Query2Prod2Vec
Grounded Word Embeddings for eCommerce

Federico Bianchi
Bocconi University
Milano, Italy
f.bianchi@unibocconi.it

Jacopo Tagliabue*
Coveo Labs
New York, USA
jtagliabue@coveo.com

Bingqing Yu
Coveo
Montreal, Canada
cyu2@coveo.com

Abstract

We present **Query2Prod2Vec**, a model that grounds lexical representations for product search in product embeddings: in our model, *meaning* is a mapping between words and a latent space of products in a digital shop. We leverage shopping sessions to learn the underlying space and use merchandising annotations to build lexical analogies for evaluation: our experiments show that our model is more accurate than known techniques from the NLP and IR literature. Finally, we stress the importance of data efficiency for product search outside of retail giants, and highlight how **Query2Prod2Vec** fits with practical constraints faced by most practitioners.

1 Introduction

The eCommerce market reached in recent years an unprecedented scale: in 2020, 3.9 trillion dollars were spent globally in online retail (Cramer-Flood, 2020). While shoppers make significant use of search functionalities, improving their experience is a never-ending quest (Econsultancy, 2020), as outside of few retail giants users complain about sub-optimal performances (Baymard Institute, 2020). As the technology behind the industry increases in sophistication, neural architectures are gradually becoming more common (Tsagkias et al., 2020) and, with them, the need for accurate word embeddings for Information Retrieval (IR) and downstream Natural Language Processing (NLP) tasks (Yu and Tagliabue, 2020; Tagliabue et al., 2020a).

Unfortunately, the success of standard and contextual embeddings from the NLP literature (Mikolov et al., 2013a; Devlin et al., 2019) could not be immediately translated to the product search scenario, due to some peculiar challenges (Bianchi et al., 2020b), such as short text, industry-specific jargon (Bai et al., 2018), low-resource languages; moreover, specific embedding strategies have often been developed in the context of high-traffic websites (Grbovic et al., 2016), which limit their applicability in many practical scenarios. In *this* work, we propose a sample efficient word embedding method for IR in eCommerce, and benchmark it against SOTA models over industry data provided by partnering shops. We summarize our contributions as follows:

1. we propose a method to learn dense representations of words for eCommerce: we name our method **Query2Prod2Vec**, as the mapping between words and the latent space is mediated by the product domain;

2. we evaluate the lexical representations learned by **Query2Prod2Vec** on an analogy task against SOTA models in NLP and IR; benchmarks are run on two independent shops, differing in traffic, industry and catalog size;

3. we detail a procedure to generate synthetic embeddings, which allow us to tackle the "cold start" challenge;

4. we release our implementations, to help the community with the replication of our findings on other shops[1].

While perhaps not fundamental to its industry significance, it is important to remark that grounded lexical learning is well aligned with theoretical considerations on *meaning* in recent (and less recent) literature (Bender and Koller, 2020; Bisk et al., 2020; Montague, 1974).

*Corresponding author. All authors contributed equally and are listed alphabetically.

[1]Public repository available at: https://github.com/coveooss/ecommerce-query-embeddings.

Proceedings of NAACL HLT 2021: IndustryTrack Papers, pages 154–162
June 6–11, 2021.

2 Embeddings for Product Search: an Industry Perspective

In product search, when the shopper issues a query (e.g. "sneakers") on a shop, the shop search engine returns a list of K products matching the query intent and possibly some contextual factor – the shopper at that point may either leave the website, or click on n products to further explore the offering and eventually make a purchase.

Unlike web search, which is exclusively performed at massive scale, product search is a problem that both big and small retailers have to solve: while word embeddings have revolutionized many areas of NLP (Mikolov et al., 2013a), word embeddings for product queries are especially challenging to obtain at scale, when considering the huge variety of use cases in the overall eCommerce industry. In particular, based on industry data and first-hand experience with dozens of shops in our network, we identify four *constraints* for effective word embeddings in eCommerce:

1. **Short text**. Most product queries are very short – 60% of all queries in our dataset are one-word queries, > 80% are two words or less; the advantage of contextualized embeddings may therefore be limited, while lexical vectors are fundamental for downstream NLP tasks (Yu and Tagliabue, 2020; Bianchi et al., 2020a). For this reason, the current work specifically addresses the quality of *word* embeddings[2].

2. **Low-resource languages**. Even shops that have the majority of their traffic on English domain typically have smaller shops in low-resource languages.

