Deep Learning Framework for Identifying Future Market Opportunities from Textual User Reviews

Jordan Kralev
Department of Systems and Control / Technical University of Sofia
Digital Computational Linguistics / Bulgarian Academy of Sciences
jkralev@ieee.org

Abstract

The paper develops an application of design gap theory for identification of future market segment growth and capitalization from a set of customer reviews for bought products from the market in a given past period. To build a consumer feature space, an encoded-decoder network with attention is trained over the textual reviews after they are pre-processed through tokenization and embedding layers. The encodings for product reviews are used to train a variational auto encoder network for representation of a product feature space. The sampling capabilities of this network are extended with a function to look for innovative designs with high consumer preferences, characterizing future opportunities in a given market segment. The framework is demonstrated for processing of Amazon reviews in consumer electronics segment.

Keywords: design gap theory, variational auto encoder, natural language processing

1 Introduction

In recent years, the large language models (LLM) such as GPT, PaLM, LLaMA demonstrate impressive performance on complex linguistic tasks in automatic text generation. There are a number of milestones marking this success in building such artificial neural network (ANN) models - encoder/decoder structures, probabilistic distribution encoding, embedding layers, long short-term memory cells, attention modeling - to name a few (Bordes et al., 2011; Lake et al., 2015; Paccanaro and Hinton, 2001). Since the growth of cloud computational capabilities and intensive research in the area, we see more claims that LLM are capable of outperforming or matching humans in number of language or image processing tasks. The so marked progress in ANN models allows development of more advanced expert systems to solve computational problems.

The focus of the present work is on developing a class of probabilistic predictive models for extracting information about consumer preferences in a particular online market, where the products in the market are divided in categories. As econometric knowledge explains, the consumer utility is the driving force for the current and future dynamics of the market. Hence, the model aim at identification of so-called ”design gaps” in the selected market segments, which allows for economic decisions for developing new products in the given segment. Information about consumer preferences is probabilistically encoded based on an unknown utility function over consumer/design pairs together with prior distributions over customers and designs. The analytical form of the probability densities and embedding representation allow representing the consumer preference information as a multilayer smooth neural network, which allows employment of gradient based methods for learning of hyperparameters from a training data set.

The design gap models allow the prediction of consumer preference for an ”unknown and not existing products”. The prediction cannot exactly tell what will be these future products, even though an effort can be made to extract such information from the underlying feature space. The model output is a bounded subset of the design space or feature space, which will be favored by the customers. Such bounded subset can be contrasted with the unbounded set of all possible future designs. In the present research, we follow the approach from (Burnap and Hauser, 2018b) for identification of design gap into a diverse design and consumer spaces, where they identify possible future products in the automotive sector. Based on the existing market data, a probabilistic consumer utility model is estimated to account for the likelihood of a certain product design to be selected by a given consumer. They also introduce auxiliary feature dimensions

for product and the consumer to improve the efficiency in estimation of the probability distribution. Similarly, (Ren and Scott, 2017) present adaptive approach for identification of optimal product designs from a finite set of possibilities, proposing Group Identification Splitting Algorithm (GISA) that picks queries that halves the so defined part-worth space but preserves entire groups.

The contribution of the present paper is that it develops a way to recover consumer and design vector representations from the raw textual reviews connecting a particular consumer to a particular product. Therefore, our assumption is that a review text is that all product or consumer features can be extracted from enough number of reviews. The first component of the model is to build a simple natural language transformer trained for estimating the underlying feature vector from each textual review. The model relies on a long short-term memory (LSTM) cells and attention mechanism such to enable the decoder subsystem to predict the next word token in a sequence to match a given input sequence. A specific feature in this model is that we don't aim a high generative accuracy characteristic for LLM mentioned above. Instead, we need to ensure that the model is simple enough to capture in the low-dimensional state vector the essential review information and the state vector can be used to recover the essence of keywords in the input sequence.

