Decision Support Systems 135 (2020) 113304

Contents lists available at ScienceDirect

Decision Support Systems

journal homepage: www.elsevier.com/locate/dss

A generic framework for sentiment analysis: Leveraging opinion-bearing data to inform decision making

Jacqueline Kazmaier*, Jan H. van Vuuren

Stellenbosch Unit for Operations Research in Engineering, Department of Industrial Engineering, Stellenbosch University, Private Bag XI, Matieland 7602, South Africa

ARTICLE INFO

Keywords:
Sentiment analysis
Machine learning
Natural language processing
Decision Support Systems

ABSTRACT

The increased exposure of the average citizen and customer to polarised content from various sources has been of significant consequence for companies and governmental organisations. Such content has, for example, served as a catalyst for violent uprisings and shifts in stock market prices. The collection and study of opinion have therefore become a necessity in many industries. Due to the vast nature of such data, manual approaches to this problem are no longer feasible. Several computational approaches have been proposed within the field of sentiment analysis, which successfully address many aspects of this problem, such as the classification of data into one of several sentiment categories. The research in the field is lacking, however, with respect to the integration and application of these techniques in practice, as well as their incorporation into the decision-making process of affected entities. In this paper, a generic framework for sentiment analysis is proposed, with a focus on facilitating the model development process for a user in a manner such that good performance may be achieved irrespective of the problem domain, as well as facilitating a flexible, exploratory analysis of model results in combination with existing structured attributes in order to gain actionable insights. The objective of the framework is to aid organisations in successfully leveraging unstructured, opinion-bearing data in combination with structured data sources to inform decision making.

1. Introduction

Public opinion has long been an area of research interest. With the exponential growth of the Internet and social media, and the resulting increase in the volume of user-generated content publicly available online, this interest has become even more pronounced. Concomitantly, however, the process of manually studying such content has become increasingly cumbersome. This situation gave rise to the research field of *sentiment analysis* or *opinion mining* — the *computational* study of people's opinions, attitudes and emotions [1].

The potential applications of sentiment analysis are vast and powerful. Polarised content on the Internet has swayed public elections, led to events of mass activism, such as the Egyptian Revolution, and has significantly impacted the reputation of various organisations, with shifts in sentiment on social media having been shown to correlate with shifts in the stock market [1–3]. The collection and study of opinion, using both external data from the Internet and internal data such as direct customer communication, have therefore become a necessity in many industries [1].

Whilst there is an abundance of research dedicated to developing algorithms for the purpose of classifying sentiment, little guidance exists on how to apply these algorithms in practice, as well as how to incorporate model results into the decision-making process of affected entities.

Existing frameworks for sentiment analysis in the literature focus on the implementation of a *specific* model with a predetermined set of features. The fact that the classification performance of such a model depends on the domain in which it is applied, as well as the feature set employed as input [4,5], is typically not addressed by these frameworks. Furthermore, the process of tuning the parameters of machine learning models, in particular, is rarely included in such frameworks in spite of it being an essential part of the model development process [6].

Moreover, only few frameworks go beyond simple summaries of the proportional representation of specified sentiment classes as determined by the results of a given classification model. In order to effectively leverage sentiment analysis in an organisation, a deeper analysis of the model results is required, focusing on the topics discussed in reviews and the relationship between sentiment and various additional sources of information that may be available.

In this paper, a generic framework for the evaluation of unstructured, opinion-bearing text data is presented that addresses these shortcomings. This framework facilitates the process of preparing the data for analysis, extracting and selecting features from the unstructured text, as well as evaluating and selecting (or combining) suitable models used for classifying sentiment. Furthermore, the analysis and synthesis of the results of selected models is accommodated, with the aim of facilitating the

* Corresponding author.
E-mail addresses: jqkazmaier@gmail.com, 18214827@sun.ac.za (J. Kazmaier), vuuren@sun.ac.za (J.H. van Vuuren).

https://doi.org/10.1016/j.dss.2020.113304
Received 3 November 2019; Received in revised form 14 April 2020; Accepted 14 April 2020
Available online 06 May 2020

extraction of patterns and information from the data and presenting these to the user in a meaningful way. The framework may be integrated into a *decision support system* (DSS) aimed at aiding organisations in successfully leveraging unstructured, opinion-bearing data in combination with structured data sources to inform decision making.

The remainder of this paper is structured as follows. An overview of existing frameworks for sentiment analysis is given in Section 2, further emphasising the shortcomings of these frameworks. Our proposed framework is then presented in detail in Section 3. Subsequently, the practical value of the framework is showcased in Section 4 by means of four case studies from various domains. Finally, the paper closes with a short discussion of its contents and some proposals for future work in Section 5.

2. Literature review

A number of frameworks for sentiment analysis have been proposed in the literature. In this section, an overview of existing frameworks is given, highlighting in particular those aspects which have not been adequately addressed by these frameworks.

In a notable paper, Khan et al. [7] presented a generic sentiment analysis framework at the heart of which lies a *sentiment engine* that encompasses preprocessing and linguistic processing, as well as topic discovery and sentiment analysis tasks. This sentiment engine takes input data from various sources and its results are presented *via* an online dashboard. At this high level of abstraction, the framework comprises the important elements required for a sentiment analysis framework: Input data are retrieved and preprocessed, information extraction is performed in order to describe the contents of the data, sentiment analysis is performed to classify the sentiment polarity of the data, and results are communicated in some summarised form through a *graphical user interface* (GUI). It is, however, challenging to find existing frameworks in the literature that incorporate all of these elements at a lower level of abstraction (so that the framework may readily be applied), whilst retaining a generic nature.

In the remainder of their paper, Khan et al. [7] proposed a more specific framework, *Twitter Opinion Mining Framework* (TOM), in detail, which was tailored to the analysis of posts from the social media site *Twitter*. In contrast to the generic framework described previously, TOM encompasses only preprocessing and sentiment classification steps, and exhibits a very specific architecture based on a lexicon-based hybrid classification model.

Several other frameworks [8–10] similarly propose a specific, rather than a generic, approach to sentiment classification, containing only data preprocessing and sentiment classification components. Typically, the aim in these frameworks is to improve classification performance for a specific domain or problem type [8,9,11] or a specific language [10] using preprocessing techniques and sentiment classification models specifically tailored to the relevant domain.

Whilst an accurate sentiment classification model is necessary to evaluate opinionated content, it is not sufficient to form an understanding of the overall sentiment present in the data. When examining customer feedback, for example, it is also necessary to identify which aspects of a product or service contributed to customer satisfaction or dissatisfaction as well as any trends that may indicate why certain customer segments are (dis)satisfied. Furthermore, it is not guaranteed that a model which is highly accurate in one domain will perform equally well in another domain. This is especially true for lexicon-based approaches, which are dominant in the literature related to sentiment analysis frameworks. Such methods often depend on lexical resources, which may not be available in a given domain or language, or may not yield the same level of performance when adapted to a new problem setting.

Perhaps the best-known opinion mining framework, the *Opinion Observer* published by Liu et al. [12], addressed the first of these issues. It focused on the summarisation of customer reviews in respect of competing products. Results were presented to the user in the form of a histogram, visualising the number of positive and negative reviews pertaining to automatically extracted product aspects for each competing product. The process of opinion polarity classification was not included in the framework, since the paper dealt with semi-structured customer reviews, where comments were already separated into *pros* (positive aspects) and *cons* (negative aspects).

Several other frameworks follow a similar approach to that of Liu et al. [12], grouping sentiment counts by aspects or product features, and either visualising these results or displaying excerpts from reviews in these aspect or product groups [13,14]. Ren and Hong [15] generated similar visualisations according to topics, based on topic-based sentiment analysis using *latent Dirichlet allocation* (LDA). Some frameworks included additional steps in an attempt to render the output more informative for the user. De Lucena [16], for example, generated meaningful textual summaries which transformed the mentioned features and sentiment polarities into recommendations for action.

Some frameworks go beyond this aspect-based summary to include other variables. Yussupova et al. [17] and Wu et al. [18], for example, included graphs visualising sentiment trends over time. Bank [19], furthermore, allowed the user to track certain customer satisfaction indices over time, as well as visualise common terms occurring in the corpus. Ganesan [20] incorporated powerful search functionalities that allow the user to retrieve sentiment summaries of only those reviews that match certain search criteria (*e.g.* reviews of hotels that are a certain distance away from a landmark and mention certain key phrases). Kasper and Vela [14], as well as Yaakub et al. [21], incorporated structured data on reviewers' demographics by displaying summary statistics of these demographics and providing summary reports for specific groups of customers. No frameworks were found, however, that explicitly facilitate the exploration of the relationship between *structured* variables and the content or opinion polarities of the *unstructured* data that are typically analysed. The only examples of frameworks facilitating a targeted analysis of model results are that of James et al. [22], who analysed which latent topics were the best predictors of the ratings of medical experiences by patients, and that of Yussupova et al. [17], who similarly analysed which features mentioned in reviews were indicative of the overall review sentiment using a decision tree analysis. In these examples, however, the relationships between features extracted from unstructured reviews were analysed, but they did not make use of any external data.

Only very few frameworks have attempted to address the problem of a lack of generalisability of frameworks. Typically, this has been done by designing frameworks in a modular, extendible manner so that they may be adapted to another problem setting or by incorporating several different models into the framework instead of a single, specific model.

