Artificial Intelligence for Engineering Design, Analysis and Manufacturing

cambridge.org/aie

Design change prediction based on social media sentiment analysis

Edwin C.Y. Koh

Design and Artificial Intelligence Programme & Engineering Product Development Pillar, Singapore University of Technology and Design, Singapore, Singapore

Research Article

Cite this article: Koh ECY (2022). Design change prediction based on social media sentiment analysis. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* **36**, e24, 1–16. https://doi.org/10.1017/S0890060422000129

Received: 21 November 2021
Revised: 18 May 2022
Accepted: 19 June 2022

Key words:
BERT; dependency modeling; design change; natural language processing; social media

Author for correspondence:
Edwin C.Y. Koh,
E-mail: edwin_koh@sutd.edu.sg

Abstract

The use of artificial intelligence (AI) techniques to uncover customer sentiment is not uncommon. However, the integration of sentiment analysis with research in design change prediction remains an untapped potential. This paper presents a method that uses social media sentiment analysis to identify opportunities for design change and the set of product components affected by the change. The method builds on natural language processing to determine change candidates from textual data and uses dependency modeling to reveal direct and indirect change propagation paths arising from the change candidates. The method was applied in a case example where 3665 *YouTube* comments on a diesel engine were analyzed. Based on the results, two engine components were recommended for design change with six others predicted as likely to be affected through change propagation. The findings suggest that the method can be used to aid decision quality in product planning through a better understanding of the change impact associated with the opportunities identified.

Introduction

Many products are developed through an incremental design change. However, design changes initially perceived to be simple can sometimes propagate and result in undesirable outcomes (Eckert *et al.*, 2004; Giffin *et al.*, 2009; Shankar *et al.*, 2012; Fernandes *et al.*, 2015). Design change is also referred to as "engineering change" in an engineering context and is described as "an alteration made to parts, drawings or software that has already been released during the product design process" (Jarratt *et al.*, 2011). The propagation of change is described as "the process by which an engineering change to parts of a product results in one or more additional engineering changes to other parts of the product, when those changes would not otherwise have been required" (Koh *et al.*, 2012). In an empirical study documented by Clarkson *et al.* (2004), chief engineers at an aerospace company consistently pointed out that design changes can propagate through indirect linkages and affect product/system components that are as far as four steps (i.e., linkages) away from the initial change. This assertion is supported in another empirical case study reported by Duran-Novoa *et al.* (2018) which traced the propagation of changes in the design of an electrical motorcycle and found that change propagation accounted for 20% of all changes made, with 67% of change propagation reaching components that were one step away from the initial change, 25% reaching components that were two steps away, and 8% reaching components that were between three to five steps away. Failure to anticipate design changes can, therefore, lead to delays and financial losses (BBC News, 2009).

Approaches to predict design change and its propagation include analyzing past changes (Suh *et al.*, 2007; Yin *et al.*, 2017) and uncovering existing change dependencies (Clarkson *et al.*, 2004; Xie and Ma, 2016; Lee and Hong, 2017; Chen *et al.*, 2020). The metric used to examine design change include engineering tolerance (Hamraz *et al.*, 2013), workload (Tang *et al.*, 2016), staff affected (Koh *et al.*, 2015), lead time (Ullah *et al.*, 2018), manufacturing (Siddharth and Sarkar, 2017), lifecycle performance (Cardin *et al.*, 2013), and profit margin (Yassine and Khoury, 2021). Endogenous change dependencies between product components can be extracted from Product Data Management systems and engineering databases to support design change analysis (Jarratt *et al.*, 2011). Exogenous change data such as customer needs are traditionally collected manually through user interviews and focus groups, which can be costly and inefficient. There is a growing effort to automate customer needs analysis from online data through artificial intelligence (AI) (Tucker and Kim, 2009; Tuarob and Tucker, 2015*b*; Sha *et al.*, 2017; Tang *et al.*, 2019; Zhou *et al.*, 2020). However, little work has been done to explore how such AI approaches can be integrated with design change prediction.

This paper reports on the development of a method that uses AI approaches to identify opportunities for design change and the set of product components affected by the change. The goal of the method is to aid decision quality in product planning by offering new ideas and insights from a customer sentiment perspective through social media data where a large volume of textual data is available and manual reading is impracticable. The work is positioned in the context of incremental design where an existing product architecture is present and early prediction of design change propagation between product components can be used to better understand the change impact associated with the new ideas. The output of this work can be used to identify key personnel responsible for the components affected by the new ideas and the resources required to support the change. In doing so, the change impact on the product architecture can be established and used to complement existing processes (e.g., cost assessment) to aid decision quality in product planning. The study connects two fields of research that are usually considered in isolation (i) the use of natural language processing in customer sentiment analysis and (ii) the use of dependency modeling in change prediction. The main contributions of this work are summarized as follows:

(1) A set of algorithms for uncovering product components that might be directly or indirectly affected by change candidates identified from social media.
(2) Analysis results from a case example that explore the feasibility and limitations of the proposed method.
(3) The dataset used in the case example (*YouTube* comments, https://github.com/NLPT/P1data).

Related work

This section is divided into two parts. The first part discusses the use of AI in customer needs analysis and focuses on the context of extracting product information from textual data in social media. The second part discusses techniques in design change prediction and establishes possible junctures to connect the two fields of research.

Analyzing customer needs from online data

Customer data come in many forms. Some data are captured in numerical and categorical format and are suitable for clustering and correlation studies. For instance, Wang *et al.* (2018) present a data-driven approach that uses the correlation between customer preferences and product attributes to predict product co-consideration for a given customer demographic. Long *et al.* (2019) introduce a framework that models the customer decision-making process during purchase and links the considerations made to design requirements. Xie *et al.* (2020) describe a dynamic network-based approach that analyses the evolution of product competitions through customer data captured across multiple years. Customer reviews can come in the form of textual data documented on professional product review websites (e.g., PC Magazine), e-commerce platforms (e.g., Amazon reviews), and social media (e.g., Twitter). Reviews on professional websites are usually much longer and are a source of valuable customer insights. Indeed, Liu *et al.* (2013) argue that a long review can better support latent customer needs elicitation as having more textual data means having more descriptions on the product attributes, user preferences, and use cases. Nevertheless, long reviews are harder to come by compared with short reviews captured on e-commerce platforms and social media.

Despite having less data, the relevance of short online reviews for customer needs analysis is advocated by Tuarob and Tucker (2015*b*) in a case study which shows that the incorporation of certain product features can shift the sentiment in social media comments found on *Facebook*. The validity of using short customer reviews is also demonstrated in a study reported by Archak *et al.* (2011) which successfully predicted product demand by using a set of algorithms to analyze sales data and customer review data from *Amazon.com*. In fact, based on a combined approach of word embedding (word2vec) and X-means clustering, Suryadi and Kim (2018) found that many product features that are mentioned in *Amazon.com* review titles are closely related to sales rank, suggesting that even textual data as short as titles can be used to better understand customer needs.

Approaches to identify opportunities from customer reviews include using keywords to search for information related to predefined product features (Lim and Tucker, 2016) and sorting all words based on the number of times they occur (i.e., term frequency) (Liu *et al.*, 2013). Affective words (e.g., "beautiful") are sometimes emphasized to identify affective design properties that can provoke emotions and enhance customer satisfaction (Chang and Lee, 2018; Wang *et al.*, 2019). Topic modeling techniques can also be used to extract customer concerns (Zhan *et al.*, 2009; Tang *et al.*, 2019). For instance, Zhou *et al.* (2020) present a machine learning approach that uses latent Dirichlet allocation (LDA) (Blei *et al.*, 2003) to extract topic areas from online review data, allowing users to manually label and scrutinize comments on each topic (e.g., "for kids", "music-related").

The sentiment of a comment or a topic (i.e., all comments in the topic) can be examined through supervised machine learning techniques with labeled review data (Tuarob and Tucker, 2015*b*). Unlabeled review data can also be analyzed through unsupervised lexicon-based techniques (Zhou *et al.*, 2020), such as Vader (Hutto and Gilbert, 2014). Methods to fine-tune sentiment analysis include detecting sarcasm in comments (Tuarob *et al.*, 2018) and moderating opinion scores through customer rating history and tendency (Lim *et al.*, 2017). The scope of analysis can be adjusted by differentiating lead users from general users (Tuarob and Tucker, 2015*a*) and focusing on latent customer needs in extraordinary use cases (Zhou *et al.*, 2015). Advancement in language models can also unlock new opportunities in customer sentiment analysis. For instance, the development of Transformer-based (Vaswani *et al.*, 2017) language representation model such as the Bidirectional Encoder Representations from Transformers (BERT) (Devlin *et al.*, 2018) along with its retrained versions [e.g., RoBERTa (Liu *et al.*, 2019)] has facilitated the use of pre-trained models that were trained on large dataset to be finetuned or used directly in similar tasks.

Analyzing design change propagation

Customer preferences can change over time. The change in preferences can be captured through customer reviews from different time periods and used to anticipate next-generation product features (Tucker and Kim, 2011; Jin *et al.*, 2016; Jiang *et al.*, 2017; Sun *et al.*, 2020). While product components directly affected by the new features can be determined based on the design options developed later, components that are indirectly affected through change propagation are usually identified through dependency modeling (Koh, 2017). For instance, Kang and Tucker (2016) present an AI approach to quantify dependencies between product components by mining textual technical descriptions from a textbook. The approach was demonstrated in a case study involving an automotive climate control system where a model that highlights direct dependencies between product components was produced with 94% accuracy compared to one that was manually created. Song *et al.* (2019) discuss a method that uses data of prior design versions to identify a network of functions between components. The method was demonstrated using patent data on spherical rolling robots to identify interdependent platform components. Sarica *et al.* (2020) describe a technology semantic network (TechNet) that uses natural language processing techniques to extract and vectorize terms from massive patent texts to establish their semantic relationships. Although it was not explicitly discussed, the semantic relationships established in TechNet can be extended to estimate dependencies between product components. The methods discussed above can be used to predict direct change propagation between components (i.e., changing A results in changes in B). Nevertheless, indirect change propagation is not considered in these methods (i.e., changing A results in changes in B, and subsequently changes in C).