The language transformer is used to process all the reviews from the selected market segments and allows for attaching a number of describing vectors to each product, for which we have a published review. On the other hand, we can characterize each customer by the state vectors of all reviews generated by him. Given this representation, we build a market transformer as a variational auto encoder (VAE) network. The VAE models a multivariate probability distribution over a small dimensional feature space, where all product designs from a given market segment are projected. The distribution can be used to generate high probability and low probability designs using a sampling layer. The VAE is tuned to recover the vectors describing a given market segment, and on the other hand to match the consumer preferences in this market segment.

The rest of the paper is organized as follows. Section 2 gives details on structure and tuning of the language transformer. Section 3 presents the components of the VAE model, and Section 4 summarized the evaluation results in 5 product categories from the online shopping site Amazon.

2 Language Transformer

Since the information about consumer preferences is carried by the consumer reviews, which are in natural language, we design a review transformer. The transformer is a connection between an encoder and a decoder network. The purpose of the present research is not to achieve extremely accurate modelling of the natural language. Instead, we aim to ensure that basic meaning carried by review is recognized, especially with respect to tokens reflecting consumer preference, individual consumer features, and individual product features.

2.1 Embedding Layer

In the consumer review modelling, we take the smallest meaning carries to be individual words. Since consumer preference is commonly characterized in terms of keywords and grammatical content is from secondary importance, there is no need to fragment further the words into smaller units. The first step in digital language modeling is to select appropriate numerical representation of language lexemes. The common norm is to represent each token in a m dimensional vector space $\mathbf{R}^m$ - called embedding space. The vector space can carry a rich algebraic structure - summation of vectors, scalar multiplication, scalar product, cross product, etc. When defining the embedding mapping operator E, usually we desire some important properties of the linguistic relations between tokens to be reflected into the algebraic structure of the vector space.

We assume a finite dictionary of words with size n, which is also justified for the purpose of consumer review processing. Therefore, the input tokens can be mapped to an integer between 0 and n, corresponding to the index of the token in the dictionary. With this assumption, the embedding operator E can be represented with a $n \times m$ matrix, where each row contains the embedding vector for a particular token in the dictionary. The column values of a given embedding vector can be thought as a token features in selected vector space.

The training of embedding layer is not a goal of this paper. The task is usually performed over large datasets such as WordNet and Freebase, connecting words w_1 and w_2 with a structural relation R. In training, a cost is accounting for similarity mea-

sures between triplets and for negative examples for each relation, minimization is carried with a statistical gradient method. The relation R is numerically represented as a tuple of matrices acting respectively on the left and right word embeddings. Another part of the model is kernel density estimator (KDE) where a number of kernel triplets should be embedded close to triplets with at least one identical element. Interestingly entities that are close to each other in terms of 1-norm exhibit some complex similarities like "_lawn_tennis" is close to other sports like "baseball", "cricket", etc. The method represent quite well relations for proper nouns, which is generally a difficult problem with other DLNN models. (Bordes et al., 2013; Burnap and Hauser, 2018a)

For the purpose of the current research, the Global Vectors for Word Representation (GloVe) embedding model is used (Pennington et al., 2014). Its training is based on modeling one-step correlation in the corpus or equivalently - the co-occurrence of word pairs. Therefore, words with higher probability of co-occurrence share closer embedding vectors. The model is able to assign vectors within same neighborhood to words with similar meaning. Also the model reflects the words with opposite meaning with more distanced in Euclidean norm vectors.

2.2 Encoder

The purpose of the encoder is to transform the input sequence of tokens into a sequence of n_e dimensional vectors reflecting the correlational structure of the processed consumer review. The encoder is implemented as an interconnection of an embedding layer and a long short-term memory (LSTM) node. This is a widely used RNN network used in natural language models due to its capability to conserve learned local correlations in the input sequence over long processing intervals, which is a feature for common natural grammatical structures such as gender and tense matching. The LSTM structure is beneficial, especially for backpropagation training, to minimize the vanishing of the gradient for long sequences.

