The Difference Between a Click and a Cart-Add: Learning Interaction-Specific Embeddings

Xiaoting Zhao*, Raphael Louca*, Diane Hu
Etsy, Inc
New York, U.S.A
{xzhao,rlouca,dhu}@etsy.com

Liangjie Hong**
hongliangjie@gmail.com

ABSTRACT

For large-scale online marketplaces with over millions of items, users come to rely on personalized recommendations to find relevant items from their massive inventory. One hallmark of the shopping experience in such online marketplaces is the many ways a user can interact with an item they are interested in: they can click it, favorite it, add it to cart, purchase it, etc. We hypothesize that the different ways in which a user interacts with an item indicates different kinds of intent. Consequently, a user's recommendations should be based not only on items from their past activity, but also the way in which they interacted with these items. Co-occurrence based methods have been successfully used to give recommendations that incorporate interaction types, such as the popular "Because you purchased X, you may also purchase Y". In this paper, we propose a novel method that generalizes upon the co-occurrence methods to learn interaction-based item embeddings that encode the co-occurrence patterns of not only the item itself, but also the interaction type. The learned embeddings provide a convenient way of approximating the likelihood that one item-interaction pair would co-occur with another by way of a simple inner product. To show their effectiveness, we deploy the interaction-based embeddings in an industry-scale recommendation system that serves live traffic on Etsy.com. We find that taking interaction types into account shows significant improvements in accurately modeling user shopping behavior for both online and offline metrics.

KEYWORDS

Embeddings, Candidate Set Selection, Recommendation

ACM Reference Format:
Xiaoting Zhao*, Raphael Louca*, Diane Hu and Liangjie Hong**. 2020. The Difference Between a Click and a Cart-Add: Learning Interaction-Specific Embeddings. In *Companion Proceedings of the Web Conference 2020 (WWW '20 Companion), April 20–24, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3366424.3386197

1 INTRODUCTION

As online shopping becomes more prevalent and inventory grows at an exponential scale, customers have come to rely on personalized recommendation systems for discovering items that are relevant to them. One hallmark of the shopping experience in online marketplaces is the multitude of ways in which a user can interact with an item they are interested in: they can click it, favorite it, add it to a collection, add it to cart, or purchase it. We hypothesize that the different ways in which a user interacts with an item indicate different kinds of intent. For example, a user who clicks an item must have different intent than a user who adds the same item to their cart. Thus, the two users should be shown different recommendations, despite the fact that they both interacted with the same item. Figure 1 shows several examples of target items, with potential recommendations for the user who *clicked* that item versus for the user who *carted* that item. Not only are the recommendations different, but the first shows recommendations that look more like substitutes to the target item, while the second shows recommendations that are more complementary.

Using co-occurrence based models [13, 14] has been efficient and effective in modeling different interaction types. The underlying concept assumes that if a pair of items has been clicked or purchased together within a short amount of time by the same user, there is a good chance the two items are related. However, this method (1) does not holistically consider the different ways in which a customer can interact with items, and (2) requires that items explicitly co-occur together, leading to sparsity issues.

In this paper, we propose a novel method that learns interaction-based item embeddings that not only encode the co-occurrence pattern, but also the way in which a user interacts with them. Our proposed method generalizes beyond explicit co-occurrence counts, with the ability to give recommendations along the lines of "Because you X this, you may also want to Y that", where X and Y are any interaction types. In this way, we address sparsity issues while having the ability to provide more interpretable recommendations.

Furthermore, in contrast to previous applications of embedding models, we learn multiple embeddings for each item, one for every possible interaction. These learned embeddings give us a convenient way of approximating the likelihood that one item-interaction pair would co-occur with another during a shopping session by a simple inner product. As such, we can predict not only *which* items a user may be interested in, but also *how* they will interact with them.

While the learned embeddings may be useful for many downstream applications, we choose to evaluate our model as a new candidate set selection method in many production recommendation systems that serve live traffic at Etsy. In both offline and online experiments, a significant increase is shown over competitive baseline methods, including key business KPIs such as conversion rate.

Compared to previous work on embeddings and co-occurrence models, our proposed method makes the following contributions:

*These authors have equally contributed to this work.
**The work was conducted while the author was at Etsy.

