Configuring products with natural language: a simple yet effective approach based on text embeddings and multilayer perceptron

Yue Wang, Xiang Li, Linda L. Zhang & Daniel Mo

INTERNATIONAL JOURNAL OF PRODUCTION RESEARCH
https://doi.org/10.1080/00207543.2021.1957508

Configuring products with natural language: a simple yet effective approach based on text embeddings and multilayer perceptron

Yue Wang[a], Xiang Li[b], Linda L. Zhang[c] and Daniel Mo[a]

[a]Department of Supply Chain and Information Management, The Hang Seng University of Hong Kong, Hong Kong SAR, People's Republic of China; [b]Department of Information Systems, The City University of Hong Kong, Hong Kong SAR, People's Republic of China; [c]IESEG School of Management, Univ. Lille, CNRS, UMR 9221 – LEM – Lille Economie Management, Lille, France

ABSTRACT
Product configurators are recognised as critical toolkits enabling customers to co-create products with companies. Most available product configurators require customers to select suitable product attributes from predefined options. However, customers usually find the selection processes frustrating due to their lack of product knowledge. In view of the fact that customers often express their needs in imprecise and vague natural language, we define a new needs-based configuration mechanism and propose an implementation approach based on text embeddings and multilayer perceptron. Specifically, we leverage the massive amount of product reviews by encoding them into text embeddings. A multilayer perceptron is trained to map text embeddings to product attribute options. Experiment results indicate that the mapping has good generalisation capability to map customer needs into product configurations. The performance of our approach is comparable to that of deep learning-based approaches but with much higher efficiency in terms of computational complexity. Our needs-based configuration thus provides a quick and effective means of facilitating product customisation. It also demonstrates an innovative way of utilising customer resources in unstructured text to co-create products with companies.

ARTICLE HISTORY
Received 30 December 2020
Accepted 10 July 2021

KEYWORDS
Product customisation; needs-based configuration; text embedding; multilayer perceptron; natural language processing

1. Introduction

Since Digital Equipment Corporation introduced R1 (later called XCON) to configure VAX computer systems (McDermott 1982), product configuration and configurators have drawn increasing attention from academia and sparked lasting interest in industry. When configuring products, customers need to select product attribute options (also called 'components' in the literature) from a predefined set. The combinations of the options form the satisfactory product variants. Product configurators are recognised as efficient tools for facilitating product customisation and as critical toolkits enabling customers to co-create products with companies and realise mass customisation and even product innovation (Forza and Salvador 2007; Wang et al., 2020a). Thanks to the potential benefits of product configurators, such as lead-time reduction, routine work reduction, and product quality improvement (Kristjansdottir et al. 2018), they have been widely applied by various companies to configure customised products (Zhang 2014). Some well-known examples include Dell, Cisco Systems, American Power Conversion, and Reebok (Chen, Wang, and Tseng 2009; Trentin, Perin, and Forza 2012). In accordance with the growing trend of customer-centred design and design for user experience, configurators may play an increasingly critical role in business in the future. According to the cyLEDGE configurator database (2021), the number of existing online configurators exceeds 1400, spanning 17 industries.

Nevertheless, challenges persist for customers to select suitable product attributes that satisfy their needs in product customisation and co-creation processes. Most available configurators are parameter-based and require customers to possess certain domain knowledge to select the desired attribute choices (Piller and Walcher 2012). However, customers generally do not possess sufficient technical knowledge about products, and they do not have the expertise to deal with unfamiliar products (Wang, Mo, and Tseng 2018). It is not uncommon that customers express their expectations and needs in laymen's terms or natural languages. For example, customers describe their needs using, e.g. 'a cool racing car,' 'a good mobile phone,' or 'a high-quality laptop.' As a result of the gap between customer needs and product specifications, customers are often overwhelmed by the many complicated options offered in

CONTACT Yue Wang yuewang@hsu.edu.hk

configurators and become confused and frustrated when making their selections. Consequently, such unpleasant, stressful selection experiences often lead customers to abandon configuration processes.

In a large-scale industrial benchmark study on mass customisation, i.e. the Customisation 500 project (Piller and Walcher 2012), it is revealed that parameter-based configurators are suitable in B2B businesses where customers are professionals and have much knowledge of products, instead of B2C businesses where customers quite often lack product knowledge. Authors have highlighted the importance of developing user-friendly choice navigation capabilities in configurators (Salvador, De Holan, and Piller 2009; Trentin, Perin, and Forza 2013). Recognising the limited applicability of conventional parameter-based configurators in B2C business, Randall, Terwiesch, and Ulrich (2007) introduced needs-based systems. These systems include predefined customer expectation-related statements, e.g. 'The laptop should be portable so that I can easily carry it.' When configuring products, customers merely need to indicate on a Likert scale the degree to which the given statements represent their needs. With these selections, potentially satisfactory products are configured and recommended to customers. While these needs-based systems allow customers with little or no product knowledge to configure products, they do not offer customers the ability to freely express their true needs. In this regard, such systems are not completely needs-based.

In this study, we put forward true needs-based configuration to address the above limitations of parameter-based configurators and configuration activities. The needs-based configuration proposed in this study transforms customer needs expressed in natural languages to satisfactory product configurations. It can greatly improve user experience (UX) through the easy access to the content in the product co-design space and the intuitive choice navigation using natural languages, corresponding to the sensory design and navigation design elements in UX theory (Garrett 2010). Specifically, the needs-based configurator is defined as follows:

- Given: (A) a product configuration space consisting of a set of pre-defined attributes (or components), and (B) some description of customer needs in a natural language;
- Build: A list of attribute choices that are most likely to satisfy customer needs.

To implement the proposed needs-based configuration, it is necessary to understand the complicated statements in needs texts using natural language processing (NLP) techniques (Barco, Vareilles, and Osorio 2018). This requires an ample amount of customer needs text to develop viable mapping from needs to product attribute specifications. However, there is no large-scale, publicly available customer needs dataset available (Cooper and Edgett 2009). Being widely available on e-commerce websites, product review data are similar to customer needs data in terms of information format, scope, and content. Timoshenko and Hauser (2019) demonstrated that in terms of efficiency and completeness, customer needs mined from product review data are as good as, if not better than, those obtained from traditional customer needs elicitation methods. Therefore, we leverage the massive amount of online product review text to carry out our needs-based configuration.

Thanks to their capability of extracting semantic information from text, deep learning-based approaches have achieved state-of-the-art performance in various NLP tasks. However, these approaches require a very long model training time and demand a large amount of training data (Fu and Menzies 2017). Consequently, it is difficult to apply these approaches for industrial uses (Akinosho et al. 2020). Thus, it is deemed important and relevant to develop a lightweight approach yet with sufficient capability to extract information from the text to facilitate practical use. Recognising their superiority in information extraction and less data requirement, in this study, we develop an implementation approach based on two techniques: text embeddings and a multilayer perceptron neural network (MLP). Based on the approach, free-form product review texts are transformed into fixed-length numerical vectors using word embeddings; text embeddings are subsequently constructed by encoding reviews based on word vectors. With text embeddings as input, MLP is employed to configure products by classifying product attribute choices and subsequently mapping customer needs to the classified product attribute choices. We conduct experiments and analysis to demonstrate the applicability of our proposed approach. The results show that our approach is effective, efficient, and robust.

Our study makes several contributions to both literature and practice. First, we define needs-based configuration to allow customers to express their needs in natural languages when configuring products. In accordance with the fact that user-centred design and B2C business are becoming more and more popular (even prevailing) in many industries, developing user-friendly configurations for customers to obtain products that truly meet their needs is vitally important. Based on the latest cyLEDGE configurator database (2021), needs-based configurators are still fewer than 1% among the total 1,400 configurators, whilst practitioners expect such a tool. Second, we develop an implementation approach based on text embeddings and multilayer perceptron.