Hsueh [23] proposed a generic software framework for opinion mining using object-oriented, modular constructs. Due to the design approach, this framework is flexible in respect of change and extension. The framework, however, provides little guidance on actually constructing the sentiment analysis models and does not include functionality for results summarisation or presentation. Lloret et al. [24] proposed a unified framework encompassing information retrieval, opinion classification and opinion summarisation. This framework is also modular in nature and can therefore be extended beyond the functions it was initially designed to facilitate (which include only lexicon-based sentiment modelling approaches). Lloret et al. [24] mentioned that each component of their framework may be tuned by choosing among different parameters and options. No guidance is, however, given in the framework as to how such a tuning procedure should be conducted. Isah et al. [25] proposed the only framework which included a generic description of both machine learning and lexicon-based approaches to sentiment analysis. During the application of their framework, however, only one of the approaches was selected and no comparison was carried out of the relative performance of the different approaches. The hybrid classification framework by Liu and Lee [26], on the other hand, is capable of comparing the performance of three different modelling approaches, but does not include any reporting features for summarising the results of these models.

Schouten et al. [27] proposed the *Heracles* framework, which facilitates the flexible development and comparison of text mining algorithms (including sentiment classification algorithms) and incorporates various preprocessing activities as well as a structured model comparison accommodating both machine learning and lexicon-based algorithms. Important components of the sentiment modelling phase, such as feature engineering and hyperparameter tuning are, however, not addressed, since automated packages (*e.g. Weka*) are used for this purpose. Furthermore, an analysis of model results beyond test performance is not facilitated.

In summary, the existing literature on frameworks for evaluating opinionated data is primarily concerned with providing frameworks for deploying single, specific and typically lexicon-based models by following predefined sequences of preprocessing activities and employing a predetermined set of features. Even in rare cases where a comparison of several models is facilitated, little guidance is provided within the framework in respect of model development. Furthermore, although research has certainly been conducted on the use of machine learning models for sentiment analysis, the machine learning approach is rarely included in *holistic* frameworks for sentiment analysis (incorporating text preprocessing, sentiment analysis and summarisation) such as the one proposed in this paper. According to Kumar et al. [6], the most time-consuming activity in any process that makes use of machine learning is selecting the machine learning algorithm, generating and selecting suitable features from the data and tuning the parameters and hyperparameters of the model. To the authors' best knowledge, there are no frameworks in the literature for sentiment analysis or opinion mining that address this activity when machine learning models are incorporated. Moreover, existing frameworks do not explicitly facilitate an analysis of the relationship between the results of the sentiment analysis model and additional structured data that may be available to the user. Finally, most currently available frameworks do not take into consideration the problem that the effectiveness of certain preprocessing activities and associated algorithm choices may depend on the data set in question.

One possible reason for these shortcomings in the literature is the objective of most of these frameworks in terms of producing fully automated sentiment analysis systems. In this paper, however, a decision support approach is adopted where the objective is not to *automate* the process of evaluating opinionated data, but to *facilitate* this process. The framework proposed in this paper is distinguished from those in the literature largely by the following characteristics:

1. The framework is *interactive* with a focus on facilitating rather than automating the evaluation of opinionated data by a user.
2. Rather than incorporating a specific model into the framework, the user is guided through the *model development* process, with a particular focus on the machine learning approach, where algorithm selection, parameter and hyperparameter tuning, as well as feature selection, are required.
3. Instead of merely presenting model results, the framework facilitates an *exploratory analysis* of these results, including an investigation of the relationship between sentiment and data attributes from supplementary, structured data sources.

Furthermore, the framework is designed with the objectives of generalisability and flexibility, as is elucidated in the following sections, further supporting its applicability to various problem domains.

3. The proposed sentiment analysis framework

In this section, our sentiment analysis framework is presented in detail. A generic data science paradigm is first described, within which the proposed framework is set. Subsequently, a high-level overview of the proposed framework is given, and this is followed by a detailed description of its primary components using *data flow diagrams* (DFDs).

3.1. A generic data science paradigm

In order to develop a framework that is generic and adaptable to various problem settings, our framework was designed to function within a generic data science paradigm and such that it is *modular* in nature. Any modules incorporated into the framework may thus be exchanged, modified or deleted in accordance with new findings in the literature and the objectives of the analysis in question without disrupting the correct functioning of other modules of the framework. Additional modules may also be added to substitute for existing modules of the framework. By deriving the framework from a data science paradigm based on widely used sources, it is ensured that all stages necessary for extracting knowledge and insights from data are included in the framework, thereby preserving its generic nature.

A high-level overview of the proposed generic data science paradigm is shown in Fig. 1. It comprises three primary components, namely a GUI, which facilitates communication with the user, a database, in which relevant data are stored, and a central functional component, which is partitioned into three subcomponents, namely a *processing* component, a *modelling* component and an *analysis* component. This reflects the three primary components of a DSS (the database component, the model component and the user interface) [28,29], as well as the three stages into which the model base of a *model-driven* DSS are typically partitioned (the formulation stage, the solution stage and the analysis stage) [28]. This paradigm also exhibits similarities with the typical data science process described by O'Neill and Schutt [30], as is illustrated next.

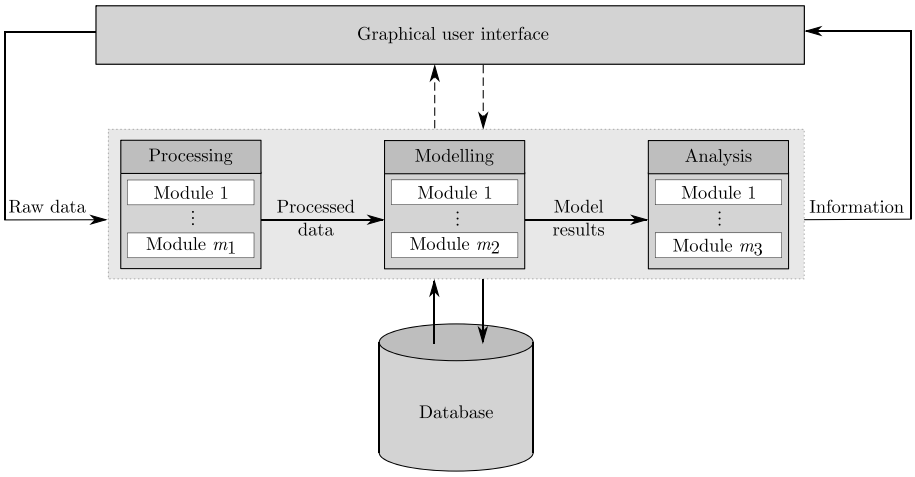


Fig. 1. A high-level abstraction of a generic data science paradigm.

Raw data are provided by the user *via* the GUI. The *data collection* step of the data science process is therefore assumed to have already been completed by the user. These data are then processed by m_1 functionally coherent modules residing in the processing component, which may perform several tasks, including data cleaning, data transformations (*e.g.* of unstructured data into structured data) and normalisation. This component thus incorporates the data processing and data cleaning steps of the data science process.

Processed data are then provided as input to the modelling component. The m_2 modules residing in this component perform tasks related to the *model building and evaluation* step in the data science process. The nature of the models included in this component is determined by the objectives of the particular study and may include classification models, regression models and traditional forecasting models. Of the, possibly several, models evaluated and compared during this process, the selected model may then be applied to new data to obtain a classification for each new observation.

In order to extract valuable information and insight from these results, however, a subsequent analysis of the results is necessary. This may take the form of a simple visualisation of results or a more complex analysis aimed at identifying patterns explaining the model results. Such functions are performed by the m_3 modules residing in the analysis component, as shown in Fig. 1. The resulting information is then communicated to the user by means of the GUI, reflecting the *communication of results* step of the data science process.

Throughout the execution of the modules in the processing, modelling and analysis components, data may be stored in and retrieved from a common database in order to reduce the interdependence between the modules in each of these components. The processed data may, for example, flow from the processing component to the database from where they may be retrieved by the modules in the modelling component. The data flows in Fig. 1 are shown to flow directly between model components merely for the sake of intuitive understanding. Furthermore, user input may be elicited and feedback given to the user at any stage of the process. The modules need not be executed sequentially, and can also be implemented in an iterative fashion, incorporating user input and allowing the user to alter this input after having observed the effects of changes to the input. In this manner, users may perform, to some extent, the *exploratory data analysis* step of the data science process. User interaction is, however, not required, as indicated by the dashed lines in Fig. 1.

3.2. The ECCO framework

Given the general paradigm in Fig. 1, our proposed sentiment analysis framework can now be introduced. This framework is named ECCO (*Evaluating a Corpus Characterised by Opinion-bearing language*), alluding to the notion that the voices or opinions of the authors of the documents in the corpus *'echo'* throughout the analysis. A high-level overview of the framework is shown schematically in Fig. 2, where the *generic* paradigm of Section 3.1 has been populated with the *domain-specific* modules proposed for evaluating opinionated content. This is reflected in the alteration of the name of the modelling component from *modelling* in general to *sentiment modelling* in particular. Whilst the development of this framework takes place within the realm of sentiment analysis, the framework is not necessarily limited to this domain. The modules in the modelling component may also be configured to detect other aspects of unstructured text data, such as document type or topic. In the remainder of this paper, the focus will nevertheless remain on the modelling of sentiment.