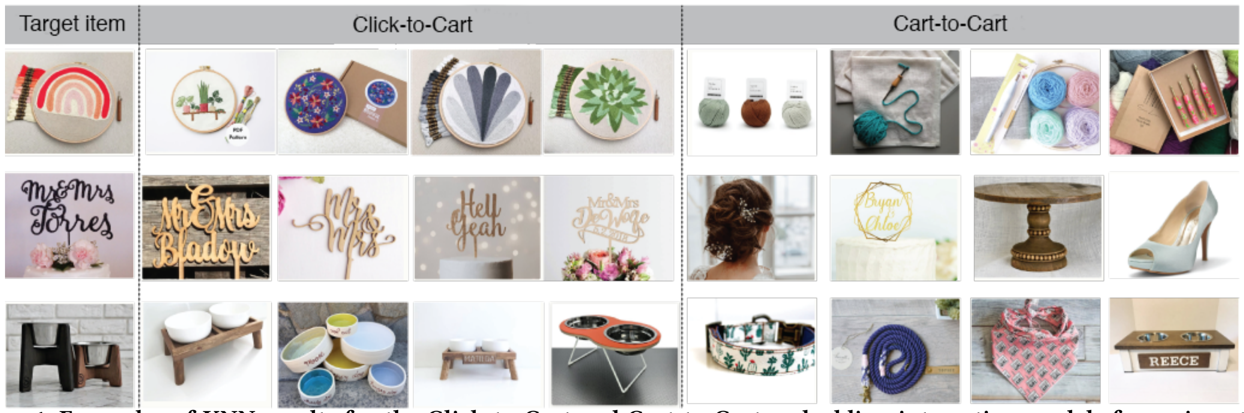

Figure 1: Examples of KNN results for the Click-to-Cart and Cart-to-Cart embedding-interaction models for a given target item. One can see that the candidate set for a users who clicked the target item are different that the recommendations for a user who added the target item to their shopping cart. In particular, we observe that the Click-to-Cart embedding interaction model returns items that are substitutes whereas the Cart-to-Cart model returns more complementary items.

- Learns multiple embeddings for each listing to model different user intents, expressed through the different ways in which a user interacts with a listing.
- Generalizes upon co-occurrence based models for accurately predicting which listing a user may want to interact with next without falling prey to sparsity. The model incorporates a co-occurrence regularization term that guarantees equal or better performance than baseline co-occurrence models.
- Captures co-occurrence patterns in a low-dimensional representation that makes it computationally efficient to compare any two items, allowing for use as a feature or candidate set selection method in production recommender systems.
- Provides more interpretable recommendations based on the interaction type of a customer's previous activity.

In the following, we describe related lines of work (Section 2) and motivation (Section 3), introduce the proposed model (Section 4), and discuss offline and online experiment results (Section 5).

2 RELATED WORK

There are two broad areas of research that closely relate to our line of work: (1) neural language models that learn embeddings and (2) the application of embeddings and co-occurrence models in production recommender systems.

In its original form, neural language models such as continuous bag-of-words (CBOW) and skip-gram (SG) learn semantically meaningful, low-dimensional representation of words by modeling patterns of co-occurring words in sentences [17]. In recent years, extending these neural embedding models to applications outside of the NLP domain has been gaining in popularity, making appearances in many domains including search, recommendations, and e-commerce [2, 8, 9, 12, 18]. Just as word embeddings can be trained by treating a sequence of words in a sentence as context, item embeddings can be trained in a similar fashion by treating a sequence of user actions as context and learning low-dimensional embeddings for each item. In [8], the authors develop item and user embedding models that are used in search ranking and similar item recommendation modules at *Airbnb*. The embedding models they describe incorporate explicit negative signals and global context in order to improve the accuracy of low-dimension user and item representations. In a similar line of work, Grbovic et al. [9] propose the *prod2vec* and *bagged-prod2vec* methods to deliver personalized product advertisements to *Yahoo Mail* users. Lastly, in [18] the authors leverage embeddings for a context-aware music recommendation system.

One primary reason for the rise of embedding model usage in industry-scale recommender systems is due to its ability to quickly quantify relevance between two items simply by taking the inner product of its embeddings. As such, embeddings are often used in a *top-k* recommender system setting [6, 8, 10], or as the first stage of retrieval in a two-pass re-ranking system [1, 4, 5, 7, 15]. In both cases, the goal is to quickly retrieve a small set of relevant items out of a large candidate set (i.e., in the range of hundreds of millions).

