Unsupervised Construction of a Product Knowledge Graph

Omar Alonso, Vasileios Kandylas, Rukmini Iyer
Microsoft
omalonso,vakandyl,rukmini@microsoft.com

ABSTRACT

We describe techniques for generating a commercial knowledge graph with the goal of discovering relationships between brands, products, and categories. Using a diverse set of input data sources, we implement an unsupervised, efficient and scalable data pipeline that produces a brand-product graph. We also outline a number of challenges encountered when processing this type of data in the context of commercial products.

ACM Reference format:
Omar Alonso, Vasileios Kandylas, Rukmini Iyer. 2019. Unsupervised Construction of a Product Knowledge Graph. In *Proceedings of ACM SIGIR Workshop on eCommerce, Paris, France, July 2019 (SIGIR 2019 eCom),* 4 pages.

1 INTRODUCTION

We are interested in constructing a commercial knowledge graph data asset that revolves around brands, products, and categories. Our graph allows users to query for a brand, say Microsoft, and retrieve associated products (e.g., Surface 3, Windows 10, Xbox, etc.) and to query for a product, say jeans, or a category, say clothing, and retrieve the associated brands (e.g., Calvin Klein, Hudson, Armani, etc.).

There are number of challenges in the construction a product graph at scale. Compared to previous work in building knowledge graphs that use Wikipedia as a source, there are no major sources that contain clean information about brands and products. Commerce is a very dynamic domain as new brands and products can appear, or old ones can be retired. It is hard to define and to detect products: articles of clothing tend to have descriptive rather than distinctive names (e.g., men's black cotton t-shirt, large), cheap and mass-produced products often don't have a mentioned brand (e.g., 1 inch iron nails), and service-oriented products are not well defined (e.g., term life insurance). Popular brands have distinct names but homonymous brands from different fields can be problematic (e.g., Delta, Apple). The distinction between brand and product is sometimes blurred (e.g., Is Amazon Video a product or a brand?) and, while there are textual descriptions in product catalogs that can be used for extracting structured data, retailers do not always provide clean data.

Several machine-readable knowledge bases (KBs) have been built in the last two decades that include the latest advances on information extraction, harvesting Web resources at scale, and machine learning. One of the few published reports on the end-to-end implementation and maintenance of KBs in industrial settings is the work by Deshpande et al. [1] that describes Kosmix (later acquired by Walmart). Very recently, Amazon is working on Product Graph, an authoritative knowledge graph for all products in the world that includes a knowledge extraction framework on semi-structured product websites by mining DOM trees [2] and OpenTag, an active learning approach for extracting attributes and values from product titles and descriptions [5]. Finally, existing popular KBs have knowledge gaps and detection of long-tail entities is one of main challenges with respect to coverage [4].

2 PRODUCT GRAPH

We define terminology as follows. A *brand* is a term or phrase that distinguishes an organization or product (e.g., Adidas, Calvin Klein, Microsoft). A *domain* is the most common URL associated with the brand (e.g., Microsoft's domain is `www.microsoft.com`). A *product* is an item that is manufactured for sale (e.g., Microsoft Surface 3, Gucci Guilty, Google Pixel) or a service provided (e.g., insurance, pet cleaning, food delivery). An *alias* is a name that an item is otherwise called or known as. In the case of Apple, a set of aliases is Apple Inc., Apple Computer Inc., and AAPL. *Categories* group items into a given label or name (e.g., BMW→vehicles).

Our proposed unsupervised approach is as follows. We start with generating lists of brands from multiple sources using aggregation functions. We tag brands with domains and categories using a combination of query log mining and modeling. Finally, we derive products using multiple modeling approaches on advertiser provided data (bidded keywords and product offers from their product catalogs). Since advertisers also provide us brand information, we can associate products directly back to brands in the graph. We start with a bottom-up strategy with a focus on simplicity and data cleaning, that allows fast iteration and debugging. The proposed graph construction methodology is designed so that new sources of data can be added easily without affecting the existing process significantly and without requiring extensive redesign of the graph.