A seemingly small, but notable, difference between the ECCO framework in Fig. 2 and the generic data science paradigm in Fig. 1 is the transformation of the arrows between the GUI and the central processing component from dashed lines to solid lines. User input is therefore necessarily elicited in the ECCO framework during each of the processing stages. As mentioned previously, the performance of a classification algorithm, particularly in the machine learning domain, depends strongly on the model variant and feature set employed, as well as on the data set in respect of which the model is trained. Fully specifying the model and features employed in the framework would, therefore, require the delineation of various assumptions about the data sets for which the framework may be used. This is also true for some of the algorithms used to process the data and to analyse model results. The ECCO framework therefore *facilitates* the evaluation of unstructured, opinionated data by a user, rather than *automating* this process.

The only assumption imposed on the input data for the framework is that they are of an unstructured (free-form) text format and that a significant proportion of the data constitute sentiment-bearing content. These data, referred to as *documents*, may, furthermore, be enhanced with *supplementary data* which describe the document in more detail. These data, if present, are assumed to be presented in a relational form (*i.e.* stored in one or several tables which are linked to the documents and to one another by means of primary keys).

In the remainder of this section, each of the subcomponents of the ECCO framework is discussed in detail. Central to this discussion are the DFDs of each of the subcomponents, which describe the processes involved in each subcomponent, as well as the data flows between these processes, where required.

3.3. The processing component

The processing component comprises five modules numbered 1.0 to 5.0 as shown in the *level-one* DFD in Fig. 3. As mentioned previously, the input data are assumed to be partitioned into two data sets — an unstructured component (the documents) and an optional structured component (the supplementary data). Modules 2.0 − 4.0 are designed to accommodate the unstructured component, whilst Module 5.0 is configured for the structured component. Module 1.0, on the other hand, is applied to both the structured and unstructured components of the data.

The first step in the processing component is the categorisation of data attributes. This entails specifying the location of certain data, such as the document text and their associated class labels, if present, as well as identifying the data types of attributes in the structured data component, so that subsequent modules can easily distinguish between qualitative and quantitative attributes. Furthermore, certain attributes may be designated as '*distinct*' attributes, indicating, for example, a location or date, that are treated differently in the analysis component later on.

The *raw* (unprocessed) documents are then *tokenised* or split into their constituent parts. The resulting sets of tokens are subsequently *filtered* in order to remove irrelevant or uninformative tokens from the data. Finally, the remaining tokens are *normalised* to their standard or root form in order to reduce redundancy in the data. These three processes are executed sequentially according to the desired settings selected by the user. These settings may include algorithms and parameter values to be used in any of the modules, and may also allow for the exclusion of certain processing steps. After the execution of the first three modules, the user is given feedback as to how these processing steps have affected the input data in the form of an *effect summary*. This summary should allow the user to gauge how effective the processing methods have been in transforming the data into a form that is useful for later analysis. Subsequently, the user is able to adjust the chosen settings and repeat the process until a satisfactory outcome is achieved. This iterative procedure is indicated by the curved arrows in Fig. 3.

The structured supplementary data are subjected to a data cleaning process, the aim of which is to remove errors and inconsistencies in the data. The processed documents and supplementary data are then stored in a central database where they can be accessed by other modules of the framework.

A more detailed representation of the processing component is shown in its level-two DFD in Fig. 4. Here the categorisation of data

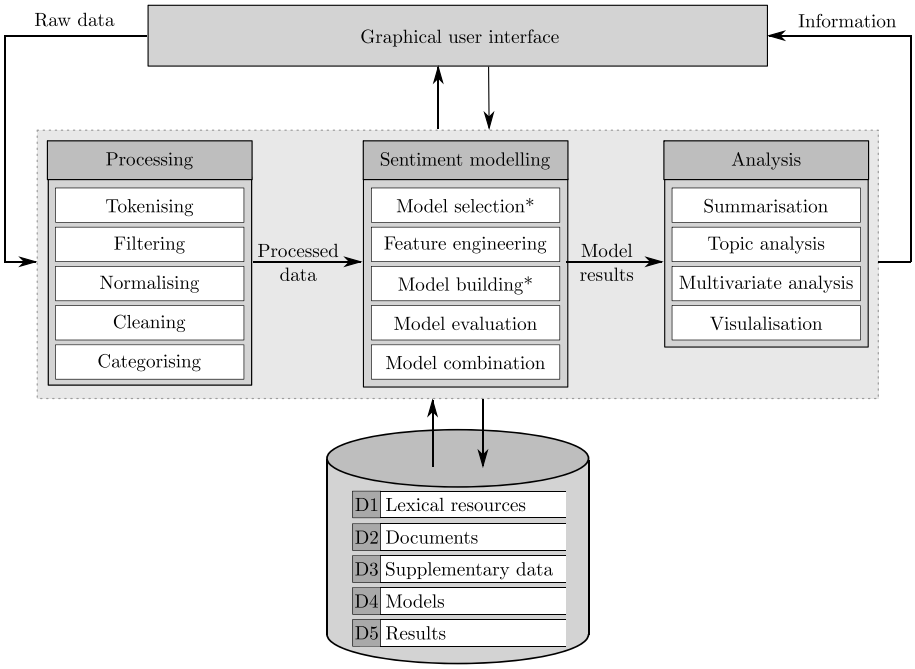


Fig. 2. A high-level overview of the proposed ECCO framework. Modules marked with an asterisk are summary modules which represent one module for the machine learning approach and one module for the lexicon-based approach to sentiment modelling.

attributes (Module 1.0) is partitioned into three child processes 1.1 − 1.3. The first child process entails tagging distinct attributes that have a special function within the framework, as previously explained. This step also includes distinguishing documents from their class labels. Secondly, structured data attributes are categorised as qualitative or quantitative based on their data type. Lastly, the fields constituting the primary keys by which the supplementary data and the documents are linked are identified.

Module 2.0 is not exploded into its child processes, since it represents a simple process. Module 3.0, on the other hand, is partitioned into two child processes which apply three filters: One for uninformative words, one for uninformative punctuation and one for uninformative numbers. The removal of common stopwords is a widely used technique for reducing the feature space. This is typically achieved by comparing the tokens resulting from the tokenisation step with a pre-compiled list of stop words and removing these from the input data. Often, these lists contain punctuation marks that are also typically removed from the data set. Most of these lists were, however, compiled for general text mining purposes and may therefore cause the exclusion of words important for sentiment classification, such as negation words (*e.g.* '*not*' and '*no*'), intensifiers (*e.g.* '*too*' and '*very*') or punctuation indicative of sentiment (*e.g.* '*!!!*'). The user should, therefore, be able to customise such a list in order to exclude such cases. Lastly, uninformative numbers are removed from the data set. These include reviewer's phone numbers or amounts of currency, which are rarely informative, since any model would identify these as distinct features and therefore not be able to *learn from experience*. Such numbers may be grouped into a single feature indicating the presence of some numerical value, or may be removed entirely. In this case too, however, the user should be able to exclude certain numbers from the process. If the authors of the documents to be analysed were prompted to include, for example, a rating out of five or a monetary value as a gauge for their level of satisfaction, such numbers should be treated as separate features, since it can be assumed that several observations will exhibit these features.

The normalisation of documents (Module 4.0) is partitioned into three child processes. First, each of the tokens is converted to a common

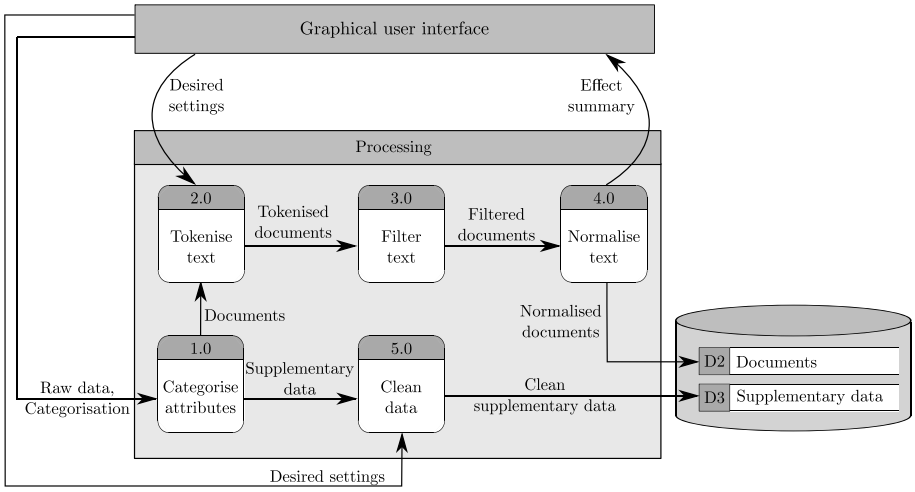


Fig. 3. Level-one DFD of the processing component.