The approach automatically transforms customer needs expressed in natural languages into satisfactory product configurations, thus significantly facilitating product customisation in B2C businesses. As demonstrated in theoretical analysis and computational experiment, our proposed approach achieves comparable performance with deep learning-based ones yet with higher efficiency and much lower requirement in terms of computational complexity. This resource-effective approach is, thus, particularly useful for practical use. Third, our needs-based configuration and the implementation approach are expected to advance product co-design research. Our study leverages external crowd wisdom, i.e. customers' product reviews, to improve the effectiveness and efficiency of product customisation service. Thus, from the perspective of companies, our proposal can serve as a co-design tool helping companies utilise various formats of customer resources to co-create company value. To summarise, our approach (1) requires far fewer computational and expertise resources from companies; (2) bridges the semantic gap between product specifications and customer needs, which are usually expressed in natural language utterances; and (3) provides a new perspective for companies to mine external unstructured data and to improve product customisation service.

2. Relevant work

Understanding the potential benefits of configuration, researchers have made countless investigations to address various issues. Consequently, a multitude of articles has been published. We classify the related work into two groups. The first group focuses on product configuration solutions and the second group configurator development.

2.1. Configuration solutions

Many studies have been carried out addressing configuration solving. Some authors focused on generating feasible configurations based on constraints and rules (Xie, Henderson, and Kernahan 2005; Yang and Dong 2013), while some others investigated configuration optimisation in addition to configuration generation (Pitiot et al. 2020; Zhou, Lin, and Liu 2008). With regard to the techniques applied or approaches proposed in the solutions, some authors used information theory (Wang and Tseng 2007), genetic algorithm (Yeh and Wu 2005), binary tree algorithm (Tseng and Chang 2006), case-based reasoning (Tseng, Chang, and Chang 2005), and data mining (Song and Kusiak 2009). Deep learning-based approaches, e.g. attention networks, have also been investigated to map customer needs to product configurations (Wang et al, 2020; Wang, Zhao, and Wan 2021). These approaches are generally complicated, and entail heavy computational workload in model training and application stages, which limits their usage in practice (Fu and Menzies 2017; Akinosho et al. 2020). In view of the negative influences of carbon emissions on social and economic development, some authors started involving emission reduction in product configuration. Tang, Wang, and Uilah (2017) developed a bi-objective optimisation model to consider both customers' satisfaction towards product configuration and the corresponding environmental impact.

Some researchers developed configuration solutions involving recommendation modules to present the relevant customised products to consumers. De Bruyn et al. (2008) leveraged preference models to design questionnaire-based configuration interface to capture customers' preferences. They applied cluster classification, Bayesian tree regression, and stepwise componential regression to offer online recommendations. Tiihonen and Felfernig (2010) applied various machine-learning algorithms to recommend configurable products. The algorithms they considered are nearest neighbour, naïve Bayes classifier, majority voting, and most-popular choice. Similarly, using a content-based reasoning approach, Triki, Mazo, and Salinesi (2014) combined configuration with recommendation. However, their configuration process is not optimal and the cold-start problem in recommendation exists.

While the above studies present different solutions or consider different criteria for implementing product configuration, they address configuration solving from an engineering perspective and include predefined product attribute options. Most of them do not consider customer needs to be expressed in natural languages. Though the deep learning-based approaches excel in handling NLP tasks, they require high computational resources, making them unsuitable for industrial applications.

2.2. Configurator development

In an early study on configuration, Franke (1998) characterised the configuration models underpinning product configurators. The models he discussed start with product attributes and predefined options without considering customer needs expressed in natural languages. Upon examining the empirical evidence, Bramham and MacCarthy (2003) described a framework for matching configurator attributes with business strategies. One attribute is customer's selection of predefined attribute options.

Drawing on different perspectives, researchers have also developed approaches and frameworks to support configurator development. To facilitate the automation of configurator development, Honigsberg, Kollwitz, and

Dinter (2019) proposed a reference model. Their reference model does not include customer needs expressed in natural languages. It also does not address how the needs are linked to product attribute options. Ong, Lin, and Nee (2006) discussed an approach based on the object-oriented technique and invasion-based search algorithms with a web-based configurator prototype. Their approach and prototype involve customers' selection of predefined functional features and attribute options. Zheng et al. (2017) put forward a framework for developing personalised configurators that were applicable in engineering-to-order environments. Unlike customers in engineering-to-order environments, customers in B2C environments do not have sufficient product knowledge. As a result, their framework may not be useful for developing configurators for B2C businesses.

Researchers also discussed tools or techniques to facilitate configurator development by effectively modelling and documenting configuration knowledge. Haug and Hvam (2007) discussed Product Variant Masters and highlighted their striking feature: easily understandable in modelling domain knowledge. Later, Haug, Hvam, and Mortensen (2010) presented Class Responsibility Collaboration cards and Vertically Aligned Class Diagrams, respectively, and discussed how they can be used to document the knowledge base of configurators. Recognising the limitations inherent in the above modelling techniques, Rasmussen et al. (2020) developed an approach combining several modelling and analysis techniques, such as unified modelling language and the scope, commonality, and variability analysis, to facilitate configuration knowledge modelling. Jiao and Yang (2019) proposed a production configuration approach based on online data. By using the supervised predictive model of relevance vector machine (RVM) classifier, an efficient and competitive configuration approach was developed. The common feature in the above techniques is that they are developed to handle the selection of predefined product attribute options, instead of handling customer needs expressed in natural languages.

To summarise, while most relevant studies have addressed configuration issues in relation to the selection of predefined product attribute options, very few deals with customer needs expressed in natural languages.

3. Text embedding and MLP-based configuration approach

3.1. Approach overview

Being consistent with literature, in this study, we view a product as a combination of its attribute values, for example, (i8 processor, 16GB DDR3 RAM, 512 GB SSD, Nvidia V100 graphic processor, 17' monitor) for a laptop. To be more rigorous, we use vector $(a_{k1}, a_{k2}, \ldots, a_{kn})$ to represent product k, which consists of n attributes. Each review of the product is actually a word sequence in a free format. The reviews $\{R_{ki}\}_{i=1}^{N_k}$ of product k are associated with a set of labels $[a_{k1}, a_{k2}, \ldots, a_{kn}]$, where N_k is the number of reviews. In this way, product $(a_{k1}, a_{k2}, \ldots, a_{kn})$ has all the features mentioned in the reviews.

NLP and machine learning techniques are adopted in this study to understand language utterances and to distil the relevant product information from text. We model the needs-based configurator in the proposed approach as a set of MLP classifiers, which organise product knowledge, to map customer needs to product attributes. The approach includes four major steps as shown in Figure 1, including (1) encoding words in online reviews to numerical vectors using word embeddings v_i; (2) constructing text embeddings vector T based on word vectors; (3) training a set of MLP classifiers $f_i : R \rightarrow a_i\ (i = 1 \ldots n)$ to classify product attribute options based on the set of product review $R = \{\{R_{ki}\}_{i=1}^{N_k}, k = 1, 2, \ldots m\}$, where m is the number of products in the whole text corpus; and (4) configuring products using the trained MLP classifiers. The following sections elaborate the four steps.