3. **Data sparsity**. In *Shop X* below, only 9% of all shopping sessions have a search interaction[3]. Search sparsity, coupled with vertical-specific jargon and the usual long tail of search queries, makes data-hungry models unlikely to succeed for most shops.

4. **Computational capacity**. The majority of the market has the necessity to strike a good trade-off between quality of lexical representations and the cost of training and deploying models, both as hardware expenses and as additional maintenance/training costs.

The embedding strategy we propose – **Query2Prod2Vec** – has been designed to allow efficient learning of word embeddings for product queries. Our findings are useful to a wide range of practitioners: large shops launching in new languages/countries, mid-and-small shops transitioning to dense IR architectures and the raising wave of multi-tenant players[4]: as A.I. providers grow by deploying their solutions on multiple shops, "cold start" scenarios are an important challenge to the viability of their business model.

3 Related Work

The literature on learning representations for lexical items in NLP is vast and growing fast; as an overview of classical methods, Baroni et al. (2014) benchmarks several count-based and neural techniques (Landauer and Dumais, 1997; Mikolov et al., 2013b); recently, context-aware embeddings (Peters et al., 2018; Devlin et al., 2019) have demonstrated state-of-the-art performances in several semantic tasks (Rogers et al., 2020; Nozza et al., 2020), including document-based search (Nogueira et al., 2020), in which target entities are long documents, instead of product (Craswell et al., 2020). To address IR-specific challenges, other embedding strategies have been proposed: *Search2Vec* (Grbovic et al., 2016) uses interactions with ads and pages as context in the typical context-target setting of skip-gram models (Mikolov et al., 2013b); *QueryNGram2Vec* (Bai et al., 2018) additionally learns embeddings for n-grams of word appearing in queries to better cover the long tail. The idea of using vectors (from images) as an aid to query representation has also been suggested as a heuristic device by Yu et al. (2020), in the context of personalized language models; *this* work is the first to our knowledge to benchmark embeddings on lexical semantics (not

[2]Irrespectively of how the lexical vectors are computed, query embeddings can be easily recovered with the usual techniques (e.g. sum or average word embeddings (Yu et al., 2020)): as we mention in the concluding remarks, investigating compositionality is an important part of our overall research agenda.

[3]This is a common trait verified across industries and sizes: among dozens of shops in our network, 30% is the highest *search vs no-search* session ratio; *Shop Y* below is around 29%.

[4]As an indication of the market opportunity, only in 2019 and only in the space of AI-powered search and recommendations, we witnessed Coveo (Techcrunch), Algolia (Techcrunch, 2019a) and Lucidworks (Techcrunch, 2019b) raising more than 100M USD each from venture funds.

tuned for domain-specific tasks), *and* investigate sample efficiency for small-data contexts.

4 Query2Prod2Vec

In **Query2Prod2Vec**, the representation for a query q is built through the representation of the objects that q refers to. Consider a typical shopper-engine interaction in the context of product search: the shopper issues a query, e.g. "shoes", the engine replies with a noisy set of potential referents, e.g. pairs of shoes from the shop inventory, among which the shopper may select relevant items. Hence, this dynamics is reminiscent of a cooperative language game (Lewis, 1969), in which shoppers give noisy feedback to the search engine on the meaning of the queries. A full specification of **Query2Prod2Vec** therefore involves a representation of the target domain of reference (i.e. products in a digital shop) and a denotation function.

4.1 Building a Target Domain

We represent products in a target shop through a *prod2vec* model built with anonymized shopping sessions containing user-product interactions. Embeddings are trained by solving the same optimization problem as in classical *word2vec* (Mikolov et al., 2013a): *word2vec* becomes *prod2vec* by substituting *words* in a *sentence* with *products* viewed in a *shopping session* (Mu et al., 2018). The utility of *prod2vec* is independently justified (Grbovic et al., 2015; Tagliabue and Yu, 2020) and, more importantly, the referential approach leverages the abundance of browsing-based interactions, as compared to search-based interactions: by learning *product* embeddings from abundant behavioral data first, we sidestep a major obstacle to reliable word representation in eCommerce. Hyperparameter optimization follows the guidelines in Bianchi et al. (2020a), with a total of 26,057 (*Shop X*) and 84,575 (*Shop Y*) product embeddings available for downstream processing[5].