In addition to embedding models, many earlier works also depend on IR-based heuristic for quickly determining the goodness of match between the user and items based on product category or taxonomy matching schemes, or simple popularity rankings. Beyond basic heuristics, co-occurring signals have been a popular method for simple and efficient candidate set selection. Amazon tracks pairs of items that are frequently co-purchased by the same customers and constructs candidate sets by retrieving items that have been frequently co-purchased with a customer's last clicked items [13]. Pinterest's related pin recommendations system selects candidate pins based on *board co-occurrence*, the number of times a pin has been pinned to the same board [14].

While our recommender system tasks is similar, we depart from past work by proposing a novel embedding that distinguishes between different interaction types and explicitly incorporates co-occurrence information. This allows us to leverage the power of sequence-based embeddings and co-occurrence signals, while still maintaining a single low-dimensional representation (but for each item-interaction pair) that allows it to be used in industry-scale recommender systems. A recent flurry of work has focused on encoding heterogeneous user-behavior via sequence-based or attention-based modeling [19] to predict what to recommend next based on a user's recent interactions. That approach is predominantly used in the second-pass reranking stage, as it can mostly be used to rank

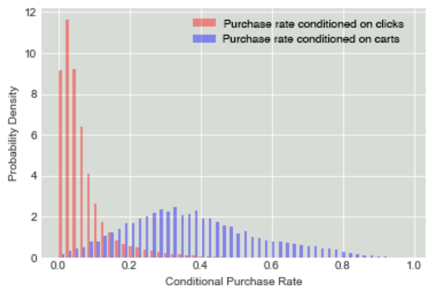

Figure 2: Probability density function estimated for the purchase rate conditioned on clicks and cart-adds, respectively.

candidate sets whose size is restricted to several hundred items. Our approach is different in that it is designed to quickly generate candidate sets for hundreds of millions of users and items; therefore it can be efficiently deployed to large-scale production systems.

3 MOTIVATION

In this section, we motivate our decision for explicitly incorporating interaction types into our embedding model and discuss potential applications of our item-interaction embeddings in a typical two-pass production recommender system.

3.1 Interaction Types

The majority of buyers land on an item's product page either through search or by interacting with a recommendations module. After finding and exploring an item, the user decides whether to go back to discovery mode or to purchase the item by adding it to their shopping cart. In Figure 2, we show probability density functions for purchase rate conditioned on item clicks versus cart-adds across all items over the period of 6 months. These are estimated using relevant data across user sessions. As shown, these two interaction types exhibit very distinct distributions of purchase intent on the items: the one conditioned on clicks is heavily tailed with a lower purchase desire while, the one conditioned on cart-adds is more symmetric and exhibits a much higher buying intention with larger variance. Thus, we argue that in addition to the identity of the item, it is just as important to incorporate the way in which a user interacts with it in our model. Furthermore, by incorporating interaction types we hope to make a user's shopping journey more efficient by effectively recommending substitutable and/or complementary items in their *learn* and *decide* browsing experience.

3.2 Two-Pass Ranking

Our modeling of a user intent must consider the constraints of building a production-ready recommender system. As discussed in Section 2, most large-scale recommendation system in production typically consists of a two-pass ranking: First, a candidate set selection method that can quickly retrieve a small set of relevant items out of a larger set (potentially in the range of hundreds of millions) of candidate items. During this stage, it is crucial to quickly prune irrelevant items while retrieving items that are likely to be relevant to the user at low computational cost as it is often necessary to apply it to over billions of user and item pairs. Top-k style recommender systems directly use the score from this retrieval step and provide the top k items as recommendations. However, for *two-pass* recommender systems, a second ranker is deployed to re-rank the candidate items using a more sophisticated machine learning model and features that are likely to achieve fine-grained ranking toward a specific metric (such as purchase or click optimization). As such, these two stages can be being considered with the problem of (1) retrieval, followed by (2) precision.

Our motivation for capturing interaction type as well as co-occurrence information in a low-dimensional representation comes from the desire to improve the accuracy of next item recommendation not only in our second pass ranker, but also in our first pass ranker. The learned embeddings give us a convenient way of approximating the likelihood that one item-interaction pair will co-occur with another by way of a simple inner product, and naturally lend themselves for use in a top-k ranking system or a candidate set selection method for two-pass recommendation system, as we will show in Section 5.4.