3.2. Encode words to numerical vectors using word embeddings

Words are the basic element in a human language, and a meaningful language expression usually consists of a set of words. Conventionally, a word is represented using a one-hot method that quantifies each word with a binary vector. Only one dimension in the binary vector has value 1 indicating the order of the word in the vocabulary. All the other dimensions have value 0. Thus, the number of dimensions of the one-hot representation is the same as the size of the vocabulary. However, the one-hot representation contains no semantic information of the text. Furthermore, the vocabulary size is usually big, leading to an extremely high dimension of one-hot representation. This makes it hard to process the text efficiently. To solve this issue, word embedding is introduced to represent words in a continuous and relatively lower dimensional vector space (Turian, Ratinov, and Bengio 2010). A two-layer neural network is used to map a word to a word embedding, i.e. the vector of real numbers with lower dimensions. In addition, the word embedding can encode the semantic information for the word. For example, the embeddings of the words man, woman, king and queen have the approximated relationship v (man) $-$ v (woman) $=$ v (king) $-$ v (queen), where v

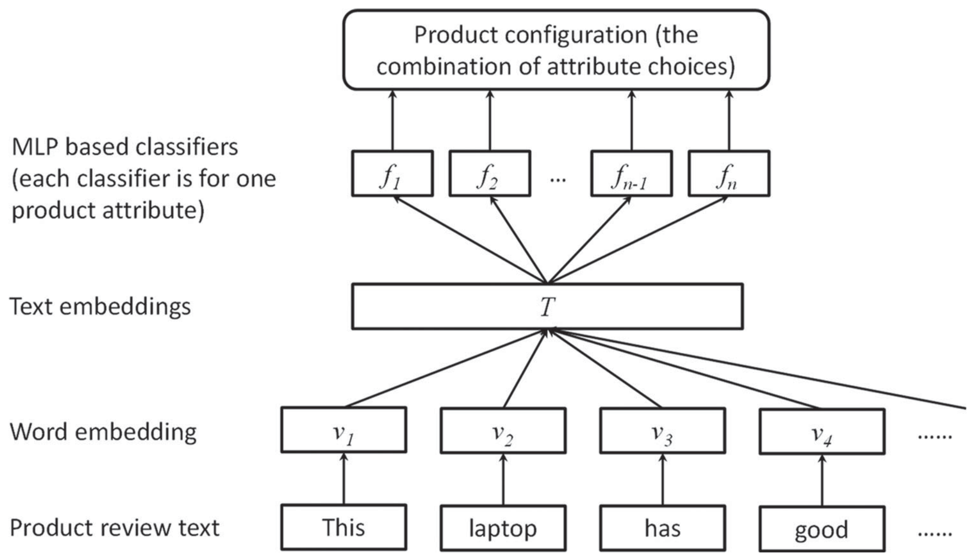

Product configuration (the combination of attribute choices)
MLP based classifiers (each classifier is for one product attribute)
f_1 f_2 ... f_{n-1} f_n
Text embeddings
T
Word embedding
v_1 v_2 v_3 v_4 ......
Product review text
This laptop has good ......

Figure 1. Overview of the text embedding and MLP-based approach.

() represents the operation to get the word embedding. Currently, there are many methods to realise word embedding. Popular methods include Continuous Bag-of-Words (CBOW), Skip-Gram Model, and Global Vector (GloVe) (Pennington, Socher, and Manning 2014). They can achieve effective word embedding and have been widely used in different NLP tasks. We adopt GloVe word embedding to encode the words in the text. For example, the word apple is represented by a 300-dimension numerical vector [0.52042001, −0.83139998, 0.49961001, 1.28929996, 0.1151, 0.057521, ...]. In this way, any word's embedding can be looked up in GloVe.

3.3. Construct text embeddings

With the word embeddings obtained above, this step encodes online reviews to numerical values, i.e. text embeddings. We treat a text embedding as a function of word embeddings and combine word embeddings into text embeddings. To combine word embeddings, we adopt three popular approaches, including (1) element-wise average, (2) max pooling, and (3) concatenation of element-wise average and max pooling (Shen et al. 2018).

The element-wise average approach retains all the information about each word in customer needs text via the average operation. Given a specific customer need S consisting of L words: $S = \{w_1, w_2, \ldots, w_L\}$ and the word embeddings of the L words: $\{v_1, v_2, \ldots, v_L\}$, the approach calculates the embedding of S, denoted as T^{aver}, as the element-wise average of the word vectors: $T^{\text{aver}} = \frac{1}{L}\sum_{i=1}^{L} v_i$.

In practice, it is not uncommon that a small number of keywords play a critical role in customer needs. For example, 'fast' is the keyword in 'I want a fast laptop.' To consider the saliency of keywords, the max pooling approach is used. More specifically, this approach ignores words that do not contain much information or are unimportant, extracts the salient elements from each word embedding, and combines them to form text embeddings. Based on this approach, the text embedding $T_i^{\max}$ of a specific customer need S consisting of L words is $T_i^{\max} = \max\{v_{1i}, v_{2i}, \ldots, v_{Li}\}$, where v_{ki} is the i-th element of word vector v_k.

The above two approaches encode different kinds of information about customer needs, thus complementing each other. To fully leverage the benefits of both approaches, we adopt the approach of concatenated element-wise average and max pooling text vectors to form a new vector with the dimensions doubled, i.e. $T^{\text{concat}} = [T^{\text{aver}} : T^{\max}]$.

3.4. Train MLP classifiers

We train the MLP with one input layer, two hidden layers, and one output layer as the classifiers to realise the needs-based configurator. In relation to the fact that a word might be associated with multiple attributes, these MPL classifiers in our approach can be learned to map a word to the corresponding attribute using the training data. The MLP structure is shown in Figure 2. Let $\boldsymbol{x}_1$ denote the input layer. $\boldsymbol{x}_1$ is a text embedding vector $(x_{1,1}, x_{1,2}, \ldots, x_{1,n_1})$ obtained from the preceding step, i.e. $\boldsymbol{x}_1 = \boldsymbol{T}^{\boldsymbol{j}}, j = 1, 2, 3$, where j is the index of the three approaches to encoding review texts. Each hidden layer is modelled as $\boldsymbol{x_i} = \boldsymbol{\sigma}(\boldsymbol{W^i x_{i-1}} + \boldsymbol{b^i})$, $i = 2{,}3$. $\boldsymbol{W^i}$, $i = 2{,}3$ is the weight matrix connecting the preceding layer with

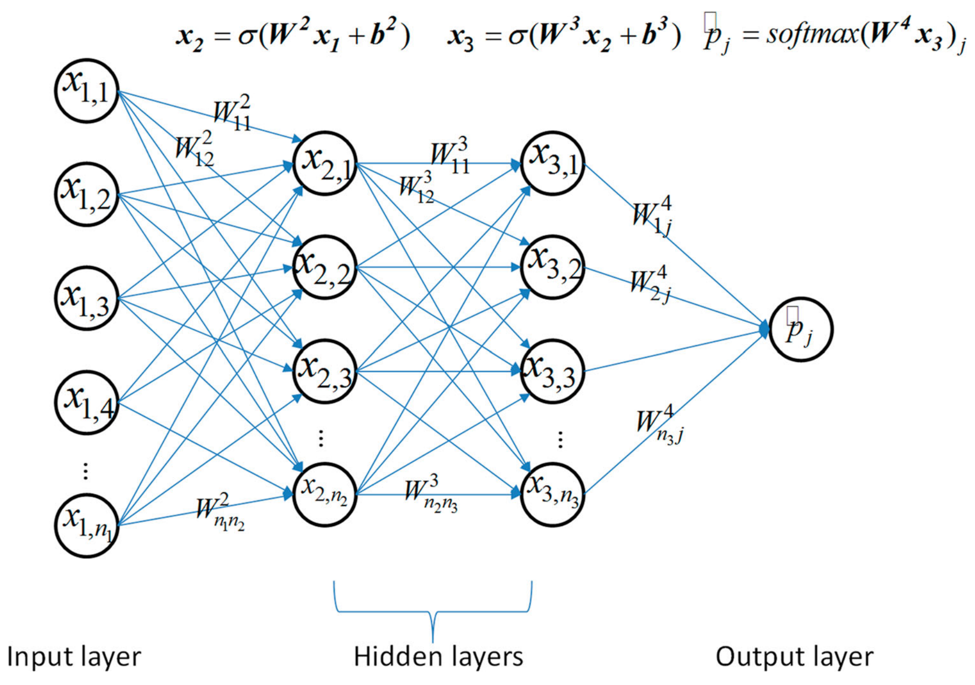


Figure 2. The structure of the MLP.

