Product Bundle Identification using Semi-Supervised Learning

Hen Tzaban
Ben-Gurion University of the Negev
Beer-Sheva, Israel
tzabanh@post.bgu.ac.il

Ido Guy
eBay Research
Netanya, Israel
idoguy@acm.org

Asnat Greenstein-Messica
Ben-Gurion University of the Negev
Beer-Sheva, Israel
asnatm@post.bgu.ac.il

Arnon Dagan
eBay Research
Netanya, Israel
ardagan@ebay.com

Lior Rokach
Ben-Gurion University of the Negev
Beer-Sheva, Israel
liorrk@bgu.ac.il

Bracha Shapira
Ben-Gurion University of the Negev
Beer-Sheva, Israel
bshapira@bgu.ac.il

ABSTRACT

Many sellers on e-commerce platforms offer buyers product bundles, which package together two or more different items. The identification of such bundles is a necessary step to support a variety of related services, from recommendation to dynamic pricing. In this work, we present a comprehensive study of bundle identification on a large e-commerce website. Our analysis of bundle compared to non-bundle listed items reveals several key differentiating characteristics, spanning the listing's title, image, and attributes. Following, we experiment with a multi-modal classifier, which takes advantage of these characteristics as features. Our analysis also shows that a bundle indicator input by sellers tends to be highly noisy and carries only a weak signal. The bundle identification task therefore faces the challenge of having a small set of manually-labeled clean examples and a larger set of noisy-labeled examples, in conjunction with class imbalance due to the relative scarcity of bundles.

Our experiments with basic supervised classifiers, using the manually-labeled and/or the noisy-labeled data for training, demonstrates only moderate performance. We therefore turn to a semi-supervised approach and propose *GREED*, a self-training ensemble-based algorithm with a greedy model selection. Our evaluation over two different meta-categories shows a superior performance of semi-supervised approaches for the bundle identification task, with GREED outperforming several semi-supervised alternatives. The combination of textual, image, and some metadata features is shown to yield the best performance, reaching an AUC of 0.89 and 0.92 for the two meta-categories, respectively.

CCS CONCEPTS

• **Information systems → Electronic commerce**; **Online shopping**; • **Theory of computation → Semi-supervised learning**.

KEYWORDS

electronic commerce; ensemble learning; product bundling; self-training; semi-supervised learning

ACM Reference Format:
Hen Tzaban, Ido Guy, Asnat Greenstein-Messica, Arnon Dagan, Lior Rokach, and Bracha Shapira. 2020. Product Bundle Identification using Semi-Supervised Learning. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20), July 25–30, 2020, Virtual Event, China.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3397271.3401128

SIGIR '20, July 25–30, 2020, Virtual Event, China

ACM ISBN 978-1-4503-8016-4/20/07…$15.00
https://doi.org/10.1145/3397271.3401128

1 INTRODUCTION

Product bundling, where two or more different goods are grouped together as one package offered for sale, typically at a discounted price, is one of the most effective marketing tools in e-commerce, as it brings value to both consumers and sellers [4]. Consumers receive a more diverse and customized set of offers, at discount rates and lower shipping costs [3, 5]. On the other hand, sellers benefit from higher revenue with fixed operational costs involved with the packaging and shipping processes, while exposing consumers to new items that they may have not considered in isolation [18, 40].

The identification of bundle listed items ("listings") is fundamental to a wide variety of e-commerce applications, from matching of listings to catalog products [31], through providing better search and recommendation [27, 47, 49], to optimizing dynamic pricing strategies [14]. In a recent work, Shah et al. mention bundle identification as a key challenge for the automation of matching listings to products and enabling product search [52]. Bundle recognition is also essential to the user experience on e-commerce websites, allowing to link between individual products and associated bundles (and vice versa), and accurately present and compare price per unit.

Despite their central role in e-commerce, no previous work, to the best of our knowledge, has focused on the identification of bundles. Rather, many of the previous works on bundle composition, recommendation, and pricing considered bundles based on the co-purchasing of listings by the same user, either in the same transaction or within a close time proximity, up to a week [3, 5, 19, 32, 60]. This definition of bundles might often be inaccurate, as users commonly co-purchase non-related items together, in order to reduce shipping costs, number of packages, and transaction fees or for the mere convenience of going through the online purchase process once. A few studies considered a bundle indicator provided by sellers [3], but as we show in this work, such a flag tends to be highly

noisy, likely due to sellers' confusion regarding the notion of a bundle, carelessness, or error, as well as their interest to fill such information in a way that would optimize their listing's sale. A small set of studies manually curated a modest set of listings or product supersets as bundles [11, 40], leading to an overall "golden set" of only a few hundred bundles, since the manual labeling process is a costly effort and as bundles are rather scarce.

In this work, we focus on the automatic identification of bundle listings. We define a bundle listing as a listing that includes two or more different products that are also offered separately for sale [53]. We experiment with two meta-categories on eBay, one of the world's largest e-commerce websites, in which bundles are relatively widespread: Cameras&Photo and Video Games&Consoles. Figure 1 presents two examples of bundle listings, one for each of the two meta-categories.

We first experiment with a fully-supervised approach, by considering 10,000 listings from each meta-category with a seller flag indicating if they are a bundle or not, of which we manually labeled nearly 1,000, at random, per meta-category. By intersecting the seller flag with our manual labels, we show that the former is highly noisy. Since our work, to the best our knowledge, is the first to study the differences between bundle and non-bundle listings, we start by conducting a comparative analysis to gain insights about their distinctive characteristics. Our analysis reveals a variety of differences that span the vocabulary used in the listing titles and properties such as price, leaf category, and number of images. Building on these findings, Applying various supervised classifiers, using the manually-labeled data, the seller's flag, or both for training over these features, yields only a moderate outcome, with AUC below 0.8.

The findings described above require a solution that addresses the situation wherein a small set of clean data is available, together with a larger noisy-labeled data, with class imbalance leading to a skewed distribution. Many recent studies have addressed challenges such as noisy-labeled data (often without any clean data) or class imbalance (e.g., [21, 23, 35, 48]), but the specific combination in our case calls for the use of semi-supervised approaches. Our proposed algorithm is called GREED– a GReedy Ensemble-based EmbeDded self-trained classifier. It is based on a classic self-training approach [38], which iteratively identifies portions of the noisy-labeled data that can be labeled and used for training in the next iteration. The noisy labels are used as features in this process. In each iteration, an equal number of positive and negative examples are added to the training set, to gradually balance it. In addition, rather than taking the last model in the process, GREED builds an ensemble of models, which is created during the self-training iterations in an "embedded" manner (rather than by pre- or post-processing). The selection of models to include in the ensemble is performed greedily, by including only models that lead to a performance enhancement over preceding ones. Finally, all models selected for the ensemble are combined using a meta-classifier that learns a linear combination of their weights.

In order to examine the importance of the different elements of GREED, we compare its performance for the bundle identification task to baselines that include plain self-training and an ensemble-based self-training without greedy selection. In addition, we examine the use of a popular semi-supervised algorithm that uses a diverse ensemble for this task [34] and the use of a ladder network for semi-supervised learning [45]. Performance evaluation of these algorithms shows the effectiveness of semi-supervised learning for our task. The results achieved by GREED outperform all baselines, reaching an AUC of 0.92 for Cameras&Photo and 0.89 for Video Games&Consoles.

