frontiers | Frontiers in Artificial Intelligence

TYPE Original Research
PUBLISHED 29 April 2024
DOI 10.3389/frai.2024.1382566

OPEN ACCESS

EDITED BY
Sara Ramezani,
University of Tehran, Iran

REVIEWED BY
Alberto Ochoa Zezzatti,
Universidad Autónoma de Ciudad Juárez, Mexico
Salim Karimi Takalo,
Vali-E-Asr University of Rafsanjan, Iran
Mahdi Dehghani Soltani,
Vali-E-Asr University of Rafsanjan, Iran

*CORRESPONDENCE
Miguel A. Casal
✉ miguel.casal@udc.es
Juan Romero
✉ jj@udc.es

RECEIVED 05 February 2024
ACCEPTED 08 April 2024
PUBLISHED 29 April 2024

CITATION
Grelier G, Casal MA, Torrente-Patiño A and Romero J (2024) Image sequence sorting algorithm for commercial tasks.
Front. Artif. Intell. 7:1382566.
doi: 10.3389/frai.2024.1382566

Image sequence sorting algorithm for commercial tasks

Guillaume Grelier[1], Miguel A. Casal[2]*, Alvaro Torrente-Patiño[1,2] and Juan Romero[1,2]*

[1]PhotoILike, Bergondo, Spain, [2]RNASA Lab-IMEDIR, Department of Computer Science and Information Technologies, Faculty of Communication Science, University of A Coruña, A Coruña, Spain

Introduction: The sorting of sequences of images is crucial for augmenting user engagement in various virtual commercial platforms, particularly within the real estate sector. A coherent sequence of images respecting room type categorization significantly enhances the intuitiveness and seamless navigation of potential customers through listings.

Methods: This study methodically formalizes the challenge of image sequence sorting and expands its applicability by framing it as an ordering problem. The complexity lies in devising a universally applicable solution due to computational demands and impracticality of exhaustive searches for optimal sequencing. To tackle this, our proposed algorithm employs a shortest path methodology grounded in semantic similarity between images. Tailored specifically for the real estate sector, it evaluates diverse similarity metrics to efficiently arrange images. Additionally, we introduce a genetic algorithm to optimize the selection of semantic features considered by the algorithm, further enhancing its effectiveness.

Results: Empirical evidence from our dataset demonstrates the efficacy of the proposed methodology. It successfully organizes images in an optimal sequence across 85% of the listings, showcasing its effectiveness in enhancing user experience in virtual commercial platforms, particularly in real estate.

Conclusion: This study presents a novel approach to sorting sequences of images in virtual commercial platforms, particularly beneficial for the real estate sector. The proposed algorithm effectively enhances user engagement by providing more intuitive and visually coherent image arrangements.

KEYWORDS

image ordering, semantic representation, image embedding, real estate, evolutionary computing, feature selection

1 Introduction

In the digital age, the potency of e-commerce hinges significantly on the art of presentation, especially within the visual-centric realm of online marketing. A paramount aspect of this presentation is the strategic deployment of images to convey accurate and appealing product narratives. This strategy, however, faces its quintessential challenge in ensuring the transmission of precise information (Jin and Gallimore, 2010). The conventional reliance on detailed written descriptions is undeniably foundational, yet the transformative power of images in enhancing these descriptions cannot be overstated. Quality imagery has a critical role in capturing attention and bolstering consumer confidence, ultimately driving purchase intentions (Chen and Teng, 2013; Di et al., 2014; Zakrewsky et al., 2016).

This visual imperative is particularly pronounced in the realm of real estate marketing. Here, the breadth and quality of property images directly influence potential buyers' interest (Thaler and Koch, 2023), and distinguish listings in a crowded market (Siebert and Seiler, 2022). The crux of effective visual presentation, therefore, lies in the intuitive and logical sequencing of these images (David Koch and Mayr, 2021). A coherent arrangement

that clusters images by room or similarity not only eases navigation but significantly reduces cognitive strain, thereby enhancing the overall decision-making process (Cacioppo et al., 1996; Garbarino and Edell, 1997; Franco-Watkins et al., 2006).

Yet, the rapidly evolving real estate sector, with its increasing tilt toward automation in listing processes, often overlooks the nuanced task of image sequencing. The indiscriminate uploading of property photos without regard to their order poses a glaring challenge, highlighting a gap in the automated sorting mechanisms explored in other domains (Chaudhuri et al., 2018; Fan et al., 2022). The pivotal question, then, revolves around developing an algorithm capable of organizing these images in a manner that respects the inherent grouping without prior knowledge of these groups' specifics.

This problem can be seen and formalized as an Ordering Problem (see Section 2.1). While one approach might involve using a clustering algorithm to generate potential sub-groups, ordering elements based on these sub-groups can be challenging. If the number of clusters obtained from the algorithm differs from the initial number of subsets, achieving an accurate final order becomes unlikely. Therefore, this paper suggests an alternative method that increases the likelihood of obtaining an accurate order without requiring prior knowledge of the number of clusters. Note that the challenge arises precisely from not knowing the exact number of distinct rooms present; otherwise, applying a clustering algorithm would suffice.

Then, addressing the ordering problem requires a foundational shift toward a methodology grounded in the principle of similarity. By devising a similarity metric among the images, we aim at defining an algorithm solution to organize the elements in a way that those belonging to the same subset (e.g., images of the same room) are placed adjacently, thereby preserving the partition's integrity. This approach, while methodical, navigates the complexity of limited knowledge about the partition's structure and the count of subsets, challenging the straightforward application of clustering algorithms.

2 Methods

2.1 Ordering problem

An ordering problem consists of finding a permutation that maximizes (or minimizes) a certain quantity, and it can be mathematically formalized in a general manner, focusing on identifying permutations of elements in set $\mathcal{A}$ that preserve the structure of a given partition $\mathcal{P}$. In simpler terms, the objective is to find orders of elements in $\mathcal{A}$ such that elements belonging to each subset within partition $\mathcal{P}$ appear consecutively (see Definition 2.2). To illustrate, consider $\mathcal{A}$ as a collection of bedroom photos, with $\mathcal{P}$ being a partition where each subset comprises photos of the same bedroom.

For notation purposes, let $n \geq 1$ be a finite set, and S_n denote the symmetric group of order n. The cardinality of a set A is denoted by $|A|$. Any $\sigma \in S_n$ is referred to as a permutation. The vector space of real square matrices of size n is denoted by $\mathcal{M}_n(\mathbb{R})$.

