# Bayesian Deep Learning for Small Datasets: Leveraging Information from Product Pictures

Remi Daviet

Wharton Marketing Department, University of Pennsylvania

August 4, 2020

**Abstract**

Marketers are often confronted with datasets that contain many variables but are limited in the number of observations, leading to a"large P, small N" problem. With unstructured data, such as product pictures, commonly used deep learning models require the estimation of a large number of parameters, also resulting in a "large K" problem. In this research, we propose a pipeline to process and exploit such unstructured data. We apply it to a novel dataset aggregating all retail sales of distilled spirits in Pennsylvania. We first reduce the high pixel-based dimensionality of the product pictures using a Conditional Generative Adversarial Variational Auto-Encoder (CGAVAE). We then use the result in a deep learning model to predict sales volumes, using Bayesian estimation to mitigate overfitting issues. We show that using the product pictures' information, in addition to traditional variables such as price and product characteristics, increases the out-of-sample prediction performance for sales volumes by nearly half its base value ($R^2$ increasing from 0.24 to 0.35). We also propose a method to interpret the results and identify relevant product features, potentially allowing for the creation of new theories. Lastly, we use our model in a design optimization exercise, where we identify classes of bottle designs that are predicted to maximize expected revenue.

**Keywords:** Bayesian Neural Networks, Machine Learning, Product Pictures, Retailing.

**Acknowledgments:** We are very grateful to the Pennsylvania Liquor Control Board for providing us with their sales data and their assistance in processing it. We also warmly thank Gideon Nave, Eric Bradlow, Juanjuan Zhang, Fred Feinberg, Longxiu Tian, Krista Li, Xiao Liu, Puneet Manchanda, Alex Burnap, and Ryan Dew for the numerous useful discussions and the critical advice provided.

## 1 Introduction

The last few decades have witnessed an exponential increase in the accumulation of data. These datasets have allowed firms and marketers to leverage the information they contain to improve targeting, forecasting and their understanding of consumers. However, a lot of the available data are high dimensional (*Large* $P$), and unstructured, that is, it does not have a natural data model or it is not organized in a pre-defined manner. Machine learning models, and in particular deep learning models, have been developed and show impressive results for a multitude of tasks when applied to these data. However, these models are usually designed with a large number of parameters to estimate (*large* $K$), and thus require large datasets to be trained (*large* $N$). In marketing, these large datasets are not always available and we are presented with a "large P, large K, small N" statistical problem. For instance, the number of products (SKUs) on a given market might be limited (e.g., cars, breakfast cereals, etc.), or the number of observations might be small (e.g., we observe sales for a small number of locations).

Among the several types of unstructured data available to marketers, product pictures are an important factor influencing consumer demand (Underwood and Klein, 2002). Automating the extraction of meaningful information could have important consequences for the field of marketing. In this paper, we propose a pipeline, represented in Figure 1, to effectively capture and exploit information from limited datasets of product pictures typically available in marketing applications. We achieve this result by proposing novel methods based on state-of-the-art advances from the deep learning and Bayesian inference literature. We first reduce the dimensions of the unstructured data (Step 1) through the combination of a *Conditional Variational Auto-Encoder* (CVAE) and a *Conditional Generative Adversarial Network* (CGAN). To our knowledge, it is the first time that the conditional versions of these two technologies are combined in such an architecture, and we name the method *Conditional Generative Adversarial Auto-Encoder* (CGAVAE). We then adopt a Bayesian deep learning approach, training a neural network to predict the quantity of interest (e.g., sales volumes) from the compressed representation of the product pictures (Step 2). This Bayesian perspective allows us to quantify model uncertainty and to avoid overfitting problems typically faced when fitting a flexible model to a small dataset.

Traditionally, a high dimensional input is reduced to a few meaningful features guided by theory. The features are then used in a model to link them to the outcome of interest. For instance, the characteristics of a car can be summarized into model year, MPG, horsepower, weight, and manufacturer. However, it might not always be clear what features are important in a product picture, and manually labeling the pictures one by one can be extremely costly and potentially inaccurate. Indeed, this labeling task often requires expertise and some features might be subjective (e.g., beauty, attractiveness, etc.). Automating the process and allowing for fast large-scale labeling to reduce the input dimensionality could save a consequential amount of time, effort, and money to companies. The other common approach to reduce dimensions is to rely on data-driven methods such as *Principal Components Analysis* (PCA), which identifies the *linear projection* to a lower

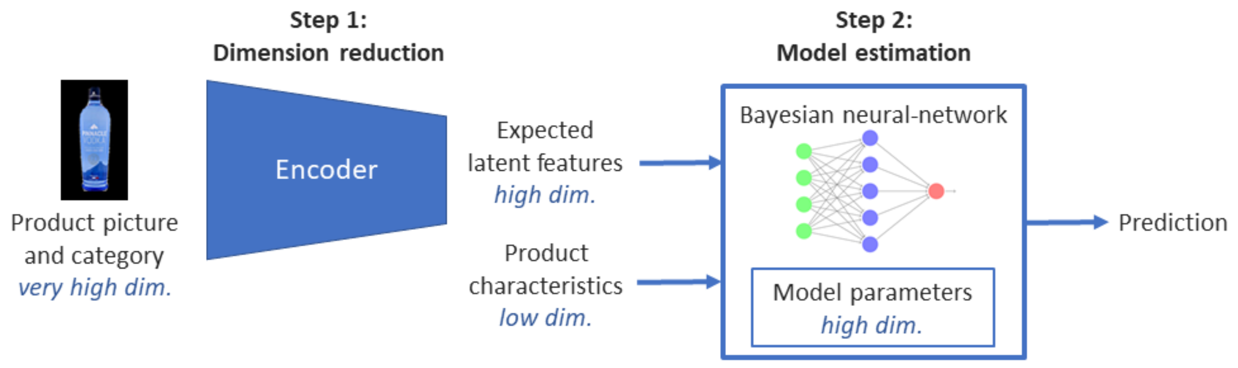

Figure 1: Data processing and estimation pipeline.

dimensional space that minimizes information loss. However, PCA decomposition of pictures has been known to lose a consequential amount of information and to produce low quality reconstructed images as a result (e.g, Calder et al., 2001). Moreover, the space of principal components generated is hardly interpretable in a meaningful manner, as shown in Section 3.1. In this paper, we use instead a CVAE approach as they constitute the state-of-the-art in terms of theory-free dimensionality reduction for pictures. CVAEs are a powerful class of non-linear models allowing one to obtain a compressed representation (the *latent features*) of a high dimensional input. In addition to reducing the dimensionality of the product pictures, the CVAE allows us to generate new product pictures by navigating the newly created space of latent features. In parallel, we use a CGAN approach, by training a deep convolutional neural network to classify if an image is an original picture or has been generated by the CVAE. The CVAE loss has thus two main components: first, a mean squared error component to measure the discrepancy between the original product picture and the reconstructed one; second, the classifier's estimated probability that an image is not an original one, to force the CVAE to generate pictures sharing the characteristics of real pictures.

While the latent features obtained in Step 1 are of a lower dimensionality, they still require complex non-linear transformations to predict effectively the quantity of interest (e.g,. sales volumes). We thus implement in Step 2 a deep neural network using the latent features, as well as traditional variables such as price and product category, for the prediction task. Deep neural networks have a high dimensional parameter space, and thus can easily overfit small datasets when using an optimization-based approach (e.g., maximum likelihood), reaching near perfect prediction in-sample, but very poor prediction out-of-sample. Traditionally, this overfitting phenomenon is mitigated by introducing a prior, shrinking the parameters of the neural network toward 0, a technique known as regularization. The prior, however, might introduce a bias, and, as shown in this paper, a false sense of confidence concerning the prediction accuracy. Following a Bayesian perspective, we propose to sample from the full posterior distribution of parameters, instead of maximizing over the parameter space. We show that using the expected value as a predictor substantially mitigates the overfitting issue and increases out-of-sample accuracy. Computational issues arise from the high-dimensionality and non-linearity of the model. The high-dimensional posterior distribution is unusually complex (heavily multimodal) and hard to sample from (Li et al., 2018). We thus propose a novel and computationally efficient stochastic version of the Hamiltonian Monte-Carlo (HMC) approach (Neal, 2012). We leverage automatic differentiation technology largely used in deep learning (Baydin et al., 2017), combined with sub-sampling and gradient correction, to approximate the objective function's gradient.

We apply our method to a dataset of 1,483 bottles pictures of alcoholic spirits and their corresponding sales volumes in the state of Pennsylvania for the financial year of 2018. As alcoholic spirit retail is a state-run monopoly in Pennsylvania, our dataset accounts for all the sales to individual consumers in the state. We first reduce the 18,432-dimensional product pictures to a 256-dimensional latent features vector, as we find that reducing the dimensions further does not preserve sufficient information for satisfactory image reconstruction. We then train a deep neural network to predict the monthly sales volumes for each product. The deep neural network first projects the latent features vector onto a 2-dimensional plane, before injecting the new representation into a fully connected layer that combines it with other traditional variables. The 2-dimensional projection allows us to obtain a representation that is easily interpretable by a marketer, via visual identification of clusters and their characteristics. This identification can be particularly useful to define relevant features, and potentially lead to the development of new theories. We show that our method improves sales volume prediction by increasing the out-of-sample $R^2$ from 0.24 to 0.35, when compared against a generalized linear model relying only on traditional variables, such as price or product attributes (e.g., bottle capacity). We also use our model in an optimization exercise, where we identify classes of bottle designs that maximize the expected sales volumes for a given product.

It is important to note that since our application is realized on observational market data, it is not meant to draw any causal conclusion. Instead, our method should be considered as a type of automated *exploratory data analysis* and *data pre-processing*. This should facilitate the tasks of marketing managers as they are inundated by unstructured data, and conventional labor-intensive data processing has become cost-prohibitive. Once candidate patterns have been identified in the data, they can be further investigated using causal research methods. In our case, the proposed pipeline allows us to determine a set of candidate features for the product pictures that could have a causal impact on the outcome variable. Since these candidate features are of low dimensionality, their causal effect can easily be assessed in an experiment with a small sample requirement, effectively eliminating the "large P, small N" issue we had at the beginning.

The remainder of the paper is as follows. We first define the general research problem and review the relevant literature in Section 2. We then describe our proposed method in Section 3, introducing the state-of-the-art dimensionality reduction method (Section 3.1), followed by the Bayesian deep learning approach (Section 3.2). We show the effectiveness of our approach with simulated data in Section 4. We then present our application to the alcoholic spirit market in

Section 5, with a first part discussing predictive performance, and a second part covering results interpretation through visualization. We apply our fitted model to a design optimization exercise in Section 6, generating a series of new bottle designs as a result. Finally, we discuss other potential applications of our method and future research in Section 7. Section 8 concludes the paper.

## 2 Related Research and General Problem

In this section, we describe the problems associated with including information from high-dimensional product pictures in the researcher's model. We review various existing approaches, as well as their benefits and shortcomings. We also discuss why we believe that the limited dataset sizes available in marketing have kept advanced data-driven methods mostly absent from our literature. Our proposed approach aims at making these methods applicable, even with smaller datasets.

Product pictures convey a lot of implicit information to consumers, and affect their behavior as a result (Underwood and Klein, 2002). Marketers can leverage some of this information to improve prediction, optimize presentation, and understand decision mechanisms. Visual representations of products are, for instance, used in visual conjoint analysis to predict market shares (Dahan and Srinivasan, 2000). When the set of product pictures is small, such as when comparing images of a dozen of prototypes, each picture can be evaluated individually and rated accordingly. However, when the set of products is larger, describing the pictures with a selection of relevant features becomes necessary. The smaller set of relevant features allows the researcher to reduce the dimension of the space of possible pictures to a manageable number of variables that can in turn be integrated in quantitative or behavioral models.

This feature-based approach faces three main challenges: *identification*, *selection*, and *labeling*. First, the set of potential features to consider needs to be identified (identification), which might be a difficult task as the space of features is potentially extremely large (there are infinite ways to describe the pixels of a picture). Among the set of all potential features, the subset of features relevant to the problem at hand needs to be selected (selection). Finally, once the subset of relevant features are identified, each image needs to be classified or rated according to these features (labeling), which could require consequential resources and be poorly scalable if the process is not automated. These issues are not unique to product pictures data, and are also studied in the related fields of design and packaging. We thus include in this section some relevant research from these fields.

There are two main approaches to feature identification. The first one is theory-driven, where industry experts and researchers identify features relevant for a particular issue. The second approach is data-driven, sometimes referred to as feature learning, and relies on statistical methods for the task. Both of these methods have advantages and shortcomings that we discuss in this section.

Traditionally, marketers could rely on theory or industry practices to identify features to consider including in their model. Some features might be general to all types of products and applicable in all contexts. For instance, in the field of product packaging, truthfulness, sincerity, comprehensibility, and legitimacy have been proposed as a general set of features (Underwood and Ozanne, 1998). Other general features might come from the more distant fields of psychology, neuroscience or art. For instance, a marketing practitioner could consider including visual salience (Jarvenpaa, 1990; Lans, Pieters, and Wedel, 2008), color choice and coordination (Labrecque and Milne, 2012; Singh, 2006), or location of textual and pictorial elements (Rettie and Brewer, 2000) in her analysis. Besides general features, one might want to rely on product-specific features. For example, Han et al. (2004) identified that for cellphone design, features such as number of exposed buttons, body length, body shape, and number of colors for various parts, are critical (among other features) for consumer satisfaction. One fundamental issue is that the product of interest might not have clear features identified in the literature. Moreover, even if features are identified, they might be neither comprehensive, nor the most relevant for our particular application.

With all the potential features to choose from, marketing practitioners might be overwhelmed by the number of available features and struggle to select the relevant ones. For instance, Burnap and Hauser (2018) identified 2,428 features applicable to car design that could potentially be used to predict demand, while having only 297 products in their dataset. To select the relevant features, experiments and surveys can be used (e.g., Silayoi and Speece, 2007). However, the number of required observations grows rapidly as the set of candidate features expands, consequently increasing the required data collection efforts. Moreover, all the features need to be labeled before analysis, which presents another set of issues discussed later in this section. In this paper, we propose to use available market data, instead of costly surveys, to identify a set of candidate features for the marketing problem considered. The relevance and causal effect of these few features that can then be confirmed experimentally at a later stage at minimal costs.