The main contributions of this work can be summarized as follows:

- We introduce the first comprehensive study of bundle identification in e-commerce and present a comparative analysis of bundle versus non-bundle listings on eBay, revealing a handful of distinctive characteristics.
- We release a dataset with nearly 2,000 listings from two different meta-categories in eBay, with each labeled as either a bundle or not.
- We propose a multimodal learning approach for the task of bundle identification, which takes advantage of a rich set of features spanning image, text, and metadata.
- We introduce GREED, a novel semi-supervised algorithm based on ensemble learning, leveraging a small set of well-labeled examples combined iteratively with subsets of noisy-labeled examples, using a self-training greedy approach.
- We demonstrate the effectiveness of our proposed algorithm across two principal e-commerce meta-categories, reaching a high AUC with precision of well over 90% and recall ranging from 79% to 86%.

2 RELATED WORK

Our study focuses on the identification of bundle listings and our proposed semi-supervised approach aims to address the case of having a small set of cleanly-labeled examples and a larger set of examples with noisy labels, both with class imbalance. Accordingly, we review related work in two areas: research of bundles on e-commerce and machine learning approaches for dealing with noisy-labeled unbalanced data.

2.1 Bundles in E-commerce

Since product bundling is a popular selling strategy, it has attracted substantial research interest in the commerce literature for many years [1, 24, 56]. With the rise of electronic commerce, research of bundles has focused on different aspects of online shopping, including personalized recommendation [5, 40, 60], pricing optimization strategies [18, 53], and package composition [4, 32, 33, 51, 55, 57]. Research of bundles span a wide variety of commerce domains, including travel [33, 51], food [13], personal care [32], video games [40], computers [18], and telecommunication [11].

Recommendation of items and products is one of the most well-studied tasks in e-commerce [50]. It is therefore of no surprise that a decent portion of the literature on bundles focuses on the task of bundle recommendation. Zhu et al. [60] referred to the set of e-commerce recommended items as a bundle, considering the relationships among them. Their proposed algorithm modeled the task as a quadratic knapsack problem and showed superior performance over baselines using Walmart data in terms of conversion rate and revenue. Garfinkel et al. [18] designed a shopbot for bundle purchases, by extending user-based and item-based collaborative

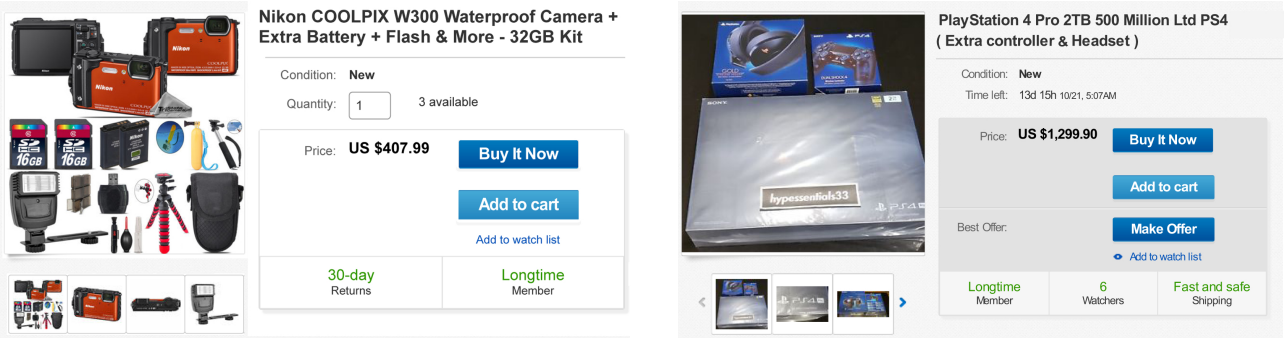

Figure 1: Example bundle listings for Cameras&Photo (left) and Video Games&Consoles (right) on eBay.

filtering with price optimizations. Experimentation over retail data showed that their approach can lead to substantial savings for bundle-purchasing consumers. Beladev et al. [5] introduced an algorithm for recommending optimal and personalized bundles, by integrating collaborative filtering techniques with personal demand curves and price models. Pathak et al. [40] studied personalized recommendation of bundles within a video game platform. Features such as the size of the bundle and the compatibility of its items were found to play a key role in the recommendation's success. Smart Plan [11] introduced a recommender system for product and service bundles in the telecommunication industry, allowing users to easily generate a combined service plan that best suits their needs. Bundle recommendation to groups rather than individuals has been investigated by Shuyao et al. [44], who presented several practical applications, including the recommendation of vacation packages to tourist groups, entertainment collections to groups of friends, or sets of courses to groups of students. Their group recommendation algorithm introduced a notion of fairness, so that no user is consistently slighted by the item selection in the package.

Related to the research of bundle recommendation is the study of their assembly. Xie et al. [55] generated personalized bundles by capturing individual preferences using a linear utility function learned from implicit user feedback. Liu et al. [33] focused on the personalized composition of travel package. At the core of their algorithm was a topic model that considered the characteristics of both the target tourist and the travel location. Ge et al. [19] studied bundling strategies by e-commerce platforms and found that products with more reviews, especially reviews with photos, are included in more bundles. Liu et al. [32] modeled bundle items in a graph and presented an algorithm for recommending items to form a new bundle. Their algorithm was shown effective over data from Chinese e-commerce site TaoBao in both Cosmetics and Perfume categories. In a recent work, Bai et al. [3] introduced a bundle generation network, which addressed both bundle quality and diversity using an encoder-decoder framework. Experiments showed that their approach improved both the quality and diversity of the assembled bundles.

Aside from the selection of items that compose the bundle, setting its price, or associated discount, plays a key role in a bundle's success. Stremersch and Tellis [53] studied different bundling strategies and their optimization with regards to profit maximization, market competition, and consumer perception. Ghose and Sundararajan [20] assessed the optimality of price discrimination in the software industry, where bundling characteristics are a key factor. Yang et al. [57] suggested an algorithm for product bundling that optimized the spread in viral marketing and showed its superiority over competitive baselines, in terms of both computational overhead and bundle quality.

In all previous work, bundle identification is given as an input, typically derived from co-purchases by the same user [3, 5, 19, 32, 60]. In a few studies, a small set of manually-labeled bundles, either by the e-commerce platform or by the user, was considered [3, 11, 40]. To the best of our knowledge, our work is the first to focus on the automatic identification of bundle offerings. Since co-purchases can often occur with non-related items, e.g., to save shipping cost or transaction fees, and since manually-labeled data is scarce and often noisy, high-quality bundle identification is the first necessary step to enable the various tasks described above.

2.2 Learning with Noisy Labeling

The problem of classification with label noise has been an active research area, as modern machine learning models work best when trained on large volumes of labeled data [16]. Yet, acquiring labels is costly, motivating the need for effective semi-supervised learning techniques that leverage unlabeled examples and noisy labeling, as labelers may pose bias due to lack of expertise, dedication, or personal preference [10, 21, 35, 48, 59].