The ordering problem can be seen as a variation on the traveling salesman problem, where the aim is to travel through all the towns once and only once, minimizing the total distance and without returning to the starting point. Formally, the definition is the following:

Definition 2.1. Let $M = (m_{i,j})_{1\leq i,j\leq n} \in \mathcal{M}_n(\mathbb{R})$ and $S \subset S_n$. The **ordering problem associated to M and S** is the following optimization problem:

$$\max_{\sigma\in S} \sum_{i=1}^{n-1} m_{\sigma(i),\sigma(i+1)}.$$

For simplicity, we define the function $f_M : S_n \to \mathbb{R}$ by $f_M(\sigma) = \sum_{i=1}^{n-1} m_{\sigma(i),\sigma(i+1)}$. The ordering problem associated to M and S consists in maximizing f_M over S.

Of course, since the set over which we are minimizing is finite, there is always a solution. The difficulty lies in finding a solution computationally in a reasonable time. In fact, it can be proved that the previous problem is an NP-hard problem in combinatorial optimization when $S = S_n$ (Opatrny, 1979).

There are many well-known algorithmic methods for solving this kind of problem. If n is small, it is possible to go through all the permutations of S_n in order to maximize f_M. However, if n is large (typically, when $n > 10$), the cardinal of S_n being $n!$ we need to find an alternative method. There are a number of methods for finding an approximate solution to the traveling salesman problem. Here we will quickly present a well-known algorithm that is easy to implement, fast and more than sufficient for our purposes: the nearest neighbor algorithm. Depending on the problem, it may of course be necessary to use a more complex algorithm to maximize f_M since, in general, the nearest neighbor algorithm does not give an optimal solution.

Let $M = (s(x_i, x_j))_{1\leq i,j\leq n} \in \mathcal{M}_n(\mathbb{R})$. Define f_M as in Definition 2.1. Let x_{i_1} be an aleatory starting point. We then choose the element x_{i_2} closest to it, i.e., the one that maximizes the quantity $s(x_{i_1}, x_{i_2})$. We repeat the process, looking for the element x_{i_3} closest to x_{i_2}. When all the elements have been used, we stop. For each starting point x_i, we obtain a permutation σ_i. The final permutation is obtained by choosing the σ_i permutation satisfying $f_M(\sigma_i) = \max_{1\leq j\leq n} f_M(\sigma_j)$. The pseudo-code detailed in Algorithm 1:

Parameters: a matrix M
For $i \in \{1, ..., n\}$:
  Define $\sigma_i(1) = i$
  Define $J = \{1, ..., n\} \setminus \{i\}$
  Define the current element $x = i$
  While $J \neq \emptyset$:
    Choose $j \in J$ such that $m_{j,x} = \max_{k\in J} m_{k,x}$
    Define $\sigma_i(n - |J| + 1) = j$
    Update $J \longleftarrow J \setminus \{j\}$
    Update $x \longleftarrow j$
Define $f_M(\sigma) = \sum_{i=1}^{n-1} |m_{\sigma(i),\sigma(i+1)}|$
Return argmax$(f_{M|\{\sigma_1,...,\sigma_n\}})$

Algorithm 1. Nearest neighbor algorithm.

In many applications, a starting point can be necessary. For example, in our case, it might be interesting to choose the photo with the highest commercial appeal as the initial point. If a starting

point x_{i_0} is required, the for loop should of course be removed and only the permutations σ such that $x_{\sigma(1)} = x_{i_0}$ are considered.

2.2 Ordering algorithm with unknown partition

Definition 2.2. Let $n \geq 1$ and let $\mathcal{A} = \{x_1, ..., x_n\}$ be a set of cardinal n. Let $\mathcal{P} = \{A_1, ..., A_p\}$ be a partition of $\mathcal{A}$. We say that a permutation $\sigma \in S_n$ **preserves** $\mathcal{P}$ if

$$\forall i \in \{1, ..., p\} \; \exists j \in \{1, ..., n\} \text{ such that } x_{\sigma(j)}, x_{\sigma(j+1)}, ..., x_{\sigma(j+|A_i|-1)} \in A_i,$$

i.e., for all $i \in \{1, ..., p\}$, the elements of A_i are found successively in the ordered sequence $x_{\sigma(1)}, x_{\sigma(2)}, ..., x_{\sigma(n)}$. The set of permutations preserving $\mathcal{P}$ will be denoted by $\mathcal{O}(\mathcal{P})$.

The partition $\mathcal{P}$ is not known, and we want to find a method to discover a permutation that preserves $\mathcal{P}$. We will see that such permutations can be obtained by solving an ordering problem associated with a well-chosen matrix M.

In our case, $\mathcal{O}(\mathcal{P})$ represents the set of permutations in which the photos belonging to the same room are consecutive. Since the objective is to determine the permutations that preserve $\mathcal{P}$, it is useful to determine how many there are.

Proposition 2.3. Let $\mathcal{A}$ be a finite set and let $\mathcal{P} = \{A_1, ..., A_p\}$ be a partition of $\mathcal{A}$. Then $|\mathcal{O}(\mathcal{P})| = p! \prod_{k=1}^{p} |A_k|!$.

Proof. There are $p!$ ways of ordering the sets A_i. Then, in each set A_i, there are $|A_i|!$ ways of permuting its elements. The result is then obtained.

Although intuitively trivial, we deduce that any permutation preserves $\mathcal{P}$ if and only if $p = 1$ or $p = n$. This corresponds to cases where the partition has only one set, or where all the sets in the partition are singletons.

Since $\mathcal{P}$ is not necessarily known, we have to find a way to determine the set of permutations preserving $\mathcal{P}$. One way of doing this is to determine the functions defined on S_n whose maxima are permutations that preserve $\mathcal{P}$:

Definition 2.4. Let $f : S_n \to \mathbb{R}$. We say that f is **compatible with** $\mathcal{P}$ if $\text{argmax}(f) \subset \mathcal{O}(\mathcal{P})$ and we write $f \in C(\mathcal{P})$. If moreover, $f \geq 0$, we write $f \in C^+(\mathcal{P})$.

