Amazon Shop the Look: A Visual Search System for Fashion and Home

Ming Du
mingdu@amazon.com
Amazon Visual Search & AR
Palo Alto, USA

Arnau Ramisa
aramisay@amazon.com
Amazon Visual Search & AR
Palo Alto, USA

Amit Kumar K C
amitkrkc@amazon.com
Amazon Visual Search & AR
Palo Alto, USA

Sampath Chanda
csampat@amazon.com
Amazon Visual Search & AR
Palo Alto, USA

Mengjiao Wang*
mj.wang.lang@gmail.com
Amazon Visual Search & AR
Palo Alto, USA

Neelakandan Rajesh
rneel@amazon.com
Amazon Visual Search & AR
Palo Alto, USA

Shasha Li
shashli@amazon.com
Amazon Visual Search & AR
Palo Alto, USA

Yingchuan Hu
huyc@amazon.com
Amazon Visual Search & AR
Palo Alto, USA

Tao Zhou
taozho@amazon.com
Amazon Visual Search & AR
Palo Alto, USA

Nagashri Lakshminarayana
nagasl@amazon.com
Amazon Visual Search & AR
Palo Alto, USA

Son Tran
sontran@amazon.com
Amazon Visual Search & AR
Palo Alto, USA

Doug Gray
douggray@amazon.com
Amazon Visual Search & AR
Palo Alto, USA

ABSTRACT

In this paper, we introduce Shop the Look, a web-scale fashion and home product visual search system deployed at Amazon. Building such a system poses great challenges to both science and engineering practices. We leverage large-scale image data from the Amazon product catalog and adopt effective strategies to reduce the human effort required to annotate data. By employing state-of-the-art computer vision techniques, we train detection, recognition, and feature extraction models to bridge the domain gap between in-the-wild query images and product images which are taken under controlled settings. Our system is designed to achieve a balance between result accuracy and efficiency. The run-time service is optimized to provide retrieval results to users with low-latency. The scalable offline index-building pipeline adapts to the dynamic Amazon catalog that contains billions of products. We present both quantitative and qualitative evaluation results to demonstrate the performance of our system. We believe that the fast-growing Shop the Look service is shaping the way that customers shop on Amazon.

CCS CONCEPTS

• **Information systems** → **Information retrieval**; • **Applied computing** → **Electronic commerce**.

KEYWORDS

Visual Search, Online Shopping, Image Retrieval, Deep Learning

KDD '22, August 14–18, 2022, Washington, DC, USA

ACM ISBN 978-1-4503-9385-0/22/08.
https://doi.org/10.1145/3534678.3539071

ACM Reference Format:
Ming Du, Arnau Ramisa, Amit Kumar K C, Sampath Chanda, Mengjiao Wang*, Neelakandan Rajesh, Shasha Li, Yingchuan Hu, Tao Zhou, Nagashri Lakshminarayana, Son Tran, and Doug Gray. 2022. Amazon Shop the Look: A Visual Search System for Fashion and Home. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22), August 14–18, 2022, Washington, DC, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3534678.3539071

1 INTRODUCTION

E-commerce platforms such as Amazon offer vast catalogs with a massive number of products, which become challenging to navigate for customers. Traditionally, text-based search engines have been widely used to surface products that match the customer's shopping intent. However, for products that have an important visual component, difficult to describe with words, visual search (or Content-Based Image Retrieval), provides a convenient alternative which often returns more relevant results. In 2019, we launched the StyleSnap service for fashion products, and in 2020 we expanded the service to support home product search. We renamed it to Shop the Look in 2022. These commerce verticals are selected because of their image-search-friendly nature and persistent customer interest: they are among the most searched verticals, and the largest product categories. In 2022, the estimated Total Addressable Market (TAM) in the US alone is greater than $200B for fashion e-commerce[1], and around $150B for furniture[2]. Therefore, we choose to optimize the system architecture design and model training for fashion and home

* Mengjiao Wang contributed to the paper when she worked at Amazon. She is currently affiliated with Meta Platform Inc.

[1] https://www.statista.com/statistics/278890/us-apparel-and-accessories-retail-e-commerce-revenue/

[2] https://www.statista.com/statistics/257524/us-furniture-and-home-furnishings-e-commerce-revenue/

instead of building a generic visual search service. The Shop the Look service focuses on the use case often referred to as "street-to-shop" or "home-to-shop", in which customers can upload any kind of picture: a product image from Amazon or another e-commerce website, screenshots from a lifestyle blog or a social media post, or even their own photos, and purchase the products in the image. Customers can access Shop the Look on Amazon mobile app (see Figure 1), or by visiting https://www.amazon.com/shopthelook on desktop.

Developing and deploying a web-scale visual search system like Shop the Look is technically challenging. Among the many factors that need to be taken into account, we focus on the following three:

Domain Gap: Typically, photos uploaded by customers differ from Amazon product images in many aspects: background, lighting condition, resolution, etc. Directly matching images from the two domains usually produces poor results. Therefore, how we bridge the domain gap fundamentally determines the quality of search results.

Scalability: The Amazon fashion and home product catalog contains billions of items that are organized in a hierarchy which cannot be described with a simple tree structure. Multiple parallel category divisions (*e.g.* color, pattern, size variants) and product misclassification by merchants make building a search index with such data source a challenge in and of itself. To return search results with low latency at run-time, the feature index should also enable efficient nearest-neighbor search. In addition, due to the nature of fashion and home product industries, both Amazon and third party sellers attempt to track the trends and adapt the products offered to seasonal changes. As a result, our index building pipeline needs to efficiently handle a highly dynamic catalog.

User Experience: In visual search applications, result quality as measured by visual similarity does not always align with the intent of the customers. Satisfaction with search results also depends on other factors such as product popularity, availability, or shipping options. Optimizing the complete customer shopping experience is therefore a non-trivial task.

