Brand Recommendations for cold-start problems using Brand Embeddings

David Azcona[1] and Alan F. Smeaton[2]

[1] Zalando SE, Valeska-Gert-Straße 5, 10243 Berlin, Germany
david.azcona@zalando.ie,
WWW home page: https://computing.dcu.ie/~dazcona/
[2] Insight Centre for Data Analytics, Dublin City University, Dublin, Ireland
alan.smeaton@dcu.ie
WWW home page: https://computing.dcu.ie/~asmeaton/

Abstract. This paper presents our work to **recommend brands** to customers that might be **relevant** to their **style** but the brands are new to them. To promote the exploration and discovery of new brands, we leverage *article-embeddings*, also known as Fashion DNA, a learned encoding for each article of clothing at Zalando, that is utilized for product and outfit recommendations. The model used in Fashion DNA's work proposed a Logistic Matrix Factorization approach using sales data to learn customer style preferences. In this work, we evolved that approach to circumvent the **cold-start problem** for recommending new brands that do not have enough sales or digital footprint. First, we computed an embedding per brand, named Brand DNA, from the Fashion DNA of all articles that belong to a given brand. Then, we trained a model using Logistic Matrix Factorization to predict sales for a set of frequent customers and brands. That allowed us to learn customer style representations that can be leveraged to predict the likelihood of purchasing from a new brand by using its Brand DNA. Customers are also able to further explore Zalando's assortment moving from the more popular products and brands.

Keywords: Embeddings, Neural Networks, Latent Representations, Deep Learning

1 Introduction

Zalando is Europe's leading online platform for fashion and lifestyle with 45 Million active customers, 560 Million visits per month, more than 1 Million clothing articles and operates in 23 countries. At this time, Zalando offer articles for more than 4,700 brands. In Zalando's Fashion Store, we strive to deeply understand customers and enable them to express their unique style. However, Zalando's recommender systems are biased towards popularity rankings. Zalando would thus like to promote the exploration and discovery of new brands and for that, in this paper we propose a new way to learn high-level representations for brands that as yet do not have many interactions with customers as a way to circumvent the cold-start problem.

2 Related Work

An embedding is a vector in a relatively low-dimensional space used to encode meaningful information from a particular domain. They are often used as inputs to Machine Learning models after converting from large sparse vectors.

In industry applications, embeddings are widely used. They are first-class citizens of **Twitter**'s Machine Learning infrastructure [1], made available to other applications though publishing in Twitter's own feature store and generating embeddings for entities such as words (hashtags, usernames, emojis, etc) and users. They, in addition, pre-select users for whom they have enough information to assign an embedding to. In a similar manner, **Drop** generates brand embeddings as they are very focused on both customers and brands entities [2]. They leverage brand embeddings to recommend brands to customers. Similarity between brands is also measured using the learned latent space. Finally, **Sephora** learned embeddings for beauty products by utilizing data from customers browsing their products generated from sliding windows [3]. A matrix factorization approach with a bias vector is employed to avoid overly recommending popular products [4].

At **Zalando**, researchers have been working on mapping and recommendation of clothing articles [5], building Fashion DNA, a learned vector representation for each article. On their first iteration, a Deep Neural Network was trained using both images and tags of articles to predict sales for frequent customers. The model utilized was a Logistic Matrix Factorization approach and no temporal information was used. The product of Fashion DNA embeddings and customer style vectors gives the forecast purchase likelihood for a $customer - article$ pair. The network to train *article-embeddings* predicts tags (not sales) and those embeddings are used statically when training the logistic factorization network using customer sales events. The cosine distance between *article-embeddings* vectors is used as a measure of item similarity.

Fashion DNA *article-embeddings* overcomes the 'cold–start problem' of collaborative recommendation approaches when there is no sales data for new articles by using only article information and images. Embeddings for customers can also be extracted in the latent space of customer style preferences. The embeddings are used for several applications at Zalando such as tagging images, outfit generation and similar-item recommendations. **Etsy** has also learned a latent space for article embeddings, however they use customer interaction events such as article viewings and favorings, adding to a collection or to cart, or purchasing [6]. **StitchFix** explored how to learn style in a latent space and how to to explain and interpret in higher dimensions [7].

