Semantic In-Domain Product Identification for Search Queries

Sanat Sharma[1], Jayant Kumar[1], Twisha Naik[1], Zhaoyu Lu[1], Arvind Srikantan[1] and Tracy Holloway King[1]

[1]*Adobe Inc., San Jose, California, USA*

Abstract

Accurate explicit and implicit product identification in search queries is critical for enhancing user experiences, especially at a company like Adobe which has over 50 products and covers queries across hundreds of tools within those products. Whether users come to learn about and purchase new products, to launch or download products they have already purchased, or to get help on products, accurate product identification is key to surfacing relevant search results and product cards. In this work, we present a novel approach to training a product classifier from user behavioral data. Our semantic model led to: >25% relative improvement in CTR (click through rate) across the deployed surfaces; a >50% decrease in null rate; a 2x increase in the app cards surfaced, which helps drive product visibility.

Keywords

semantic search, explicit NER, implicit NER, autocomplete, query understanding

1. Introduction

Adobe boasts over 50 products for a variety of creative use cases (e.g. editing photos, videos, and audio, creating illustrations, animations, and vector graphics). When users come to Adobe.com or to Creative Cloud (CC, a subset of Adobe products focused on creativity), it is critical to route them to the right product for their use case. Users issues queries on these surfaces to learn about and purchase new products, to launch or download products they have already purchased, or to get help on products. In all of these cases, accurate product identification is key to surfacing relevant results.

Adobe.com and CC have product-focused search experiences that are augmented by contextual app card suggestions in autocomplete and at the top of search results. App cards provide users an easy way to discover, learn more about, or simply launch the Adobe product that matches their query intent. These app cards are the most clicked items on app-agnostic surfaces like CC and Adobe.com and are critical in driving new-user acquisition and product discovery, as well as providing existing users with help for their queries. Example app card triggering in autocomplete for implicit product intent is shown in Figure 1.

Initially, this matching was done via regular expression rules and simple named entity recognition. While this approach gave product experts the ability to curate the experience for end users, there were multiple problems with this approach.

eCom'24: ACM SIGIR Workshop on eCommerce, July 18, 2024, Washington, DC, USA

sanatsha@adobe.com (S. Sharma); jaykumar@adobe.com (J. Kumar); tnaik@adobe.com (T. Naik); lolu@adobe.com (Z. Lu); asrikantan@adobe.com (A. Srikantan); tking@adobe.com (T. H. King)

- **Scalability**: Due to the rule-based nature of the matching, this approach was hard to scale to torso and tail queries. Minor variations in phrasing would result in app cards not triggering and queries with implicit product intent (e.g. *edit video* should trigger app cards for Premiere Pro and Rush) rarely triggered app cards. This led to a high null rate (>50%). Furthermore, it was particularly hard to scale across the different languages supported by Adobe products.
- **Non-uniformity**: There was a lack of cohesion between the behavior of app cards shown in autocomplete and search results, which are maintained by different teams. This led to a poor user experience.

Our system was able to solve both issues found in the previous approaches. We present a low-latency query-to-product semantic matching system that provides contextual app card suggestions for the search and autocomplete services.