Once a list of features is identified, pictures need to be classified or rated accordingly (labeling). While some features might be clearly and objectively identifiable (e.g., number of pockets on a shirt), they could require a human intervention for each picture to be labeled. When the features are perceptual and subjective (e.g., perceived beauty, quality, or safety of a product), one expert might not be sufficient and companies might rely instead on using several ”experts” to achieve a consistent and stable labeling, increasing the costs even further. A typical approach for subjective evaluation of pictures is to use crowdsourcing and to average the multiple evaluations. For instance, this approach is used to rate pictures according to their generated emotions (Lang, Bradley, and Cuthbert, 1997), or to evaluate the perceived safety of a car from its body shape (Ren, Burnap, and Papalambros, 2013). However, even when using crowds, responses might be highly variable in quality, and identifying the relevant experts is a task that is subject to active research in itself (Burnap et al., 2017). Overall, the feature identification and labeling process might be extremely resource and time intensive for companies. Note, however, that the labeling of pictures can be automated and used at a large scale. For instance, Zhang et al. (2020) trained a Convolutional Neural Network to classify 510,000 AirBnB images according to 12 theory-driven dimensions representative of aesthetic qualities. However, even when labeling tasks can be automated, most algorithms still require a large initial manually labeled training set. For complex features, the required training set size becomes excessively large and might thus be unavailable given the limited quantity of products in a particular market.

The second approach, data-driven feature identification, relies on statistical methods to reduce the picture dimensionality to a set of latent features and already benefits from a large body of research in the machine learning literature (e.g., Coates, Ng, and Lee, 2011; Netzer et al., 2011). Data-driven methods have the advantage of automatically identifying features and labeling each image. They consequently are less resource intensive and more scalable than their theory-driven counterparts. They present however three major shortcomings. First, the identified latent features might not be readily interpretable. Second, the space of latent features, typically in the hundreds, remains quite high-dimensional relative to the number of observations that might be available. Third, the more advanced methods require large image datasets just to be effectively trained. In the best scenario, the dataset at hand is large enough. However, in a large number of marketing applications, both the number of pictures (e.g., SKUs) and the number of observations of the dependent variable (e.g., monthly sales volumes) can be limited. This small N issue rules out most of the traditional machine learning methods that we discuss here. When the dataset is too small, practitioners usually rely on algorithms trained with larger datasets first, and then apply the trained algorithms directly, or with small modifications, to the dataset of interest, a method know as transfer learning (Pan and Yang, 2009). This method is effective when the feature space of the large dataset is similar to the feature space of target dataset. While the number of publicly available datasets has grown in the last decade, it remains hard to find a dataset suitable to each marketing application. For instance, it is unlikely that any given dataset will have hundreds of thousands of pictures of digital cameras or breakfast cereals.

In terms of existing statistical methods, one straightforward approach is to use PCA directly on available pictures to reduce the number of dimensions (e.g., Calder et al., 2001). However, PCA has several shortcomings that are discussed in Section 3.1. More advanced methods, stemming from deep learning and computer vision, have shown impressive results in their ability to reduce pictures to a small set of meaningful values. One popular class of algorithms are auto-encoder and their variations, which can be considered as a non-linear extension of PCA (Kramer, 1991). A notable extension of this class proposes a Bayesian approach to the latent feature identification problem and is known as *Variational Auto-Encoder* (VAE; Kingma and Welling, 2013; Rezende, Mohamed, and Wierstra, 2014). It has shown good results for encoding images into a manipulable latent feature space, and also allows for generating new images from this latent space (e.g., Bouchacourt, Tomioka, and Nowozin, 2018; Hou et al., 2017). In addition, VAEs make possible the generation of new product pictures from the latent feature space, a property known as generative learning. However, the most effective technique to generate realistic pictures is to use *Generative Adversarial Networks* (GAN), where two neural networks are trained in parallel (Goodfellow et al., 2014). The first network, the generator, tries to generate pictures from a latent feature space, while the second network, the discriminator, is trained to classify these generated images, as well as a set of real product pictures, as being real or generated. The goal of the generator is to generate pictures that fool the discriminator into classifying them as real pictures. While GANs generate realistic pictures, they are not designed to map real pictures to the latent feature space. To accomplish this task, they can be associated with VAEs, resulting in a family of algorithms known as VAE-GANs (Larsen et al., 2016), which is the approach we adopt in this paper.

As of the writing of this paper, these feature learning methods remain mostly absent from the field of marketing. One notable exception is a method proposed by Dzyabura and Peres (2019) which first assigns 20 textual tags to each picture, using a machine learning platform trained on a corpus of millions of pictures. The feature identification is then done by identifying clusters of tags that are semantically close, using natural language processing algorithms also trained on millions of documents. These clusters can finally be manually labeled by an expert to obtain a meaningful interpretation of the results. Another important application is proposed by Burnap, Hauser, and Timoshenko (2019), who pretrained a VAE-GAN with 180,000 car design images and then used the generated latent features to predict the aesthetic appeal of new designs.

We believe that the large absence of advanced machine learning methods in marketing applications is mostly due to the lack of sufficiently large datasets. We thus focus our paper into making these data-driven methods applicable to datasets of a few thousand products. We first combine extensions of the VAE and GAN methods to drastically reduce the picture dimensionality to a few hundred dimensions while preserving a maximum amount of information given the dimensionality constraint. We then apply a Bayesian deep learning approach to identify the most relevant subspace of features, while mitigating the issues associated with small datasets. This offers the advantage of being both cost effective and easily scalable. It also has the benefit of automatically labeling pictures without human intervention. However, as our approach is data-driven, the latent features are not initially interpretable. We allow for interpretation at a later stage, once the relevant dimensions of the feature space have been selected with our Bayesian deep learning algorithm. Indeed, once classes of products sharing similar characteristics have been automatically identified, experts can look at a handful of pictures from a particular class and label the whole class at once. This process requires very little resources compared to manually labeling pictures one by one and is relatively insensitive to scale issues. A subsequent benefit of our method is that it has the potential to identify new features that are important for theory.

## 3 Methodology

We have discussed why the small size of the datasets typically available in marketing have limited the adoption of advanced data-driven features identification methods. In this section we propose a new method that is designed to achieve a good performance in these smaller datasets.

Our aim is to predict an outcome of interest $y_j$ (e.g., sales volumes) for each product $j = 1, ..., J$. To achieve this, we use the product picture represented by a $(height \times width \times colors)$ matrix $X_j$, as well as the product category $c_j \in \{1, ..., C\}$ and a vector of observed attributes $a_j$ (e.g., price). The dataset $\mathcal{D} = \{y_j, X_j, c_j, a_j; j = 1, ..., J\}$ is divided into three subsets: (1) a *training set* $\mathcal{D}_{train}$ used to fit the model, (2) a *validation set* $\mathcal{D}_{val}$ used to adjust tuning parameters, and (3) a *test set* $\mathcal{D}_{test}$ used to measure out-of-sample performance. The subsets of corresponding indices $j$ are respectively denoted $\{\mathcal{J}_{train}, \mathcal{J}_{val}, \mathcal{J}_{test}\}$, and the cardinalities $\{N_{train}, N_{val}, N_{test}\}$.

As represented in Figure 1, our pipeline includes 2 steps. First, we train a CVAE integrated into a CGAN to reduce the very high dimensionality of $X_j$ to a vector of latent features $z_j$ of a few hundred dimensions (step discussed in Section 3.1). Given our typical dataset size for a few thousand products, $z_j$ is still considered high dimensional. In a second step, we integrate the latent features $z_j$ and the other product attributes into a deep neural network to predict $y_j$ (discussed in Section 3.2). The neural network also selects an optimal low-dimensional projection of the latent feature space, maximizing the relevant information for the prediction task, and allowing for a meaningful interpretation of the results (see Section 3.2.3). Since both the input $(z_j, c_j, a_j)$ and the model parameters are of high dimensionality, we adopt a Bayesian inference framework to mitigate overfitting issues and preserve predictive power out-of-sample. We assess the performance of our method in Section 5 with an application to sales volumes prediction of alcoholic spirits. Some intermediate results from our application are used to illustrate the discussion in this section.

## 3.1 Conditional Generative Adversarial Variational Auto-Encoder (CGAVAE)

In this section, we discuss how to reduce the dimensionality of $X_j$ into a set of latent features $z_j$ that can describe a particular product picture. These latent features can then be used to reconstruct the product picture. We review various conventional approaches and compare their characteristics with the approach we propose. We also discuss how our method can be used to generate new product pictures by navigating the latent space generated.

There exist various data-driven methods to perform the dimension reduction, sometimes referred to as *encoding* or *compression*, and evaluate them on three criteria: *fidelity*, *image sharpness*, and *latent space navigation* (LSN). Our method, called *Conditional Generative Adversarial Variational Auto-Encoder* (CGAVAE), tries to balance the trade-offs between these various criteria while reducing effectively the dimensionality of the product picture.

Fidelity indicates whether the reconstructed product presents the same characteristics as the original product. A low fidelity reconstruction, might look like a realistic product, but present different characteristics (product shape, logo design, etc.). Sharpness indicates whether the reconstructed image is sharp or blurry. We show in Figure 2 an example comparing an original product with a low fidelity and a low sharpness reconstruction. The third criterion, LSN, is the ability of a method to reconstruct products with desirable graphical properties (realism, sharpness, etc.) when sampling points from the latent space that were not included in the original training set. For instance, an algorithm with poor LSN might generate unrealistic pictures when randomly

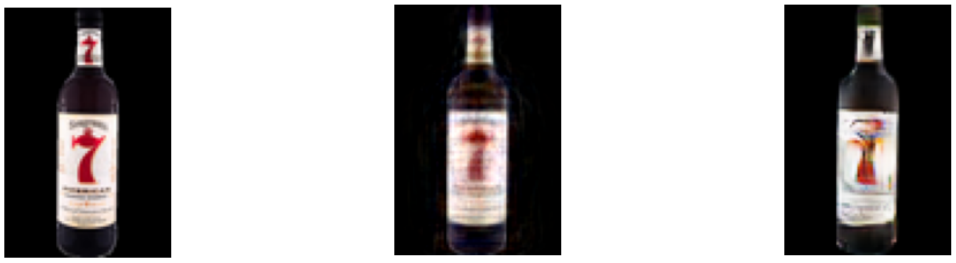

Figure 2: Comparison of an original image (left) with a high fidelity/low sharpness representation (middle), and a high sharpness/low fidelity representation (right).

exploring the space around the latent features of an in-sample product. For instance, we compare in this section the LSN properties of PCA and CGAVAE by showing the reconstructed images of an out-of-sample product (Figure 5), by randomly stepping around in-sample products' latent representation (Figure 3), and by interpolating between two in-sample products (Figure 6).

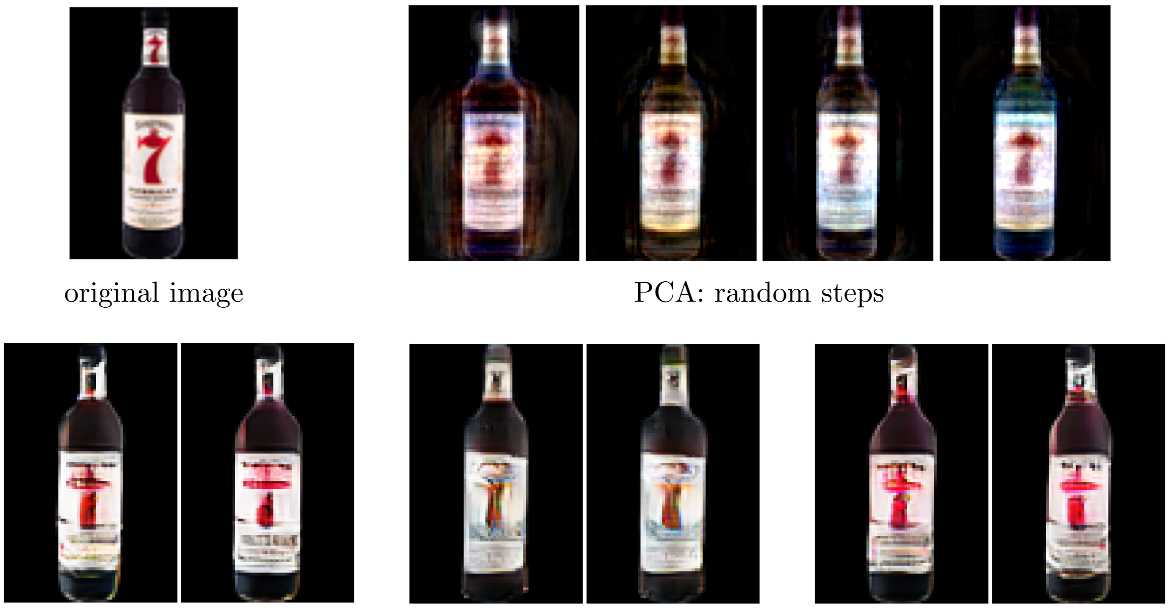

Figure 3: Reconstructed images from taking random Gaussian steps around the original image's latent representation. The steps' standard deviation is set to half the standard deviation of $z$ in each dimension. In the CGAVAE case (bottom), we also varied the product category.

Historically, PCA has been extensively used to reduce dimensionality while preserving information. While PCA has a high in-sample fidelity, it shows several shortcomings due to the linear nature of the projection to the latent feature space. First, the reconstructed images tend to be blurry (low sharpness). Second, as it is not a generative algorithm, PCA presents poor LSN performance. For instance, we show in Figure 3 (top right) the reconstructed images of various points sampled from the neighborhood of the latent representation of a real product picture. Instead of obtaining products with slight variations in their characteristics, we obtain a blurry representation of the original product with various graphical artifacts.