The identification of indirect change dependencies through machine learning was explored by Wickel and Lindemann (2015) where past change data were examined using association rules analysis (Agrawal *et al.*, 1993) to uncover component sets affected by frequently co-occurring changes. Approaches based on fault-tree analysis are available as well to reveal indirect component dependencies through functional failure (Roth *et al.*, 2015). In addition, indirect change dependencies between components can be derived through the modeling of people and documents pertinent to each product component (Lindemann *et al.*, 2009), through shared process dependencies such as tasks and events (Kreimeyer and Lindemann, 2011; Kasperek *et al.*, 2016), and through shared association in design specifications and requirements (Kreimeyer, 2016).

An established method for modeling indirect change dependencies is the change prediction method (CPM) introduced by Clarkson *et al.* (2004). CPM calculates the combined likelihood of change propagation between any two product components by considering all possible change propagation paths between them using logic trees, where the likelihood of change propagation for a path is computed by multiplying the direct change propagation likelihood between consecutive components along the path. Over the years, extensions to CPM include the handling of concurrent design changes initiated on multiple components (Ahmad *et al.*, 2013) and the addition of a reachability factor to account for finite change propagation resources (Koh *et al.*, 2013). Industry evaluations were also carried out and documented (Hamraz and Clarkson, 2015). Other methods with a similar approach are put forward as well to address design changes carried out at the module-level (a module consists of multiple components) (Yin *et al.*, 2021) and at the feature-level (a component consists of multiple features) (Chen *et al.*, 2020). However, it remains unclear how efforts in design change prediction can be integrated to provide insights during the identification of design opportunities from social media.

Method

This section describes a method that uses social media sentiment analysis to identify opportunities for design change and the set of product components affected by the change. The method consists of three stages as shown in Figure 1. In Stage 1, social media comments related to a given product are captured for natural language processing to identify frequently used words and the sentiment of the comments containing those words. In Stage 2, frequently used words with comments that are negative in sentiment are checked against the component list of the given product to identify suitable components for design improvement. These components are referred to as "change candidates" in this paper. Lastly, in Stage 3, a product model that describes the change dependencies between its components is created and aligned with the change candidates to predict design change.

Collecting and processing social media comments

The goal of this stage is to gather online comments related to a given product of interest and determine areas that are frequently discussed along with the sentiment of the discussions. There are a number of ways to gather online comments, including the use of Application Programming Interfaces (API) and web scraping tools. The one presented in this paper uses the *YouTube* API to search for relevant videos and collect comments and replies associated with the videos found (for simplicity, replies are also referred to as comments in the remainder of this paper). The comments found are subsequently put through preprocessing to convert all text into lowercase, remove hyperlinks, remove non-alphanumeric characters except for $ and %, remove redundant spaces between words, reduce all words into their root form through stemming (i.e., "class" and "classes" stemmed as "class"), and remove stop words (i.e., words that have little meaningful semantics). In this work, the list of stop words builds on the Natural Language Toolkit (NLTK, https://www.nltk.org/) open-source Python library (e.g., "he", "she", "this", "that") and is extended with additional words specific to *YouTube* (e.g., "subscribe", "channel", "video", "thanks"). The work also uses Google Translate through the deep-translator Python library (https://deep-translator.readthedocs.io) to translate all comments into English.

After the comments are collected and preprocessed, the comments for each video are combined into a corpus and the top words (i.e., most frequently used words) are identified by tokenizing the corpus into separate words for occurrence counting. Each top word is a unigram and the number of top words can be limited by observing a threshold frequency (e.g., more than 30 times). Subsequently, by going through one top word at a time for each video, comments with the top word are combined into a corpus to uncover linked words that are frequently mentioned along with the top word. Each linked word can be a single word (i.e., unigram) or a phrase consisting of a set of consecutive words (i.e., *n*-gram, *n* consecutive words). Similar to top words, the number of linked words can also be limited through a threshold frequency (e.g., more than 25% of the top word frequency). A discussion on the choice of threshold frequency is provided in the section "Setting threshold frequency for top words and linked words". In this work, the "everygrams" function from the NLTK Python library (https://www.nltk.org/) is used together with a counter to identify the top words and their linked words for each video.

Once the top words and linked words are identified for each video, sentiment analysis is conducted for each unprocessed comment containing either a top word or a pairing of both a top word and its linked word through a RoBERTa-base language model that was trained on 58 million tweets and finetuned for sentiment analysis with the TweetEval benchmark (Barbieri *et al.*, 2020) (model available at https://huggingface.co/cardiffnlp/twitter-

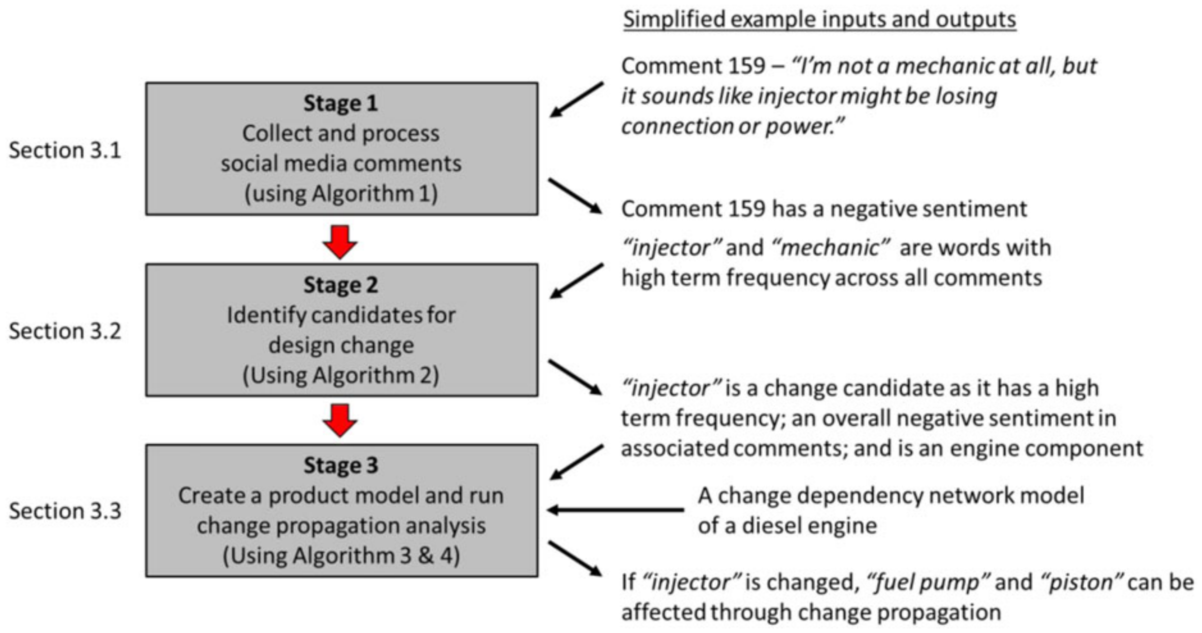

Simplified example inputs and outputs
Section 3.1
Stage 1
Collect and process social media comments (using Algorithm 1)
Comment 159 – "I'm not a mechanic at all, but it sounds like injector might be losing connection or power."
Comment 159 has a negative sentiment
Section 3.2
Stage 2
Identify candidates for design change (Using Algorithm 2)
"injector" and "mechanic" are words with high term frequency across all comments
"injector" is a change candidate as it has a high term frequency; an overall negative sentiment in associated comments; and is an engine component
Section 3.3
Stage 3
Create a product model and run change propagation analysis (Using Algorithm 3 & 4)
A change dependency network model of a diesel engine
If "injector" is changed, "fuel pump" and "piston" can be affected through change propagation

Fig. 1. Overview of the method using a diesel engine as an example.

roberta-base-sentiment). Each comment is classified as "Negative", "Neutral", or "Positive". Subsequently, based on the computation of net sentiment described by Pratama *et al.* (2019), a top word is classified as having a net negative sentiment if it is associated with more comments of "Negative" sentiment than "Positive" ones. "Neutral" comments are ignored as they do not affect the polarity of net sentiment (Pratama *et al.*, 2019). The net sentiment for each pairing of top word and its linked word is determined using the same classification scheme. The pseudo-code for the processing of frequently used words (i.e., term frequency) and sentiment analysis is shown in Algorithm 1.

```
Algorithm 1: The term frequency analysis and sentiment analysis algorithm
for a collection of online comments

Input: $V$: Set of videos found. Each $v \in V$ is an individual video. $C$:
Set of comments found in a video. Each $c \in C$ is an individual
comment string in a tuple of (preprocessed version $c'$,
unprocessed version $c''$). $D$: Set of $C$ according to $V$.
Output: $T$: Set of top words in a video. Each $t \in T$ is a top word. $U$:
Set of linked words of a top word in a video. Each $u \in U$ is a linked
word of a top word. $S$: Set of sentiment classification for either a $t$
or a $t$ and $u$ pair. $A$: Set of net sentiment classification for all $t$ or $t$
and $u$ pair in $V$.

$T = \emptyset$
$U = \emptyset$
$A = \emptyset$
for $C$ in $D$:
    copy all $c'$ and join them into a corpus_of_v
    tokenise and count each word in corpus_of_v
    add words with frequency above threshold to $T$
    for $t \in T$:
        for $c \in C$:
            if $t$ in tokenised $c'$ is true:
                check sentiment of $c''$ and add classification to $S$
                copy $c'$ and join it to a corpus_of_t
        end
        add "NEG" to $A$ if number of "Negative" > "Positive" in $S$ else add
        "not NEG"
        clear $S$
        tokenise and count each word and n_gram in corpus_of_t
        add words and phrases with frequency above threshold to $U$,
        excluding $t$
        clear corpus_of_t
        for $u \in U$:
            for $c \in C$:
                if $t$ and $u$ in tokenised $c'$ is true:
                    check sentiment of $c''$ and add classification to $S$
                end
                add "NEG" to $A$ if number of "Negative" > "Positive" in $S$ else
                add "not NEG"
                clear $S$
                end
            end
        end
        return $T$, $U$, and $A$
```

Identifying candidates for design change

Although the top words and their linked words along with the sentiment in which they are mentioned can be extracted as described in the previous stage, it is unlikely that all top words and linked words are directly linked to product components (e.g., top words such as “issue”, “good”, and “really”). Hence, in this stage, the top words extracted are placed in a word collection to be checked against a product component list. In anticipation of components with names that have more than one word (e.g., “wiring harness”), each pairing of top word and linked word are also combined and permutated, with each permutation added to the word collection. Product components that appear in the word collection are classified as likely to be changed if the net sentiment classification of the matching top word (or top word and linked word permutation) is negative. The assumption is that components that are frequently mentioned and discussed in a negative sentiment are suitable candidates for improvement through design change. These components are referred to as change candidates in this paper.