In this paper, we share the experiences we had with these challenges while building Shop the Look. To handle the domain gap, we employ deep learning techniques to build a computer vision solution with three main components: product localizer, fine-grained classifier, and feature extractor. The product localizer helps identify candidates for the customer search intention, and it also helps the downstream tasks avoid interference from the background in images. The purpose of fine-grained category recognition is to narrow down the search and reduce defective results. Finally, the feature extractor computes a representation for the candidate objects that allows us to search for the most similar products in the index, even when there is a significant domain variation. Since learning models for these three modules require a large amount of training data, various augmentation or synthesis strategies can be applied to supplement the high quality yet expensive human annotations.

We also present how to design the index building pipeline and run-time system for scalability. Computationally expensive tasks, such as localization and feature extraction for product images, are distributed over cloud infrastructure as batch computing jobs. Changes in product catalog can be tracked by regularly calculating incremental inputs on elastic Map-Reduce clusters, which in turn triggers differential builds. We use approximate nearest neighbor algorithms to achieve the desired balance between latency and accuracy for run-time retrieval.

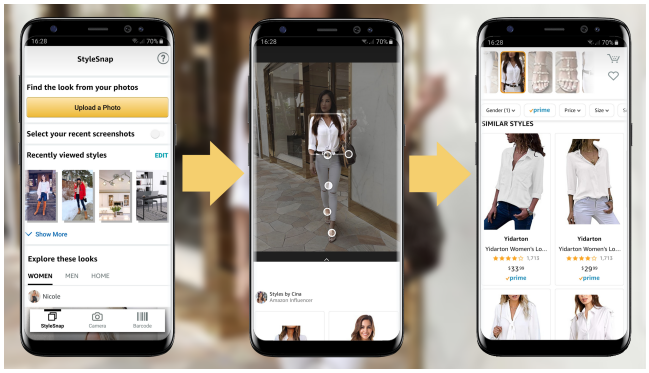

Figure 1: With Shop the Look, customers can quickly find fashion and home products that are similar to their uploaded query image on Amazon. The experience is available for both mobile users via Amazon mobile app and desktop users at https://www.amazon.com/shopthelook.

Finally, we describe a customer-preference-based re-ranking model which takes additional non-visual factors into consideration. Duplicate products in results is a common problem that may degrade user experience. This can be mitigated by removing duplicates both at index building stage and at run-time, with specialized methods.

Shop the Look has shown great impact on customers and has been experiencing a rapid growth since launching. On TikTok, the total number of views for videos that people upload to showcase their Shop the Look experience has exceeded 10 billion [3]. In 2021, the number of monthly active user (MAU) of Shop the Look has increased by 250%.

The remainder of the paper is organized as follows. In Section 2, we review existing relevant visual search architectures, as well as other relevant work. We present the system architecture and describe in detail the components in Section 3. Section 4 presents the experimental results. Finally, Section 5 concludes our paper with paths for further improvement.

2 RELATED WORK

Visual search systems: The outstanding results of deep learning in recent years have established visual search as a standard system in online retail, with many companies [2, 10, 25, 29–31] offering it to navigate their catalogs. Most visual search systems start by localizing the relevant items in the query image. Afterwards, they compute a feature representation of each extracted region of interest, and then use it to search for similar items in their catalogs. These systems have slightly different use cases (*e.g.*, Bing [10] must handle generic visual search queries in addition to products, eBay [29] is a marketplace where anyone can sell), therefore they differ in the specific techniques used, such as how relevant items

[3] https://www.tiktok.com/tag/StyleSnap. The hashtag was associated with the original product name StyleSnap

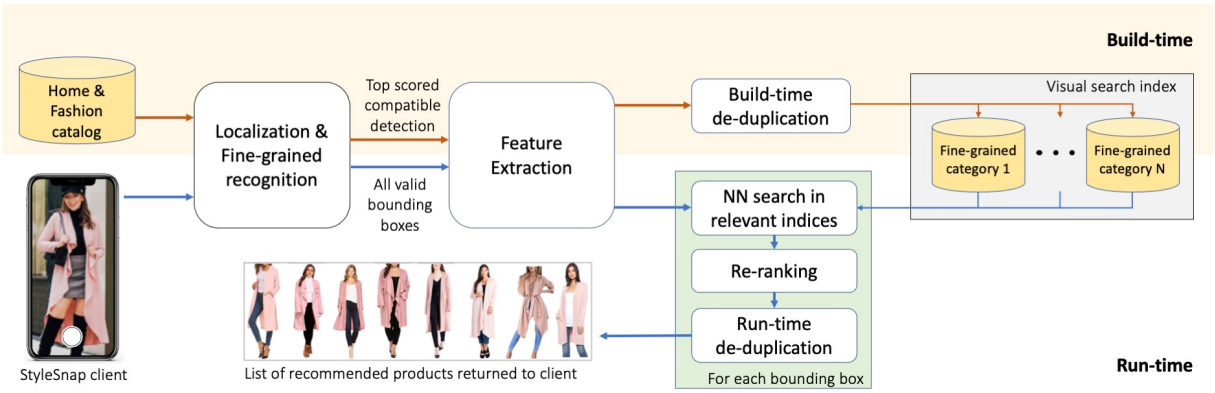

Figure 2: Overview of the Shop the Look architecture. The arrows in red represent data flow at build-time, and those in blue represent data flow at run-time.

are localized, how feature vectors are computed, or how search is performed.

Considering the large scale of the image indexes, strategies to reduce the search effort are often used. For example feature binarization and Hamming distance are used in [29, 31] to quickly discard a large volume of images that are irrelevant for a query. We instead reduce the search space by splitting the index according to the fine-grained category, and use approximate nearest neighbor search with compact feature vectors.

Another strategy to reduce the footprint of a visual search system is to use multi-task learning to concentrate functionality into a single model. For example, in [2, 31] authors train a single (or joint) network for different tasks or business verticals. However, we decided to train a separate model for each task. Doing so, we are able not only to quickly react to customer search patterns and deploy models to expand to new categories or whole new verticals, but also to fine-tune a specific model without affecting the others.