the succeeding layer. Each matrix element $\boldsymbol{W}^{\boldsymbol{i}}_{\boldsymbol{j}}$ connects the input from the i-th layer to the j-th unit of the i-th layer. $\boldsymbol{b}^{\boldsymbol{i}}$ represents the bias vector of a hidden layer i. In this study, the ReLU function $\boldsymbol{\sigma}$ presented in (Nair and Hinton 2010) is used as the activation function for the two hidden layers, that is, $\boldsymbol{\sigma}(\boldsymbol{x}) = \max(0, x)$.

As the output layer, a SoftMax layer (Gao and Pavel 2017) is trained as the function to classify product attribute choices. SoftMax calculates the probability that each attribute choice satisfies the customer's needs, i.e. $\hat{p}_j = \text{softmax}(W^4 x_3)_j = \frac{exp(W^4_j x_3)}{\sum_{k=1}^{C} exp(W^4_k x_3)}$, where $\boldsymbol{W}^{\boldsymbol{4}}$ is the weight matrix, and C is the total number of product attributes (i.e. the total number of classes). The numerator of the SoftMax equation, $exp(W^4_j x_3)$, quantifies the likelihood that each attribute choice satisfies the customer's needs; $\sum_{k=1}^{C} exp(W^4_k x_3)$ is the sum of all the likelihoods. Thus, $\hat{p}_j$ indicates the probability value at which each attribute choice satisfies customer needs. Accordingly, the trained MLP classifiers organise n lists of attribute choices based on a descending order of probability values. All the parameters of the weight matrix and bias vector can be learned by minimising the cross entropy between predicted and true class distributions using a backpropagation algorithm (Haykin 1998).

3.5. Configure products by mapping customer needs to attribute choices

Upon completion of training in Step 3, in Step 4 the set of n trained MLP classifiers are used to configure products by mapping customer needs texts to product attribute choices listed in the descending order of the corresponding probability values. In detail, a new customer needs text is encoded into text embeddings by combining the numerical vectors of each word in the text. The trained MLP classifiers take the text embeddings as input. For each of the n product attributes, they output the probability value for which the corresponding attribute choice satisfies the input text. For each attribute, the choices are then presented to customers based on a descending order of probability values.

4. Experiment settings

4.1. Dataset

We use laptops as an example to verify the proposed approach to implementing needs-based configuration. As well-modularised products, laptops/PCs are among the earliest successful applications of product configurators (Tseng, Wang, and Jiao 2018). For example, Dell gained great success using the configure-to-order approach in the 1990s.

We obtain product review data by data-crawling amazon.com. The dataset includes review data published in January 2017–June 2018 for 91 laptop models and provides a total of 7227 reviews. (Note: These 7227 reviews are the original reviews without data cleaning.) The product review text and the laptop specification pairs are used as the training set to train all the parameters in the MLP, particularly the weight matrix and the bias vector of each unit.

In pre-processing the product review data, we first identify five typical laptop product attributes, including processor, monitor, RAM, hard disk, and graphic processor. Because each attribute may have many possible alternatives, we consolidate the attributes by combining similar options. For example, i7-8665UE and i7-8550U are both eighth-generation processors introduced by Intel in 2018. Their performance and prices are similar, and they can be used interchangeably in a laptop model. Thus, they are combined as one option. After pre-processing, the number of options for each attribute is as follows: 10 options for processor, 6 options for monitor, 6 options for RAM, 8 options for hard disk, and 8 options for graphic processor.

We also conduct an online survey to collect customer needs text for the dataset. Survey respondents are exclusively university students with an engineering or business background. All respondents have their own laptops and possess basic knowledge on laptops. Each respondent is required to write down their needs for a laptop; subsequently, he (or she) is asked to select the most satisfactory option from the laptop models in the dataset. In total, we obtain 1,124 need texts and the corresponding satisfactory laptops. Each need text contains different words and the number of words in each need text ranges from 10 to 150 (Example of a long need with 91 words:

> I am currently using a Lenovo Z series laptop which was bought three years ago. It functions well but it is too heavy and not very convenient to carry around. So I want to buy a light and portable laptop. I have enough budget. So price is not a concern. I want the performance to be as good as possible. As I don't use iOS, Macbook will not be considered. I usually play video game using the laptop. Therefore, the screen resolution should be high and screen size should be big;

Example of a short need: 'I am a student and want a small laptop for the basic purpose of email, word processing and web surfing.') The needs and laptop specifications (i.e. attribute choices) are used as the testing dataset.

4.2. Experiment setting and performance metric

In this study, 300-dimension GloVe word embeddings are used as the basic elements to build text embeddings. Words that do not exist in the vocabulary are initialised using a uniformly distributed random vector in the range of $[-0.01, 0.01]$.

The trained MLP classifiers work in a way that is similar to an information retrieval system. Specifically, the classifiers take new customer needs texts as input and generate a list of attribute choices as output, based on the probabilities calculated by the SoftMax layer in MLP. The attribute choice with the highest probability of satisfying the active customer's needs will be shown at the top of the list. It is followed by other choices with lower probabilities of satisfaction. In such a system, the results presented not only consider the relevance and likelihood of satisfaction of each attribute choice, but also the order in which the choices are presented. Intuitively, the satisfactory attribute choice should be placed high on the list so that customers' time and effort in screening and evaluating the attribute choices can be minimised. To address this issue, the normalised discounted cumulative gain (nDCG), which is widely applied in information retrieval research (Järvelin and Kekäläinen 2002), is adopted as the evaluation metric. nDCG considers both the attribute choice's satisfaction degree and the choice's order on the presented list shown to the customer. It progressively reduces each presented attribute choice's degree of satisfaction with respect to the order to reflect a real customer decision-making process. In this way, if a customer finds a satisfactory attribute choice at the top of the presented list, the corresponding nDCG has a higher value.

In this study, nDCG is defined as $n\mathrm{DCG}_k = \frac{\mathrm{DCG}_k}{\mathrm{IDCG}_k}$, where $\mathrm{DCG}_k = \sum_{i=0}^{k} \frac{2^{rel_i}-1}{\log_2(i+1)}$. rel_i represents whether the i-th attribute choice satisfies the customer needs. $rel_i = 1$ indicates that the i-th attribute choice is satisfactory, and $rel_i = 0$ indicates otherwise. k is the number of presented attribute choices. IDCG_k is defined as $\mathrm{IDCG}_k = \sum_{i=0}^{|REL_k|} \frac{2^{rel_i}-1}{\log_2(i+1)}$, where REL_k is the list of attribute choices sorted based on their probabilities of satisfaction and calculated up to position k. nDCG can, thus, effectively measure the effectiveness of the trained MLP classifiers in mapping customer needs to satisfactory attribute choices.

