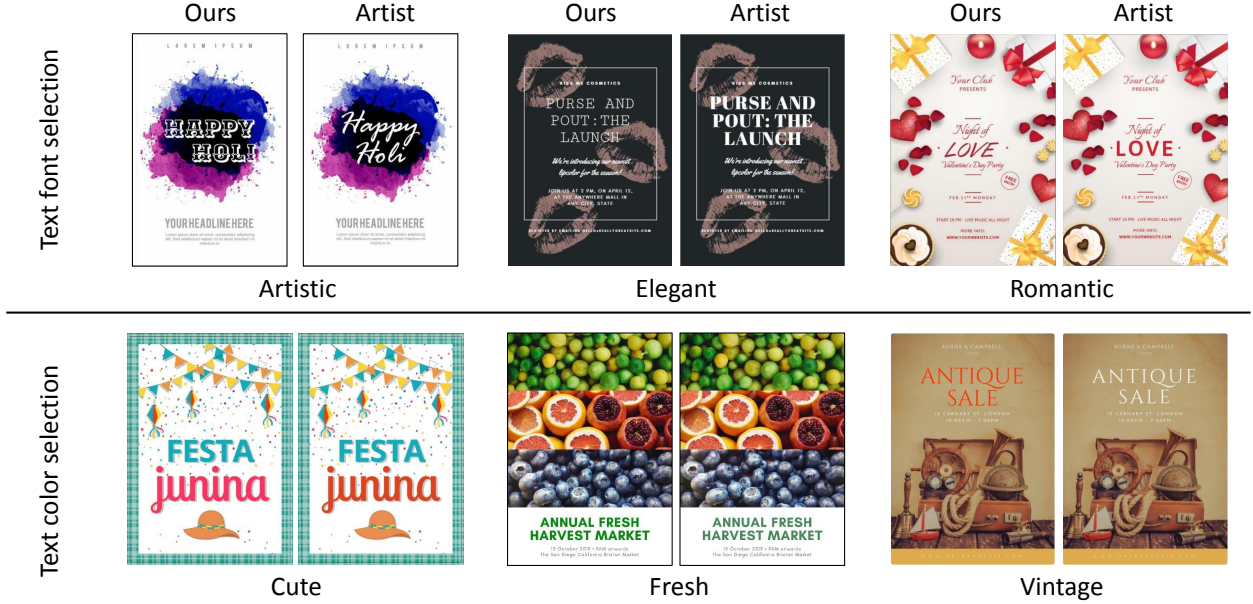


Figure 22: Element-level design suggestion results on text font and color selection. For each design case, we show the top results by our method (Ours) and results by professional designers (Artist).

S5 The Example-based Personality Transfer Method

5.1 Representation

We define a design (i.e., poster) as a set of visual elements, including text and images. Each element is represented as a property set θ . For an image element, the property set θ includes its center position (x, y) , width w , height h , and enhancement factors e (containing four entries: saturation, brightness, contrast, and sharpness). For a text element, the property set θ includes its center position (x, y) , text size s , and text font f . A design can thus be represented as $\mathcal{X} = \{(\theta)\}$, $\theta \in \{\theta_i, \theta_t\}$, where $\theta_i = (x_i, y_i, w_i, h_i, e_i)$ and $\theta_t = (x_t, y_t, s_t, f_t)$.

5.2 Energy Terms

We formulate the design adjustment as a constrained optimization problem. Formally, let \mathcal{E} and \mathcal{X}_θ be the reference and source designs, respectively. θ is a vector encoding property configurations of all the elements in the source design. For user-specified constraints, we express them as equality/inequality relations that are abstracted as hard constraints: $\mathcal{H}(\theta) = 0$. Our personality transfer is achieved by solving the following optimization problem:

$$\arg \min_{\theta} \alpha_1 E_{trans}(\mathcal{E}, \mathcal{X}_\theta) + \alpha_2 E_{prior}(\theta) \quad s.t. \quad \mathcal{H}(\theta) = 0, \quad (1)$$

where $E_{prior}(\theta)$ is a prior term to encourage the source design to conform with the design guidelines. $E_{trans}(\mathcal{E}, \mathcal{X}_\theta)$ is a transfer term to force the source and reference designs to be close in the design feature space \mathcal{F} :

$$E_{trans}(\mathcal{E}, \mathcal{X}_\theta) = d(\mathcal{F}_{\mathcal{E}}, \mathcal{F}_{\mathcal{X}_\theta}), \quad (2)$$

where d is a cosine distance function. In our implementation, we set $\alpha_1 = 0.8$ and $\alpha_2 = 0.2$. We next discuss other energy terms in more details.