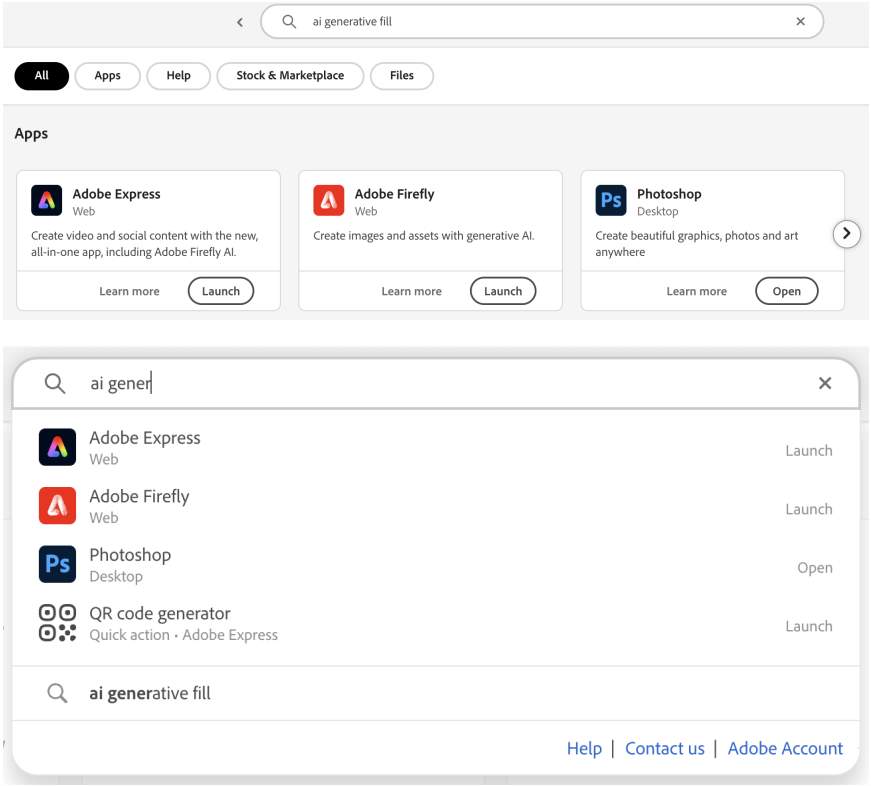

Figure 1: Product App Card Experiences: Top: App cards at the top of search results for *ai generative fill*. Bottom: Autocomplete for *ai genera*; textual query suggestions are shown below the app cards. In both of these query prefixes, the product intent is implicit, i.e. no Adobe product name is mentioned.

2. Prior Art

Product disambiguation has often been modeled as a (named) entity recognition (NER) approach in industry. Most approaches to query product disambiguation have been to model tokens within queries and extract relevant entities based on the set of supported products [1, 2]. Recent works have also leveraged autoregressive re-writing of the query for easier disambiguation and then using a retrieval or classification head on top [3]. There has also been recent research for NER tagging in low-resource cases where named entities are specialized [4], which is the case for Adobe product intent.

In addition to NER techniques, other works have focused on a semantic search approach of mapping product embeddings and query embeddings in the same semantic space [5].

Our work utilizes a mixture of components from previous approaches. We pretrain a language model (LM) on our internal document set to learn the intricacies of Adobe products [6] and then utilize a classification approach on top of the LM for product matching. We found this approach to work better than semantic search for products which are less frequently referenced in queries and which are less popular with users due to their highly specific applications (e.g. Adobe Bridge). In addition, this approach is much better than NER in cases where no products are explicitly mentioned in the user query (e.g. *redact document* for Acrobat and *edit video* for Premiere Pro and Rush).

3. Datasets

We support 46 Adobe products in our training dataset. The datasets are in English but come from multiple locales since English queries are used in combination with non-English in most locales. Also, Adobe products have the same official name in all locales and languages. In order to learn a good representation of the products and to tackle a diverse set of queries, we utilize four datasets, from user clicks to expert-maintained spreadsheets. These are described below.

Adobe HelpX Behavioral Dataset Adobe HelpX[1] gets millions of unique visitors every year looking for tutorials and learn content (Adobe HelpX articles, Adobe help videos) related to Adobe products. We utilize the click logs from user queries → HelpX article clicked to generate our query → product dataset. From each help article, we extract the related product from the metadata. We use logs from January 2021 – August 2022 for our training set. This dataset is noisy but provides a large and diverse dataset that is critical for learning a good representation. A sample row from the dataset is shown in Table 1.

Table 1
Example training data with the query, clicked document, associated product, and log click ratio score

Field	Example Value
Query	change color of text
Document	https://HelpX.adobe.com/indesign/using/editing-text.html
Product	Acrobat
Relevance log click ratio	0.24

[1] helpx.adobe.com

We utilize a relevance field that is derived by using the log of the click ratio of the query-document pair. This is important because for a given query, there may be multiple clicked documents and we wish to pay more attention to query-document pairs with more clicks. Since we can show multiple app cards to the user (Figure 1), the applications use the relevance scores to determine which app cards to show and their order. We take the log of the max click ratio to allow less frequently clicked documents to be part of the learning process.

$$relevance = log(\frac{clicks(q_i \longrightarrow d_j)}{max(clicks(q_i \longrightarrow D_i))}) \tag{1}$$

where q_i represents the specific query; d_j represents the particular document clicked; *clicks* represents the number of clicks for the pair; D_i represents the set of documents clicked for q_i, ie $D_i = \{d_i \dots d_n\}$.

HelpX Document Dataset Adobe HelpX documents are curated by Adobe content creators to provide information about Adobe products, from tutorials to product announcements. We utilize this high-quality resource by considering the document title and description as unique query-product training pairs. Since this dataset is curated and high quality, it is given a higher weight during training (relevance = 1).