Variants robust to noise have been proposed for the most common classifiers, such as logistic regression and SVM [7, 16]. Following the success of deep learning, recent research proposed different approaches to handle noisy labeling for neural networks. Goldberger and Ben-Reuven [21] suggested viewing the correct label as a latent random variable and modeled the noise processes by a communication channel with unknown parameters. They applied expectation-maximization to tune the parameters of both the network and the noise and estimate the correct label. Han et al. [23] proposed "co-teaching" two neural networks for dealing with noisy labels. These approaches, however, did not consider the combination of noisy-labeled data with a smaller seed of clean data. Clark et al. [10] suggested mixing labeled and unlabeled data by applying supervised machine learning on the labeled data and a semi-supervised algorithm on the unlabeled data, to improve the

representations of a Bi-LSTM sentence encoding. Iscen et al. [29] introduced a method for semi-supervised learning using a label propagation approach based on nearest neighbors graphs, employed on the embeddings of a deep neural network to generate pseudo-labels for the unlabeled data. They showed this approach improves the learning performance of the neural network. Semi-supervised learning with ladder networks [45] by simultaneously minimizing the sum of supervised and unsupervised cost functions, achieved high performance for learning tasks where a small amount of labeled data is combined with a large amount of unlabeled data. We use a variant of this algorithm over noisy data with over-sampling techniques as one of our main baselines.

Active learning has been proposed to effectively utilize unlabeled data with a costly oracle [17]. Zhang et al. [59] suggested an active learning framework with multiple imperfect annotators, including two modules for label integration and instance selection to improve the learning effectiveness with unbalanced noisy labels. Recently, Samel and Miao [48] presented a new active deep denoising approach that first builds a deep neural network noise model and then applies an active learning algorithm to identify the optimal denoising function. All of the above studies focus on scenarios of noisy labels with no clean labels; the combination of clean labels with larger unlabeled set; or the selection of the most informative instances for clean labeling. None of them, however, has dealt with the scenario of unbalanced data classification based on a small predefined set of labeled data and a larger set of noisy-labeled data.

GREED, our proposed approach for bundle identification, can be combined with different classification algorithms for effective learning in the presence of noisy labels. Other commonly used algorithm-independent approaches apply pre-processing techniques, to increase the seed labeled set or balance the data using under- or over-sampling [9, 35, 58]. In contrast, GREED uses an "embedded" process to address class imbalance and noisy labels, seeking for instances to both increase and balance the training set. Unlike other methods, especially deep neural networks, GREED requires minimal tuning, since the algorithm itself iterates over many combinations of thresholds and selectively combines the results.

3 DATASET AND CHARACTERISTICS

3.1 Datasets

Our dataset includes items (listings) from eBay, a large online auction and shopping website, where people and businesses buy and sell a wide variety of goods worldwide. We focus on two different bundle-reach meta-categories: Cameras&Photo ("C&P") and Video Games&Consoles ("VG&C"). In addition, we only consider "buy it now" listings (as opposed to auctions) with condition "new" (as opposed to "used") within the United States version of the site.

Each listing includes a *bundle flag* input by the seller to indicate whether it is a bundle or not. Our complete dataset includes 10,000 listings from each meta-category, uploaded to the eBay US site during 2019. In order to obtain a reliable set of labeled listings, we randomly sampled 1,000 listings out of the 10,000 for each of the two meta-categories. Each of these listings was labeled by one annotator as either bundle or not. In accordance with previous work [1, 53], we define a *bundle* listing as an offering of two or more different items, each of which sold separately for a nonzero price. This definition excludes "multipacks" or "lots" [36], which consist of multiples of the same exact product within a listed offer (e.g., a pack of 3 identical batteries). In contrast, a bundle offer is comprised of at least two distinct products. We refer to all other listings as *single* listings.

Labeling was performed by five annotators, who were presented with the listing's page on eBay, which includes, among other elements, the listing's title, description, image, and price. Annotators received detailed guidelines, explaining the definition of a bundle listing versus a single listing, with examples of both. They also performed qualification tests, i.e., an iterative process of labeling, followed by feedback from other annotators, until the quality was aligned among all. Borderline cases included, for example, an item with a warranty, which was decided to be marked as a bundle only in the case where the warranty is separately sold, in accordance with the definition stated above. Another example is an item offered with a special added value, such as a signed copy of a book. In this case, we decided to refer to it as a single. At the end of the process, the inter-annotator agreement for the task of bundle versus single labeling, measured by the Fleiss Kappa [15] , was 0.83 for C&P and 0.86 for VG&C, calculated over a set of 150 listings labeled by three different annotators. The portion of listings labeled as a bundle was 32% for C&P and 27% for VG&C, rendering an unbalanced dataset for each of the two meta-categories. The dataset is published as part of this work.[1]

3.2 Bundle vs. Single Characteristics

For each listing in our labeled dataset, we examined a variety of characteristics that may distinguish bundles from singles. Following, we report the characteristics for which we found considerable differences between bundle and single listings.

Title. The listing's title is one of its most prominent features, as demonstrated in Figure 1. We set out to examine the most distinctive terms that characterize titles of bundle listings compared to single listings and vice versa. To this end, we used Kullback-Leibler (KL) divergence, which is a non-symmetric distance measure between two given distributions [6, 22]. Specifically, we calculated the terms that contribute the most to the KL divergence between the language model of the bundle titles versus the language model of the single titles, and vice versa. The most distinctive unigrams are presented in Table 1. It can be seen that the unigram list most characterizing bundles (relative to singles) includes, in both the C&P and VG&C meta-categories, additive conjunctions, such as `and`, `with`, `+`, and `&`. The lists also include words used to describe bundling, including `kit`, `package`, and, prominently, the word `bundle` itself, which is at the top of the VG&C list, but only third on the C&P list. On the other hand, the lists of terms characterizing titles of singles reflect listings that do not tend to be part of a bundle, such as new releases and series, as well as qualities such as `sealed` and `import` and the preposition `for`, which often characterizes parts and accessories (preceding the compatible models or brands).

Price. The upper row of Table 2 shows that both the average and median price of bundle listings are substantially higher than single listings, in both C&P and VG&C. This likely stems from the fact

[1] https://rnd.ebay.co.il/research/datasets/bundles

Table 1: Most distinctive unigrams for bundle vs. single ('bundle') and single vs. bundle ('single') listing titles.

Cameras&Photo		Video Games&Consoles	
Bundle	Single	Bundle	Single
+	photo	bundle	series
kit	new	+	for
bundle	no	limited	sealed
,	for	with	import
&	release	package	figure

Table 2: Average (median) price and number of images for bundle vs. single listings.

	Cameras&Photo		Video Games&Consoles	
	Bundle	Single	Bundle	Single
Price ($)	625.1 (399.0)	132.5 (35.0)	295.1 (90.0)	75.4 (25.6)
#images	6.4 (7.0)	4.6 (4.0)	5.2 (4.0)	3.5 (2.0)

that bundle listings include multiple products. It also implies that bundles are often associated with a relatively expensive product.

Number of images. Sellers can upload up to 12 different images per listing, either single or bundle. The lower row of Table 2 shows that the number of images uploaded for bundle listings is considerably higher than for single listings. It appears that since bundles include multiple products, they are likely to prompt more photos from sellers.

