APPENDIX A
PREDICTING GRAPHIC DESIGN IMPORTANCE

A.1 Crowdsourced Design Importance

We performed an MTurk study asking users to label important regions in a design. We did not provide an explicit definition of importance, but instead showed 7 example labelings. 1,075 graphic designs were shown to 35 MTurk users who labeled the most important regions. See the supplementary material for the MTurk study materials. Users were paid 30¢ for labeling each set of 20 designs, and 758 MTurk users completed the study. Duplicate designs were added randomly and inconsistent users were removed. These individual importance maps are often quite noisy, with significant variation between users. However, one important result of this study is that, averaged over a large number of users (20-30), the mean importance maps often give a plausible ranking of importance. See Fig. 1 for an example of individual and averaged maps.

Fig. 1: **Importance Maps.** Given a design (top left), MTurk users mark what they consider important. While the individual maps are noisy, when averaged over 20-30 users (bottom right), the mean maps are reasonable.

A.2 Features

Our features includes crowdsourced labeling of people, faces, and text. Fig. 2 shows some examples of these features. When creating features, efficiency is a concern since the model is used as part of the energy function for optimizing designs. A feature set which requires many seconds to compute is not practical. However, we also experimented with more time-consuming features including other saliency detectors [6], [7], [3], steerable pyramid filters [10], and object detectors [1] for more accurate modeling of importance. In the next section we describe results using both the fast and full model. In practice, we found the fast feature set worked well, and was used for the model in the accompanying paper.

Fig. 2: **Crowdsourced Features.** We use crowdsourcing to extract high-quality features including people and face detection, and text size. Designs courtesy of Natasha Mileshina and Ben Keenan.

A.3 Prediction Results

The model computes features $y(\mathbf{p})$ for each pixel $\mathbf{p}$ and predicts importance $r(\mathbf{p})$ using a linear regression of the features: $r(\mathbf{p}) = \mathbf{w}^T\mathbf{y}(\mathbf{p}) + b$. Parameters $\mathbf{w}$ and b are learned with LASSO [11], using L_1 regularization:

$$\min_{\mathbf{w},b} \sum_i (\mathbf{w}^T\mathbf{y}_i + b - r_i)^2 + \lambda||\mathbf{w}||_1 \quad (1)$$

where r_i is the mean importance from MTurk users. The optimal parameters $\mathbf{w}$ and b are computed via a convex optimization [2] (glmnet package), with $\lambda = 0.00037$ selected by cross-validation.

Table 1 compares our two models and existing image saliency models, using 10-fold cross-validation with a $0.9/0.1$ random train/test split of our 1,075 designs, with 1,000 pixels randomly sampled from each image. Fig. 3 shows a few example designs from the test set. We report the root-mean-square error (RMSE) for our predictor, as well as the R^2 coefficient where 1 is a perfect predictor and 0 is the baseline of simply predicting the mean importance value $\bar{y}$.

$$R^2 = 1 - \frac{\sum_i (y_i - f_i)^2}{\sum_i (y_i - \bar{y})^2} \quad (2)$$

These results show that existing image saliency models poorly predict the human-created importance maps. This result is unsurprising since these methods predict eye-fixations, not importance. Existing saliency methods are also designed for natural images, so they fail to capture text importance. Both our feature sets work quite well at predicting human importance, with the full model performing slightly better.

Fig. 3: **Design Importance.** Given an design from a test set, we show the mean MTurk importance map, the saliency model of Goferman [3], and our full and fast importance models. The fast model is used in the accompanying paper. Designs courtesy of William Berry, Dániel Perlaky, and Ben Keenan.

	Fixed	IK	HZ	J	G	Full	Fast
RMSE	0.304	0.295	0.253	0.280	0.251	**0.155**	**0.165**
R^2	0	0.054	0.306	0.154	0.318	**0.739**	**0.702**

TABLE 1: Comparison of Saliency and Importance Models. (IK) Itti and Koch [6], (HZ) Hou and Zhang [5], (J) Judd et al. [7], (G) Goferman et al. [3] (Full/Fast) our importance modeling approach

	Text	FP	Sal	TFP	G+TFP	Full
RMSE	0.223	0.297	0.231	0.200	0.175	**0.155**
R^2	0.462	0.045	0.426	0.569	0.669	**0.739**

TABLE 2: Comparison of Feature Sets. (Text) Text, (FP) Face and Person, (Sal) All Saliency, (TFP) Text, Face, and Person, (G+TFP) Goferman et al. [3] and Text, Face, and Person, (Full) All Features

To evaluate the extent that particular features contribute to perceived importance, we trained the regression model with different subsets of features. Table 3 compares the results using text, face and person, and saliency features. We also combined our crowdsourced features with the best-performing saliency measure [3]. These results suggest that text features and salience both play a large role in visual importance. This is unsurprisingly since designers often make important elements more salient or eye-catching.