The input sequence r of tokens in a given consumer review is represented with a vector

$$r = (r_1, r_2, r_3, \ldots, r_L)^T \tag{1}$$

of L elements $r_i \in [1, n]$ from a dictionary with n words. If the number of elements in the actual review are less than L, then a terminating token is substituted for the missing elements. First, each input token is mapped to its embedding vector

$$r_i^{\text{emb}} = E^T \mathbf{1}(r_i), \tag{2}$$

where $\mathbf{1}(\cdot)$ is a column vector of zeros with unit only at position r_i, hence it selects the embedding vector $r_i^{\text{emb}} \in \mathbf{R}^m$ corresponding to the token r_i. The embedding vector is sent to a feedback interconnection of two LSTM layers to obtain the encoded sequence e_i

$$\begin{aligned} x_{ff,i+1} &= F_{LSTM,ff}\left(x_{ff,i}, \begin{pmatrix} r_i^{\text{emb}} \\ e_{fb,i} \end{pmatrix}\right) \\ e_i &= G_{LSTM,ff}\left(x_{ff,i}, \begin{pmatrix} r_i^{\text{emb}} \\ e_{fb,i} \end{pmatrix}\right) \end{aligned}, \tag{3}$$

where $x_{ff,i}$ is the current state of the feed forward (ff) LSTM node and $e_{fb,i}$ is the current output of the LSTM node in the feedback (fb), while $F_{LSTM,ff}$ and $G_{LSTM,ff}$ are the state transition and output mappings of the feed forward LSTM encoder node. The respective output sequence of the encoder e is

$$e = (e_1, e_2, e_3, \ldots, e_L)^T \tag{4}$$

- a sequence of n_e dimensional vectors with the same length as the input sequence in correspondence to the input tokens r_i. The feedback LSTM node action is expressed by

$$\begin{aligned} x_{fb,i+1} &= F_{LSTM,fb}\left(x_{fb,i}, e_i\right) \\ e_{fb,i} &= G_{LSTM,fb}\left(x_{fb,i}, e_i\right) \end{aligned}, \tag{5}$$

where $x_{fb,i}$ is the current state of the feedback LSTM node, $e_{fb,i}$ is the feedback signal produced from the layer, while $F_{LSTM,fb}$ and $G_{LSTM,fb}$ are the state transition and output mappings. The feedback signal is concatenated with the input token embedding, hence it acts as a model of an extracted input sequence context. As a result, the encoded sequence will model a cumulative extension of the context with the incoming input tokens. The purpose of training of the encoder is to ensure that information content of the encoded vector is increasing with each processed token.

$$I(e_{i+1}) > I(e_i) + I(r_{i+1}^{\text{emb}} | e_i), \tag{6}$$

where $I(\cdot)$ is information operator taken over the current probability distribution of the encoded state.

2.3 Decoder

The decoder operates by continuously generating tokens till either a terminating token is generated or the maximal length of the sequence is reached. In every step the decoder takes the token generated in the previous step, the state vector of the decoder from the previous step and the whole encoded sequence. Also, in every step, the decoder state vector is combined with the encoded sequence using an attention mechanism. As a result, a context vector is produced which is combined with the embedding of the previous decoded token, and consequently fed to a LSTM node, which is initialized with the decoder states from the previous step. The LSTM node, in turn, produce a state vector consumed in the next step, and an output vector - a basis for token selection. The LSTM output is projected into a distribution over the input dictionary, and on this basis a maximum likelihood token is selected.