5. Experiment results

5.1. Effectiveness analysis of the proposed approach

We examine the performance in terms of effectiveness of the three text embedding approaches. The results are provided in Table 1. Based on the results, we can see that in general, the three approaches to encoding the text do not differ much in terms of nDCG values. We observe that the concatenation of element-wise average and max pooling yields the best performance for most attributes (except processor). Because the concatenation approach considers every word's semantic information (through element-wise average) and simultaneously focuses on the salient words (through max pooling), it outperforms the other two. However, the dimension of the concatenated

Table 1. nDCG values of different attributes' configuration results based on the three text embedding approaches.

Performance metric	Text embedding approach	Processor	Monitor	RAM	Disk	Graphic processor
$nDCG_1$	Element-wise average	0.29	0.77	0.68	0.38	0.58
	Max pooling	0.30	0.77	0.69	**0.43**	0.63
	Concatenation	**0.32**	**0.77**	**0.69**	0.41	**0.63**
$nDCG_2$	Element-wise average	0.48	0.87	0.77	0.74	0.82
	Max pooling	0.46	**0.87**	0.81	0.71	**0.83**
	Concatenation	**0.50**	0.87	**0.82**	**0.76**	0.83
$nDCG_3$	Element-wise average	**0.55**	**0.93**	0.83	0.80	0.89
	Max pooling	0.53	**0.93**	0.86	**0.81**	**0.90**
	Concatenation	0.53	**0.93**	**0.87**	0.80	0.90
$nDCG_4$	Element-wise average	0.54	0.93	0.88	0.83	0.90
	Max pooling	0.54	**0.94**	0.91	0.82	**0.91**
	Concatenation	0.56	0.94	**0.92**	**0.84**	**0.91**
$nDCG_5$	Element-wise average	**0.58**	0.94	0.89	0.84	0.91
	Max pooling	0.54	**0.95**	**0.92**	0.83	0.91
	Concatenation	0.55	0.94	**0.92**	**0.85**	**0.92**

text embeddings is twice as large as that of the text embeddings of element-wise average or max pooling. The corresponding MLP is more complicated because the input vectors' dimension is higher. More training data are needed for the MLP with concatenated text embeddings as the input.

The problem is further compounded when the number of attribute choices is sizeable. This explains why the concatenated text embeddings cannot achieve the best performance for the processor attribute, which has the most choices. For this attribute, the element-wise average outperforms the other two, especially the max pooling approach. This indicates that in the review text, there are not many salient words relevant to the characteristics of processors.

In terms of the effectiveness of our approach, we notice that the MLP classifiers can achieve high nDCG values of configuration for all attributes except processor. When $k = 2$, the nDCG values for monitor, RAM, disk, and graphic processor are very high – almost all being above 0.75. This means that the top two recommendations can nearly retrieve the satisfactory attribute choice. When $k >= 3$, most of the nDCG values are larger than 0.8. This further shows the effectiveness of our approach.

Compared to the other attributes, nDCG value of the processor attribute is rather low. The main reason is that the number of choices for processors is higher than those for the other attributes. There are 10 processor choices (i.e. labels from a classification perspective). Given the same amount of training data, the classification task is more difficult for classes with more labels. In addition, customers' processor choices are highly uneven. For example, the most popular processor choice has more than 1700 associated reviews, whilst the least popular choice only has fewer than 100 associated reviews. This data imbalance complicates the mapping process, resulting in a low value of nDCG of the classification of processor.

In addition, we find that the performance for the disk attribute is relatively low when $k = 1$. Upon studying the raw data of the reviews, we find that very few reviews mentioned product characteristics directly related to disk. Most of the reviews mentioned product features related to CPU and monitor. For example, many customers mentioned the speed and weight of the laptop, which are significantly affected by CPU and monitor. (Note: CPU affects the speed, and monitor affects the weight.) Very few reviews mentioned disk. Consequently, the training dataset contains less information about laptops' disks. However, in customer needs texts, some respondents mentioned their requirements for a disk with the most frequently mentioned being SSD and large storage. In this regard, needs texts and review texts differ in relation to the laptop's disk information. Nevertheless, when $k > 1$, the nDCG of disk catches up fast, indicating that our approach performs well in the real configuration situations.

5.2. *Efficiency analysis of the proposed approach*

To examine the efficiency of the proposed approach, we compare its performance with that of four typical deep-learning approaches: convolutional neural network (CNN) (Kim 2014), long-short-term memory network (LSTM) (Greff et al. 2017), ELMo (Wang et al., 2020), and hierarchical attention network (HAN) (Wang, Zhao, and Wan 2021). LSTM and CNN are the basic unit in deep learning and have been widely used in NLP tasks thanks to their capabilities of capturing the long-range contextual dependency in the text and of extracting ordering information from the text. Being a complicated neural network, HAN contains two layers of bidirectional LSTM

Table 2. Performance comparison between deep learning-based approaches and the concatenation approach.

Performance metric	Classifier	Processor	Monitor	RAM	Disk	Graphic processor
nDCG$_1$	LSTM	**0.36**	**0.78**	0.60	0.36	**0.67**
	CNN	0.24	0.66	0.56	0.33	0.65
	HAN	0.33	0.72	0.62	**0.45**	0.62
	ELMo	0.31	0.72	0.64	0.40	0.58
	Concatenation	0.32	0.77	**0.69**	0.41	0.63
nDCG$_2$	LSTM	0.42	0.85	**0.83**	0.61	0.79
	CNN	0.42	**0.88**	0.81	0.54	0.79
	HAN	0.48	0.86	0.81	0.69	0.63
	ELMo	0.46	**0.88**	0.82	0.67	0.80
	Concatenation	**0.50**	0.87	0.82	**0.76**	**0.83**
nDCG$_3$	LSTM	**0.54**	0.90	0.85	0.72	0.85
	CNN	0.50	**0.93**	**0.88**	0.65	0.84
	HAN	**0.54**	0.91	**0.88**	0.79	0.77
	ELMo	0.53	0.92	**0.88**	0.75	0.85
	Concatenation	0.53	**0.93**	0.87	**0.80**	**0.90**
nDCG$_4$	LSTM	0.56	0.91	0.89	0.76	**0.92**
	CNN	0.55	**0.95**	0.90	0.72	0.87
	HAN	**0.60**	0.92	0.90	0.79	0.82
	ELMo	0.56	0.93	0.90	0.79	0.88
	Concatenation	0.54	0.94	**0.92**	**0.84**	0.91
nDCG$_5$	LSTM	0.60	0.93	0.90	0.79	**0.94**
	CNN	0.59	**0.95**	0.91	0.75	0.89
	HAN	**0.63**	0.92	0.91	0.80	0.82
	ELMo	0.60	0.94	0.91	0.81	0.89
	Concatenation	0.55	0.94	**0.92**	**0.85**	0.92

and uses an attention mechanism to quantify the importance of words in the text. ELMo leverages multiple layers of bidirectional LSTM to extract the contextual representation of words in the text. Because the concatenation approach achieves better results (see in the above subsection) and has the same computational complexity as the element-wise average and max pooling approaches,[1] we compare it with the four deep learning-based approaches. Table 2 provides the comparison results. We highlight the best performance in **bold**.

As shown in Table 2, the performance of the concatenation of embeddings is comparable to those of the four deep learning-based ones. In fact, our approach achieves the best $n\mathrm{DCG}_k$ values for two or three attributes for different values of k. Although deep neural networks are good at extracting high-level representations of domain information from the text, they are complicated and contain a vast number of parameters whose values are to be estimated. HAN and ELMo structures are composed of multiple layers of bidirectional LSTM and require even more parameters to be estimated in the training stage. However, our review text corpus is of medium size, which may not be sufficient for the four deep learning-based approaches. In addition, the review corpus contains a significant amount of short reviews. Half of the reviews contain no more than 65 words, and some even have a couple of words only, e.g. 'nice laptop!' Consequently, there may not be sufficient contextual information to extract. This explains why HAN and ELMo cannot achieve satisfactory results.