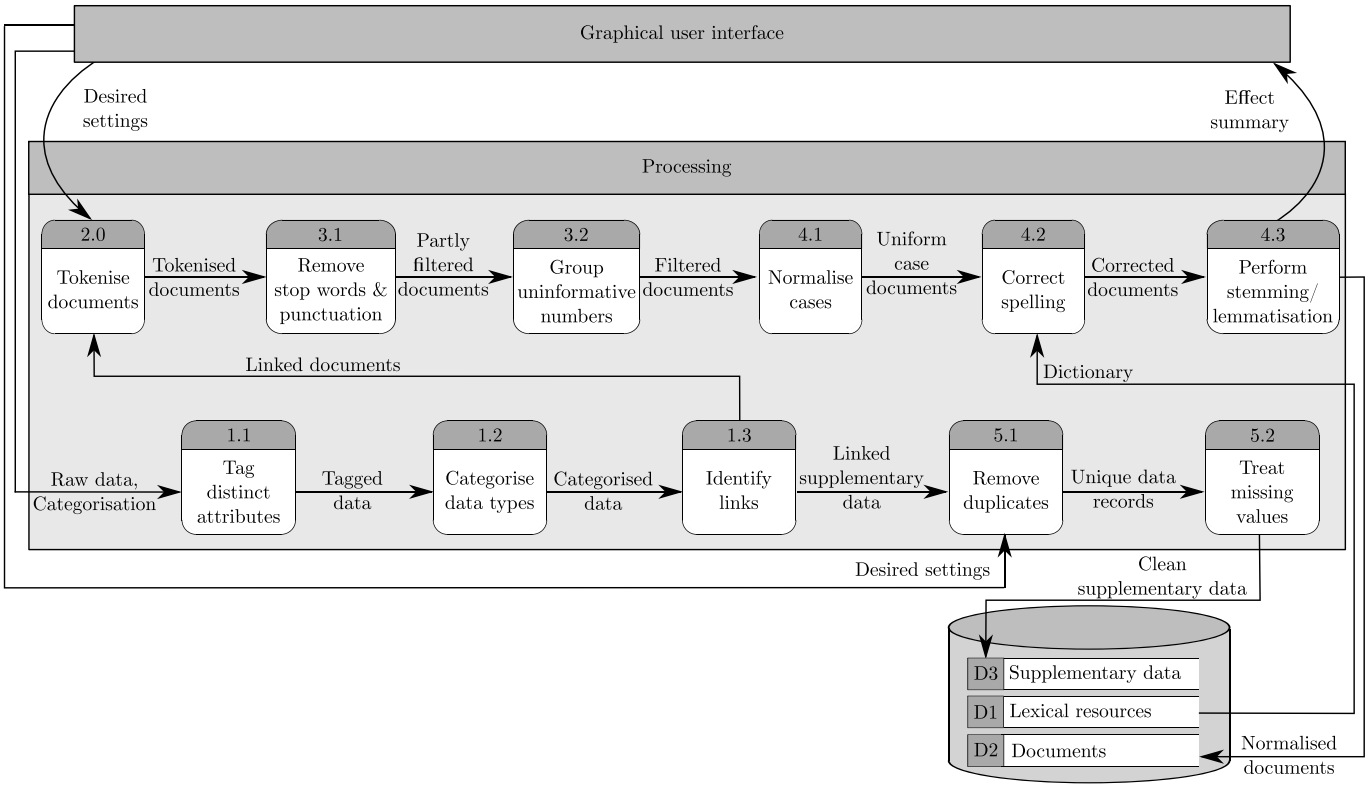


Fig. 4. Level-two DFD of the processing component.

case (either all uppercase or lowercase letters). Then the tokens are checked for correct spelling and, where applicable, spelling correction is performed. This is typically achieved by comparing each word with entries in a standard dictionary and, if the word is not contained in the dictionary, successively removing, altering or exchanging letters in the word until it does match one of the dictionary terms. Care has to be taken, however, when domain-specific words or jargon are encountered that are not contained in the dictionary, but are nevertheless spelt correctly. An investigation of word usage in the corpus to be analysed can provide guidance in such cases. After spelling correction has been performed, each of the tokens is transformed into a root form by means of a stemming or lemmatisation algorithm, such as Porter's Stemming Algorithm or the Lancaster Stemming Algorithm.

Finally, the data cleaning process in Module 5 is represented in the ECCO framework by two child processes, namely the removal of duplicate entries and the treatment of missing values. Typical data cleaning activities also include handling error values and outliers. Without prior knowledge of the contents of the data, however, these activities are difficult to perform. A value of 1000 kg for a person's weight, for example, surely constitutes an outlier value indicating an error that should be corrected. An unusually high credit balance of a bank customer may, however, be helpful to identify customers that are at risk or in need of assistance, and should not be reduced to a more common value. For this reason, these activities are excluded from the framework. After duplicate entries have been removed, missing entries are identified and treated according to the user's desired settings. Typically, this entails either imputing missing values with logical values or removing observations or attributes exhibiting a large proportion of missing entries.

3.4. The modelling component

The sentiment modelling component is shown schematically in Fig. 5. It comprises seven modules, numbered 6.0 − 12.0 in continuation of the numbering of the processing component. Modules 6.0 − 8.0 are employed to develop machine learning models for sentiment analysis, whilst modules 9.0 and 10.0 facilitate the development of lexicon-based models. Modules 11.0 and 12.0 are aimed at evaluating and combining the models generated by the preceding modules.

The major feedback loop between the user and the functional modelling component is represented by the curved arrows flowing to and from the GUI in the figure, as before. The user selects a set of machine learning (abbreviated as ML in the figure) and lexicon-based algorithms to be configured from a given set. Furthermore, the user selects the desired settings employed to construct sets of features to be used in combination with the machine learning algorithms. The selected algorithms are then used to generate deployable models for classifying sentiment. For the machine learning algorithms, this entails finding good parameters by training the models in respect of a training data set and making adjustments to hyperparameters in an attempt to improve generalisability. Lexicon-based algorithms, on the other hand, require the creation of a suitable sentiment lexicon. The deployable models are then evaluated in respect of a test data set according to the user's evaluation criteria. Typically, these criteria take the form of *accuracy*, the area under the receiver operating characteristic curve (*ROC-AUC*), *precision*, *recall* or the *F1-score* [31], which may be applied to any classification model. Feedback on the models' performance is provided to the user by means of a *performance summary*, on the basis of which the user may change the input settings in an attempt to achieve the desired outcome more closely.

After having observed the performance comparison summary of the models, the user is also able to combine several models in order to form an *ensemble*. A similar performance summary of this ensemble is then relayed to the user. The model selection and ensemble selection process is repeated until a satisfactory level of performance has been achieved. Then, the user selects the models that are to be stored in the central database for later use.

A more detailed representation of the modelling component is shown in its level-two DFD in Fig. 6. In this diagram, the process of feature engineering for machine learning (Module 7.0) is exploded into

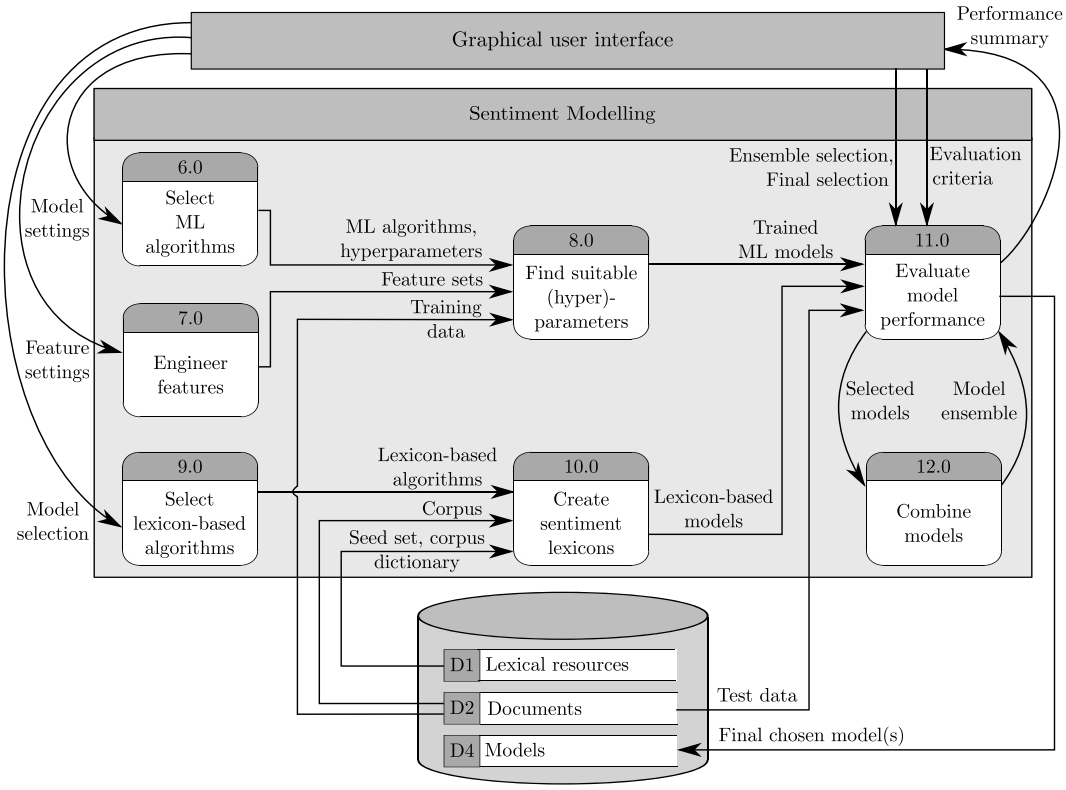


Fig. 5. Level-one DFD of the sentiment modelling component.

two child processes, namely feature generation and feature selection. Feature generation, in the context of sentiment analysis, refers to the process of text vectorisation. This may include extracting term-based features, linguistic features or topic-oriented features from free-form text in order to represent this text as a numerical vector. A bag-of-words model may, for example, be applied in conjunction with a specified range of *n*-grams to be extracted and an associated document model (*i.e.* the presence-based Bernoulli model, the frequency-based multinomial model or the weighted TF-IDF scheme). Feature selection refers to reducing the resulting feature space by reducing the size of the dictionary, applying feature transformations or training a model to learn a lower-dimensional representation (*word embeddings*).

