Visual Font Pairing

Shuhui Jiang, Zhaowen Wang, *Member, IEEE*, Aaron Hertzmann, *Senior Member, IEEE*, Hailin Jin, *Member, IEEE*, and Yun Fu, *Fellow, IEEE*

Abstract—This paper introduces the problem of **automatic font pairing**. Font pairing is an important design task that is difficult for novices. Given a font selection for one part of a document (e.g., header), our goal is to recommend a font to be used in another part (e.g., body) such that the two fonts used together look visually pleasing. There are three main challenges in font pairing. First, this is a fine-grained problem, in which the subtle distinctions between fonts may be important. Second, rules and conventions of font pairing given by human experts are difficult to formalize. Third, font pairing is an asymmetric problem in that the roles played by header and body fonts are not interchangeable. To address these challenges, we propose automatic font pairing through learning visual relationships from large-scale human-generated font pairs. We introduce a new database for font pairing constructed from millions of PDF documents available on the Internet. We propose two font pairing algorithms: dual-space k-NN and asymmetric similarity metric learning (ASML). These two methods automatically learn fine-grained relationships from large-scale data. We also investigate several baseline methods based on the rules from professional designers. Experiments and user studies demonstrate the effectiveness of our proposed dataset and methods.

Index Terms—Font, Pairing, Recommendation, Metric learning.

1 Introduction

Applying artificial intelligence based artistic style transfer, generation and recommendation to facilitate the design of painting [1], photo [2], pose [3], dance [4], fashion [5], [6], etc. has drawn a lot of attention in the filed of multimedia recently. Visual font style and text detection, recognition and prediction [7], [8], [9], [10], [11], [12], [13], [14] play an important role not only in image classification and retrieval, scene detection, video tracking but also automatic generation of visual-text presentation layout [15] and visual document analysis [16], [17]. Pairing fonts is an important task in graphic design for documents, posters, logos, advertisements and many other types of design. A designer typically picks a title font, sub-header fonts, body text fonts, and does so in a way that is harmonious and appropriate for the style. Fonts should complement each other without "clashing" or appearing disconnected. For example, Figure 1 shows the same advertisement with two choices of font for the sub-header ("Continues"), and the same font for the header ("The Heritage"). Each pair of choices conveys a different design quality: in one case, the fonts complement each other and appear more interesting, whereas when they are nearly the same the layout is less appealing. Each pair of header and sub-header fonts represents a *font pair*. Despite the importance of font pairing, current design tools do not provide much assistance in the challenging task for pairing fonts, aside from providing a few template designs. Design books and websites provide many rules-of-thumb for font pairing, such as "Use Fonts from the Same Family", "Mix Serifs and Sans Serifs" and "Create Contrast"[1,2], but such rules can be hard to apply in practice or to formalize. Font pairing is especially challenging for novices creating designs, who may lack the intuitions for selecting fonts.

Fig. 1. Examples for PDF pages with different font pairing. The font of headers are the same, while the fonts of sub-headers are different. We show the same PDF page rendered with two choices of font for the sub-header ("Continues"), and the same font for the header ("The Heritage").

This paper introduces the problem of automatic font pairing. Given a font selection in a document, our goal is to recommend matched fonts that produce pleasing visual effect when they are used together in different parts of a document. For example, given a header font, we recommend a body font, or vice versa. There are three main challenges in font pairing, compared to most pairing and recommendation problems in the fields of vision and multimedia [6], [18], [19], [20], [21]. First, this is a fine-grained problem, which means that subtle distinctions between fonts may be important, as opposed to object-level co-occurrence problems (e.g., sky and airplane) [22], [23], [24] or category-level

• *S. Jiang is with the Department of Electrical and Computer Engineering, Northeastern University, Boston, MA 02115 USA (e-mail: shjiang@ece.neu.edu). Y. Fu is with the Department of Electrical and Computer Engineering, College of Engineering, and Khoury College of Computer Sciences, Northeastern University, Boston, MA 02115 USA (e-mail: yunfu@ece.neu.edu). Z. Wang, A. Hertzmann and H. Jin are with Adobe Systems Inc. (e-mail: {zhawang, hertzman, hljin}@adobe.com).*

Manuscript received April 19, 2005; revised August 26, 2015.

1. https://www.canva.com/learn/combining-fonts-10-must-know-tips-from-a-designer/

2. https://www.creativebloq.com/advice/9-golden-rules-for-combining-fonts

pairing (e.g., tops and skirts [25], [26], [27], [28], [29], [30]). Second, designers have listed many rules for font pairing, but they are difficult to formalize. Font pairing is not simply a problem of similarity: designers typically pair contrasting fonts as well as similar fonts. Third, font pairing is an asymmetric problem. Pairing font A as header and font B as body is different as pairing font B as header and font A as body.

It should be noted that font pairing is a complex task: a good font combination is decided by many factors beyond font itself, e.g., text sentiment, layout, the background image and even personal taste. As a first attempt to attack this problem, the goal in this paper is to recommend font pairs that satisfy majority users' aesthetics only based on visual font information. We believe this is less challenging than also considering other elements in design context. It is also a well-posed problem since most tutorials and books [31] on this topic recommend font pairs in the same setting.

To address these challenges, we propose to learn font pairing from large-scale human-designed font pairs. However, obtaining appropriate data for this problem is challenging, as there is no existing dataset and the font pairs from Internet web pages are noisy or biased to a small set of popular fonts. We collected a new database called "FontPairing" from millions of PDF pages on the Internet. These PDFs embody a very diverse set of designs with font meta data embedded. We devise some heuristics to automatically identify header/sub-header and header/body pairs from the PDF pages with a high accuracy verified on a subset.

Given this data, we investigate two algorithms to learn font pairing: dual-space k-NN and asymmetric similarity metric learning method. The intuition behind dual-space k-NN is that the users may choose the same body font for similar header fonts, and vice-versa. For example, if the font Univers is similar to Helvetica, then fonts that pair well with Univers should also pair well with Helvetica. Given an input query font (e.g., header), we first find the k most similar fonts from the training header fonts according to the similarity of visual appearance, which is measured by a deep neural network trained for this task. We then rank their corresponding body fonts by the number of times the body font is repeated in the training data.

The goal of asymmetric similarity metric learning is to learn a distance function by which fonts that pair well have small distances to each other, and, conversely, mismatched fonts are far apart. The metric is discriminatively trained from our training font pairs. Especially, we jointly learn the model that bridges the asymmetric similarity and distance metric. At test time, an online prediction entails finding the nearest font pairs in the dataset.

To the best of our knowledge, this is the first time that font pairing problem has been recognized as a multimedia and computer vision problem. Since there is no prior work, we compare with several baseline methods (e.g., same font family, similarity, contrast) according to the rules provided by professional designers. Experimental results show the effectiveness of our visual learning based dual-space kNN and asymmetric similarity metric learning methods.

2 Related Work

To the best of our knowledge, this is the first work of visual font pairing problem, and there is very few related works.

In the fields of multimedia and computer vision, there are several methods being proposed for font recognition [8], [9], [32], [33] and font prediction [10] based on large datasets of fonts and their images. Wang et al. [8], [9] worked on visual font recognition with deep neural networks. Zhao et al. [10] proposed a multi-task neural network method for recommending proper font properties (e.g, font face, size, and color) for web design, given both text and semantic tags such as Design and HTML tags. Our work is also related to systems for learning to parse web pages, such as WebZeitGeist [34]. The one most relevant to our work is by O'Donovan et al. [7], who present interfaces for finding fonts based on learned models of font style. However, their work focuses only on single fonts in isolation, whereas we consider how two fonts pair with each other. Font pairing is also related to visual document analysis (e.g., [16], [17]) and visual-textural presentation layout generation [15].

In terms of methodology, our work is highly related to other visual pairing tasks, particularly pairing clothing [25], [26], [27], [28], [29], [30], furniture [21], and food [20]. Here we address font pairing, which entails particular difficulties including lack of an appropriate data source, fine-grained difference between font types and asymmetrically pairing entities of the same category instead of different categories.

3 A Database for Font Pairing

In this section, we introduce the new database that we generated for visual font pairing task called "FontPairing". We collect millions of freely-available PDFs on Internet, analyze and extract header/body font pairs and header/sub-header pairs from the PDF pages.