Localization: This is usually the first step in visual search pipelines, although different use cases may require different approaches. For instance [31] only cares about a single prominent product per image (assumed to be the query intent), and can thus leverage click data to train a semi-supervised model. In other systems [10, 12] off-the-shelf detector models are adapted, but only on queries from certain verticals like fashion. There are also exceptions where localization is not used at all, like at eBay [29] that only allows intra-catalog searches.

The two dominant (and fundamentally distinct) paradigms for object detection using deep networks are two-stage [22] and one-stage detectors [3, 15, 21]. The former delegates generating region proposals to a specialized sub-network, while the latter combines the region proposal and classification into a single branch. More recently strategies leveraging transformers have been proposed [5].

Image representation: Probably the most critical piece of a visual search system is how to represent the images, as it directly impacts the quality of the returned results and the latency of the system. Currently, the most popular technique for this purpose are deep metric learning networks trained with losses such as the contrastive loss [4] or the triplet loss [9], as they can bridge the gap between different image distributions using pairs or triplets of related pictures. Several improvements, like smart negative sampling or new losses have been proposed in recent years [7, 13, 18, 27, 28]. Of particular interest for our case are the developments in the street/home-to-shop domain [1, 16, 24]. In [31] a metric learning loss is added to the localizer to have a single multitask model, while [10] uses an array of different models with categorization and deep metric learning losses organized in a three-tier system.

3 ARCHITECTURE

The system architecture of Shop the Look is composed of several deep neural networks, as well as various support modules. Figure 2 shows an overview of the complete system.

The system takes an input image and uses an array of detectors and fine-grained classifiers to detect and identify the relevant objects. Then, each object is described with an embedding network that takes as input the image information around the detected object area. This computed object representation is then used to search in a catalog index with fashion and home products.

The visual search process in the figure is divided into two flows, *build-time* and *run-time*, which share components such as localization, fine-grained classification, and feature extraction. The *build-time* process is run offline and refers to the building of the visual search product catalog index. We detail the optimizations done to handle large volumes of data during the build process in Section 3.6. This process is applied to all the supported products in the fashion and home categories of the Amazon catalog, yet only products with a clear image are added to the visual search index. Once the index is built, the online *run-time* process returns the most similar products when a customer uploads a query image. To improve the customer experience, the results are further re-ranked and de-duped.

In the remainder of the section, we detail each of our components, namely localizer, fine-grained classifier, embedding model, re-ranking and de-duping, and indexing.

3.1 Localizer

Other visual search services can expect one single relevant object to be featured in the center of the query image uploaded by the

customer, which can simplify the localization component of their architecture. However, in highly aesthetic verticals like fashion and home, several products are often combined together in a single composition, such as a full outfit or a professionally decorated room. For these types of images, customer intent is not clear and multiple relevant objects appearing in the picture must be detected. For these cases, visual search systems usually employ single-stage object detectors, as they offer a good balance between precision and recall at a low latency. After evaluating several well-known object detector methods, we selected the lightweight YOLOv3 [21], which fits our latency and performance requirements.

Our service is open-ended in the sense that users can input any picture, and we should still return the best list of products for each instance of relevant object, while simultaneously not reacting to images where there are none. To this end, detector networks are trained using datasets that contain both catalog and in-the-wild images, annotated with bounding boxes and category labels. To reduce false positives, we added un-annotated out-of-distribution images to each batch of training. In our evaluations, we found that with a 3:1 ratio we could reduce false positives from 77.7% to 4.8% without impacting recall. Finally, averaging weights [11] with the last training checkpoints leads to better performance.

The properties of the objects differ significantly between Fashion and Home. Furniture is rigid in most cases, while apparel items are deformable, and can be presented in multiple ways (e.g. hanging in a wardrobe, folded on top of a bed, or being worn by a person). On the other hand, images containing furniture items tend to have high variance in viewpoint, while apparel items in fashion images are usually taken from the front.

As a consequence, it is not straightforward for the detector network to re-use features between the two domains, and we have observed that the accuracy may decrease as the number of categories increases, especially across the two different domains. In our experiments, we have found that two separate detectors in the architecture produce better results, although we continue to explore more efficient joint models in future work.

To increase accuracy, it can be useful to group apparel items according to which person is wearing them, by e.g. adding person types (man, woman) as additional localization categories. Furthermore, being able to predict apparel person type (such as women's, men's, etc) for each detected apparel item, can help correcting misclassifications, and allow the customer to easily filter the results.

3.2 Fine-grained category recognition

To improve retrieval performance, we train classifiers to determine the fine-grained category of an apparel or furniture item. For a given query item, the search would traverse the index of the product listings corresponding to the fine-grained category at the query item, rather than going through all the products in the catalog. This reduces the index size and speeds up image retrieval performance, which is crucial for the end-to-end system. Additionally, by decoupling localization and fine-grained classification, it allows for easier extensibility in the fine-grained categories. For example if we want to distinguish between indoor and outdoor chairs, only the specific chair classifier would require re-training. For this, once we localize an item of interest (apparel or furniture) and determine its top-level category (e.g., tops, bottoms, bed, chair, etc.), we extract an image patch around the detected region and feed it to a classifier to infer its fine-grained category (e.g., tops to jackets).

As the end-to-end system has to respond promptly to user queries, the fine-grained category recognition stage should introduce minimal overhead. We use ResNet-18 [8] as it would be sufficiently lightweight for this purpose. Class imbalance causes the cross-entropy loss to struggle with minority classes. Many approaches have been proposed in the literature to remedy the class imbalance problem, one of which is the focal loss [14].

Let us denote the i-th image by $\boldsymbol{x}_i$, the corresponding one-hot encoded label vector by $\boldsymbol{g}_i$, and the prediction vector by $\boldsymbol{p}_i$. If the focal loss is applied to mitigate the class imbalance problem, it can be minimized at each mini-batch of size b in the following form:

$$L_{cls} := -\frac{1}{b}\sum_i \boldsymbol{g}_i^\top [(1-\boldsymbol{p}_i)^\gamma \log(\boldsymbol{p}_i)], \tag{1}$$

where *focusing* parameter $\gamma = 2$ can be set as suggested in [14].