4.2 Learning Embeddings

The fundamental intuition of **Query2Prod2Vec** is treating clicks after q as a noisy feedback mapping q to a portion of the latent product space. In particular, we compute the embedding for q by averaging the product embeddings of all products clicked after it, using frequency as a weighting factor (i.e. products clicked often contribute more). The model has one free parameter, *rank*, which controls how many embeddings are used to build the representation for q: if *rank=k*, only the k most clicked products after q are used. The results in Table 1 are obtained with *rank=5*, as we leave to future work to investigate the role of this parameter.

The lack of large-scale search logs in the case of new deployments is a severe issue for successful training. The referential nature of **Query2Prod2Vec** provides a fundamental competitive advantage over models building embeddings from past linguistic behavior *only*, as synthetic embeddings can be generated as long as cheap session data is available to obtain an initial *prod2vec* model. As detailed in the ensuing section, the process happens in two stages, event generation and embeddings creation.

4.3 Creating Synthetic Embeddings

The procedure to create synthetic embeddings is detailed in Algorithm 1: it takes as input a list of words, a pre-defined number of sampling iterations, a popularity distribution over products[6], and it returns a list of synthetic search events, that is, a mapping between words and lists of products "clicked". Simulating the *search* event can be achieved through the existing search engine, as, from a practical standpoint, *some* IR system must already be in place given the use case under consideration. To avoid over-relying on the quality of IR and prove the robustness of the method, all the simulations below are not performed with the actual production API, but with a custom-built inverted index over product meta-data, with a simple TF-IDF weighting and Boolean search.

For the second stage, we can treat the synthetic click events produced by Algorithm 1 as a drop-in replacement for user-generated events – that is, for any query q, we calculate an embedding by averaging the product embeddings of the relevant products, weighted by frequency[7]. Putting the two stages together, **Query2Prod2Vec** can not only produce reliable query embeddings based on historical data, but also learn approximate embeddings for a large vocabulary *before* being exposed

[5]Final parameters for *prod2vec* are: $dimension = 50$, $win\_size = 10$, $iterations = 30$, $ns\_exponent = 0.75$.

[6]Please note that data on product popularity can be easily obtained through marketing tools, such as Google Analytics.

[7]Please note that while *this* work focuses on lexical quality, the same strategy can be applied to complex queries in a "cold start" scenario.

Algorithm 1: Generation of synthetic click events.

Data: a list of words W, a pre-defined number N of simulations per word, a distribution D over products.
Result: A dataset of synthetic clicked events: E

```
E ← empty mapping;
foreach word w in W do
    product_list ← Search(w);
    for i = 1 to N do
        p ← Sample (product_list, D);
        append the entry (w, p) to E;
    end
end
return E
```

to any search interaction: in Section 7 we report the performance of **Query2Prod2Vec** when using only synthetic embeddings[8].

5 Dataset and Baselines

5.1 Dataset

Following best practices in the multi-tenant literature (Tagliabue et al., 2020b), we benchmark all models on different shops to test their robustness. In particular, we obtained catalog data, search logs and anonymized shopping sessions from two partnering shops, *Shop X* and *Shop Y*: *Shop X* is a sport apparel shop with Alexa ranking of approximately *200k*, representing a prototypical shop in the middle of the long tail; *Shop Y* is a home improvement shop with Alexa ranking of approximately *10k*, representing an intermediate size between *Shop X* and public companies in the space. Linguistic data is in Italian for both shops, and training is done on random sessions from the period June-October 2019: after sampling, removal of bot-like sessions and pre-processing, we are left with 722,479 sessions for *Shop X*, and 1,986,452 sessions for *Shop Y*.

5.2 Baselines

We leverage the unique opportunity to join catalog data, search logs and shopping sessions to extensively benchmark **Query2Prod2Vec** against a variety of methods from NLP and IR.

- **Word2Vec and FastText**. We train a CBOW (Mikolov et al., 2013a) and a FastText model (Bojanowski et al., 2017) over product descriptions in the catalog;
- **UmBERTo**. We use RoBERTa trained on Italian data – *UmBERTo*[9]. The $\langle s \rangle$ embedding of the last layer of the architecture is the query embedding;
- **Search2Vec**. We implement the skip-gram model from Grbovic et al. (2016), by feeding the model with sessions composed of search queries and user clicks. Following the original model, we also train a time-sensitive variant, in which time between actions is used to weight query-click pairs differently;
- **Query2Vec**. We implement a different context-target model, inspired by Egg (2019): embeddings are learned by the model when it tries to predict a (purchased or clicked) item starting from a query;
- **QueryNGram2Vec**. We implement the model from Bai et al. (2018). Besides learning representations through a skip-gram model as in Grbovic et al. (2016), the model learns the embeddings of *unigrams* to help cover the long tail for which no direct embedding is available.