The purpose of the decoder is to generate an output sequence with no more than L tokens, hence it can be represented as a vector

$$y = (y_1, y_2, y_3, \ldots, y_L)^T, \tag{7}$$

where $y_i \in [0, n]$ represents a single token from the input dictionary mapped to an integer interval. Again, to represent the meaning of the token in relation to other tokens in the dictionary an embedding matrix $E \in \mathbf{R^{n \times m}}$ is applied to produce an m-dimensional vector

$$y_i^{\text{emb}} = E^T \mathbf{1}(y_i), \tag{8}$$

where $\mathbf{1}(\cdot)$ is a column vector of zeros with unit only at position y_i, hence it selects the embedding vector $y_i^{\text{emb}} \in \mathbf{R}^m$ corresponding to the token y_i. The actual decoding is produced by an LSTM memory cell taking the embedding y_i^{emb} together with an m-dimensional context vector c_i to produce

$$\begin{aligned} x_{i+1} &= F_{LSTM,D}\left(x_i, \begin{pmatrix} y_i^{\text{emb}} \\ c_i \end{pmatrix}\right) \\ d_i &= G_{LSTM,D}\left(x_i, \begin{pmatrix} y_i^{\text{emb}} \\ c_i \end{pmatrix}\right) \end{aligned}, \tag{9}$$

where $x_i \in \mathbf{R}^{2n_d}$ represents the current state of the LSTM node with n_d units in the forward and backward channels, $F_{LSTM,D}$ and $G_{LSTM,D}$ are state transition and output functions of the LSTM node, and $d_i \in \mathbf{R^{n_d}}$ is the output signal. The output from the LSTM layer will be interpreted as distribution over dictionary $\mathcal{L}$. Hence, a dense layer is trained to make a projection from LSTM n_d dimensional space into embedding m dimensional space,

$$\mu_i = \sigma\left(EW_3(d_i)\right), \tag{10}$$

where $\sigma(\cdot)$ is a `softmax` operator translating the n-dimensional vector into a probability distribution

$$\mu_i : [0, n-1] \to (0, 1) \tag{11}$$

over the input dictionary. The token at the position $i+1$ is then selected using maximal likelihood criteria as

$$y_{i+1} = \underset{z}{\operatorname{argmax}}\, \mu_i(z). \tag{12}$$

The attention mechanism is used to generate the context vector c_i into the embedding m-dimensional space. Being in that space, the context can be expressed in the words of input dictionary with the help of scalar product. First, the previous decoder LSTM state vector x_i is projected into n_d dimensional LSTM output space through a linear dense layer $W_2 \in \mathbf{R}^{n_d \times 2n_d}$. Similarly, every vector from the encoded sequence e with length L is projected into decoder LSTM output space with another linear mapping $W_1 \in \mathbf{R}^{n_d \times n_e}$. The attention distribution vector

$$a_i = (a_{i1,}, a_{i,2}, \ldots, a_{i,L}), \tag{13}$$

where $a_{i,j} \in (0, 1)$ is then produced with a softmax operation

$$a_i = \sigma(s_i) \tag{14}$$

from the scores vector s_i with elements $s_{i,j} \in \mathbf{R}$ calculated as

$$s_{i,j} = V(W_1 e_j + W_2 x_i), \tag{15}$$

through a linear mapping $V \in \mathbf{R^{1 \times n_d}}$ acting over the LSTM output space. As can be seen, the projection of current decoder state is used to bias the projections of the encoded tokens. This allows shifting of attention focus from one part of the input sequence to another, as a function of current decoder state. The attention values $a_{i,j}$ function as weights over the elements of the encoded sequence producing the context vector

$$c_i = \sum_{j=1}^{L} a_{i,j} e_j. \tag{16}$$

2.4 Training

The training of the language transformer is performed over a random sample of sequences from the database of review texts between minimal and maximal word length. First, each input word sequence r is encoded into a sequence of vectors e. Then, each encoded vector is processed through the decoder till a sequence y with the same length L as input is produced. The loss function is calculated as

$$J_{LT} = -\sum_{i=1}^{L} E_{r_i}(ln(\mu_i)), \tag{17}$$

where $-E_{r_i}(ln(\cdot))$ represents cross entropy operator. The training is performed for as many epochs as we see a satisfactory generated sequences from decoder over the train and validation sample.