During inference, we first perform fine-grained classification on the query image and then pick the best matching index to search in. The matching index selection strategy first sorts the predicted score vector and then picks the categories until the accumulated score exceeds a predetermined threshold.

3.3 Embedding model

In visual search systems, image relevance is measured based on the distance between feature vectors. We train a deep neural network to generate embedding vectors for image inputs. The challenges for such a deep metric learning task are two-fold: (1) how to teach the model to learn the concept of visual similarity; and (2) how to make the model insensitive to the image domain variations. The concept of visual similarity is inherently subjective. For example, people may argue whether two dresses with the same color but different neckline design should be labeled as similar or dissimilar. To mitigate this issue, we compile a dataset which contains a large number of Amazon fashion and home products and has multiple images per product to train our image embedding model. In other words, similarity can be treated as a generalized concept of identity. Because the multiple images of the same product in this database cover a variety of image domains, such as studio-style (as seen on Amazon product page), social-media style, or phone-camera grade, it facilitates the training of a domain adaptive model. The raw dataset usually contains substantial noise, like images of accessories relevant to the product, size tables, or images of multiple pieces. A localization model is applied to the training images and those that don't have detections consistent with the meta-data are discarded.

The embedding model uses a bottleneck layer which is attached to the last pooling layer of the backbone network to extract feature vectors. In our experiments, we found that a ResNet-50 backbone usually achieves better balance between embedding performance and computational cost. The design decision of not sharing backbone with the fine-grained classifier was a result of considering the practical demand for the flexibility to be able to update each model separately. Our experiments also showed that the best results are obtained by margin-based contrastive loss [28], defined as:

$$l_{ij} = \max(0, \alpha + y_{ij}(D_{ij} - \beta)), \tag{2}$$

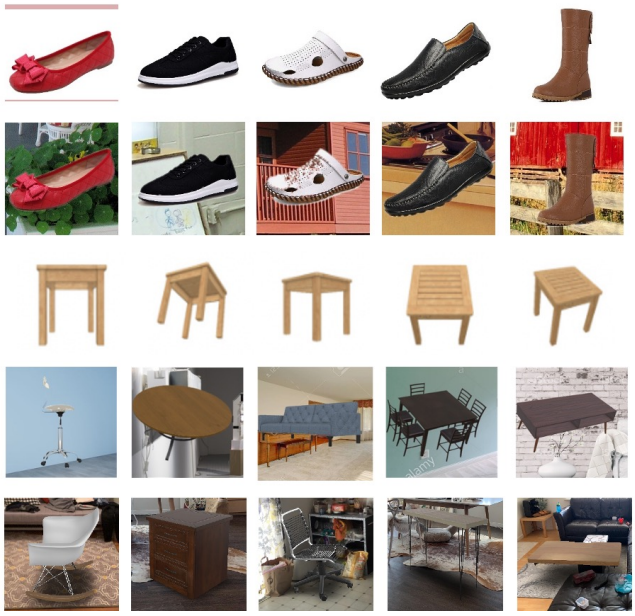

Figure 3: Samples from the augmented training dataset. From the top to the bottom: (1) Fashion product catalog images; (2) Synthesized in-the-wild fashion product images by superimposing on random backgrounds; (3) Rendered 3D Amazon home product dataset; (4) Synthesized Home images by superimposing 3D model on random scene images; (5) AutoPlacement synthesized home dataset. The first two are for fine-grained classifier training, and the remaining three are for embedding model training.

where y_{ij} is -1 for negative training pairs and $+1$ for positive ones, D_{ij} is the distance between the pair, α is a margin parameter and β is boundary between positive and negative pairs that can be tuned. Here, we define any pair of images from the same product as positive, and those from a different product as negative. Usually, the number of all the negative pairs is formidably large. This problem can be mitigated by performing distance-weighted sampling [28]:

$$Pr(n^* = n|a) \propto \min(\lambda, q^{-1}(D_{an})), \tag{3}$$

where $Pr(n^* = n|a)$ is the probability that a negative pair (n, a) is chosen given an anchor image a, λ is a small clipping constant number to avoid noisy samples, $q(d)$ is the probability density function of distances for samples uniformly distributed on $\mathbb{S}^{N-1}$, and D_{an} is the distance between image a and n, calculated using the model with the most recently updated parameters during training.

3.4 Data Augmentation and Synthesis

When training our models, in addition to the basis datasets that are mentioned in previous sections, we further employ various image synthesis approaches to augment the training data. Image synthesis provides an efficient way to generate a large amount of data because it is usually automatic and requires little human intervention. By intentionally introducing variations in certain dimensions into the data, we can guide the models to learn the corresponding invariance. In [6], a "cut-paste" style synthesis is used to generate training data for a detector. Although there is no special practice to guarantee the physical correctness of the generated images, their results suggest that patch-level realism is sufficient to improve model performance. We arrive at similar conclusions with our experiments, but also observe that improving the fidelity of the synthesized images can further boost model performance.

Fine-Grained Classifier: The purpose of using synthesized images for fine-grained classifier training is to achieve domain adaptivity. We have an abundance of catalog images compared to in-the-wild images. To address the inherent domain imbalance between these image sets, we experimented with a method similar to "cut-paste" to obtain more synthesized in-the-wild images. We first segment the foreground object from the catalog images by using thresholding and morphological operations, and then blend them with random background images. In the first two rows of Figure 3, we show some examples of such augmentation.