3.1 Font Pairings From Web

Perhaps the most obvious approach to gather font pairing data is to obtain them from webpages such as Google Fonts[3], Typ.io[4], Typewolf[5]. For example, each font on Google Fonts is provided with a list of suggested pairings. However, we found these datasets inadequate for two reasons. First, they each provide a small set of pairings. The second and more significant problem is that these pairing lists are extremely unbalanced: these websites generally recommend only popular body fonts. Out of the above pairs, 43% of the font pairs involve one of the five most-popular fonts.

3.2 PDF Dataset

To address these problems, we propose to detect font pairs from millions of PDF documents available on the Internet. We have collected more than 300,000 PDF files from various websites such as Commons.wikimedia.org and Digital-library.usma.edu. Each PDF usually contains dozens to hundreds of pages; from all these PDFs, we obtain more than 15 million pages in total. As shown in Figure 2, these PDF pages exhibit various layouts, topics, and font styles. We believe this dataset could be potentially useful for training other models for document design as well.

A key challenge is then to extract visual information from this large dataset. Although PDF is a structured document format, it is complex and does not include the annotations we need (e.g.,"header font"). Parsing such structured representations is a major challenge in itself [34]. Rather than attempting to fully parse the document, we focus only on identifying the font pairs containing the header, sub-header, and/or body fonts.

3. https://fonts.google.com
4. http://typ.io/
5. https://www.typewolf.com

Fig. 2. Examples for font pair detection in freely-available PDFs. The left four columns show documents with detected header/body pairs, marked with red bounding boxes. The right four columns show examples of documents with detected header/sub-header pairs, with headers marked in red and sub-headers in blue.

Of the PDFs we collected, 43% are scanned documents. We omit these from the dataset, to simplify parsing and avoid additional noise caused by the parsing processing. For the remaining data, we apply open PDF tools to extract text, image, and layout information from each page of each document. We define a *text box* as a several words with the same font style and size in a line. Each text box is annotated with the font style, font size, and the bounding box. We discard pages that contain fewer than two text boxes. We also focus only on pages with the Roman alphabet, which we identify using Python language detection tools. Of the dataset, 75% of documents are in English.

To detect a header and sub-header pair on a page, we first find the largest text box, and call this the header text. We then identify the largest text box that lies within a fixed threshold of the header text box. We then call this a header/sub-header pair, and extract the fonts from the two text boxes. Only one header/sub-header pair is found for each document. We also detect body text boxes by finding text boxes with number of characters above a threshold. The nearest body text box to a header is used to form a header/body pair. Figure 2 shows example detection results on both header/body and header/sub-header pairs.

To evaluate the accuracy of automatic pair detection on PDF, we manually label header/body and header/sub-header pairs on a small subset of PDFs (i.e., 20 PDFs with varies topics and layouts totalling 3,000 pages). Here, our purpose is not to evaluate whether these are good pairing or not. We manually compare the automatic detection results with human labeling for verification. By adjusting our detection thresholds (e.g., the distance between the text boxes of header and sub-header), we achieve about 95% precision (true positives) in our automatic detection. For header/subheader pairs, our detector achieves 85% precision (true positives). There are more variations in the layout of header/subheader, which makes this task much harder than detecting header/body pairs.

The number of total unique header fonts, sub-header fonts, body fonts and pairs are shown in Table 1. Figure 3 shows the top 5 header and body fonts used in header/body pairs and Figure 5 shows the histogram of frequency that a header or body font appears in unique font pairs. Only 2.7% of header/body pairs involve one of the five most popular header fonts, and 7.5% of pairs involve one of the five most popular body fonts. This indicates a far more diverse set of pairings than web recommended pairings, of which, as reported above, 43% involve one of the five most-popular body fonts.

Fig. 3. Top 5 header and body fonts used in header/body pairs. Fonts are written in the PostScript font name format, which typically includes both font family (e.g., “Helvetica”) and style (e.g., “Bold”).

Fig. 4. Examples of head/sub-header pairing in FontPairing dataset, for the header font “CaeciliaLTStd-Heavy”.

Figure 4 shows sample font pairings in our dataset when given a query header font “CaeciliaLTStd-Heavy”. For this font, our dataset includes 10 font pairs, and 20 for the entire “Caecilia” family (including Bold, Heavy, Italic, etc). This is much more diverse than those in web font sources. For example, for this same font, there are only 4 header/sub-header pairings on Fontinuse.com, 1 pairing on typewolf.com, 2 on typ.io.com, and 0 on Google Fonts.

3.3 Quality Verification

Following Veit et al. [26], we conduct an online user study to compare whether designers and ordinary users prefer the real font pairs we detected from PDFs or the random alternatives. Our

TABLE 1
Number of unique fonts and pairs of header/body pairing (upper) and header/sub-header pairing (bottom) are shown in the "full" column. Number after removing pairs with top 50 famous body/sub-header fonts are shown in the "non-popular" column.

font set	full	non-popular
unique headers	2,086	616
unique bodies	1,443	1,343
header/body pairs	13,251	5,337
unique headers	2,159	1,054
unique sub-headers	2,168	1,573
header/sub-header pairs	8,733	5,174

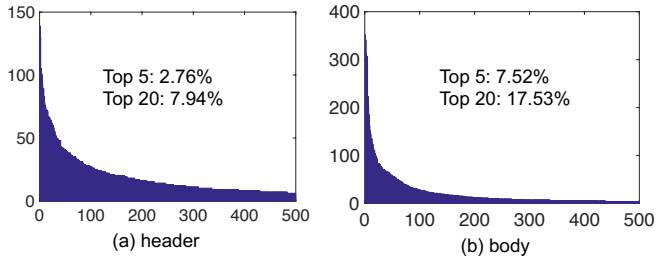


Fig. 5. Data distribution of head/body pairs in FontPairing dataset. The histogram of the number of times a header font appears in unique font pairs is shown in (a) and the histogram of a body font is shown in (b).

study includes 60 participants: 15 experts in graphic design (either students in art design major or staff in design company) from Upwork [6], and 45 non-designers with other backgrounds from Amazon Mechanical Turk [7] and volunteers.

The study comprised a set of paired comparisons. In each comparison, a user is shown two images of the same layout, but with one font changed (Figure 1). In particular, either the header or sub-header font is replaced by selecting a random alternative. The viewer is then asked which design they prefer. We perform two variants of the study: in the first one, the entire page layouts are shown to the viewer; in the second, the user is only shown part of the page containing the relevant text boxes, so that they will focus more on the font choices rather than the context. In whole-page setting, we show 20 comparisons to the users, and in sub-page setting, we show 50 comparisons to the users. These samples are randomly sampled from all the pairs.

Under both whole-page and sub-page settings, experts prefer the original layout 75% of the time. Non-experts prefer the original 65% of the time when viewing the full page, and 60% of the time when viewing the sub-page. Note that the original layout is not necessarily superior to the font choice in the random selection, for various reasons; however, we would expect that it would be more likely to be better. Hence, these results indicate that the pairing combinations included in the dataset are aligned with the preference of expert and common users. These results also suggest that non-experts are much less sensitive to good font choices than experts, and that there is potential value to recommend good pairings to them.

We would like to clarify that, although the font pairs extracted from PDF dataset are of varying quality, by training on a large amount of data, we aim to smooth out the noise therein and discover the general pairing rules that match majority users' preference.

4 METHODS

In this section, we first introduce the problem definition of visual font pairing problem. Second, we present two methods designed for font pairing: dual-space k-NN (DS-kNN) and asymmetric similarity metric learning (ASML).

4.1 Problem Definition

Let us define a query header font q with feature vector $\mathbf{x}_q \in R^D$, and a set of body or sub-header fonts with features $Y = \{\mathbf{y}_1, ..., \mathbf{y}_j, ...\}$, $(\mathbf{y}_j \in R^D)$. Our task is to recommend a list of good body or sub-header fonts to go with $\mathbf{x}_q$ i.e., $\mathcal{P}_q = \{\mathbf{y}_{q1}, \mathbf{y}_{q2}, ...\}$. The goal of our problem is to learn a scoring function $f(\cdot, \cdot)$ from a large-scale font paring dataset to measure the pairing score for $f(\mathbf{x}_q, \mathbf{y}_j), \forall \mathbf{y}_j \in Y$, so that good suggestions are in the highest-ranked results according to their pairing scores. Without loss of generality, we discuss the header/body pairing in the rest of the section.

