Beer2Vec: Extracting Flavors from Reviews for Thirst-Quenching Recommendations

Jean-Thomas Baillargeon
jean-thomas.baillargeon@ift.ulaval.ca
Université Laval
Québec, Canada

Nicolas Garneau
nicolas.garneau@ift.ulaval.ca
Université Laval
Québec, Canada

ABSTRACT

This paper introduces the Beer2Vec model that allows the most popular alcoholic beverage in the world to be encoded into vectors enabling flavorful recommendations. We present our algorithm using a unique dataset focused on the analysis of craft beers. We thoroughly explain how we encode the flavors and how useful, from an empirical point of view, the beer vectors are to generate meaningful recommendations. We also present three different ways to use Beer2Vec in a real-world environment to enlighten the pool of craft beer consumers. Finally, we make our model and functionalities available to everybody through a web application.

CCS CONCEPTS

• Information systems → Retrieval models and ranking;
• Computing methodologies → Artificial intelligence.

ACM Reference Format:
Jean-Thomas Baillargeon and Nicolas Garneau. 2022. Beer2Vec: Extracting Flavors from Reviews for Thirst-Quenching Recommendations. In Proceedings of (NLPIR ‘22). ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/XXXXXXX.XXXXXXX

1 INTRODUCTION

A decade ago, selecting a beer from the liquor store was an easy task. Major companies were offering a limited array of products from large scale production. Nowadays, an influx of craft beers offers new tasting experiences to amateurs, and many brewers are tapping into this market. Ten years ago, there were 310 Canadian breweries\(^1\) competing for the average Canadian palette. Today, over 1100 breweries\(^2\) compete, not only for the average amateur money but also for rare product niches, whose price range overlaps with the ones of good wine bottles.

This vast offer isn’t without drawbacks for both sides of the value chain. On the one hand, it overwhelms the consumer with too many hard-to-distinguish products. On the other hand, the sellers may not have a beer connoisseur readily available to help the customer find the most appropriate beer among the ones he has in-store.

\(^1\)https://industry.beercanada.com/statistics
\(^2\)https://en.wikipedia.org/wiki/Beer_in_Canada

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

NLPIR ‘22, December 16–18, 2022, Bangkok, Thailand
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-XXXX-X/18/06...
https://doi.org/10.1145/XXXXXXX.XXXXXXX

Rating systems such as untappd\(^3\) or ratebeer\(^4\) provided important tools for beer geeks around the world to distinguish a beer (or a brewery) from its peer. These systems allow them to assess the quality—to some extent—of the next beer they will drink and enable them to share their beverage appreciation. This exchange of information is made on the promise of an enhanced recommendation for their next beer.

These recommendation systems, however, suffer from three main problems. They use a similarity ranking based on user preference proximity that forgoes the beer’s flavor profile aspect. These systems don’t consider the purchase context, where a user needs to select a beer from a limited amount of products or how a user would like to select a beer based on a personal choice of flavors. Moreover, some clients may have difficulty expressing the flavors they are looking for.

We provide a solution for these three problems by introducing the Beer2vec model, a vector representation learning procedure that pulls its roots from the word2vec’s skip-gram model[7]. Beer2vec naturally enables vector arithmetics, allowing beers to be expressed in terms of flavors. Transforming beers into vectors implies that a user can select an excellent beer to his liking when faced with an immense wall of beer by comparing two beers’ taste profiles with a straightforward vector distance calculation. He can obtain the optimal beer from the shelf without requiring the merchant to hire a beer sommelier. Furthermore, the vectorial operation allows analogies in the Beer2vec application, where a user can add and subtract tastes from a beer to interpolate towards another one. The code and the beer vectors will be publicly available upon acceptance.

The Beer2vec application is also showcased on a website that contains all the experimentations and results from Section 6. Although the scope of this work is limited to beer, we believe that other communities using flavor tagging, such as Vivino\(^5\) for vine amateurs and Connors\(^6\) for whisky amateurs would benefit from this application.

The next sections are organized in the following way; we first introduce in Section 2 the related work regarding representation learning for recommendations, with a specific scope in the nutrition field. We then present in Section 3 how we created the dataset needed to learn flavorful vector representations, followed by the training methodology and architecture in Section 4. In Section 6, we further describe the experiments we conducted to assess our vectors’ palatability. We conclude this paper with possible future works in Section 7.