Process 8.0 is exploded into two child processes, aimed at estimating the parameters of the model and tuning the model hyperparameters, respectively. As is indicated by the curved arrows in the figure, these processes are executed in an iterative fashion. For each of the algorithm-feature combinations specified by the user, the model parameters are estimated for a fixed set of hyperparameters in respect of a set of training data. The model performance is then evaluated in respect of a validation data set (a certain proportion of the training data not used for parameter tuning), and the values of the hyperparameters are adjusted accordingly.

The algorithms employed in Process 8.1 typically include optimisation techniques, such as gradient descent or *quasi-Newton* methods, and constitute hyperparameters of a model. Other hyperparameters include, for example, the number of nearest neighbours *k* considered in the *k nearest neighbours* model, the tuning parameter *C* for *support vector machines* (SVM), which controls the bias-variance trade-off in the model, and the number of hidden layers included in an *artificial neural network* [32]. The hyperparameter tuning processes in Module 8.2 may include a manual search, a grid search, a random search or an *'intelligent'* search (using, for example, *metaheuristics*). If a manual search is employed, user input is elicited during every iteration, as indicated by the dashed arrows flowing between Module 8.2 and the GUI. If, however, a grid or random search is performed, a set or range of hyperparameter values to explore is specified by the user during algorithm selection in Module 6.0. Finally, if an intelligent search is carried out, user input is rarely required. A desired level of automation may therefore be achieved using the ECCO framework.

Finally, Process 10.0 is exploded into two child processes used to create sentiment lexicons. The lexicon-based algorithms selected in Process 9.0 are combined with an initial *seed set* of sentiment terms and their associated polarities. This set, which constitutes the initial sentiment lexicon, is then used to identify related terms and estimate their predicted polarities. This may be achieved by employing a dictionary-based method or a corpus-based method. If a dictionary-based approach is used, the seed set is expanded by its synonyms (which are assigned the same sentiment polarity) and antonyms (which are assigned the opposite sentiment polarity). These related terms are then integrated into the existing sentiment lexicon and the procedure is repeated until no other synonyms or antonyms are found. If a corpus-based approach is adopted, related terms are extracted *via* statistical co-occurrence patterns of the seed words with other terms in a given corpus. This corpus may either be generic, or may be supplied by the user and thus constitute a domain-specific corpus. In the latter case, the same documents used for training machine learning models are employed. Otherwise, the use of lexicon-based algorithms in the ECCO framework presupposes the availability of suitable lexical resources that are stored in the database prior to the analysis. The final lexicon-based model then includes the generated sentiment lexicon and the algorithm or rules employed to predict the polarity of a document using this lexicon (*e.g.* comparing the number of positive and negative adjectives from the lexicon that appear in the document).

As mentioned in Section 2, developing a suitable model is one of the most time-consuming activities in the machine learning process [6]. This activity involves three tasks, namely feature engineering, algorithm selection and (hyper) parameter tuning. Kumar et al. [6] refer to this as the *model selection triple* (MST) and further define the iterative procedure adopted to select a suitable MST as comprising three steps. The first is *steering*, where the user decides on one MST and specifies it. The second step is *execution* and entails the building and evaluation of the model by a computerised system. Finally, the user assesses these

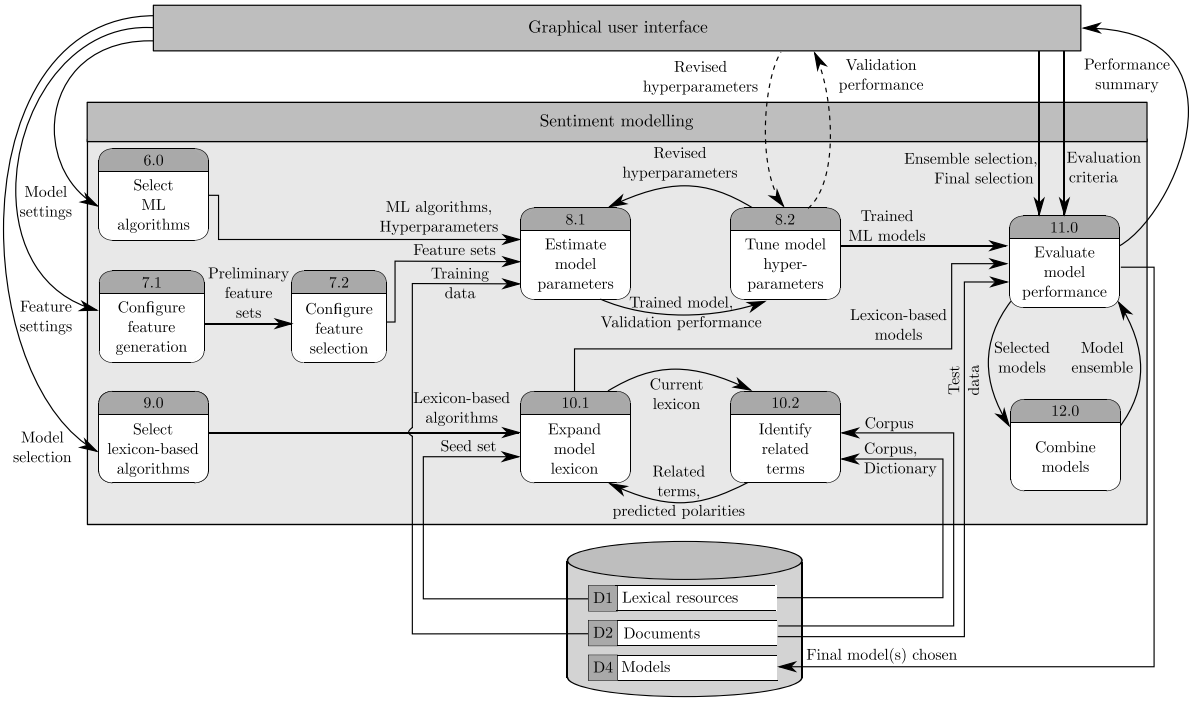


Fig. 6. Level-two DFD of the sentiment modelling component.

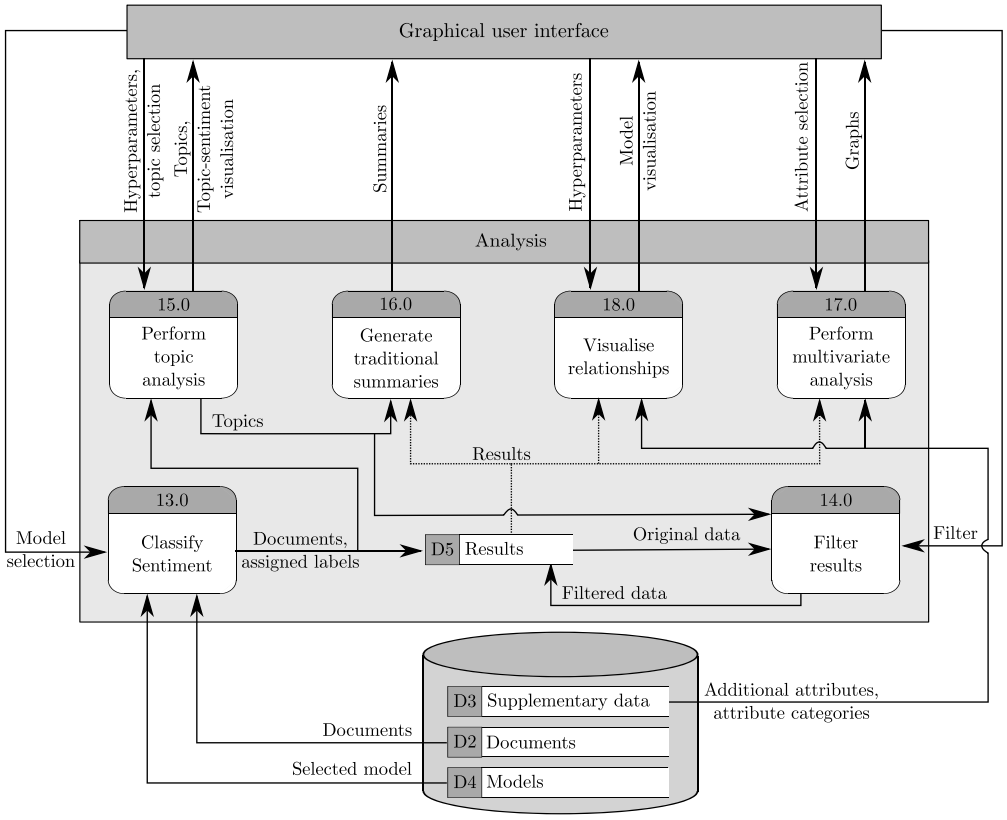


Fig. 7. Level-one DFD of the analysis component.

results in the *consumption* step and decides whether the process will be terminated or whether another MST will be tested during the next iteration. In a typical system, only one MST can be tested per iteration. Due to the importance of this process and its expensive nature, this leaves considerable room for improvement. In the ECCO framework, several efforts are made to address this problem.