Embedding Model: Due to the 3D nature of home products, visual search for home queries is more prone to errors caused by viewpoint variations. In light of this, we consider using extra augmented and synthesized data for home to improve the viewpoint invariance of the embedding model. The source data for the augmented dataset consists of 3D Amazon home product models rendered at multiple canonical viewpoints. To further augment the training set and to simulate more realistic images, we explored two data synthesis approaches to add backgrounds to the rendered home products: (1) **RandomScene**: Similar to the fine-grained classifier case, the home objects which are masked and segmented from the rendered images are directly pasted to randomly sampled indoor scene images. Affordance errors and scale mismatches are expected for the synthesized images, as can be seen in Figure 3. (2) **AutoPlacement**: An augmented reality tool is used to first find plausible regions on floor-like surfaces in the room to place the 3D home product model and then to generate realistic renderings. The position and orientation of the product can be varied within the determined regions. In row 1-3 of Figure 3, we present examples of images synthesized with RandomScene and AutoPlacement.

Both background synthesis methods improve the performance of the embedding model, but better yet, they have complementary properties. On the one hand, AutoPlacement data leads to a better result benefiting from its advantage in realism. On the other hand, the RandomScene dataset can theoretically grow to an arbitrarily large scale very efficiently and with low-cost, whereas the size of AutoPlacement dataset is limited by the number of available scenes. Finally, we found that by combining the Amazon product dataset, the rendered 3D model dataset, and the synthesized image dataset, the best retrieval performance is achieved. In Figure 4, we compare the image retrieval results before and after training the embedding model with augmented data. As we can see from both examples in the figure, the model trained with the augmented dataset successfully retrieves the correct items despite the large viewpoint variations. The new model also seems to be robust against occlusions, as shown in the second example.

3.5 Re-ranking and de-duping

The Amazon catalog contains a very large number of products from many sources. There is a significant variation in product quality

Figure 4: Comparison of the image retrieval results before and after training with the rendered 3D model dataset and the synthesized dataset. The results in the top row for each query are before, and those in the bottom row are after.

Figure 5: Duplicates detected by build-time (top) and run-time (bottom) algorithms. We show the duplicates detected by our build-time algorithm on the top part. Due to space constraints, we show only 8 results. We mark the high quality product from each cluster with a blue rectangle. Bottom part shows the query image, top-11 retrieved results with black rectangles around the duplicates, and the result after de-duping.

and popularity, and duplicate listings are common. Re-ranking aims to surface high-quality products from the visually similar candidates, while de-duping diversifies the results. Both are crucial for enhancing user experience.

3.5.1 Re-ranking. The initial retrieval results are computed using visual similarity only, but to ensure that high-quality products are returned to the costumer we re-rank them using several factors related to product quality and popularity, such as customer review ratings. We use a linear ranking model, and we learn the weights for each of the semantic and visual features so that a good balance between visual similarity and product quality is reached.

3.5.2 De-duping. We explore a two-step solution to remove near duplicate results. At build-time, the products are clustered using a graph-based formalism, tuned to achieve almost perfect precision. To remove any potential duplicates during run-time, a greedy algorithm is applied to remove products that are too similar from the nearest neighbor list. In the following, we denote the feature vector, detailed in Section 3.3, for the i-th product by $\boldsymbol{f}_i \in \mathbb{R}^N$ such that $\|\boldsymbol{f}_i\|_2 = 1$, where N is the feature dimension.

Build-time de-duping: The core of the build-time de-duping step is an unweighted graph where each node corresponds to a product, and with an edge between two nodes only when $\|\boldsymbol{f}_i - \boldsymbol{f}_j\|_2 \leq T$, where T is a distance threshold. This graph can be efficiently constructed, *e.g.*, by using a KD-tree to query products within a radius T from an origin product. The distance threshold T controls the density of graph and hence adjusts the trade-off between recall and precision. In our experiments, we use $T = 0.25$. After the graph is constructed, nodes are grouped by using clustering algorithm such as label propagation [20], with near-linear scalability with respect to the number of nodes, and we only keep in our index the high-quality product, as determined by our re-ranking algorithm.

Run-time de-duping: The purpose of run-time de-duping is to get rid of duplicates that might have passed through the build step. A greedy algorithm, similar to the Folding algorithm [26], can be used for this purpose: initialize the result set $\mathcal{R}$ with the first element in the list and then consider the i-th element to be non-duplicate (and add it to $\mathcal{R}$) only if $\|\boldsymbol{f}_i - \boldsymbol{f}_j\|_2 \geq k\sigma \quad \forall j \in \mathcal{R}$, where σ is the average distance between the feature vectors and their mean feature vector, and k is a hyper-parameter that controls the aggressiveness of the greedy algorithm. In our experiments, we use $k = 0.65$.

See Figure 5 for some examples of build- and run-time de-duping results. From the figure, we can see that the build-time algorithm is able to cluster images representing the same product but with different aspect ratios, crops, backgrounds, etc. Similarly, run-time de-duping is able to remove duplicates from the retrieval list.

3.6 Image Indexing and Retrieval

The data ingestion and index building pipeline converts hundreds of millions of items in the Amazon fashion and home catalog into a searchable index for the run-time retrieval service to consume. Designing this system to meet strict scalability and latency standards is challenging and requires significant engineering optimizations.

The index building pipeline is triggered by a distributed job scheduler. A data extraction job is used to pull the product information from the categories of interest in the Amazon catalog. However, the product category association in e-commerce catalogs can be very noisy and may require further clean-up in subsequent stages.

We first download the product images and process them using three component services: localization, fine-grained classification, and feature extraction. For images with multiple detection results, the most relevant bounding box can be identified based on detection and classification scores, as well as some product metadata. Furthermore, inconsistencies between the predicted category and metadata often indicate misplaced products, which could reduce the precision of the retrieval results if not discarded.

The Amazon product catalog is volatile. Items are frequently added and removed due to availability, seasonal change or trend

Fashion			Home		
Category	Recall	Prec.	Category	Recall	Prec.
Top	86%	89%	Chair	86%	78%
Bottom	89%	91%	Table	87%	79%
Dress	84%	84%	Lighting	85%	84%
Shoes	89%	95%	Sofa	91%	89%
Bag	86%	93%	Ottoman	89%	86%

Table 1: Precision and recall of the localization models.