Product NER Explicit Dataset From user queries in Creative Cloud, we utilize a rule-based product NER to extract query-product pairs containing explicit product names. This dataset allows us to train on a wide set of explicit, high precision intent queries.

Adobe Express Dataset Adobe Express is one of the newer Adobe products and hence has very few user clicks in our behavioral datasets.[2] To bolster additional training data for this product and to learn a good representation for its queries, we utilize top Express in-product queries as part of our dataset.

Finally, we merged the four datasets. Each unique query may have 1 or more products assigned to it. The overall dataset counts are shown in Table 2.

Table 2
Dataset Size: Size is in number of rows of query-document-product-click ratio (see Table 1)

Dataset	Unique Rows
Adobe HelpX Behavioral Dataset	177500
Adobe HelpX Document Dataset	11757
Adobe Express Dataset	6637
Product NER Explicit Dataset	5208

4. Model

We divide the task of learning a good representation of user queries into two parts. The first part is pretraining a language backbone to utilize for downstream finetuning. The second is training the classifier head on top of the language backbone.

[2]We did not add other product-specific datasets since there was already sufficient HelpX data for them and because in-product help-related queries are routed to HelpX.

4.1. Language Model Pretraining

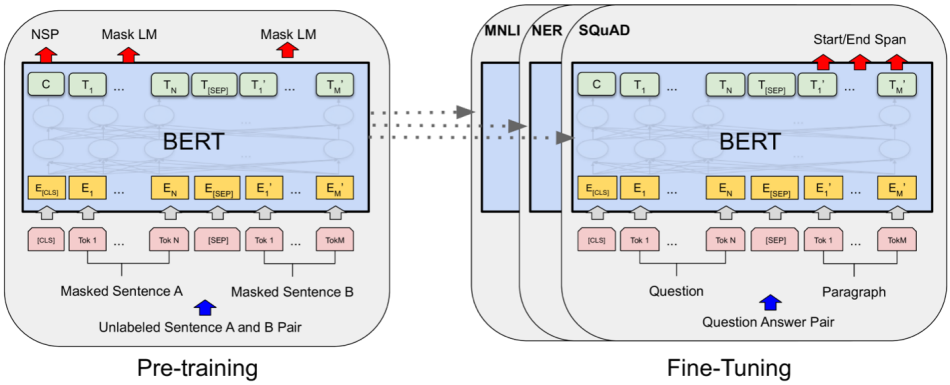

Figure 2: DeBERTa Pretraining: We break HelpX documents into blocks of 128 tokens and pretrain. This allows the LM to understand Adobe product vocabulary and features better.

We found open-source language models (LMs) like BERT [7] to be inadequate for Adobe user queries (see also [6] on training an Adobe-specific language model for semantic search). This is primarily due to two reasons:

1. **Lack of knowledge of Adobe products**: Open-source models are trained on general web data and do not understand the intricacies of Adobe products. Product features such as cropping or generative fill were not understood by the general models. In addition, some product names (e.g. Illustrator, Rush) are also common English words.
2. **Poor product disambiguation**: We found open-source models to have trouble disambiguating products with similar names. Products like Premiere Pro and Premiere Rush or Photoshop and Photoshop Express were lumped together despite being unique products.

To counter this, we pretrained a LM based on Microsoft DeBERTa v3 [8] starting from publicly available pretrained weights on the HelpX document dataset using masked language modeling techniques. We arbitrarily split our Adobe-specific datasets (section 3) into training and validation sets. We trained on block sizes of 128 and found the model to showcase good perplexity. Perplexity in language modeling gauges how well a probability model predicts a sample. See Table 3 for a summary.

Table 3
Training Details

Break HelpX documents into blocks (128 tokens)
Concatenate
Train size: 107240 examples
Validation size: 5645
Trained model perplexity: **7.47**

Pretraining the LM on HelpX data results in a **14%** improvement in downstream classification accuracy compared to using a pretrained LM. This reinforced our hypothesis that domain-

specific workflows such as Adobe help content have a different data distribution than open datasets.

4.2. Classifier Training

Once we had our domain-specific LM backbone, we trained a classifier to predict Adobe products given a user query. We utilized the training datasets described in section 3 for this classifier head. We experimented with freezing the LM backbone (no weights are updated in the LM) and found the best combination to be to freeze the backbone for the initial few epochs and then train the full system for a few additional epochs.