It is clear that there always exists a function f which is compatible with $\mathcal{P}$. In fact, let $\Gamma \subset \mathcal{O}(\mathcal{P})$ and define f by

$$f(\sigma) = \begin{cases} 1 & \text{if } \sigma \in \Gamma \\ 0 & \text{if not} \end{cases}$$

It is clear that $f \in C(\mathcal{P})$. The following result shows that there are many functions compatible with $\mathcal{P}$. We recall that $\mathcal{C}(S_n)$ is the set of continuous real functions defined on S_n (since S_n is discrete, this refers to simply real functions defined on S_n).

Proposition 2.5. $\mathcal{C}(S_n) = C^+(\mathcal{P}) - C^+(\mathcal{P})$.

Proof. Let $f \in \mathcal{C}(S_n)$ such that $f \neq 0$. Since $C^+(\mathcal{P})$ is stable by adding a positive constant, we can and do suppose that $f > 0$. We define $f_1, f_2 \in \mathcal{C}(S_n)$ by

$$f_1(\sigma) = \begin{cases} f(\sigma) & \text{if } \sigma \in \mathcal{O}(\mathcal{P}) \\ 0 & \text{if not} \end{cases} \quad \text{and } f_2(\sigma) = \begin{cases} 0 & \text{if } \sigma \in \mathcal{O}(\mathcal{P}) \\ -f(\sigma) & \text{if not} \end{cases}.$$

It is clear that $f = f_1 - f_2$. Since $f > 0$, f_1 and f_2 reached their maxima on $\mathcal{O}(P)$, i.e., $f_1, f_2 \in C(\mathcal{P})$. Now pick $\alpha > 0$ such that $g_1 := f_1 + \alpha > 0$ and $g_2 := f_2 + \alpha > 0$. We have $f = g_1 - g_2 \in C^+(\mathcal{P}) - C^+(\mathcal{P})$.

If $\mathcal{P}$ is unknown, it can therefore be difficult to determine a function f which is compatible with $\mathcal{P}$. Depending on the desired order, the f function must be chosen on the basis of the data so that it is compatible with $\mathcal{P}$. Then, by maximizing f, we will obtain an acceptable solution associated to $\mathcal{P}$.

We are going to look for functions that are compatible with $\mathcal{P}$ and have a particular form:

Definition 2.6. Let $M = (m_{i,j})_{1 \leq i,j \leq n} \in \mathcal{M}_n(\mathbb{R})$. The permutations maximising the function $f_M : S_n \to \mathbb{R}$ defined by

$$f_M(\sigma) = \sum_{i=1}^{n-1} m_{\sigma(i),\sigma(i+1)}$$

are called **longest paths of** M. We say that M is **compatible with** $\mathcal{P}$ if f_M is compatible with $\mathcal{P}$. In other words, M is compatible with $\mathcal{P}$ if any of its longest paths is a permutation preserving $\mathcal{P}$.

One can ask if any function of $C(\mathcal{P})$ is of the form f_M for some $M \in \mathcal{M}_n(\mathbb{R})$. Unfortunately it is easily seen that $\{f_M\}_{M \in \mathcal{M}_n(\mathbb{R})}$ is a strict subset of $C(\mathcal{P})$. In fact, the map $T : \mathcal{M}_n(\mathbb{R}) \to \mathcal{C}(S_n)$ defined by $T(M) = f_M$ is linear. It follows that $\{f_M\}_{M \in \mathcal{M}_n(\mathbb{R})} \subset \text{Im}(T)$ with $\dim(\text{Im}(T)) \leq n^2$. We have that $\dim(\text{span}(C(\mathcal{P}))) = n!$ by Proposition 2.5. In particular, if $n > 1$, we have that $\{f_M\}_{M \in \mathcal{M}_n(\mathbb{R})} \subsetneq C(\mathcal{P})$. Although the f_M functions form a small subset of $C(\mathcal{P})$, they are the functions that are most natural to define in practice. In fact, the matrix M can be seen as a similarity matrix where $m_{i,j}$ is the similarity coefficient between x_i and x_j.

Definition 2.7. Let $s : \mathcal{A}^2 \to \mathbb{R}$ and define $M = (s(x_i, x_j))_{1 \leq i,j \leq n}$. We say that s **describes** $\mathcal{P}$ if M is compatible with $\mathcal{P}$.

The *best* function describing $\mathcal{P}$ is defined as follows:

Example 2.8. Let $s : \mathcal{A}^2 \to \mathbb{R}$ defined by $s(x_i, x_j) = 1$ if there exists $k \in \{1, ..., p\}$ such that $i, j \in A_k$ and $s(x_i, x_j) = 0$ if not. Define $M = (s(x_i, x_j))_{1 \leq i,j \leq n}$. It is easy to see that $f_M(\sigma) \leq n - p$ for all $\sigma \in S_n$ with equality if and only if $\sigma \in \mathcal{O}(P)$. In particular, M is compatible with $\mathcal{P}$, i.e., s describes $\mathcal{P}$.

We are now ready to propose a simple algorithm (Algorithm 2) to find permutations that are compatible with a given partition:

Parameters: a set $\mathcal{A}$ and a partition $\mathcal{P}$ of $\mathcal{A}$
Define a measure of similarity $s : \mathcal{A}^2 \to \mathbb{R}^+$
Compute $M = (s(x_i, x_j))_{1 \leq i,j \leq n}$
Solve the ordering problem associated to M and S_n, i.e., maximize f_M on S_n
Return argmax(f_M)

Algorithm 2. Ordering algorithm.

The entire challenge lies in finding a similarity measure that captures the essence of $\mathcal{P}$. In our case, the goal is to discover a similarity measure where the value is high for photos of the same room and low for photos of different rooms.

It is possible that the longest path does not pass through two points with high similarity. This phenomenon is amplified in practice when a starting point is chosen. Consider the similarity matrix

$$M = (s(x_i, x_j))_{1 \leq i,j \leq 4} = \begin{pmatrix} 1 & 0.5 & 0.7 & 0.5 \\ 0.5 & 1 & 0.5 & 0.2 \\ 0.7 & 0.5 & 1 & 0.5 \\ 0.5 & 0.2 & 0.5 & 1 \end{pmatrix}.$$

Suppose we want our starting point to be x_1. In view of the similarity matrix, points x_1 and x_3 are the most similar. We therefore expect them to be part of the same group and the optimal path to visit them successively. However, the maximum path starting at x_1 is given by x_1, x_2, x_3, x_4 and has a length of 1.5, while the path x_1, x_3, x_2, x_4 has a length of 1.4. It can therefore be useful to force the algorithm to pass successively through pairs of points with a high similarity. For this reason, we propose the following modification (Algorithm 3) to the previous algorithm that can be applied to any matrix M.