Like PCA, auto-encoders are another class of dimensionality reduction models and are described by two functions (usually non-linear): an encoder $f_e(X_j)$ mapping each picture to a latent representation $z_j$, and a decoder $f_d(z_j)$ mapping $z_j$ to a reconstruction of $X_j$. For encoding of complex pictures, the encoder and decoder are usually convolutional neural networks with parameters $\theta_e$ and $\theta_d$. Auto-encoders are trained to minimize the mean squared loss between the source image $X_i$ and its reconstructed counterpart:

$$Loss_{AE}(\theta_e, \theta_d) = \| X_j - f_d\left(f_e(X_j;\theta_e);\theta_d\right)\|_2^2 . \tag{1}$$

Compared to PCA, auto-Encoders present the advantage of taking into consideration non-linear dependence between pixels when searching for an optimal compressed representation of the picture, potentially preserving more information. In practice, they however present the same shortcomings as PCA: low sharpness of reconstructed images, and poor LSN.

With the goal of improving LSN and obtaining effective generative models, VAEs take a Bayesian approach to auto-encoders, and consider the latent representation $z_j$ to be an unknown quantity to be learned. As such, VAEs assume a prior distribution $p(z_j)$ over the latent space, usually a standard multivariate Gaussian distribution, and try to recover the posterior distribution $p(z_j|X_j)$ for each product picture. In other words, instead of producing a point estimate of the latent representation $z_j$ for each $X_j$, the variational auto-encoder defines a probability distribution over the latent space. The posterior distribution is approximated using variational Bayes methods, using independent Gaussian distributions for each dimension of the latent space. The encoder is thus a function mapping each $X_j$ to a set of conditionally independent Gaussian distributions approximating $p(z_j|X_j)$ and represented by the parameter vectors $(\mu, \sigma)$:

$$\text{Variational encoder:} \qquad \mu(X_j;\theta_{ve}),\ \sigma(X_j;\theta_{ve}). \tag{2}$$

Conversely, the decoder infers the expected $X_j$ given a latent representation $z_j$:

$$\text{Variational decoder:} \qquad E[X_j|z_j] = f_{vd}(z_j;\theta_{vd}). \tag{3}$$

The VAE has a two part loss function: the first part measures the dissimilarity between the reconstructed image $E[X_j|z_j]$ and its source $X_j$, and the second parts acts as a regularization component by measuring the Kullback-Leibler divergence between the prior and the approximated posterior. During training, a different set of $z_j$ is sampled from the distribution $N(\mu(X_j;\theta_{ve}), \sigma(X_j;\theta_{ve}))$ for each loss evaluation, resulting in a stochastic loss function:

$$Loss_{V1}(\theta_{ve}, \theta_{vd}) = \|X_j - f_{vd}\left(\mu(X_j; \theta_{ve}) + \sigma(X_j; \theta_{ve}) \cdot \epsilon_j; \theta_{vd}\right)\|_2^2, \qquad \epsilon_j \sim \mathcal{N}(0, 1), \tag{4}$$

$$Loss_{V2}(\theta_{ve}) = 0.5 \sum_{k=1}^{K} \Big(\sigma(X_j; \theta_{ve})^2 + \mu(X_j; \theta_{ve})^2 - 2 \cdot ln\ \sigma(X_j; \theta_{ve}) - 1\Big), \tag{5}$$

$$Loss_V(\theta_{ve}, \theta_{vd}) = w_{V1} \cdot Loss_{V1}(\theta_{ve}, \theta_{vd}) + w_{V2} \cdot Loss_{V2}(\theta_{ve}), \tag{6}$$

where "$\cdot$" denotes an element-wise product, and $(w_{V1}, w_{V2})$ are weights that the researcher sets to balance fidelity and LSN as discussed below. A conditional variation of the VAE, the CVAE, approximates instead the conditional distribution $p(z_j|X_j, c_j)$ and expectation $E[X_j|z_j, c_j]$, leveraging the information about the product category $c_j$. The main advantage of VAEs and CVAEs is that by assuming a distribution over the latent space, they improve the LSN properties of the model. The trade-off is that they usually lose some fidelity in exchange. The weights $(w_{V1}, w_{V2})$ can control this trade-off, with a higher $w_{V1}$ increasing fidelity, and a higher $w_{V2}$ improving LSN properties. In the extreme case where $w_{V2}$ is set to 0, the learned posterior distributions collapse to point mass distributions with $\sigma(X_j; \theta_{ve}) = 0$, effectively becoming a standard auto-encoder. In terms of sharpness, VAEs and CVAEs unfortunately suffer from the same low image quality as auto-encoders. Our solution is to integrate an adversarial loss from the GAN literature into a CVAE model, maintaining the CVAE advantages while increasing sharpness.

Among generative models, GANs have achieved incredibly sharp and realistic images (Radford, Metz, and Chintala, 2015). This high performance is obtained by training in parallel to the model generating the images, a second neural network $S(X_j; \theta_s)$ (the discriminator) estimating the probability that a picture has not been generated by another algorithm. During training, it is exposed to a mix of real and fake images and their label $(X_j, r_j)$, where $r_j$ is a binary variable set to 1 if the image is real, and 0 otherwise. In this paper, we use a conditional extension of the GAN, adding the product category $c_j$ as an input to the discriminator. While many loss functions have been proposed for the discriminator of a GAN, we retain a standard mean squared error loss, as proposed by Mao et al. (2017), as it does not make a significant difference on our dataset:

$$Loss_S(\theta_s) = \|r_j - S(X_j, c_j; \theta_s)\|_2^2. \tag{7}$$

We add the newly defined discriminator to the loss function of our CVAE, which acts as the generator. We obtain a conditional variation of VAE-GAN models (Larsen et al., 2016), that we call CGAVAE. During training, two types of generated images are used: first, some of the generated images $\widehat{X}_j$ are decoded counterparts of inputs $X_j$; second, an equivalent number of images $\widetilde{X}_j$ are generated by sampling latent features $\tilde{z}_j$ from the prior distribution, with corresponding product categories $\tilde{c}_j$ sampled uniformly over all categories. This process is represented in Figure 4. The

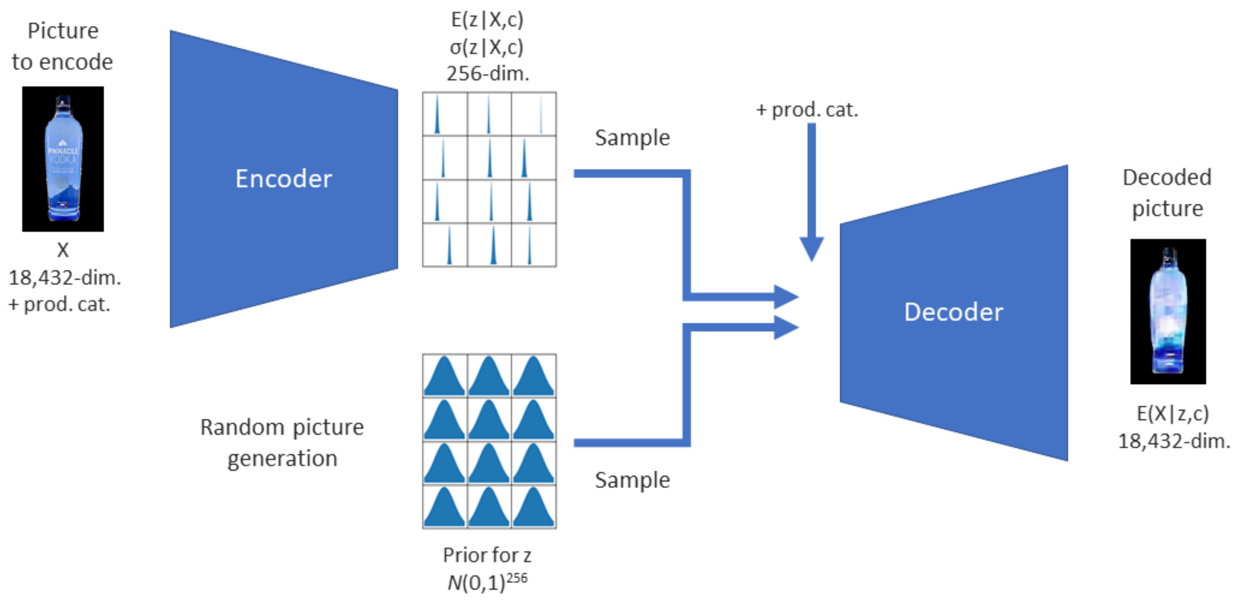

Figure 4: CGAVAE picture generation scheme. Variable indices have been omitted for clarity.

additional component to the CVAE loss is:

$$\widehat{X}_j(\theta_{ve}, \theta_{vd}) = f_{vd}\left(\mu(X_j;\theta_{ve}) + \sigma(X_j;\theta_{ve}) \cdot \epsilon_j;\theta_d\right), \quad \epsilon_j \sim \mathcal{N}(0, I) \tag{8}$$

$$\widetilde{X}_j(\theta_{vd}) = f_{vd}\left(\tilde{\epsilon}_j;\theta_d\right), \quad \tilde{\epsilon}_j \sim \mathcal{N}(0, I) \tag{9}$$

$$Loss_G(\theta_{ve}, \theta_{vd}) = -\frac{1}{J}\sum_{j=1}^{J}\left(\log S(\widehat{X}_j(\theta_{ve}, \theta_{vd}), c_j;\theta_s) + \log S(\widetilde{X}_j(\theta_{vd}), c_j;\theta_s)\right). \tag{10}$$

Finally, GANs are known to be numerically unstable during training, with two notorious phenomena appearing in practice (Thanh-Tung, Tran, and Venkatesh, 2019). The first type, *mode collapse*, results in the generative part of the model only producing one type of image, no matter the vector of latent feature used. The second type, *gradient explosion*, results in the model parameter increasingly diverging and the network becoming unable to reconstruct any image correctly. Following best practices, we solve this issue by adding two loss components $Loss_{RG}$ and $Loss_{RD}$, known as orthogonal weight regularization (Brock et al., 2016), and discussed in Appendix B.1. Our final loss functions for the CVAE generator and the discriminator are:

$$\begin{aligned} Loss_{CVAE}(\theta_{ve}, \theta_{vd}) = {} & w_{V1} \cdot Loss_{V1}(\theta_{ve}, \theta_{vd}) + w_{V2} \cdot Loss_{V2}(\theta_{ve}) \\ & + w_G \cdot Loss_G(\theta_{ve}, \theta_{vd}) + Loss_{RG}(\theta_{ve}, \theta_{vd}), \end{aligned} \tag{11}$$

$$Loss_D(\theta_s) = Loss_S(\theta_s) + Loss_{RD}(\theta_s), \tag{12}$$

where the tuning weights $(w_{V1}, w_{V2}, w_G)$ can be adjusted to the researcher's taste to balance fidelity, LSN, and realism. The CVAE generator and the discriminator are trained alternatively with $\mathcal{D}_{train}$, using a stochastic gradient descent algorithm, such as ADAM (Kingma and Ba, 2014).

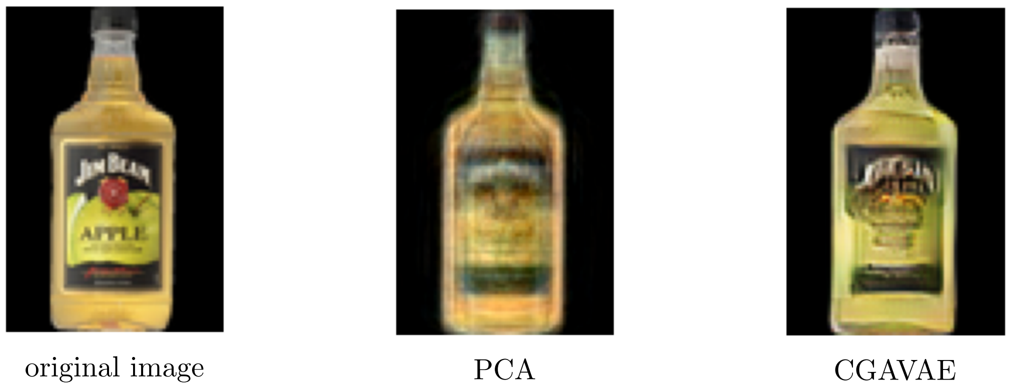

Figure 5: Out-of-sample reconstruction performance comparison between PCA and CGAVAE.

The CGAVAE achieves comfortable realism, sharpness and LSN in our bottle pictures application, and can be expected to generalize well to other applications given the general performance of CVAEs and GANs on multiples datasets documented in the literature (Brock, Donahue, and Simonyan, 2018; Karras, Laine, and Aila, 2019). The out-of-sample products are encoded and reconstructed correctly, with the main features preserved (Figure 5). Linear interpolation in the latent space produce pictures looking like plausible bottles (Figure 6), with the bottle characteristics progressively changing: neck length, body width, label shape, etc. The expected latent features for each bottles can then be used in a marketing model as discussed in Section 3.2. The fitted model can then be used to optimize the product design and presentation as shown with an application in Section 6.

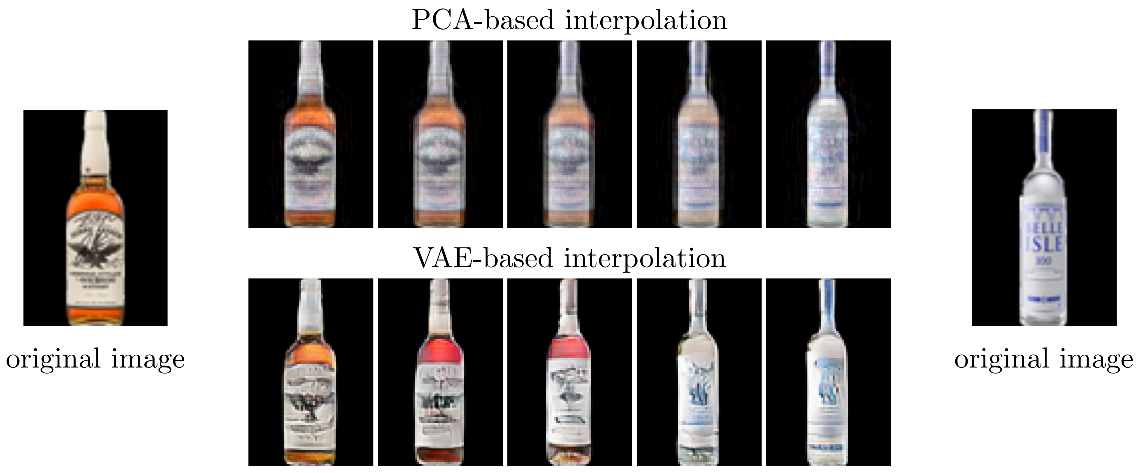

Figure 6: Comparison of PCA-based interpolation and CVAE-based interpolation from a bottle of *Jesse James America's Outlaw Bourbon* (left) to a bottle of *Belle Isle Premium Moonshine* (right).