To this end, we seek to learn $f(\cdot, \cdot)$ in a supervised manner from the visual relationship of font pairs in the training set. Suppose we have m training header fonts with feature vectors $\{\mathbf{x}_1, ..., \mathbf{x}_m\}$, and n training body fonts with features $\{\mathbf{y}_1, ..., \mathbf{y}_n\}$. For each header font $\mathbf{x}_i$, there is a list of body fonts that pair with it, i.e., $\mathcal{P}_i = \{\mathbf{y}_{i1}, \mathbf{y}_{i2}, ...\}$. We regard $(\mathbf{x}_i, \mathbf{y}_\alpha)$, $\mathbf{y}_\alpha \in \mathcal{P}_i$ as a positive pair and $(\mathbf{x}_i, \mathbf{y}_\beta)$, $\mathbf{y}_\beta \notin \mathcal{P}_i$ as a negative pair. In training stage, our goal is to learn a scoring function $f(\cdot, \cdot)$, so that positive pairs always have higher pairing scores than negative pairs. In this way, in the test stage, when we rank body fonts according to the pairing scores, the good suggestions are at the top of the results.

We use pretrained feature of each font from DeepFont method [8], [9] as the input font feature representation. DeepFont model is trained for font recognition on the large-scale Visual Font Recognition (VFR) dataset. This dataset consists of 201,780 real-world text image collected from various typography forums, including 4,384 images with reliable font labels covering 617 classes, and 1,000 synthetic image for each class. The data augmentation for synthetic data includes 6 steps: Noise, Blur, Perspective rotation, Shading, Variable character spacing and Variable aspect ratio, to reduce the mismatch between real-world image and synthetic text image. DeepFont investigates a Convolutional Neural Network (CNN) decomposition and Stacked Convolutional Auto-Encoder (SCAE) based domain adaptation approach, to address the discrepancies between real-world data and synthetic data. Using this model, we obtain the feature vector for a font i denoted as $\mathbf{x}_i \in R^D$, where $D = 768$. As shown in Figure 6, distances in the DeepFont feature space are correspond to the perceptual similarity between fonts. It demonstrates the effectiveness of DeepFont feature for searching for perceptually-similar fonts. We do not choose to apply the end-to-end deep neural network (DNN) to learn font pairing. The main reason is that the number of unique header/body pairs and header/sub-header pairs in our database is 13,251 and 8,733 respectively as shown in Table 1, which is not enough to train an end-to-end DNN.

In the following, we discuss two methods dual-space k-NN (DS-kNN) and asymmetric similarity metric learning (ASML) for font pairing.

6. www.upwork.com
7. www.mturk.com/mturk/welcome

4.2 Dual Space k-NN Search based Method

The intuition behind dual space k-NN search based method (DS-kNN) is that, if fonts F_1 and F_2 are similar, then fonts that pair with F_1 should be good pairings with F_2.

Suppose we are querying which body fonts $\mathcal{P}_q = \{\mathbf{y}_{q1}, \mathbf{y}_{q2}, ...\}$ will go with a header font $\mathbf{x}_q$. We first find the top K_1 nearest header fonts $[\mathbf{x}'_1, ..., \mathbf{x}'_i, ..., \mathbf{x}'_{K_1}]$, based on the cosine similarity $\cos(\mathbf{x}_q, \mathbf{x}_i)$ between $\mathbf{x}_q$ and all the training headers $\{\mathbf{x}_1, ..., \mathbf{x}_i, ..., \mathbf{x}_m\}$. Each header $\mathbf{x}'_i$ has a list of body fonts that pair with it, i.e., $\mathcal{P}'_i = \{\mathbf{y}'_{i1}, \mathbf{y}'_{i2}, ...\}$. Note that fonts may repeat in this list, so that the popularity of pairings can be captured in the data. The fonts in $\tilde{\mathcal{P}} = \{\mathcal{P}'_1, ..., \mathcal{P}'_i, ..., \mathcal{P}'_{K_1}\}$ are regarded as candidate body fonts for pairing $\mathbf{x}_q$. We assume that there are N_1 fonts in candidate body font set $\tilde{\mathcal{P}}$. Fonts may also repeat in $\tilde{\mathcal{P}}$. A higher repeat frequency in this list demonstrates that more similar headers are paired with this body font in the training set.

The candidate body fonts may only cover a part of the good pairings among all the body fonts. Fonts similar to the candidate body fonts may also result in pleasing pairings. Therefore, we rank all the n body fonts $\{\mathbf{y}_1, ..., \mathbf{y}_j, ..., \mathbf{y}_n\}$ based on the similarity score $\tilde{S}(\mathbf{y}_j)$ compared with candidate body fonts, and recommend top N fonts with the highest scores.

Here we introduce the way to calculate $\tilde{S}(\mathbf{y}_j)$ for font $\mathbf{y}_j$. We first calculate the cosine similarity $\cos(\cdot, \cdot)$ between $\mathbf{y}_j$ and each candidate body font in $\tilde{\mathcal{P}}$. Second we select top K_2 candidate body fonts with the largest $\cos(\mathbf{y}_j, \mathbf{y}_l)(l \in \{1, ..., K_2\})$. Then we calculate $\tilde{S}(\mathbf{y}_j)$, which is the average of multiplying $\cos(\mathbf{y}_j, \mathbf{y}_l)$ and $\cos(\mathbf{x}_q, \mathbf{x}_l)$. $\cos(\mathbf{x}_q, \mathbf{x}_l)$ is similarity of the query header $\mathbf{x}_q$ and similar header $\mathbf{x}_l$ in the top K_1 nearest header list. $(\mathbf{x}_l, \mathbf{y}_l)$ is a font pair in the collection. $\tilde{S}(\mathbf{y}_j)$ is calculated as:

$$\tilde{S}(\mathbf{y}_j) = \frac{1}{K_2} \sum_{l=1}^{K_2} \cos(\mathbf{y}_j, \mathbf{y}_l) \cdot \cos(\mathbf{x}_q, \mathbf{x}_l). \quad (1)$$

Note that fonts may also repeat in the list of top K_2 candidate body fonts. It is similar as the idea of adding a term frequency (tf) weight in tf-idf (short for term frequency-inverse document frequency [35]) for each unique font in K_2 candidate body fonts. We aim to increase the weight of body fonts that appear frequently in the top K_2 candidate body font list, and reduce the impact of body fonts which are widely used for pairing different header fonts, regarded as popular body fonts. The tf score is added by considering the repetitiveness of a body font in the list of top K_2 candidate body fonts. It adjusts the weight of a body font in proportional to the times it appears in this list. The inverse document frequency (idf) score adjusts the weight of a body font in inverse ratio to the number of header fonts in the training list which this body font is paired with. The idf score could be further integrated as Eq. (1) by multiplying $\mathrm{idf}(\mathbf{y}_l)$ for each $\mathbf{y}_l$ as:

$$\hat{S}(\mathbf{y}_j) = \frac{1}{K_2} \sum_{l=1}^{K_2} \cos(\mathbf{y}_j, \mathbf{y}_l) \cdot \cos(\mathbf{x}_q, \mathbf{x}_l) \cdot \mathrm{idf}(\mathbf{y}_l), \quad (2)$$

where $\mathrm{idf}(\mathbf{y}_l) = \frac{m}{t_l}$. m is the number of total header fonts and t_l is the number of header fonts with which body font $\mathbf{y}_l$ is paired in the training set.

After calculating $\hat{S}(\mathbf{y}_j)$ for each body font $\mathbf{y}_j$, we rank body fonts based on their scores from high to low and obtain the top ranked body fonts $\mathcal{P}_q = \{\mathbf{y}_{q1}, \mathbf{y}_{q2}, ...\}$ as good pairings for $\mathbf{x}_q$.