Parameters: a matrix M and a threshold $\alpha \in \mathbb{R}$
Define $M' = (m'_{i,j})_{1 \leq i,j \leq n} = M$
Define $\mathcal{S} = S_n$ and $\mathcal{S}' = \mathcal{S}$
While $m = \max_{i \neq j} m'_{i,j} \geq \alpha$ and $\mathcal{S} \neq \emptyset$:
  $\mathcal{S}' = \mathcal{S}$
  Pick $i_0 \neq j_0$ such that $m'_{i_0,j_0} = m$
  Remove from $\mathcal{S}$ the permutations σ such that there does not exist k such that $\sigma(k) = i_0$ and $\sigma(k+1) = j_0$, or $\sigma(k) = j_0$ and $\sigma(k+1) = i_0$
  $m'_{i_0,j_0} = \min_{i \neq j} m'_{i,j}$ and $m'_{j_0,i_0} = \min_{i \neq j} m'_{i,j}$
Solve the ordering problem associated to M and S_n, i.e., maximize f_M on $\mathcal{S}'$
Return argmax($f_{M|\mathcal{S}'}$)

Algorithm 3. Improved ordering algorithm.

The previous algorithm forces pairs of elements whose similarity is sufficiently high to follow each other in the final order. Of course, if a starting point x_{i_0} is desired, then $\mathcal{S}$ should be defined as the subset of S_n consisting of the permutations σ satisfying $x_{\sigma(1)} = x_{i_0}$. It should be borne in mind that the use of a starting point, although necessary for certain applications, can only worsen the performance of the algorithm.

2.3 Dataset

Our dataset comprises 295 property ads, each containing varying numbers of photos. Each photo is labeled, identifying the specific room category it belongs to (e.g., bedroom, living room, exterior,...). For the purpose of this analysis, we focus solely on bedroom photos within each property listing, since it is likely to find a varying number of bedrooms across ads. Therefore, our dataset can be represented as a collection of sets: $\mathcal{C} = \{\mathcal{A}_i\}_{1 \leq i \leq 295}$, where each $\mathcal{A}_i$ corresponds to a set of n_i bedroom photos (with potential errors from the initial model, although this doesn't impact the ordering algorithm's structure). For each $\mathcal{A}_i$, the objective is to arrange the photos in such a manner that pictures of the same bedroom are contiguous. Consequently, for each $\mathcal{A}_i$, there exists an unknown partition $\mathcal{P}_i$ of $\mathcal{A}_i$, where each subset comprises photos of the same bedroom. An example of bedroom-categorized advertisement photos is shown in Figure 1.

In case $|\mathcal{P}_i| = 1$ or $|\mathcal{P}_i| = n_i$, any order would be a solution (see Proposition 2.3). We will therefore eliminate the elements $\mathcal{A}_i$ verifying this condition since, regardless of the model, the result will be good. After these eliminations, our dataset $\mathcal{C}$ contains 211 elements. In order to avoid over-complicating the notations, we will still write $\mathcal{C} = \{\mathcal{A}_i\}_{1 \leq i \leq k}$ with $k = 211$ and where each $\mathcal{A}_i$ is a set of n_i.

2.4 Application to image ordering

Recall that our dataset a set of $\mathcal{C} = \{\mathcal{A}_i\}_{1 \leq i \leq k}$, where $k = 211$ and each $\mathcal{A}_i$ is a set of n_i bedroom photos. For each A_i, the objective is to order the photos in such a way that the photos of the same room follow one another. That is, if $\mathcal{P}_i$ is the unknown partition of A_i where each set of which consists of photos of the same room, we want to find a permutation preserving $\mathcal{P}_i$.

2.4.1 Embedding vectors

An embedding vector serves as a condensed representation that encapsulates the informational essence held within an image or text (Gutiérrez and Keith, 2019; Schwalbe, 2022). Through a complex process of feature extraction and transformation, an image is translated into a numerical vector that captures key visual characteristics and patterns. This vector, known as the embedding vector, effectively distills the image's relevant information into a more compact and manageable format. This process enables efficient storage, comparison, and analysis of images, making it an indispensable tool in various applications such as image retrieval, similarity assessment, and even machine learning tasks where image data is a crucial input.

This embedding process is commonly achieved through the utilization of a neural network known as an image encoder. The image encoder is specifically designed to take raw image data as input and transform it layer by layer into a meaningful and compact embedding vector. This network leverages its learned weights and architecture to extract hierarchical features from the image, gradually abstracting away from low-level pixel information to higher-level semantic representations. As the image progresses

FIGURE 1
Photos of bedrooms in an ad.

through the layers of the encoder, its distinctive features and attributes are progressively distilled into the final embedding vector, creating a powerful and condensed representation that captures the image's essence. This integration of deep learning techniques within the image embedding process enhances the ability to capture intricate patterns, allowing for more effective downstream tasks that rely on these embeddings.

Notation 2.9. When using an image encoder e, the dimension of the embedding vector will be denoted as m. Moreover, if I is an image, the corresponding embedding vector will be denoted by $e(I) \in \mathbb{R}^m$.

In our case, we will use the image encoders of two algorithms: CLIP (Radford et al., 2021) and BLIP (Li et al., 2022), two of the currently most effective algorithms.

2.4.1.1 CLIP

The embedding vector obtained thanks to the image encoders of the CLIP algorithm is a vector of a vector of $m = 512$ coordinates. To obtain this image encoding, CLIP uses a deep neural network called the "image encoding network". This network is a crucial part of the overall CLIP model, which combines image encoding with text encoding to perform multimodal tasks. The image encoding network is pretrained on a large dataset of images from the internet in an unsupervised manner before being fine-tuned using the contrastive process, as mentioned in the previous response. Through this training, the network learns to extract important and semantically meaningful visual features from images, creating a 512-dimensional vector that captures key information about the image.

Notation 2.10. The image encoder of CLIP will be denoted by c.

2.4.1.2 BLIP

The BLIP image encoder returns an embedding matrix of dimensions 32×256 (meaning that $m = 8192$). They incorporate a visual transformer as our image encoder, which breaks down input images into patches and converts them into a sequence of embeddings. An additional token is included to represent the overall image feature. This choice offers advantages over the use of pretrained object detectors for visual feature extraction, as it is computationally more efficient. This approach has gained popularity in recent methodologies for its effectiveness in image processing tasks.