To guarantee a fair comparison, all models are trained on the same sessions. For all baselines, we follow the same hyperparameters found in the cited works: the dimension of query embedding vectors is set to 50, except that 768-dimensional vectors are used for **UmBERTo**, as provided by the pre-trained model.

As discussed in Section 1, a distinguishing feature of **Query2Prod2Vec** is *grounding*, that is, the relation between words and an external domain – in this case, products. It is therefore interesting not only to assess a possible *quantitative* gap in the quality of the representations produced by the baseline models, but also to remark the *qualitative* difference at the core of the proposed method: if words are *about* something, pure co-occurrence patterns may be capturing only fragments of lexical meaning (Bianchi et al., 2021).

[8] All the experiments are performed with $N = 500$ simulated search events per query.

[9] https://huggingface.co/Musixmatch/umberto-commoncrawl-cased-v1

6 Solving Analogies in eCommerce

As discussed in Section 2, we consider evaluation tasks focused on *word meaning*, without using product-based similarity (as that would implicitly and unfairly favor referential embeddings). Analogy-based tasks (Mikolov et al., 2013a) are a popular choice to measure semantic accuracy of embeddings, where a model is asked to fill templates like *man : king = woman : ?*; however, preparing analogies for digital shops presents non trivial challenges for human annotators: these would in fact need to know both the language and the underlying space ("air max" is closer to "nike" than to "adidas"), with the additional complication that many candidates may not have "determinate" answers (e.g. if *Adidas* is to *Gazelle*, then *Nike* is to what exactly?). In building our testing framework, we keep the intuition that analogies are an effective way to test for lexical meaning and the assumption that human-level concepts should be our ground truth: in particular, we programmatically produce analogies by leveraging existing human labelling, as indirectly provided by the merchandisers who built product catalogs[10].

6.1 Test Set Preparation

We extract words from the merchandising taxonomy of the target shops, focusing on three most frequent fields in query logs: *product type*, *brand* and *sport activity* for *Shop X*; *product type*, *brand* and *part of the house* for *Shop Y*. Our goal is to go from taxonomy to analogies, that is, showing how for each pair of taxonomy **types** (e.g. *brand : sport*), we can produce two pairs of **tokens** (*Wilson : tennis*, *Cressi : scubadiving*), and create two analogies: *b1 : s1 = b2 : ? (target: s2)* and *b2: s2 = b1 : ? (target: s1)* for testing purposes. For each **type** in a pair (e.g. *brand : sport*), we repeat the following for all possible values of *brand* (e.g. "Wilson", "Nike") – given a brand B:

1. we loop over the catalog and record all values of *sport*, along with their frequency, for the products made by B. For example, for $B = Nike$, the distribution may be: {"soccer": 10, "basketball": 8, "scubadiving": 0 }; for $B = Wilson$, it may be: {"tennis": 8};

2. we calculate the Gini coefficient (Catalano et al., 2009) over the distribution on the values of *sport* and choose a conservative Gini threshold, i.e. $75th$ percentile: the goal of this threshold is to avoid "undetermined" analogies, such as *Adidas : Gazelle = Nike : ?*. The intuition behind the use of a dispersion measure is that product analogies are harder if the relevant label is found across a variety of products[11].

With all the Gini coefficients and a chosen threshold, we are now ready to generate the analogies, by repeating the following for all values of *brand* – given a brand B we can repeat the following sampling process K times ($K = 10$ for our experiments):

1. if B's Gini value for its distribution of *sport* labels is below our chosen threshold, we skip B; if B's value is *above*, we associate to B its most frequent *sport* value, e.g. *Wilson : tennis*. This is the *source* pair of the analogy; to generate a *target* pair, we sample randomly a brand C with high Gini together with its most frequent value, e.g. *Atomic : skiing*;

2. we add to the final test set two analogies: *Wilson : tennis = Atomic : ?*, and *Atomic : skiing = Wilson : ?*.