2.1 Brands

For brand generation, we use as input n catalog sources, $s_1, \ldots, s_n$, that contain information about brands and produce a table where each source assigns 1 vote according to the presence of such term in their respective sources and a 0 vote if there is no presence. A final score is computed based on such votes as presented in the example in Table 1. Because source size varies and different sources may have different name variations and spelling, a grouping of similar brands is performed using alias data that updates the final information for names and votes (e.g., The Gap Inc and Gap, Apple and Apple Inc). Some sources may be more reputable than others in terms of data quality so we perform the final selection based on source authoritativeness and $score > t$, where t is a threshold for the number of votes. This simple mechanism allow us to incorporate and maintain input sources in a flexible way and, at the same time, identify areas that brand coverage may be low (e.g., end-consumer electronics vs. pharmaceutical).

Term	Alias	s_1	s_2	...	s_n	score
Apple	Apple Inc.	1	1	...	1	1
Bose	Bose audio	1	1	...	0	0.7
Gap	The Gap Inc.	0	1	...	1	0.8

Table 1: Brand list generation example. Each source s votes on the presence of a term.

2.2 Brand Domains

A brand domain is the URL that is associated with a brand and we would like to automatically extract this information for all possible brands in our graph as the domains are important for categorization and also for linking product information. Sometimes brands will have more than one URL, like in the case of Microsoft (e.g., `microsoft.com`, `bing.com`, `visualstudio.com`). For head brands, domain information can be obtained by using Wikipedia's infoboxes, but for tail brands such information is not available. To make it scale, we implement an algorithm that uses Bing search query logs to derive domain information by extracting queries and their associated top-10 search result page (titles & links) for queries that are normal (not bots) and no adult content. By computing the most frequent URLs at positions 1 through 10 over 3 months of historical data, we extract the most frequent URL at position 1 as the domain owner for a brand query.

The final brand list contains alias, domains (one or more domains associated), and score as presented in Table 2. This data is then used as input for the categorization step that we describe in the next subsection.

Term	Alias	Domain	score
Apple	Apple Inc.	`apple.com`	1
Bose	Bose audio	`bose.com`	0.7
Gap	The Gap Inc.	`gap.com`, `gapinc.com`	0.8

Table 2: Brand list generation example with domains.

2.3 Categorization

We now describe a method where brand categories are computed via the queries which are related to the brands through the brand URLs. Brands with no URLs discovered for them are not categorized. The categorization happens in the following sequential steps: brands $\longrightarrow$ brand URLs $\longrightarrow$ search queries to URLs $\longrightarrow$ query categories $\longrightarrow$ aggregation of categories $\longrightarrow$ categories.

The computation relies on factorization of probabilities on a tri-partite $\{B, U, Q\}$ graph to generate the probability of category given brand using:

$$P(C|B) = \sum_U \left[\sum_Q P(C|Q) \cdot P(Q|U) \right] P(U|B)$$

All URLs per brand are treated as equal, so $P(U|B)$ is uniform.

We use Bing search query logs to get query-URL click information and perform the following filtering: navigational queries, queries issued less than 2 times in 6 months and URLs with less than 5 clicks in 6 months are removed. These thresholds can be optimized but the goal is to remove the rare queries and rare URLs. The URLs are matched to the brand URLs according to whether they fall under them or not. For the brand URL `www.microsoft.com/en-us`, the query `www.microsoft.com/en-us/surface` does fall under it, but the query `www.microsoft.com/surface` does not. The probability of a query given the URL is based on the clicks on the URL for the query:

$$P(Q|U) = \frac{clicks(Q, U)}{clicks(U)}$$

.

Each query is categorized using an internal categorizer that returns categories using an internal commercial ad taxonomy. The categorizer can return more than 1 category per query and category scores lie between 0 and 1. It operates on a hierarchical taxonomy of categories (e.g. Apparel > Footwear > Athletic Shoes). The categorizer uses the web result snippet besides the query, which improves its accuracy. The categorizer scores are used as the probabilities of category given query for the computation $P(C|Q) = score_C(Q)$.