Notation 2.11. The image encoder of BLIP will be denoted by b.

2.4.2 Image ordering algorithm

In order to compare two embedding vectors, we will consider a similarity measure $S:(\mathbb{R}^m)^2 \to \mathbb{R}^+$ and then calculate the arithmetic mean of this measure evaluated on the images themselves and on their symmetrical images. We then obtain a similarity measure s which will be used to order the images. Its calculation is given in the following algorithm. Before that, we just introduce the following notation:

Notation 2.12. Let $m \geq 1$. If $x \in \mathbb{R}^m$ and $v \in \mathbb{R}^m$ is a vector such that $v_i \in \{0, 1\}$ for all $i \in \{1, ..., m\}$, we denote by x_v the vector given by $v_x = \sum_{i=1}^{m} v_i x_i e_i$ where $(e_i)_{1 \leq i \leq m}$ is the canonical basis of $\mathbb{R}^m$.

Many image encoders are sensitive to axial symmetries due to the nature of the neural networks it employs. The deep neural networks are designed to learn and recognize patterns and features in images. When the image is transformed by axial symmetry (horizontal or vertical flipping), the spatial arrangement

of visual elements changes. Consequently, the network may extract different features from the transformed image compared to the original one. Since these algorithms encode images based on these extracted features, the resulting feature vector representing the transformed image will differ from the one representing the original image. As a result, when these algorithms compare an image with its axially symmetrical counterpart or with textual descriptions, the differences in feature extraction can influence the perceived similarity between the images or the alignment with the provided text. This is why we propose the following computation (Algorithm 4) of the similarity matrix:

Parameters: a image encoder e, a set of images $\mathcal{A} = \{I_1, ..., I_n\}$, a similarity measure $S : (\mathbb{R}^m)^2 \to \mathbb{R}^+$ and a binary vector v
For $i \in \{1, ..., n\}$:
    Compute I_i^s the image obtained from I_i by vertical axial symmetry
    Compute $e(I_i)$ and $e(I_i^s)$
For $i, j \in \{1, ..., n\}$ such that $i \leq j$:
    If $i = j$:
        $s(i, j) = 1$
    Else:
        $s(i, j) = \frac{1}{3}\left(S(e(I_i)_v, e(I_j)_v) + S(e(I_i^s)_v, e(I_j)_v) + S(e(I_i)_v, e(I_j^s)_v)\right)$
Return $M = (s(x_i, x_j))_{1 \leq i,j \leq n}$

Algorithm 4. Similarity matrix.

The v vector is a binary vector which selects variables. If there is no knowledge of the importance of the variables, it is sufficient to consider that v is the vector of $\mathbb{R}^m$ whose coordinates are all 1. We will then propose a method for selecting the important variables using a genetic algorithm (see Section 2.4.3).

The choice of the similarity measure S depends, of course, on the problem in question. For example, one can consider the cosine similarity or metrics based on geometrical distances (Martinez et al., 2024) given by $S(x, y) = \frac{1}{1+\|x-y\|}$ where $\|.\|$ is a norm of $\mathbb{R}^m$. Here we will use the ℓ_2-norm $\|.\|_2$. In our case, the measure that gave the most significant results is the cosine similarity.

We are now ready to present our image ordering algorithm (Algorithm 5) as follows:

Parameters: a image encoder e, a set $\mathcal{A}$ of n images, a similarity measure $S : (\mathbb{R}^m)^2 \to \mathbb{R}^+$, a threshold α, $n_{\max} \in \mathbb{N}$ and a binary vector v
Compute the similarity matrix M by applying Algorithm 4 to e, A, S and v
If $n \leq n_{\max}$:
    Return a permutation given by Algorithm 3 applied to M and α
Else:
    Return a permutation given by Algorithm 1 applied to M

Algorithm 5. Ordering algorithm.

2.4.3 Feature selection using genetic algorithm

Feature selection utilizing a genetic algorithm is pivotal for refining our understanding and application of multivariable analysis in this study. Specifically, we delve into the intricate relationships between multiple predictor variables—features extracted from image embeddings—and the response variable, which is the accuracy of image sequencing, quantified as the well-ordering rate. Through this analysis, we aim to uncover the most impactful features that enhance the precision of image ordering and explore the dynamics of their interactions.

In our study, multivariable analysis critically examines the interplay between multiple predictor variables, specifically features extracted from image embeddings, and the response variable—image sequencing accuracy, measured by the well-ordering rate. This analysis is pivotal for identifying features that significantly enhance ordering accuracy and understanding their synergistic interactions.

In this endeavor, the genetic algorithm emerges as a cornerstone, facilitating the discernment of a subset of features within the high-dimensional embedding vectors that are paramount for efficacious image sorting. This entails a meticulous process of evaluating the performance imparted by various feature subsets on the image ordering task, followed by an iterative refinement of these subsets to optimize the well-ordering rate.

Feature selection is not merely about enhancing algorithmic accuracy; it's also about augmenting data clarity by mitigating noise and bolstering computational efficiency. Our goal is to pinpoint the specific coordinates within the embedding vector that hold significance for image classification. With a preliminary understanding that the binary vector v constituted all ones, the introduction of a genetic algorithm to refine the selection of v represents a strategic pivot. We focus on the embedding vectors generated by BLIP, specifically targeting the pivotal coordinates for image ordering. Given that these vectors comprise 32 sub-vectors, each with 256 coordinates, our selection process simplifies each sub-vector into a singular variable for feature selection.

Genetic algorithms, a well-entrenched optimization technique (Slowik and Kwasnicka, 2020; Sohail, 2023), are adeptly employed for variable selection. The aim is to isolate an optimal subset of variables that elevate the designated fitness function. Within the population of the genetic algorithm, each individual signifies a potential subset of variables, denoted by a binary vector that marks the inclusion (1) or exclusion (0) of each variable (each sub-vector). This systematic approach culminates in the identification of the optimal variable subset, significantly enhancing the efficiency and efficacy of image ordering in our dataset. The primary steps of the genetic algorithm are illustrated in Figure 2, with detailed explanations provided below:

(1) Initialize population: The GA initiates by creating a population of individuals, each possessing a binary string of 0s and 1s, indicating the inclusion (1) or exclusion (0) of variables in the subset.
(2) Fitness evaluation: Compute the fitness score for each individual in the population. In this specific application, the fitness function is defined as the well-ordering rate, where a higher fitness score signifies a more effective subset of variables.