## 3.2 Bayesian Deep Learning Model

We now have a compressed representation $\hat{z}_j = E(z_j|X_j, c_j)$, as well as product category $c_j$, and attributes $a_j$, for each product. We want to use this information to predict the variable of interest $y_j$ (the log-sales volumes for each SKU in our application). For notation convenience, $\mathcal{D} = \{\mathcal{D}_{train}, \mathcal{D}_{val}, \mathcal{D}_{test}\}$ now includes $\hat{z}_j$.

In this section, we first describe our method to perform this prediction task effectively with a limited sample size, leveraging the benefits of the Bayesian framework, and introducing a new type of Hamiltonian Monte-Carlo algorithm (Section 3.2.1). We then define a new criterion, the *overfitting score*, to measure the degree of overfitting associated with a model and estimation method (Section 3.2.2). Finally, we discuss how the fitted model identifies the low-dimensional projection of the latent feature space that is the most informative for the prediction task, and how this projection can be used for a meaningful interpretation of the results (Section 3.2.3).

### 3.2.1 Model Specification

We are currently facing two main issues: (1) we do not know the link between $\hat{z}_j$ and $y_j$ (likely non-linear with several variables interacting), and (2) the latent features vector $\hat{z}_j$ is still high dimensional relative to our dataset size. We thus need a model that allows for flexible functional forms between our explanatory and dependent variables, while mitigating the risks of overfitting.

To allow for flexible functional forms between our explanatory and dependent variable, we use a neural network $\text{NN}(z_j, c_j, a_j; w)$ parameter weights $w$ such that:

$$y_j \sim \mathcal{N}(NN(z_j, c_j, a_j; w), s^2), \tag{13}$$

where $s^2$ is a parameter controlling the variance of the error term. To mitigate overfitting issues, we set the model in a Bayesian framework, defining a prior distribution equivalent to a RIDGE regularization (with tuning parameter $\lambda$) for the network's weights . The prior distribution, the likelihood, and the resulting posterior distribution are discussed in Appendix B.2.

Traditionally, this model would be fitted to the training set $\mathcal{D}_{train}$, by using a stochastic gradient descent algorithm to obtain a *maximum a posteriori* (MAP) parameter estimate. With that approach, the regularization parameter $\lambda$ is set to maximize the predictive performance in the validation set $\mathcal{D}_{val}$. This predictive performance can for instance be measured with the coefficient of determination $R^2$. Finally, the overall performance of the method is measured in the set $\mathcal{D}_{test}$ that was neither used for training, nor parameter tuning. Most objective functions for deep learning models are highly irregular, with many local optima (Li et al., 2018). As a result, by selecting only the best fitting parameter value, an optimization-based approach might ignore several other highly likely values that would be more appropriate out-of-sample. Moreover, optimization is usually done through gradient descent methods and have a high risk in reaching only a local optimum. These issues get magnified in small samples, creating a high risk of overfitting, unless strong regularization is used, which impairs performance through heavily biased estimates (Marquardt, 1980).

To avoid the overfitting issues, and account for a larger panel of likely parameter values, we propose instead to simulate the full posterior distribution through *Markov Chain Monte-Carlo* (MCMC) techniques. With this approach, we do not rely on a point estimate maximizing in-sample fit, leading to potential overfitting, and instead obtain a sample of points that fit the training sample reasonably. This allows us to quantify uncertainty, both on the model weights estimates, and on the model output (MacKay, 1992). We propose to use an annealed version of the posterior distribution: $\frac{1}{\alpha}\log P(w, s^2|y, z, c, a)$, by introducing the annealing tuning parameter $\alpha$. When $\alpha$ tends to 0, the sampled points converge toward the MAP estimate. On the other hand, values greater than 1 smooth the posterior distribution, approaching a uniform distribution in the extreme case where $\alpha$ tends toward infinity. This parameter can be tuned in conjunction with the regularization parameter $\lambda$ using the validation sample.

The MCMC simulation of the posterior distribution presents several computational challenges. The number of parameters to estimate can be potentially high, making it a high-dimensional simulation problem. For this purpose, Hamiltonian Monte-Carlo (HMC) methods have been shown to be particularly appropriate for Bayesian Neural Network models (Neal, 1996). Unfortunately, HMC is computationally very expensive, requiring to compute the gradient of the log posterior distribution many times for each MCMC step. We propose instead a novel Stochastic Hamiltonian Monte-Carlo (SHMC) algorithm, where the gradient of the log posterior is approximated by sub-sampling observations. We discuss this algorithm in Appendix A.

Our method allows for the effective fitting of complex deep learning models on datasets of limited size. We compare the predictive performance with several other models and methods in the application presented in Section 5. To allow for a meaningful comparison, we first introduce a way to measure overfitting in Section 3.2.2, and discuss how to translate the results to interpretable characteristics in Section 3.2.3.

#### 3.2.2 Measuring overfitting

Traditionally, overfitting has been measured via the coefficient of determination $R^2$, measured out-of-sample:

$$R^2 = 1 - \frac{\sum(y_j - NN(z_j, c_j, a_j; w))^2}{\sum(y_j - \bar{y})^2}. \tag{14}$$

The presence of overfitting would be detected when the in-sample $R^2$ is significantly higher than its out-of-sample counterpart. In the Bayesian literature however, there is no clear way to detect overfitting. An alternative "Bayesian R-squared" measure has been proposed by Gelman et al. (2019), which intends to mitigate the influence of overfitting but does not explicitly measure it. We propose here instead to complement the traditional $R^2$ measure with an *overfitting score* (OFS) based on the ratio of the predicted residual variance $s^2$ over the observed counterpart:

$$OFS(w, s) = 1 - \frac{s^2}{\widehat{Var}(y_i - NN(z_j, c_j, a_j; w))}. \tag{15}$$

An overconfident model fit would predict a residual variance that is lower than its observed counterpart, and $OFS$ would tend toward 1. A model where the predicted residual variance is accurate would have an $OFS$ close to 0. Finally, models where the predicted residual variance is lower than the observed one would have a negative $OFS$. When the dataset is divided in training, validation and test samples, the corresponding scores are denoted $OFS_{train}$, $OFS_{val}$ and $OFS_{test}$. Since the $OFS$ depends on parameter values, for the Bayesian case we report the Monte-Carlo approximation of the expected $OFS$ instead:

$$EOFS = \int OFS(w, s)\; p(w, s|\mathcal{D}_{train})\; dw\; ds. \tag{16}$$

### 3.2.3 Interpreting the Results

Besides benefiting from unequaled performance in prediction (see Section 5.1), our approach also allows for identification of visual characteristics that are informative for our prediction task. This interpretation might be of high value to practitioners, who might be more interested in identifying relevant features than in predicting an outcome.

The space of latent features $z_j$ being of relatively high dimensionality, interpretation might still be difficult, and the latent features not necessarily relevant for our study. We can however use the fitted neural network parameters to only retain a handful of meaningful features. Indeed, the weights $W_0$ of the input layer of the model define a projection of the latent feature space to a lower-dimensional space through the vector-matrix multiplication $z_jW_0$. Moreover, this projection is selected to optimally predict the variable of interest. This ensures that the projected representation is highly informative for the task at hand. We can then easily visualize the location of our products in this projection space, through one of many graphics, to understand how the visual characteristics vary across that space.

If the projection space is still too high-dimensional for a full visual representation, we can instead identify clusters of products in our newly defined space. We can then visualize a small set of products from each cluster to understand the defining cluster characteristics (e.g., specific color or shape variations). This ultimately results in new ways to meaningfully characterize the products, and can drive the creation of new theories. As another benefit, the algorithm allows for automatic large-scale labeling based on these characteristics. We provide an illustrated example of an interpretation of the results in Section 5.2.

## 4 Simulation

In this section, we use simulation to show to what extent our Bayesian approach outperforms the traditional MAP approach in terms of out-of-sample predictive accuracy. To simulate a dataset, we first create a simple fully connected neural network $NN(z_j, a_j)$. The neural network architecture is a combination of linear operations followed by Softplus rectifications, as described in Appendix B.3. This architecture is identical to the neural network used in our application in Section 5. Since the Softplus function heavily shrinks values below zero, we need to ensure that the output of each layer does not get too negative, to avoid a vanishing gradient and make sure that the variations of the explanatory variables propagate well through the network. To this end, we draw the bias parameters from a $\mathcal{N}(2, 2^2)$ distribution, and the weight parameters are randomly generated to be orthogonal[1], preventing overly large variations at the output (see Saxe, McClelland, and Ganguli, 2013).

We then generate random explanatory variables by drawing $z_j$ and $a_j$ from standard multivariate Gaussian distributions. The dependent variable $y_j$ is obtained by feeding $(z_j, a_j)$ to the neural network, and artificially injecting a $\mathcal{N}(0, 0.5)$ distributed noise in the last layer before the rectifying Softplus operation.

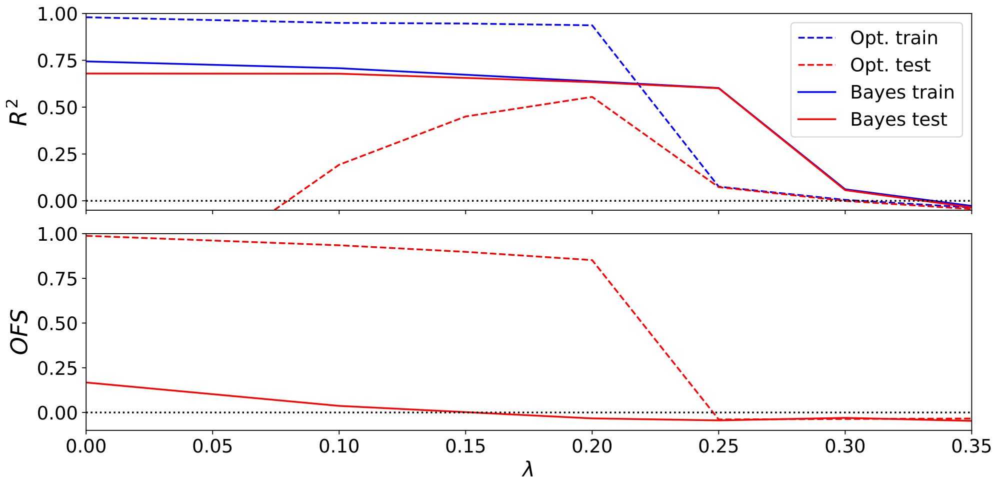

Figure 7: Comparison of the Bayesian (solid lines) and optimization-based (dashed lines) simulation results in the training and test sets, for various regularization parameter values. An increasing $\lambda$ indicates a stronger shrinkage of the model weights towards 0.

We now compare the predictive performances of the Bayesian and optimization-based (MAP) approaches, represented graphically in Figure 7. Fist, with MAP, we can see that the in-sample $R^2$ (blue dashed line) remains close to 1 for regularization parameter $\lambda < 0.20$, before suddenly dropping, suggesting that a different mode of the objective function has become the new global optimum. The MAP's out-of-sample $R^2$ starts at negative values and progressively rises to reach

[1]Their product is a multiple of the identity matrix.

0.554 at $\lambda = 0.20$, before dropping similarly to its in-sample counterpart. The out-of-sample overfitting score remains very high for $\lambda = 0.20$, indicating that while the out-of-sample predictive performance might reach acceptable levels, the model remains overconfident regarding the quality of its predictions. By contrast, the Bayesian approach has a slightly lower in-sample $R^2$ but keeps a very high out-of-sample $R^2$ for $\lambda < 0.25$. Moreover, the overfitting score remains very low in the same regularization range ($OFS < 0.25$), compared to the MAP ($OFS > 0.80$). This suggests that our Bayesian approach vastly outperforms the optimization-based MAP for small samples.

## 5 Empirical Application

In this section, we apply our approach to a dataset of alcoholic spirits, using the product's characteristics and corresponding bottle picture to predict monthly sales (in USD). Our data contain all of the retail sales of spirits in the state of Pennsylvania for 2018, as provided by the Pennsylvania Liquor Control Board. We consider spirits in the following categories: *Brandy* (including Cognac), *Gin*, *Rum*, *Tequila*, *Vodka* and *Whiskey*. In Pennsylvania, a legal monopoly is established for alcoholic spirits retail. As such, we expect our dataset to include the near totality of retail sales in the state. Note that we only have sales data, no inventory data, and it is difficult for us to identify products that might have been offered at a given time but not sold. We limit our analysis to products sold for under \$100 USD as they represent the vast majority of sales, and products above that threshold likely fall in a different type of market. The distribution of prices for each category is represented in Figure 8.

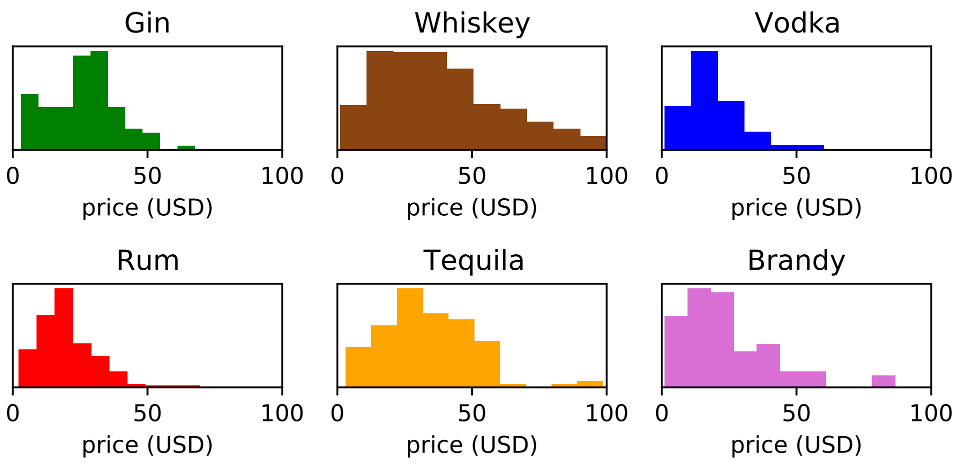

Figure 8: Price distribution for each category of alcoholic spirit.