Appendix B
Hierarchical Segmentation

Designers often use grids or rectangular regions to organize elements. A viewer perceives this structure and relates alignment, grouping, and symmetry to these regions. Our system estimates the layout structure and calculates energy terms based on it. The algorithm takes as input the layout, binary masks for each element, and element classes (graphic or text); the output is a hierarchical segmentation of the design into non-overlapping rectangular regions.

The proposed algorithm segments a design by vertically or horizontally splitting regions which contain both text and

a b c

Fig. 4: The algorithm segments a design into rectangular regions with three criteria. First, segmentation boundaries, or *cuts*, should avoid intersecting elements. (a,b) show the intersection penalty for horizontal and vertical cuts; a cut placed near an element center would pay a high penalty. Second, regions should contain only text or graphics. Third, cuts should lie near the parent region's center. (c) shows a final segmentation.

graphics. Each split is evaluated using a cost function which measures the intersection of the split with elements, the separation between graphic and text elements, and distance to the region center. The algorithm recursively segments regions until a region contains only elements of the same class, or a user-specified maximum depth is reached. Lastly, empty or adjacent regions with the same element classes are merged. Figure 4 illustrates the horizontal and vertical intersection penalties, and the final segmentations.

Appendix C
Graphic Design Model

The model includes many different terms. In Fig. 5 we help justify our terms by showing examples of training the model with various terms removed. We next discuss the various model terms in more detail.

C.1 Alignment

Correct alignment is an important aspect of good design; poor alignment is both distracting and confusing. The model prefers that elements align with each other. One energy term measures the fraction of element pairs that align with a type a such as Left or Top:

$$E^a_{align} = -\mathcal{S}\left(\frac{1}{n^2}\sum_{i\in(all)}\left(N^a_i + \sum_{j\in(all)} I^a_{ij}\right); \alpha_a\right) \quad (3)$$

where n is the number of elements and I^a_{ij} indicates if element i and j are aligned by type a. We define separate energy terms for different alignment types. N^a_i indicates

Fig. 6: Element Alignment Labeling. We show the x-axis alignment labels for a set of center and nearly left-aligned elements. Elements align if the differences between their bounding box edge or center are less than a threshold. Note that the misaligned elements on the right are still labelled as aligned. The model will then penalize these misalignments. Elements do not initially align if other elements lie between them. For example, elements a and c, and d and f are not initially aligned. These elements are then combined into an **alignment group** so that $I^{\text{X-Center}}_{ac} = 1$ and $I^{\text{Left}}_{df} = 1$.

if element i is internally aligned by type a. For example, the date in Fig. 9 has an X-Center internal alignment. Each feature is transformed by a sigmoid $\mathcal{S}(x;\, \alpha_a)$ (Fig. 8) Fig. 6 illustrates the alignment labelling. **[P: new]**

The model penalizes misalignments with following term:

$$E^x_{misalign} = \frac{1}{3n^2}\sum_a \sum_{i\in(all)} \sum_{j\in(all)} I^a_{ij} C(d^a_{ij}) \quad (4)$$

where $C(d^a_{ij})$ is a robust cost function for a given distance which heavily penalizes slight misalignments: $C(d) =$

Fig. 5: **Model Terms.** Given the two training layouts, the system attempts to learn the style with different energy terms removed.

$5\arctan(\frac{d}{0.015})$. d is the distance in pixels normalized by the design size. $E^{y}_{misalign}$ is defined similarly for y-axis alignments.

The model also encourages larger alignment groups:

$$E_{agroup} = -\mathcal{S}\left(\frac{1}{nm}\sum_{g}\sum_{i\in(all)} I^{i}_{g};\ \alpha_{ag}\right) \tag{5}$$

where n is the number of elements, m is the number of alignment groups and I^{i}_{g} indicates if element i belongs in alignment group g. See Fig. 7 for examples. **[P: new]**

An alternative to this approach would be to include the alignment detection when evaluating this energy term, as in Vollick et al. [13]. However, there are a few advantages to separating the analysis step into a separate module. This separation allows a simpler energy function. Alignment detection is also used in the optimization step to reduce misalignments.