More recently, embeddings have been leveraged to aid the selection of product sizes for customers, a major problem for online retailers. **ASOS** learned a latent space for product sizes using only past purchases and brand information [8]. **Zalando** and **Shopify** proposed a method to learn size and fit embeddings using $customer - article$ interactions for both customers and articles [9]. Customer and article attributes can be mapped into a latent space to derive personalized size and fit recommendations.

3 Our Proposed Methodology

Our proposed recommendation system is to help customers find brands in Zalando's large assortment that they might not have thought to search for on their own. It is straightforward to recommend brands that customers have bought in the past so we want to go beyond that in this work and to provide deeper recommendations based on customers' style. Our first step is to identify customers and brands with enough information from which to learn embeddings. For that, we collect a list of sales events and other metadata that we use to train models that learn customer preferences. Then, a model is trained using the extracted data. We will base our approach on Fashion DNA's original work developed by Bracher *et al.* at Zalando Research [5]. A Brand DNA representation will be learned for each brand as a Fashion DNA representation is learned for each article. No temporal information will be used in this approach either. Figure 1 depicts the proposed architecture of a Logistic Matrix Factorization where the product of Brand DNA and customer style vectors gives the predicted purchase likelihood for the $customer - brand$ pair. The probability of customer j of purchasing brand i is defined as the following:

$$\rho_{ij} = \sigma(fb_i \cdot s_j + b_j) \tag{1}$$

where σ is the sigmoid function, fb is the Brand DNA embedding of brand i, s is the customer style embedding of customer j and an associated bias term b for customer j.

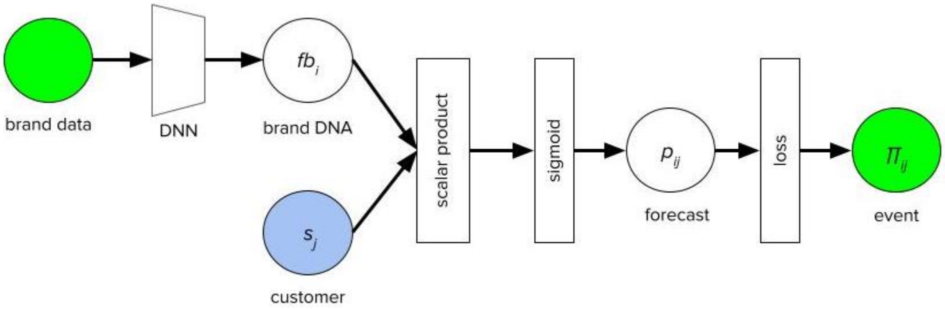


Fig. 1: Logistic Matrix Factorization Approach using a Brand DNA

4 Data

Our first step is to retrieve, extract and transform the data that will serve for training a model that learns brand preferences for our customers. That is often one of the most time consuming stages of training and serving pipelines in production. In Zalando, event data is streamed and stored in a Datalake and, for brand recommendations, models are trained or re-trained on a daily basis for offline predictions. The first data stream we retrieved is **sales** events, also known as checkouts. This was composed of 25,791,970 customers purchasing from 3,510 brands, taken from May to November 2020. From this we extract the information we need such as customer, timestamp and brand purchased.

This dataset is **sparse** in nature as customers have usually bought from a small number of brands with respect to all the brands available at the shop (the dataset would be even more sparse for products). Therefore we choose to select a subset of **frequent customers** and **frequent brands** with enough sales events that will aid us in training a model that learns brand preferences for customers and generalizes well with respect to new brands. From those sales events, we proceeded to extract cohorts of frequent customers, i.e. customers that bought the most number of brands. We acknowledge that such customers may not be the same customers who bought the greatest number of products. Table 1 shows the distribution of the number of brands bought by cohorts of frequent customers. As more customers are added to the frequent customers cohort, the mean number of brands purchased decreases. This table also shows that frequent customers buy a great portion of all the purchases at the shop during the six months of data logging.