However, DS-kNN does not perform well if accurate similar header fonts are hard to find in the dataset. Also, popular body fonts have more chances of pairing similar header fonts and appearing in the set of candidate body fonts in this method. Meanwhile, there are some font pairing rules that may be missed by DS-kNN. For example, using the same font family for the header and body (e.g., Helvetica Bold for header and Helvetica for body). These rules are difficult to capture and could not be easily solved by calculating font similarity in the original feature space. These concerns motivate us to learn the metric in the next section to capture the common pairing strategies.

Fig. 6. Two examples for similar font retrieval based on DeepFont features. In each column, the first row is the input query font. The following rows present top 5 similar fonts measured with the distance of DeepFont feature. The robustness of DeepFont features facilitate the performance of dual-space kNN method.

4.3 Asymmetric Similarity Metric Learning

The goal of Asymmetric Similarity Metric Learning method is to learn a better distance scoring function between two fonts, so that fonts that pair well have a lower distance, and mismatched fonts have a larger distance. We train this scoring function offline. Then predictions are generated for a given query $\mathbf{x}_q$ by finding the fonts $\mathcal{P}_q = \{\mathbf{y}_{q1}, \mathbf{y}_{q2}, ...\}$ with the lowest distance based on the new scoring function.

We treat the training dataset as comprising font pairs $(\mathbf{x}_i, \mathbf{y}_j)$, and an indicator function $S(i, j) = 1$ when fonts are paired in the training dataset. Since our FontPairing dataset only contains positive pairs, we randomly sample negative pairs among all the other possible pairs excluded these positive pairs. It is worth pointing out that while there may exist good pairings among the negative set, these should be in the minority, especially since the user study found that positive pairs were more attractive than randomly picked pairs to designers. Generally speaking, the original font pairs in PDFs are usually specifically designed and should achieve higher accordance than randomly picked ones. The number of negative pairs and positive pairs are equal. The indicator function $D(i, j) = 1$ when fonts are negative pairs.

The main idea for conventional metric learning [36] is to learn a better scoring function $||\mathbf{x} - \mathbf{y}||^2_{\mathbf{M}} = (\mathbf{x} - \mathbf{y})^{\intercal}\mathbf{M}(\mathbf{x} - \mathbf{y})$, to enlarge the two font points of non-matching pairs and narrow the font points for matched pairs. The learning objective function is:

$$\begin{aligned} &\min_{\mathbf{M}} \sum_{i,j}^{m,n} ||\mathbf{x}_i - \mathbf{y}_j||^2_{\mathbf{M}} \cdot S_{ij} \\ &s.t.\ \mathbf{M} \succ= 0,\ \sum_{i,j}^{m,n} ||\mathbf{x}_i - \mathbf{y}_j||^2_{\mathbf{M}} \cdot D_{ij} \geq 1. \end{aligned} \quad (3)$$

Although metric learning methods are effective for many supervised learning applications such as classification, they have

a few limitations in our font pairing problem. First, instead of applying nearest neighbor classifiers with ML in classification problem, after ML, we still need to make a decision, such as with a constant threshold d:

$$f_{ML}(\mathbf{x}_i, \mathbf{y}_j) = d - (\mathbf{x}-\mathbf{y})^\intercal \mathbf{M}(\mathbf{x}-\mathbf{y}). \tag{4}$$

However, a simple constant threshold may still cause sub-optimal solutions, even with a correct associated metric. Another challenge is that, font pairing is an asymmetric problem. It means that paring A as header and B as body is different from pairing B as header and A as body. To address these challenges, we consider a jointly model that bridges the learning of a distance metric and an asymmetric similarity [37], [38] decision rule and propose asymmetric similarity metric learning as:

$$f_{(\mathbf{M},\mathbf{G})}(\mathbf{x}_i, \mathbf{y}_j) = \mathbf{x}^\intercal \mathbf{G}\mathbf{y} - (\mathbf{x}-\mathbf{y})^\intercal \mathbf{M}(\mathbf{x}-\mathbf{y}), \tag{5}$$

where $\mathbf{G}$ is asymmetric. $\mathbf{x}^\intercal\mathbf{G}\mathbf{y}$ measures the similarity of font pairs.

Let $P = S \cup D$ denotes the index set of all pairwise constraints. Let $y_{i,j} = 1$ if $S(i,j) = 1$ and $y_{i,j} = -1$ if $D(i,j) = 1$. We drive the formulation of the empirical discrimination using hinge loss:

$$\begin{aligned} &\min_{\mathbf{M},\mathbf{G}} \sum_{(i,j)\in P} (1 - y_{i,j} f_{(\mathbf{M},\mathbf{G})}(\mathbf{x}_i, \mathbf{y}_j))_+ \\ &+\gamma/2(\|\mathbf{M} - I\|_F^2 + \|\mathbf{G} - I\|_F^2), \end{aligned} \tag{6}$$

where the regularization term $\|\mathbf{M}-I\|_F^2 + \|\mathbf{G}-I\|_F^2$ prevents the vector being distorted too much. $\|\cdot\|_F$ is the frobenius norm. γ is the trade-off parameter. This objective function could be solved with dual formulation as [39]. However, in [39], a symmetric similarity decision rule is applied, and it addresses the symmetric learning problem (i.e., face verification).

After off-line learning the new scoring function as Eq. (5), in online pairing, given a query $\mathbf{x}_q$, we calculate $f_{ML}(\mathbf{x}_q, \mathbf{y}_j)$ between header $\mathbf{x}_q$ and each body font $\mathbf{y}_j$ based on the new scoring function. Then we rank body fonts by their scores from high to low and obtain the top ranked body fonts $\mathcal{P}_q = \{\mathbf{y}_{q1}, \mathbf{y}_{q2}, ...\}$ as good pairings for $\mathbf{x}_q$.

5 Experiments

5.1 Compared Methods

We implement the following baselines for comparison, including several methods based on design rules-of-thumb.

Popularity: This method aims at recommending most popular body fonts. First, we rank all body fonts according to the frequency that they appear in font pairs in the collected dataset. The top-ranked body fonts are defined as popular fonts. These same fonts are always recommended, regardless of the query header.

Simple kNN (S-kNN): This method aims at recommending body fonts with the highest visual similarity to the query header font as pairs. The distance of similarity between two fonts is measured using DeepFont feature representation.

Contrast similarity (ConSim): The main intuition in this work is that the ideal pairing has similarities and contrasts in equal importance. They manually designed a contrast similarity distance metric. More details could be found at [40].

Similarity metric learning (SML): We also implement a similarity metric learning method (SML) to evaluate the effectiveness of making the metric $\mathbf{G}$ asymmetric in ASML. We replace asymmetric $\mathbf{G}$ in Eq.(5) and (6) with the symmetric metric. This idea is similar as [39], but [39] addresses on the face verification problem.

Dual-space kNN (DS-kNN)(Ours): Our proposed dual-space kNN method.

Asymmetric similarity metric learning (ASML) (Ours): Our proposed asymmetric similarity metric learning method.

5.2 Experimental Setting

We perform quantitative evaluation similar to other pairing tasks [18], [25], [26]. We conduct two sets of experiments: top-N recommendation and binary classification. Without loss of generality, we discuss the setting that given a header font, we recommend body pairings as an example in the rest of the section.

The first evaluation is to formalize the pairing problem as a retrieval problem. Given a header font, we rank all the body fonts and recommend top-N body fonts as good pairings. The second evaluation is to formalize the pairing problem as a binary classification problem: given a font pair, we want to classify whether it is a good pairing or not.

We split the header fonts in FontPairing dataset into training header set and test header set by a ratio of 9:1 with no overlap. Only the pairings with training headers are used as positive training pairings. In this way, in the test stage, we are able to evaluate the performance of recommending body fonts to pair an unseen header font. The body fonts in the training and test set may have overlaps.

5.2.1 Top-N recommendation

In real-world font pairing interfaces, we would like to recommend multiple candidate fonts for pairing, and let the user pick from this list [7]. Thus, we evaluate top-N recommendation performance, namely, precision and recall at N, which are widely used in recommender systems [41].

Assuming the user gets a top-N recommended list of fonts, **recall** is the percentage of relevant fonts selected out of all the ground truth fonts, and **precision** is the percentage of the N results that are good recommendations.