At this stage, each brand is associated with multiple categories through multiple queries and brand URLs. The same categories appearing many times with different scores are aggregated to produce a final set of unique categories with scores for the brand. The aggregation reduces the impact of mistakes in the query categorization. The final score per category for a brand is basically the weighted average of the query category scores.

The final score accuracy depends on the number of queries and clicks available for the brand. Therefore we reduce the final score based on the uncertainty:

$$Score(Q|B) = max \left(0, P(Q|B) - \frac{\alpha_1}{\sqrt{clicks(B)}} - \frac{\alpha_2}{\sqrt{queries(B)}} \right)$$

where α_1 and α_2 are manually tuned parameters. The score is the probability reduced by the second and third terms in the equation above, which are corrections based on the uncertainty of the data. Given n data points, the mean estimate has a standard error proportional to $1/\sqrt{n}$, so the fewer clicks or queries are available, the less reliable the score is and the more it is discounted. As the number of clicks and the number of queries for the brand increase, the corrections vanish. The fewer clicks and queries are available, the less reliable the estimate is and the lower the score.

For each category and brand, we maintain a small set of categorized queries as *provenance*. These explain why a brand belongs to a category. E.g. Apple is categorized, among other things, as Apparel, because of the query "apple watch" which belongs to category Apparel $\longrightarrow$ Jewelry $\longrightarrow$ Watches & Watch Accessories.

A useful side effect of the previous computations is the popularity of the brand, which is based on the clicks to the brand URLs. The overall brand popularity is then

$$P(B) = clicks(B) / \sum_b (clicks(b)$$

The brand popularity per category is computed as

$$P(B|C) = \frac{P(C|B)P(B)}{P(C} \propto P(C|B)P(B)$$

.

2.4 Products

Identifying products is a much more difficult problem because products can be referenced in a coarse or granular manner depending on the source of data. In search queries, we see simplified representations of products, for example "Aveeno lotion". But, in product catalogs, we observe very detailed description of products like "Aveeno Active Naturals Daily Moisturizing Lotion 2 x 18 oz".

Products also include services provided by brands (examples include insurance, car cleaning, printing) and not necessarily only retail products. Another complication is that products contain more generic terms and specifications (e.g., models, parts, size, etc.).

We have two very different approaches to deriving products, to suit the source of data that we are deriving them from. A sample of products for Microsoft and Apple derived with the two methods is presented in Table 3.

2.4.1 Products from Retailer Catalogs. Retailers upload product catalogs and the Bing search engine serves product ads for search queries. The product ads corpus typically contains brand, product title or name, GTIN (Global Trade Item Number) and MPN (Manufacturer Product Number) information among other data. To generate products for brands, we first perform a simple grouping of products within each brand. Since many ads are missing GTIN and MPN and the MPNs tend to be noisier than GTINs, we first group together all products with the same name and GTIN. Then we group all products with the same name, regardless of GTIN and MPN. As a next step, we use a CDSSM [3] trained on the product ads corpus to embed the product names in a lower dimensional vector space. The lower dimensional space facilitates the next round of product clustering using k-means with some heuristics to split clusters of large size or high variance. This process produces a hierarchical organization of products, where each higher level is more general than the previous. Each cluster represents a group of semantically related product titles and contains many similar product names.

We generate a set of M representative labels for each cluster using a simple generative algorithm. We compute the average length l of product names within a cluster. We generate top N n-grams of length l, embed the n-grams using the same CDSSM encoding as what we used in the original clustering, and choose the best M n-grams based on the distance of their CDSSM vectors from the cluster centroid. We repeat this process for higher and lower lengths of n-grams, iterating until we stop finding any additional labels to insert or replace in the top M labels.