Table 1: Cohorts of Frequent Customers: Customers that Bought the Most Brands

Number of Most Frequent Customers		Distribution of Brands bought by these Customers					
		Mean	Std Dev	Min	Max	Number of purchases	
10	(0.000038%)	506.1	73.94	410	656	20,405	(0.008%)
100	(0.00038%)	247.17	98.96	183	656	104,949	(0.04%)
1,000	(0.0038%)	152.83	46.27	126	656	563,313	(0.22%)
10,000	(0.038%)	101.63	24.66	83	656	3,063,498	(1.20%)
100,000	(0.388%)	62.83	17.22	48	656	16,136,453	(6.36%)
1,000,000	(3.88%)	32.65	13.05	21	656	71,533,791	(28.20%)
25,791,970	(100%)	5.41	7.36	1	656	253,621,895	(100%)

Table 1 shows that the 1 million most frequent customer cohort, representing almost 4% of the total customers, bought from around 30 brands on average with a standard deviation of around 13 brands. Among these, the customer who bought the least number of brands was 21 brands and the customer who bought the greatest number of brands were 656. There were more than 71 million articles bought by this cohort of customers which made up to almost 30% of all purchases.

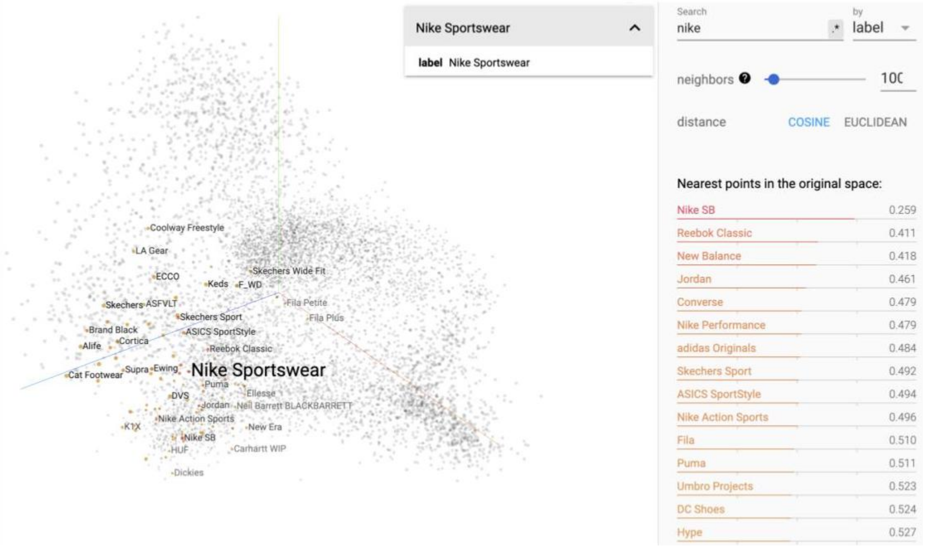
In addition, from these sales events, we also extracted cohorts of frequent brands. These were brands that were bought by the largest number of customers with respect to other less purchased brands. Table 2 shows the distribution of the number of customers that bought from those cohorts of frequent brands. In a similar manner, the more frequent brands we consider, the fewer number of customers on average that bought from them. This table also highlights that these frequent brands have very high purchase rates with respect to other less popular brands.

Table 2: Cohorts of Frequent Brands: Brands that have been purchased most often. Max is 3,734,907 for all rows

Number of Most Frequent Brands		Distribution of Number of Customers buying those Brands				
		Mean	Std Dev	Min	Number of purchases	
5	(0.14%)	3,145,814.6	418,630.55	2,747,288	34,349,406	(13.54%)
50	(1.42%)	1,371,174.58	837,279.01	579,228	139,554,061	(55.02%)
100	(2.85%)	880,285.88	771,005.58	280,726	172,453,228	(68.00%)
200	(5.70%)	529,912.7	648,124.53	119,047	202,497,631	(79.84%)
500	(14.25%)	250,487.55	469,034.24	34,047	232,551,415	(91.69%)
1,000	(28.49%)	134,758.91	351,161.21	10,625	246,892,382	(97.35%)
3,510	(100%)	39,723.87	196,751.33	1	253,621,895	(100%)

Table 2 shows that the 50 most frequent brands, around 1.4% of all the brands, were bought by around 1.3 million customers on average with a standard deviation of around 837 thousand customers. The brand within that top 50 that was bought by the least number of customers had around 580 thousand customers and that bought by the greatest number of customers had around 3.7 million. These top 50 brands were in more than half, 55%, of all purchases during those six months.