Besides the conventional top-N precision and recall, we also apply **weighted precision** and **weighted recall** as the evaluation metrics. Since popular fonts are easier to be considered and may be less interesting to users, we add an IDF weight [42] (the popular the lower) to each font to decrease the impact of popular fonts. The weighted precision and recall metrics are defined as:

$$\text{weighted\_precision} = \text{weighted\_TP}/N, \tag{7}$$

$$\text{weighted\_recall} = \text{weighted\_TP}/\text{weighted\_GT}, \tag{8}$$

where weighted_TP is the sum of all the IDF weights of true positive fonts. weighted_GT is the sum of all the IDF weights of ground truth fonts.

5.2.2 Binary classification

Following the experimental settings from clothing pairing works [25], [26], we formalize evaluation in terms of binary classification. Given a header and body font pair, we want to classify whether or not it is a good pairing.

We regard all the font pairs we extracted from PDFs as positive samples. The test set is formed with positive samples and negative samples of equal proportion. Thus, chance performance is 50%

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMM.2019.2952266, IEEE Transactions on Multimedia

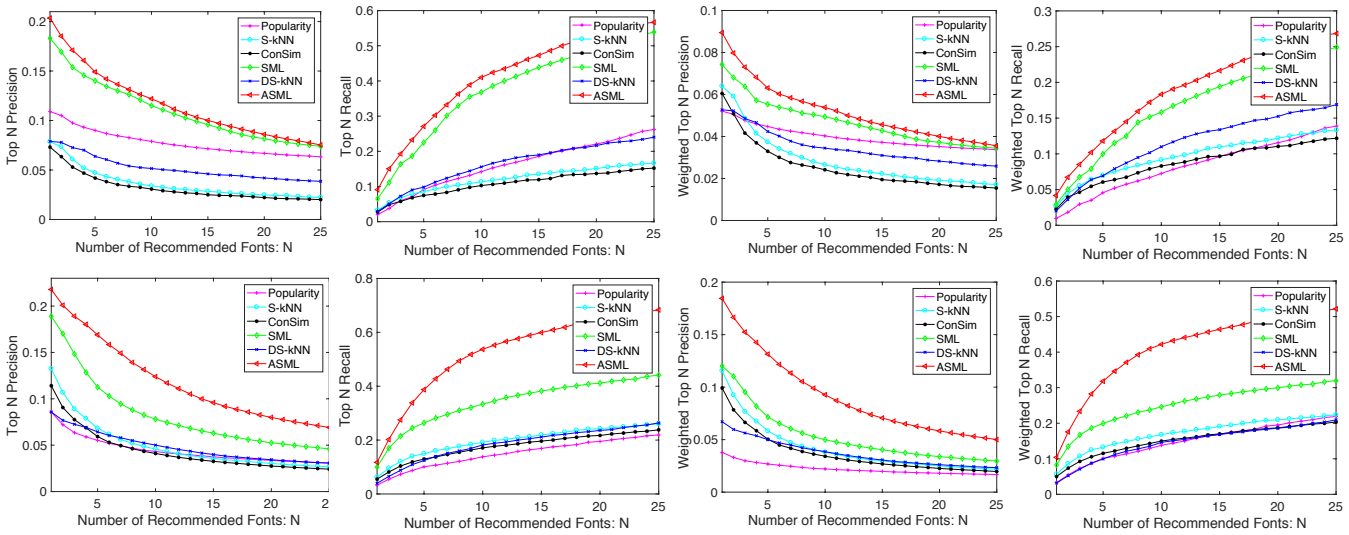


Fig. 7. Performance of top N recommendation on header/body (first row) and header/sub-header (second row) pairing with top N precision and recall and weighted top N precision and recall evaluation metrics.

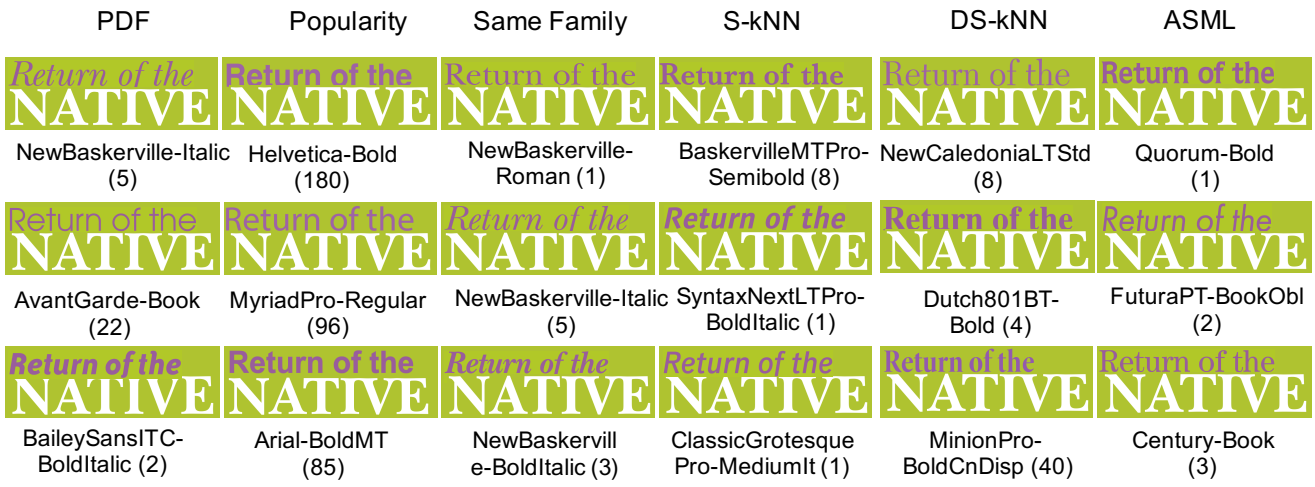


Fig. 8. Examples of header/sub-header font pairing results. The input is a query header font "NewBaskerville-BoldSC". All the pairs are rendered with same format and header font, but only with different sub-header fonts. The most left column shows three pairings in our collection. In each column, we show the top 3 recommended sub-header fonts by each methods. The PostScript name of each sub-header is shown below the image. The number of times the sub-header font appears in the unique font pairs are shown within the parentheses. We recommend to zoom in to get more details of the fonts.

for all experiments. The negative samples are randomly-sampled pairs, excluding all positive pairs, following [25], [26]. In "Quality Verification" section, we have shown that both designers and average users generally prefer the pairings extracted from PDF documents than random chances.

5.3 Top-N Recommendation Results

Figure 7 shows the performance of top-N recommendation on header/body pairing and header/sub-header pairing under 6 methods. We show the performance of each method under two metrics: top-N precision and recall and weighted top-N precision and recall. The number of recommended fonts N is shown in x-axis and the corresponding top-N precision and recall are shown in y-axis. The best results of each method are shown in the figure and the parameters are set based on cross-validation.

In all the cases, ASML achieves the highest performance among 6 methods. It demonstrates the effectiveness of regarding visual font pairing as the asymmetric metric learning problem. Also, ASML outperforms SML, which demonstrates the effectiveness of the asymmetric constraint. ASML could automatically learn various font pairing rules and outperform existing rules such as similarity (S-kNN) and contrast similarity (ConSim).

Figure 8 shows the visualization of top recommended sub-header fonts by comparison methods, given a query header font "NewBaskerville - BoldSC". The most left column shows the pairings extracted from PDFs. Popularity method tends to recommend fonts with the highest number of frequency shown in parentheses. Same Family method randomly picks fonts from the same font family. Generally, the font pairings in PDF include many cases that fonts are from the same font family, since it is easy to implement. While Same Family and Popularity method would hit more of PDF extractions, it is very easy for users to pick font with family by themselves, so that the users especially designers may be not interested in these recommendations. Another concern is that it

TABLE 2
Performance on binary classification of header/body and header/sub-header pairing under two settings as Table 1.

task	header/body		header/sub-header	
setting	full	non-popular	full	non-popular
Popularity	73.60	55.29	68.04	61.35
S-kNN	52.87	60.43	62.32	67.82
ConSim [40]	55.81	67.28	61.32	65.79
SML [39]	60.80	67.34	67.61	72.69
DS-kNN	**76.93**	59.28	**71.30**	63.46
ASML	64.97	**68.23**	68.43	**73.41**

may fail to find same family font for some less popular fonts. S-kNN shows the pairings with the smallest visual distances. In DS-kNN(ours) and ASML(ours) methods, we are able to recommend font pairings that are both interesting and unpopular, and meanwhile achieve the coordination of pairing.