\(^3\)https://untappd.com/
\(^4\)https://www.ratebeer.com/
\(^5\)https://www.vivino.com/
\(^6\)https://www.connors.com/
2 RELATED WORK

Recent advances in learning dense representations for words are using shallow neural networks such as GloVe [9] and word2vec [7]. The latter model’s name and architecture led to a plethora of “2vec models” [7]. From these models, we can find vector representations for code, emojis, time, food, and even names [2], to name a few.

It has been shown that vector representation learned by following a training scheme similar to what [7] proposed conveys a certain analogical reasoning [8] i.e., the possibility to perform vector arithmetics (to some extent [5, 6, 10]). Nonetheless, vector representations are deemed useful in recommender systems by capturing semantic similarity among terms in a vocabulary [11]. The learning procedure has also been applied to learn metadata embeddings to provide better cold-start recommendations for users amongst a set of items [3, 4]. We believe this recommendation technique can improve the idea of food recommendations presented in [1].

However, while being probably the alcoholic beverage the most consumed across the world, we could not find vector representations dedicated to the beer universe. This absence is perhaps due to the lack of datasets for this specific case. We thus introduce in the following section how we created such dataset to train yet another flavor of word2vec, Beer2vec.

3 DATASET CREATION

The dataset we used in the experimentation was obtained from untappd in 2018 and consists of check-ins, the user’s appreciation of a beer. The excerpt of data contains a total of 500,000 check-ins for over 146,000 distinct beers.

The check-ins represent user input for a specific beer. The check-in contains metadata to identify the beer, such as the beer name and the brewer’s name. The check-ins also include information regarding the user’s beer appreciation, such as a text review and a tasting score, but more importantly, a list of standardized flavor tags drawn from a list of 264 predetermined elements. These tags indicate the positive and negative aspects of the many facets of beer taste, namely malt, hop, and yeast. In our experiment, we only used the flavor tags since they are rich and easy to use.

We evaluated the feasibility of using the text portion of the review. However, after inspecting the quality of the written reviews, we had an important doubt and decided not to consider them in our analysis.

4 BEER2VEC

Beer2Vec is heavily inspired by the Skip-Gram architecture introduced by [8]. Given the set of beers $\mathcal{B}$ and the set of flavors $\mathcal{F}$, Beer2Vec is parameterized by two embedding matrices; $\Theta = \{\mathcal{B} \in \mathbb{R}^{|\mathcal{B}| \times k}, \mathcal{F} \in \mathbb{R}^{|\mathcal{F}| \times k}\}$, which are the beers and flavors embeddings respectively, and where $k$ is a hyperparameter setting the dimension of both matrices.

Given the set of check-ins $\mathcal{C}$ that constitutes our training set, where every check-in $c_i$ contains a beer $b_i$ and a set of flavors $f_i$, the Beer2Vec model tries to maximize the average log probability of the following function:

$$ J = \frac{1}{|\mathcal{C}|} \sum_{i=1}^{|\mathcal{C}|} \log p(f_i | b_i). \quad (1) $$

More concretely, Beer2Vec learns to predict flavors $f_{ij}$ given a beer $b_i$. Similar to [8], $p(f_{ij} | b_i)$ is expressed using the softmax function and we thus optimize the following loss function w.r.t to the parameters $\Theta$:

$$ J(\Theta) = \frac{1}{|\mathcal{C}|} \sum_{i=1}^{|\mathcal{C}|} \sum_{j=1}^{|\mathcal{F}|} \frac{\exp(b_i^\top f_{ij})}{\sum_{m=1}^{|\mathcal{F}|} \exp(b_i^\top f_m)}. \quad (2) $$

In practice, we use a dimension $m$ of 5, Adam as the optimizer with a standard learning rate of 0.001, a batch size of 128, and train the model for a maximum of 300 epochs.

5 APPLICATIONS

We present three applications of Beer2Vec in terms of recommendation and information retrieval. The first one allows customers to select the most appropriate beverage while visiting a beer store. The second allows him to search for beer based on a custom-made flavor profile. The last one allows a person drinking a beer to obtain the correct words to describe its beer.