Prior term. We further add the prior term to enhance the visual aesthetics of the target design.

- *Visual balance term.* In our current implementation, we have only added the balance term E_b to make the target design more visually pleasing (i.e., $E_{prior} = E_b$), due to the importance of layout balance in graphic designs. E_b is used to penalize a large distance between the center of design and the center of gravity:

$$E_b = \max(0, |C_{\mathcal{X}} - C_d| - 0.1),$$

where $C_{\mathcal{X}}$ is the center of gravity of the design (i.e., the sum of the centers of the bounding boxes for all elements, weighted by their areas) and C_d is the center of the design. All the coordinates are normalized to $[0, 1]$ relative to the height and width of the design. In this way, $C_d = [0.5, 0.5]$. A margin of 0.1 is acceptable to allow a more dynamic configuration of the design.

Hard constraints. Designs need to conform with certain hard constraints that are derived from user-given constraints and visual design rules.

- *Fixed element constraint.* Users may specify which properties of an element can be changed. For example, a user may want to change other properties of an element but keep its layout (i.e., its size and position) fixed. By default, all properties of each element may be changed.
- *Element importance constraint.* A visual hierarchy is often employed to make sure that a design can communicate information effectively, by guiding viewers' eyes to important texts or images. Users can provide an importance value for each element (a larger value indicates that the element is more important). Here, we use a simple rule to build such a visual hierarchy by enforcing the sizes of important elements larger than the less important elements by a threshold (5 points for font size, and 1% area of smaller element for image). Note that we only compare elements of the same type, either image or text. By default, all elements have the same importance value.
- *Overlapping constraint.* We represent the area of each element that cannot be overlapped using a bounding box. The overlapping constraints are satisfied only if all of the bounding boxes have no overlapping with each other. By default, all elements cannot be overlapped.
- *Visibility constraint.* In order to make sure all the elements are visible to viewers. Text elements must be completely inside the design, and the interest areas of image elements are also required to be completely inside the design. Interest areas of an image are specified by the user using a bounding box. By default, all elements must be completely inside the design. Besides, the sizes of elements must be larger than a threshold (5 points for size of a font and 400pixels² for the area of an image in our implementation).
- *Text readability constraint.* It is desirable to avoid text elements that have a similar color as their background. Thus, a solution is invalid if it causes the text color to be similar to the average color of the neighborhood of the text, measured by Euclidean distance in the LAB color space.

5.3 Optimizer

Since our objective function is highly multi-modal, we use the Metropolis-Hastings algorithm [MRR⁺53, Has70] to iteratively sample the design until a maximum number of iterations (i.e., 5000) is reached or the change in $E(\mathcal{X})$ between two consecutive iterations is smaller than 0.001. In each iteration, we maintain a design sample \mathcal{X} , and generate a new sample \mathcal{X}' by randomly selecting from several proposal moves:

- Update the position of a single element. A Gaussian noise is added to the position of a random element.
- Update text size. A Gaussian noise is added to the size of a random text element.
- Update image size. A Gaussian noise is added to the height of a random image element. The corresponding width is updated to keep the image aspect ratio.
- Update text font. Replace the text font of a random text element with a random font from a specific list. The list is generated by selecting 300 fonts randomly from the Google Font list.
- Update image enhancement factors. Replace the enhancement factors of a random image element with random enhancement factors from a specific list. The list contains enhancement factors uniformly sampled from the whole solution space (see S4.1), and some extreme cases are deleted from the list. For example, a very high value for brightness will turn the design into a blank image, which is undesirable.
- Align two elements. A random element is adjusted to align with another random element along a random axis (i.e., horizontal center, vertical center, left edge, right edge, top edge, bottom edge), by updating its position.
- Swap two elements. The position of two random elements are swapped.