5.4 Binary Classification Results

Table 2 shows binary classification results on header/body and header/sub-header paring under settings: (1) with all the font (full) and (2) removing top 50 popular body/sub-header fonts (non-popular) as described in Table 1. Classification thresholds are set by cross-validation with training data in each method respectively.

In setting "full", DS-kNN achieves the highest performance in both header/body and header/sub-header pairing. In header/body, Popularity achieves the second highest performance. It is consistent with the phenomenon shown in Figure 5 that popular body fonts take a large proportion of head/body pairs in PDF designs. In "full" setting, DS-kNN achieves higher accuracy than ASML. It is mainly because there are dominant popular fonts in "full" setting. In "full" setting in DS-kNN, popular fonts appear frequently in the list of top candidate body fonts which pair similar headers. Thus these popular fonts appear frequently in both recommended results and ground truth, and the recommended fonts have more chances to hit the ground truth, which causes the high performance of both DS-kNN and Popularity.

To decrease the effects of dominant popular fonts, we also conduct the experiment under "non-popular" setting. In setting "non-popular", ASML achieves the highest performance, followed by SML. Since there is no dominant popular fonts phenomenon, DS-kNN achieves lower accuracy than ASML. Tables 2 demonstrates the effectiveness of DS-kNN and ASML in both regular (full setting) and non-popular font pairing tasks.

5.5 Subjective Evaluation

Besides the quantitative evaluations, we also conduct subjective evaluation through user study on AMT and Upwork, which are crowdsourcing platforms targeting average people and professionals respectively.

The study comprises a set of paired comparisons. One is either from DS-kNN or ASML, the other is from one of the compared methods. The users are only shown the sub-page contains the text as shown in Figure 8. We evaluate 500 comparisons and each comparison receives at least 11 ratings by average users and 3 ratings by designers.

Before describing the evaluation results, we firstly analyze the consistency of users' rating. If the users have consistent opinions about which pair is superior than the other, the ratings are more convincing and could be applied to the following studies. It is important to analyze the rating consistency. If the users' ratings of two pairs are divergent on most of the comparisons, it shows that users do not have consistent opinions on the font pairing task. Very likely the font pairing task is too subjective and could not be learnable. On the contrary, if the users have consistent opinions about which pair is superior than the other, the ratings are more convincing and could be applied to the following studies.

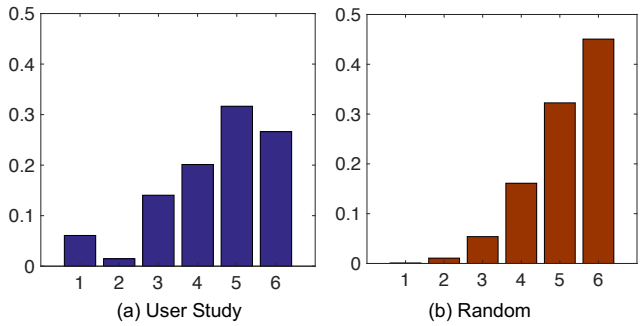


Fig. 9. pdf of the normalized difference of average users' ratings (a) and pdf of pure random ratings (b). There are six bins for both pdfs. The x-axis from left to right demonstrates the consistency from highest to lowest.

As described, we evaluate about 500 comparisons on AMT and Upwork and each comparison receives at least 11 ratings by average users and 3 ratings by designers. There are almost 150 average users and 10 designers in total.

Suppose that there are N comparisons, and for the i-th comparison, we denote the hits of pair1 as $hit1_i$ and the hits of pair2 as $hit2_i$. The normalized difference d_i of the i-th comparison is as:

$$d_i = \frac{|hit1_i - hit2_i|}{hit1_i + hit2_i}. \tag{9}$$

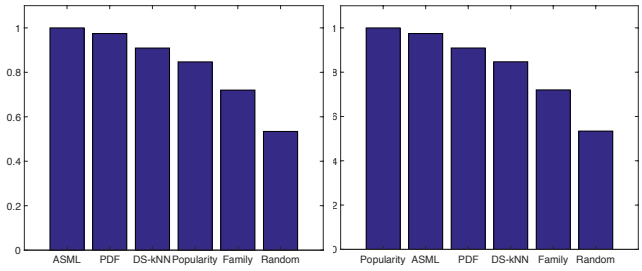
For example, assuming there are 11 ratings for one comparison, if the ratio of hits of two methods is 5:6, $d = (6-5)/(6+5) = 1/11 \simeq 0.09$. The value of d is between 0 to 1. The higher the normalized difference, the higher the consistency is.

To justify that users' ratings are consistent, we compare the distribution of users' rating with the distribution of pure random, and use hypothesis testing to test whether the two distributions are significantly distinct.

We firstly introduce calculating the rating consistency for average user on AMT. There are three steps. In the first step, we turn the continuous comparison results into binned data by grouping the comparisons into specified ranges according to d. We evenly divide [0,1] into six ranges from lowest to highest. The pdf of the normalized difference of users' ratings is shown Figure 9 (a).

In the second step, we calculate the pdf of pure random choice analytically, shown in Figure 9 (b).

In the third step, we apply χ^2 hypothesis testing to test whether these two distributions are significantly distinct. Suppose that n_j is the number of events observed in the jth bin, and e_j is the number expected according to random distribution. The χ^2 statistic is calculated as:



(a) average user (b) designer

Fig. 10. Subjective evaluation score of six methods by average user in (a) and designer in (b) by Bradley-Terry method.

$$\chi^2 = \sum_j \frac{(n_j - e_j)^2}{e_j}. \tag{10}$$

Any term j with $e_j = 0$ should be omitted from the sum. The average χ^2 is 717.43 when we sum $j = 1$ to 6. For χ^2 testing, it is also suggested to omit the bins in which $e_j < 5$. In most cases, e_1 is very small in random distribution. Thus, we also calculate χ^2 regarding $j = 2$ to 6. The average χ^2 is 117.32. According to Chi-Square Distribution Table, $\chi^2_{.005}$ = 16.750 under 6 bins and $\chi^2_{.005}$ = 14.860 under 5 bins [8]. Thus we could safely draw the conclusion that the users' ratings are consistent and significantly different ($> 99.95\%$) from the pure random distribution.

We also analyze designers' rating consistency in the same way. In about 43% pairs, all the designers make the same choice (highest consistency). When calculating the pdf of pure random choices, only 25% pairs are with the highest consistency. In hypothesis testing, when comparing the pdf of designers' ratings and pure random ratings, we could safely make a conclusion that the designers' choices are consistent and significantly different ($> 99.95\%$) from pure random distribution.

5.5.1 User Study Results

We apply the Bradley-Terry models [43] to get rankings for pairwise comparisons of ASML and DS-kNN to PDF, Random, Popularity and Family. The ranking scores of each methods based on average users' and designers' ratings are shown in Figure 10. For average users, the ranking results of these methods are ASML, PDF, DS-kNN, Popularity, Family and Random. For designers, the ranking results are Popularity, ASML, PDF, DS-kNN, Family, Random.

Average users' ratings demonstrate that ASML outperforms hand-craft methods or even the pairs extracted from PDFs. Designers would prefer popular fonts most. We analyze that designers are more familiar with these popular fonts. However, as discussed before, only recommending popular fonts maybe less interesting to designers. Other ranks of designers are the same as average users'.

5.6 Users' Rating Prediction

In this section, we want to evaluate the performance of predicting users' preference between pair1 (header A/sub-header B), and pair2 (header A/sub-header C).

8. http://kisi.deu.edu.tr/joshua.cowley/Chi-square-table.pdf

5.6.1 Experimental settings

For each comparison, the ground-truth label is the pairing which receives a higher rating from user study. We predict users' choices by each method and compare the results with ground-truth labels as prediction accuracy of each method. For both average user and designer, we only use the ratings with the highest rating consistency as the evaluation set, which are more convincing.

We compare the performance of Popularity, S-kNN, ConSim, SML, DS-kNN and ASML. In Popularity, we compare the popularity of two sub-headers, and choose the more popular sub-header as the result. In S-kNN, we calculate the distance between header and each sub-header. We choose the pair with smaller distance as the result for S-kNN. The performance of ConSim, DS-kNN, SML, ASML are evaluated in a similar way as S-kNN, but with different scoring functions for calculating the distance between header and sub-header fonts.