The procedure is designed to generate test examples conservatively, but of fairly high quality, as for example *Garmin : watches = Arena : bathing cap* (the analogy relates two brands which sell only one type of items), or *racket : tennis = bathing cap : indoor swimming* (the analogy relates "tools" that are needed in two activities). A total of 1208 and 606 test analogies are used for the analogy task (**AT**) for, respectively, *Shop X* and *Shop Y*: we benchmark all models by reporting Hit Rate at different cutoffs (Vasile et al., 2016), and we also report how many analogies are covered by the lexicon learned by the models (*coverage* is the ratio of analogies for which all embeddings are available in the relevant space).

7 Results

Table 1 reports model performance for the chosen cutoffs. **Query2Prod2Vec** (as trained on real

[10] It is important to note that this categorization is done by product experts for navigation and inventory purposes: all product labels are produced independently from any NLP consideration.

[11] In other words, *Wilson : tennis = Atomic : ? (skiing)* is a better analogy than *Adidas : Gazelle = Nike : ?*.

Model	**HR@5,10 for X**	**HR@5,10 for Y**	**CV (X/Y)**	**Acc on ST**
Query2Prod2Vec (real data)	**0.332 / 0.468**	**0.277 / 0.376**	0.965/0.924	**0.88**
Word2Vec	0.206 / 0.242	0.005 / 0.009	0.47 / 0.03	0.68
Query2Vec	0.077 / 0.113	0.065 / 0.120	0.97 / 0.93	0.54
QueryNGram2Vec	0.071 / 0.122	0.148 / 0.216	**0.99** / 0.92	0.82
FastText	0.068 / 0.116	0.010 / 0.012	0.52 / 0.03	0.57
UmBERTo	0.019 / 0.042	0.030 / 0.103	0.99 / **1.00**	0.57
Search2Vec (time)	0.018 / 0.025	0.232 / 0.329	0.23 / 0.90	0.17
Search2Vec	0.016 / 0.024	0.095 / 0.150	0.23 / 0.90	0.17

Table 1: Hit Rate (HR) and coverage (CV) for all models and two shops on **AT**; on the rightmost column, Accuracy (Acc) for all models on **ST**.

data) has the best performance[12], while maintaining a very competitive coverage. More importantly, following our considerations in Section 2, results confirm that producing competitive embeddings on shops with different constraints is a challenging task for existing techniques, as models tend to either rely on specific query distribution (e.g. **Search2Vec (time)**), or the availability of linguistic and catalog resources with good coverage (e.g. **Word2Vec**). **Query2Prod2Vec** is the only model performing with comparable quality in the two scenarios, further strengthening the methodological importance of running benchmarks on more than one shop if findings are to be trusted by a large group of practitioners.

7.1 Sample Efficiency and User Studies

To investigate sample efficiency, we run two further experiments on *Shop X*: first, we run **AT** giving only 1/3 of the original data to **Query2Prod2Vec** (both for the *prod2vec* space, and for the denotation). The small-dataset version of **Query2Prod2Vec** still outperforms all other full-dataset models in Table 1 (*HR@5,10 = 0.276 / 0.380*). Second, we train a **Query2Prod2Vec** model *only with simulated data* produced as explained in Section 4 – that is, with *zero* data from real search logs. The entirely simulated **Query2Prod2Vec** shows performance competitive with the small-dataset version (*HR@5,10 = 0.259 / 0.363*)[13], outperforming all baselines.

As a further independent check, we supplement **AT** with a small semantic similarity task (**ST**) on *Shop X*[14]: two native speakers are asked to solve a small set (46) of manually curated questions in the form: "Given the word *Nike*, which is the most similar, *Adidas* or *Wilson*?". **ST** is meant to (partially) capture how much the embedding spaces align with lexical intuitions of generic speakers, independently of the product search dynamics. Table 1 reports results treating human ratings as ground truth and using cosine similarity on the learned embeddings for all models[15]. **Query2Prod2Vec** outperforms all other methods, further suggesting that the representations learned through referential information capture some aspects of lexical knowledge.

7.2 Computational Requirements

As stressed in Section 2, accuracy and resources form a natural trade-off for industry practitioners. Therefore, it is important to highlight that, our model is not just more accurate, but significantly more efficient to train: the best performing **Query2Prod2Vec** takes 30 minutes (CPU only) to be completed for the larger *Shop Y*, while other competitive models such as **Search2Vec(time)** and **QueryNGram2Vec** require 2 to 4 hours[16]. Being able to quickly generate many models allows for cost-effective analysis and optimization; moreover, infrastructure cost is heavily related to ethical and social issues on energy consumption in NLP (Strubell et al., 2019).