C.2 Balance

We measure the global symmetry using a binary map of text or graphical elements, flipped along an axis. More precisely,

Fig. 7: Alignment Groups. Rectangle colors indicate the detected alignment groups, with the orientation and position of the rectangles indicating the alignment type. Colored connector lines groups the elements, with the deviation of the rectangle from the connector line indicating the misalignment. The top design illustrates a perfectly aligned group (in blue) and misalignment of the bottom and center elements (in red and green). The bottom design illustrates that elements cannot group with inconsistent internal alignments (e.g., the center-aligned address), and that elements cannot group with unaligned elements between them.**[P: new]**

for vertical or x-axis symmetry :

$$S^{text}_{x-symm} = \sum_{c\in(text)} \left(\frac{\sum_p |I^c_p - I^c_{\text{flip}(p,x,G)}| I^c_p}{\sum_p I^c_p} - 1 \right) \quad (6)$$

$$E^{text}_{x-symm} = \mathcal{S}(S^{text}_{x-symm}; \alpha_{txs}) \quad (7)$$

where I^c_p is a binary variable indicating if pixel p is of class $c \in (graphic, text)$, and $\text{flip}(p, x, G)$ indicates the symmetric counterpart of pixel p along the x-axis of image G. We define a similar term E^{text}_{y-symm} with $\text{flip}(p, y, G)$ indicating y-axis symmetry. We also define an asymmetry term $E^{text}_{x-asymm} = \mathcal{S}(S^{text}_{x-symm} - 1; \alpha_{txa})$, and identical terms for graphical elements, giving 8 global symmetry terms total.

Designers often symmetrize elements not with the overall page, but with regions of the page (Fig. 9). Given the hierarchical segmentation of Sec. 4, we define $E^{text}_{regionSymm}$ as above, using the symmetric counterpart of pixel p along the x-axis in region r.

Fig. 8: Sigmoid function $\mathcal{S}(x; \alpha)$ used to shape energy terms. The parameter α controls the smoothness of the step function.

Fig. 9: Symmetry Types. **Global symmetry** is defined over the entire design; **region symmetry** is defined with respect to the segmentation regions (show with red lines). The top element has both symmetry types; the elements in the two bottom regions have only region symmetry.

C.3 Emphasis

The model also matches the perceived importance of elements to their desired importance. Using the method of Sec. 4.1, the system first estimates the hidden variable q_i, the perceived visual importance of element i. Estimated values are compared to fixed scalar importance values F^p_i for each element using Pearson correlation:

$$E^{text}_{emp} = -\mathcal{S}(-corr(\mathbf{q}, \mathbf{F}^p); \alpha_{emp}) \quad (8)$$

where $corr(x, y)$ is the Pearson correlation between paired samples x and y. $\mathbf{q}$ and $\mathbf{F}^p$ are vectors of the perceived and desired importance values for all elements. $E^{graphical}_{emp}$ is defined similarly for graphical elements.

The desired importance values are provided as meta-data during the design creation process and are not estimated by the system. In practice, these values are usually simple to specify. White [14] recommends designers establish an importance hierarchy of at most 3 levels, as more can become confusing. In our examples we usually specify 3 levels of importance: low, medium, and high, though occasionally we used 4 levels.

C.4 White Space

The model encourages white space with the following term:

$$E_{whiteSpace} = -\mathcal{S}\left(\frac{\sum_p I_p^e}{wh};\, \alpha_{ws}\right) \tag{9}$$

where I_p^e is a binary variable indicating if pixel p is not covered by an element, and w and h are the design width and height.

The model penalizes large regions of empty white space using the cubed distance to an element over the entire image:

$$E_{spread} = \mathcal{S}\left(\frac{1}{wh}\sum_p \min_i (D(p, M_i)^3);\, \alpha_{spr}\right) \tag{10}$$

where $D(p, M_i)$ is the Euclidean distance of element M_i to pixel p.

The model encourages separation between elements. The distance between elements i and j is denoted as $\hat{d}_{ij}$, the minimum of the mean squared distance between elements:

$$\hat{d}_{ij} = \min_{p \in I} \sqrt{\frac{1}{2}(D(p, M_i)^2 + D(p, M_j)^2)} \tag{11}$$

To avoid computationally-expensive distance transforms during optimization, distance maps from the element boundary are pre-computed for an area around each element and scaled. If the maps for two elements do not overlap, the bounding box distance is used.

The distance energy is the mean of the nearest distance for each element:

$$E_{dist} = \frac{1}{n}\sum_{i \in (all)} \left(1 - \mathcal{S}\left(\min_{j \in (all)} (\hat{d}_{ij});\, \alpha_{dist}\right)\right) \tag{12}$$

$E_{textDist}$ is defined similarly for text elements.

The model encourages uniform vertical spacing of text elements:

$$E_{textSepVar} = \mathcal{S}(\text{var}(\cup_{i,j \in olText} d_{ij}^y);\, \alpha_{tsm}) \tag{13}$$

where $\text{var}(\mathbf{x})$ is the variance of a set $\mathbf{x}$, $\cup_{i,j \in olText}$ indicates all pairs of text elements that overlap along the x-axis, and there are no elements between them. d_{ij}^y is the vertical bounding box distance.