In modules 6.0 and 7.0, the user is afforded the ability to specify several machine learning algorithms and several different settings for feature engineering in order to test various combinations during a single iteration. The framework thereby attempts to improve the efficiency of the steering step of the model selection process. Modules 8.1 and 8.2 facilitate the parameter tuning process. As mentioned earlier, this process may be

automated, whilst still taking the expertise of the user into account, if a random or grid search is employed in Module 8.2, and can further be improved by means of intelligent hyperparameter search algorithms. By transferring the computational load from the user to the computer, the execution step of the model selection process is simplified. Furthermore, allowing the user to compare several MSTs simultaneously can successfully guide the steering step for the next iteration. Finally, the consumption step is also enhanced by issuing the user with a meaningful performance summary of previously tested models for easy and rapid comparison purposes.

3.5. The analysis component

The working of the final component of the ECCO framework, the analysis component, is illustrated in Fig. 7. It comprises six modules, numbered 13.0 − 18.0, which are designed to facilitate the analysis of the results returned by the sentiment classification model selected by the user. This component thus aims to execute and enrich the process of sentiment summarisation. Module 13.0 facilitates the deployment of the selected model, taking as input the unlabelled documents that have been processed by modules 1.0 − 5.0 and the model that was developed in modules 6.0 − 12.0, and returning the sentiment class assigned to each document by the model. These *results* are stored in a local database for later use by other modules in the analysis component. Modules 15.0 and 16.0 are aimed at analysing the unstructured, textual component of the data, whilst modules 17.0 and 18.0 facilitate an investigation of the relationship between the structured, supplementary data attributes and the documents' sentiment classification. Finally, Module 14.0 enables the user to filter the results of the analyses in modules 16.0 − 18.0 according to the outputs of Module 15.0.

In Module 15.0, the corpus is analysed in terms of its underlying topics or frequently occurring key words. As is shown in the level-two DFD in Fig. 7, this entails first performing topic analysis and then visualising the sentiment exhibited for each of the identified topics. Topics may be extracted using dependency trees, noun phrase extraction or topic modelling techniques such as LDA. In the latter case, model hyperparameters may have to be specified by the user. Additionally, topics may also be identified manually by the user. The visualisation of selected sentiment-topic relationships may take the form of a histogram or alternative representations such as rose plots or low-dimensional representations of word distributions.

Module 16 is responsible for generating the summaries typically associated with text mining for the current selection of data. Such *traditional* summaries take the form of textual summaries, which are generated using either template instantiation or passage extraction, or visual summaries, either of which is typically accompanied by sample documents for a clearer overview. These summaries may also make use of the topics discovered by Module 15.0 in order to give a more detailed overview.

Modules 15.0 and 16.0 provide the user with insight into the typical contents of the documents in the corpus and the possible underlying topic distribution, as well as an overview of the sentiment distribution in the entire corpus and for each identified topic. If supplementary data are available, it is also desirable to analyse the effect of these additional attributes on the sentiment distribution in order to further deepen the insight gained in respect of the corpus.

This analysis is facilitated by Module 17.0 in the form of visual summaries. The user is afforded the ability to select a subset of attributes of the supplementary data, and a visualisation is then returned, depicting the relationship between these attributes and the relevant sentiment categories. As is shown in the exploded view in Fig. 8, the attributes are first classified by their attribute type, which was categorised by the user during the processing procedure in Module 1.0. This allows for the creation of *type-specific* visualisations, such as histograms for qualitative attributes, box plots or scatter plots for quantitative variables, and geographical maps for location data. By exploring the visualisations generated for various attributes, the user may identify trends or relationships that may exist in the data.

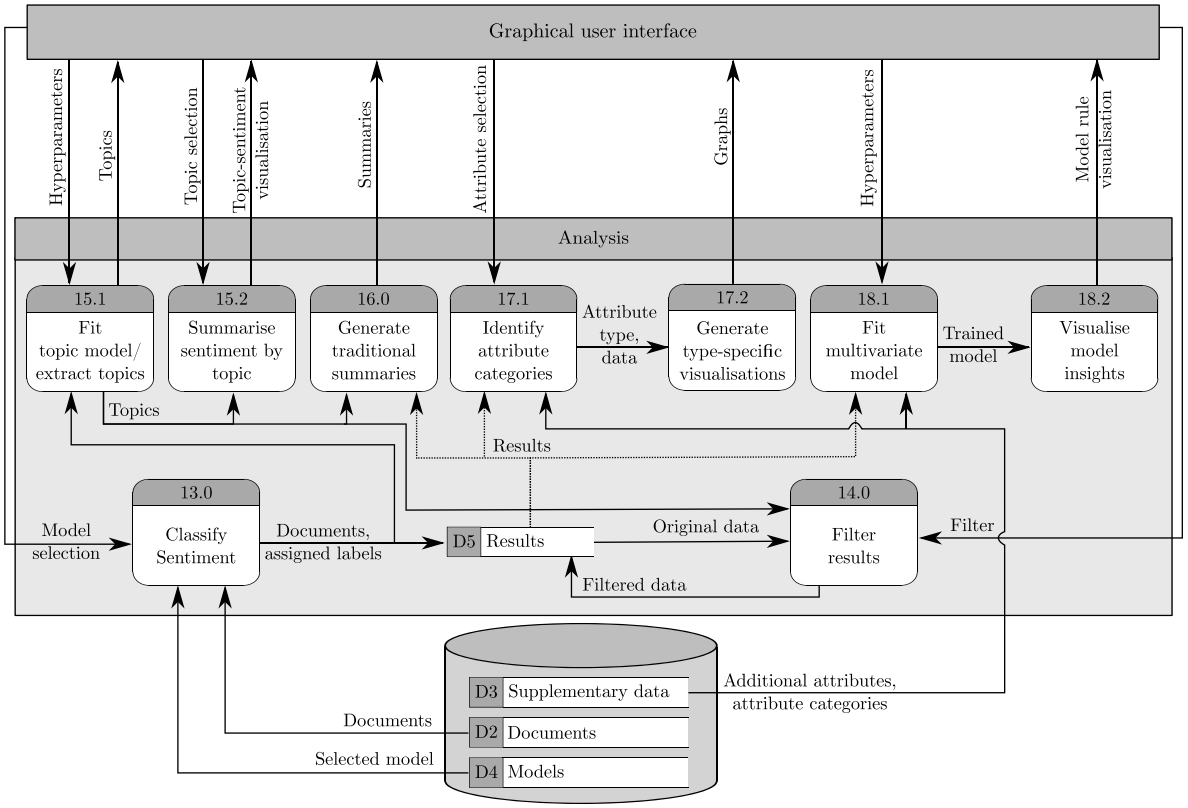


Fig. 8. Level-two DFD of the analysis component.

Table 1
A summary of the case study data sets.

Data set	Domain	Problem type	Supplementary data
PL04	Movie reviews	Binary	–
Yelp	Business reviews	Binary	Business & user data
Twitter	Social media	Ternary	–
SMS	Banking survey	Ternary	Branch & customer data

In Module 18.0, a different approach is taken in the form of a *multivariate analysis*. More specifically, the approach proposed in the ECCO framework is to fit a statistical model to the data which seeks to model the relationship between the supplementary data attributes and the sentiment categories. Many such models may subsequently be analysed or visualised in order to reveal the *'rules'* or hidden relationships that have been discovered by the model. A classification tree may, for example, be visualised by means of an hierarchical tree diagram. Similarly, the most *predictive* features and the most likely sentiment classification arising from various values of these features may be identified upon examining a trained logistic regression model or maximum entropy model. Adopting this approach, the user may discover hidden insights in the data that were not detectable *via* the visualisations in Module 17.0. As shown in Fig. 8, the user may be required to specify hyperparameter values for these statistical models. As with most processes in the ECCO framework, these values may be adjusted iteratively until a desired level of performance or granularity is achieved by the model.

By executing modules 14.0 − 17.0 concurrently and iteratively varying the input parameters to these modules, the user is afforded the opportunity to explore the corpus with respect to the contents of the documents, the overall sentiment distribution and the relationship between sentiment and content, as well as the relationship between additional structured data and sentiment. In this manner, the three components of the ECCO framework facilitate the extraction of actionable insight from raw data.

4. Case study experiments

In order to demonstrate the potential of the proposed ECCO framework, an instantiation of the framework was implemented in the form of a computer program and applied to four case studies from four different domains. To this end, *generic* components of the framework were populated with *specific* algorithms and elements in order to illustrate the functionality of the framework. The resulting program, called the *ECCO system*, is an example of the type of DSS that may be developed using the ECCO framework as a blueprint.

The system was developed in Python 3.7. Data were stored, where applicable, in DataFrame constructs native to the Python library Pandas or arrays native to the Python library Numpy. Three different application frameworks were employed during the development of the ECCO system. The first is Qt, which was used to develop the user interface that houses the processing component and the sentiment modelling component of the ECCO framework. The Tensorboard framework was used to visualise the performance of deep learning models during the hyperparameter tuning process. Finally, the Dash web application framework was used to implement the analysis component of the ECCO framework, which executed by means of a user interface hosted on the user's web browser. Machine learning models were implemented by means of the Scikit-learn and Keras libraries. Any additionally employed packages are described later along with their functions.

In this section, a brief description of the case study data is first given. This is followed by illustrative examples of case study experiments performed on the data, which showcase the working of the framework. These examples are partitioned according to the three primary components of the ECCO framework, related to (1) processing steps performed on the data, (2) the model development process and (3) the analysis and synthesis of results returned by a selected model in conjunction with structured data sources.