[12] HR@20 was also computed, but omitted for brevity as it confirmed the general trend.

[13] A similar result was obtained on *Shop Y*, and it is omitted for brevity.

[14] *Shop X* is chosen since it is easier to find speakers familiar with sport apparel than DIY items.

[15] Inter-rater agreement was substantial, with *Cohen Kappa Score*=0.67 (McHugh, 2012).

[16] Training is performed on a *Tesla V100 16GB* GPU. As a back of the envelope calculation, training **QueryNGram2Vec** on a *AWS p3 large* instance costs around 12 USD, while a standard CPU container for **Query2Prod2Vec** costs less than 1 USD.

8 Conclusion and Future Work

In *this* work, we learned reference-based word embeddings for product search: **Query2Prod2Vec** significantly outperforms other embedding strategies on lexical tasks, and consistently provides good performance in small-data and zero-data scenarios, with the help of synthetic embeddings. In future work, we will extend our analysis to *i*) specific IR tasks, within the recent paradigm of the dual encoder model (Karpukhin et al., 2020), and *ii*) compositional tasks, trying a systematic replication of the practical success obtained by Yu et al. (2020) through image-based heuristics.

When looking at models like **Query2Prod2Vec** in the larger industry landscape, we hope our methodology can help the field broaden its horizons: while retail giants indubitably played a major role in moving eCommerce use cases to the center of NLP research, finding solutions that address a larger portion of the market is not just practically important, but also an exciting agenda of its own[17].

9 Ethical Considerations

Coveo collects anonymized user data when providing its business services in full compliance with existing legislation (e.g. GDPR). The training dataset used for all models employs anonymous UUIDs to label events and sessions and, as such, it does not contain any information that can be linked to shoppers or physical entities; in particular, data is ingested through a standardized client-side integration, as specified in our public protocol.

Acknowledgements

We wish to thank Nadia Labai, Patrick John Chia, Andrea Polonioli, Ciro Greco and three anonymous reviewers for helpful comments on previous versions of this article. The authors wish to thank *Coveo* for the support and the computational resources used for the project. Federico Bianchi is a member of the Bocconi Institute for Data Science and Analytics (BIDSA) and the Data and Marketing Insights (DMI) unit.

[17] Please note that a previous draft of this article appeared on *arxiv* – https://arxiv.org/abs/2104.02061 – *after* the review process, but before the camera-ready submission.

References

Xiao Bai, Erik Ordentlich, Yuanyuan Zhang, Andy Feng, Adwait Ratnaparkhi, Reena Somvanshi, and Aldi Tjahjadi. 2018. Scalable query n-gram embedding for improving matching and relevance in sponsored search. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, pages 52–61, New York, NY, USA. ACM.

Marco Baroni, Georgiana Dinu, and Germán Kruszewski. 2014. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 238–247, Baltimore, Maryland. Association for Computational Linguistics.

Baymard Institute. 2020. *Site Search for Ecommerce.*

Emily M. Bender and Alexander Koller. 2020. Climbing towards NLU: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5185–5198, Online. Association for Computational Linguistics.

Federico Bianchi, Ciro Greco, and Jacopo Tagliabue. 2021. Language in a (Search) Box: Grounding Language Learning in Real-World Human-Machine Interaction. In *NAACL-HLT*. Association for Computational Linguistics.

Federico Bianchi, Jacopo Tagliabue, Bingqing Yu, Luca Bigon, and Ciro Greco. 2020a. Fantastic embeddings and how to align them: Zero-shot inference in a multi-shop scenario. In *Proceedings of the SIGIR 2020 eCom workshop*.

Federico Bianchi, Bingqing Yu, and Jacopo Tagliabue. 2020b. Bert goes shopping: Comparing distributional models for product representations. *arXiv preprint arXiv:2012.09807*.

Yonatan Bisk, Ari Holtzman, Jesse Thomason, Jacob Andreas, Yoshua Bengio, Joyce Chai, Mirella Lapata, Angeliki Lazaridou, Jonathan May, Aleksandr Nisnevich, Nicolas Pinto, and Joseph Turian. 2020. Experience grounds language. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8718–8735, Online. Association for Computational Linguistics.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Michael Catalano, Tanya Leise, and Thomas Pfaff. 2009. Measuring resource inequality: The gini coefficient. *Numeracy*, 2.