Leaf category. Both the C&P and VG&C meta-categories include a variety of sub-categories. The portion of bundles may vary according to the listing's leaf category, which is the lowest level in eBay's taxonomy. Table 3 shows the portion of bundles within the most common leaf categories in both C&P and VG&C. It can be seen that the bundle portions substantially vary across the leaf categories, ranging from below 10% to nearly 85%.

Seller's flag. As mentioned above, sellers can indicate on eBay whether their offering is a bundle or not when they list it for sale on the site. Table 4 shows the portion of bundle listings in our dataset according to the seller's flag. It can be seen that when the seller indicates the listing is a bundle, the actual portion of bundles ranges from about 60% to 65% for both meta-categories. On the other hand,

Table 3: Portion of bundle listings for top leaf categories.

Cameras&Photo		Video Games&Consoles	
Leaf Category	%Bundle	Leaf Category	%Bundle
Digital Cameras	84.7%	Video Game Consoles	71.6%
Lenses	54.7%	Chargers & Charging Docks	37.5%
Camcorders	48.3%	Game Merchandise	35.1%
Batteries	46.9%	Bags Skins & Travel Cases	18.8%
Flashes	40.0%	Video Games	13.4%
Lighting & Studio	14.8%	Controllers & Attachments	7.7%
Tripods & Monopods	9.5%	Toys to Life	7.5%

Table 4: Portion of bundle listings per seller's flag value.

Cameras&Photo		Video Games&Consoles	
Single	Bundle	Single	Bundle
24.1%	64.1%	14.9%	60.3%

when the listing is indicated to be a single by the seller, the portion of actual bundles ranges from 15% to 24%. These findings show that while the seller's flag carries a clear signal as for whether the listing is a bundle or not, it is very noisy: a large portion of the bundle-flagged listings are actually singles, while a decent portion of the listings not flagged as bundles are in fact bundles. Relying solely on the seller's input would therefore result in a rather low precision and recall. From this point onward, we refer to the seller's flag for each listing as its "*noisy label*". In addition, we refer to the portion of the complete dataset that has not been manually-labeled (9000 listings per meta-category) as the "*noisy-labeled dataset*."

4 SUPERVISED CLASSIFIERS

Our initial attempt to distinguish between bundle and single listings was using fully-supervised classifiers, including Naïve Bayes, Logistic Regression, Support Vector Machine (SVM) with RBF kernel, Random Forest, Gradient Boosting, and XGBoost. All classifiers were run using the scikit learn library [41].

Based on the analysis presented in the previous section, the features of the models included the following:

(1) The listing's title. We considered all unigrams with frequency of 2 and above in the entire dataset as features.[2] In addition, we used a word embedding representation of the titles. We pre-trained word vectors with 10 million titles for each of the two meta-categories using GloVe [42]. Each title was embedded by a weighted average of the word vectors, using smooth inverse frequency.[3]
(2) The listing's main image, represented using image embedding based on a 1024-dimension float vector trained using ResNet-50 [26], over more than 50 million images from the eBay site. This representation is the result of the last embedded layer in the network, whose end goal is the categorization of images to leaf categories [43].
(3) The listing's metadata including its price, leaf category, and number of images .

We also experimented with a fastText classifier [30] over the listing's titles. We did not examine the use of product descriptions [12, 39] for the bundle identification task, and leave this direction for future work.

In all settings, for each of the two meta-categories, the labeled dataset was divided into training (65%), validation (15%), and test (20%) sets. We experimented with three different settings for the training set: in the first, only the allocated labeled set was used for training; in the second, only the noisy-labeled data, described in Section 3, was used for training; and in the third, both the manually-labeled data and the noisy-labeled data were used for training. We

[2]Experimenting with bigrams did not show any performance gain.
[3]We used the code published by the authors, with the default hyper-parameters [2].

Table 5: AUC performance results of supervised classifiers with three different training sets: manually-labeled data only ('Clean'), noisy-labeled data only ('Noisy'), and both manually-labeled and noisy-labeled data ('C+N').

	Cameras&Photo			Video Games&Consoles		
	Clean	Noisy	C+N	Clean	Noisy	C+N
Naïve Bayse	0.684	0.697	0.710	0.677	0.674	0.692
Random Forest	0.691	0.678	0.683	0.634	0.621	0.671
Logistic Regression	0.743	0.712	0.695	0.764	0.731	0.742
XGBoost	0.768	0.745	0.754	0.711	0.695	0.699
SVM	0.771	0.761	0.744	0.765	0.721	0.744
fastText	0.773	0.703	0.717	0.768	0.727	0.731
Gradient Boosting	**0.784**	0.738	0.757	**0.777**	0.72	0.755

refer to these as the *clean*, *noisy*, and *clean+noisy* settings, respectively. In all three settings, validation and testing were performed solely over manually-labeled data. We carried out an extensive process of hyper-parameter tuning for each of the classifiers over the validation set. For example, for the gradient boosting classifier, our tuning included the maximum depth of the estimators, the learning rate, and the maximum number of features to consider when searching for the best split.

Table 5 presents the area under the ROC curve (AUC) for the different supervised classifiers, across the clean, noisy, and clean+noisy settings. The clean setting generally yielded higher AUC than the noisy setting, in both meta-categories (Naïve Bayse and Random Forest in the C&P category were the only exceptions). Using the combination of both in the clean+noisy setting, led to a slight performance improvement on top of the clean setting over some of the classifiers and meta-categories, but in many cases the clean setting yielded the highest results. The best performance, for both meta-categories, was attained by the gradient boosting classifier with the clean setting. We henceforth refer to this setting as the *"supervised"* baseline.

5 SEMI-SUPERVISED APPROACHES

5.1 Overview

The supervised classifiers described in the previous section, reached a maximum AUC of 0.78, which yielded 81.8% precision for C&P and 81.2% for VG&C, with 66.1% and 64.3% recall for the two meta-categories, respectively. As evident from our experimentation with the different supervised classifiers, using the noisy-labeled data "as is" for training, together with the clean yet smaller data, leads to a very limited performance gain, if at all, on top of using solely the clean data. We therefore turn to semi-supervised approaches, self-training (also referred to as incremental learning) in particular [38]. First, a supervised learning algorithm is trained based on the cleanly-labeled data to produce a base model. Then, the classifier is applied to the noisy-labeled data, with the noisy label as a feature, to generate more labeled examples as input for the supervised learning algorithm. This process repeats for multiple iterations, attempting to gradually improve the classification performance.

Our algorithm is designed to deal with the situation of having unbalanced data, with a small set of clean labels and a larger set of noisy labels . It applies three principles to the iterative self-training process: (1) Ensemble learning: rather than selecting a single model at the end of the iterative process, as done by traditional self-training methods [38], it combines multiple models identified throughout the iterations. Ensemble models have often been found to be more robust, as they reduce the overall error by averaging the variance of multiple individual models [46]; (2) Embedded: the selection of models to include in the ensemble classifier is performed throughout the iterative process, rather than before or after, while traversing the two key parameters of the iterative process: the confidence threshold and the iteration size; (3) Greedy: the selection of models is performed on a greedy basis, where only models that outperform previous models are included in the final ensemble. Ultimately, the selected models are weighted using a meta-classifier, referred to as the *combiner*, to produce the final prediction (single or bundle). Accordingly, we name our algorithm *GREED*: a GReedy Ensemble-based EmbeDded classifier. To the best of our knowledge, we are the first to propose this combination of qualities for a self-trained classifier.