Fashion			Home		
Category	Top-1	Top-2	Category	Top-1	Top-2
Top	72%	81%	Chair	89%	96%
Bottom	71%	86%	Table	87%	94%
Dress	N/A	N/A	Lighting	90%	97%
Shoes	72%	85%	Sofa	93%	97%
Bags	75%	89%	Ottoman	94%	98%

Table 2: Accuracy of the fine-grained classifiers. For dress we found no benefit in adding a fine-grained classifier.

Fashion				Home			
Category	@1	@10	@100	Category	@1	@10	@100
Top	93%	97%	100%	Chair	86%	97%	99%
Bottom	91%	97%	100%	Table	87%	96%	99%
Dress	90%	98%	100%	Lighting	81%	95%	99%
Shoes	92%	96%	99%	Sofa	87%	96%	100%
Bags	83%	92%	97%	Ottoman	90%	99%	100%

Table 3: Recall@k of the image embedding model.

Component	Latency (ms)
Localization model	127
Fine-grained recognition model	63
Embedding model	90
Nearest neighborhood query	12
Re-ranking and run-time de-duping	175

Table 4: Latencies of different modules in the system.

shift. To closely track these changes, we need to re-build the index regularly, which can be done incrementally by keeping track of the previous iterations: only new products and products which were changed will need to undergo the new indexing computation. Due to the very large number of products in the catalog, the task of determining the differential input is itself computationally intensive, and benefits from Map-Reduce jobs in terms of efficiency.

To achieve fast retrieval at run-time, we build an independent approximate nearest neighbor index for each fine-grained category. We implement our local optimized FLANN (Fast Library for Approximate Nearest Neighbors) [19], which reduces latency in half at p50, and by 40% at p90, compared to distributed HNSW (Hierarchical Navigable Small World) [17] for our usual volume of search traffic.

We use Map-Reduce to order, group, and aggregate data for ranking, and we store it in the index for fast access at run-time. In order to scale build-time de-duplication (see Section 3.5.2), this operation is performed in parallel for each fine-grained index. Further ad-hoc optimization (e.g. skipping the graph nodes that have no updates in their labels, choice of correct data structures and libraries) on the build-time de-duplication achieves a 10x speedup.

By storing artifacts from different steps such as localization, feature extraction and de-duplication they could be used to effortlessly perform analysis, debugging and sanity checks using interactive SQL query services. Keeping the embedding vectors in DynamoDB [23], a highly scalable key-value store, enables efficient querying in batches during build and run-time. For example, it can serve as a feature repository for re-ranking and de-duplication.

4 EXPERIMENTS

We present quantitative results for various components of the system in our internal human-annotated evaluation datasets, composed of thousands of randomly sampled images, that were also used to make decisions for the system in production. We also show the performance of the end-to-end system both quantitatively and qualitatively. For most results, we report category-wise numbers.

Object localization: Table 1 shows the precision and recall obtained by the fashion and home models on our evaluation set at the production operating point, which is a more relevant analysis from an application point of view. We discard overly truncated or occluded detections to improve the customer experience, as they often result in less accurate recommendations. Typical failure cases include small footwear or bag instances, strong occlusion, or unusual viewpoint. For dresses with differently colored top and bottom parts, they are sometimes mistaken as separate garments.

Fine-grained classification: We report top-1 and top-2 accuracy for our classifiers in Table 2. It is worth noting that although top-1 accuracy for some categories (e.g., bottom) is relatively low, the top-2 accuracy is high. This is relevant as multiple indices can be queried. Furthermore, as fine-grained recognition is run at both build- and run-time, consistent recognition errors are alleviated as the product will be stored and searched in the same index.

Embedding models: We use Recall@k to evaluate embedding performance on our test set, defined as the percentage of queries for which at least one correct match is included in the top-k returned results (each query has at least one exact match in the database). This should not be confused with the end-to-end evaluation protocol, where relevance is the main metric and there is no guarantee that the exactly same item exists in our product catalog. We present the per-category Recall@k for $k = 1, 10, 100$ in Table 3. The fashion result and home result are evaluated on two different test sets, so the two groups of numbers are not directly comparable. In the Fashion retrieval case, we see that bags have a lower Recall@k than the other categories. It might be due to the fact that bags suffer from viewpoint variation challenges, and that the straps of bags are often too long to be included in the localized bounding boxes. The synthesis methods described in Section 3.4 improve the embedding performance, for example, Recall@1 of sofa increases from 82% to 87% and for lighting increases from 77% to 81%.

Latency: Table 4 shows the average latencies of different components in the system. These include the pre- and post-processing steps for each model, and GPU communication. We see that re-ranking and run-time de-duping steps incur the highest latency, since the computation is CPU-bound, and needs to process several hundreds of retrieved results. We have identified several paths to optimize the run-time de-duping algorithm as part of future improvements. Also, we plan to experiment with techniques like