Our aim is to predict the monthly sales volume for each product, with the dependent variable defined as follows $y_{jt} = \log(1 + sales_{jt})$. Note the addition of a time index to represent the variations across time. The product attributes are: the price (monthly average), the bottle capacity, the alcohol proof, the product category (dummy: brandy, gin, etc.), and the month of the year (dummy). Our dataset is split into a training set of 846 products, a validation set of 300 products, and a test set of 300 products. We first compare the predictive performance of our model with various competing models in Section 5.1. We then interpret and visualize the results in Section 5.2. We finally use the fitted model in an optimization exercise in Section 6.

## 5.1 Predictive Performance

In this section, we compare the out-of-sample performance of our neural network $NN(z_j, c_j, a_{jt})$ with 3 *Generalized Linear Models* (GLMs) of varying complexities, for both our Bayesian approach and an optimization-based (MAP) method. The neural network $NN(z_j, c_j, a_j; w)$ used for this specific application in described in detail in Appendix B.3. For each of the GLMs, we use a Softplus (SP) inverse link function:

$$SP(x) = ln(1 + e^x). \tag{17}$$

The 3 competing GLMs are:

1. A 2 stage model $GLM_{NN}(z_j, c_j, a_{jt})$, where the latent features are first fed through a simple neural network before being linearly combined with other attributes. In other words, the model is identical to the $NN(z_j, c_j, a_{jt})$ model but with the product's characteristics only introduced in the last layer:

$$GLM_{NN}(z_j, c_j, a_{jt}) = SP(\beta_0 + a_{jt}\beta_a + c_j\beta_c + NN(z_j, c_j)). \tag{18}$$

2. A simple GLM including the latent features $GLM_F(z_j, c_j, a_{jt})$:

$$GLM_F(z_j, c_j, a_{jt}) = \ SP(\beta_0 + a_{jt}\beta_a + c_j\beta_c + z_j\beta_z). \tag{19}$$

3. A simple GLM based on product attributes and category only $GLM_A(c_j, a_{jt})$. This represents the case where we do not have product pictures information and acts as a baseline comparison in our analysis:

$$GLM_A(z_j, c_j, a_{jt}) = SP(\beta_0 + a_{jt}\beta_a + c_j\beta_c). \tag{20}$$

We first compare the results of our Bayesian approach with a traditional optimization-based approach (MAP) for our neural network (see Figure 9). The best out-of-sample fit for the Bayesian method is achieved for $\lambda = 0$ (no regularization), with $R^2_{test} = 0.346$ and $OFS_{test} = 0.166$. The best optimization-based results are achieved with a regularization of $\lambda = 1$, with $R^2_{test} = 0.308$ and an $OFS_{test} = 0.313$. We see not only that the Bayesian approach achieves superior out-of-sample prediction overall, but also does it with a largely lower overfitting score. Interestingly, the Bayesian and optimization approach achieve very similar out-of-sample $R^2$ and $OFS$ performance for higher regularization parameter values ($\lambda > 1.5$). The main differences are observed when almost no regularization is applied, with the optimization-based method having a very high in-sample fit ($R^2_{train} = 0.729$), and an out-of-sample fit that is worse than using the sample average as a predictor ($R^2_{test} < 0$), for $\lambda = 0$. The difference in performance between the Bayesian and MAP approaches are present, but not as pronounced as in the simulation. This can be explained by the presence of more noise (and hence lower predictive accuracy overall), or by the hypothetical true model having parameters values close to 0. This second factor would minimize the bias resulting from the regularization shrinking parameter values toward 0.

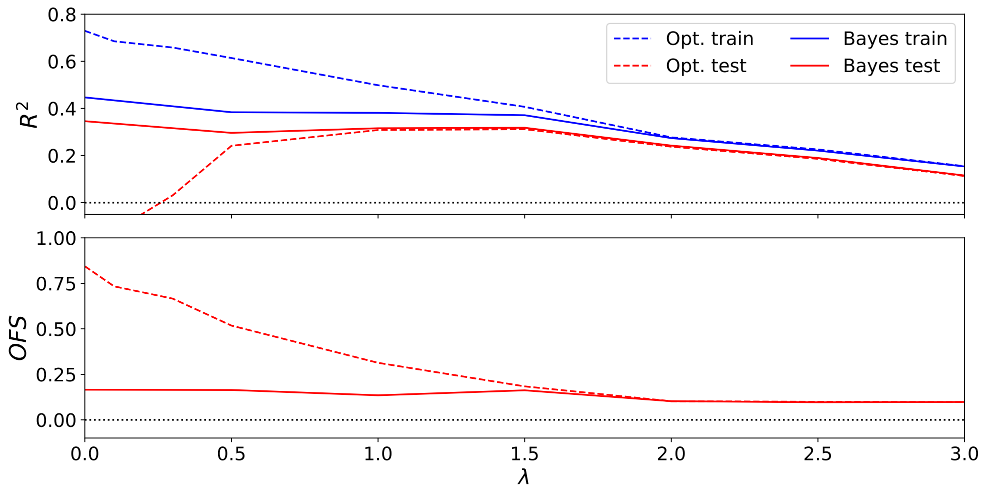

Figure 9: Comparison of the Bayesian (solid lines) and optimization-based (dashed lines) results in the training and test sets, for various regularization parameter values. An increasing $\lambda$ indicates a stronger shrinkage of the model weights towards 0.

We now compare the neural network results with the results of the other models represented in Table 1. The results are reported for optimal tuning parameters values, adjusted to maximize the validation set performance.

| Model | Best $R^2_{test}$ ($OFS_{test}$) | | Num. parameters |
|---|---|---|---|
| | Bayesian | Optim. | |
| $NN(z_j, c_j, a_{jt})$ | **0.346** (0.166) | 0.308 (0.313) | 740 |
| $GLM_{NN}(z_j, c_j, a_{jt})$ | 0.321 (0.199) | 0.296 (0.346) | 602 |
| $GLM_F(z_j, c_j, a_{jt})$ | 0.273 (0.293) | 0.274 (0.316) | 278 |
| $GLM_A(c_j, a_{jt})$ | 0.240 (0.103) | 0.240 (0.103) | 22 |

Table 1: Comparison of out-of-sample (test set) $R^2$ and $OFS$ for the Bayesian and optimization-based methods. The results are reported for the tuning parameter values that maximize the validation set's $R^2$.

## 5.2 Results Visualization and Interpretation

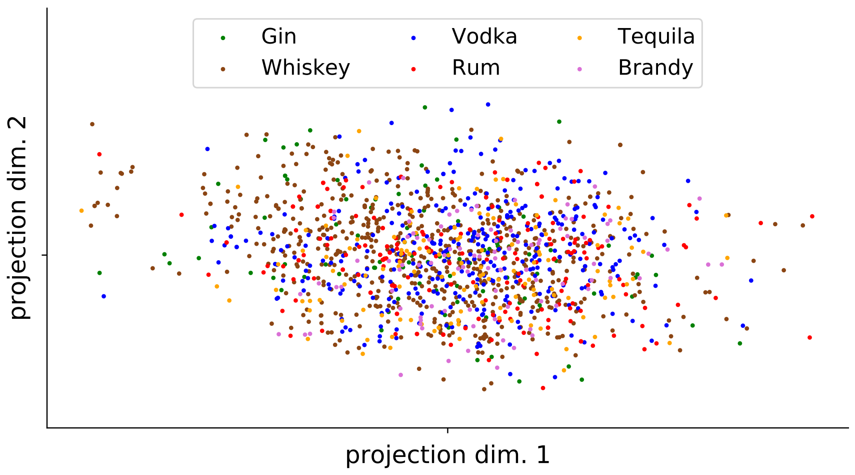

Figure 10: Standardized distribution ($s.d. = 1$ in every dimension) resulting from the projection of the first layer of our fitted neural network. The distribution appears to be close to a bivariate Gaussian distribution ($cor = -0.18$).

As discussed in Section 3.2.3, the results can be interpreted by visualizing the projection of the latent features done through the first layer of the neural network. This projection has the advantage of being selected by the neural network as being the most relevant for predicting the outcome variable. In our application, increasing the number of projection dimensions above 2 did not improve the out-of-sample predictive performance, and the results can thus be visualized through a single graph. We show the distribution of all of the products in the projection space in Figure 10. We can see that the distribution looks mostly Gaussian for every product category, with a few clusters of points that can be identified in the outer parts of the distribution.

To interpret this distribution, it is useful to see which picture corresponds to each point on the graph. We do this analysis separately for each category, as the meaning of each dimension is not necessarily homogeneous across them. In this section we discuss the case of the "brandy" products as an example (Figure 11). The visualizations for other product types are provided in Appendix C, and interpretation is left to the reader. We can see several clear separations on the graph. First, on a color scale, bottles in the bottom half are more colorful and lighter than in the upper area. We can see, in the upper right corner, a concentration of slender bottles, while the upper left has a concentration of very dark bottles (black or blue). In the left area, labels are mostly dark and of various shapes, while the right half sees a higher proportion of rectangular labels with lighter tones, such are white, yellow, or cream color. This representation can also be used to potentially identify design gaps in the market, and to explore some combination of features that have been mostly ignored by the competition. Notably, we can see two unpopulated areas respectively left and right of the center. Various new designs could be tried to see if they could fill these gaps. Designs could also be explicitly generated by the model while maximizing some criterion, which we discuss in the

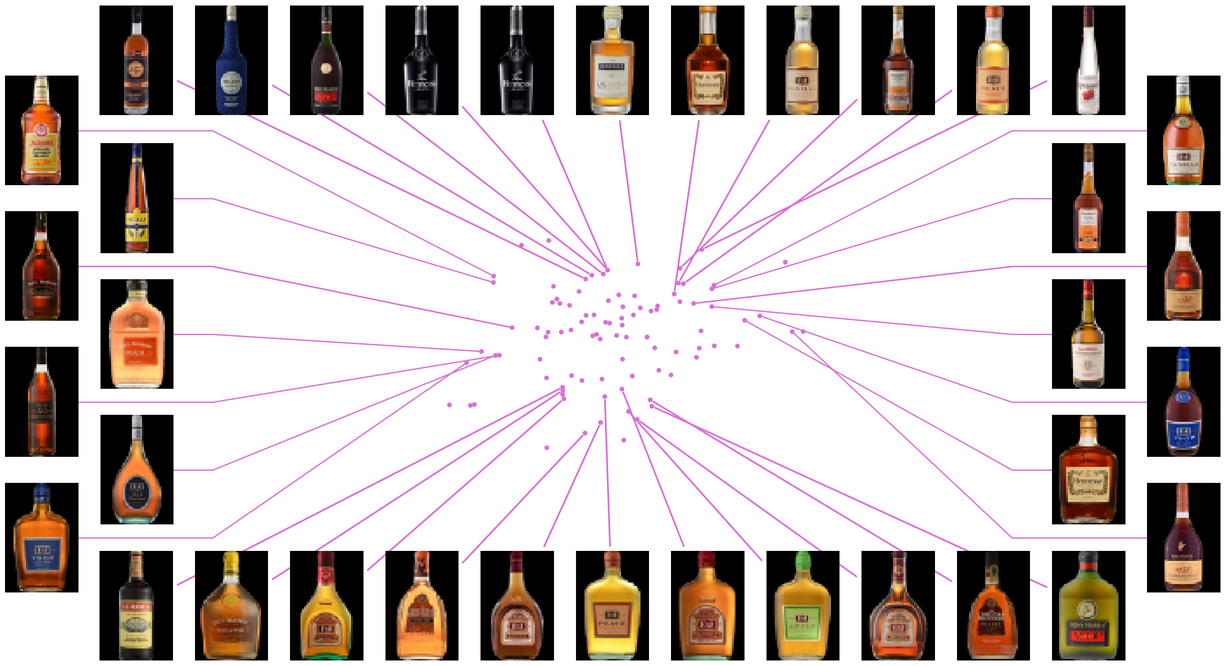

Figure 11: Visualization of the location of several product pictures in the projected latent feature space for the Brandy category.

following section.

## 6 Optimization for Product Design

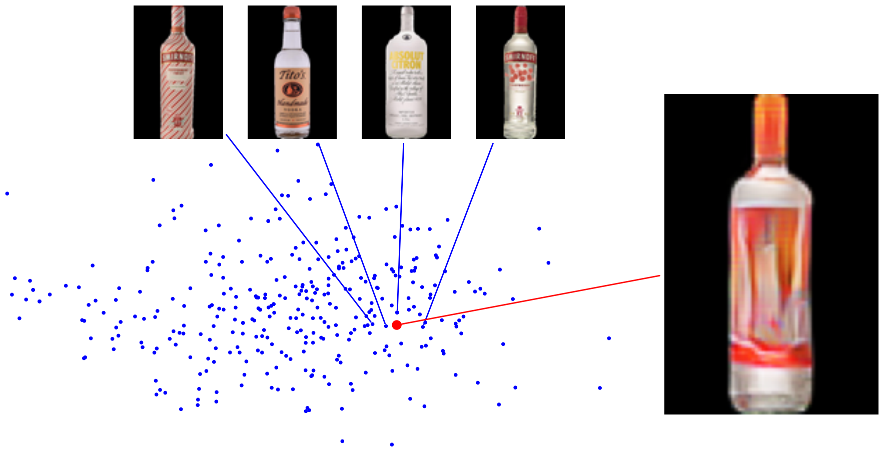

Figure 12: Representation of the generated optimal design, with corresponding localization in the projected features space, and its 4 nearest neighbors.

We propose here an optimization exercise consisting in identifying potential product designs from our fitted model. We want to mention again that, with observational data, causal links cannot be established and the fitted model might just suggest to copy the category's best seller. The exercise, as presented here, is mostly useful to provide ideas and concepts for designers that can be later assessed more thoroughly, for instance through surveys and focus groups.