The second data stream we leverage is **article embeddings**, also known as Fashion DNA in Zalando and we use a total of 5,807,733 articles (SKUs) and 6,967 brands. For each article, a Fashion DNA embedding has been computed and stored in the Datalake which we were able to retrieve for those articles. The idea for Brand DNA in the future is to learn a high level representation for brands by training a Deep Neural Network with brand information, as depicted earlier in Figure 1, inputting images and other metadata regarding brands in order to predict some brand characteristics. That is similar to how Fashion DNA is computed for articles. However, that is outside the scope of this work so in order to leverage a learned representation for brands we computed the average of their Fashion DNA for each brand with respect to all articles that belong to it giving us a 128-D embedding per brand. By applying a dimensionality reduction technique, Figure 2 shows that, qualitatively, these Brand DNAs computed by just averaging the articles' Fashion DNA have some meaningful encoded value.



Nike Sportswear

label Nike Sportswear

Search nike .* by label

neighbors 10(

distance COSINE EUCLIDEAN

Nearest points in the original space:

Nike SB	0.259
Reebok Classic	0.411
New Balance	0.418
Jordan	0.461
Converse	0.479
Nike Performance	0.479
adidas Originals	0.484
Skechers Sport	0.492
ASICS SportStyle	0.494
Nike Action Sports	0.496
Fila	0.510
Puma	0.511
Umbro Projects	0.523
DC Shoes	0.524
Hype	0.527

Fig. 2: Dimensionality Reduction for Brand DNA using PCA

5 Training Learning Models

We carried out several experiments based on varying the thresholds used for selecting frequent customers and frequent brands. Brands were input in batches to train our learning algorithm while all customer style vectors were learned at the same time. Therefore, for each experiment, we split the set of brands into three subsets: training (with 70% of the brands), validation (with 20% of the brands) and test (10% of the brands). Table 3 shows some of the experiments carried out in terms of combinations of numbers of frequent customers and frequent brands, and the corresponding split into the three subsets. For each split, we show the number of brands on each (it is not exact as we are using Spark for parallel processing). For each split, we also show how dense (as opposed to sparse) was the purchase matrix, our groundtruth to learn from. As we add more frequent customers and brands to the experiments, the sparsity of this matrix increases because there are fewer sales events and hence there is an increase in difficulty for learning patterns in a supervised task.

To illustrate, the experiment with 1,000 customers and 500 brands was split into 341 brands for training, 112 for validation and 47 for testing. In training, the purchase matrix had 341,000 items (1,000 customers times 341 brands) and 21.96% of those were actually purchased, i.e. there was a ‘1’ in the purchase matrix given a customer and a brand.

We then proceeded to train models that learn customer preferences with regards to brands. We trained Logistic Matrix Factorization models, adapted from Bracher *et al.*’s work that was introduced in Section 3 and Figure 1, for each experiment with the specified thresholds for frequent customers and brands.

Table 3: Experimental settings and the corresponding split of brands

Experiment		Data Split: splitting brands					
Frequent Customers	Frequent Brands	Train 70%		Validation 20%		Test 10%	
		Brands	Density	Brands	Density	Brands	Density
10	5	5	100%	0	0%	0	0%
100	50	33	70.48%	12	64%	5	72.8%
1,000	100	70	48.29%	23	43.63%	7	52.61%
1,000	500	341	21.96%	112	20.42%	47	22.66%
10,000	200	140	27.07%	44	26.72%	16	29.03%
100,000	1,000	679	5.85%	219	6.39%	102	4.87%
1,000,000	200	140	11.23%	44	10.87%	16	11.55%
25,791,970	3,510	2,391	0.16%	759	0.13%	360	0.14%

Customer style vectors were learned as weights of the model while training it using batches of brand embeddings. The loss function used was binary cross entropy and the weight matrix was initialized using Glorot's (also known as Xavier's) initialization [10]. In order to address the overfitting, we applied L2 regularization to our optimizer, Adam, using the weight decay hyperparameter. The learning rate was 1e-3 for the experiments.

6 Evaluation

We evaluated the performance of our models trained for each experiment in two ways. First, our models were trained in batches of brands with brands split between train, validation and test. The validation split helped us fine tune our models and the performance on the test split was measured as the final unbiased metric. Table 4 shows some preliminary results for each experiment and, if we do not apply any regularization, models tend to overfit fairly easily and memorize the training set. We proceeded then to apply regularization to these models using weight decay.