While it is possible to start off the mining of product information using pre-defined keywords (Lim and Tucker, 2016), where the keywords are component names of the product of interest, it is necessary to go through the process presented thus far as a product can be described at different levels of granularity (Maier *et al.*, 2017) making it hard to anticipate the granularity used in social media comments. Given that the “correct” granularity level is fixed at the time the comments were made, an approach to get around the granularity issue is to use component lists of different granularity levels to be checked against. The component lists can be elicited from domain experts, extracted directly from company Product Data Management systems (Koh *et al.*, 2015), or created using textual technical descriptions in textbooks (Kang and Tucker, 2016) and technology semantic networks such as TechNet (Sarica *et al.*, 2020). However, in this work, the component lists are assumed to be directly available through company Product Data Management systems. The pseudo-code for creating a word collection and checking against component lists is shown in Algorithm 2.

Algorithm 2: The word collection and component lists checking algorithm

Input: T: Set of top words in a video. Each $t \in T$ is a top word. U: Set of linked words of a top word in a video. Each $u \in U$ is a linked word of a top word. V: Set of videos found. Each $v \in V$ is an individual video. A: Set of net sentiment classification for all t or t and u pair in V. Each $a \in A$ is an individual net sentiment classification. B: Set of component lists. Each $b \in B$ is a component list of a different granularity.
Output: W: Set of top words and permutations of top words and their linked words from V (i.e., word collection). Each $w \in W$ is an entry in the word collection. X: Set of change candidates.

```
1   $W = \emptyset$
2   $X = \emptyset$
3   for $v \in V$:
4       for $t \in T$:
5           add $t$ to $W$
6           for $u \in U$:
7               join $u$ after $t$ with a space in between and add to $W$
8               join $t$ after $u$ with a space in between and add to $W$
9           end
10      end
11  end
12  for $b \in B$:
13      for $w \in W$:
14          if $w$ in $b$ is true and corresponding $a$ == “NEG”:
15              add $w$ to $X$ if $w$ not in $X$
16      end
17  end
18  return $W$ and $X$
```

Creating a product model and running change propagation analysis

This stage consists of two steps. The first step is to create a product model that considers both direct and indirect change propagation. The second step aligns the product model created with the change candidates identified in the section “Identifying candidates for design change” for design change prediction.

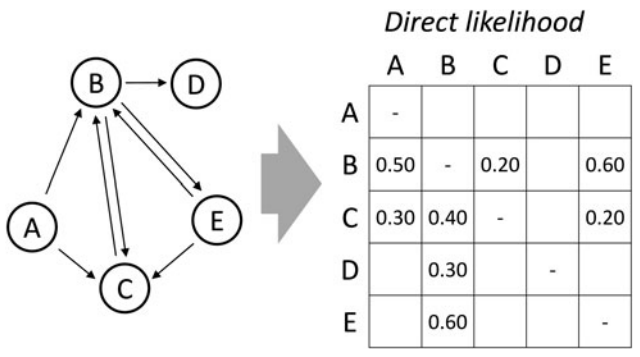
Creating a product model that considers direct and indirect change propagation

While there is no easy way to determine what the finest granularity level should be for a given product (Maier *et al.*, 2017), the aim here is to create a product model that is as fine-grained as possible so that it can act as a baseline and is flexible to be adjusted to a coarser granularity when required. In this work, the product model is presented in the form of a design structure matrix (DSM) (Eppinger and Browning, 2012) where the components are listed as column and row headers (see Fig. 2). The off-diagonal cells of the matrix indicate the direct change propagation likelihood between components and follow the convention where the column components affect the row components. The cell entries can be elicited from domain experts and range between 0 (will not propagate) and 1 (will propagate) (Clarkson *et al.*, 2004). Other approaches, such as the use of past change records (Suh *et al.*, 2007) and the use of textual mining techniques on domain texts (Kang and Tucker, 2016; Sarica *et al.*, 2020), can also be adapted here to populate the cells. Cells with a 0 value are left blank for simplicity. For instance, with reference to the example in Figure 2, there is a 0.5 likelihood that changing Component A will result in direct change propagation to Component B (i.e., Column A, Row B) while changing Component B will have 0 likelihood of affecting Component A (i.e., Column B, Row A, shown as a blank entry).

Indirect change propagation between product components is accounted for in this work through the CPM algorithm presented by Clarkson *et al.* (2004). The CPM considers change propagation along three dimensions – likelihood, impact, and risk. However, instead of reproducing the CPM in its entirety, this paper focuses on the likelihood of change propagation as described in Eqs (1)–(3). Note that the reachability of change propagation is also included in this paper to account for finite change propagation resources (Koh *et al.*, 2013).

$$L_{k,j} = 1 - \prod_{z \in Z} [1 - (l_z \times \alpha_z)], \tag{1}$$

$$l_z = (l_{k,k-1} \times l_{k-1,k-2} \times \cdots \times l_{j+1,j}), \tag{2}$$



	A	B	C	D	E
A	-				
B	0.50	-	0.20		0.60
C	0.30	0.40	-		0.20
D		0.30		-	
E		0.60			-

Fig. 2. An example of a product model in a DSM (DSM shown on right).

$$\alpha_z = (\alpha_{k,k-1} \times \alpha_{k-1,k-2} \times \cdots \times \alpha_{j+1,j}), \tag{3}$$

where $L_{k,j}$ represents the combined (direct and indirect) change propagation likelihood from component j to k; l_z and α_z represent the change propagation likelihood and reachability for a particular path z, respectively; Z represents the entire set of change propagation paths from component j to k.

As pointed out by Clarkson *et al.* (2004), Giffin *et al.* (2009), and Duran-Novoa et al. (2018), design changes rarely propagate beyond four steps. Hence, in this work, the change propagation reachability between successive components is set as 1 for the first propagation step (i.e., $\alpha_{j+1,j}$) and 0.3 for the second step onwards (e.g., $\alpha_{k,k-1}$) to bring the likelihood of five-step change propagation down to less than 0.01 (i.e., $1 \times 0.3^4 < 0.01$). In doing so, the likelihood of indirect change propagation paths is reduced and paths with more steps have less influence. Note that the first propagation step was assigned a reachability of 1 to ensure that the combined change likelihood computed does not fall below the direct change likelihood used in the computation. Equation (3) can therefore be rewritten as Eq. (4), where α_F refers to the reachability factor ($\alpha_F = 0.3$ in this work) and n refers to the number of change propagation steps in a particular path z.

$$\alpha_z = \alpha_F^{n-1}. \tag{4}$$

Through the use of Eqs (1)–(4), the product model shown in Figure 2 can be updated to consider indirect change propagation as shown in Figure 3. For instance, Component E in Figure 3 is not directly linked to Component D and the likelihood of direct change propagation is 0 (left DSM, Column E, Row D, shown as a blank entry). By considering indirect change propagation via Component B, the combined (direct and indirect) change propagation likelihood is updated to 0.05 (i.e., right DSM, Column E, Row D, $1-[1-((0.60 \times 0.30) \times 0.3^{2-1})] = 0.05$).

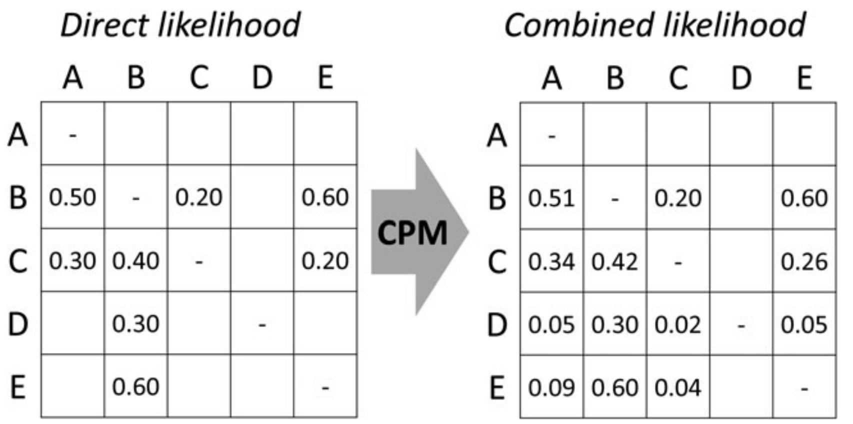
Although a standalone version of the CPM is available (Keller, 2007), a Python implementation of the CPM is used in this work to align the data structure. In the first instance, the product model with direct change propagation likelihood is stored as a comma-separated values (.csv) file for easy processing. The computation of combined change propagation likelihood is later carried out according to Eqs (1)–(4) using the NetworkX Python library (https://networkx.org/). Essentially, the NetworkX "from_numpy_array" function is used to convert the product model (.csv file) into a digraph where each direct change propagation likelihood value is mapped as the "weight" of the corresponding directed edge. Subsequently, by going through each row and column in the product model, all paths between any two components are identified through the "all_simple_paths" function. Next, by using Eqs (1)–(4) to process the "weight" of the directed edges connecting the paths found, the combined change propagation likelihood between each component pair is computed and the product model is updated to be used in the next stage. The pseudo-code for the product model is shown in Algorithm 3.