Border margins M_i^k for each element are the distance of the bounding box edge to the respective boundary. The energy is the mean of the nearest margin distances:

$$E_{margin}^{text} = \frac{1}{n}\sum_{i \in (text)} (1 - \mathcal{S}(\min_k (M_i^k);\, \alpha_{tm})) \tag{14}$$

$E_{margin}^{graphic}$ is similarly defined for graphical elements.

C.5 Scale

Element scale is an important practical and stylistic decision. Elements must be large enough to view, but not so large that the design becomes cluttered and aesthetically displeasing. For a given layout, the size of a text element M_i^s is defined as the element height M_i^h divided by the number of lines F_i^l, weighted by a scaling parameter τ_s, and normalized by the design height h. That is, $M_i^s = (\tau_s M_i^h)/(F_i^l h)$, For almost all examples in this paper, $\tau_s = 1$. Using a smaller value, for example $\tau_s = 0.4$, will increase the element sizes. The size for graphical elements is the bounding box area $(M_w^i M_h^i)/(wh)$ where M_w^i and M_h^i are element's bounding box width and height, and w and h are the design width and height. We encourage larger text with the following term:

$$E_{textSize} = -\frac{1}{n_t}\sum_{i \in (text)} \mathcal{S}(M_i^s;\, \alpha_{ts}) \tag{15}$$

where n_t is the number of text elements; $E_{graphicSize}$ is defined similarly.

Energy terms also penalize the variance of text and graphic sizes:

$$E_{textVar} = \mathcal{S}(\sigma(\cup_{i \in (text)} M_i^s);\, \alpha_{tv}) \tag{16}$$

The model prefers elements to be above a minimum size:

$$E_{minTextSize} = \sum_{i \in (text)} \max(\tau_t - M_i^s, 0) \tag{17}$$

$E_{graphicVar}$ and $E_{minGraphicSize}$ are defined similarly. The minimum sizes for text and graphics were set to $\tau_t = 0.0275$ and $\tau_g = 0.04$.

C.6 Overlap and Boundaries

Overlapping elements are common in many designs. We define several types of overlap, including overlap of elements on text, overlap of text on graphical elements, and overlap of graphical elements on other graphical elements. Separate energy terms are defined for each overlap type.

$$O_{textOverlap} = \frac{\sum_p A_p^t}{wh} \tag{18}$$

$$E_{textOverlap} = \mathcal{S}\left(O_{textOverlap};\, \alpha_{to}\right) \tag{19}$$

where A_p^t indicates the alpha component of any element overlapping any text element. We define $E_{graphicTextOverlap}$ and $E_{graphicGraphicOverlap}$ similarly to indicate text overlapping graphical elements, and graphical elements overlapping each other.

For graphical elements, users may also provide fixed binary masks as meta-data, to prevent overlapping important regions such as faces or logos. The model includes an energy term which measures the overlap in these regions:

$$O_{impOverlap} = \frac{\sum_p F_p^{no} A_p^g}{wh}; \tag{20}$$

$$E_{impOverlap} = \mathcal{S}(O_{impOverlap}\alpha_{no}) \tag{21}$$

where F_p^{no} indicates if the binary mask has been drawn on pixel p, and A_p^g is the alpha component of any overlapping element.

Lastly, the model penalizes text overlapping graphical elements which is hard to read:

$$E_{textContrast} = \frac{1}{n_t} \sum_{i \in (text)} \mathcal{S}(M_i^{con}); \alpha_{tc}) \tag{22}$$

where M_i^{con} is a text contrast measure defined as the difference between an element and the design before the element was drawn. The mean difference for each vertical line of the difference images are sorted, and the mean of the worst 20% used. This measure produces a high penalty even if a small part of the text has low contrast.

The model controls how much elements may extend past the boundaries of the design:

$$B_{graphic} = \frac{1}{n} \sum_{i \in (graphic)} \left(1 - \frac{\sum_{p \in i} A_p I_p}{\sum_{p \in i} A_p}\right) \tag{23}$$

$$E_{graphicBoundary} = \mathcal{S}(B_{graphic}; \alpha_{tg}) \tag{24}$$

where $\sum_{p \in i} A_p$ denotes the sum of the alpha values for all pixels in element i. $\sum_{p \in i} I_p A_p$ denotes the sum of alpha values which are within the design boundaries. A similar energy term for text and important regions $E_{impBoundary}$ is defined.

C.7 Flow

A good design layout presents information in a clear read-order. However, modeling the visual flow of graphic designs is a difficult open problem. To address this issue, our model uses simple positioning heuristics. First, more important text elements are placed higher and to the left of less important elements:

$$E_{flowX} = \mathcal{S}\left(\sum_{i \in (all)} \sum_{j \in (all)} L_{ij} d_{ij}; \alpha_{flowX}\right) \tag{25}$$

$$L_{ij} = \begin{cases} \max(F_j^p - F_i^p, 0) & \text{if } M_i^f \leq M_j^f \\ \max(F_i^p - F_j^p, 0) & \text{otherwise} \end{cases} \tag{26}$$

where M_i^f is the left boundary or center of element M_i, depending on which is closest between the two elements. The difference in desired importance $F_j^p - F_i^p$ is weighted by the two elements' bounding box distance d_{ij}. Similar terms are defined for the y-axis. See Fig. 10 for an example.