3 Market Transformer

The market transformer is developed as a variational auto encoder, which acts as a model for the probability distribution

$$p(x_d|x_c), \tag{18}$$

where $x_d \in \mathbf{R^{n_e}}$ is a vector of design features and $x_c \in \mathbf{R^{n_e}}$ is a vector of consumer features for a particular market segment. The probability distribution contains the information of preferred designs by a particular consumer. Because the output dimension of the review encoder can become relatively large depending on the linguistic complexity of the textual reviews, we assume an underlying low dimensional feature space for the products $h_d << n_e$ and for the consumers $h_c << n_e$.

3.1 Dataset Preparation

For the customer review dataset, we use publicly available data for Amazon Product Reviews available by product categories collected in 2023. The whole dataset contains about 48 million products, 571 million reviews and 54 million consumers (Hou et al., 2024). Additionally, a metadata is provided for each item describing user rating, item price, and other features. The dataset parsing is organized by focusing on a finite number of market subsegments identified by name. Data is processed on a monthly basis from a fixed starting year to a fixed ending year. The information is organized into table with columns - review text, sampling period, client identification, product identification, price, product category. The review rating is used to filter only reviews with 4 and 5 stars, which mark the positive consumer preferences. Also, review texts are filtered by minimal and maximal length to keep consistency.

The review text is filtered by converting all words to lower case and discarding symbols which are not from the alphabet or digits. Each of the reviews in the selected market segments is processed through the encoder to produce the corresponding encoded sequence from n_e dimensional vectors. As noted in section (), due to the feedback structure of the encoder, the latest element in the encoded sequence will be with the highest information content. Hence, we can take only the last element from an encoded sequence, namely e_L to characterize the input sequence r.

The dataset information is in the form of triplets (C, R, P), C standing for client, R standing for textual review and P standing for product. Since each textual review is encoded and represented with a vector e_L, we assume that the vector identifies partly both - the customer who generated the review and the product which receives such a review. In this respect, each product can be uniquely identified by all final encoder states produced by its reviews. Instead, on product level, the model is trained to work with a category of products. Such a generalization is justified to improve the statistical properties of the model. The encoding vectors e_L of a product category P form the design space for this category, denoted as

$$\hat{X}_P = (e_L(r^1), e_L(r^2), \dots e_L(r^{N(P)})), \tag{19}$$

for each review r^l for a product in the category P. The number of reviews $N(P)$ may vary for different product categories.

To each product category corresponds a customer category. A customer u is characterized with all the reviews he has generated collected in a matrix from final encoder states as

$$\hat{X}_u = (e_L(r^1), e_L(r^2), \dots e_L(r^{N(u)})), \tag{20}$$

and the customer category contains all the characterizations from all the customers

$$\hat{X}_C = (X_{u,1}, X_{u,2}, \dots, X_{u,N(C)}), \tag{21}$$

with $N(C)$ being the number of customers in a category C.

3.2 Consumer Utility

The consumer utility model characterizes the interaction between product category and consumer category with a multivariate probability distribution in a product feature space X_H. The product characterized by a matrix X_P is processed through a dense linear layer W_4 into small dimensional representation as

$$Y_P = W_4 H_P. \tag{22}$$

Then the multivariate mean and multivariate standard deviations of the underlying feature space with dimension h_d are obtained by two layers with rectified linear activation as

$$\mu_p = W_\mu(y_P), y_P \in X_P \tag{23}$$

and

$$\sigma_p = W_\sigma(y_P). \tag{24}$$

This estimates a multivariate normal distribution with parameters $N(\mu, \sigma)$. After this a sampler layer generates a random sample from this disribution in the h_d dimensional space as

$$s_p \propto N(\mu, \sigma) \tag{25}$$

The random sample is sent to a dense decoder layer composed of linear hidden layer and output layer with n_e units to produce a esimate of the encoded product

$$\hat{y}_P = W_{dec}(s_p) \tag{26}$$

3.3 Training

The training of the market transformer is with respect to 3 objectives - minimize the difference between a given product vector e_p and the decoded vector $\hat{e}_p$