Ethan Cramer-Flood. 2020. *Global Ecommerce 2020. Ecommerce Decelerates amid Global Retail Contraction but Remains a Bright Spot.*

Nick Craswell, Bhaskar Mitra, E. Yilmaz, Daniel Fernando Campos, and E. Voorhees. 2020. Overview of the trec 2019 deep learning track. *ArXiv*, abs/2003.07820.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Econsultancy. 2020. *Site search: retailers still have a lot to learn.*

Alex Egg. 2019. *Query2vec: Search query expansion with query embeddings*.

Mihajlo Grbovic, Nemanja Djuric, Vladan Radosavljevic, Fabrizio Silvestri, Ricardo Baeza-Yates, Andrew Feng, Erik Ordentlich, Lee Yang, and Gavin Owens. 2016. Scalable semantic matching of queries to ads in sponsored search advertising. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '16, page 375–384, New York, NY, USA. Association for Computing Machinery.

Mihajlo Grbovic, Vladan Radosavljevic, Nemanja Djuric, Narayan Bhamidipati, Jaikit Savla, Varun Bhagwan, and Doug Sharp. 2015. E-commerce in your inbox: Product recommendations at scale. In *Proceedings of KDD '15*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Thomas K. Landauer and Susan T. Dumais. 1997. A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge.

David Lewis. 1969. Convention. *Mass.: Harvard UP*.

Mary McHugh. 2012. Interrater reliability: The kappa statistic. *Biochemia medica : časopis Hrvatskoga društva medicinskih biokemičara / HDMB*, 22:276–82.

Tomas Mikolov, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, page 3111–3119, Red Hook, NY, USA. Curran Associates Inc.

Richard Montague. 1974. English as a formal language. In Richmond H. Thomason, editor, *Formal Philosophy: Selected Papers of Richard Montague*, pages 188–222. Yale University Press, New Haven, London.

Cun Mu, Guang Yang, and Zheng Yan. 2018. Revisiting skip-gram negative sampling model with regularization. *CoRR*, abs/1804.00306.

Rodrigo Nogueira, Zhiying Jiang, and Jimmy Lin. 2020. Document ranking with a pretrained sequence-to-sequence model. In *EMNLP*.

Debora Nozza, Federico Bianchi, and Dirk Hovy. 2020. What the [MASK]? making sense of language-specific BERT models. *arXiv preprint arXiv:2003.02912*.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in nlp. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650.

Jacopo Tagliabue and Bingqing Yu. 2020. Shopping in the multiverse: A counterfactual approach to in-session attribution. In *Proceedings of the SIGIR 2020 Workshop on eCommerce (ECOM 20)*.

Jacopo Tagliabue, Bingqing Yu, and Marie Beaulieu. 2020a. How to grow a (product) tree: Personalized category suggestions for eCommerce type-ahead. In *Proceedings of The 3rd Workshop on e-Commerce and NLP*, pages 7–18, Seattle, WA, USA. Association for Computational Linguistics.

Jacopo Tagliabue, Bingqing Yu, and Federico Bianchi. 2020b. The embeddings that came in from the cold: Improving vectors for new and rare products with content-based inference. In *Fourteenth ACM Conference on Recommender Systems*, RecSys '20, page 577–578, New York, NY, USA. Association for Computing Machinery.

Techcrunch. coveo-raises-227m-at-1b-valuation.

Techcrunch. 2019a. Algolia finds $110m from accel and salesforce.

Techcrunch. 2019b. Lucidworks raises $100m to expand in ai finds.

Manos Tsagkias, Tracy Holloway King, Surya Kallumadi, Vanessa Murdock, and Maarten de Rijke. 2020. Challenges and research opportunities in ecommerce search and recommendations. In *SIGIR Forum*, volume 54.

Flavian Vasile, Elena Smirnova, and Alexis Conneau. 2016. Meta-prod2vec: Product embeddings using side-information for recommendation. *Proceedings of the 10th ACM Conference on Recommender Systems*.

Bingqing Yu and Jacopo Tagliabue. 2020. Blending search and discovery: Tag-based query refinement with contextual reinforcement learning. In *EComNLP*.

Bingqing Yu, Jacopo Tagliabue, Ciro Greco, and Federico Bianchi. 2020. An image is worth a thousand features: Scalable product representations for in-session type-ahead personalization. *Companion Proceedings of the Web Conference 2020*.