5.2 The GREED Algorithm

As mentioned above, GREED is a semi-supervised self-training algorithm, which is based on an iterative process. Given a confidence threshold and the desired number of new examples (instances) to add in each iteration, a subset of the noisy-labeled data is added to the training set and a model is induced for the new aggregate training set. Only models that improve preceding models are combined into the final ensemble model. The building blocks, presented in Algorithm 1, are based on a general ensemble framework [46]:

Data: As mentioned in Section 3.1, our algorithm uses a labeled dataset, and a larger "noisy-labeled" dataset. As in the case of the supervised classifiers, described in Section 4, the labeled dataset is divided into a training set, a validation set, and a test set, with a 65%, 15%, and 20% respective proportions. The training set, denoted by T, changes throughout the iterations of the algorithm and is initialized to the respective portion of the manually-labeled dataset. Along the iterations of the algorithm, subsets of the noisy-labeled dataset, denoted as NL, are added to the training set T. The validation set V is used for hyper-parameter tuning, model selection, and meta-classification, as detailed below.

Base model: The base classifier iteratively used by the algorithm is denoted as $\mathcal{F}$. We used the gradient boosting model as our base classifier, as it showed the best performance among all supervised classifiers, as detailed in Table 5. As mentioned in Section 4, hyper-parameter tuning was performed for this classifier across each of the two meta-categories. The initial training set, based solely on the labeled data, is used to create an initial model, which is included in the set of models selected for the ensemble (Algorithm 1 line 4). The performance of this model serves as a basis for the greedy selection process across all values of the confidence threshold and iteration size, as later described.

Model accumulator: The "model accumulator", described in Function 1, is iteratively applied to generate training sets over which the base model is activated, and serves to cope with both

Algorithm 1 GREED

$\mathcal{F}$ - Base model
W - Meta-classifier for ensemble weighting
T - Manually-labeled training set
V - Manually-labeled validation set
NL - Noisy-labeled dataset
$MaxItr$ - Maximum number of iterations
$ItrSizeList$ - List of varying iteration sizes
$ThresholdList$ - List of varying confidence thresholds

1: Initialize ensemble $C \leftarrow \{\}$
2: $model \leftarrow \mathcal{F}(T)$
3: $e \leftarrow model.computeAUC(V)$
4: $C \leftarrow C \cup model$
5: **for all** $conf \in ThresholdList$ **do**
6: **for all** $itrSize \in ItrSizeList$ **do**
7: $args \leftarrow \mathcal{F}, T, V, NL, C, conf, itrSize, model, e, MaxItr$
8: $C \leftarrow$ AccumulateModels($args$)
9: $EnsembleModel \leftarrow W(C, V)$

Function 1 AccumulateModels($args$)

$args$: $\mathcal{F}, T, V, NL, C, confidence, itrSize, model, e, MaxItr$

1: **for all** $iteration \in MaxItr$ **do**
2: $R \leftarrow model.predict(NL)$
3: $R' \leftarrow (R > conf)$
4: **if** $len(R'[class = x]) < itrSize/2$ **then**
5: break;
6: $A \leftarrow itrSize/2$ samples from R' when single
7: $B \leftarrow itrSize/2$ samples from R' when bundle
8: $NL \leftarrow NL - (A \cup B)$
9: $T \leftarrow T \cup (A \cup B)$
10: $model \leftarrow \mathcal{F}(T)$
11: $e' \leftarrow model.computeAUC(V)$
12: **if** $e' > e$ **then**
13: $e \leftarrow e'$
14: $C \leftarrow C \cup model$
15: **return** C

the noisy labels and class imbalance. It handles the noisy labels by considering only examples for which the prediction in the current iteration bypasses a *confidence threshold*. The base model provides class confidence scores over the noisy-labeled dataset, so instances classified with high confidence are added to the training set in the next iteration. Class imbalance is addressed by adding an equal number of examples with positive and negative predictions in each iteration. The overall number of examples added in each iteration is determined by the *iteration size*. Both the confidence threshold and the iteration size, which together control the iterative process of the model accumulator, are given to it as input. The model accumulator is repeatedly run with different values of confidence threshold and iteration size (Algorithm 1, lines 5-6). Specifically, we set confidence threshold values to range from 0.85 to 0.99, with leaps of 0.01 and iteration size values to range from 30 to 150, with leaps of 30. These iterations over the values of the two parameters that control the model accumulator's iterative process spare the need of adjusting them using classic hyper-parameter tuning, since the meta-classifier performs this final "tuning", as later described. The only hyper-parameter tuned, in addition to those of the base model, is the maximum number of iterations per individual run of the model accumulator (Function 1, line 1).

In each of its iterations, the model accumulator learns a model over the aggregate training set T and applies it to the remaining noisy-labeled set NL (Function 1, line 2). It then identifies the set of instances in NL for which the prediction confidence produced by the model is higher than the confidence threshold input parameter (R' in Function 1, line 3). The algorithm aims at adding a number of new instances equal to the iteration-size parameter to the training set that was aggregated until the current iteration (T in Function 1, line 9). To this end, half of the instances are sampled uniformly at random out of all instances in R' that were classified as positive and the other half are sampled uniformly at random out of those in R' that were classified as negative (Function 1, lines 6-7). These are then moved from NL to T and used as training examples in the next iteration, with their respective prediction as the label. As already mentioned, this step helps to gradually reduce class imbalance in the original training set. In case there are not enough positive or negative instances that pass the threshold, the algorithm stops (Function 1, lines 4-5). Otherwise, it continues for the next iteration, until this stopping condition is met or until the maximum number of iterations has been executed.

Selector: The embedded selection step added to the ensemble framework greedily selects the models to be combined by the meta-classifier "on-the-fly", as part of the iterative process (Function 1, lines 12-14). For a given pair of confidence threshold and iteration size, the model accumulator selects only models that outperform the preceding models, in terms of AUC over the validation set V, to be included in the final ensemble model.

Combiner: After the iterative process is concluded across all pairs of confidence threshold and iteration size, the selected models are joined together into a final model by the combiner (Algorithm 1, line 9). The combiner is a meta-classifier that optimizes the integration of the different models using a logistic regression.[4] The input for the meta-classifier is the actual prediction (bundle or single) per each of the selected models. The meta-classifier learns an optimized combination of these predictions over the validation set.

5.3 Baseline Methods

We compare the performance of GREED to the best performing supervised classifier (gradient boosting, as shown in Table 5). In addition, to examine the effect of the unique characteristics of GREED – the ensemble approach and the greedy selection – we compare it to a plain self-training algorithm and to a GREED variant that does not apply the greedy selection. We also compare GREED to the DECORATE semi-supervised algorithm, which also applies self-training with ensemble learning and promotes diversity across the individual models using artificially-constructed training examples. Finally, we consider the ladder network semi-supervised classifier as a baseline. This model is trained to simultaneously minimize the

[4]We also experimented with SVM and neural network combiners, however logistic regression consistently yielded a slightly better performance.

sum of supervised and unsupervised cost functions by backpropagation.