5.1 Select a Beer when Visiting a beer store

Whenever this customer visits a beer outlet and requires a recommendation, he simply compares the vector representations (using the dot product) of available beers and the ones from his favorite beers. The customer sorts the available beers using the calculated distance metric to obtain the most similar beverages compared to his favorite list, hence providing a flavorful recommendation. For example, let $B_i$ be the current set of beers available in the outlet and $U_j$ be the customer’s set of favorite beers, a customer can obtain the $n$ most interesting beers using the following retrieval formula:

$$ b_j = \arg\text{sort}(B_i \times U_j^\top) \quad \text{for} \quad n $$

5.2 Search Beer by Flavor Profile

The second way of obtaining a recommendation is by creating a flavorful recommendation based on its own preferences. In this application, a user selects flavors, hereby identified as $f$, and uses vector arithmetics to combine them. One easy way to combine such vectors is by averaging Beer2Vec vectors using custom weights to create something unique. Finally, using the dot product, the user compares its created flavor profile vector with all the beers available in the beer2vec database and sorts by distance to find the one with the most similar to its request. Similar to the previous use case, we obtain the set of desired beers using the following retrieval formula:

$$ b_j = \arg\text{sort}(B_i \times \left( \sum_{i=1}^{|f|} a_i \cdot f_i^\top \right)), \quad \text{for} \quad n $$

As listed here; https://gist.github.com/mrw0381/333acfc00bd508501268fa7b40e4e4e. there are 130 and counting 2vec models.

Pun intended.
with \( \sum_{i=1}^{n} a_i = 1 \).

### 5.3 Obtain Vocabulary to Describe a Beer

Another way to exploit the Beer2Vec model is to use it to decompose a beer vector into its flavor components. This allows a user to compare the vector representation of the beer with the flavors in a database using the dot product. Ranking the flavors by their proximity enables the user to express its beverage with connoisseur vocabulary. Working the other way around, we retrieve the set of \( n \) flavors given a beer’s vector representation \( b_j \):

\[
    f_i = \text{arg}_\text{sort}(F_i \times b_j)
\]

### 6 MODEL EVALUATION, RESULTS AND ANALYSIS

We evaluate our model on two axes. The first evaluation axis regards will qualitatively evaluate the recommendations’ quality. The second axis of evaluation is the relevance of the vectors in recommendation quality, amount of flavor embedded in the beer embedding, and how these embeddings are robust to mathematical operations. Taste is very subjective by nature and a seasoned homebrewer with over ten years of serious experience will qualitatively evaluate the recommendations’ quality.

#### 6.1 Taste Vector Visualisation

To give us a qualitative sense of the embeddings learned, we project the \( F \) matrix in a two-dimensional plot in Figure 1 using the principal component analysis. The vector projection presented generates a flavor map that represents the flavor in beer in a (surprisingly) realistic way. We clustered the taste by their source using marker shapes. The analysis presented below shows how our taste vectors capture the information.

![Figure 1: A 2D Projection of the learned flavor matrix over our training set, \( F \). We can identify several clusters of flavors that interpolate from one to another.](image)

The first element to observe is the large flavor spectrum brought by hops in beer. The transition is shown along the x-axis: on the left side are sharper aspect of the hop notes (citrus, grapefruit, orange) and on the right side are the rounder flavors (pinnapple, passion fruit, mango, coconut). This wide array of flavors corroborates the idea that hops bring the most taste to the beer.

The second element one notices is how the yeast impact is condensed. This observation reflects another expected taste in beer: yeast flavor, except for sours and Belgian beer, should be tamed. Otherwise, those taste would be seen as off-flavors (phenolic, medicene, apple, booz), and are outside the yeast impact region.

The third element to observe is the barrel-aging effect that is isolated and separated from the other flavors. This is not surprising as barrel-aging is not a common practice in beer making and is an additional process that any beer can go through. However, beers that are the most improved by barrel-aging are stouts. Stouts use a lot of specialty grain that imparts flavors such as oatmeal, roasted, coffee, chocolate. These flavor types are the closest to the barrel-aging region and enhance our confidence in our methodology.

#### 6.2 Baseline comparison

The baseline we selected in our experiment is the PCA, an unsupervised algorithm that extracts the most salient information into vectors. Comparing our method to this baseline is natural since both approaches aim to encode data into dense vectors. We use the same amount of components as our model’s embeddings size (i.e., 5) from Section 4. The individual items for the input matrix \( C \) are given by

\[
    C[x, y] = \text{count}(f_y \in C_x),
\]

where \( f_y \) is the \( y^{th} \) flavour in \( F \) and \( C_x \) the subset of check-in that contains the beer \( b_x \), the \( x^{th} \) beer from \( B \).