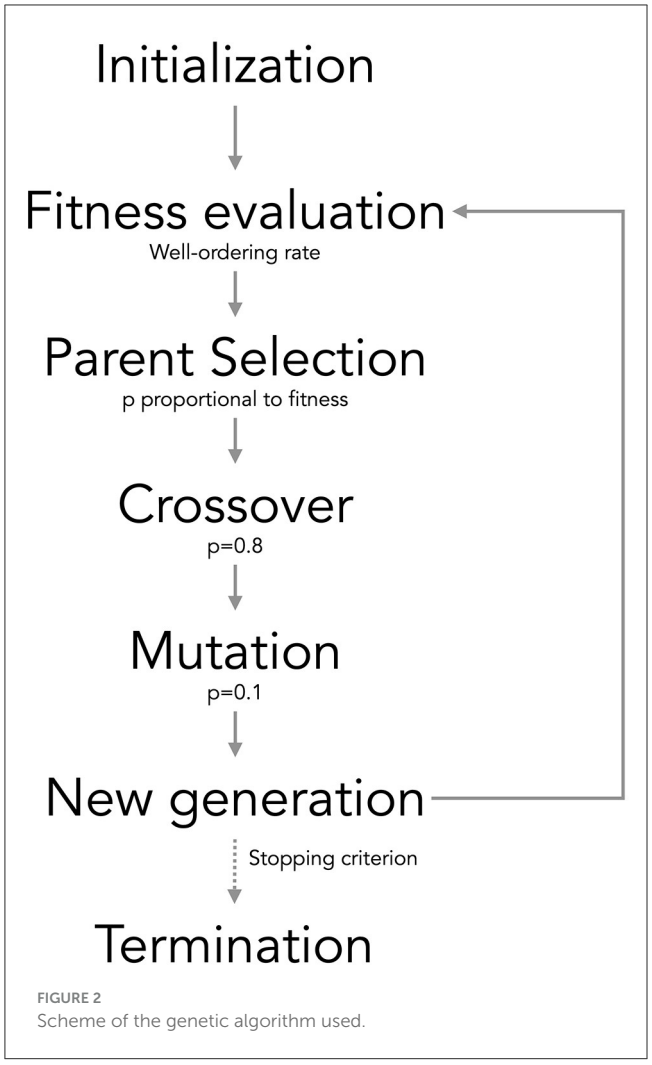

FIGURE 2
Scheme of the genetic algorithm used.

(3) Parent selection: Individuals are chosen for reproduction based on their fitness scores. The probability of selecting each individual as a parent is proportional to its fitness score such that $p_{\text{selection}} = \frac{f_i}{\sum_i f_i}$.
(4) Crossover: Crossover is performed with a probability ($p_{\text{crossover}} = 0.8$). Two parents are randomly selected from the parent pool, and a crossover point is chosen. The offspring are generated by swapping the binary strings of the parents at the crossover point, creating two new individuals.
(5) Mutation: Mutation occurs with a specified probability ($p_{\text{mutation}} = 0.1$) for each bit of an individual. If mutation occurs, the bit is flipped (0 becomes 1, and vice versa), introducing exploration and aiding the algorithm in escaping local optima.
(6) Next generation: The new population is formed by repeating the crossover and mutation process for the entire population.
(7) Termination: The GA runs for a predefined number of generations. Upon completion of the specified number of generations, the algorithm returns the best individual found during the process, representing the optimal subset of variables.

As aforementioned, the fitness function is given by the well-ordering rate, that is detailed in Algorithm 6:

Parameters: an individual v (which is a binary vector), a set $\mathcal{C} = \{A_i\}_{1\leq i\leq k}$ where A_i is a set of images, $\{\mathcal{P}_i\}_{1\leq i\leq k}$ a set where $\mathcal{P}_i$ is a partition of A_i, a similarity measure $S : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}^+$, a threshold α and $n_{\max} \in \mathbb{N}$
Define $t = 0$:
For $i \in \{1, ..., k\}$:
    Pick a permutation σ_i given by Algorithm 5 applied to b, A_i, S, α, $n_{\max} \in \mathbb{N}$ and v
    If σ_i preserves $\mathcal{P}_i$:
        $t \longleftarrow t + 1$
Return t/k

Algorithm 6. Fitness function.

3 Results

In this section, we present the results of Algorithm 5 applied to our problem. To do so, we will evaluate the results based on the numbers of adds that are well-ordered. This means, with an accuracy metric for all the ads called the well-ordering rate:

Definition 3.1. Let $\mathcal{C} = \{\mathcal{A}_i\}_{1\leq i\leq k}$ our set of sets of images and for each $i \in \{1, ..., k\}$, let $\mathcal{P}_i$ be a partition of $\mathcal{A}_i$ and $\sigma_i \in S_{n_i}$ (where $n_i = |A_i|$). We define **well-ordering rate** by

$$\tau = \frac{1}{k} \left| \{i \in \{1, ..., k\} : \sigma_i \text{ preserves } \mathcal{P}_i\} \right|.$$

Here $\mathcal{A}_i$ is a set of images, $\mathcal{P}_i$ is the partition of $\mathcal{A}_i$ whose sets contains the photos of the same bedroom and σ_i is a permutation obtained thanks to Algorithm 5. Results are shown in Table 1.

It is worth recalling that the value of $n_{\max}$ determines the threshold at which we employ an approximate solving method to determine an optimal solution. The more you increase $n_{\max}$, the better the results will be. However, for computational reasons, it is wise to choose a value between 10 and 12.

As mentioned in Section 2.4.2, image encoders are sensitive to axial symmetries. We can indeed confirm this here. Results from replacing the similarity measure of Algorithm 4 by $s(i, j) = S(e(I_i)_v, e(I_j)_v)$ are shown in Table 2.

One thing that is interesting to note is that, unlike CLIP, BLIP seems to have little sensitivity to image orientation, since the results are very similar (with τ varying by only 0.4% for BLIP, compared to 3.3% for CLIP.)

Previous results were obtained without performing any feature selection. We then upgraded the model by applying a genetic algorithm to perform feature selection (see Section 2.4.3). Note that since the results are significantly better with BLIP than with CLIP, we confine ourselves here to improving on the results of the algorithm using BLIP. We used cosine similarity for the fitness similarity measure S. Considering a population of 50 and 10 generations, we obtained a vector v_{GA} that we can use in

TABLE 1 Results with symmetrization.