The overall location of graphical and text elements also affects the design style. The model uses a simple positioning heuristic using the mean of the element centers:

$$E_{x-location}^{text} = -\frac{1}{n_t} \sum_{i \in (text)} \frac{M_i^{x-center}}{w} \tag{27}$$

We also define a reverse term by calculating the mean of $1 - M_i^{x-center}/w$, similar terms for the y-axis, as well as terms for graphical elements, giving 8 total.

Fig. 10: The model uses simple heuristics for specifying read-order. Top: unimportant elements are above and to the right of more important elements. Bottom: corrected design.

We also measure the variance of the element positions for both text and graphical elements:

$$E_{x-var}^{text} = \mathcal{S}\left(\text{var}\left(\cup_{i \in text} \frac{M_i^{x-center}}{w}\right); \alpha_{x-var}\right) \tag{28}$$

C.8 Unity

Another important design principle is unity, when elements appear to belong together. We model element unity by allowing users to group elements with scalar group IDs provided as meta-data. The model encourages group members to have a similar size:

$$E_{groupSizeVar} = \frac{1}{|G|} \sum_{g \in G} \text{var}(\cup_{i \in g} M_i^s) \tag{29}$$

where $|G|$ is the number of user groups, $\cup_{i \in g}$ indicates all the elements in a group g, and M_i^s is the size of element i.

Group members are encouraged to have a similar perceived importance. $E_{groupImpVar}$ is defined as above using q_i. We enforce a weak position constraint that group members should be close:

$$E_{groupDistMean} = \frac{1}{|G|} \sum_{g \in G} \frac{1}{n_g} \sum_{i \in g} \min_{j \in g}(d_{ij}) \tag{30}$$

where n_g is the number of elements in group g, and d_{ij} is the bounding box distance between elements.

C.9 Previous Layout

When improving or retargeting designs, the model uses a previous layout of the same elements. We often wish to preserve properties of the design including relative locations, sizes, and importance. Given a previous design,

the model includes the following terms:

$$E^{text}_{prevHeight} = \mathcal{S}\left(\frac{1}{n}\sum_{i\in(text)} |M^h_i - M^{oh}_i|;\ \alpha_{ph}\right) \quad (31)$$

$$E^{text}_{prevPosition} = \mathcal{S}\left(\frac{1}{n}\sum_{i\in(text)} ||M^c_i - M^{co}_i||_2;\ \alpha_{pp}\right) \quad (32)$$

$$E^{text}_{prevImp} = \mathcal{S}\left(\frac{1}{n}\sum_{i\in(text)} |q_i - q^o_i|;\ \alpha_{pi}\right) \quad (33)$$

$$E^{text}_{relHeight} = \mathcal{S}\left(\frac{1}{n^2}\sum_{i,j\in(text)} |r^h_{ij} - r^{oh}_{ij}|;\ \alpha_{pr}\right) \quad (34)$$

where M^h_i and M^{oh}_i are the current and original heights of element i, M^c_i and M^{co}_i are the relative positions of the element centers in the current and original design, and q_i and q^o_i are the current and original perceived importance values. $E^{text}_{relHeight}$ enforces relative differences in heights: $r^h_{ij} = M^h_i - M^h_j$. Similar terms are defined for graphical elements.

The model also compares two global properties of the designs, overlap and elements beyond the boundaries:

$$E_{prevOverlap} = \mathcal{S}(|O^c_{gt} - O^o_{gt}|;\ \alpha_{po}) \quad (35)$$

$$E_{prevGraphicBoundary} = \mathcal{S}(|B_c - B_o|;\ \alpha_{pg}) \quad (36)$$

where O^c_{gt} and O^o_{gt} measure text overlapped on graphical elements in the current and original designs, and $B^c_{graphic}$ and $B^o_{graphic}$ measure the fraction of graphical elements which extend beyond the boundary. See Sec. C.6 for details on these properties.

Appendix D
Simulated Annealing Proposals

Our optimization uses several different proposals to deal with the complexity of our function and dependencies between elements:

Update Single Element Position. For updating the position, a normally distributed offset is added to the current position ($\sigma_{loc} = 0.1$), elements are moved along an axis to fixed positions, or elements moved to an empty part of the design.

Update Height. Element heights are updated by adding a normally distributed offset to the current height ($\sigma_{hei} = 0.2$).

Align Elements. Element align with another on a single axis, either with a single alignment type, or on all three (i.e., by changing the height or width as well).