To compare against our method, we selected three beers that cover a wide array of flavors, and we found the three most similar beers for both models, as illustrated in Table 1. The selected beers are an IPA (juicy and hoppy), a Pilsner (crisp and aromatic), and a stout (malty and sweet). The similarity metric used in this context is the dot product between \( b_i \) and \( b_j \), the vector representation of beer \( b_i \) and \( b_j \). The three beers with the highest similarity score are presented for each approach.

As one can see, the Beer2Vec model performed very well on those beers. Most beers are in line with the style of the base beer. On the other hand, the baseline performs very poorly. In the case of the Catnip, only one recommendation out of three was an IPA. The
others are either a black lager (crisp) or a Gose (sour), which is totally off the expected flavor profile. The conclusion is similar for the two other beers, where the PCA model recommended Stouts for the Pilsner, and a Lager and Bitter ale for the Stout. The Beer2Vec model recommended, unsurprisingly, beer very close to the requested type. We experimented with the PCA with a higher number of principal components, and the results were as overwhelming as they are presented in Table 1.

6.3 Recommendation Quality
The second element to validate is the value of the recommendation given by our model. Even if the Beer2Vec recommendation from Table 1 were adequate, two elements are worth discussing. The first one is the wheat beer recommendation for the Pilsner. Wheat beer has a usual ester character (banana) that is unwanted in a Pilsner. However, the joufflue contains lemon, hinting to the crisp mouthfeel common to every Pilsner. The second one is the SpécialB, a Belgian Quadrupel recommended for the Stout. Belgian beers have a highlighted fruity profile that is not usually found in Stout. Upon investigation, all those beer had a strong Chocolate, Coffee, Toffee taste profile.

These two observations support the hypothesis that flavors are not one-hot encoding: most of them overlap, and our models capture this uncertainty and subjectivity in flavor detection by design.

6.4 Flavors Embedded in Beer Embeddings
To further support our point, we extracted other recommendations to ensure extensive flavor tag coverage. Table 2 presents five additional beers with two recommendations, each accompanied by the recommended beer’s prevalent flavors. To obtain the most prevalent tags, we computed the dot product between the beer’s vector representation and every flavor representation, i.e., F. We kept the three vectors with the highest score. This table’s impressive results confirm the flavor clustering intuition from Figure 1. One must note that, even if the prevalent taste profiles are not the same for all recommendations, the various aspects of the base beer’s full profile are present in the proposed beer and would, in the aggregate, compose the vector of the base beer style.

This experiment shows that our method allows us to align vectors so that aggregation (beer) can be explained by its constituents (flavor).

6.5 Flavour Arithmetics
One final evaluation scheme would be to verify how flavor arithmetic works within our framework. To achieve this, we start from one beer vector \( b_a \) we subtract the flavor vector \( f_i \) and add the flavor vector \( f_j \) such that \( b_{a'} = b_a - f_i + f_j \). We then find the most similar beer to \( b_{a'} \) in our database. Table 3 presents 5 examples of flavour arithmetics.

As one can see, the results are not conclusive enough to assert that our algorithmic method works flawlessly. Some results are very interesting, such as the first three examples. The last two are not exactly where they were expected. For instance, The fourth example should be closer to an Oud Bruin or Red Flander, which are two sweet and sour beers. However, the American Wild Ale, a dry and sour beer, is acceptable. The last one, Jetlag does not work at all. The brown ale, very English type, is far from the added belgiany flavor. One probable cause of the problem is the scarcity of the belgiany tag and the small number of check-in for the jetlag beer.

6.6 External User Validation
Our validation scheme relies heavily on the expertise of a brewer. We encourage external users to validate these vectors’ quality by trying the beer2vec web application for themselves. The reader can find the application at the following address www.anonymous.ca.

This application contains the beer2vec database used for this paper and the following use cases:

- A user can select a beer and obtain its associated flavors;
- A user can select a beer and obtain its associated flavors;
- A user can apply flavor vector arithmetic on a base beer (as presented in Figure 2) and obtain a recommendation.