Self-training: The basic self-training baseline iteratively expands the training set with instances from the noisy-labeled dataset, based on their confidence scores. Similarly to GREED, an initial base model is trained using gradient boosting, and instances are added each iteration based on the confidence threshold and the iteration size parameters. As opposed to GREED, these parameters are not traversed as an embedded part of the algorithm, but rather optimized as hyper-parameters over the validation set, in addition to the total number of iterations. Also in contrast to GREED, no ensemble of models is built and the applied model is the one produced at the final iteration [38].

Non-greedy GREED: In order to examine the effect of the greedy selection of models embedded during the iterative process of GREED, we compare it to a non-greedy version. GREED selects only models that show improvement over prior models based on the performance measured over the validation set. The non-greedy variant, in contrast, uses all the models created along the iterations as part of the ensemble. Aside from this difference, the algorithm works similarly to GREED, as described in Section 5.2.

DECORATE: DECORATE is an ensemble of highly diverse models induced from additional artificially-generated training data. It works in the same ensemble framework as GREED, but rather than the model accumulator it employs a diversity generator over the unlabeled data (noisy-labeled in our case, with the noisy label used as a feature, similarly to GREED). In each iteration, the diversity generator samples, uniformly at random, a set of instances from the noisy-labeled set for which the classification confidence is equal or higher than a confidence threshold. These examples are added to the original training set, which is created, as in GREED, solely based on the labeled set. However, to increase diversity, the label for these examples is artificially inverted before adding them to the training set. This artificial diversification aims at improving the overall ensemble's generalization capabilities. In the next iteration, the set of instances for which the labels were inverted are removed from the training set and a new set of artificially-created instances are added to the original training set in the same manner. The different models are ultimately joined using a voting mechanism based on their prediction confidence rather than using a meta-classifier as in GREED. In addition, the selection of models to be included in the ensemble is different: while in GREED the selection is based on the performance of the individual model, since DECORATE aims at high diversity, it examines the overall performance of the ensemble and adds the new model only if it reduces the ensemble's error rate. Moreover, since the training set in GREED incrementally increases along its iterations, it is reasonable to require that the individual model performance would also grow. By contrast, in DECORATE the training set's size does not change over the iterations (starting from the second one). Overall, we conjectured that DECORATE might be less suitable for unbalanced data, since wrong examples over a minority class might degrade performance more acutely. Moreover, the balancing of examples and emphasis on individual model performance as part of the greedy selection may be more appropriate to our task.

Ladder Network: A ladder network is structured as an autoencoder with skip connections from the encoder to the decoder. The learning task is similar to that in denoising autoencoders, but applied at every layer, not just the inputs. The skip connections relieve the pressure to represent details at the higher layers of the model because, through the skip connections, the decoder can recover any details discarded by the encoder. When used for semi-supervised learning, the ladder network uses latent variable models, which are effective in jointly reducing supervised and unsupervised loss functions [45]. In recent years, ladder networks have shown to achieve state-of-the-art results for a variety of semi-supervised learning tasks (e.g., [28, 37, 54]). We used the Keras implementation of the ladder network described in [45],[5] with 150 epochs and the default hyper-parameter configuration. As the method did not demonstrate good performance over the original class balance, we experimented with a variety of over-sampling techniques. The best results were achieved using 10% duplicate records and additional 15% records generated using Adaptive Synthetic Sampling Method for Imbalanced Data (*ADASYN*) [25]. ADASYN is an adaptation of the popular SMOTE algorithm [8], which focuses on synthesising more examples for the minority examples that are considered difficult to learn. ADASYN uses a weighted distribution for different minority class examples according to their learning difficulty in order to gradually shift the classification decision boundary towards the difficult examples [25].

6 EXPERIMENTAL RESULTS

Table 6 presents the performance results of the different classifiers described in Section 5 over the (manually-labeled) test set. Our main metric is the AUC, while precision and recall for the bundle class are also reported. As mentioned before, we use gradient boosting, which achieved the best performance when trained over the clean data, as our supervised classifier baseline. It can be seen that the DECORATE classifier achieves a substantial improvement compared to the supervised classifier's performance, for both meta-categories (AUC up 7.8% for C&P and 6.4% for VG&C), giving a first indication that the iterative use of the noisy-labeled data in a semi-supervised framework yields a considerable performance gain. The Self-Training classifier achieves slightly higher results than the DECORATE classifier, reinforcing our conjecture that the aggressive diversity imposed by the DECORATE algorithm may be inappropriate in our case. The ensemble version, non-greedy GREED, further improves the results, with AUC up 3.7% for C&P and 1.1% for VG&C, compared to Self-Training, indicating that combining the models produced throughout the self-training iterations is productive. The Ladder Network demonstrates a generally good performance, up 0.5% for C&P and 3.7% for VG&C compared to non-greedy GREED. Finally, the greedy ensemble-based GREED reaches the highest performance for both meta-categories, with a high AUC of 0.917 for C&P and 0.891 for VG&C, up 4.6% and 5.8%, respectively, relative to non-greedy GREED and up 4.1% and 2.1%, respectively, relative to the Ladder Network. Overall, the AUC attained by GREED is higher by 17.0% and 14.7% compared to the best performing supervised classifier, for C&P and VG&C, respectively. Precision reached by GREED for both meta-categories is higher than 92%, while recall ranges from nearly 80% for VG&C to over 85% for C&P.

[5] https://github.com/divamgupta/ladder_network_keras

Table 6: AUC, precision, and recall results of different algorithms for the bundle identification task.

Classifier	Cameras&Photo			Video Games&Consoles		
	AUC	Prec.	Recall	AUC	Prec.	Recall
Supervised	0.784	64.6%	80.7%	0.777	67.3%	83.2%
DECORATE	0.845	84.2%	77.5%	0.827	89.1%	69.2%
Self-Training	0.846	75.6%	88.2%	0.833	85.0%	83.7%
Non-greedy GREED	0.877	74.9%	93.1%	0.842	85.2%	77.5%
Ladder Network	0.881	83.6%	86.1%	0.873	90.3%	76.1%
GREED	**0.917**	94.3%	86.6%	**0.891**	92.1%	78.5%

Figure 2 presents a comparison between the three ensemble methods as a function of the number of models participating in the ensemble (ensemble size), for the C&P meta-category. As can be observed, the superiority of GREED is becoming consistent after an ensemble of 40 models and continues to grow with the size of the ensemble. In other words, the performance of GREED improves more substantially as more models are added to the ensemble, based on its embedded greedy selection process described in Section 5.2.

To better understand the contribution of the different features, we trained the GREED classifier with different feature subsets. Table 7 presents the AUC results. It can be seen that the title and image features show the highest performance when used on their own. In addition, they lead to the highest decrease of performance when excluded. Title text and embedding yield rather comparable performance, while the combination of both leads to a further performance gain in both meta-categories, indicating that joining lexical and semantic features yields the best discriminative outcome. The item's price and leaf category show higher contribution than the noisy label, giving another indication to the generally low reliability of the seller-input labels for the bundle identification task.

The lower section of Table 7 shows the performance of additional feature subsets. The combination of title (both text and embedding) and image (embedding) achieves some performance gain over using each alone. Adding the noisy label leads to a further noticeable increase in the results. Overall, we see that the multi-modal approach indeed yields an improved performance. For the C&P meta-category, the combination that achieves the best performance includes the title (text only), image embedding, price, noisy label, and number of images. For VG&C, the best combination includes the title (text and embedding), image embedding, price, noisy label, and leaf category.