Swap Two Elements. The position of two elements are swapped.

Update Element Group. If the user specifies an element group, height and position changes are proposed for the entire group, since these heights and positions are correlated in the energy function.

Switch Alternate. If an element has alternates, this proposal will randomly switch to one of the alternates.

Update Alignment Group. If the alignment labeling has detected an alignment group, the entire group's position is shifted by a normally distributed offset ($\sigma_{aloc} = 0.05$).

Reduce Alignment Error. Given a detected alignment, the current misalignment error is reduced. The direction of minimum error is simply $\mathbf{x} = \mathbf{Cb}$ where is $\mathbf{C}$ is an $2n \times 2n$ matrix indicating if elements i and j are aligned along the x and y axes, and $\mathbf{b}$ is the alignment difference. n is the number of elements. A line-search is performed along $\mathbf{x}$ to choose the state with lowest energy.

Fill Image. Some designs have very large graphics, one proposal scales a graphical element to match the design height or width.

Proposals types are all equally likely to be selected except for the *Fill Image* proposal, which is proposed less frequently (20%) due to it's low acceptance rate.

Appendix E
Nonlinear Inverse Optimization

NIO learns parameters based on one or more examples. Given an example layout $\mathbf{X}_T$, we assume it is optimal according to a parameter vector θ which we want to find. We express this with the following objective function:

$$G(\theta) = E(\mathbf{X}_T;\theta) - \min_{\mathbf{X}} E(\mathbf{X};\theta) \quad (37)$$

This function says that we want to minimize the difference in energy between the example layout $\mathbf{X}_T$ provided by the designer, and the optimal layout for this θ. If we find a global minimum at $G(\theta) = 0$, then we have found the θ that makes $\mathbf{X}_T$ optimal.

Given the complexity of the model, we find it useful to add a weighted prior on the parameters:

$$G(\theta) = E(\mathbf{X}_T;\theta) - \min_{\mathbf{X}} E(\mathbf{X};\theta) + \lambda \sum_i \mathbf{s}(\theta_i - \bar{\theta}_i)^2 \quad (38)$$

where $\mathbf{s}$ is a binary vector and $\bar{\theta}$ is the initialization of the parameter vector. For example, if the algorithm learns from a design with slight misalignments, this prior can still enforce that alignment errors should be penalized heavily. See the supplementary material for $\bar{\theta}$ and $\mathbf{s}$. We use $\lambda = 10$ for the retargeting and improvement applications, and $\lambda = 0$ for the design styles application. To avoid negative parameters, we re-parameterize $\theta_i = \exp(\beta_i)$ and optimize for β_i to ensure that $\theta_i > 0$.

To evaluate G(θ), we must first compute an optimal layout (the min term). We approximate this optimal layout using optimization as described in the previous section. We then minimize G using gradient descent with line search. When G is differentiable:

$$\frac{dG}{d\theta} = \frac{\partial}{\partial\theta}E(\mathbf{X}_T;\theta) - \frac{\partial}{\partial\theta}E(\mathbf{X}_S(\theta);\theta) + 2\lambda\mathbf{s}(\theta - \bar{\theta}) \quad (39)$$

$$\mathbf{X}_S(\theta) = \min_{\mathbf{X}} E(\mathbf{X};\theta) \quad (40)$$

since $\frac{\partial E}{\partial \mathbf{X}} \frac{dX_s}{d\theta} = 0$ if $\frac{dX_s}{d\theta}$ exists [13]. Intuitively, following the gradient direction has the effect of reducing the energy of the training example $\mathbf{X}_T$ while increasing the energy of the counter example $\mathbf{X}_S$, as visualized in Figure 11. When learning with multiple examples, we decrease the mean energy difference over all training examples.

NIO can be seen as a form of Contrastive Divergence learning [4]. Inspired by Persistent Contrastive Divergence [12], we persist all previous counter examples to improve performance and efficiency. At the start of each iteration, the training example and previous counter examples are checked to find the lowest energy and this layout used to initialize the optimization. Every 5 iterations, we perform a longer optimization to escape local minima.