All the proposal moves are equal likely to be selected. In addition, we ignore the moves that violate any one of the hard constraints mentioned above.

5.4 More Results

Figure 23 show more example-based personality transfer results generated by our method.

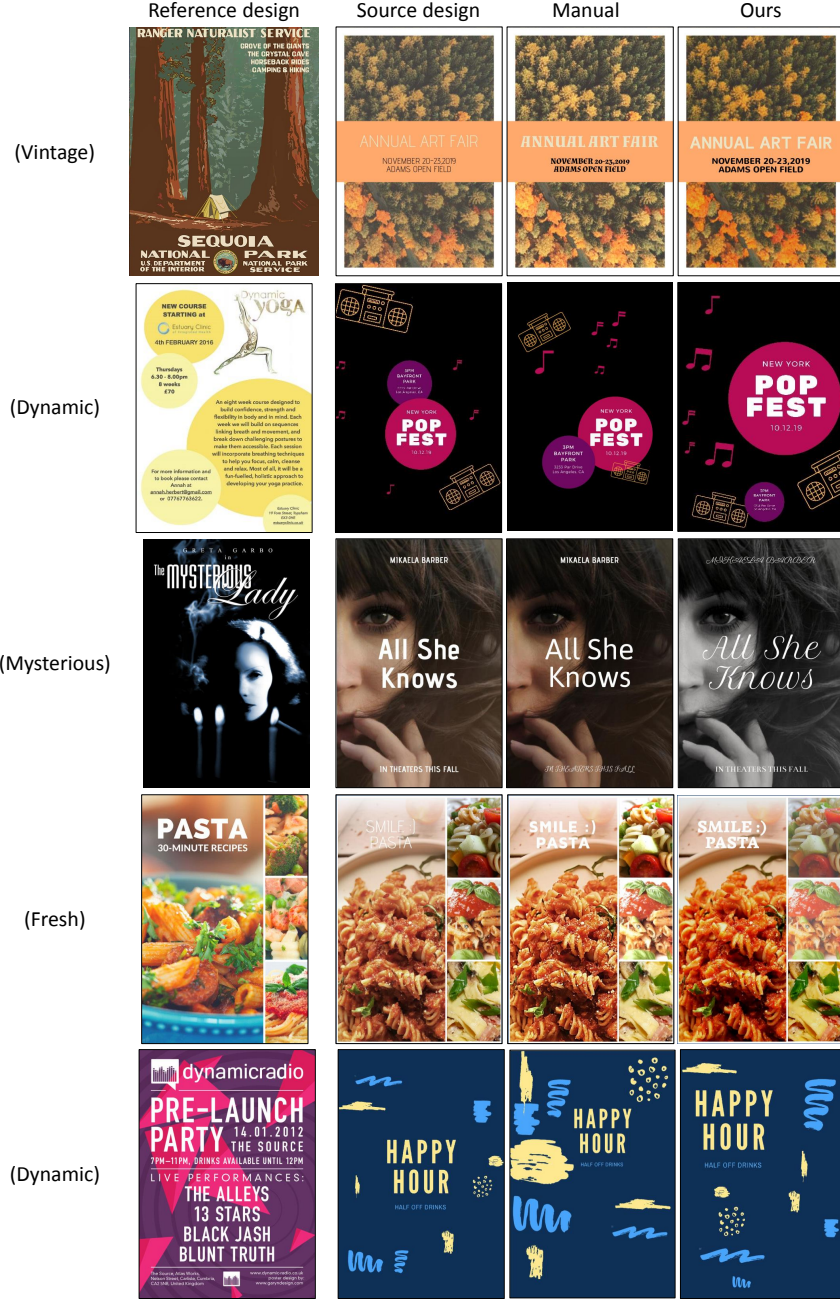


Figure 23: Example-based personality transfer. Given a reference design and a source design, our method can generate a plausible new design that matches the personality represented by the reference design. We also show the results generated manually by novices for comparison. To make the comparisons more clear, we list the personalities of reference designs on the left.

References

- [Has70] Keith Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.
- [MH08] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.
- [MRR⁺53] Nicholas Metropolis, Arianna Rosenbluth, Marshall Rosenbluth, Augusta Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953.