2.4.2 Products from Bidded Keywords. For this approach we use a data set of keywords on which the advertisers bid to display their ads. Besides the bidded keywords, the data set also contains the URL of the ad and the frequency that the bidded keyword was triggered. First, the bidded keywords are matched to the brands by ensuring that the ad URL matches the known brand domain and that the bidded keywords contain the brand name. From the set of keywords selected, we generate n-grams (n <= 4), after removing stop words and locations. For every brand, each n-gram associated with the brand is scored twice: one uses the overall n-gram frequencies from the whole data set and the other uses the brand-specific n-gram frequencies. The score itself is a traditional language model score, computed under a Markov assumption:

$$P(w1w2w3w4...) = P(w1)P(w2|w1)P(w3|w2,w1)P(w4|w3,w2)...$$

where the individual probabilities are smoothed using linear interpolation with lower length n-grams. The n-grams are ranked using KL divergence:

$$score_{phrase} = P(phrase|brand) \cdot log(P(phrase|brand)/P(phrase))$$

The phrases above a certain threshold are used as products associated with the brand. This method tends to produce more general products, like "LED TV", or "men's t-shirt", so it complements well the previous method, which tends to produce more specific products, like "40W flood light bulb, medium base, dimmable".

3 RESULTS AND EVALUATION

The complete back-end data pipeline and algorithms have been implemented in Cosmos, Microsoft's computation service and run on a distributed cluster continuously. Our data pipeline identifies around 130K brands, 1M products, and 22 top-level categories for brands. The average number of top-level categories per brand is 7.6. We can re-build the graph from scratch within a few hours which allow us to deploy new changes very efficiently. The graph is timestamped so it is possible to observe brand and product evolution over time.

Due to confidentiality, however, we do not disclose user engagement or other behavioral data. Instead, we present an offline relevance evaluation for different components of the graph. We conducted an offline evaluation of 3,000 brands selected at random with an internal tool for collecting labels using in-house editors and report precision numbers in Table 4 grouped by different brand scores. The task consisted in showing a brand to a judge and requires answering if such brand is indeed a brand, not a brand, or if the judge cannot tell.

Evaluation for categories was also conducted using the same internal tool where we provide editors with a brand and a list of categories we have associated with it. The task is to mark the categories that are not correct. We also ask them to check a box if they think a category is missing from those provided. Optionally, they can also write what category is missing, if any. Marking the incorrect categories allows us to estimate precision, while the checkbox to calculate recall. We evaluated a sample of 2,000 brands, stratified according to score. We gathered 3 responses per brand, treating the majority answer as correct. Figure 1 shows the ROC curve of our brand categorizer for selected brand score thresholds.

4 CONCLUSION AND FUTURE WORK

In this paper we presented an efficient and scalable brand-product graph generation data pipeline that is currently used in an industrial setting. We described a number of unsupervised techniques for deriving brands, brand categories, and products using different input data sources. Preliminary evaluation results show that our implementation produces good quality output. We described the

Brand	Bidded keyword products	Retailer catalog-based products
microsoft	microsoft word	Microsoft Office Home & Student 2016 For Mac
microsoft	microsoft office	Microsoft Office 365 Personal Subscription 1 Year 1 User Pc/Mac
microsoft	microsoft office 365	Microsoft Office Professional Plus 2016 365 Pro Word Excel Key Windows
microsoft	microsoft outlook	Microsoft Office Home and Business 2016 Retail License
microsoft	microsoft surface	Microsoft 13.5" Surface Book 2-in-1 Notebook 256GB SSD 8GB RAM Core i5 2.4 GHz
microsoft	microsoft teams	12.3 Surface Pro 6 - 128GB / Intel Core m3 / 4GB RAM (Silver)
microsoft	microsoft project	Surface Pro 6 - 512GB / Intel Core i7 / 16GB RAM (Platinum)
microsoft	microsoft excel	Surface Pro 4 12.3" Bundle: Core i5, 4GB RAM, 128GB SSD, Surface Pen, Type Cover
microsoft	microsoft office 2016	Microsoft Project Professional 2016 - Digital Download
microsoft	microsoft powerpoint	Microsoft Project 2019 Professional w/ 1 Server CAL Open License
apple	apple watch	Apple Watch Series 3 44MM Rose Gold
apple	apple airpods	Apple Watch Series 4 44 mm Gold Stainless Steel Case with Gold Milanese Loop
apple	apple watch series 3	24K Gold Plated 42MM Apple Watch Series 3 Diamond Polished Modern Gold Link Band
apple	apple watch series 4	Apple Airpods MMEF2J/A Wireless Earphone For Iphone/Apple Watch/Ipad/Mac
apple	apple iphone	Apple Airpods Genuine Left-Only Airpod (Without Charging Case)
apple	apple watch 4	Apple Airpods - White MMEF2AM/A Genuine Airpod Retail Box
apple	apple ipad	Apple iPhone 8 Plus - 64GB - Space Gray (Unlocked) A1864 (CDMA + GSM)
apple	apple iphone xs max	Apple iPhone XR 256GB, Yellow - Sprint
apple	apple tv	Apple iPad Wi-Fi 128GB Silver
apple	apple earpods	Apple iPad 6th Gen. 32GB, Wi-Fi + Cellular (Unlocked), 9.7in - Silver #15718