4.1. Case study data

In order to showcase the general applicability of the ECCO framework, case study data sets were selected from four vastly different domains, as summarised in Table 1. Three of these are well-known data sets from the sentiment analysis literature that may serve as performance benchmarks for the modelling component.

4.1.1. PL04

This data set[1] contains 1,000 *positive* and 1,000 *negative* processed movie reviews written before 2002. The data set was originally published by Pang and Lee [33] and has been used as a benchmark data set in many subsequent studies. In order to generate the data set, the authors selected only those reviews which contained some rating information (*e.g.* "*four out of five stars*"). Ratings of at least 3.5/5, 3/4 or letter grade *B* were considered *positive* whilst ratings of at most 2/5, 1.5/4 or letter grade *C*- were considered *negative*. Such rating information was subsequently removed from the review. Furthermore, a limit of twenty reviews per author per category was imposed.

4.1.2. Yelp

The Yelp Research data set[2] contains data related to business reviews from the popular review site *Yelp*, including the review text, its star rating and additional data on the establishment subject to the review (*e.g.* name, location, categories, average star rating received), as well as on the users who submitted the review (*e.g.* review count, average star rating given). In order to transform this data set into one suitable for classification, star ratings of 1 or 2 were classified as *negative*, ratings of 4 or 5 were classified as *positive*, and ratings of 3 were discarded, following the same process as that adopted by Salinca [34]. Due to limited computing power, the size of the data set was reduced by restricting the time frame to the year 2018 and randomly selecting 10% of the reviews on each of the most reviewed establishments (those which were mentioned more than 1,000 times in the 2018 subset).

4.1.3. Twitter

This data set[3] was published along with the popular lexicon-based sentiment analysis model, *Vader* [35], and comprises 4,200 randomly selected microblogs from the social media site *Twitter* along with a sentiment score assigned to each review by a team of human annotators, as described in the accompanying paper. As was prescribed by the authors, the ratings were first normalised to the interval $[-1,1]$ and then transformed into sentiment categories using the threshold values of -0.05 and $+0.05$, where messages with normalised ratings below the negative threshold were classified as *negative*, those with normalised ratings above the positive threshold were classified as *positive*, and the remainder were classified as *neutral*.

4.1.4. SMS

These data pertain to customer feedback on the services of a South African retail bank.[4] More specifically, the available data comprised 10,636 unstructured customer reviews in the form of text messages, as well as structured data attributes describing the customer and the branch visit associated with the review. Furthermore, the data included a numerical rating of 1, 2 or 3 submitted by the customer (where 3 denotes the worst possible rating). A subset of 2,500 review messages

[1] Available at http://www.cs.cornell.edu/people/pabo/movie-review-data as "*polarity data set v2.0*."
[2] Available at https://www.yelp.com/dataset.
[3] Available at https://github.com/cjhutto/vaderSentiment.
[4] The raw data are protected by a non-disclosure agreement.

were labelled as *positive*, *negative* or *neutral* in sentiment by means of a wisdom of the crowd approach, whereby each message was reviewed by at least two annotators, adding additional annotators until a consensus was achieved by majority vote. The final inter-annotator agreement was 93.92%. Although it does not constitute a benchmark, unlike the previous three data sets, this data set was deemed valuable to the study since it presents the opportunity to evaluate the framework in the context of raw, unprocessed data originating from an African context, thus exhibiting linguistic nuances not often found in benchmark data sets and sentiment lexicons. Furthermore, it represents a real-world scenario in which a business aims to make decisions based on sentiment information, and is therefore suitable for assessing the utility of the ECCO framework in aiding such decision-making processes.

Some metadata on the case study data sets are shown in Table 2. As is evident from this table, the data sets are not only diverse in domain, but also in class distribution, document length and vocabulary size $|V|$, defined as the number of unique *tokens*,[5] commonly referred to as *types*, present in the original data set.

4.2. Processing the data

The *effect summary* returned to the user during the preprocessing stage, as shown in Fig. 3, was implemented in the form of a word cloud representation of the most frequent terms in the corpus, as well as a summary of the number of unique tokens present in the data before and after preprocessing. In this manner, the user can effectively gauge the changes made to the data, both in the general and in the specific case. Furthermore, the classification accuracy (the proportion of correctly classified instances) of the smallest, computationally least expensive model considered in this study, namely the naïve Bayes model, was evaluated for various preprocessing settings in order to gauge the relative effect of preprocessing on model performance. These values are shown in Table 3 for all four data sets. Selected configurations are highlighted in bold.

At first, no preprocessing was applied. In the second experiment, stopword and punctuation removal, the grouping of tokens with primarily numerical characters into a single token *_num* and case normalisation were performed. Stopword removal was achieved using the standard stopword list form the NLTK Python library [36]. Since the NLTK stop word list is not tailored to a sentiment analysis problem, however, certain words were excluded from this removal list. These include negation words, intensifiers such as *too* and *very*, and the exclamation mark. For the SMS data set, the numbers 1, 2 and 3 were furthermore preserved and not aggregated along with other numerical tokens.[6] In the third experiment, a lemmatisation algorithm using WordNet [37] was applied. Additional normalisation effects that may have been achieved by applying stemming algorithms were forgone in light of the fact that lexicon-based models were also applied in the subsequent modelling step, which require words contained in the English dictionary to function correctly. Finally, in the last experiment, a custom spell checking algorithm was applied, which corrects incorrect words based on contextual use in the corpus by means of the *Aspell* spell checking library [38].

In general, applying the various proposed preprocessing steps led to an increase in classification accuracy and a decrease in vocabulary size. In respect of the Yelp and SMS data sets, for example, superior performance was achieved by the model adopting the last preprocessing configuration, which resulted in a feature space 45% and 60% smaller than the original feature space, respectively. For the PL04 and Twitter data sets, however, the application of the spelling correction algorithm had a negative effect on model accuracy. This is likely because the spell checking library did not recognise several domain-specific expressions, such as *LOL* in the Twitter data set or *run-of-the-mill* in the PL04 data set. For the Twitter data set, the spelling algorithm was therefore not applied. For the PL04 data set, it was decided that no preprocessing would be applied, since the data set was already processed by the original authors. This, furthermore, allowed for a direct comparison of model results with benchmarks during the sentiment modelling stage.

The iterative application of preprocessing steps and utilisation of an effect summary, as per the ECCO framework, allowed for the selection of the best preprocessing configuration for each of the data sets, which was not necessarily the same in each case. Furthermore, this approach offered an initial insight into the data set, in terms of its vocabulary size, baseline performance and word frequency distribution, as is illustrated by the word cloud representations in the case of the SMS data set in Fig. 9.

From this figure, it is evident that stop words such as *and* and *it* are the most frequent words in the unprocessed corpus. In the processed corpus, on the other hand, apart from the retained stop words *no*, *not* and *but*, and the aggregated token *_num*, words typically associated with a financial bank constitute the most frequently observed words in the corpus. The preprocessing component of the ECCO framework was applied to reduce the vocabulary of each corpus *via* filtering and normalisation without compromising model accuracy, and to bring information-bearing words to the forefront. Furthermore, the user was able to customise and selectively apply each of these processes in order to tailor the preprocessing approach to the data set in question.

4.3. Developing a suitable sentiment classification model

One of the objectives of the modelling component of the ECCO framework is to facilitate an improved MST selection process, as described in Section 3.4. By enabling the user to test various model-hyperparameter-feature combinations rapidly during a single iteration, for example, the computational load on the user is reduced.

A user interface was constructed by means of which both a grid search and a manual hyperparameter search can be conducted by the user, as illustrated in Fig. 6. The manual search was facilitated (specifically for neural networks) by a Tensorboard interface. Graphs illustrating the training and validation accuracies and losses of a particular neural network were visualised in order to inform the decision on changes in hyperparameter values. An example of the resulting output is shown in Fig. 10.

The starting point for this particular algorithm was a simple feed-forward neural network with no regularisation applied. Examining Fig. 10, it may be concluded that the network is indeed *learning* during training, since the training loss decreases and the training accuracy increases with the number of epochs. The validation loss, however, diverges as the number of training epochs increases. This indicates that the model may be overfitting the data. In an attempt to curb this effect, some form of regularisation may be applied to the network. Furthermore, since there was no evidence of a stagnation in the training accuracy after ten epochs, the number of training epochs may be increased. These changes to the network structure had the desired effect on the graphs in Fig. 10. The validation loss now followed the decreasing trend of the training loss and the validation accuracy followed the increasing trend of the training accuracy. In a similar manner, other hyperparameters may be tuned using this approach.

For fine-tuning these hyperparameters and for tuning those of less complex models, a grid search function was implemented, which the user can deploy with default or custom parameter grids. Furthermore, several feature representation and sentiment classification models can be selected, trained and tuned simultaneously in order to allow the user to compare several alternatives in respect of the data set directly.

[5] *Tokens* may refer to individual words, numbers, punctuation marks or composites of these, such as '*2nyte!*'.

[6] Prior to submitting free-form text responses, customers were asked to submit a numerical rating of the bank. Since customers were likely to mention these ratings in their subsequent reviews, these numbers were deemed informative.

Table 2
Metadata on the selected data sets.