$$J_{MT,vae} = E_{X_P}(e_p - \hat{e}_p), \tag{27}$$

minimize the cross entropy predicted by the model for a particular consumer $p_M(e_p, e_c)$ with respect to the ground truth triplet π of whether this customer actually bought the item

$$J_{MT,ch} = E_\pi(-ln(p_M)), \tag{28}$$

and a regularization term of not allowing underlying mean and variance vectors to grow unbounded

$$J_{MT,reg} = E_{X_C}(W_\mu(X_P, e_c)^2 + W_\sigma(X_P, e_c)^2). \tag{29}$$

The three objective are combined in the cost

$$J_{MT} = J_{MT,reg} + J_{MT,ch} + J_{MT,vae} \tag{30}$$

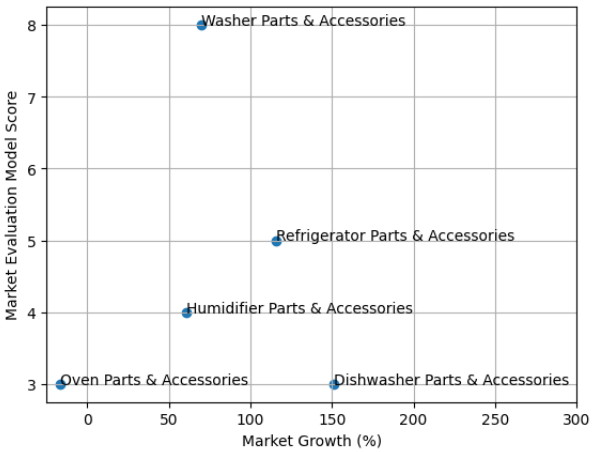

Figure 1: Model Prediction vs Market Capitalization Growth

4 Results

The following figures present the obtained results for 5 categories of Amazon reviews. The design gap in a given market segment is predicted by looking for low probability designs (i.e. not yet discovered) through a Monte-Carlo sampling of the underlying multivariate probability distribution. The low probability products are evaluated with respect to the consumer preference probability function $p_M(x_d|x_c)$. Hence in case of low probability design, which is highly preferable by customers we can conclude that in a given market segment we have unrealized potential for innovation.

To validate the correctness of such predictions, we compare the observed market growth in the 5 segments from 2013 to 2018 year. Figure 1 gives information about observed market capitalization growth and predicted design gap in the sectors. We observe positive correlation between models scores and capitalization growth, which means that the model correctly predicts future capital allocation in the observed sectors.

In Figure 2 we observe the product survival rate vs market prediction, where again we see a positive correlation with the model predictions. The product survive rate is characteristic to how much a given product is in demand during the observed period. In Figure 3 we examine correlation between market scores and actually appearing new products in the observed domains. Here also a positive correlation with the predicted design gap is observed.

For a reference of the size of the observed market segments, we calculate their market capitalization at the starting year in Figure 4.