4 PROPOSED METHOD

In the following sections, we describe a model that allows us to efficiently encode the ways in which users interact with items, and their co-occurrence relationships with other similar items. We show how to learn multiple interaction-based embeddings for each item, generate candidate sets, and produce recommendations for users.

4.1 Item-Interaction Embeddings

The same method that originated in the NLP domain can be used to model users' journeys in aggregate as a sequence of user engagements on listings, i.e., clicked, favored, add-to-carted or purchased. Each user session is analogous to a sentence, and each user action on a listing is analogous to a word in NLP word2vec parlance. This method of modeling interactions allows us to represent items or other entities (i.e., shops, users, queries) as low dimensional continuous vectors, where the similarity across two different vectors represents their co-relatedness. Semantic embeddings are agnostic to the content of items such as their titles, tags, descriptions, and allow us to leverage aggregate user interactions on the site to extract items that are semantically similar.

Let $\mathcal{L}$ denote the set of items and $\mathcal{I}$ the set of interaction types a user with an item (e.g., click, favorite, add-to-cart, purchase). Each user's visit defines a session $S = \{p_1, \dots, p_k\}$, which consists of a sequence of item-interaction pairs $p_j \in \mathcal{L} \times \mathcal{I}$. For example, the sequence $(\ell_1, \text{click}), (\ell_2, \text{click}), (\ell_2, \text{add-to-cart}), (\ell_2, \text{purchase})$ specifies that a user first clicked item ℓ_1 and then clicked item ℓ_2; the user then added item ℓ_2 to his shopping cart, and eventually purchased it. The training data consists of such sequences collected from multiple users over a set period of time.

The skip-gram model uses a single item-interaction pair to predict the output of $2m$ neighboring pairs, where m is a hyper-parameter of the model and could also be drawn from a discrete uniform distribution to favor the shorter distance of contexts to the target item. Then, given an item-interaction pair $p_i \in \mathcal{L} \times \mathcal{I}$, the probability of observing the pair p_{i+j} is given by

$$\mathbb{P}(p_{i+j} \mid p_i) = \exp\left(u_{p_i}^\top v_{p_{i+j}}\right) \Big/ \sum_{p=1}^{n} \exp(u_{p_i}^\top v_p) \text{ with } |m| \le j, \quad (1)$$

where $u_p, v_p \in \mathbf{R}^d$ are the input and output representations of the pair p, respectively, and n is the number of unique item-interaction pairs. Implicit in the skip-gram model is the assumption that the dimension $d << n$. The objective is to maximize the function

$$\sum_{S \in \mathcal{S}} \sum_{p_i \in S} \sum_{-m \leq j \leq m} \mathbb{P}(p_{i+j} \mid p_i), \tag{2}$$

where $\mathcal{S}$ denotes the set of all sessions in the training data. It follows from (1) and (2) that item-interaction pairs with similar contexts will have "similar" low-dimensional vector representations.

The objective function is optimized using stochastic gradient ascent. In practice, however, the computation of an optimal solution, can be computationally cumbersome because the size of the ambient dimensional space $|\mathcal{L}| \times |\mathcal{I}|$ can be prohibitively large. To account for this, we use the negative sampling approach proposed in [17].

More precisely, let $\mathcal{P}_p \subseteq \mathcal{L} \times \mathcal{I}$ and $\mathcal{N}_p \subseteq \mathcal{L} \times \mathcal{I}$ denote the set of positive and negative samples associated with the item-interaction pair p, respectively. The negative samples $q \in \mathcal{N}_p$ are drawn uniformly at random from the set of all item-interaction pairs $\mathcal{L} \times \mathcal{I}$. The following optimization problem describes the skip-gram model with negative sampling.

$$\underset{u_p, v_q \in \mathbf{R}^d}{\operatorname{argmax}} \sum_{p \in \mathcal{L} \times \mathcal{I}} \left[\sum_{q \in \mathcal{P}_p} \log \sigma(u_p^\top v_q) + \sum_{q \in \mathcal{N}_p} \log(1 - \sigma(u_p^\top v_q)) \right],$$

where $\sigma : \mathbf{R} \to [0, 1]$ is the sigmoid function. In [8], the authors observe that random negative sampling can result in sub-optimal within market similarities. This arises because positive samples, unlike negative, mostly come from items from the same market, thus creating an imbalance.