Algorithm 3: The combined (direct and indirect) change propagation algorithm for a given product model with direct change dependencies

```
   Input: $M$: A product model with direct change propagation
   likelihood between components.
   Output: $N$: A product model with combined change propagation
   likelihood between components.
1  open file containing $M$ (.csv)
2  convert a copy of $M$ into digraph
3  for column in $M$:
4      for row in $M$:
5          if row !=column:
6              find all paths between the row component and the
               column component
7              compute combined likelihood using Eqs. (1)–(4) based on
               paths found
8              insert combined likelihood to the corresponding row and
               column in $N$
9      end
10 end
11 return $N$
```

Aligning the product model and change candidates to predict design change

While the product model created in the previous step is intentionally as fine-grained as possible, the change candidates identified in the section "Identifying candidates for design change" can be of different granularity levels. Adjustment to the product model is therefore necessary if its components do not match the change candidates. An example where an adjustment is necessary is illustrated in Figure 4, which shows a product described by two sets of components with different granularity levels – a coarse-grained set with two components (i.e., Component δ and θ) and a fine-grained set with five components (i.e., Components A to E). Component δ is made up of Component A and B while Component θ is made up of Component C to E. Such mapping between components of different granularity is assumed to be available in this work (e.g., through Product Data Management systems) and accessible in the format shown in Figure 4, where the entity in the first row of each column is a coarse-grained component and the entities in the subsequent rows of each column are the constituent fine-grained components. In fact, the component lists with different granularity (P) described earlier in Algorithm 2 are basically derived from the first row of each mapping. For ease of computation, the fine-grained components in each mapping are standardized by using the same set of components in the product model described in the section "Creating a product model that considers direct and indirect change propagation"


Direct likelihood
Combined likelihood
CPM

Direct likelihood	A	B	C	D	E
A	-				
B	0.50	-	0.20		0.60
C	0.30	0.40	-		0.20
D		0.30		-	
E		0.60			-

Combined likelihood	A	B	C	D	E
A	-				
B	0.51	-	0.20		0.60
C	0.34	0.42	-		0.26
D	0.05	0.30	0.02	-	0.05
E	0.09	0.60	0.04		-

Fig. 3. Updating the product model using the change prediction method (CPM).

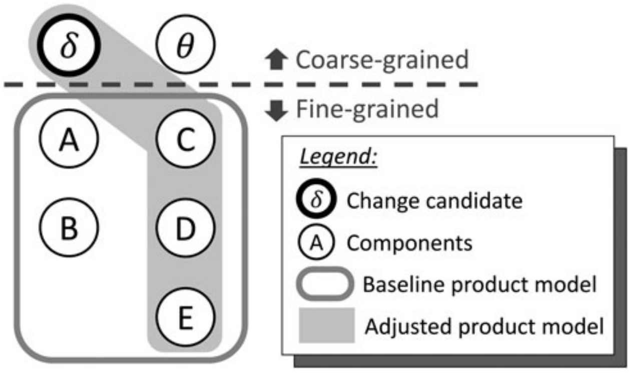


Fig. 4. Adjusting the model granularity to align with change candidates.

and each mapping is stored as a comma-separated values (.csv) file for processing.

Given that the change candidate is Component δ (a coarse-grained component) in the example shown in Figure 4, the product model which consists of Component A to E (i.e., fine-grained components) needs to be adjusted in order to predict change propagation arising from the change candidate. The adjustment is carried out in this work based on a component merging technique (Ahmad *et al.*, 2013) where the change propagation likelihood of a coarse-grained component is estimated as the root sum of squares of its fine-grained components. By using this merging technique, the change propagation likelihood of the coarse-grained component will be no lower than the individual likelihood values of its fine-grained counterparts and will fall within the 0 to 1 value range. In this paper, the technique is expressed as an equation as follows:

$$L_{k,x} = \min\left(\sqrt{\sum_{y \in Y} L_{k,y}^2}\ ,\ 1\right), \tag{5}$$

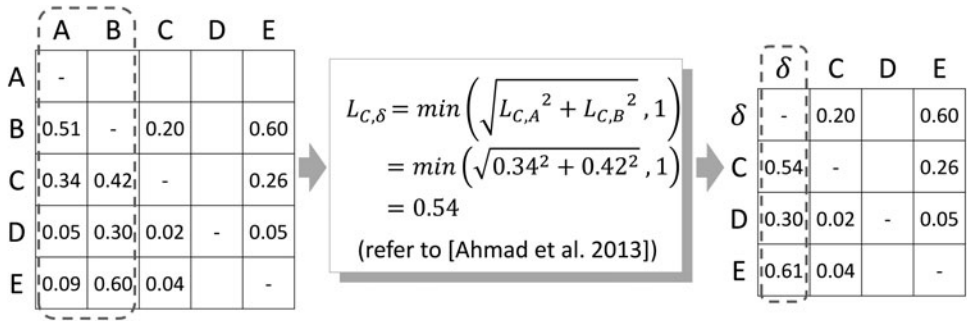
where $L_{k,x}$ represents the combined (direct and indirect) change propagation likelihood from a coarse-grained component x to component k; $L_{k,y}$ represents the combined change propagation likelihood from a particular fine-grained component y to component k; Y represents the entire set of fine-grained components that make up component x.

An example calculation of an adjustment is shown in Figure 5 where the change propagation likelihood from Component δ to Component C ($L_{C,\delta}$) is estimated as the root sum of squares of its fine-grained counterparts – from Component A to C ($L_{C,A}$) and from Component B to C ($L_{C,B}$). By repeating the technique described for all affected cells, the entire column of Component δ (i.e., the change candidate) is adjusted as shown. For instance, changing Component δ can result in change propagation to Components C, D, and E through direct and indirect paths with a likelihood of 0.54, 0.30, and 0.61, respectively. For completeness, the technique is also applied to the row of Component δ to keep the adjusted product model in a square matrix form. With the assumption that there are limited resources to prevent or prepare for the change propagation predicted, a minimum likelihood threshold can subsequently be introduced to remove components less likely to be affected by change propagation. For example, the results will be narrowed down to Components C and E if a minimum likelihood threshold of 0.50 is applied. The pseudo-code for aligning the product model with the change candidate is shown in Algorithm 4.

Algorithm 4: The product model adjustment algorithm for a set of change candidates

Input: N: A product model with combined change propagation likelihood between components. X: Set of change candidates. Each $x \in X$ is a change candidate. G: Set of component mappings across different granularity levels, including a self-mapping of components used in the product model to themselves. Each $g \in G$ is a component mapping between a set of coarse-grained components and their fine-grained counterparts.
Output: P: Set of components affected by X through change propagation. Each $p \in P$ is a set of (component name, combined change propagation likelihood) tuples for x.

```
1  $P = \emptyset$
2  for $x \in X$:
3    for $g \in G$:
4      open file containing $g$ (.csv)
5      for column in $g$:
6        if value in first row == $x$:
7          look for component(s) in subsequent row(s) of the column
8          open file containing $N$ (.csv)
9          adjust $N$ by using Eq. (5) on component(s) found
10         for row in column $x$ in $N$:
11           if cell value above likelihood threshold:
```



A
B
C
D
E
-
0.51
0.20
0.60
0.34
0.42
0.26
0.05
0.30
0.02
0.09
0.60
0.04
$L_{C,\delta} = min\left(\sqrt{L_{C,A}^2 + L_{C,B}^2}, 1\right)$
$= min\left(\sqrt{0.34^2 + 0.42^2}, 1\right)$
$= 0.54$
(refer to [Ahmad et al. 2013])
δ
0.54
0.30
0.61

Fig. 5. An example calculation for the adjusted product model.

```
12                  add (row header, cell value) to $p$
13              end
14              add $p$ to $P$
15              clear $p$
16          end
17      end
18  end
19  return $P$
```

Case example

A case example involving a heavy-duty diesel engine is presented in this section to demonstrate the feasibility of using the proposed method on an existing product. The actual process was automated. However, the intermediate results are broken down into two parts in this section to provide details. The first part describes the data collection of online comments from the *YouTube* platform and the identification of change candidates using the method (see Section "From online reviews to change candidates"). The second part describes the product model of the heavy-duty diesel engine used in this work and the prediction of change propagation arising from the change candidates (see Section "From change candidates to change propagation prediction"). A discussion on the generalizability of the proposed method on other products is provided in the section "Generalizability of the proposed method".

From online reviews to change candidates

Search words in the format of "brand/company product model" were used in this case example. The "product" in this case was an "engine". For confidentiality reasons, the "brand/company" of the engine is coded as "BX" in this paper. Three sets of "model" were used in the search. The first model was "truck", which is a generic description of the model type. The second and third models have unique names and are coded as "M1" and "M2" in this paper. The maximum number of *YouTube* videos to be found by each set of search words was limited to the top 20 results to filter out videos that were less relevant.

Based on the data collected on 19 March 2021, 1634 comments were captured for the search words "BX engine truck", 529 comments were captured for the search words "BX engine M1", and 1502 comments were captured for the search words "BX engine M2". The comments found were subsequently preprocessed according to the proposed method and a summary of the data is shown in Table 1. The entire set of comments used is available on https://github.com/NLPT/P1data. It can be seen that the number of comments found was highly dependent on the search words used. For instance, there were approximately 3 times more comments on videos associated with "M2" compared to "M1". The number of words found in each comment can differ as well with some having as many as 670 words in a single comment while others were left with no words after preprocessing. The average number of words per comment was found to be between 17 and 22 from the data collected.

Table 1. A summary of the data collected

Search words	Comments found	Max. word count in a comment	Min. word count in a comment	Avg. word count in a comment
BX engine truck	1,634	670	0	19
BX engine M1	529	646	0	17
BX engine M2	1,502	320	0	22

Table 2 shows a breakdown on the number of words analyzed for each video. The maximum number of words analyzed for a video was 8874 and the minimum was 0 (i.e., either no comment found or was left with no word after preprocessing). Note that some results were identical as they were derived from the same video. For instance, Video 20 of search words "BX engine truck" was the same video as Video 18 of search words "BX engine M1". Hence, for clarity, results derived from the same video are labeled with a superscript in Table 2. In total, one video was identical across the three sets of search words, two videos were identical between "BX engine truck" and "BX engine M1", and seven videos were identical between "BX engine truck" and "BX engine M2".