Suppose that we are producing a new type of vodka and are looking to position our product. We want to create a bottle design and set a price that jointly maximize the expected revenue over the year:

$$annual\ revenue = \sum_{t=1}^{12} sales_t \cdot price_t. \tag{21}$$

We can use our fitted model to forecast the potential sales volumes given a set of product characteristics. First we set the bottle capacity to 750ml and the alcohol proof at 80, which are the industry's most common values. For simplicity we consider these values as given, and focus on the product design, represented by its latent features $z$, and the set of monthly prices $p = \{p_1, ..., p_{12}\}$. We denote the model $NN(z, p_t; w)$ for simplicity. Since there is some uncertainty associated to our model parameters, the sales forecast is also uncertain and we need to compute the expected revenue by integrating over the space of parameters, as is customary in a Bayesian framework. We thus define the loss function as:

$$Loss_{rev}(z, p) = -\log\left(\sum_{t=1}^{12} \int NN(z, p_t; w) \cdot p_t \cdot p(w|\mathcal{D}_{train}) \cdot dw\right). \tag{22}$$

However, defining this function is not sufficient, as the latent features affect the sales prediction only through their low-dimensional projection $u(z) = z'w_0$. As of now, we can only identify an optimal $u^*$ in the projection space to maximize the revenue, but this corresponds in turn to infinitely many possible latent features combinations. We thus need to set additional criteria to ensure a well defined problem. First, we want to maintain the projected $u(z)$ within the space of reasonable values. We thus add a regularization loss that penalizes deviation from the average projection $\bar{u}$ among the existing bottles in our market:

$$Loss_{proj}(z) = \|u(z) - \bar{u}\|_2^2. \tag{23}$$

Without this term, the optimal latent feature location can end up in a part of the space where no other product is located, resulting in a reconstructed image that does not even depict a recognizable product.

Second, we might want the latent features to be similar to the representation of other products in the category. To achieve this, we add a term that minimizes the squared norm between $z$ and its $l$-closest neighbors, denoted $(\tilde{z}_1, ..., \tilde{z}_l)$. The value of $l$ (set to 4 in this application) can be varied to taste to obtain different types of optimal products. This term is important as it ensures that our optimization problem has a finite number of solutions. The overall loss $Loss_{main}$ is obtained by simple summation of the previously defined losses:

$$Loss_{feat}(z) = \frac{1}{l}\sum_{j=1}^{l} \|z - \tilde{z}_j\|_2^2\,, \tag{24}$$

$$Loss_{main}(z,p) = Loss_{rev}(z,p) + Loss_{proj}(z) + Loss_{feat}(z), \tag{25}$$

$$(z^*, p^*) = \arg\min_{z,p} Loss_{main}(z,p). \tag{26}$$

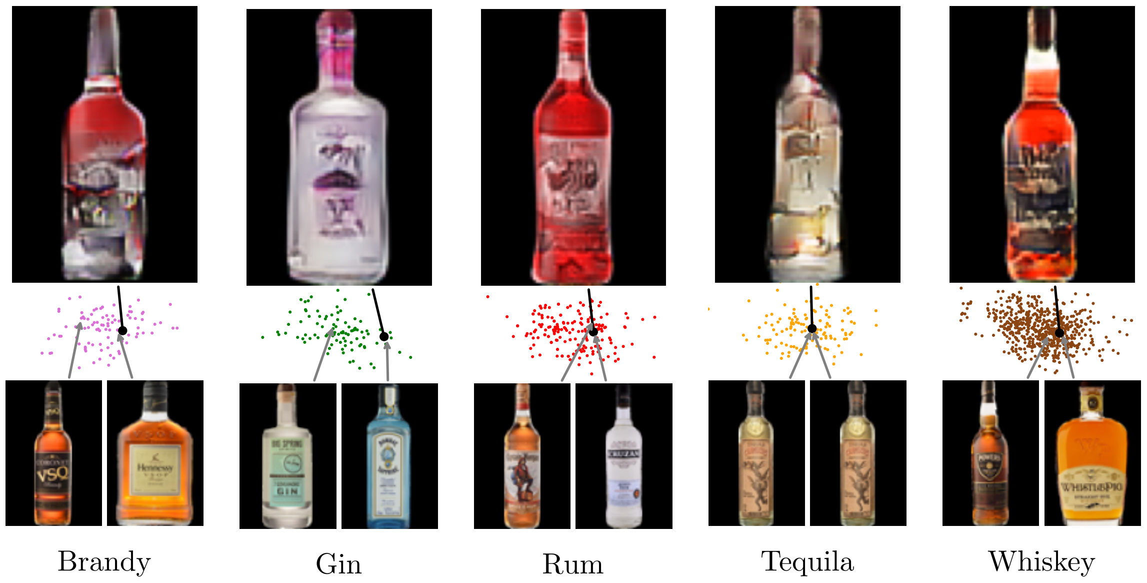

Figure 13: Various optimized designs for different categories of products, with the nearest neighbor in the feature space (bottom left) and projection space (bottom right).

We now discuss the results, obtained with a standard gradient descent algorithm. The optimal bottle design for the selected product, reconstructed from its latent features by the CGAVAE, is represented in Figure 12. In that figure, we also situate the design in the projected feature space, and provide the pictures of the four nearest neighbors for comparison. The optimal design is set in the lower-left part of the projection space where bottles have warm colors (yellow/red) and has a resulting design reminiscent of fire. The projection not being too far from the center of the distribution, the bottle shape is quite standard, neither thin not bulky, and with a straight body. The price is set to \$35.95, constant throughout the year [2]. Note that nearly all of the bottles in our dataset have constant prices, which might be due to the fact that the retail of spirits is a state monopoly in Pennsylvania. The algorithm might have failed to identify any price sensitivity given the absence of within-product variations in the dataset. The forecasted monthly sales volumes are mostly stable throughout the year, with a slight decline at the beginning of the summer, and an increase toward the end of the year (see Figure 14).

[2]Price variations were minimal and lost in the rounding to the nearest 5 cents.

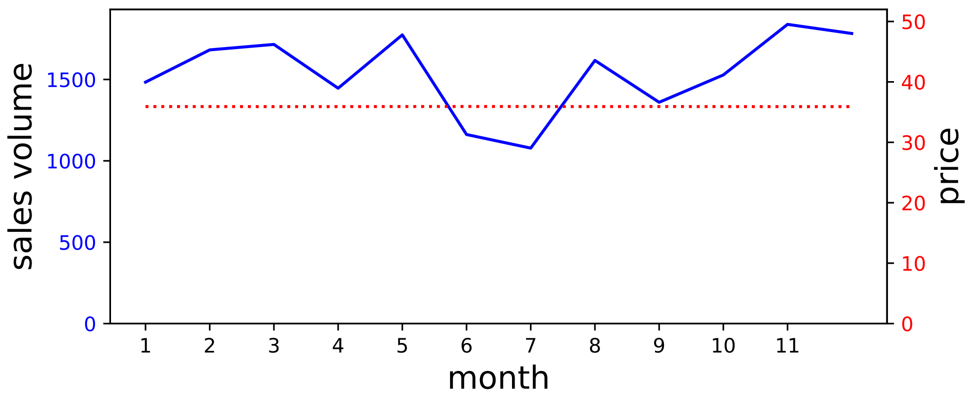

Figure 14: Sales volume forecast for the optimized product, with corresponding suggested sales price.

We applied the same approach to other product categories and present the results in Figure 13. We can see that the algorithm produced interesting designs for each type of product. We believe this demonstrates that our Bayesian pipeline allows marketing researchers and practitioners to leverage the power of deep learning tools even with limited dataset sizes. The pipeline can be adapted to many other problems, with some of them discussed in the next section.

## 7 Future Research and Potential Applications

Pictures and other types of high dimensional unstructured data can be informative for many other applications in marketing and we believe our pipeline could be applied directly for several of them. There are already a large range of domains where exceptional performance has been reached with auto-encoders and GANs. Some of these algorithms could be used as-is for the encoding stage of our pipeline (Stage 1). We can think, for instance, about the family of algorithms encoding pictures of rooms, which could be used for to study demand in the lodging, real estate, and architecture industries. Another family of algorithms have been very effective at encoding and generating faces. These could be used in advertising to study desirable facial characteristics for digital ads, or to study responses to perceived emotions, as has been done is other fields of research (Guan, Ryali, and Angela, 2018). Many products and classes of pictures that benefit from existing published encoding algorithms are prime candidates for our pipeline, since the first stage has essentially already been completed, and the network has likely already been trained on large datasets to achieve unrivaled performance.

While finding a suitable auto-encoder architecture for a particular type of pictures might be easy, given the growing literature in deep learning, using the resulting encoded features $z_j$ in a quantitative model might not be straight forward. Our pipeline proposes a solution to the overfitting issue, but it does not specify which type of model to use for a particular application.

This decision relies mostly on the researcher's expertise, but there are a few specific approaches that can be applied. The main issue is that the latent feature representation cannot be used directly, and generally needs to be transformed for the information to be exploited to its full potential. One simple solution is to feed all the information through a neural network, and let the algorithm recover the correct functional form. However, this approach does not benefit from all the already established research that identified particular functional forms and mechanisms as relevant for some applications. Another solution is to feed the latent representation to a neural network $NN(z_j)$ first, and to then use the output of this network in a model of interest $f(NN(z_j), x_j)$, where $x_j$ represents other relevant variables. This allows to use a theory-informed structure on the function $f(\cdot)$, while still benefiting from the information provided by the picture. The $GLM_{NN}$ model used in Section 5.1 follows this approach, feeding the neural network output to a GLM, and achieves a performance close to the more flexible neural-network form.

Another promising application, would be to use the picture in an instrumental variable estimation approach. Indeed, in several contexts, pictures might satisfy the exogeneity condition required to be used as an instrument. In that case, the endogenous variable $v_j$ is first fitted to the latent feature representation through the neural network, and then replaced by it's fitted value $\hat{v}_j = NN(z_j)$ in another model to remove the endogeneity problem. This approach could significantly improve market analysis performance where endogeneity is generally an issue.

Overall, we believe that our method is very flexible and can be applied to a broad range of fields by simply modifying the second step to use a theory-informed functional form. Applications are not limited to quantitative research, as we can think of using this new source of data to inform behavioral models, or to generate new pictures on demand to help experiments (Hagen et al., 2020).

## 8 Conclusion

In this paper, we have presented a novel approach to allow for the use of deep learning models with datasets of limited sizes that are frequently encountered in marketing. Our pipeline uses state-of-the-art dimensions reduction methods with Bayesian deep learning models to leverage the information contained in unstructured high dimensional data such as product pictures. The method is particularly well suited for prediction, with an out-of-sample performance superior to competing methods, while quantifying uncertainty and mitigating overfitting issues. More interestingly, our approach also facilitates results interpretation by allowing researchers to identify features of the unstructured data that are relevant for the problem at hand. Finally, the method can be used to generate new products by creating potential designs that are optimized to achieve certain goals such as revenue, product ratings, or consumer satisfaction.

We have presented an application to alcoholic spirits sales in a market of under 2000 products, and demonstrated the small sample performance of our approach. We then interpreted the results by identifying clusters of products with similar characteristics in the space of relevant features.

We presented a design optimization exercise based on our results and generated a series of bottles predicted to have high revenue potential, given the products' characteristics. Finally, we discussed how this approach can be easily adapted to other applications, as it solves a problem encountered in many subfields of marketing, from product creation to structural demand analysis.

We believe that our method will allow for the expansion of deep learning models through many marketing applications, benefiting both researchers and practitioners. With the rapid development of ever improving deep learning models, and the progresses made in associated computational methods, we expect deep learning to become a pillar of marketing in the near future, no matter the size of the dataset.

## References

Baydin, A. G., B. A. Pearlmutter, A. A. Radul, and J. M. Siskind (2017). "Automatic differentiation in machine learning: a survey". In: *The Journal of Machine Learning Research* 18.1, pp. 5595–5637.

Bouchacourt, D., R. Tomioka, and S. Nowozin (2018). "Multi-level variational autoencoder: Learning disentangled representations from grouped observations". In: *Thirty-Second AAAI Conference on Artificial Intelligence*.

Brock, A., J. Donahue, and K. Simonyan (2018). "Large Scale GAN Training for High Fidelity Natural Image Synthesis". In: *arXiv:1809.11096*.

Brock, A., T. Lim, J. M. Ritchie, and N. Weston (2016). "Neural Photo Editing with Introspective Adversarial Networks". In: *arXiv:1609.07093*.

Burnap, A. and J. Hauser (2018). "Predicting" Design Gaps" in the Market: Deep Consumer Choice Models under Probabilistic Design Constraints". In: *arXiv:1812.11067*.

Burnap, A., J. R. Hauser, and A. Timoshenko (2019). "Design and evaluation of product aesthetics: a human-machine hybrid approach". In: *SSRN 3421771*.

Burnap, A., R. Gerth, R. Gonzalez, and P. Y. Papalambros (2017). "Identifying experts in the crowd for evaluation of engineering designs". In: *Journal of Engineering Design* 28.5, pp. 317–337.

Calder, A. J., A. M. Burton, P. Miller, A. W. Young, and S. Akamatsu (2001). "A principal component analysis of facial expressions". In: *Vision research* 41.9, pp. 1179–1208.

Coates, A., A. Ng, and H. Lee (2011). "An analysis of single-layer networks in unsupervised feature learning". In: *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 215–223.

Dahan, E. and V Srinivasan (2000). "The predictive power of internet-based product concept testing using visual depiction and animation". In: *Journal of Product Innovation Management* 17.2, pp. 99–109.

Dzyabura, D. and R. Peres (2019). "Visual Elicitation of Brand Perception". In: *SSRN 3496538*.

Gelman, A., B. Goodrich, J. Gabry, and A. Vehtari (2019). “R-squared for Bayesian Regression Models”. In: *The American Statistician* 73.3, pp. 307–309.

Goodfellow, I., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio (2014). “Generative adversarial nets”. In: *Proceedings of Advances in neural information processing systems*, pp. 2672–2680.

Guan, J., C. K. Ryali, and J. Y. Angela (2018). “Computational modeling of social face perception in humans: Leveraging the active appearance model”. In: *bioRxiv*.

Hagen, L., K. Uetake, N. Yang, B. Bollinger, A. J. Chaney, D. Dzyabura, J. Etkin, A. Goldfarb, L. Liu, K Sudhir, et al. (2020). “How can machine learning aid behavioral marketing research?” In: *Marketing Letters (forthcoming)*.