Co-Occurrence Based Regularization: In this paper, we introduce a regularization term whose objective is to bring item-interaction pairs that co-occur frequently *across all sessions* closer in space, encoding *intra-session* dependency in addition to regular *inter-session* similarity. More precisely, the difference between the first term in the above optimization problem and the proposed regularization term is that the former contains only positive samples that occur *in the same session* (i.e., to encode *inter-session* similarity), while the latter contains positive samples that occur *across all sessions* (i.e., for *intra-session* dependency) per interaction type.

For each item-interaction pair p, we compute all other item-interaction pairs that co-occurred with p across all sessions $S \in \mathcal{S}$ and use those as positive samples. Let $\mathcal{R}_p \subseteq \mathcal{L} \times \mathcal{I}$ denote the set of all such item-interaction pairs. For example, if $p = (\ell_1, \text{click})$ and $q = (\ell_2, \text{purchase}) \in \mathcal{R}_p$, then there must exist at least one session S, such that item ℓ_2 was purchased after item ℓ_1 was clicked. The directional co-occurrence pairs are explicitly encoded in order to capture purchases intention. Finally, the following optimization problem describes the skip-gram model with negative sampling and co-occurrence based regularization.

$$\underset{u_p, v_q \in \mathbf{R}^d}{\operatorname{argmax}} \sum_{p \in \mathcal{L} \times \mathcal{I}} \left[\sum_{q \in \mathcal{P}_p} \log \sigma(u_p^\top v_q) + \sum_{q \in \mathcal{N}_p} \log(1 - \sigma(u_p^\top v_q)) + \sum_{q \in \mathcal{R}_p} \log \sigma(u_p^\top v_q) \right]. \tag{3}$$

In general, the cardinality of the set $\mathcal{R}_p$ can be large resulting in computational inefficiencies. We provide more details about the implementation of our approach in section 5.1.2.

Intuition: The learned embeddings provide a convenient way of encoding co-occurrence patterns between items and the way users interact with them. A nice property is that the inner product between two such embeddings should approximate the likelihood that one item-interaction pair would co-occur with another. For example, to answer a question such as "since a user clicked on item A, what is an item they may add to cart next?", we can simply find the nearest "cart" item embeddings to item A's "click" embedding.

5 EXPERIMENTS

In this section, we investigate the strengths and limitations of the proposed approach and discuss evaluation results on a dataset of browsing sessions from an e-commerce website.

5.1 Dataset

The training data we use spans a one year period of visit logs and it consists of implicit feedback collected from users visiting during that period. In particular, each training instance is defined by a user's session and consists of a sequence of item-interaction pairs sorted in a chronological order. We restrict attention to sessions which have more than three item-interaction pairs and at least one purchase to eliminate bounces. The resulting dataset has about half a billion item-interaction pairs from over 100 millions distinct pairs.

5.1.1 Negative Sampling and Global Context. As discussed in section 4.1, the model we propose uses the negative sampling approach introduced in [17] to facilitate computational efficiency in training and improve the quality of the low-dimensional vector representations. In particular, we use random negative sampling, where we take five random samples from the set of all item-interaction pairs. We also tried taxonomy-based, shop-based, and TFIDF-similarity based negative sampling, but found that the approach in [17] worked best. We also did not observe a significant increase in offline metrics with global context defined in [8].

5.1.2 Co-Occurrence Based Regularization. Our model incorporates the co-occurrence based regularization introduced in section 4. As discussed, the size of the co-occurrence set $\mathcal{R}_p$ can be prohibitively large, making training computationally cumbersome. For this reason, for each item-interaction pair we only take the k highest co-occurring items for each interaction we consider. Beyond computational reasons this choice is made to avoid overshadowing in-session information with historical information. In our offline experiments, we experimented with different values of k, but found that $k = 3$ had the highest performance across offline metrics.

5.2 Implementation Details

In order to support the co-occurrence based regularization term proposed, we used the *fastText* library [3] in C++ from Facebook's AI research lab to train our embeddings with extended functionalities we built on top of the existing library. *fastText* enriches the word2vec model [17] with subword information by adding a bag of character ngrams. We chose this framework in favor of its easiness in extensibility as well as its efficiency in training. We experimented with tuning several hyper-parameters of the model and eventually chose to set the context window to $m = 5$ and the embedding dimension to $d = 200$ (cf. section 4.1). In addition to the aforementioned co-occurrence appended tokens as regularization, we also added five random negative samples in each of our sequences.