The top words and linked words associated with each set of search words were subsequently identified and a snippet of the results derived from Video 5 and 8 for "BX engine M2" is shown in Table 3. A threshold frequency of more than 30 occurrences was applied in the identification of top words. In addition, only words that appeared more than 25% of their top word occurrences would satisfy the threshold frequency as linked words. (A

Table 2. A summary on the number of words analyzed for each video

	Number of words		
Video ID	BX engine truck	BX engine M1	BX engine M2
1	8	836	[b]3,053
2	2,547	986	[c]1,348
3	155	521	[f]8,874
4	[a]186	3	2,934
5	[b]3,053	324	2,884
6	[c]1,348	1,036	68
7	1,677	5	815
8	238	4	[d]2,339
9	[d]2,339	788	52
10	118	615	934
11	[e]726	389	260
12	1,371	0	[e]726
13	[f]8,874	0	0
14	305	[d]2,339	[a]186
15	542	0	6,613
16	[g]734	0	[g]734
17	5,951	6	180
18	121	[h]1,095	583
19	132	37	621
20	[h]1,095	0	390

discussion on threshold frequency is provided in the section "Setting threshold frequency for top words and linked words".) For instance, the top words found for Video 5 were "injector" and "cup". "injector" appeared 52 times while "cup" appeared 41 times from the 149 comments collected from Video 5. The number of comments that included the top words were lower than the number of times the top words occurred as each top word can be mentioned more than once in a comment. For instance, the word "injector" only appeared in 41 comments but was mentioned 52 times. Sentiment analysis was later carried out on these 41 comments and the net sentiment was found to be "NEG" (i.e., negative). This implies that, out of the 41 comments, more comments were found to be negative than positive.

The word "cup" was found to be a linked word of "injector" and was mentioned 29 times in 20 comments that contained the word "injector". Indeed, the word "injector" was also found to be a linked word of "cup", suggesting that the two words were often discussed together. In addition, "injector cup" was mentioned 12 times in 9 comments as well, suggesting that some comments on "cup" and "injector" were referring to the "injector cup". The uncovering of "injector" (i.e., a component) and "injector cup" (i.e., a part of a component) highlights the potential of mining *n*-grams (i.e., consecutive words) to expose areas of concern at different levels of granularity.

Table 4 shows five comments containing the word "injector" and "cup" extracted from Video 5. Comment 1 focuses on the injector cup, Comment 2 treats the injector and the cup as two separate entities, Comment 3 mentions the injector cup alongside with the injector (the company was coded as "BX" in the comment), Comment 4 is a question on the injector and cup with Comment 5 as one of the replies. The sentiment of these comments was assessed independently by the author (i.e., ground

Table 3. A summary of top/linked words and the sentiment associated with those words

				No. of comments		
BX engine M2 Video ID	Top word	Linked word	Freq.	Total	Include	Net Sentiment
5	Injector	–	52	149	41	NEG
5	Injector	Cup	29	149	20	NEG
5	Cup	–	41	149	30	NEG
5	Cup	Injector	30	149	20	NEG
5	Cup	Injector cup	12	149	9	NEG
8	BX	–	55	67	34	Not NEG
8	BX	Truck	31	67	17	Not NEG
8	BX	Engine	24	67	13	Not NEG
8	Truck	–	36	67	21	Not NEG
8	Truck	BX	34	67	17	Not NEG
8	Truck	Engine	17	67	7	Not NEG
8	Truck	Problem	12	67	7	Not NEG
8	Engine	–	34	67	22	NEG
8	Engine	BX	21	67	13	Not NEG
8	Engine	Truck	20	67	7	Not NEG
8	Engine	Fuel	16	67	8	NEG
8	Engine	Problem	14	67	7	NEG

Table 4. Comments with "injector" and "cup" in Video 5 and their sentiment scores

		Sentiment	
No.	Comments	Author	Language model
1	Injector cup tip broken and stuck on cylinder head. Any suggestions how to remove it?	Negative	Negative
2	When pulling injectors, loosen either the feed or return line on the head to let fuel drain. When pulling cups, use a 15/16 ratcheting wrench ($20 from Home Depot) on the big nut, stick a rolling head prybar or screwdriver in a rocker shaft hole to keep the 13 mm wrench from moving. You'll have all cups pulled in 3 min.	Positive	Neutral
3	Great video 👍<br />I have been going from shop to shop with injector cups problem. Just day ago pulled out of the BX dealer drove 550 miles and my coolant starts leaking again pushing it through the injectors, pressurizing the system. Had two full engine rebuilds on two separate trucks, basically every year I need an engine rebuild. Do you think that the injector cup leak issue could cause a future engine failure? Because I have to rebuild whole engine after injector cups leak I don't know if it could be connected in anyway but it seems like coolant washes out cylinder oil and engine running dry long enough and after the leak fix that cylinder locks up and drops rod destroying entire engine. Idk 🙈	Negative	Negative
4	Do I have to chance the injectors everytime I change a cup? Or could I just change the cups only.	Neutral	Neutral
5	Cups need to be changed any time injectors are removed. Injectors need to be replaced only if its worn or pitted on mating surface to cup.	Neutral	Neutral

truth) and listed next to those classified by the RoBERTa-base language model (Liu *et al.*, 2019; Barbieri *et al.*, 2020) in Table 4. It can be seen that the sentiment classification produced by the language model is not always aligned with the author's (e.g., Comment 2). Hence, in an effort to examine the extent of classification error, all 149 comments from Video 5 were labeled independently by the author and subsequently compared with the classification results produced by the RoBERTa-base language model used in this work (see Table 5). The precision, recall, and F1 score of the results were also calculated to support comparison (see Table 6). Precision refers to the number of correct classifications (e.g., true positive) out of all the entities predicted for that class (i.e., true positive and false positive). Recall refers to the number of correct classifications (e.g., true positive) out of all the entities that are actually in that class (i.e., actual positive). F1 score refers to the harmonic mean of precision and recall (equations can be found in Tuarob *et al.* (2018)). It was found that the overall precision, recall, and F1 score for the 149 comments analyzed were 0.74, 0.68, and 0.70, respectively. As pointed out in the introduction, this paper focuses on the integration of a pre-trained language model for social media sentiment analysis instead of the fine-tuning of it. Hence, the precision, recall, and F1 score reported here only serve as an indication of the classification error when using the RoBERTa-base language model in this work. As a comparison, a similar overall recall of 0.726 was reported in Barbieri *et al.* (2020) where the same RoBERTa-base language model was applied on a different set of text (precision and F1 score not reported).

Table 5. Confusion matrix for comments in Video 5

	Predicted (i.e., Language model)		
Actual (i.e., Author)	Positive	Negative	Neutral
Positive	26	2	14
Negative	0	19	13
Neutral	8	4	63

Table 7 shows a summary on the word collections created and change candidates found for each set of search words. It can be seen that 34 top words and 125 linked words were identified from videos found under "BX engine M2", which was the highest across the search words used. These top words and their linked words, together with the permutations between them, resulted in 284 word collection entries, with 198 of the entries having a negative sentiment. The entries in the word collection were subsequently checked against a component list provided by the company that produces the engine (company coded as "BX" in this paper) and 2 components were identified as change candidates out of a set of 32 components named in the component list. The change candidates identified were the "injector" and the "fuel pump". Both components were also identified as change candidates when the analysis was carried out under search words "BX engine truck". However, no change candidate was identified under "BX engine M1".

From change candidates to change propagation prediction

A product model of the heavy-duty diesel engine was analyzed in this part of the case example to predict how changes can propagate if the change candidates are changed. Similar to the

Table 6. Precision, recall, and F1 scores for the comments analyzed

Sentiment	Precision	Recall	F1 score
Positive	0.76	0.62	0.68
Negative	0.76	0.59	0.67
Neutral	0.70	0.84	0.76
Overall	*0.74*	*0.68*	*0.70*

Table 7. A summary of word collections and change candidates for different search words

			Word collection		
Search words	Number of Top words	Number of Linked words	Number of entries	Negative sentiment	Change candidates
BX engine truck	28	91	210	136	2
BX engine M1	3	9	21	5	0
BX engine M2	34	125	284	198	2

component list, the product model used in this work was provided by the company that produces the engine (company coded as "BX" in this paper) and is shown in Figure 6 with the component names replaced with component numbers. As an indicator, the components in the model include the "cylinder block", "piston", "crank shaft", "engine anchorage", and "oil pump". The "fuel pump" and "injector" (i.e., change candidates) were also in the model and are labeled as Components 22 and 23, respectively. The cell entries in Figure 6 are direct change propagation likelihood between components and are color coded for ease of visualization. Yellow cells represent entries with low change propagation likelihood and an assigned value of 0.25. Orange cells represent entries with medium change propagation likelihood and an assigned value of 0.50. Lastly, red cells represent entries with high change propagation likelihood and an assigned value of 0.75. The product model was subsequently processed as