Encoder e	$n_{\max}$	α	$S(x,y)$	τ
c	10	0.9	$\frac{\langle x,y\rangle}{\|x\|_2\|y\|_2}$	69.7
c	10	0.9	$\frac{1}{1+\|x-y\|_2}$	66.4
c	0	–	$\frac{\langle x,y\rangle}{\|x\|_2\|y\|_2}$	65.4
c	10	0.9	$\frac{1}{1+\|x-y\|_2}$	66.4
b	10	0.9	$\frac{\langle x,y\rangle}{\|x\|_2\|y\|_2}$	81

TABLE 2 Results without symmetrization.

Encoder e	$n_{\max}$	α	$S(x,y)$	τ
c	10	0.9	$\frac{\langle x,y\rangle}{\|x\|_2\|y\|_2}$	66.4
b	10	0.9	$\frac{1}{1+\|x-y\|_2}$	80.6

TABLE 3 Results with GA.

Encoder e	$n_{\max}$	α	$S(x,y)$	τ
b	9	0.9	$\frac{\langle x,y\rangle}{\|x\|_2\|y\|_2}$	85.8

Algorithm 5. For computational reasons, we lower $n_{\max}$ to 9 and we obtained that well-ordering rate has therefore increased by 4.8%, as shown in Table 3.

4 Discussion

In the discussion of our findings, it's important to contextualize the implications of our approach for the broader field of image processing and its applications in online commerce. Our method, while specifically tailored to address the challenge of sorting property listing images, holds potential for a wide range of applications where image order and selection are critical. By grounding our solution in a general mathematical framework and leveraging semantic image embedding, we offer a path forward for similar challenges across various domains.

The employment of a genetic algorithm for semantic variable selection underscores the adaptability of our approach. This adaptability is crucial, as it suggests that with appropriate tuning, our method can be extended to other types of datasets or sorting criteria. For instance, in e-commerce, where the visual presentation of products can significantly influence consumer behavior, our algorithm could be used to optimize the arrangement of product images to highlight features most likely to appeal to customers.

Irrespective of the specific field of application, the predominant methodology for image sorting has traditionally leveraged clustering techniques, which range from classical machine learning approaches like k-means or k-nearest neighbors to more advanced neural network frameworks (Rorissa and Hastings, 2004; Liang et al., 2023). While such methods offer rapid processing and effectively categorize images into distinct subsets, they fundamentally execute a classification-centric sorting strategy. This approach does not account for the ordering within these subsets, nor does it address the aspect of sequentiality that is critical in the context of the current study and others focused on e-commerce applications.

Previous research has explored the automation of product image sequencing in the e-commerce domain (Chaudhuri et al., 2018; Fan et al., 2022). However, the realm of real estate listings presents a notably more intricate challenge. This complexity stems from the diversity and uncertainty associated with subproducts, such as the various rooms and types of rooms, each with their unique 3D shapes and elements. This scenario is markedly different from simpler products like smartphones or clothing, where the variability and spatial considerations are significantly less pronounced. Consequently, the methodologies and solutions developed for simpler product categories fall short of addressing the intricate demands and nuances of the real estate sector.

One of the notable challenges in our study is associated with the methodology grounded in similarity distance metrics. This approach inherently positions images with high similarity—including identical ones—next to each other in the sequence, inadvertently overlooking the potential issue of redundancy. Our research operates under the presumption that each image contributes a meaningful and complementary information when compared to others. However, this assumption might not always hold true, particularly in scenarios where photographs are captured and selected for marketing purposes, with the aim of automating the process of compiling images for a commercial advertisement. An example of redundancy within the realm of real estate marketing might involve multiple shots of the same room captured from slightly varied angles, which, in essence, fail to furnish additional valuable information to the prospective buyer.

In practical applications, especially in digital marketing and online advertising, the presence of redundant images could dilute the effectiveness of the visual narrative, potentially overwhelming or disengaging the target audience. Recognizing and eliminating redundancy not only streamlines the content but also enhances the user experience by presenting a concise and focused array of images that collectively convey a more powerful message. Image sequence generation for products in e-commerce has been already addressed taking into account redundant images (Chaudhuri et al., 2018; Fan et al., 2022). However, the methodologies applied are far from the approach required to sorting with subsets restrictions and could not be integrated in the present study.

Therefore, addressing the limitation of redundancy is paramount for advancing the utility and efficiency of automated image sorting systems. Future research directions should include the development of algorithms capable of discerning and filtering out redundant images. This involves integrating more sophisticated measures that go beyond mere visual similarity, incorporating aspects such as semantic content, emotional appeal, and narrative coherence to ensure that each selected image adds distinct value to the collection. Following on the previous example of redundant images, another approach could involve leveraging image stitching algorithms to pinpoint and identify redundant photographs (Wu and Cai, 2012). Such advancements could significantly refine the process of automating commercial advertisements, offering more dynamic, engaging, and effective visual presentations.

By tackling this challenge, subsequent studies can pave the way for more intelligent automation solutions that are not only adept at organizing images but also at curating content that

resonates more deeply with viewers. This progression is crucial for leveraging the full potential of automated systems in enhancing the strategic presentation of images in commercial contexts, ultimately driving greater engagement and conversion in digital marketing campaigns.

Furthermore, our study opens up several additional avenues for future research. One immediate area of interest is the exploration of different semantic features and their impact on sorting efficacy. As we have seen, certain features contribute more significantly to performance than others. A deeper understanding of these dynamics could lead to more refined algorithms that are both more efficient and effective.

Another potential direction is the investigation of alternative algorithms or enhancements to the genetic algorithm used in our study. While our approach has demonstrated promising results, there is always room for improvement in computational efficiency or sorting accuracy. Exploring machine learning techniques that can learn optimal sorting strategies from data, rather than relying on predefined semantic features, might offer new insights and improvements.

Lastly, the practical implications of our research for real-world applications should not be overlooked. Integrating our sorting solution into online platforms could significantly enhance user experience by providing more intuitive and visually coherent image arrangements. However, achieving this integration poses its own set of challenges, including scalability, platform compatibility, and user interface design considerations.

In conclusion, while our study specifically addresses the challenge of sorting images in property listings, the implications of our work extend far beyond this context. The principles and methodologies we have developed offer valuable insights and tools for a wide range of applications in online commerce and beyond. Future research, building on our findings, has the potential to further refine and expand the applications of these techniques, contributing to the advancement of the field of image processing and its practical applications.