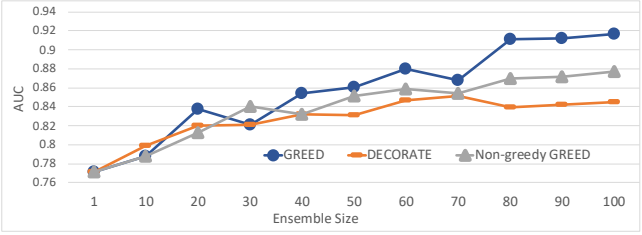

Figure 2: AUC of DECORATE, non-greedy GREED, and GREED for the Cameras&Photo meta-category, as a function of the ensemble size.

Table 7: AUC performance results of the GREED classifier when using ('Only') or disregarding ('Exclude') different features. The 'All' row refers to all features jointly.

Features	C&P		VG&C	
	Only	Exclude	Only	Exclude
#images	0.643	0.883	0.692	0.891
Noisy label (nl)	0.710	0.892	0.691	0.859
Price	0.727	0.902	0.696	0.857
Leaf Category	0.741	0.893	0.761	0.847
Image embedding	0.804	0.845	0.778	0.845
Title text	0.795	0.867	0.805	0.844
Title embedding	0.798	0.872	0.771	0.855
Title (text+embedding)	0.849	0.827	0.815	0.828
Title+image	0.861	0.750	0.836	0.757
Title text+image+nl	0.912	0.831	0.860	0.793
Title text+image+leaf+nl	0.872	0.810	0.874	0.748
Title text+image+price+nl+#images	**0.917**	0.811	0.823	0.778
Title+image+leaf+price+nl	0.883	0.643	**0.891**	0.691
All	0.894	–	0.871	–

7 CONCLUSION

Our analysis revealed several characterizing differences between bundle and single listings: bundles tend to have higher price, higher number of images, and various distinctive terms in their titles. They are also spread differently across leaf categories. Last but not least, the bundle flags input by sellers provide only a noisy signal. Taking advantage of these characteristics, we examined different learning approaches to identify bundles with higher accuracy. We found semi-supervised approaches, self-training in particular, to be effective for the task. Using ensemble-based self training with greedy selection reached the highest performance, with AUC ranging from 0.89 to 0.92 across two different meta-categories abundant with bundle listings: Cameras&Photo and Video Games&Consoles. The title (represented both by n-grams and word embedding) and image (embedded representation) were the two primary sources of features, but signals such as the leaf category and price, in addition to the seller's noisy label, added to the overall performance.

Effective identification of bundle listings can help a variety of core e-commerce tasks, from presenting the price per unit, through matching listings to catalog products, to producing successful recommendations to customers. Future research directions may include the study of bundles in additional e-commerce domains; the identification of the bundle's individual products; and the application of additional machine learning techniques to the bundle identification task.

REFERENCES

[1] William James Adams and Janet L Yellen. 1976. Commodity Bundling and the Burden of Monopoly. *The quarterly journal of economics* 90, 3 (1976), 475–498.
[2] Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A Simple but Tough-to-Beat Baseline for Sentence Embeddings. In *Proc. of ICLR*.

[3] Jinze Bai, Chang Zhou, Junshuai Song, Xiaoru Qu, Weiting An, Zhao Li, and Jun Gao. 2019. Personalized Bundle List Recommendation. In *Proc. of WWW*. 60–71.

[4] Arash Beheshtian-Ardakani, Mohammad Fathian, and Mohammadreza Gholamian. 2018. A Novel Model for Product Bundling and Direct Marketing in E-Commerce based on Market Segmentation. *Decision Science Letters* 7, 1 (2018), 39–54.

[5] Moran Beladev, Lior Rokach, and Bracha Shapira. 2016. Recommender Systems for Product Bundling. *Knowledge-Based Systems* 111 (2016), 193–206.

[6] Adam Berger and John Lafferty. 1999. Information Retrieval as Statistical Translation. In *Proc. of SIGIR*. 222–229.

[7] Jakramate Bootkrajang and Ata Kabán. 2012. Label-Noise Robust Logistic Regression and its Applications. In *Proc. of ECML-PKDD*. 143–158.

[8] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. 2002. SMOTE: synthetic minority over-sampling technique. *JAIR* 16 (2002), 321–357.

[9] Nitesh V Chawla, Nathalie Japkowicz, and Aleksander Kotcz. 2004. Special Issue on Learning from Imbalanced Data Sets. *ACM SIGKDD Explorations Newsletter* 6, 1 (2004), 1–6.

[10] Kevin Clark, Minh-Thang Luong, Christopher D Manning, and Quoc V Le. 2018. Semi-Supervised Sequence Modeling with Cross-View Training. *arXiv preprint* abs/1809.08370 (2018).

[11] Paolo Dragone, Giovanni Pellegrini, Michele Vescovi, Katya Tentori, and Andrea Passerini. 2018. No More Ready-made Deals: Constructive Recommendation for Telco Service Bundling. In *Proc. of RecSys*. 163–171.

[12] Guy Elad, Ido Guy, Slava Novgorodov, Benny Kimelfeld, and Kira Radinsky. 2019. Learning to Generate Personalized Product Descriptions. In *Proc. of CIKM*. 389–398.

[13] Yan Fang, Xinyue Xiao, Xiaoyu Wang, and Huiqing Lan. 2018. Customized Bundle Recommendation by Association Rules of Product Categories for Online Supermarkets. In *Proc. of DSC*. 472–475.

[14] Noelia Oses Fernandez, Jon Kepa Gerrikagoitia, and Aurkene Alzua-Sorzabal. 2015. Dynamic Pricing Patterns on an Internet Distribution Channel: The Case Study of Bilbao's Hotels in 2013. In *Information and Communication Technologies in Tourism 2015*. 735–747.

[15] Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin* 76, 5 (1971), 378–382.

[16] Benoit Frenay and Michel Verleysen. 2014. Classification in the Presence of Label Noise: A Survey. *IEEE transactions on neural networks and learning systems* 25, 5 (2014), 845–869.

[17] Dragan Gamberger, Nada Lavrac, and Saso Dzeroski. 2000. Noise Detection and Elimination in Data Preprocessing: Experiments in Medical Domains. *Applied Artificial Intelligence* 14, 2 (2000), 205–223.

[18] Robert Garfinkel, Ram Gopal, Arvind Tripathi, and Fang Yin. 2006. Design of a Shopbot and Recommender System for Bundle Purchases. *Decision Support Systems* 42, 3 (2006), 1974–1986.

[19] Xinyu Ge, Yousha Zhang, Yu Qian, and Hua Yuan. 2017. Effects of Product Characteristics on the Bundling Strategy Implemented by Recommendation Systems. In *Proc. of ICSSSM*. 1–6.

[20] Anindya Ghose, Arun Sundararajan, et al. 2006. Evaluating Pricing Strategy using E-Commerce Data: Evidence and Estimation Challenges. *Statist. Sci.* 21, 2 (2006), 131–142.

[21] Jacob Goldberger and Ehud Ben-Reuven. 2017. Training Deep Neural-Networks using a Noise Adaptation Layer. In *Proc. of ICLR*.