function NONLINEARINVERSEOPT($\mathbf{X}_T, \lambda, \bar{\theta}, \mathbf{s}$)
  $\hat{\theta} \leftarrow \bar{\theta}$
  $i \leftarrow 1$
  $\mathbf{C} \leftarrow \mathbf{X}_T$
  while not done do
    if $i \bmod 5 = 1$ **then**
      Perform long optimization
      $\mathbf{X}_S \leftarrow \text{Optimize}(\hat{\theta}, [])$
    else
      Find lowest energy layout $\mathbf{X}_I$ *to start optimization*
      $\mathbf{X}_I \leftarrow \text{argmin}_{\mathbf{X} \in \mathbf{C}} E(\mathbf{X}; \hat{\theta})$
      $\mathbf{X}_S \leftarrow \text{Optimize}(\hat{\theta}, \mathbf{X}_I)$
    end if
    Add new counter example $\mathbf{X}_S$ *to working set* $\mathbf{C}$
    $\mathbf{C} \leftarrow \mathbf{C} \cup \mathbf{X}_S$
    $\Delta\theta \leftarrow \frac{\partial}{\partial\theta} E(\mathbf{X}_T; \theta) - \frac{\partial}{\partial\theta} E(\mathbf{X}_S; \theta) + 2\lambda \mathbf{s}(\hat{\theta} - \bar{\theta})$
    $\rho \leftarrow$ LineSearch$(\hat{\theta}, \Delta\theta, \mathbf{X}_T)$
    $\hat{\theta} \leftarrow \hat{\theta} - \rho\Delta\theta$
    $i \leftarrow i + 1$
  end while
end function

APPENDIX F
PORTRAIT RATIO RESULTS

In Fig. 12 we demonstrate style learning with three portait-ratio styles: a highly symmetric style with smaller elements and large margins, an asymmetric style with larger elements, and a style with higher graphics with larger, left aligned text. In Fig. 13 we show retargeting results for landscape-to-portrait retargeting.

APPENDIX G
FEATURE LIST

We next show the full set of features in our model as well as the initial weights used for NIO.

ID	Name	Init Weight
1	No Overlap Out Of Bounds	2500
2	Text Imp Pearson	50
3	Graphic Out Of Bounds	500
4	Text Height	75
5	Graphic Size	50
6-7	Graphic/Text Size Variance	50/0.2
8-9	Graphic/Text Size Constraint	125
10	Text Seperation Variance	50
11	Text Mean Contrast	750
12	Graphic & Text Mean Imp Overlap	25
13	Text Overlap Mean Worst	5
14	Graphic & Text Mean Overlap	250
15	Graphic Mean Overlap	250
16-17	Graphic X/Y Position	0.2
18-19	Graphic X/Y Position - Reverse	0.2
20-21	Text X/Y Position	0.2
22-23	Text X/Y Position - Reverse	0.2
24-25	Text X/Y Position Variance	12.5
26-27	Graphic X/Y Position Variance	12.5
28-29	Alignment Err X /Y	50
30	Alignment Bonus X - Left	37.5
31	Alignment Bonus X - Center	50
32	Alignment Bonus X - Right	12.5
33	Alignment Bonus Y - Top	50
34	Alignment Bonus Y - Center	50
35	Alignment Bonus Y - Bottom	12.5
36	Group Sizes	12.5
37-38	Region Graphic/Text X Symmetry	75
39-42	Global Graphic/Text X/Y Symmetry	50
43-46	Global Graphic/Text X/Y Symm - Reverse	0.2
47	Element Margins	75
48	Text Distances	50
49-50	Graphic/Text Border Margins	25/50
51-52	Text X/Y Flow	75/150
53-54	Graphic/Text Group Size Variance	2.5
55-56	Graphic/Text Group Imp Variance	2.5
57-58	Graphic/Text Group Dist Mean	250
59	White Space Area	0.2
60	Spread	50
61	Orig Layout Text Position Diff	125
62	Orig Layout Graphic Position Diff	75
63	Orig Layout Text Height Diff	50
64	Orig Layout Graphic Height Diff	50
65	Orig Layout Text Importance Diff	12.5
66	Orig Layout Graphic Importance Diff	12.5
67	Orig Layout Text Relative Size	12.5
68	Orig Layout Graphic Relative Size	12.5
69	Orig Layout Overlap Diff	12.5
70	Orig Layout Out Of Bounds Diff	125

TABLE 3: All features used in our model, including the initial weights for the NIO learning.

APPENDIX H
ENERGY TERM AND SCORE CORRELATION

We can further analyze our model using the retargeting designs from MTurk users. Each of these 880 designs has an associated set of energy term values, as well as a ranking score. By computing the Pearson correlation between individual energy terms and the scores, we can determine which terms are the most closely tied to the quality of the designs. Note that these scores are not ideal, as they are relative to other 9 designs. To partially alleviate this issue, for each term we subtract the mean over all 10 designs in the ranking set. To evaluate statistical significance, we use a significance level of $0.05/70 = 0.0007$; the Bonferroni correction on the significance level is required as we are

Fig. 11: Intuition for NIO. The goal is to find a θ for which $\mathbf{X}_T$ is at the bottom of the energy function. Initially $\mathbf{X}_T$ is not at the bottom. So in each step the algorithm generates a layout $\mathbf{X}_S$ with lower energy than $\mathbf{X}_T$, and then adjusts θ to push $\mathbf{X}_T$ down and $\mathbf{X}_S$ up. From Liu et al. [9].

comparing 70 terms.