Han, S. H., K. J. Kim, M. H. Yun, S. W. Hong, and J. Kim (2004). “Identifying mobile phone design features critical to user satisfaction”. In: *Human Factors and Ergonomics in Manufacturing & Service Industries* 14.1, pp. 15–29.

Hou, X., L. Shen, K. Sun, and G. Qiu (2017). “Deep feature consistent variational autoencoder”. In: *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, pp. 1133–1141.

Jarvenpaa, S. L. (1990). “Graphic displays in decision making—the visual salience effect”. In: *Journal of Behavioral Decision Making* 3.4, pp. 247–262.

Karras, T., S. Laine, and T. Aila (2019). “A style-based generator architecture for generative adversarial networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4401–4410.

Kingma, D. P. and J. Ba (2014). “Adam: A method for stochastic optimization”. In: *arXiv:1412.6980*.

Kingma, D. P. and M. Welling (2013). “Auto-encoding variational bayes”. In: *arXiv:1312.6114*.

Kramer, M. A. (1991). “Nonlinear principal component analysis using autoassociative neural networks”. In: *AIChE journal* 37.2, pp. 233–243.

Labrecque, L. I. and G. R. Milne (2012). “Exciting red and competent blue: the importance of color in marketing”. In: *Journal of the Academy of Marketing Science* 40.5, pp. 711–727.

Lang, P. J., M. M. Bradley, B. N. Cuthbert, et al. (1997). “International affective picture system (IAPS): Technical manual and affective ratings”. In: *NIMH Center for the Study of Emotion and Attention* 1, pp. 39–58.

Lans, R. Van der, R. Pieters, and M. Wedel (2008). “Research Note—Competitive Brand Salience”. In: *Marketing Science* 27.5, pp. 922–931.

Larsen, A. B. L., S. K. Sønderby, H. Larochelle, and O. Winther (2016). “Autoencoding beyond pixels using a learned similarity metric”. In: *International conference on machine learning*, pp. 1558–1566.

Li, H., Z. Xu, G. Taylor, C. Studer, and T. Goldstein (2018). “Visualizing the loss landscape of neural nets”. In: *Advances in Neural Information Processing Systems*, pp. 6389–6399.

Li, Z., T. Zhang, S. Cheng, J. Zhu, and J. Li (2019). “Stochastic gradient hamiltonian monte carlo with variance reduction for bayesian inference”. In: *Machine Learning* 108.8-9, pp. 1701–1727.

MacKay, D. J. C. (1992). “A Practical Bayesian Framework for Backpropagation Networks”. In: *Neural Computation* 4.3, pp. 448–472.

Mao, X., Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley (2017). “Least squares generative adversarial networks”. In: *Proceedings of the IEEE international conference on computer vision*, pp. 2794–2802.

Marquardt, D. W. (1980). “A critique of some ridge regression methods: Comment”. In: *Journal of the American Statistical Association* 75.369, pp. 87–91.

Neal, R. M. (1996). *Bayesian Learning for Neural Networks.* Vol. 118. Lecture Notes in Statistics. Springer New York.

— (2012). “MCMC using Hamiltonian dynamics”. In: *arXiv:1206.1901.*

Netzer, Y., T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng (2011). “Reading digits in natural images with unsupervised feature learning”. In: *Working Paper.*

Pan, S. J. and Q. Yang (2009). “A survey on transfer learning”. In: *IEEE Transactions on knowledge and data engineering* 22.10, pp. 1345–1359.

Radford, A., L. Metz, and S. Chintala (2015). “Unsupervised representation learning with deep convolutional generative adversarial networks”. In: *arXiv:1511.06434.*

Ren, Y., A. Burnap, and P. Papalambros (2013). “Quantification of perceptual design attributes using a crowd”. In: *Proceedings of the 19th International Conference on Engineering Design (ICED13)*, pp. 139–148.

Rettie, R. and C. Brewer (2000). “The verbal and visual components of package design”. In: *Journal of product & brand management* 9.1, pp. 56–70.

Rezende, D. J., S. Mohamed, and D. Wierstra (2014). “Stochastic backpropagation and approximate inference in deep generative models”. In: *arXiv:1401.4082.*

Saxe, A. M., J. L. McClelland, and S. Ganguli (2013). “Exact solutions to the nonlinear dynamics of learning in deep linear neural networks”. In: *arXiv:1312.6120.*

Silayoi, P. and M. Speece (2007). “The importance of packaging attributes: a conjoint analysis approach”. In: *European journal of marketing* 41.11/12, pp. 1495–1517.

Singh, S. (2006). “Impact of color on marketing”. In: *Management decision.*

Thanh-Tung, H., T. Tran, and S. Venkatesh (2019). “Improving generalization and stability of generative adversarial networks”. In: *arXiv:1902.03984.*

Underwood, R. L. and N. M. Klein (2002). “Packaging as brand communication: effects of product pictures on consumer responses to the package and brand”. In: *Journal of Marketing Theory and Practice* 10.4, pp. 58–68.

Underwood, R. L. and J. L. Ozanne (1998). “Is your package an effective communicator? A normative framework for increasing the communicative competence of packaging”. In: *Journal of Marketing Communications* 4.4, pp. 207–220.

Zhang, S., D. D. Lee, P. V. Singh, and K. Srinivasan (2020). “How much is an image worth? Airbnb property demand estimation leveraging large scale image analytics”. In: *Working Paper.*

## A Stochastic Hamiltonian Monte-Carlo Algorithm

**Algorithm 1:** Stochastic Hamiltonian Monte-Carlo with Gradient Correction

```
draw initial position: $\theta_0 = (w_0, s_0^2)$;
randomly partition $\mathcal{D}_{train}$ in $B$ batches $\mathcal{B}_1, ..., \mathcal{B}_B$ with cardinality $n_1, ..., n_B$;
compute initial log likelihood gradient: $g_0 = -\sum_{b=1}^{B} \nabla \log \mathcal{L}(\mathcal{B}_b; \theta_0)$ ;
compute potential energy: $U_0 = \frac{1}{\alpha}(-\log \mathcal{L}(\mathcal{D}_{train}; \theta_0) + 0.5\lambda \left\|w_0\right\|_2^2)$;
repeat
    draw initial momuntum (same dimensionality as $\theta_0$): $m_0 \sim \mathcal{N}(0, I)$;
    compute kinetic energy : $M_0 = 0.5 \left\|m_0\right\|_2^2$;
    draw leapfrog size: $u \sim \mathcal{U}(10^{-5}, 10^{-4})$;
    momentum half step: $m_1 = m_0 - 0.5 \cdot u \cdot \frac{1}{\alpha}(g + N_{train} \cdot \lambda \cdot w)$;
    for $b = 1, ..., B$ do
        parameter step, set: $\theta_b = \theta_{b-1} + u \cdot m$;
        approximate gradient, set:
            $$\tilde{\nabla} U_b = \frac{1}{\alpha} \left( g_{b-1} + \frac{N_{train}}{n_b} \left( \nabla \log \mathcal{L}(\mathcal{B}_b; \theta_b) - \nabla log \mathcal{L}(\mathcal{B}_b; \theta_0) \right) + N_{train} \cdot \lambda \cdot w \right);$$
        if $b < B$ then
            momentum step, set: $m_{b+1} = m_b - u \cdot \tilde{\nabla} U_b$;
        end
        set: $g_b = g_{b-1} + \nabla \log \mathcal{L}(\mathcal{B}_b; \theta_b) - \nabla \log \mathcal{L}(\mathcal{B}_b; \theta_0)$;
    end
    momentum half step: $m_B = m_{B-1} - 0.5 \cdot u \cdot \tilde{\nabla} U_B$;
    compute potential energy: $U_B = \frac{1}{\alpha}(-\log \mathcal{L}(\mathcal{D}_{train}; \theta_B) + 0.5\lambda \left\|w_B\right\|_2^2)$;
    compute kinetic energy: $M_B = 0.5 \left\|m_B\right\|_2^2$;
    compute accept-reject probability: $p_{acc} = \min\{1, exp(U_0 - U_B + M_0 - M_B)\}$;
    do with probability $p_{acc}$:
        accept MCMC step: $\theta_0 = \theta_B$, $U_0 = U_B$;
        randomly partition $\mathcal{D}_{train}$ in $B$ batches $\mathcal{B}_1, ..., \mathcal{B}_B$ with cardinality $n_1, ..., n_B$;
        compute initial log likelihood gradient: $g_0 = -\sum_{b=1}^{B} \nabla \log \mathcal{L}(\mathcal{B}_b; \theta_0)$ ;
    end
until chain reaches stationary distrubution;
```

To simulate our posterior distribution, we propose a novel Hamiltonian Monte-Carlo approach, where the gradients necessary to compute the leapfrogs iterations are approximated by subsampling of the observations in our dataset (Algorithm 1). To minimize the subsampling noise, we add a gradient correction term in our approximation. This approach is similar to another algorithm proposed by Li et al. (2019), that also includes a gradient correction mechanism. However, our method possesses significant differences to improve performance: most notably, at the beginning of each MCMC step, a random initial momentum $m$ is drawn, and a Metropolis-Hastings accept-reject draw is done at the end of each step. We also implement a random leapfrog size, as suggested by Neal (2012), to accommodate for irregular objective function shapes (i.e., variations in smoothness and slope). The SHMC with gradient correction offers a close approximation of the target distribution while remaining computationally very efficient.

## B Neural Networks

### B.1 Orthogonal Weights Regularization

In order to prevent mode collapse (the generation of the same image for every latent feature vector), and gradient explosion, we add two regularizing component to our loss function. This type of regularization is called orthogonal weights regularization (Brock et al., 2016) and forces the weights of the generator $W_{kG}$ for each layer $k = 1, ..., K_G$, and the weights of the discriminator $W_{kD},\ k = 1, ..., K_D$ to satisfy an orthogonality condition. The resulting loss functions are:

$$Loss_{RG}(\theta_{ve}, \theta_{vd}) = \sum_{k=1}^{K_G} |W_{kG} W'_{kG} - I|, \tag{27}$$

$$Loss_{RD}(\theta_{ve}, \theta_{vd}) = \sum_{k=1}^{K_D} |W_{kD} W'_{kD} - I|. \tag{28}$$

$$\tag{29}$$

This regularization has been shown to improve stability and is used notably in the BigGAN (Brock, Donahue, and Simonyan, 2018), which regroups recent best practices from the GANs literature to achieve photo-realistic image generation for a large number of categories.

### B.2 Bayesian Model

We define here the prior distributions, the likelihood, and the resulting posterior distribution used for our Bayesian model. The neural networks weights $\{w_k\}_{k=1}^{K}$ follow independent Gaussian distributions with precision $N_{train} \cdot \lambda$, corresponding to an $L_2$ regularization of the weights (RIDGE). The variance parameter $s^2$ follows an improper inverse gamma distribution, corresponding to a uniform density over $[0, \infty)$]:

$$w_k \sim \mathcal{N}(0, N_{train}^{-1} \cdot \lambda^{-1}), \qquad \forall k, \tag{30}$$

$$s^2 \sim \mathcal{IG}(1, 0). \tag{31}$$

As a result, the log likelihood and log-posterior density can be written as:

$$\log \mathcal{L}(\mathcal{D}_{train}; w, s^2) = -N_{train} \log(s) - 0.5 \sum_{j \in \mathcal{J}_{train}} \left( \frac{y_j - NN(z_j, c_j, a_j; w)}{s} \right)^2 + C_1, \tag{32}$$

$$\log P(w, s^2 | \mathcal{D}_{train}) = \log \mathcal{L}(\mathcal{D}_{train}; w, s^2) - 0.5 \cdot N_{train} \cdot \lambda \sum_{k=1}^{K} w_k^2 + C_2, \tag{33}$$

where $C_1$ and $C_2$ are constants.

## B.3 Complete Neural Network

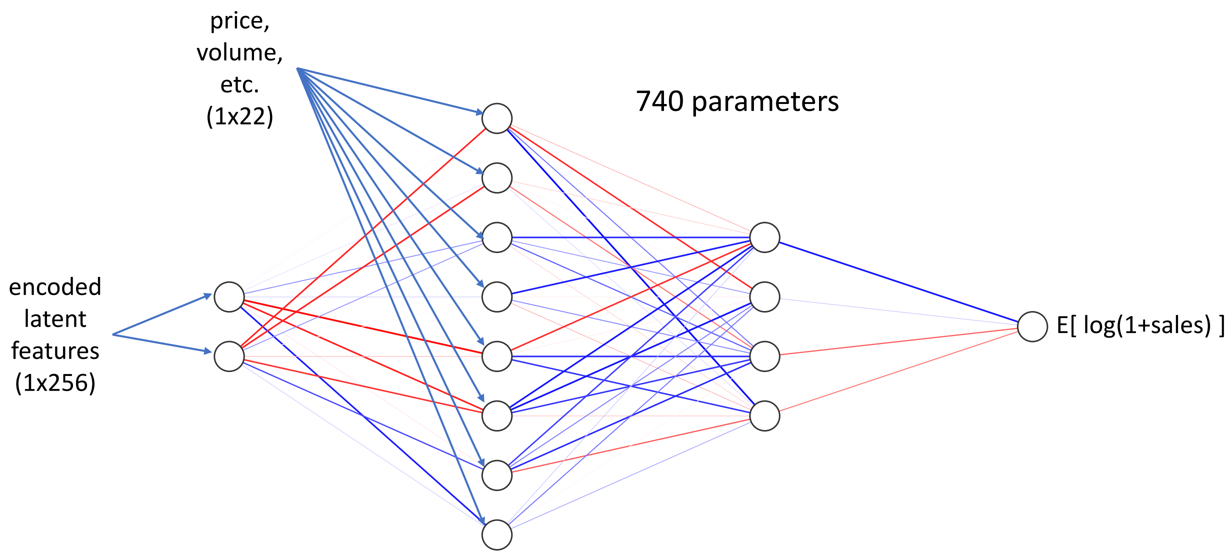

Figure 15: Neural Network architecture. Each node represent the combination of a linear transformation followed by a Softplus function.