In addition to customer needs-related information, the review texts contain irrelevant product information and informal and creative language, such as emojis, idiomatic and ambiguous expressions. The concatenation-based approach can extract salient and important information from the text through the max-pooling operation and normal information through the element-wise averaging operation. From the perspective of signal processing, these two operations capture both low-frequency information (by element-wise averaging) and high-frequency information (by max-pooling) from the data. Thus, our embeddings-based approach shows comparable performance to the four deep learning-based approaches with the medium-size data set.

Additionally, the proposed approach has an advantage over LSTM and CNN in terms of computational complexity. In computer science, computational complexity is used to quantify the amount of resources, time in particular, required to run the algorithm. The complexities of CNN and LSTM have been proved to be $O(nLKd)$ and $O(Ld^2 + LKd)$, respectively, where K is the dimension of word embeddings; n is the filter width in CNN; d is the dimension of the final outputted sequence; and L is the number of words in the text corpus (Shen et al. 2018). Because HAN and ELMo are based on bidirectional LSTM, their complexity is higher than that of LSTM. The complexity of our text-embedding and MLP-based approach is only $O(LK)$, which is much lower than the four deep learning-based approaches. Furthermore, being non-deep learning-based, our approach requires a much smaller number of parameters to be estimated in the classifier-training phase. Thus, the embeddings with the MLP achieves a comparable performance but with much less complexity and fewer computational resources. Based on the analysis above, we conclude that the proposed text embeddings based approach is both effective (thanks to its good performance) and efficient (thanks to its low computational complexity).

5.3. *Analysis of effect of text length on performance*

Data properties (e.g. length) play critical roles in determining the performance of data driven approaches. In this study, the MLP classifiers are trained based on review texts crawled from Amazon's website, which vary in length. In NLP research, the models trained on longer text corpora tend to perform better (Giuseppe et al. 2017). In this section, we, thus, investigate the effect of text length on the performance of the MLP classifiers.

We use the median length of reviews (65 words) as the cut-off point to partition the whole review text corpus into long and short texts. For each attribute, two classifiers are trained using the long text and the short text, respectively. The trained classifiers are then used to map the needs to the attribute choices. We also use the concatenation approach to construct text embeddings. The results are shown in Figure 3. Because the values, $nDCG_2$, $nDCG_3$, and $nDCG_4$ in particular, have similar patterns, for illustrative simplicity, we provide the $nDCG_1$, $nDCG_3$, and $nDCG_5$ of the classifiers in Figure 3.

Based on the results, we find that when k is small (e.g. $k = 1$), long review texts produce better classifiers for all attributes. The advantages are marginal for

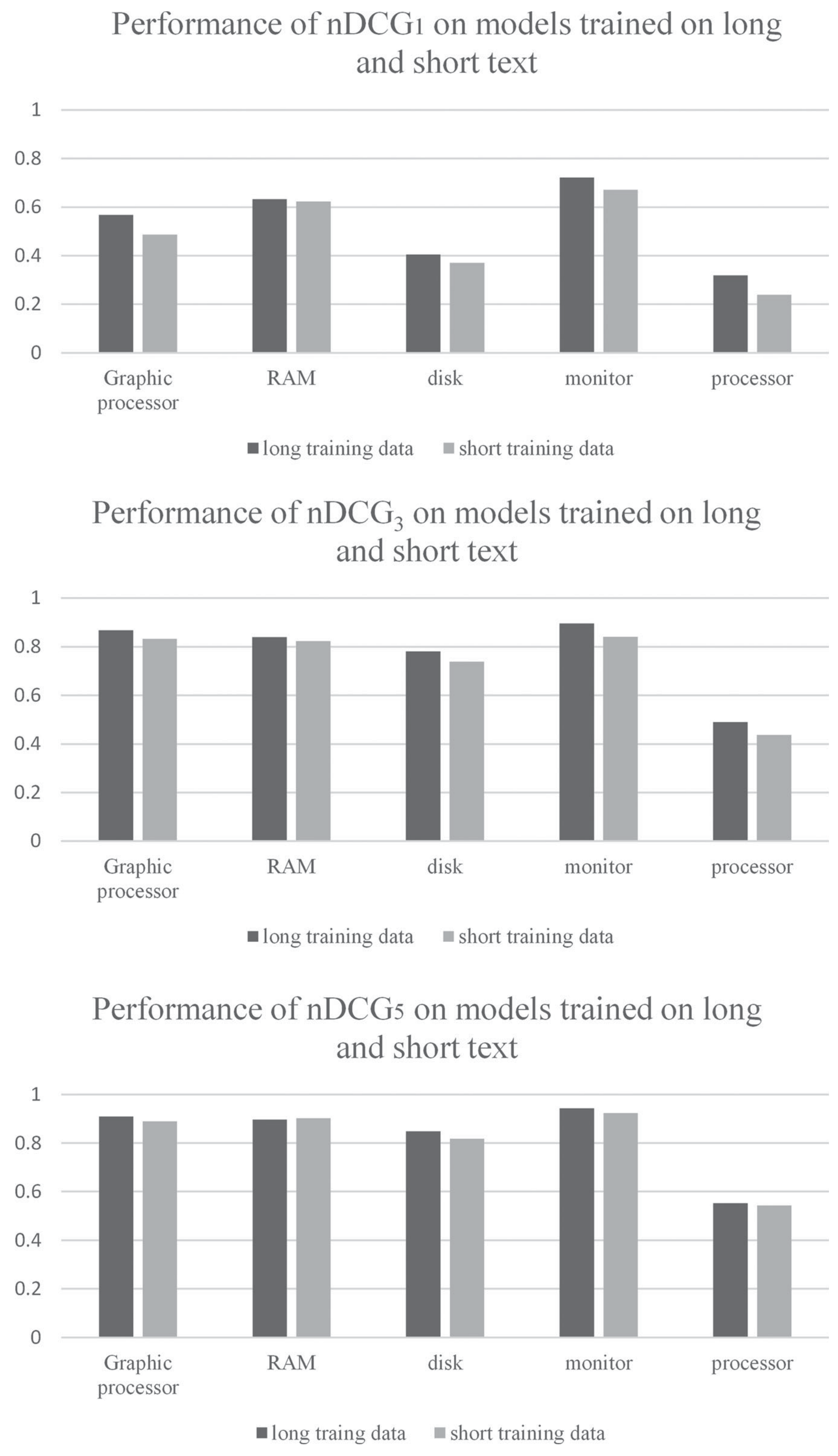


Figure 3. The performance of $nDCG_1$, $nDCG_3$ and $nDCG_5$ for classifiers trained from long review and short review texts.

RAM and monitor, but are significant for the remaining attributes. When k increases, the nDCG value increases for all attributes, whilst the difference between classifiers trained on long text and short text becomes smaller. When k is large (e.g. $k = 5$), while the nDCG values are larger than 0.8 for all attributes except processor, there is not much difference between the classifiers trained on long texts and short texts. Accordingly, we conclude that our proposed approach is robust with respect to the length of the training data when k is large.

In conclusion, with our proposed text embedding and MLP-based configuration approach, customers can freely express their true needs in natural languages, instead of going through the perplexing and stressful product attribute selection processes. Thus, our approach significantly contributes to user-friendly configuration development and serves to better bridge customers' needs and companies' offerings. Furthermore, our proposed approach automatically transforms customer needs expressed in laymen's terms into satisfactory product configurations. It, therefore, facilitates B2C operations by liberating designers from tediously linking customer needs, especially these in natural languages, to product configurations.