5.6.2 Rating prediction results

Table 3 shows the accuracy of average users' and designers' ratings prediction with comparison methods under the highest consistency level.

For predicting average users' ratings, DS-kNN and ASML achieve the highest and second highest performance respectively. For predicting designers' ratings, DS-kNN, Popularity and ASML achieve the top 3 highest performance. It is generally consistent with the user study results in Section 5.5.

When looking into S-kNN under average users' and designers' ratings, it is interesting to see that average users prefer the pairing with similar header and sub-header fonts, while designers prefer the pairing with contrast header and sub-header fonts. It shows the difficulty of predicting both tasks in the uniform scoring function. Although we can automatically implement some rules-of-thumb such as S-kNN and ConSim, in practice, it is still hard to model user's pairing taste using one or two rules-of-thumb. It is consistent with the challenge that rules and conventions of font pairing given by human experts are difficult to formalize. However, ASML and DS-kNN learn users' pairing taste from learning visual relationships from large-scale human-generated font pairs, instead of directly implementing human rules-of-thumb. DS-kNN achieves the highest performance in both tasks. ASML achieves the second and the third highest performance in two tasks, which shows the effectiveness of the learned scoring function in ASML.

We also observe that ASML outperforms DS-kNN in Top-N recommendation results but is inferior to DS-kNN in users' rating prediction. In top-N recommendation, according to the evaluation metric, the top recommended fonts have higher weights than the bottom ones. It means that if the top recommended fonts are not the same as ground truth, it is hard to achieve a high top-N accuracy. In DS-kNN, since we rank all the test bodies with the similarity score compared with candidate bodies fonts, it has a high probability that fonts similar to the ground truth are ranked higher than the ground truth itself. It degrades the top-N precision and recall score of DS-kNN. Since both fonts similar to the ground truth and the ground truth itself are good for pairing, both of them are ranked before those fonts with less satisfied pairing. Then in users' rating prediction and binary classification tasks, the order does not degrade the evaluation results.

TABLE 3
Accuracy of predicting average users' and designers' ratings with comparison methods.

	average user	designer
Popularity	55.56	57.89
S-kNN	55.33	45.45
ConSim	54.32	52.15
SML	56.22	54.59
DS-kNN(ours1)	**68.18**	**59.81**
ASML(ours2)	58.67	56.94

6 Conclusion

In this paper, we introduced the problem of visual font pairing. To our best knowledge, it is the first time that automatic font pairing has been addressed in the fields of multimedia and computer vision. We introduced a new database called FontPairing, from millions of PDF documents on the Internet. We automatically extracted header/sub-header, header/body pairs from PDF pages. We proposed two automatic font pairing methods through learning fine-grained visual relationships from large-scale human-generated font pairs: dual-space k-NN and asymmetric similarity metric learning. Comparisons are conducted against several baseline methods based on rules from professional designers. Experiments and user studies demonstrate the effectiveness of our proposed dataset and methods.

References

[1] W.-T. Chu and Y.-L. Wu, "Image style classification based on learnt deep correlation features," *IEEE Transactions on Multimedia*, 2018.

[2] J. Liu, W. Yang, X. Sun, and W. Zeng, "Photo stylistic brush: Robust style transfer via superpixel-based bipartite graph," *IEEE Transactions on Multimedia*, vol. 20, no. 7, pp. 1724–1737, 2018.

[3] Y. Wu, T. Lu, Z. Yuan, and H. Wang, "Freescup: A novel platform for assisting sculpture pose design," *IEEE Transactions on Multimedia*, vol. 19, no. 1, pp. 183–195, 2017.

[4] T. Han, H. Yao, C. Xu, X. Sun, Y. Zhang, and J. J. Corso, "Dancelets mining for video recommendation based on dance styles," *IEEE Transactions on Multimedia*, vol. 19, no. 4, pp. 712–724, 2017.

[5] S. Jiang and Y. Fu, "Fashion style generator," in *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. AAAI Press, 2017, pp. 3721–3727.

[6] X. Gu, Y. Wong, L. Shou, P. Peng, G. Chen, and M. S. Kankanhalli, "Multi-modal and multi-domain embedding learning for fashion retrieval and analysis," *IEEE Transactions on Multimedia*, 2018.

[7] P. O'Donovan, J. Lībeks, A. Agarwala, and A. Hertzmann, "Exploratory font selection using crowdsourced attributes," *ACM Transactions on Graphics*, vol. 33, no. 4, p. 92, 2014.

[8] Z. Wang, J. Yang, H. Jin, J. Brandt, E. Shechtman, A. Agarwala, Z. Wang, Y. Song, J. Hsieh, S. Kong *et al.*, "Deepfont: A system for font recognition and similarity," in *Proceedings of the 23rd ACM international conference on Multimedia*. ACM, 2015, pp. 813–814.

[9] Z. Wang, J. Yang, H. Jin, E. Shechtman, A. Agarwala, J. Brandt, and T. S. Huang, "Deepfont: Identify your font from an image," in *Proceedings of the 23rd ACM international conference on Multimedia*. ACM, 2015, pp. 451–459.

[10] N. Zhao, Y. Cao, and R. W. Lau, "Modeling fonts in context: Font prediction on web designs," *Computer Graphics Forum*, vol. 37, no. 7, pp. 385–395, 2018.

[11] J. Ma, W. Shao, H. Ye, L. Wang, H. Wang, Y. Zheng, and X. Xue, "Arbitrary-oriented scene text detection via rotation proposals," *IEEE Transactions on Multimedia*, 2018.

[12] X. Ren, Y. Zhou, J. He, K. Chen, X. Yang, and J. Sun, "A convolutional neural network-based chinese text detection algorithm via text structure modeling," *IEEE Transactions on Multimedia*, vol. 19, no. 3, pp. 506–518, 2017.

[13] L. Wu, P. Shivakumara, T. Lu, and C. L. Tan, "A new technique for multi-oriented scene text line detection and tracking in video." *IEEE Transactions on Multimedia*, vol. 17, no. 8, pp. 1137–1152, 2015.

[14] C. Yan, H. Xie, J. Chen, Z.-J. Zha, X. Hao, Y. Zhang, and Q. Dai, "An effective uyghur text detector for complex background images," *IEEE Transactions on Multimedia*, 2018.

[15] X. Yang, T. Mei, Y.-Q. Xu, Y. Rui, and S. Li, "Automatic generation of visual-textual presentation layout," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 12, no. 2, p. 33, 2016.

[16] A. Kembhavi, M. Salvato, E. Kolve, M. Seo, H. Hajishirzi, and A. Farhadi, "A diagram is worth a dozen images," in *Proceedings of the European Conference on Computer Vision*. Springer, 2016, pp. 235–251.

[17] N. Siegel, Z. Horvitz, R. Levin, S. Divvala, and A. Farhadi, "Figureseer: Parsing result-figures in research papers," in *Proceedings of the European Conference on Computer Vision*. Springer, 2016, pp. 664–680.

[18] S. Jiang, X. Qian, J. Shen, Y. Fu, and T. Mei, "Author topic model-based collaborative filtering for personalized poi recommendations," *IEEE Transactions on Multimedia*, vol. 17, no. 6, pp. 907–918, 2015.

[19] S. Huang, J. Zhang, D. Schonfeld, L. Wang, and X.-S. Hua, "Two-stage friend recommendation based on network alignment and series expansion of probabilistic topic model," *IEEE Transactions on Multimedia*, vol. 19, no. 6, pp. 1314–1326, 2017.

[20] Y.-Y. Ahn, S. E. Ahnert, J. P. Bagrow, and A.-L. Barabási, "Flavor network and the principles of food pairing," *Scientific reports*, vol. 1, p. 196, 2011.

[21] T. Liu, A. Hertzmann, W. Li, and T. Funkhouser, "Style compatibility for 3d furniture models," *ACM Transactions on Graphics*, vol. 34, no. 4, p. 85, 2015.

[22] C. Galleguillos, A. Rabinovich, and S. Belongie, "Object categorization using co-occurrence, location and appearance," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2008, pp. 1–8.