After training, we use the approximate K-nearest neighbor search algorithm, *Hierarchical Navigable Small World* [16] in C++ from Faiss (HNSW) [11], in order to get the top k similar item-interaction pairs for each learned item-interaction pair. To balance efficiency with accuracy, we set the following hyper-parameters in HNSW (*efSearch* = 256, *efConstruction* = 128, *linksPerVector* = 64) to scale up the approximated neighborhood search over hundreds of millions item-interaction pairs from our embedding training.

5.3 Offline Experiments

We decide to evaluate our proposed embeddings as a new candidate set selection method in our production recommender system, as we reasoned that this first stage of ranking would be most effectively impacted. We use the following offline set-up to evaluate our embeddings as a valid candidate set selection method using data collected from shopping sessions that start in the 24 hour window following the last day of training.

5.3.1 Evaluation Methodology. We evaluate our candidate set selection method by computing a *purchase intent hit rate* – a metric similar to recall – for the generated candidate sets. Given a candidate set S and a multiset of items P that where either purchased or added to cart (in the test set), the hit rate is equal to $|P \cap S|/|P|$.

We evaluate hit rates on two broad families of candidate sets. The first is based on listing representations that are agnostic to interactions. Henceforth, we refer to these candidate sets as *no-interaction* candidate sets. The second is based on interaction-based item representations. For the latter family, we compute two distinct candidate sets. The first is named *Click-to-Cart* and is constructed by finding the 200 nearest item embeddings having interaction equal to *cart* to each item's *click* embedding. The second is named *Cart-to-Cart* and is obtained similarly. For our experiments, we restrict our attention to click interactions to capture "browse" intent and cart interactions to capture "purchase" intent. Although feasible to extend to all possible interactions (i.e., click, favor, cart-add, purchase) in KNN tasks, we have chosen such reduction when deployed to the production for engineering costs. We remark that all purchase interactions are a superset of cart interactions and all favorite interactions are a superset of click interactions.

We use a KNN algorithm to construct these candidate sets and compare them against several baselines, which we describe below:

No-Interaction based Baselines:

- **No-Interaction Embedding:** This method learns a single embedding for each item without differentiating between interaction types, similar to [8]. The candidate set is generated by finding the nearest 200 items to each target item.
- **No-Interaction Embedding with Regularization:** Similar to the *No-Interaction Embedding*, but with added co-occurrence regularization as defined in (3).
- **Matrix Factorization:** This computes low-dim representations for items and users based on their all-time favorites.
- **Co-Purchase:** This method represents each item as a *co-purchase vector*, a sparse vector indicating other items that have been co-purchased with it, similar to [13]. The candidate set is generated by finding the top K items with the most similar co-purchase vectors as the target item. We chose this as a baseline because it is the dominant candidate set selection method currently used in our production system.

	No-Interaction	**Interaction**	
	Item-to-Item	Click-to-Cart	Cart-to-Cart
Reg. Embedding	8.55%	12.00%	8.36%
Embedding	7.98%	10.78%	7.54%
Matrix Fact.	0.69%	1.77%	1.28%
Co-Occurrence	6.27%	10.00%	4.77%

Table 1: Hit rates of the co-occurrence regularized interaction and no-interaction embeddings against baselines. Hit rate is a proxy for purchase probability.

Interaction-based Baselines:

- **Interaction-Based Matrix Factorization:** To incorporate the interaction type, the $(i, j)^{\text{th}}$ entry of the matrix is equal to one if user i interacted with the j^{th} item-interaction pair. The candidate set for each item-interaction pair is computed by first fixing an interaction and then finding the nearest 200 items to the item-interaction pair having said interaction.
- **Co-Occurrence:** This computes candidate sets based on items that co-occur within a single visit, and is a natural extension of the *Co-Purchase* model described above as it extends to include other interaction types. The candidate set of each item-interaction pair is aggregated over co-occurred item-interaction pairs from all sessions we consider.

5.3.2 Offline Experiment: Accuracy by Hit Rate. Table 1 depicts the average hit rate for the no-interaction and interaction based models. In particular, we observe that the co-occurrence regularized embedding model, which does not incorporate interactions, outperforms all other no-interaction baselines. Similarly, we observe that the regularized interaction embedding model outperforms all other interaction baselines, both for the Click-to-Cart and the Cart-to-Cart candidate sets. Therefore, the regularization term we incorporated in our skipgram model is able to identify similarities among items that help predict purchase intent. In addition, when comparing among the four embedding models, we observe that the no-interaction embedding model has the smallest average hit rate among all other models. Therefore, by incorporating interaction types in our model we obtain candidate sets, which contain more items that are eventually purchased. A similar pattern is observed when we compare the co-occurrence model against the co-purchase model as well as interaction-based matrix factorization to its no-interaction counterpart.