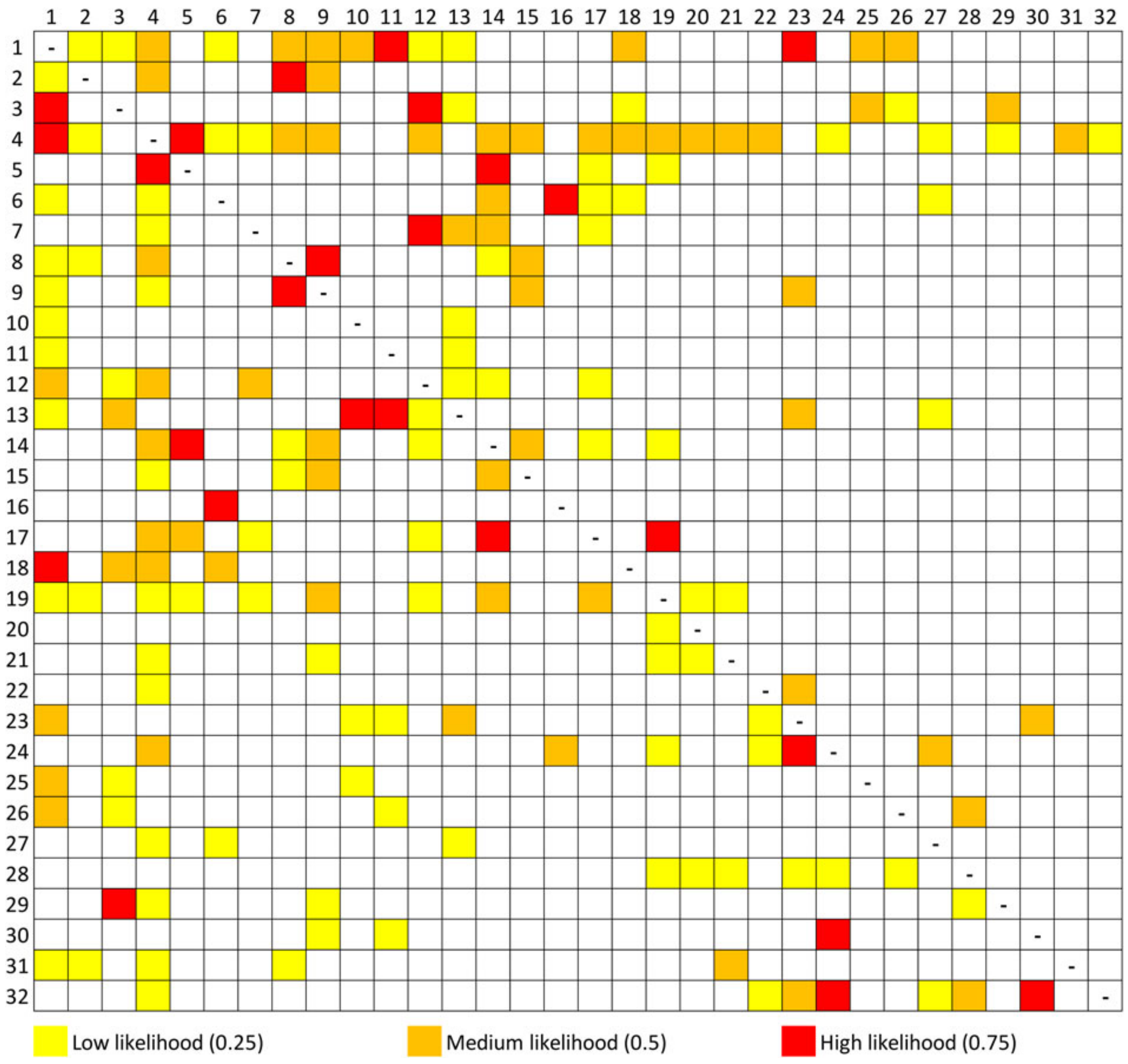


Fig. 6. Direct change propagation likelihood between engine components.

Table 8. Combined change propagation likelihood from the "injector" to other components

Component number	1	2	3	4	5	6	7	8	9	10	11
Combined Likelihood	**0.81**	0.30	0.35	**0.55**	0.23	0.20	0.25	0.32	**0.57**	0.12	0.12
Component number	12	13	14	15	16	17	18	19	20	21	22
Combined Likelihood	0.32	**0.58**	0.29	0.19	0.04	0.27	0.34	0.31	0.02	0.12	**0.52**
Component number	23	24	25	26	27	28	29	30	31	32	
Combined Likelihood	–	**0.79**	0.17	0.22	0.11	0.33	0.19	0.24	0.18	**0.67**	

described in the proposed method and was updated with combined change propagation likelihood. Adjustment to the model granularity was not required in this case example as only one component list was used.

Table 8 shows a summary of the combined change propagation likelihood from the "injector" (i.e., a change candidate) to all engine components. It can be seen that all components can be affected by change propagation if the "injector" was changed (Component 23 is the "injector"). A likelihood threshold of 0.5 was later applied to identify components that were more likely to be affected and Components 1, 4, 9, 13, 22, 24, and 32 were found to be above the likelihood threshold (see Tables 8 and 9, entries in bold). This information can subsequently be used to better prepare for design changes. For instance, Figure 7 shows a change propagation tree with the "injector" as the point of change initiation and the other components plotted according to their shortest path from the "injector". If the "injector" is to be changed, efforts should be made to prevent change propagation to Components 1, 4, 9, 13, 22, 24, and 32 as they were predicted as more likely to be affected. This is especially true for Components 1, 9, 22, 24, and 32 as they have direct linkages with the "injector" and changing any of these components can open up an indirect change propagation path for Component 4 to be affected.

The combined change propagation likelihood from the "fuel pump" to all engine components is summarized in Table 9. Similar to the "injector", all components can be affected by change propagation if the "fuel pump" was changed (Component 22 is the "fuel pump"). However, only Component 4 was found to have a combined change propagation likelihood above the 0.5 likelihood threshold. By referring to the change propagation tree shown in Figure 8, it can be seen that Component 4 is directly linked to the "fuel pump" and is also directly linked to 17 downstream components. Hence, even though the combined change propagation likelihood for other components were below the likelihood threshold, changing Component 4 can open up a gateway for changes to propagate further. Effort should, therefore, be made to prevent changes on Component 4.

In summary, by using the proposed method, a query on "BX engine M1" did not return any change candidate but queries on "BX engine truck" and "BX engine M2" identified the same set of change candidates (i.e., "injector" and "fuel pump"). Both queries also uncovered additional engine components that may be affected through change propagation (see Table 10). Note that Component 4 is listed twice under "BX engine truck" and "BX engine M2" as it can be affected through direct change propagation from Component 22 and indirect change propagation from Component 23.

Discussions

The proposed method shows promise as demonstrated in the case example. Nevertheless, limitations were observed and are discussed here along with the assumptions made in this work.

Setting threshold frequency for top words and linked words

The threshold frequency to be used is influenced by a number of factors. Unlike traditional means of complaining through telephones and letters where companies can decide when and whether to respond, negative online comments through social media are visible to the public and need to be addressed swiftly (Laufer and Coombs, 2006; Einwiller and Steilen, 2015;

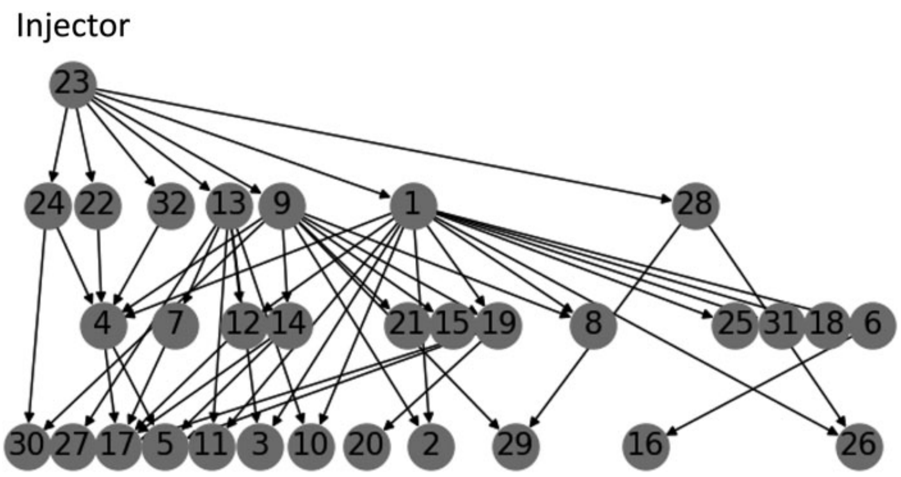


Fig. 7. Change propagation tree of "injector".

Table 9. Combined change propagation likelihood from the "fuel pump" to other components

Component number	1	2	3	4	5	6	7	8	9	10	11
Combined Likelihood	0.21	0.14	0.11	**0.54**	0.17	0.11	0.12	0.15	0.13	0.02	0.02
Component number	12	13	14	15	16	17	18	19	20	21	22
Combined Likelihood	0.15	0.09	0.17	0.09	0.02	0.18	0.15	0.16	0.01	0.06	–
Component number	23	24	25	26	27	28	29	30	31	32	
Combined Likelihood	0.28	0.37	0.04	0.05	0.06	0.07	0.08	0.10	0.09	0.39	

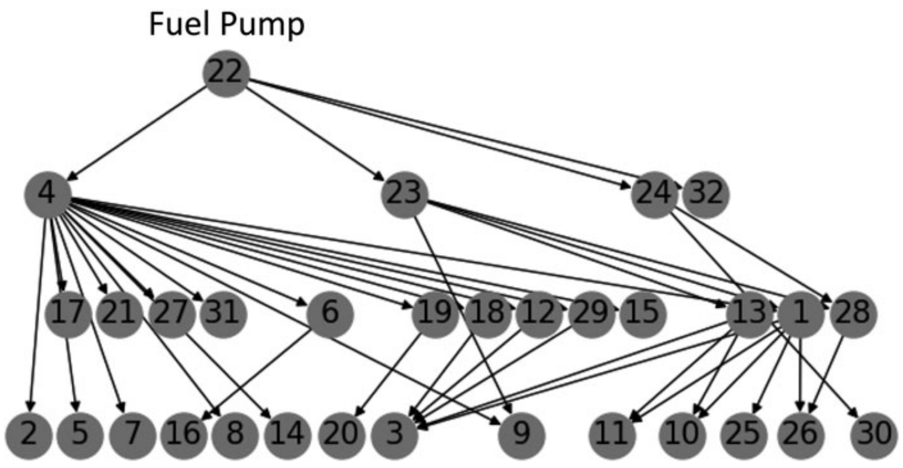


Fig. 8. Change propagation tree of "fuel pump".