As Table 4 shows, there are many statistically significant, though fairly weak, correlations with different energy terms. As expected, since this a retargeting test, the terms measuring the difference from the original layout all have a positive correlation. The highest correlation ($r = 0.39$) is for the difference in text position from the original layout. Some other high-level conclusions are that higher scores are correlated with bigger text ($r = 0.32$), less overlap of text on images is preferred($r = 0.2$), elements should be spread out on the page ($r = 0.26$) (i.e., less white space), the model's flow heuristic is useful ($r = 0.3$), and text should be lower on the page ($r = 0.25$).

Feature	r
Orig Layout Text Position Diff	0.39
Text Height	0.32
Text Y Flow	0.3
Text Size Constraint	0.29
Orig Layout Text Height Diff	0.28
Orig Layout Text Height Diff	0.28
Spread	0.26
Orig Layout Graphic Position Diff	0.26
Text Y Position	0.25
Text Y Position - Reverse	-0.25
Graphic/Text Mean Imp Overlap	0.2
Orig Layout Graphic Height Diff	0.19
Orig Layout Graphic Height Diff	0.19
Orig Layout Text Importance Diff	0.18
Orig Layout Text Importance Diff	0.18
Element Margins	0.18
Global Text Y Symmetry	0.17
Graphic X Position Variance	-0.16
White Space Area	-0.16
Global Text Y Symmetry - Reverse	-0.16
Region Graphic X Symmetry	0.13
Text Overlap: Mean Worst	0.13
Graphic Y Position	-0.13
Graphic Y Position - Reverse	0.13
Text X Position Variance	0.12
Graphic/Text Mean Overlap	0.11

TABLE 4: Energy terms which show a statistically significant correlation with scores from MTurk users. The correlations were computed using features for 880 designs with the average scores given to those designs by other MTurk users.

Appendix I
Further Supplementary Materials

For completeness, we also include supplementary materials such as the web pages for our MTurk HITs, more importance prediction results, more design styles, and the full set of retargeting and improvement results and evaluations. Please see the project website at www.dgp.toronto.edu/~donovan/design/

References

[1] Pedro Felzenszwalb, David McAllester, and Deva Ramanan. A Discriminatively Trained, Multiscale, Deformable Part Model. In *Proc. CVPR*, 2008.

[2] Jerome H. Friedman, Trevor Hastie, and Rob Tibshirani. Regularization Paths for Generalized Linear Models via Coordinate Descent. *Journal of Statistical Software*, 33(1), 2010.

[3] Stas Goferman, Lihi Zelnik-Manor, and Ayellet Tal. Context-Aware Saliency Detection. In *Proc. CVPR*, pages 2376–2383, 2010.

[4] Geoffrey Hinton. Training Products of Experts by Minimizing Contrastive Divergence. *Neural Computation*, 14:2002, 2000.

[5] Xiaodi Hou and Liqing Zhang. Saliency Detection: A Spectral Residual Approach. In *Proc. CVPR*, pages 1–8, 2007.

[6] Laurent Itti and Christof Koch. A Saliency-Based Search Mechanism for Overt and Covert Shifts of Visual Attention. *Vision Research*, 40, 2000.

[7] Tilke Judd, Krista Ehinger, Frédo Durand, and Antonio Torralba. Learning to Predict Where Humans Look. In *Proc. ICCV*, 2009.

[8] R. Landa. *Graphic Design Solutions*. Cengage Learning, 2010.

[9] C. Karen Liu, Aaron Hertzmann, and Zoran Popovic. Learning Physics-Based Motion Style with Nonlinear Inverse Optimization. *ACM TOG (Proc. SIGGRAPH)*, 24:1071–1081, 2005.

[10] E. P. Simoncelli and W. T. Freeman. The Steerable Pyramid. In *Proc. ICIP*, volume 3, 1995.

[11] R. Tibshirani. Regression Shrinkage and Selection Via the Lasso. *Royal. Statist. Soc B*, 58(1):267–288, 1996.

[12] T. Tieleman. Training Restricted Boltzmann Machines using Approximations to the Likelihood Gradient. In *Proc. ICML*, pages 1064–1071, 2008.

[13] Ian Vollick, Daniel Vogel, Maneesh Agrawala, and Aaron Hertzmann. Specifying Label Layout Styles by Example. In *Proc. UIST*, 2007.

[14] A.W. White. *The Elements of Graphic Design*. Allworth Press, 2002.

Fig. 12: Style parameters are learned from the left two examples and used to generate layouts for other designs. (a) highly symmetric style with smaller elements and large margins, (b) asymmetric style with larger graphics and text, (c) higher placed graphics with larger, left aligned text.

Fig. 13: **Landscape-to-Portrait Retargeting**. We show retargeting results from an example layout by a professional designer, the best crowdsourced retarget (out of 9), and our automatic retarget.