The second evaluation of the models was in time with the next purchases, if any, for the frequent customers for whom we learned customer style vectors. Plus, we only consider brands that are part of the frequent brands cohort for this experiment. For that, we measured their NDCG [11] as a metric to measure ranking quality. The NDCG ranking is computed at positions 4, 10 and 100, that is scoring the ranking. Table 5 shows, for each experiment, the number of customers that bought on the following day, week and month that were part of the frequent customers cohort and that those brands were part of the frequent brands cohort used in each experiment. Table 6 shows the performance with all our evaluation metrics, including the time evaluation, of two of our models trained for two different experiments: 100 frequent customers and 50 frequent brands, and 1 million frequent customers and 200 frequent brands respectively.

Table 4: Preliminary Results per Experiment

Experiment			Evaluation			
Frequent Customers	Frequent Brands	Weight decay	Train	Val	Test	Next day
			F1	F1	F1	NDCG @ 10
100	50	0	100%	73.76%	78.83%	0.0867 ± 0.2232
100	50	5	83.17%	76.19%	82.91%	0.1077 ± 0.1833
1,000	100	0	100%	55.41%	59.10%	0.0441 ± 0.1453
1,000	100	5	50.97%	49.14%	55.28%	0.0417 ± 0.1384
10,000	200	0	98.90%	36.93%	40.64%	0.0097 ± 0.0741
10,000	200	3	23.42%	31.48%	32.36%	0.0227 ± 0.1123
1,000,000	200	0	99.76%	20.04%	21.72%	0.0118 ± 0.0805
1,000,000	200	10	16.25%	16.19%	16.44%	0.0321 ± 0.1242

Table 5: Time Evaluation: Customers that Bought After the Training Phase

Experiment		Evaluation					
Frequent Customers	Frequent Brands	Next Day		Next Week		Next Month	
		Customers	Mean ± SD[a]	Customers	Mean ± SD	Customers	Mean ± SD
10	5	4	2.25 ± 0.50	8	2.38 ± 1.06	10	4.20 ± 1.14
100	50	44	3.32 ± 3.09	80	7.14 ± 5.86	95	13.12 ± 8.12
1,000	100	256	3.02 ± 3.03	769	6.00 ± 4.84	956	14.32 ± 8.42
1,000	500	303	4.28 ± 4.52	813	9.67 ± 8.39	959	26.34 ± 16.66
10,000	200	1,718	3.07 ± 2.86	6,414	5.46 ± 4.35	9,037	13.01 ± 8.20
100,000	1,000	11,525	3.27 ± 3.08	49,786	5.29 ± 4.31	84,446	12.38 ± 8.49
1,000,000	200	51,977	2.48 ± 2.14	285,298	3.20 ± 2.60	616,864	5.1 ± 3.94
25,791,970	3,510	327,916	2.16 ± 1.94	2,018,860	2.50 ± 2.28	6,742,352	3.55 ± 3.54

[a] SD stands for Standard Deviation

7 Discussion & Conclusions

In this work, we extracted data from cohorts of frequent customers and brands, and computed embeddings from brands using their Fashion DNA. We then trained models to learn customer preferences using batches of brand embeddings. We achieved **reasonable performance** when training and validating models in batches of brand embeddings.There is a trade-off between the threshold being used to generate customer style embeddings from the frequent customers cohort and the sparsity of the purchase matrix and the difficulty to learn patterns. The problem gets harder as we add those more frequent cohorts and the weight matrix to be learned is larger. Further experimentation would need to be carried out using other types of regularizers such as batch normalization, dropout or L1 penalty though that was outside the scope of this paper.