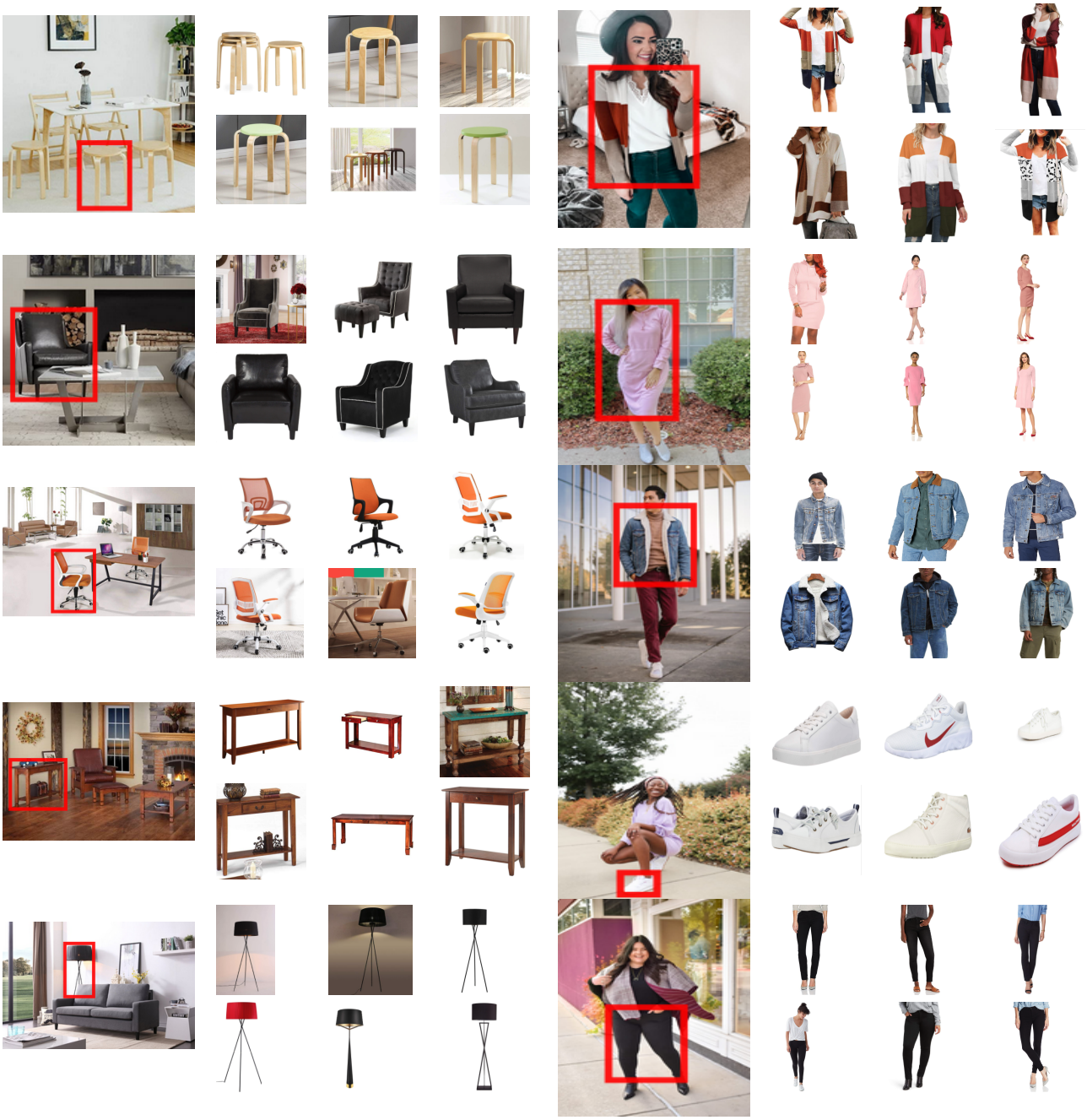

Figure 6: Qualitative end-to-end results of the visual search system. For each query image, we show a red bounding box around the object of interest and present the top-6 retrieved results arranged in two rows.

pruning and quantization to reduce our model latencies further, while not affecting the performance.

End-to-end evaluation: Figure 7 shows the NDCG (Normalized Discounted Cumulative Gain) @k on an internal test dataset, where k corresponds to the number of items considered in the retrieved result. The results have undergone re-ranking and de-duping, and annotators follow standardized guidelines to label the relevance of each (query, result) pair, with only a few products in the database being considered relevant for each query. Results with different product category or gender than the query are considered as search defects regardless of visual similarity. As can be seen, for all categories the top ten results consist of highly relevant items. Finally, we present qualitative results in Figure 6. We observe that Shop the Look is able to retrieve visually similar products from

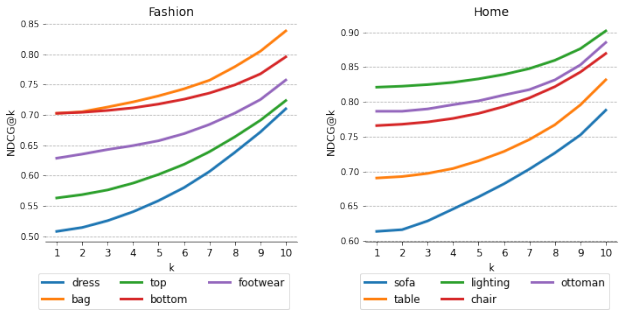

Figure 7: System end-to-end performance. Note that the y-axes have different scales for fashion and home.

the Amazon catalog even when large pose variations, cluttered backgrounds, varying illumination, and occlusions are present in the query images.

5 CONCLUSION

In this paper, we introduced a visual search engine for fashion and home products. We discussed the main challenges of building such an end-to-end system under strict accuracy and latency constraints at web-scale. We presented how the state-of-the-art computer vision technologies can be used to model visual similarity, and specifically how to address the domain gap issue in the targeted "street/home-to-shop" application scenario. We performed optimizations at bottlenecks of the processing pipeline to improve system scalability. We presented extensive experimental results to demonstrate the performance of the system.

Some important lessons are learned, like that image synthesis-based data augmentation can go a long way towards improving the retrieval performance. Furthermore, in the verticals considered in this work, we have the opportunity of taking advantage category-specific priors to improve the model performance.

We are interested in exploring several future directions to improve Shop the Look. First, to increase the accuracy of our models, we plan to leverage weakly-supervised learning approaches to train on even larger datasets. This will allow us to avoid the formidable cost of using human annotations. Second, we also plan to experiment with relevance feedback to improve customer experience. Introducing user feedback signals into Shop the Look may help with the most ambiguous search queries from our customers. Finally, we will continue to expand the scope of Shop the Look to support more product categories suitable for visual search.

REFERENCES

[1] Sean Bell and K. Bala. 2015. Learning visual similarity for product design with convolutional neural networks. *ACM Trans. on Graphics (TOG)* 34 (2015), 1 – 10.

[2] Sean Bell, Yiqun Liu, Sami Alsheikh, Yina Tang, Edward Pizzi, M Henning, Karun Singh, Omkar Parkhi, and Fedor Borisyuk. 2020. GrokNet: Unified computer vision model trunk and embeddings for commerce. In *Proc. of the 26th ACM SIGKDD Int. Conf. on Knowledge Discovery & Data Mining*. 2608–2616.