We utilized a classic 2-hidden-layer Multilayer Perceptron network, with a 0.5 dropout rate and a learning rate of 1e−5 and trained the classifier in a multi-label approach, i.e. each product was given a probability score between 0–1 given a query. The multi-label approach is necessary because a large number of implicit product queries are associated with multiple products and even explicit product queries can be associated with multiple products (e.g. *photoshop* is primarily associated with three Photoshop products (web, mobile, and desktop) as well as with Photoshop Express).

We use the Weighted Binary Cross Entropy loss function for our training and leverage the relevancy weights (see equation 1) to pay more attention to more important examples during training.

5. Offline Evaluation and AB Testing

5.1. Quantitative Evaluation on Behavioral Queries

We reserve 10% of our initial dataset (section 3) for evaluation. We compute per-product and per-source metrics. Since the dataset comprises past user queries, it reflects the final product use cases (see section 1 and figure 1). However, it is focused on explicit product mentions since the production app cards are primarily triggered for explicit mentions. Even with explicit product intent, a given query may have multiple products associated with it based in the past user click behavior. Each of these is considered in the quantitative evaluation. As shown in Table 4, precision and recall are well balanced and result in an F1 score of .949. Detailed per-dataset analysis is shown in Table 5. Results on clean and easier datasets like the HelpX document dataset and the explicit NER dataset outperform those on behavioral data. This is because the former two datasets often have the product in the query itself, thus making it easier to predict.

Table 4
Quantitative Evaluation

Quantitative Metrics on the Testset				
Rows	**Precision**	**Recall**	**Accuracy**	**F1**
22849	.961	.941	.970	.949

We also see that the model is robust and can identify products in queries with spelling errors. This includes both small errors (1 edit distance) and large errors (2-3 edit distance). A few examples are shown in Table 6.

Table 5
Evaluation Metrics per Dataset Source

Dataset	Precision	Recall	Accuracy	F1
Product NER Explicit Dataset	.99	.962	.995	.975
Adobe HelpX Document Dataset	.99	.973	.991	.983
Adobe HelpX Behavioral Dataset	.952	.933	.986	.943

Table 6
Queries with Spelling Errors

Query	Product Suggested	Score
creativ cloudd	Creative Cloud	.766
illustator	Illustrator	.991
ilistartor	Illustrator	.933
sparkk	Adobe Express[3]	.952
potosop	Photoshop	.971

5.2. Qualitative Manual Annotation of Implicit Intent

Through quantitative evaluation on the test set (previous subsection), we determined that the model did well when the product was mentioned in the query. We then focused on queries with implicit information about the product, e.g. *keyframe caddy*, *fashion poster*, etc. We utilized a set of 2700 production CC queries for evaluation. These queries were previously unseen by the model. We leveraged Adobe-internal product experts (e.g. product managers for CC) to judge relevancy of the predicted product to the user query. For each query, the model predicts the most likely output(s). Then the product experts mark the suggested output as correct/relevant or incorrect/irrelevant. In the cases where multiple products were predicted, the evaluators were asked to mark the predications as correct/relevant only when all the products predicted were useful. That is, the entire product intent prediction from the model had to be correct, not just a subset of the predictions. Table 7 shows the accuracy results for the 2700 queries in the qualitative evaluation.

Table 7
Qualitative Metrics for Implicit Product Queries: The entire set of intents for a given query must be correct to count as correct.

Qualitative Metrics for Implicit Product Queries			
Rows	**Correct**	**Incorrect**	**Accuracy**
2700	2452	181	**.931**

[3] *Spark* is the original name for Adobe Express.

5.3. AB Testing

We AB-tested the new product intent model for showing app cards in autocomplete and at the top of search results.[4] The new model was tested for all locales against the production model on the CC app and CC web site. Although the LM is trained for English queries, the large number of English queries in non-English locales and the fact that Adobe product names are identical in all languages means that the model triggers app cards in all locales.

As hypothesized, app cards surfaced significantly more with the new model due to triggering on implicit product intent queries and on misspelled queries. Previously, a few fixed queries and key terms triggered app cards, but there was no semantic model to provide broader coverage. The AB test showed a 2-fold increase in surfacing and a >50% decrease in queries with no app cards. We did not expect app cards for all queries since some queries do not have app intent; so, there should always be some queries without app cards surfacing.

Both autocomplete and search result app cards saw an increase in click-through rate for all surfaces and an increase in the unique users who engaged with app cards. Overall, there was a >25% relative improvement in CTR (click-through rate) across the deployed surfaces.[5] These increases reflect the fact that the increased surfacing, e.g. for queries with implicit product intent, was high precision and provided information users needed.