5.3.3 Offline Experiment: Coverage. Candidate sets relying on historical co-purchase have low coverage rates. For example, the *co-purchase* baseline covers lower than 10% of active items. This is due to stringent requirements for constructing such candidate sets. For example, the co-purchase candidate set requires two or more items to be purchased within a small time window. Such criteria is hardly applicable for the majority of items sold at the online marketplace we consider at Etsy due to their one-of-a-kind nature. Low coverage is also evident in the size of the candidate set for the co-purchase based candidate set (approximately 40 items). Compared to the *co-purchase* baseline, the item-interaction and no-interaction methods

	Average Cosine Similary	
	Click Embedding	Cart Embedding
Same Taxonomy	0.54	0.66
Diff. Taxonomy	0.43	0.58

Table 2: Cosine similarity between embedding vectors (Co-Occurrence Based Interaction model) sampled from the same and different taxonomies for each interaction type.

Target Item Interaction	Candidate Item Interaction Click	Cart
Click	9.20%	12.00%
Cart	4.36%	8.36%

Table 3: Hit rates for the co-occurrence regularized embedding-interaction model.

cover at least 70% of distinct active items, which account for more than 80% of website's traffic. In addition, the size of the candidate set for all embedding models can be as large as we desire since it relies on KNN algorithms. To consider a full higher coverage rate in production, we also leverage a similar approach to train and learn shop-interaction embeddings, and roll up cold-start listings to shop-level representations.

5.3.4 Offline Experiment: Quality. In Table 2 we quantify the quality of the interaction embeddings by comparing the average cosine similarity of pairs of items sampled randomly from the same taxonomy versus pairs of items sampled from different ones. As expected, for both interaction types (click and cart), pairs from the same taxonomy have higher cosine similarities compared to pairs from different taxonomies. In particular, we observe a 25.47% and 13.93% increase in the cosine similarity among items in the same taxonomy versus items in different taxonomies for the click and cart embeddings, respectively. In addition, we observe that the cart embeddings are more similar among listings from same taxonomy as compared to the click embeddings. This may be due to the fact that users having a purchase intend tend to click many items, but only add a small subset of these items to their shopping carts.

5.4 Online Experiments

To demonstrate the effective of our candidate set selection methodology, in this section we discuss two A/B experiment results The first online experiment is a user-to-item module on the home page, while the second is an item-to-item module on the cart page. For each interaction, we chose the candidate set that had the highest hit rate. In particular, Table 3 shows that for items having interaction equal to click (cart), candidate sets consisting of items having interaction equal to cart (cart) yield higher hit rates than their click (click) counterparts. As a result, we focus on the Click-to-Cart and Cart-to-Cart candidate sets for our online experiments. We train an embedding-interaction model and generate the candidate sets for each module as described below.

User Recommendation Module: An A/B test was run for 7 days in which we evenly bucketed eligible signed-in users into a *control* group and two *treatment* groups. The *control* group received a recommendations module that recommends items using a candidate set that matched items based on the last 100 items that the user interacted with, regardless of the interaction type. The number of candidates for each user was at most 800 items. The first *treatment group* received a module based on the last four items the user had clicked, while the second treatment group received a module based on the last four items the user had showed intent of purchasing (e.g., items that the user had added to the shopping cart). The candidate sets for the first (second) treatment group is then constructed by finding the 200 nearest items with interaction equal to cart to the click (cart) embeddings of each of the latest clicked (added-to-cart) items. In essence, the candidate set for each user in the treatment groups consisted of 800 items.

Item Recommendation Module on Cart Page: An A/B test was run for 7 days in which we bucketed 50% of eligible signed-in users browsing on the Cart Page into a *control* group and the remainder into the *treatment* group. The *control group* received a candidate set that contained items that come from the same shop as the last item added to the shopping cart. The candidate set of the *treatment group* is obtained by taking the the nearest 200 items with interaction equal purchase to the *cart* embedding of the last item added to cart. If a cart embedding did not exist for the last item added to cart, we used the click embedding of the item.