6. Conclusion

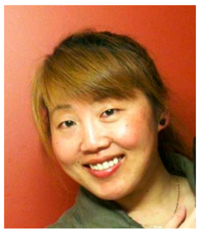
In this study, we defined needs-based configuration and proposed an implementation approach based on text embeddings and MLP. As demonstrated by the experiment results, our approach can achieve comparable performance with the complicated deep learning approaches. On the other hand, theoretically, the computational complexity of our approach is much lower than that of the deep learning approaches. Practically, our approach is lightweight and much efficient for industrial use. It sheds light on mining unstructured text for product customisation practice. In addition, it is robust concerning the length of review texts.

Nevertheless, this study has its limitations and can be improved in the following directions. We implicitly assumed that the product specification space is fixed and that no new product specifications will arise. However, customer needs may evolve, and new specifications may subsequently emerge. Therefore, it is both interesting and relevant to explore new ways of identifying new and/or unmet customer needs in product review texts in the future. Additionally, the performance of our approach may not be superior when the number of attribute choices is high (e.g. the processor attribute in our experiment). Thus, future efforts need to investigate new ways of handling classification cases in which the number of classes (i.e. attribute choices) is very high. Furthermore, we collected customer needs from university students who had, more or less, similar educational backgrounds and/or laptop knowledge. This suggests a limitation of our study. In this regard, in the future, we would like to expand the pool of respondents to general customers and verify our approach using empirical experiments. Finally, yet importantly, domain knowledge plays a critical role for customers to configure products. Although the semantics extracted from the review text could quantify relevant product information, they are not exhibited in an explicit format and can hardly guide customers in configuration processes. Thus, another potential avenue for future research is to explore the external domain knowledge to help customers find what they want with less burden in configuration processes.

Notes on contributors

Yue Wang received the B.S. degree in Electronics and M.E. in Information Science from Peking University, Beijing China, and the Ph.D. degree in Industrial Engineering from The Hong Kong University of Science and Technology. He is an Associate Professor in the Department of Supply Chain and Information Management, the Hang Seng University of Hong Kong. His research interests include machine learning, natural language processing, design and manufacturing informatics, healthcare informatics, and supply chain management. His Erdos number is 3.

Li Xiang received the B.S. degree in Internet of Things Engineering from Central South University, Chang Sha, China in 2018 and M.Sc. degree in Information Technology from The Hong Kong University of Science and Technology, Hong Kong in 2019. He is a Research Assistant in Department of Supply Chain and Information Management, The Hang Seng University of Hong Kong. His research interests focus on the data mining and natural language processing.



Dr. Zhang is currently a Professor of Operations Management in Department of Operations Management at IÉSEG School of Management (LEM-CNRS 9221), Lille-Paris, France. She obtained her BEng and Ph.D. degrees in Industrial Engineering in 1998 and 2007, respectively. Her research interests include mass customisation, product and production configuration, sustainable supply chain management, healthcare operations management, etc. On these areas, she has published a number of articles in international refereed journals, such as *Decision Support Systems, IIE Transactions, IEEE Transactions on Engineering Management, European Journal of Operations Research, International Journal of Production Economics*, etc.

Daniel Y. Mo received the Ph.D.degree in industrial engineering and logistics management from the Hong Kong University of Science and Technology (HKUST), Hong Kong, in 2013. He is currently an Associate Professor with the Department of Supply Chain and Information Management, The Hang Seng University of Hong Kong. He has authored or co-authored research papers in international journals such as the IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING, IEEE TRANSACTIONS ON ENGINEERING MANAGEMENT, etc. His current research interests include service parts operations management, logistics systems optimisation, design of intelligence systems, and Lean Six Sigma implementation. Dr.Mo was a recipient of the Kayamori Best Automation Paper Award Finalist for one of his research works. He is a professional of Certified Six Sigma Black Belt by the Hong Kong Society for Quality and a Member of the Institute of Industrial Engineers.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This work was supported by Hong Kong Research Grant Council Faculty Development Scheme [grant number UGC/FDS14/E06/18].

Note

1. The complexity of MLP is $O(LK)$, where L is the number of words in the text corpus and K is the dimension of word embeddings. The complexity of the concatenation approach is $O(2LK) = O(LK)$ as $O(f(n)) = O(f(cn))$, where c is a constant. Thus, the element-wise average, max pooling and concatenation have the same computational complexity.

ORCID

Yue Wang http://orcid.org/0000-0002-0185-6172
Daniel Mo http://orcid.org/0000-0001-8307-9065

References

Akinosho, T. D., L. O. Oyedele, M. Bilal, A. O. Ajayi, M. D. Delgado, O. O. Akinade, and A. A. Ahmed. 2020. “Deep Learning in the Construction Industry: A Review of Present Status and Future Innovations.” *Journal of Building Engineering* 32: 101827.

Barco, A. F., E. Vareilles, and C. I. Osorio. 2018. “Insights for Configuration in Natural Language.” *Proceedings of the 20th Configuration Workshop*, Graz, Austria, 15–18.

Bramham, J., and B. MacCarthy. 2003. “Matching Configurator Attributes to Business Strategy.” *Proceedings of the 2nd Interdisciplinary World Congress on Mass Customization and Personalization,* Munich, October 6–8.

Chen, S., Y. Wang, and M. M. Tseng. 2009. “Mass Customization as a Collaborative Engineering Effort.” *International Journal of Collaborative Engineering* 1 (2): 152–167.

Cooper, R. G., and S. J. Edgett. 2009. *Generating Breakthrough New Product Ideas: Feeding the Innovation Funnel.* Ancaster, Canada: Product Development Institute Inc.

cyLEDGE Configurator Database. 2021. Online Resource. date of access: May 25th, 2021. https://www.configurator-database.com/configurators#/.

De Bruyn, A., J. C. Liechty, E. Huizingh, and G. L. Lilien. 2008. “Offering Online Recommendations with Minimum Customer Input Through Conjoint-Based Decision Aids.” *Marketing Science* 27 (3): 443–460.

Forza, C., and F. Salvador. 2007. *Product Information Management for Mass Customization: Connecting Customer, Front-Office and Back-Office for Fast and Efficient Customization.* New York: Palgrave Macmillan.

Franke, D. W. 1998. “Configuration Research and Commercial Solutions.” *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* 12 (1): 295–300.

Fu, W., and T. Menzies. 2017. “Easy Over Hard: A Case Study on Deep Learning.” *Proceedings of 2017 11th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering*, Paderborn, Germany, 49–60.

Gao, B., and L. Pavel. 2017. “On the Properties of the Softmax Function with Application in Game Theory and Reinforcement Learning.” arXiv:1704.00805.

Garrett, J. J. 2010. *Elements of User Experience: The User-Centered Design for the Web and Beyond.* San Francisco, USA: Pearson Education.

Giuseppe, R., P. Bianca, V. Andrea, V. Erp Marieke, and E. C. B. Amparo. 2017. “Lessons Learnt from the Named Entity Recognition and Linking (NEEL) Challenge Series.” *Semantic Web Journal* 8 (5): 667–700.

Greff, K., R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber. 2017. “LSTM: A Search Space Odyssey.” *IEEE Transactions on Neural Networks and Learning Systems* 28 (10): 2222–2232.

Haug, A., and L. Hvam. 2007. “A Comparative Study of two Graphical Notations for the Development of Product Configuration Systems.” *International Journal of Industrial Engineering* 14 (2): 107–116.

Haug, A., L. Hvam, and N. H. Mortensen. 2010. “A Layout Technique for Class Diagrams to be Used in Product Configuration Projects.” *Computers in Industry* 61 (5): 409–418.