Table 6: Time Evaluation: Results per Experiment

Experiment		Results			
Frequent Customers	Frequent Brands		Next Day	Next Week	Next Month
100	50	Customers	44	80	95
		NDCG @ 4	0.0453 ± 0.1453	0.0874 ± 0.2461	0.1018 ± 0.2796
		NDCG @ 10	0.1077 ± 0.1833	0.2050 ± 0.2736	0.2562 ± 0.2794
		NDCG @ 100	0.2998 ± 0.1081	0.4145 ± 0.1672	0.5245 ± 0.1728
1,000,000	200	Customers	51,977	285,298	616,864
		NDCG @ 4	0.0110 ± 0.0864	0.0126 ± 0.0914	0.0178 ± 0.1091
		NDCG @ 10	0.0243 ± 0.1078	0.0302 ± 0.1186	0.0484 ± 0.1488
		NDCG @ 100	0.1337 ± 0.1274	0.1550 ± 0.1292	0.2039 ± 0.1332

In order to improve performance solely using sales events as the ground truth, deeper networks can be utilized. However, when evaluating on purchases bought by customers after we trained models and calculated the ranking metrics (NDCG), our approach was **far less accurate** than models based on pure supervised learning where we compute features of brands being viewed and previously purchased by customers. It is important to note that this **multi-label problem** is hard. Our ground truth is not one-hot encoded where we can predict probabilities that sum up to one using softmax. Each probability is the likelihood of each customer buying from each brand and is between 0 and 1.

We have shown how **Fashion DNA** and embeddings for articles and collections of articles can be leveraged to circumvent the **cold-start problem** not only for articles but for collections of articles such as brands. For instance, we can now compute a Brand DNA for a new brand with insufficient sales data and then for customers for whom we have learned customer style vectors, we compute the likelihood of those customers purchasing from that brand. We can then target customers with the highest likelihood of buying from that brand. In that way we are flipping who we create predictions for; instead of predicting which brands each customer is likely to buy, we target customers that are the most likely to buy from a particular brand. This approach is **extensible** to other collections of articles besides brands based on attributes such as color or sustainability.

Some open questions still remain such as how far back should we look for training models and whether 6 months is enough? What about seasonality? What are the thresholds we should use for frequent customers and brands? To answer these, and other questions, we should run similar experiments over time so to be able to provide confidence intervals for time periods and thresholds.

More importantly, we are working towards solutions that are customer centric, where we can provide **deeper recommendations** to customers in terms of style and facilitate their exploration of Zalando's vast assortment in a more intuitive way. That will be better measured with a **customer lifetime value (CLV) KPI**. In this use case, we could see how a handful of brands drive an enormous fraction of the sales in Zalando and we hope to balance this out by pushing the recommendation of new brands.

References

1. L. B. Dan Shiebler, Chris Green and A. Tayal. (2018) Embeddings@twitter. [Online]. Available: https://blog.twitter.com/engineering/en_us/topics/insights/2018/embeddingsattwitter
2. G. Tenorio. (2019) Building a recommender system using embeddings. [Online]. Available: https://drop.engineering/building-a-recommender-system-using-embeddings-de5a30e655aa
3. S. Truong. (2018) Collaborative embeddings for lipstick recommendations. a sephora sea machine learning case study. [Online]. Available: https://towardsdatascience.com/collaborative-embeddings-for-lipstick-recommendations-98eccfa816bd
4. M. Kula, "Metadata embeddings for user and item cold-start recommendations," *arXiv preprint arXiv:1507.08439*, 2015.
5. C. Bracher, S. Heinz, and R. Vollgraf, "Fashion dna: merging content and sales data for recommendation and article mapping," *arXiv preprint arXiv:1609.02489*, 2016.
6. X. Zhao, R. Louca, D. Hu, and L. Hong, "Learning item-interaction embeddings for user recommendations," *arXiv preprint arXiv:1812.04407*, 2018.
7. E. Boyle. (2018) Understanding latent style. [Online]. Available: https://multithreaded.stitchfix.com/blog/2018/06/28/latent-style/
8. K. Dogani, M. Tomassetti, S. Vargas, B. P. Chamberlain, and S. De Cnudde, "Learning embeddings for product size recommendations." in *eCOM@ SIGIR*, 2019.
9. A.-S. Sheikh, R. Guigourès, E. Koriagin, Y. K. Ho, R. Shirvany, R. Vollgraf, and U. Bergmann, "A deep learning system for predicting size and fit in fashion e-commerce," in *Proceedings of the 13th ACM conference on recommender systems*, 2019, pp. 110–118.
10. X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2010, pp. 249–256.
11. Y. Wang, L. Wang, Y. Li, D. He, W. Chen, and T.-Y. Liu, "A theoretical analysis of ndcg ranking measures," in *Proceedings of the 26th annual conference on learning theory (COLT 2013)*, vol. 8. Citeseer, 2013, p. 6.