Although the beers contained in the dataset are limited to the available data points we had for our experiment, additional beer can be easily obtained with more beer check-ins and recalculating the flavor vectors using the training procedure discussed in Section 4.

| Base Beer + Recommendation | Prevalent flavour |
|-----------------------------|-------------------|
| Ta Meilleure (IPA)          | Refreshing, Fruity |
| Les Têtes Boisées (Grisette) | Hoppy, Fruity |
| Moralité (IPA - American)  |                  |
| IPA Du Nord-Est (NEIPA)     | Peachy, Hoppy |
| Ta Plus Meilleure (NEIPA)   |                  |
| L’Escouade B-13 (NEIPA)     | Juicy, Hoppy |
| Belle Saison (Saison)       | Sweet, Floral |
| Benedictus (Belgian Tripel) |                  |
| Blanche Beaumont (Wheat beer)| Yeasty, Dry |
| Rouge de Méninac (Flanders Red Ale) | Fruity, Sour |
| Genèse Sauvignon (Belgian Tripel) | Sour, Fruity |
| Solstice d’été (Berliner Weisse) |                   |
| Hudson Bitter (English Bitter) | Smooth, Malty |
| Boréale Rousse (Red Ale)    |                  |
| Doppelsticke (Altbier)      | Caramel, Bitter |

Table 2: Most Similar Beers According to Beer2Vec with their respective prevalent flavors.

7 CONCLUSION AND FUTURE WORK
This paper presents the beer2vec application that allows users to obtain beer recommendations based on flavor analogies. We presented the algorithm through an analysis of a dataset and presented relevant examples of its functionality.

We intend to expand our application to identify similar breweries, providing an insightful first step to our analogies and helping, for example, the touristic industry to develop personalized beer routes for specific tasting types.

9Website will be publicly available upon acceptance.
Beer2Vec: Extracting Flavors from Reviews for Thirst-Quenching Recommendations

| $\mathbf{b}_a$ | $f_i$   | $f_j$       | $\mathbf{b}_a$                |
|----------------|---------|-------------|-------------------------------|
| Catnip (IPA)  | Tropical| Caramel     | La Marquis De Vaudreuil       |
| Porter Baltique (Porter) | Coffee  | Brett       | Wallonade (Belgian Golden)   |
| Full Time I.P.A. (IPA) | Malt    | Tropical    | Mosaika (Pale Ale - English) |
| Sentinelle (Kölsch) | Clean   | Sour        | Gaspésie Sauvage (American Wild Ale) |
| Jetlag (Helles) | Crisp   | Beligiany   | La Marquis De Vaudreuil (Brown Ale - English) |

Table 3: Illustration of flavors arithmetic. Given a beer vector $\mathbf{b}_a$, we subtract one of its flavor $f_i$ and add a new flavor $f_j$ to obtain a new abstract beer vector, $\mathbf{b}_a$. This vector is then used to retrieve the most similar beer in the beer dataset.

Figure 2: The web interface where the users can build themselves a beer. They begin by selecting a "base beer", and then add and/or remove selected flavors accordingly. A suggested list of drinks is then provided by our model, along with its overall appreciation among beer geeks.

We believe this paper opens the exploitation of formal and standardized tasting cards such as WSET\textsuperscript{10} to infer flavor vectors. Many official tasting practices, such as sommeliers from governmental alcohol agencies, have this data readability for many products.

REFERENCES

[1] El-Dosuky, M., Rashad, M., Hamza, T., El-Bassiouny, A.: Food recommendation using ontology and heuristics. In: International conference on advanced machine learning technologies and applications. pp. 423–429. Springer (2012)

[2] Foxcroft, J., d’Alessandro, A., Antonin, M.: Name2vec: Personal names embeddings. In: Canadian Conference on AI (2019)

[3] Gáspár, P., Kompan, M., Koncal, M., Bieliková, M.: Improving the personalized recommendation in the cold-start scenarios. 2019 IEEE International Conference on Data Science and Advanced Analytics (DSAA) pp. 606–607 (2019)

[4] Kula, M.: Metadata embeddings for user and item cold-start recommendations. ArXiv abs/1507.08439 (2015)

[5] Levy, O., Goldberg, Y.: Linguistic regularities in sparse and explicit word representations. In: Proceedings of CoNLL 2014. pp. 171–180 (2014)

[6] Linzen, T.: Issues in evaluating semantic spaces using word analogies. In: Proceedings of the 1st Workshop on Evaluating Vector Space Representations for NLP. pp. 13–18 (2016)

[7] Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)

[8] Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: NIPS (2013)

[9] Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: EMNLP (2014)

[10] Rogers, A., Drozd, A., Li, B.: The (too many) problems of analogical reasoning with word vectors. In: Proceedings of StarSEM 2017. pp. 135–148 (2017)

[11] Rudolph, M., Ruiz, F.J.R., Mandt, S., Blei, D.: Exponential family embeddings. In: NIPS (2016)

\textsuperscript{10}https://www.wsetglobal.com/