[3] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. 2020. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint:2004.10934* (2020).

[4] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. 1994. Signature verification using a" siamese" time delay neural network. *Advances in neural information processing systems* (1994), 737–737.

[5] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. 2020. End-to-end object detection with transformers. In *European Conf. on Computer Vision*. Springer, 213–229.

[6] D. Dwibedi, I. Misra, and M. Hebert. 2017. Cut, Paste and Learn: Surprisingly Easy Synthesis for Instance Detection. In *2017 IEEE Int. Conf. on Computer Vision (ICCV)*. 1310–1319.

[7] Ben Harwood, Vijay Kumar B G, Gustavo Carneiro, Ian Reid, and Tom Drummond. 2017. Smart Mining for Deep Metric Learning. In *Proc. of the IEEE Int. Conf. on Computer Vision (ICCV)*.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proc. of the IEEE Conf. on computer vision and pattern recognition*. 770–778.

[9] Elad Hoffer and Nir Ailon. 2015. Deep metric learning using triplet network. In *International workshop on similarity-based pattern recognition*. Springer, 84–92.

[10] Houdong Hu, Yan Wang, Linjun Yang, Pavel Komlev, Li Huang, Xi Chen, Jiapei Huang, Ye Wu, Meenaz Merchant, and Arun Sacheti. 2018. Web-scale responsive visual search at Bing. In *Proc. of the 24th ACM SIGKDD Int. Conf. on knowledge discovery & data mining*. 359–367.

[11] P Izmailov, AG Wilson, D Podoprikhin, D Vetrov, and T Garipov. 2018. Averaging weights leads to wider optima and better generalization. In *34th Conf. on Uncertainty in Artificial Intelligence 2018, UAI 2018*. 876–885.

[12] Yushi Jing, David Liu, Dmitry Kislyuk, Andrew Zhai, Jiajing Xu, Jeff Donahue, and Sarah Tavel. 2015. Visual search at Pinterest. In *Proc. of the 21th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*. 1889–1898.

[13] Sungyeon Kim, Dongwon Kim, Minsu Cho, and Suha Kwak. 2020. Proxy Anchor Loss for Deep Metric Learning. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*.

[14] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proc. of the IEEE Int. Conf. on computer vision*. 2980–2988.

[15] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. 2016. Ssd: Single shot multibox detector. In *European Conf. on computer vision*. Springer, 21–37.

[16] Ziwei Liu, Ping Luo, Shi Qiu, Xiaogang Wang, and Xiaoou Tang. 2016. Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In *Proc. of the IEEE Conf. on computer vision and pattern recognition*. 1096–1104.

[17] Yu A Malkov and Dmitry A Yashunin. 2018. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence* 42, 4 (2018), 824–836.

[18] Yair Movshovitz-Attias, Alexander Toshev, Thomas K Leung, Sergey Ioffe, and Saurabh Singh. 2017. No fuss distance metric learning using proxies. In *Proc. of the IEEE Int. Conf. on Computer Vision*. 360–368.

[19] Marius Muja and David G Lowe. 2009. Fast approximate nearest neighbors with automatic algorithm configuration. *VISAPP (1)* 2, 331-340 (2009), 2.

[20] Usha Nandini Raghavan, Réka Albert, and Soundar Kumara. 2007. Near linear time algorithm to detect community structures in large-scale networks. *Physical review E* 76, 3 (2007), 036106.

[21] Joseph Redmon and Ali Farhadi. 2018. YOLOv3: An Incremental Improvement. *arXiv* (2018).

[22] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. *arXiv preprint arXiv:1506.01497* (2015).

[23] Swaminathan Sivasubramanian. 2012. Amazon DynamoDB: A Seamlessly Scalable Non-Relational Database Service *(SIGMOD '12)*. Association for Computing Machinery, New York, NY, USA, 729–730.

[24] William Thong, Cees G. M. Snoek, and Arnold W. M. Smeulders. 2019. Cooperative Embeddings for Instance, Attribute and Category Retrieval. arXiv:1904.01421 [cs.CV]

[25] Son Tran, Ming Du, Sampath Chanda, R Manmatha, and Cj Taylor. 2019. Searching for Apparel Products from Images in the Wild. In *The fourth international workshop on fashion and KDD*.

[26] Reinier H Van Leuken, Lluis Garcia, Ximena Olivares, and Roelof van Zwol. 2009. Visual diversification of image search results. In *Proc. of the 18th Int. Conf. on World wide web*. 341–350.

[27] Jian Wang, Feng Zhou, Shilei Wen, Xiao Liu, and Yuanqing Lin. 2017. Deep Metric Learning With Angular Loss. In *Proc. of the IEEE Int. Conf. on Computer Vision (ICCV)*.

[28] Chao-Yuan Wu, R. Manmatha, Alexander J. Smola, and Philipp Krähenbühl. 2017. Sampling Matters in Deep Embedding Learning. In *2017 IEEE Int. Conf. on Computer Vision (ICCV)*.

[29] Fan Yang, Ajinkya Kale, Yury Bubnov, Leon Stein, Qiaosong Wang, Hadi Kiapour, and Robinson Piramuthu. 2017. Visual search at Ebay. In *Proc. of the 23rd ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*. 2101–2110.

[30] Andrew Zhai, Hao-Yu Wu, Eric Tzeng, Dong Huk Park, and Charles Rosenberg. 2019. Learning a unified embedding for visual search at Pinterest. In *Proc. of the 25th ACM SIGKDD Int. Conf. on Knowledge Discovery & Data Mining*. 2412–2420.

[31] Yanhao Zhang, Pan Pan, Yun Zheng, Kang Zhao, Yingya Zhang, Xiaofeng Ren, and Rong Jin. 2018. Visual search at Alibaba. In *Proc. of the 24th ACM SIGKDD Int. Conf. on Knowledge Discovery & Data Mining*. 993–1001.