Haykin, S. 1998. *Neural Networks: A Comprehensive Foundation*. Upper Saddle River, NJ: Prentice Hall PTR.

Honigsberg, S., C. Kollwitz, and B. Dinter. 2019. “Designing A Reference Model for Digital Product Configurators.” *Proceedings of the 14th International Conference on Wirtschaftsinformatik,* Siegen, Germany, February 24–27.

Järvelin, K., and J. Kekäläinen. 2002. “Cumulated Gain-Based Evaluation of IR Techniques.” *ACM Transactions on Information Systems* 20 (4): 422–446.

Jiao, Y., and Y. Yang. 2019. “A Product Configuration Approach Based on Online Data.” *Journal of Intelligent Manufacturing* 30 (5): 2473–2487.

Kim, Y. 2014. “Convolutional Neural Networks for Sentence Classification.” *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1746–1751.

Kristjansdottir, K., S. Shafiee, L. Hvam, C. Forza, and N. H. Mortensen. 2018. “The Main Challenges for Manufacturing

Companies in Implementing and Utilizing Configurators." *Computers in Industry* 100 (1): 196–211.

McDermott, J. 1982. "R1: A Rule-Based Configurator of Computer Systems." *Artificial Intelligence* 19 (1): 39–88.

Nair, V., and G. Hinton. 2010. "Rectified Linear Units Improve Restricted Boltzmann Machines." *Proceedings of the 27th International Conference on Machine Learning, Haifa, Israel*, June 21–24, 807–814.

Ong, S. K., Q. Lin, and A. Y. C. Nee. 2006. "Web-based Configuration Design System for Product Customization." *International Journal of Production Research* 44 (2): 351–382.

Pennington, J., R. Socher, and C. Manning. 2014. "Glove: Global Vectors for Word Representation." *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, Doha, Qatar*, 1532–1543.

Piller, F., and D. Walcher. 2012. "The Customization 500 – An International Benchmark Study on Mass Customization and Personalization in Consumer E-Commerce." Research Report.

Pitiot, P., L. G. Monge, M. Aldanondo, E. Vareilles, and P. Gaborit. 2020. "Optimization of the Concurrent Product and Process Configuration: An Approach to Reduce Computation Time with an Experimental Evaluation." *International Journal of Production Research* 58 (2): 631–647.

Randall, T., C. Terwiesch, and K. T. Ulrich. 2007. "User Design of Customized Products." *Marketing Science* 26 (2): 268–280.

Rasmussen, J. B., L. Hvam, K. Kristjansdottir, and N. H. Mortensen. 2020. "Guidelines for Structuring Object-Oriented Product Configuration Models in Standard Configuration Software." *Journal of Universal Computer Science* 26 (3): 374–401.

Salvador, F., P. M. De Holan, and F. Piller. 2009. "Cracking the Code of Mass Customization." *MIT Sloan Management Review* 50 (3): 71–78.

Shen, D., G. Wang, W. Wang, M. R. Min, Q. Su, Y. Zhang, C. Li, R. Henao, and L. Carin. 2018. "Baseline Needs More Love: On Simple Word-Embedding-Based Models and Associated Pooling Mechanisms." *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, Melbourne, Australia*, 440–450.

Song, Z., and A. Kusiak. 2009. "Optimizing Product Configurations with a Data-Mining Approach." *International Journal of Production Research* 47 (7): 1733–1751.

Tang, D., Q. Wang, and I. Uilah. 2017. "Optimization of Product Configuration in Consideration of Customer Satisfaction and low Carbon." *International Journal of Production Research* 55 (12): 3349–3373.

Tiihonen, J., and A. Felfernig. 2010. "Towards Recommending Configurable Offerings." *International Journal of Mass Customization* 3 (4): 389–406.

Timoshenko, A., and J. R. Hauser. 2019. "Identifying Customer Needs from User Generated Content." *Marketing Science* 38 (1): 1–20.

Trentin, A., E. Perin, and C. Forza. 2012. "Product Configurator Impact on Product Quality." *International Journal of Production Economics* 135 (2): 850–859.

Trentin, A., E. Perin, and C. Forza. 2013. "Sales Configurator Capabilities to Avoid the Product Variety Paradox: Construct Development and Validation." *Computers in Industry* 64 (4): 436–447.

Triki, R., R. Mazo, and C. Salinesi. 2014. "Combining Configuration and Recommendation to Enable an Interactive Guidance of Product Line Configuration." *Les Systemes et Moteurs de Recommandation Hermes*. 141–160.

Tseng, H.-E., and C.-C. Chang. 2006. "Coordinating Product Configuration in the Order Fulfillment Processing: An Approach Based on the Binary Tree Algorithm." *International Journal of Computer Integrated Manufacturing* 19 (1): 716–726.

Tseng, H.-E., C.-C. Chang, and S.-H. Chang. 2005. "Applying Case-Based Reasoning for Product Configuration in Mass Customization Environments." *Expert Systems with Applications* 29 (4): 913–925.

Tseng, M. M., Y. Wang, and R. J. Jiao. 2018. *"Modular Design." In CIRP Encyclopedia of Production Engineering*. Berlin: Springer.

Turian, J., L. Ratinov, and Y. Bengio. 2010. "Word Representations: A Simple and General Method for Semisupervised Learning." *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, Uppsala, Sweden, 384–394.

Wang, Y., X. Li, and F. Tsung. 2020a. "Configuration-Based Smart Customization Service: A Multitask Learning Approach." *IEEE Transactions on Automation Science and Engineering* 17 (4): 2038–2047.

Wang, Y., D. Mo, and M. M. Tseng. 2018. "Mapping Customer Needs to Design Parameters in the Front end of Product Design by Applying Deep Learning." *CIRP Annals-Manufacturing Technology* 67 (1): 145–148.

Wang, Y., and M. M. Tseng. 2007. "An Approach to Improve the Efficiency of Configurators." *Proceedings of 2007 IEEE International Conference on Industrial Engineering and Engineering Management*, 1332–1336.

Wang, Y., J. Wu, R. Zhang, S. Shafiee, and C. Li. 2020b. "A 'User-Knowledge-Product' Co-Creation Cyberspace Model for Product Innovation." *Complexity* 7190169.

Wang, Y., W. Zhao, and W. X. Wan. 2021. "Needs-Based Product Configurator Design for Mass Customization Using Hierarchical Attention Network." *IEEE Transactions on Automation Science and Engineering* 18 (1): 195–204.

Xie, H., P. Henderson, and M. Kernahan. 2005. "Modeling and Solving Engineering Product Configuration Problems by Constraint Satisfaction." *International Journal of Production Research* 43 (20): 4455–4469.

Yang, D., and M. Dong. 2013. "Applying Constraint Satisfaction Approach to Solve Product Configuration Problems with Cardinality-Based Configuration Rules." *Journal of Intelligent Manufacturing* 24 (1): 99–111.

Yeh, J. Y., and T. H. Wu. 2005. "Solutions for Product Configuration Management: An Empirical Study." *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* 19 (1): 39–47.

Zhang, L. 2014. "Product Configuration: A Review of the State of the art and Future Research." *International Journal of Production Research* 52 (21): 6381–6398.

Zheng, P., X. Xu, S. Yu, and C. Liu. 2017. "Personalized Product Configuration Framework in an Adaptable Open Architecture Product Platform." *Journal of Manufacturing Systems* 43 (3): 422–435.

Zhou, C., Z. H. Lin, and C. T. Liu. 2008. "Customer-driven Product Configuration Optimization for Assemble-to-Order Manufacturing Enterprises." *The International Journal of Advanced Manufacturing Technology* 38 (1–2): 185–194.