Among two experiments, the control and treatment groups used the same re-ranking model on top of the candidate set selection phase, which has an objective function that is optimizing for predicting purchases, and uses a variety of historical user and item features as well as several content features.

We look at several business metrics to evaluate our method, including conversion rate, add-to-cart rate, checkout rate, search click rate, and pages browsed. For the user module, we observed that the first and second treatments showed a 0.71% and 1.05% increase in conversion rate as compared to the control, with statistical significance at $p < 0.1$ and $p < 0.01$, respectively. As a result, the second treatment was deployed to production. Similarly, for the item recommendation modules on the cart page, we observed an 0.39% increase in conversion rate, with statistical significant at $p < 0.05$, comparing to the control.

6 CONCLUSION

In this paper, we propose a model to learn interaction-based item embeddings and use them for candidate set selection. This method allows us to encode co-occurrence patterns between items and the way users interact with them in sessions. It departs from the majority of the extant literature in that it includes a co-occurrence based regularization term and minimizes the distance between item-interaction pairs that co-occur frequently across intra-sessions. We train our model on a large production dataset of browsing sessions from an online marketplace and evaluate its performance on offline metrics and online experiments. We observe that our proposed embeddings consistently beat all baselines in offline settings, and significantly improves top-level, revenue-related metrics across multiple recommender systems deployed to live traffic.

REFERENCES

[1] Nima Asadi and Jimmy Lin. 2013. Effectiveness/efficiency tradeoffs for candidate generation in multi-stage retrieval architectures. In *SIGIR*.
[2] Oren Barkan and Noam Koenigstein. 2016. Item2Vec: Neural Item Embedding for Collaborative Filtering. *IEEE* (2016).

[3] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching Word Vectors with Subword Information. *ACL* (2017).

[4] Fedor Borisyuk, Krishnaram Kenthapadi, David Stein, and Bo Zhao. 2016. CaSMoS: A framework for learning candidate selection models over structured queries and documents. In *KDD*.

[5] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Recsys*.

[6] Mukund Deshpande and George Karypis. 2004. Item-based top-N Recommendation Algorithms. *ACM Trans. Inf. Syst.* (2004).

[7] Chantat Eksombatchai, Pranav Jindal, Jerry Zitao Liu, Yuchen Liu, Rahul Sharma, Charles Sugnet, Mark Ulrich, and Jure Leskovec. 2018. Pixie: A system for recommending 3+ billion items to 200+ million users in real-time. In *WWW*.

[8] Mihajlo Grbovic and Haibin Cheng. 2018. Real-time Personalization Using Embeddings for Search Ranking at Airbnb. In *KDD*.

[9] 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 *KDD*.

[10] Y. Hu, Y. Koren, and C. Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *ICDM*.

[11] Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with GPUs. *arXiv preprint arXiv:1702.08734* (2017).

[12] Krishnaram Kenthapadi, Benjamin Le, and Ganesh Venkataraman. 2017. Personalized Job Recommendation System at LinkedIn: Practical Challenges and Lessons Learned. In *Recsys*.

[13] Greg Linden, Brent Smith, and Jeremy York. 2003. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing* (2003).

[14] David C Liu, Stephanie Rogers, Raymond Shiau, Dmitry Kislyuk, Kevin C Ma, Zhigang Zhong, Jenny Liu, and Yushi Jing. 2017. Related pins at pinterest: The evolution of a real-world recommender system. In *WWW*.

[15] Shichen Liu, Fei Xiao, Wenwu Ou, and Luo Si. 2017. Cascade Ranking for Operational E-commerce Search. In *KDD*.

[16] Yu A. Malkov and D. A. Yashunin. 2016. Efficient and robust approximate nearest neighbor search using Hierarchical Navigable Small World graphs. *arXiv preprint arXiv:1603.09320* (2016).

[17] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*.

[18] Dongjing Wang, Shuiguang Deng, Xin Zhang, and Guandong Xu. 2016. Learning Music Embedding with Metadata for Context Aware Recommendation. In *ICMR*.

[19] Chang Zhou, Jinze Bai, Junshuai Song, Xiaofei Liu, Zhengchao Zhao, Xiusi Chen, and Jun Gao. 2017. ATRank: An Attention-Based User Behavior Modeling Framework for Recommendation. *AAAI*.