[23] L. Ladicky, C. Russell, P. Kohli, and P. H. Torr, "Graph cut based inference with co-occurrence statistics," in *European Conference on Computer Vision*. Springer, 2010, pp. 239–253.

[24] L. Feng and B. Bhanu, "Semantic concept co-occurrence patterns for image annotation and retrieval," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 4, pp. 785–799, 2016.

[25] J. McAuley, C. Targett, Q. Shi, and A. van den Hengel, "Image-based recommendations on styles and substitutes," in *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2015, pp. 43–52.

[26] A. Veit, B. Kovacs, S. Bell, J. McAuley, K. Bala, and S. Belongie, "Learning visual clothing style with heterogeneous dyadic co-occurrences," in *The IEEE International Conference on Computer Vision*, 2015, pp. 4642–4650.

[27] J. McAuley, R. Pandey, and J. Leskovec, "Inferring networks of substitutable and complementary products," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2015, pp. 785–794.

[28] V. Jagadeesh, R. Piramuthu, A. Bhardwaj, W. Di, and N. Sundaresan, "Large scale visual recommendations from street fashion images," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2014, pp. 1925–1934.

[29] S. Liu, J. Feng, Z. Song, T. Zhang, H. Lu, C. Xu, and S. Yan, "Hi, magic closet, tell me what to wear!" in *Proceedings of the 20th ACM international conference on Multimedia*. ACM, 2012, pp. 619–628.

[30] L.-F. Yu, S. K. Yeung, D. Terzopoulos, and T. F. Chan, "Dressup!: outfit synthesis through automatic optimization." *ACM Transactions on Graphics*, vol. 31, no. 6, pp. 134–1, 2012.

[31] D. Bonneville, *The Big Book of Font Combinations*. BonFX Press, 2010.

[32] G. Chen, J. Yang, H. Jin, J. Brandt, E. Shechtman, A. Agarwala, and T. X. Han, "Large-scale visual font recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 3598–3605.

[33] Z. Wang, J. Yang, H. Jin, E. Shechtman, A. Agarwala, J. Brandt, and T. S. Huang, "Real-world font recognition using deep network and domain adaptation," *arXiv preprint arXiv:1504.00028*, 2015.

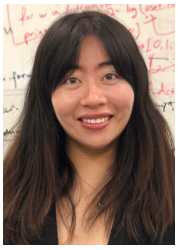
[34] R. Kumar, A. Satyanarayan, C. Torres, M. Lim, S. Ahmad, S. R. Klemmer, and J. O. Talton, "Webzeitgeist: design mining the web," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2013, pp. 3083–3092.

[35] A. Aizawa, "An information-theoretic perspective of tf–idf measures," *Information Processing & Management*, vol. 39, no. 1, pp. 45–65, 2003.

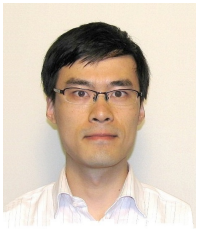
[36] K. Q. Weinberger and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification," *Journal of Machine Learning Research*, vol. 10, no. Feb, pp. 207–244, 2009.

[37] Y. Wu, S. Wang, W. Zhang, and Q. Huang, "Online low-rank similarity function learning with adaptive relative margin for cross-modal retrieval,"

in *2017 IEEE International Conference on Multimedia and Expo*. IEEE, 2017, pp. 823–828.
[38] Y. Wu, S. Wang, G. Song, and Q. Huang, "Online asymmetric metric learning with multi-layer similarity aggregation for cross-modal retrieval," *IEEE Transactions on Image Processing*, 2019.
[39] Q. Cao, Y. Ying, and P. Li, "Similarity metric learning for face recognition," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 2408–2415.
[40] "Font pairing made simple," http://fontjoy.com/, [Online].
[41] A. Gunawardana and G. Shani, "A survey of accuracy evaluation metrics of recommendation tasks," *Journal of Machine Learning Research*, vol. 10, no. Dec, pp. 2935–2962, 2009.
[42] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information processing & management*, vol. 24, no. 5, pp. 513–523, 1988.
[43] D. R. Hunter *et al.*, "Mm algorithms for generalized bradley-terry models," *The annals of statistics*, vol. 32, no. 1, pp. 384–406, 2004.



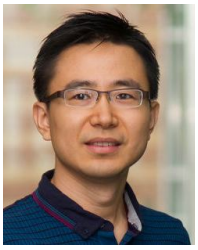
Shuhui Jiang received the B.S. and M.S. degrees in Xi'an Jiaotong University, Xi'an, China, in 2007 and 2011, respectively, and the Ph.D. degree in electrical and computer engineering from School of Electrical and Computer Engineering, Northeastern University (Boston, USA). She was the recipient of the Dean's Fellowship of Northeastern University from 2014. She is interested in machine learning, multimedia and computer vision. She has served as the reviewers for IEEE journals: IEEE Transactions on Multimedia, IEEE Transactions on Neural Networks and Learning Systems, etc. She was a research intern with Adobe research lab, San Jose, US, in summer 2016.



Zhaowen Wang (M'14) received the B.E. and M.S. degrees from Shanghai Jiao Tong University, China, in 2006 and 2009, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Illinois at Urbana-Champaign, in 2014. He is currently a Senior Research Scientist with the Creative Intelligence Lab, Adobe Inc. His research has been focused on understanding and enhancing images, videos and graphics via machine learning algorithms, with a particular interest in sparse coding and deep learning.

Aaron Hertzmann is a Principal Scientist at Adobe and an ACM Fellow. He received a BA in Computer Science and Art/Art History from Rice University in 1996, and a PhD in Computer Science from New York University in 2001. He was a Professor at University of Toronto from 2003 to 2013, and has also worked at Pixar Animation Studios, University of Washington, Microsoft Research, Mitsubishi Electric Research Lab, Interval Research Corporation and NEC Research. He was an Associate Editor for *ACM Transactions on Graphics*, for ten years. His awards include the MIT TR100 (2004), a Sloan Foundation Fellowship (2006), a Microsoft New Faculty Fellowship (2006), the CACS/AIC Outstanding Young CS Researcher Award (2010), and the Steacie Prize for Natural Sciences (2010), as well as several conference best paper awards.

Hailin Jin received the bachelors degree in Automation from Tsinghua University, Beijing, China, in 1998, and the M.S. and D.Sc. degrees in Electrical Engineering from Washington University in Saint Louis, in 2000 and 2003, respectively. From fall 2003 to fall 2004, he was a Post-Doctoral Researcher at the Computer Science Department, University of California at Los Angeles. Since 2004, he has been with Adobe Research, where he is currently a Senior Principal Scientist. He received the Best Student Paper Award (with J. Andrews and C. Sequin) at the 2012 International CAD Conference for work on interactive inverse 3D modeling. He is a member of the IEEE Computer Society.



Yun Fu (S'07-M'08-SM'11-F'19) received the B.Eng. degree in information engineering and the M.Eng. degree in pattern recognition and intelligence systems from Xi'an Jiaotong University, China, respectively, and the M.S. degree in statistics and the Ph.D. degree in electrical and computer engineering from the University of Illinois at Urbana-Champaign, respectively. He is an interdisciplinary faculty member affiliated with College of Engineering and the Khoury College of Computer and Information Sciences at Northeastern University since 2012. His research interests are Machine Learning, Computational Intelligence, Big Data Mining, Computer Vision, Pattern Recognition, and Cyber-Physical Systems. He serves as associate editor, chairs, PC member and reviewer of many top journals and international conferences/workshops. He received seven Prestigious Young Investigator Awards from NAE, ONR, ARO, IEEE, INNS, UIUC, Grainger Foundation; eleven Best Paper Awards from IEEE, ACM, IAPR, SPIE, SIAM; many major Industrial Research Awards from Google, Samsung, and Adobe, etc. He is currently an Associate Editor of the IEEE, TNNLS. He is fellow of IEEE, IAPR, OSA and SPIE, a Lifetime Distinguished Member of ACM, Lifetime Member of AAAI and Institute of Mathematical Statistics, member of ACM Future of Computing Academy, Global Young Academy, AAAS, INNS and Beckman Graduate Fellow during 2007-2008.