Given the limited size of our dataset, we build a neural network that is neither too deep (number of layers), nor wide (number of nodes per layer), as this would increase the risk of overfitting that we want to prevent in the first place. This results in the neural network represented in Figure 15. For each node, the input values are summed with the layer weights and a bias parameter is added. A Softplus function is then applied to the linear operation's result, as defined below:

$$SP(x) = ln(1 + e^x). \tag{34}$$

The last layer's output is thus defined on $\mathbb{R}_+$, which corresponds to the range of $log(1 + y_i)$.

## C Latent Features Projections

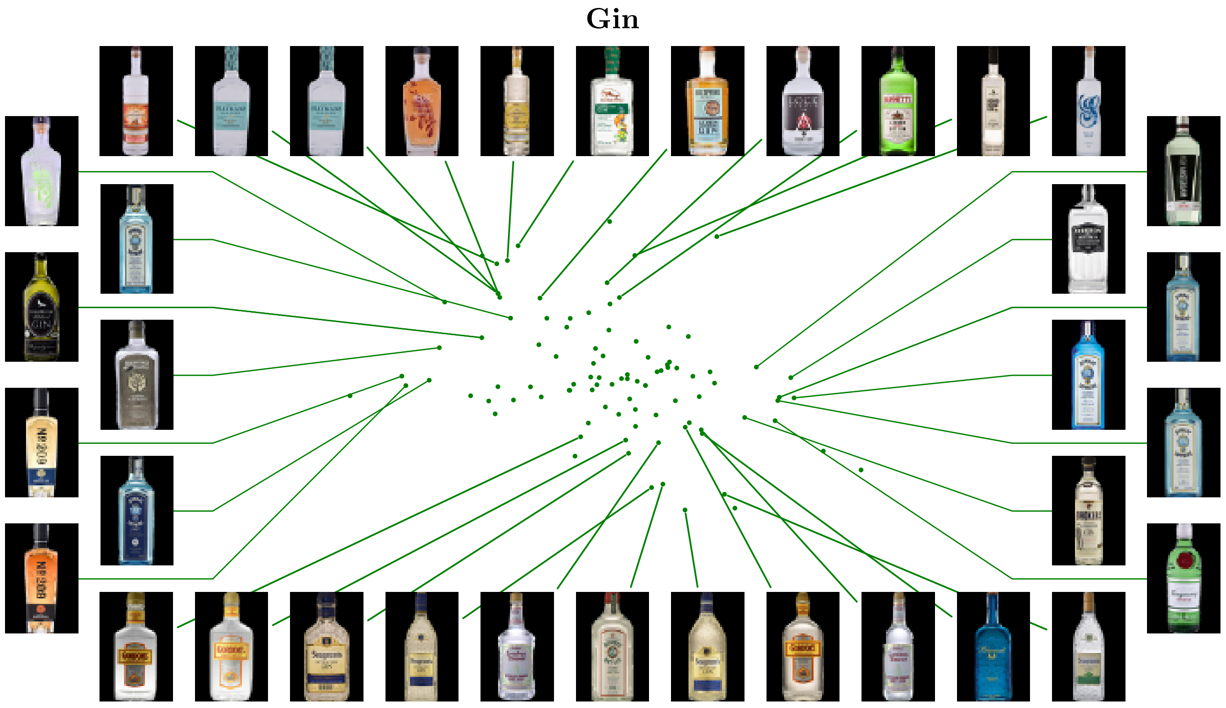

Figure 16: Projected latent features space for the Gin category.

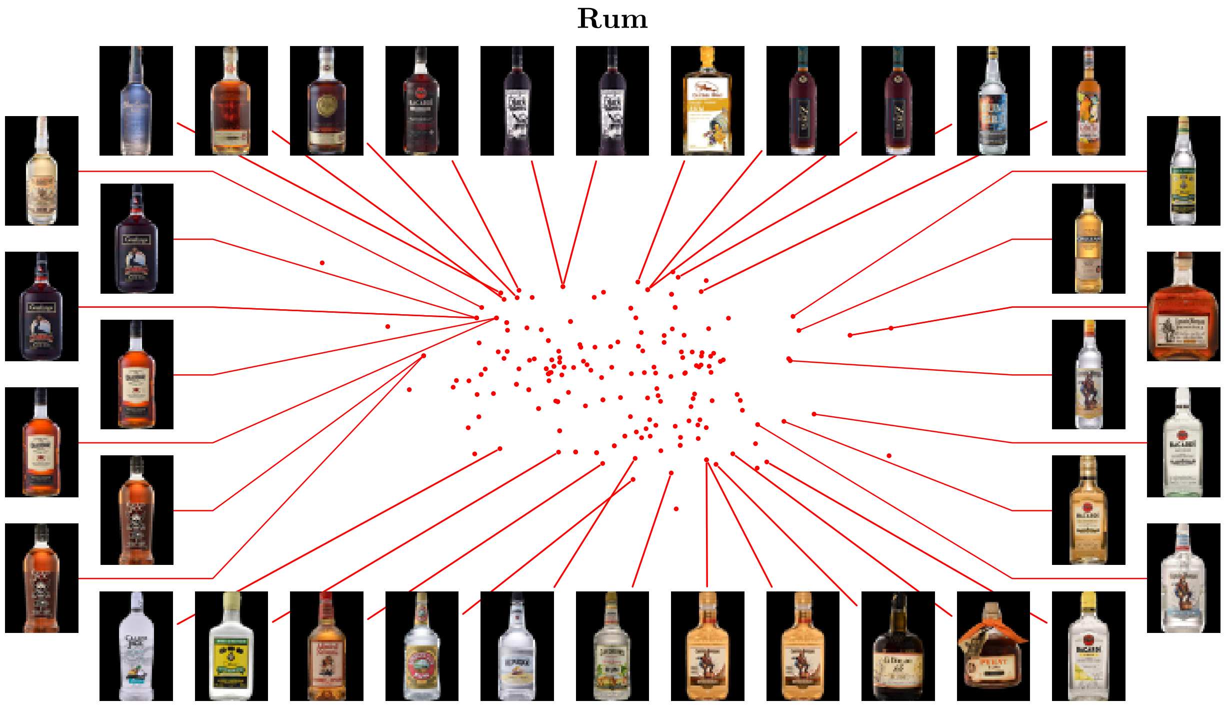

Figure 17: Projected latent features space for the Rum category.

**Tequila**

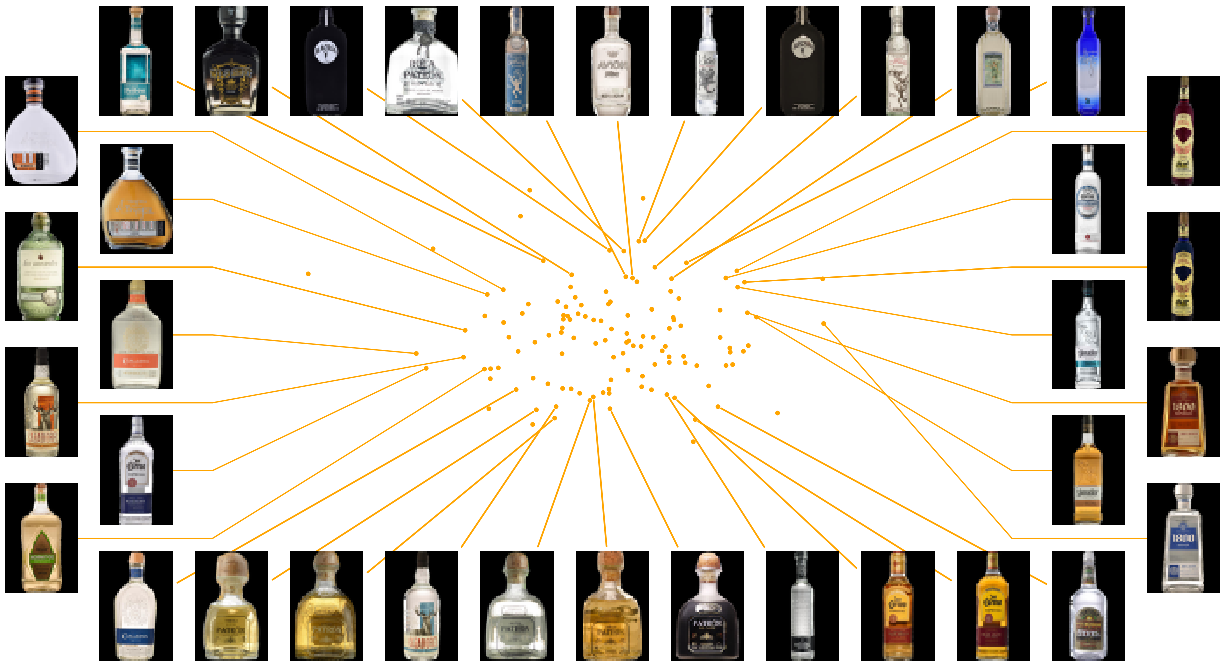

Figure 18: Projected latent features space for the Tequila category.

**Vodka**

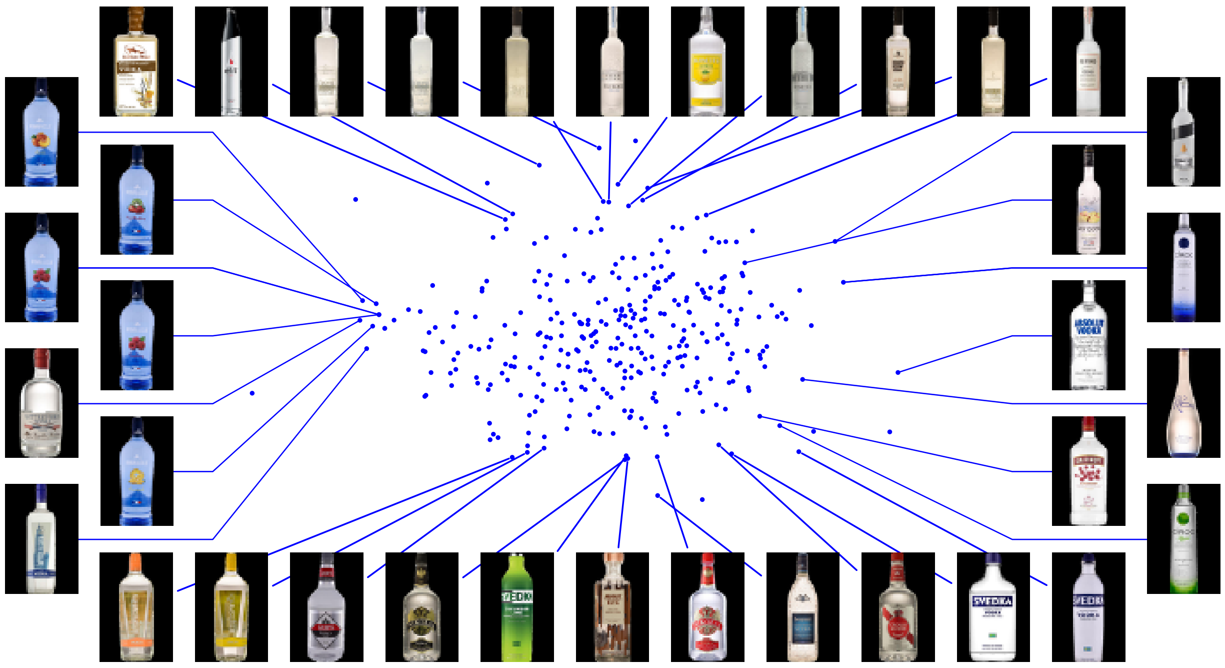

Figure 19: Projected latent features space for the Vodka category.

**Whiskey**

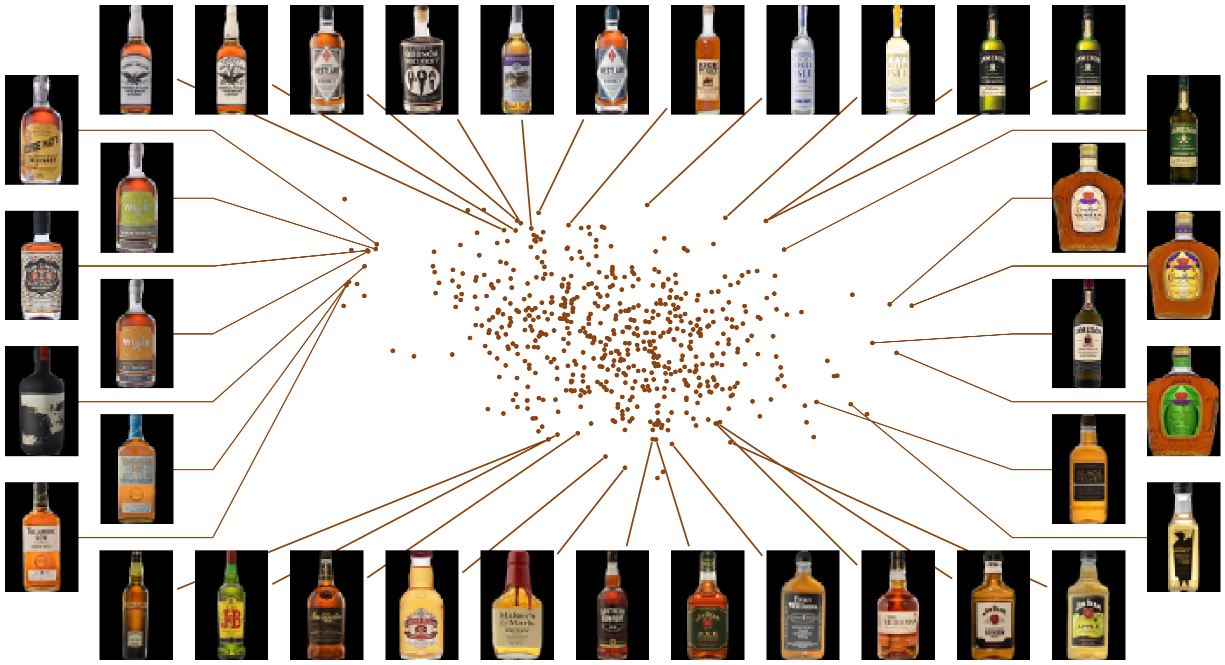

Figure 20: Projected latent features space for the Whiskey category.