Grégoire *et al.*, 2015; Javornik *et al.*, 2020). From this perspective, the threshold frequency for top words and linked words should be set as low as possible to capture any mention of product components that are negative in sentiment. However, due to limited design resources and the need to prioritize change (Koh *et al.*, 2015), the threshold frequency should be open to adjustments to identify change candidates that are mentioned more frequently. Indeed, it is important to point out that the threshold frequency is directly influenced by the data at hand. For instance, in the case example presented, the number of words analyzed in each YouTube video varied from 0 to 8874 words. Hence, having a fixed threshold frequency across unrelated studies can be impractical.

In this work, the minimum threshold frequency used for identifying top words was arbitrarily set as 30 occurrences and the threshold frequency for linked words was set as 25% of their top word occurrences. The setting is represented as Setting 1 in Table 11 and it identified two change candidates, namely the "fuel pump" and "injector", for search words "BX engine truck". Based on the proposed method, "fuel pump" was identified as a change candidate as both the words "fuel" and "pump" satisfied the minimum threshold frequency and their associated comments were negative. Indeed, the highest frequency for "fuel" was found to be 36 occurrences (i.e., more than 30) in Video 13 and "pump" was identified as a linked word for "fuel" as it appeared 14 times in the same comments where "fuel" was mentioned (i.e., 14 out of 36 occurrences = 39%, more than 25%). Hence, "fuel pump" would still be identified as a change candidate if the minimum threshold frequency for top words and linked words was set lower as shown in Setting 2 and 3, respectively. The number of change candidates found did not increase for both settings even though more top words and linked words were identified with the lowered threshold. In contrast, "fuel" would not be identified as a top word if the threshold frequency for top words was increased to 40 occurrences as shown in Setting 4. "pump" would also cease to be a linked word of "fuel" if the threshold frequency for linked words was increased from 25% to 50% in Setting 5. Hence, "fuel pump" would not be identified as a change candidate if Setting 4 and 5 were used. On the other hand, "injector" which has 47 occurrences in Video 13 would still be identified in Setting 4 and 5.

While Setting 2 to 5 demonstrate how the case example results can differ if minor adjustments were made to the threshold frequency, Setting 6 represents the scenario where the threshold frequency was set as low as possible to include all words that were mentioned at least once (i.e., >0) with negative sentiment.

Table 10. A summary of change prediction results

	Component number		
Search words	Change candidates	Direct change propagation	Indirect change propagation
BX engine truck	22, 23	1, 4, 9, 13, 22, 24, 32	4
BX engine M1	–	–	–
BX engine M2	22, 23	1, 4, 9, 13, 22, 24, 32	4

Table 11. Influence of top word and linked word threshold frequency on case example results

	Threshold frequency		"BX engine truck"		
Setting No.	Top word (min. occurrence)	Linked word (min %)	Number of Top words	Number of Linked words	Number of change candidates
1	30	25	28	91	2
2	20	25	60	326	2
3	30	10	28	721	2
4	40	25	14	39	1
5	30	50	28	14	1
6	>0	>0	957	56,710	5

Based on Setting 6, 957 top words and 56,710 linked words were found and five change candidates were eventually identified, namely the "fuel pump", "injector", "oil filter", "oil pump", and "timing gear". This demonstrates that a maximum of five change candidates can be uncovered from the case example data and the threshold frequencies used in the case example could have been adjusted lower to identify more change candidates, if needed. Note that this work uses threshold frequencies instead of defining the maximum number of change candidates to be found so as to maintain a consistent frequency threshold within the same study (i.e., consistent threshold for data collected through search words "BX engine truck", "BX engine M1", and "BX engine M2").

Generalizability of the proposed method

In this study, the proposed method was applied on three other products from different domains to explore the generalizability of this work. The three products were a vacuum cleaner (a small domestic appliance), a washing machine (a large domestic appliance), and a car (a complex machine). The search words used were "Dyson Vacuum Cleaner V10", "Samsung Washing Machine QuickDrive", and "Mercedes Benz E200", respectively. As the official component lists of these products were not available in this study, the evaluation in this section was carried out by manually identifying top words and permutations of top and linked words (i.e., word collection entries) that resemble component names for each product. A summary of the findings is presented in Table 12.

It can be seen that the number of top words and linked words identified varies between the products analyzed. In addition, having more top words and linked words do not necessarily result in more word collection entries that resemble component names. In the case of the vacuum cleaner, the word collection entries that resemble component names were "battery", "filter", "bin", "cleaning head", and "roller brush". For the washing machine, the word collection entry identified was "washer dryer". For the car, the word collection entries identified were "steering wheel", "fake exhaust", "airbags", "front", "rear", "interior", and "exterior" (the latter four are valid component references in the body-in-white stage). The findings suggest that the proposed method can be applied beyond the design of diesel engines presented in the case example, although success can vary between products based on their online presence. Future work should further evaluate the proposed method by using a diverse pool of online data across different product domains.

Conclusion

This paper presents a method that builds on a RoBERTa-based language model to identify new ideas from large volume social media data and uses the CPM as a basis to uncover the change impact associated with the new ideas. The work is positioned in the context of incremental design and the goal of the method is to enable early prediction of change impact on the product architecture to aid decision quality in product planning. A case example involving a heavy-duty diesel engine was carried out to demonstrate the feasibility of the proposed method and the results produced were promising. Based on three sets of search words, the method retrieved and processed 3665 *YouTube* comments and identified two unique candidates for design change, with six other product components predicted as likely to be affected through change propagation.

From an academic perspective, the work contributes toward connecting two fields of study that are usually considered in isolation through the provision of a set of algorithms that integrate the use of natural language processing in customer sentiment analysis with the use of dependency modeling in change prediction. The work can be used as a starting point or a construct

Table 12. Exploratory results on different products

Products	Number of Top words	Number of Linked words	Number of word collection entries	Number of entries resembling component names
Vacuum cleaner	101	407	915	5
Washing machine	11	25	61	1
Car	74	313	700	7

for other research in product planning and design change management to build upon.

From an industry perspective, the proposed method and the outcome of the analysis can empower practitioners such as the product planning function within a company to ask more advanced questions. For instance, instead of asking "Which part of the product should we change to address negative sentiment found on social media?" and "Which product components can be affected by the change?" the output of the method enables users to further query "Which teams are responsible in implementing the change?" and "What are the resources required to carry out the change?" The exploratory analysis to apply the method across different product domains such as domestic appliances and a complex machine also yielded favorable results, suggesting that the method can be applied beyond the design of diesel engines and serve companies in other product domains. Nevertheless, the author acknowledge that more work needs to be done to further evaluate the proposed method across a wider range of products.

References

Agrawal R, Imieliński T and Swami A (1993) Mining association rules between sets of items in large databases. *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pp. 207–216.

Ahmad N, Wynn DC and Clarkson PJ (2013) Change impact on a product and its redesign process: a tool for knowledge capture and reuse. *Research in Engineering Design* **24**, 219–244.

Archak N, Ghose A and Ipeirotis PG (2011) Deriving the pricing power of product features by mining consumer reviews. *Management Science* **57**, 1485–1509.

Barbieri F, Camacho-Collados J, Neves L and Espinosa-Anke L (2020) Tweeteval: Unified benchmark and comparative evaluation for tweet classification. *arXiv preprint*. arXiv:2010.12421.

BBC News (2009) Design changes cause Boeing loss. Available at http://news.bbc.co.uk/1/hi/business/8318519.stm

Blei DM, Ng AY and Jordan MI (2003) Latent Dirichlet allocation. *Journal of Machine Learning Research* **3**, 993–1022.

Cardin M-A, Kolfschoten GL, Frey DD, de Neufville R, de Weck OL and Geltner DM (2013) Empirical evaluation of procedures to generate flexibility in engineering systems and improve lifecycle performance. *Research in Engineering Design* **24**, 277–295.

Chang D and Lee C (2018) A product affective properties identification approach based on web mining in a crowdsourcing environment. *Journal of Engineering Design* **29**, 449–483.

Chen L, Zheng Y, Xi J and Li S (2020) An analysis method for change propagation based on product feature network. *Research in Engineering Design* **31**, 491–503.

Clarkson PJ, Simons C and Eckert CM (2004) Predicting change propagation in complex design. *Journal of Mechanical Design* **126**, 765–797.

Devlin J, Chang M-W, Lee K and Toutanova K (2018) BERT: pre-training of deep bidirectional transformers for language understanding. *arXiv preprint*. arXiv:1810.04805.

Duran-Novoa R, Weigl JD, Henz M and Koh ECY (2018) Designing in young organisations: engineering change propagation in a university design project. *Research in Engineering Design* **29**, 489–506.

Eckert CM, Clarkson PJ and Zanker W (2004) Change and customisation in complex engineering domains. *Research in Engineering Design* **15**, 1–21.

Einwiller SA and Steilen S (2015) Handling complaints on social network sites – an analysis of complaints and complaint responses on Facebook and twitter pages of large US companies. *Public Relations Review* **41**, 195–204.

Eppinger SD and Browning TR (2012) *Design Structure Matrix Methods and Applications*. Cambridge, MA: MIT Press.

Fernandes J, Henriques E, Silva A and Moss MA (2015) Requirements change in complex technical systems: an empirical study of root causes. *Research in Engineering Design* **26**, 37–55.

Giffin M, de Weck O, Buonova G, Keller R, Eckert CM and Clarkson PJ (2009) Change propagation analysis in complex technical systems. *Journal of Mechanical Design* **131**, 081001, 1–14.

Grégoire Y, Salle A and Tripp TM (2015) Managing social media crises with your customers: the good, the bad, and the ugly. *Business Horizons* **58**, 173–182.

Hamraz B and Clarkson PJ (2015) Industrial evaluation of FBS linkage – a method to support engineering change management. *Journal of Engineering Design* **26**, 24–47.

Hamraz B, Hisarciklilar O, Rahmani K, Wynn DC, Thomson V and Clarkson PJ (2013) Change prediction using interface data. *Concurrent Engineering* **21**, 141–154.

Hutto C and Gilbert E (2014) Vader: A parsimonious rule-based model for sentiment analysis of social media text. *Proceedings of the International AAAI Conference on Web and Social Media*, Vol. **8**, No. 1, pp. 216–225.