Data availability statement

The data analyzed in this study is subject to the following licenses/restrictions: portion of the data available upon reasonable requests sent to the authors. Requests to access these datasets should be directed to jj@udc.es.

Author contributions

GG: Formal analysis, Methodology, Writing – original draft. MC: Methodology, Supervision, Visualization, Writing – review & editing. AT-P: Data curation, Writing – review & editing. JR: Conceptualization, Funding acquisition, Supervision, Writing – review & editing, Investigation.

Funding

The author(s) declare that financial support was received for the research, authorship, and/or publication of this article. This study has been funded by the Consolidation Aid of the Xunta ED431C 2022/46. Also, this work received funding with reference PID2020-118362RB-I00, from the State Program of R+D+i Oriented to the Challenges of the Society of the Spanish Ministry of Science, Innovation, and Universities. Finally, PhotoILike received support from the Spanish Ministry for Science and Technology, through the Center for The Industrial Technological Development (CDTI), with the 2023 NEOTEC Grant (SNEO-20222114).

Conflict of interest

GG, AT-P, and JR were employed by PhotoILike.

The remaining author declares that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

References

Cacioppo, J. T., Petty, R. E., Feinstein, J. A., and Jarvis, W. B. G. (1996). Dispositional differences in cognitive motivation: The life and times of individuals varying in need for cognition. *Psychol. Bull.* 119:197. doi: 10.1037/0033-2909.119.2.197

Chaudhuri, A., Messina, P., Kokkula, S., Subramanian, A., Krishnan, A., Gandhi, S., et al. (2018). "A smart system for selection of optimal product images in e-commerce," in *2018 IEEE International Conference on Big Data* (Big Data), 1728–1736.

Chen, M.-Y., and Teng, C.-I. (2013). A comprehensive model of the effects of online store image on purchase intention in an e-commerce environment. *Electron. Commer. Res.* 13, 1–23. doi: 10.1007/s10660-013-9104-5

David Koch, S. T., and Mayr, R. (2021). Order versus disorder — the impact on value estimates of durable consumption goods. *Appl. Econ. Lett.* 28, 1635–1640. doi: 10.1080/13504851.2020.1841081

Di, W., Sundaresan, N., Piramuthu, R., and Bhardwaj, A. (2014). "Is a picture really worth a thousand words? - on the role of images in e-commerce," in *Proceedings of the 7th ACM International Conference on Web Search and Data Mining, WSDM '14* (New York, NY: Association for Computing Machinery), 633–642.

Fan, X., Zhang, C., Yang, Y., Shang, Y., Zhang, X., He, Z., et al. (2022). "Automatic generation of product-image sequence in e-commerce," in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD '22* (New York, NY: Association for Computing Machinery), 2851–2859.

Franco-Watkins, A. M., Pashler, H., and Rickard, T. C. (2006). *Does Working Memory Load Lead to Greater impulsiVity? Commentary on Hinson, Jameson, and Whitney*. Washington, DC: American Psychological Association. (2003).

Garbarino, E. C., and Edell, J. A. (1997). Cognitive effort, affect, and choice. *J. Cons. Res.* 24, 147–158. doi: 10.1086/209500

Gutiérrez, L., and Keith, B. (2019). "A systematic literature review on word embeddings," in *Trends and Applications in Software Engineering*, eds J. Mejia, M. Muñoz, Á. Rocha, A. Peña, M. and Pérez-Cisneros (Cham: Springer International Publishing), 132–141.

Jin, C., and Gallimore, P. (2010). The effects of information presentation on real estate market perceptions. *J. Prop. Res.* 27, 239–246. doi: 10.1080/09599916.2010.518404

Li, J., Li, D., Xiong, C., and Hoi, S. (2022). *Blip: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation*.

Liang, J., Cui, Y., Wang, Q., Geng, T., Wang, W., and Liu, D. (2023). "Clusterfomer: clustering as a universal visual learner," in *Advances in Neural Information Processing Systems, Vol. 36*, eds A. Oh, T. Neumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (Curran Associates, Inc.), 64029–64042.

Martinez, L., Montes, M., Zezzatti, A. O., Ponce, J., and Guzmán, E. (2024). "Searcher for clothes on the web using convolutional neural networks and dissimilarity rules for color classification using euclidean distance to color centers in the hsl color space," in *Advances in Computational Intelligence. MICAI 2023 International Workshops*, eds H. Calvo, L. Martínez-Villaseñor, H. Ponce, R. Zatarain Cabada, M. Montes Rivera, and E. Mezura-Montes (Cham: Springer Nature Switzerland), 159–169.

Opatrny, J. (1979). Total ordering problem. *SIAM J. Comp.* 8, 111–114. doi: 10.1137/0208008

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., et al. (2021). *Learning Transferable Visual Models From Natural Language Supervision*.

Rorissa, A., and Hastings, S. K. (2004). Free sorting of images: attributes used for categorization. *Proc. Am. Soc. Inf. Sci. Technol.* 41, 360–366. doi: 10.1002/meet.1450410142

Schwalbe, G. (2022). *Concept Embedding Analysis: A reView*.

Siebert, R., and Seiler, M. (2022). Why do buyers pay different prices for comparable products? A structural approach on the housing market. *J. Real Est. Finan. Econ.* 65, 1–32. doi: 10.1007/s11146-021-09841-5

Slowik, A., and Kwasnicka, H. (2020). Evolutionary algorithms and their applications to engineering problems. *Neural Comp. Appl.* 32, 12363–12379. doi: 10.1007/s00521-020-04832-8

Sohail, A. (2023). Genetic algorithms in the fields of artificial intelligence and data sciences. *Ann. Data Sci.* 10, 1007–1018. doi: 10.1007/s40745-021-00354-9

Thaler, S., and Koch, D. (2023). Real estate pictures: the role of furniture preferences in subjective valuation. *J. Housing Res.* 32, 180–203. doi: 10.1080/10527001.2023.2168585

Wu, X., and Cai, M. (2012). An auto-sorting algorithm for image sequences with different sizes in image stitching. *Intell. Automat. Soft Comp.* 18, 861–872. doi: 10.1080/10798587.2012.10643294

Zakrewsky, S., Aryafar, K., and Shokoufandeh, A. (2016). *Item popularity prediction in e-commerce using image quality feature* vectors.