[22] Ido Guy and Bracha Shapira. 2018. From Royals to Vegans: Characterizing Question Trolling on a Community Question Answering Website. In *Proc. of SIGIR*. 835–844.

[23] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. 2018. Co-Teaching: Robust Training of Deep Neural Networks with Extremely Noisy Labels. In *Proc. of NIPS*. 8536–8546.

[24] Ward Hanson and R Kipp Martin. 1990. Optimal Bundle Pricing. *Management Science* 36, 2 (1990), 155–174.

[25] Haibo He, Yang Bai, Edwardo A Garcia, and Shutao Li. 2008. ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In *Proc. of IJCNN*. 1322–1328.

[26] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *Proc. of CVPR*. 770–778.

[27] Sharon Hirsch, Ido Guy, Alexander Nus, Arnon Dagan, and Oren Kurland. 2020. Query Reformulation in E-Commerce Search. In *Proc. of SIGIR*.

[28] Ruiqi Hu, Shirui Pan, Jing Jiang, and Guodong Long. 2017. Graph ladder networks for network classification. In *Proc. of CIKM*. 2103–2106.

[29] Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondrej Chum. 2019. Label propagation for deep semi-supervised learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 5070–5079.

[30] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of Tricks for Efficient Text Classification. *arXiv preprint* abs/1607.01759 (2016).

[31] Anitha Kannan, Inmar E. Givoni, Rakesh Agrawal, and Ariel Fuxman. 2011. Matching Unstructured Product Offers to Structured Product Specifications. In *Proc. of KDD*. 404–412.

[32] Guannan Liu, Yanjie Fu, Guoqing Chen, Hui Xiong, and Can Chen. 2017. Modeling Buying Motives for Personalized Product Bundle Recommendation. *ACM Trans. Knowl. Discov. Data* 11, 3 (2017), 28:1–28:26.

[33] Qi Liu, Yong Ge, Zhongmou Li, Enhong Chen, and Hui Xiong. 2011. Personalized Travel Package Recommendation. In *Proc. of ICDM*. 407–416.

[34] Prem Melville and Raymond J Mooney. 2003. Constructing Diverse Classifier Ensembles using Artificial Training Examples. In *Proc. of IJCAI*, Vol. 3. 505–510.

[35] Yu Meng, Jiaming Shen, Chao Zhang, and Jiawei Han. 2018. Weakly-Supervised Neural Text Classification. In *Proc. of CIKM*. 983–992.

[36] Ajinkya More. 2016. Attribute Extraction from Product Titles in eCommerce. *arXiv preprint* abs/1608.04670 (2016).

[37] Ajay Nagesh and Mihai Surdeanu. 2018. Keep your bearings: Lightly-supervised information extraction with ladder networks that avoids semantic drift. In *Proc. of NAACL-HLT*. 352–358.

[38] Kamal Nigam and Rayid Ghani. 2000. Analyzing the Effectiveness and Applicability of Co-Training. In *Proc. of CIKM*. 86–93.

[39] Slava Novgorodov, Ido Guy, Guy Elad, and Kira Radinsky. 2019. Generating Product Descriptions from User Reviews. In *Proc. of WWW*. 1354–1364.

[40] Apurva Pathak, Kshitiz Gupta, and Julian McAuley. 2017. Generating and Personalizing Bundle Recommendations on Steam. In *Proc. of SIGIR*. 1073–1076.

[41] Fabian Pedregosa et al. 2011. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* 12 (2011), 2825–2830.

[42] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global Vectors for Word Representation. In *Proc. of EMNLP*. 1532–1543.

[43] Utkarsh Porwal. 2019. Learning Image Information for eCommerce Queries. *arXiv preprint* abs/1904.12856 (2019).

[44] Shuyao Qi, Nikos Mamoulis, Evaggelia Pitoura, and Panayiotis Tsaparas. 2016. Recommending Packages to Groups. In *Proc. of ICDM*. 449–458.

[45] Antti Rasmus, Mathias Berglund, Mikko Honkala, Harri Valpola, and Tapani Raiko. 2015. Semi-supervised learning with ladder networks. In *Advances in neural information processing systems*. 3546–3554.

[46] Lior Rokach. 2010. Ensemble-based Classifiers. *Artificial Intelligence Review* 33, 1-2 (2010), 1–39.

[47] Jennifer Rowley. 2000. Product Search in e-Shopping: a Review and Research Propositions. *Journal of consumer marketing* 17, 1 (2000), 20–35.

[48] Karan Samel and Xu Miao. 2018. Active Deep Learning to Tune Down the Noise in Labels. In *Proc. of KDD*. 685–694.

[49] J Ben Schafer, Joseph Konstan, and John Riedl. 1999. Recommender Systems in E-Commerce. In *Proc. of EC*. 158–166.

[50] J Ben Schafer, Joseph A Konstan, and John Riedl. 2001. E-commerce Recommendation Applications. *Data Mining and Knowledge Discovery* 5, 1-2 (2001), 115–153.

[51] Dimitris Serbos, Shuyao Qi, Nikos Mamoulis, Evaggelia Pitoura, and Panayiotis Tsaparas. 2017. Fairness in Package-to-Group Recommendations. In *Proc. of WWW*. 371–379.

[52] Kashif Shah, Selcuk Kopru, and Jean David Ruvini. 2018. Neural Network Based Extreme Classification and Similarity Models for Product Matching. In *Proc. of NAACL:HLT*. 8–15.

[53] Stefan Stremersch and Gerard J Tellis. 2002. Strategic Bundling of Products and Prices: A New Synthesis for Marketing. *Journal of marketing* 66, 1 (2002), 55–72.

[54] Jian-Hua Tao, Jian Huang, Ya Li, Zheng Lian, and Ming-Yue Niu. 2019. Semi-supervised ladder networks for speech emotion recognition. *International Journal of Automation and Computing* 16, 4 (2019), 437–448.

[55] Min Xie, Laks V. S. Lakshmanan, and Peter T. Wood. 2014. Generating Top-k Packages via Preference Elicitation. *Proc. VLDB Endow.* 7, 14 (2014), 1941–1952.

[56] Manjit S Yadav. 1994. How Buyers Evaluate Product Bundles: A Model of Anchoring and Adjustment. *Journal of Consumer Research* 21, 2 (1994), 342–353.

[57] De-Nian Yang, Wang-Chien Lee, Nai-Hui Chia, Mao Ye, and Hui-Ju Hung. 2012. On Bundle Configuration for Viral Marketing in Social Networks. In *Proc. of CIKM*. 2234–2238.

[58] Show-Jane Yen and Yue-Shi Lee. 2009. Cluster-based Under-Sampling Approaches for Imbalanced Data Distributions. *Expert Systems with Applications* 36, 3 (2009), 5718–5727.

[59] Jing Zhang, Xindong Wu, and Victor S Shengs. 2015. Active Learning with Imbalanced Multiple Noisy Labeling. *IEEE Transactions on Cybernetics* 45, 5 (2015), 1095–1107.

[60] Tao Zhu, Patrick Harrington, Junjun Li, and Lei Tang. 2014. Bundle Recommendation in Ecommerce. In *Proc. of SIGIR*. 657–666.