Jarratt TAW, Eckert CM, Caldwell NHM and Clarkson PJ (2011) Engineering change: an overview and perspective on the literature. *Research in Engineering Design* **22**, 103–124.

Javornik A, Filieri R and Gumann R (2020) "Don't forget that others are watching, too!" The effect of conversational human voice and reply length on observers' perceptions of complaint handling in social media. *Journal of Interactive Marketing* **50**, 100–119.

Jiang H, Kwong CK and Yung KL (2017) Predicting future importance of product features based on online customer reviews, *Journal of Mechanical Design* **139**, 111413, 1–10.

Jin J, Liu Y, Ji P and Liu H (2016) Understanding big consumer opinion data for market-driven product design. *International Journal of Production Research* **54**, 3019–3041.

Kang SW and Tucker CS (2016) An automated approach to quantifying functional interactions by mining large-scale product specification data. *Journal of Engineering Design* **27**, 1–24.

Kasperek D, Schenk D, Kreimeyer M, Maurer M and Lindemann U (2016) Structure-based system dynamics analysis of engineering design processes. *Systems Engineering* **19**, 278–298.

Keller R (2007) *Predicting Change Propagation: Algorithms, Representations, Software Tools* (PhD thesis). Department of Engineering, University of Cambridge, UK.

Koh ECY (2017) A study on the requirements to support the accurate prediction of engineering change propagation. *Systems Engineering* **20**, 147–157.

Koh ECY, Caldwell NHM and Clarkson PJ (2012) A method to assess the effects of engineering change propagation. *Research in Engineering Design* **23**, 329–351.

Koh ECY, Caldwell NHM and Clarkson PJ (2013) A technique to assess the changeability of complex engineering systems. *Journal of Engineering Design* **24**, 477–498.

Koh ECY, Forg A, Kreimeyer M and Lienkamp M (2015) Using engineering change forecast to prioritise component modularisation. *Research in Engineering Design* **26**, 337–353.

Kreimeyer M (2016) Implementing product architecture in industry: impact of engineering design research. In Chakrabarti A and Lindemann U (eds), *Impact of Design Research on Industrial Practice*. Cham: Springer, pp. 383–398.

Kreimeyer M and Lindemann U (2011) *Complexity Metrics in Engineering Design - Managing the Structure of Design Processes*. Berlin, Heidelberg: Springer.

Laufer D and Coombs WT (2006) How should a company respond to a product harm crisis? The role of corporate reputation and consumer-based cues. *Business Horizons* **49**, 379–385.

Lee J and Hong YS (2017) Bayesian network approach to change propagation analysis. *Research in Engineering Design* **28**, 437–455.

Lim S and Tucker CS (2016) A Bayesian sampling method for product feature extraction from large scale textual data. *Journal of Mechanical Design* **138**, 061403, 1–9.

Lim S, Conrad S and Tucker CS (2017) Mitigating online product rating biases through the discovery of optimistic, pessimistic, and realistic reviewers. *Journal of Mechanical Design* **139**, 111409, 1–11.

Lindemann U, Maurer M and Braun T (2009) *Structural Complexity Management – An Approach for the Field of Product Design*. Berlin, Heidelberg: Springer.

Liu Y, Jin J, Ji P, Harding JA and Fung RYK (2013) Identifying helpful online reviews: a product designer's perspective. *Computer-Aided Design* **45**, 180–194.

Liu Y, Ott M, Goyal N, Du J, Joshi M, Chen D, Levy O, Lewis M, Zettlemoyer L and Stoyanov V (2019) RoBERTa: a robustly optimized BERT pretraining approach. *arXiv preprint.* arXiv:1907.11692.

Long M, Erickson M and MacDonald EF (2019) Consideration-constrained engineering design for strategic insights. *Journal of Mechanical Design* **141**, 064501, 1–8.

Maier JF, Eckert CM and Clarkson PJ (2017) Model granularity in engineering design – concepts and framework. *Design Science* **3**, 1–29.

Pratama MO, Satyawan W, Jannati R, Pamungkas B, Raspiani , Syahputra ME and Neforawati I (2019) The sentiment analysis of Indonesia commuter line using machine learning based on Twitter data. *Journal of Physics: Conference Series* **1193**, 1–6.

Roth M, Wolf M and Lindemann U (2015) Integrated matrix-based fault tree generation and evaluation. *Procedia Computer Science* **44**, 599–608.

Sarica S, Luo J and Wood K (2020) Technet: technology semantic network based on patent data. *Expert Systems with Applications* **142**, 112995.

Sha Z, Saeger V, Wang M, Fu Y and Chen W (2017) Analyzing customer preference to product optional features in supporting product configuration. *SAE International Journal of Materials and Manufacturing* **10**, 320–332.

Shankar P, Morkos B and Summers JD (2012) Reasons for change propagation: a case study in an automotive OEM. *Research in Engineering Design* **23**, 291–303.

Siddharth L and Sarkar P (2017) A methodology for predicting the effect of engineering design changes. *Procedia CIRP* **60**, 452–457.

Song B, Luo J and Wood K (2019) Data-driven platform design: patent data and function network analysis. *Journal of Mechanical Design* **141**, 021101, 1–10.

Suh ES, de Weck O and Chang D (2007) Flexible product platforms: framework and case study. *Research in Engineering Design* **18**, 67–89.

Sun H, Guo W, Shao H and Rong B (2020) Dynamical mining of ever-changing user requirements: a product design and improvement perspective. *Advanced Engineering Informatics* **46**, 1–11.

Suryadi D and Kim H (2018) A systematic methodology based on word embedding for identifying the relation between online customer reviews and sales rank. *Journal of Mechanical Design* **140**, 121403, 1–12.

Tang D, Yin L, Wang Q, Ullah I, Zhu H and Leng S (2016) Workload-based change propagation analysis in engineering design. *Concurrent Engineering* **24**, 17–34.

Tang M, Jin J, Liu Y, Li C and Zhang W (2019) Integrating topic, sentiment, and syntax for modeling online reviews: a topic model approach. *Journal of Computing and Information Science in Engineering* **19**, 011001, 1–12.

Tuarob S and Tucker CS (2015*a*) Automated discovery of lead users and latent product features by mining large scale social media networks. *Journal of Mechanical Design* **137**, 071402, 1–11.

Tuarob S and Tucker CS (2015*b*) Quantifying product favorability and extracting notable product features using large scale social media data. *Journal of Computing and Information Science in Engineering* **15**, 031003, 1–12.

Tuarob S, Lim S, Conrad S and Tucker CS (2018) Automated discovery of product feature inferences within large-scale implicit social media data. *Journal of Computing and Information Science in Engineering* **18**, 021017, 1–14.

Tucker CS and Kim HM (2009) Data-driven decision tree classification for product portfolio design optimization. *Journal of Computing and Information Science in Engineering* **9**, 041004, 1–14.

Tucker CS and Kim HM (2011) Trend mining for predictive product design. *Journal of Mechanical Design* **133**, 111008, 1–11.

Ullah I, Tang D, Yin L, Hussain I and Wang Q (2018) Cost-effective propagation paths for multiple change requirements in the product design. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* **232**, 1572–1585.

Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L and Polosukhin I (2017) Attention is all you need. *arXiv preprint.* arXiv:1706.03762.

Wang M, Sha Z, Huang Y, Contractor N, Fu Y and Chen W (2018) Predicting product co-consideration and market competitions for technology-driven product design: a network-based approach. *Design Science* **4**, 1–34.

Wang WM, Tian ZG, Li Z, Wang JW, Barenji AV and Cheng MN (2019) Supporting the construction of affective product taxonomies from online customer reviews: an affective-semantic approach. *Journal of Engineering Design* **30**, 445–476.

Wickel MC and Lindemann U (2015) How to build up an engineering change dependency model based on past change data? *Proceedings of the 17th International Dependency and Structure Modelling Conference*, Texas, pp. 221–231.

Xie Y and Ma Y (2016) Well-controlled engineering change propagation via a dynamic inter-feature association map. *Research in Engineering Design* **27**, 311–329.

Xie J, Bi Y, Sha Z, Wang M, Fu Y, Contractor N, Gong L and Chen W (2020) Data-driven dynamic network modeling for analyzing the evolution of product competitions. *Journal of Mechanical Design* **142**, 031112, 1–14.

Yassine AA and Khoury WG (2021) Optimizing module upgrade decisions to maximize profits. *Research in Engineering Design* **32**, 49–70.

Yin L, Tang D, Wang Q, Ullah I and Zhang H (2017) Engineering change management of product design using model-based definition technology. *Journal of Computing and Information Science in Engineering* **17**, 041006, 1–19.

Yin L, Sun Q, Xu Y, Shao L and Tang D (2021) Risk analysis of engineering change for distributed product design. *Journal of Computing and Information Science in Engineering* **21**, 041003, 1–11.

Zhan J, Loh HT and Liu Y (2009) Gather customer concerns from online product reviews – a text summarization approach. *Expert Systems with Applications* **36**, 2107–2115.

Zhou F, Jiao R and Linsey JS (2015) Latent customer needs elicitation by use case analogical reasoning from sentiment analysis of online product reviews. *Journal of Mechanical Design* **137**, 071401, 1–12.

Zhou F, Ayoub J, Xu Q and Yang XJ (2020) A machine learning approach to customer needs analysis for product ecosystems. *Journal of Mechanical Design* **142**, 011101, 1–13.

Dr. Edwin C.Y. Koh is Senior Lecturer & Associate Program Head (Design and Artificial Intelligence) at the Singapore University of Technology and Design (SUTD). He serves on the review committee of the United States National Academy of Engineering (NAE) Grand Challenges Scholars Program (GCSP) and has held full-time and visiting positions at the National University of Singapore and New York University. Edwin received his B.Eng. in Mechanical Engineering and S.M. in Manufacturing from the Nanyang Technological University. He attended the Massachusetts Institute of Technology as part of his master's studies and holds a Ph.D. in Engineering Design from the University of Cambridge.