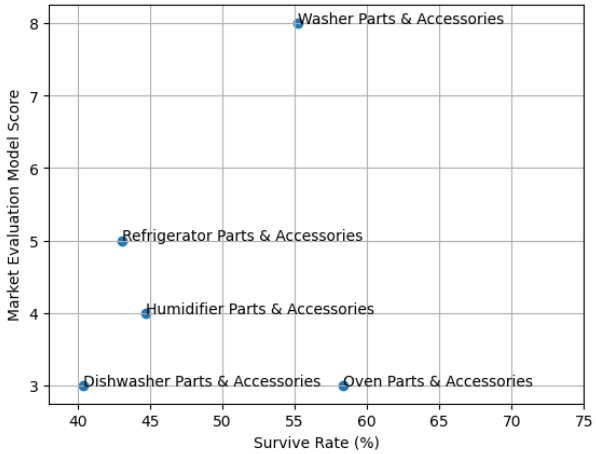

Figure 2: Model Prediction vs Survive Rate

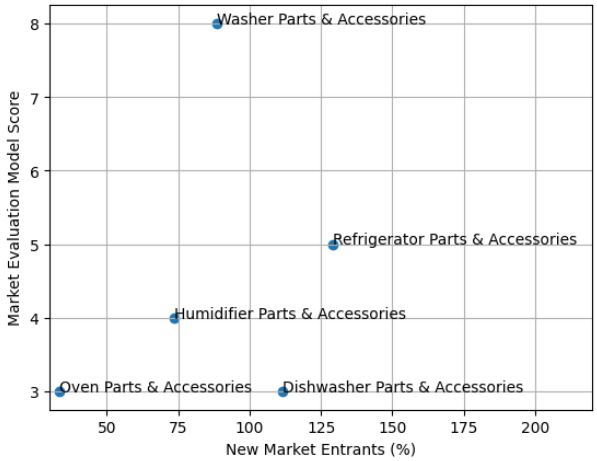

Figure 3: Model Prediction vs New Market Entrants

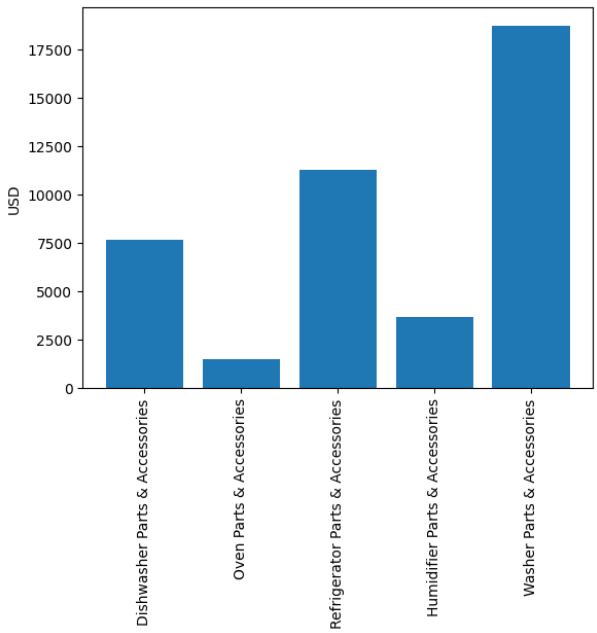

Figure 4: Market Capitalization

5 Conclusion

The present research combined key techniques from LLM research to build a simplified language transformer for encoding essential information in textual user reviews used to characterize the consumer preference in online marketing. The applicability of this encoded information is demonstrated for encoding actual consumer preferences using a variational auto encoder model. The developed market segment transformer allows prediction of possible low probability designs in a given market which might be highly favored by its eventual consumers.

Acknowledgments

This study is financed by the European Union-NextGenerationEU through the National Recovery and Resilience Plan of the Republic of Bulgaria, project no. BG-RRP-2.004-0005.

References

A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *Neural Information Processing Systems (NIPS)*, pages 1–9.

A. Bordes, J. Weston, R. Collobert, and Y. Bengio. 2011. Learning structured embeddings of knowledge bases. In *in Proc. 25th AAAI Conference on Artificial Intelligence*, pages 301–306.

A. Burnap and J. Hauser. 2018a. Predicting "design gaps" in the market: Deep consumer choice models under probabilistic design constraints.

Alex Burnap and John Hauser. 2018b. Predicting "design gaps" in the market: Deep consumer choice models under probabilistic design constraints.

Yupeng Hou, Jiacheng Li, Zhankui He, An Yan, Xiusi Chen, and Julian McAuley. 2024. Bridging language and items for retrieval and recommendation. *arXiv preprint arXiv:2403.03952*.

B.M. Lake, R. Salakhutdinov, and J.B. Tenenbaum. 2015. Human-level concept learning through probabilistic program induction. *Science AAAS*, 350(6266):1332–1338.

A. Paccanaro and G.E. Hinton. 2001. Learning distributed representations of concepts using linear relational embedding. *IEEE Transactions on Knowledge and Data Engineering*, 13(2):232–244.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Max Yi Ren and Clayton Scott. 2017. Adaptive questionnaires for direct identification of optimal product design.