6. Conclusion and Future Work

Accurate product identification is critical for enhancing user experiences, especially at a company like Adobe which has over 50 products. Users on Adobe gateway surfaces such as Adobe.com and Creative Cloud are looking to learn about, license, download, launch, and get help with Adobe products. Given the broad selection of products, new customers often do not know which product they want and so ask implicit product queries around capabilities (e.g. photo editing), while returning users tend to issue explicit product queries. In this work, we present a novel approach to training a product classifier from user behavioral data. Our semantic model led to:

- >25% relative improvement in CTR (click-through rate)
- a >50% decrease in null rate
- a 2x increase in the app cards surfaced, which helps drive product visibility.

As future work, first, we are training a multi-lingual version of the model to better support non-English queries with implicit product intent (e.g. *images gratuites* (French: 'free pictures') which is associated with the Adobe Stock image marketplace). Second, we are experimenting with better long prompt understanding for product disambiguation. This is particularly important for RAG based systems [10, 11] when dealing with retrieval for long prompts. Third, we are using the product intent signal within the ranker for the search results, not just for the product cards and autocomplete.

[4]The app card triggering leverages a hierarchical approach. The product intent model outputs the high-level product (e.g. Photoshop). For products with multiple sub-products and surfaces (e.g. desktop, mobile), the user context is leveraged to determine the most likely sub-product, which is then ranked highest. In the future, we plan to leverage our hierarchical creative intent knowledge graph [9] to directly predict the correct product and sub-product.

[5]We cannot share exact CTRs and so only include relative improvement.

References

[1] T. Luiggi, V. Guigue, L. Soulier, S. Jendoubi, A. Baelde, Dynamic named entity recognition, in: Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing, ACM, 2023, pp. 890–897. URL: https://doi.org/10.1145%2F3555776.3577603. doi:10.1145/3555776.3577603.

[2] I. Yamada, K. Washio, H. Shindo, Y. Matsumoto, Global entity disambiguation with BERT, in: M. Carpuat, M.-C. de Marneffe, I. V. Meza Ruiz (Eds.), Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics, 2022, pp. 3264–3271. URL: https://aclanthology.org/2022.naacl-main.238. doi:10.18653/v1/2022.naacl-main.238.

[3] N. D. Cao, G. Izacard, S. Riedel, F. Petroni, Autoregressive entity retrieval, in: 9th International Conference on Learning Representations ICLR 2021, OpenReview.net, 2021, pp. 3–7. Virtual Event.

[4] Z. Liu, F. Jiang, Y. Hu, C. Shi, P. Fung, NER-BERT: A pre-trained model for low-resource entity tagging, CoRR abs/2112.00405 (2021). URL: https://arxiv.org/abs/2112.00405. arXiv:2112.00405.

[5] P. Nigam, Y. Song, V. Mohan, V. Lakshman, W. Ding, A. Shingavi, C. H. Teo, H. Gu, B. Yin, Semantic product search, in: KDD '19: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2019, pp. 2876–2885.

[6] J. Kumar, A. Gupta, Z. Lu, A. Stefan, T. H. King, Multi-lingual semantic search for domain-specific applications: Adobe Photoshop and Illustrator help search, in: SIGIR '23: Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2023, pp. 3225–3229. URL: https://doi.org/10.1145/3539618.3591826.

[7] J. Devlin, M. Chang, K. Lee, K. Toutanova, BERT: pre-training of deep bidirectional transformers for language understanding, in: Proceedings of NAACL-HLT 2019, ACL, 2019, pp. 4171–4186.

[8] P. He, J. Gao, W. Chen, DeBERTaV3: Improving DeBERTa using ELECTRA-style pre-training with gradient-disentangled embedding sharing, CoRR abs/2111.09543 (2021). URL: https://arxiv.org/abs/2111.09543. arXiv:2111.09543.

[9] S. Sharma, M. Poddar, J. Kumar, K. Blank, T. H. King, Augmenting knowledge graph hierarchies using neural transformers, in: Proceedings of ECIR, 2024, pp. 298–303.

[10] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. Yih, T. Rocktäschel, S. Riedel, D. Kiela, Retrieval-augmented generation for knowledge-intensive NLP tasks, 2021. arXiv:2005.11401.

[11] S. Sharma, D. S. Yoon, F. Dernoncourt, D. Sultania, K. Bagga, M. Zhang, T. Bui, V. Kotte, Retrieval augmented generation for domain-specific question answering, 2024. arXiv:2404.14760, AAAI 2024.