Table 3: Example products from Apple and Microsoft returned by the two methods. The bidded keyword-based products are more high-level, whereas the products from retailer catalogs have finer granularity.

Brand score	Precision
1	92.2%
0.83	87.8%
0.66	87.2%
0.5	85.6%

Table 4: Brand evaluation.

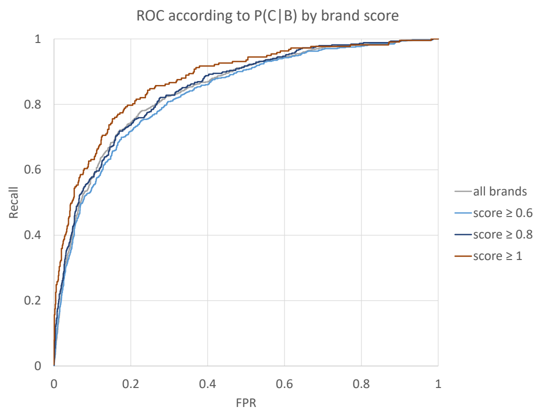

Figure 1: Root categories ROC.

methods in detail so it should be possible to reproduce the results using similar input sources.

Our philosophy is to build a graph using a rapid and iterative development process where at each stage we can test, debug, and explain changes as we ingest new data sources and the size of the graph grows. While the graph is far from complete, the data is already in use by internal applications and as training set for classifiers and NERs. Thus, we agree with the Kosmix case study [1] that an imperfect KB is still useful for real-world applications that can help test and identify problems with the new derived data set.

Future work includes product evaluation, product generation from query logs, improvements with brand conflation which should improve the quality of brand data, and functionality for detecting similar brands and products.

5 ACKNOWLEDGMENTS

We thank Karim Filali and Jaya Palli for help and assistance.

REFERENCES

[1] Omkar Deshpande, Digvijay S. Lamba, Michel Tourn, Sanjib Das, Sri Subramaniam, Anand Rajaraman, Venky Harinarayan, and AnHai Doan. 2013. Building, Maintaining, and Using Knowledge Bases: A Report from the Trenches. In *Proc. of SIGMOD*. 1209–1220.
[2] Colin Lockard, Xin Luna Dong, Prashant Shiralkar, and Arash Einolghozati. 2018. CERES: Distantly Supervised Relation Extraction from the Semi-Structured Web. *PVLDB* 11, 10 (2018), 1084–1096.
[3] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. Learning Semantic Representations Using Convolutional Neural Networks for Web Search. In *Proc. of WWW*. 373–374.
[4] Gerhard Weikum, Johannes Hoffart, and Fabian M. Suchanek. 2016. Ten Years of Knowledge Harvesting: Lessons and Challenges. *IEEE Data Eng. Bull.* 39, 3 (2016), 41–50.
[5] Guineng Zheng, Subhabrata Mukherjee, Xin Luna Dong, and Feifei Li. 2018. OpenTag: Open Attribute Value Extraction from Product Profiles. In *Proc. of KDD*. 1049–1058.