Data set	# documents	Class distribution			Document length			|V|
		Positive	Negative	Neutral	Mean	Min	Max	
PL04	2,000	50%	50%	–	747.96	18	2,680	46,462
Yelp	3,801	57%	43%	–	90.53	1	1,008	15,879
Twitter	4,200	67%	27%	5%	14.07	1	78	11,468
SMS	2,500	11%	73%	16%	13.74	1	32	12,448

Table 3
The effects of various preprocessing steps (numbering from Fig. 8) on the vocabulary size |*V*|, as well as the classification accuracy of the naïve Bayes model trained in respect of unigram term presence features. Reported accuracy scores reflect the mean test accuracy of two models with randomly generated 80% train-20% test splits. Selected configurations are highlighted in bold.

Preprocessing steps	PL04		Yelp		Twitter		SMS	
	|*V*|	Accuracy	|*V*|	Accuracy	|*V*|	Accuracy	|*V*|	Accuracy
None	**46,462**	**0.7915**	15,879	0.8755	114′68	0.8095	12,448	0.8200
2, 3, 4.1	45,860	0.7900	12,501	0.8930	9173	0.8045	9,404	0.8240
2, 3, 4.1, 4.3	41,486	0.8010	11,349	0.8895	**8462**	**0.8090**	8,948	0.8300
2–4	27,866	0.7565	**8,795**	**0.8915**	6722	0.7875	**4,948**	**0.8460**

(a) Unprocessed corpus (b) Processed corpus

Fig. 9. A word cloud illustrating the most frequent tokens in (a) the original, unprocessed corpus and (b) the final, preprocessed corpus of the SMS data set. The name of the bank associated with the case study was replaced by the token "_bankname" in order to preserve anonymity.

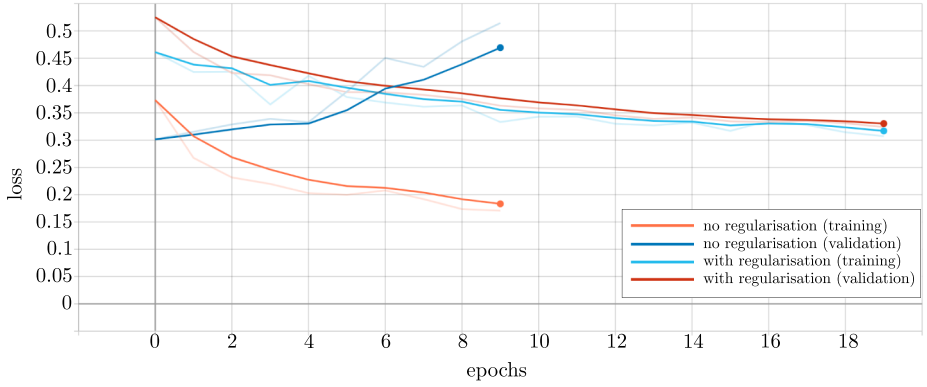


Fig. 10. Smoothed Tensorboard graph for a *feedforward neural network* (ANN) with and without regularisation. The unsmoothed data points are shown as shadow lines.

In these case study examples, ten models were compared, including four lexicon-based models from the literature (*Vader*, *Pattern*, *SentiWordNet* and the *Hu and Liu opinion*) and six machine learning models (*naïve Bayes* (NB), SVM, *logistic regression* (LogReg), an ANN, a *convolutional neural network* (CNN) and a *long short-term memory network* (LSTM)). The former four machine learning models were implemented taking several variants of the bag-of-words representation as input, whereas for the remaining models, word embeddings were trained end-to-end. Along with the desired models and feature representations, the parameter grids for each machine learning model could be specified, allowing the user to compare several model-feature combinations directly with optimised hyperparameter values.

The performance results of each model-feature combination (henceforth referred to as *experiments*) are presented to the user in the form of a *table* widget in the GUI, as shown in Fig. 11. The column entries include the experiment number, the model or algorithm used, the document representation and *n*-gram range employed as features as well as the scores for various metrics. The user is able to sort the table based on any of the columns. Furthermore, upon clicking on any of the entries in the table, the user is provided with more information on the model and its classification performance. In particular, the hyperparameters selected during the grid search are shown for the machine learning models, along with the score achieved (in respect of the metric selected by the user — in this case, accuracy) during cross-validation in the bottom left-hand corner of the screen. Moreover, a confusion matrix containing the classification results in respect of the test data set is shown in the bottom right-hand corner of the screen. The entries of the confusion matrix are shaded according to the number of observations in each category, where a larger number of observations results in a darker shade of blue. It can thereby be determined at a glance that the

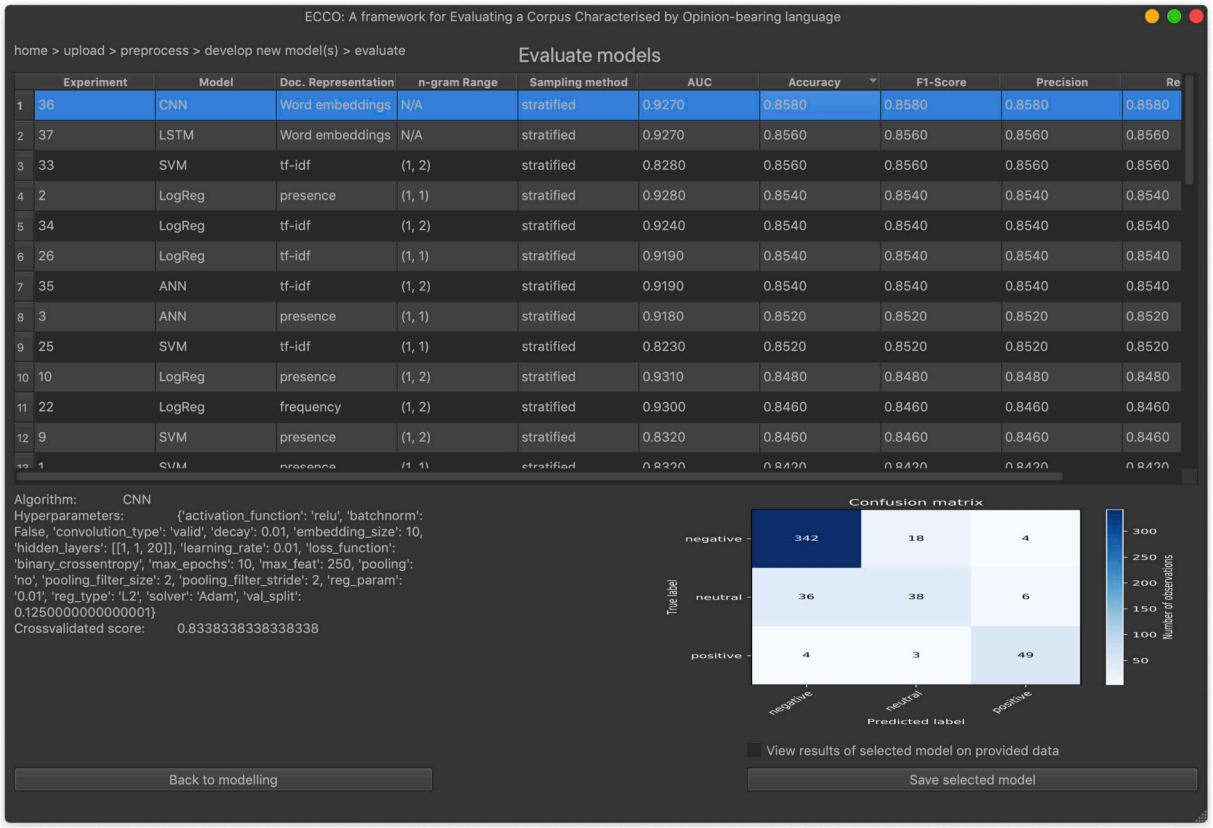


Fig. 11. The *evalute models* page of the GUI. The performance of each experiment is shown in a tabular format, and the hyperparameters and confusion matrix of the selected model are given below this table.

model selected in the Fig. 11 was primarily *'confused'* between negative and neutral observations. This interface enables the user to determine which models have performed best in respect of various metrics, and to elicit more detailed information about selected models and their shortcomings.

In order to account for the variability of the performance of all the models originating from the training data and test data, as well as the variability of the deep learning models originating from the random initialisation of network weights, each experiment was repeated ten times with a different random seed for each replication. The results are therefore presented in the form of box plots, indicating the median performance, as well as the degree of variation around this median. The accuracy scores achieved by each algorithm during the ten experimental runs are shown in Fig. 12 for each case study data set. Where multiple models were trained by means of the same algorithm (*i.e.* where several different input data representations were employed as input), the best performing model was selected.

It is evident from the figure that most, if not all, of the machine learning models trained according to the process outlined in the ECCO framework outperformed the off-the-shelf lexicon-based models by a large margin across all four domains. The CNN and LSTM algorithms fared slightly poorer than the remaining models for the PL04, Yelp and Twitter data sets. This is likely due to the specific model architecture chosen for the case study experiments, which includes a word embedding matrix trained end-to-end with the remainder of the network. With reference to Table 2, however, all of the data sets are relatively small, containing only 2,000–4,200 documents, compared to the large embedding matrices resulting from the vocabulary sizes of between 11,468 and 46,462 tokens. The machine learning models therefore likely did not have sufficient data to learn adequate embedding parameters. This effect is especially pronounced for the PL04 data set, which has both the largest vocabulary size and longest mean document length.

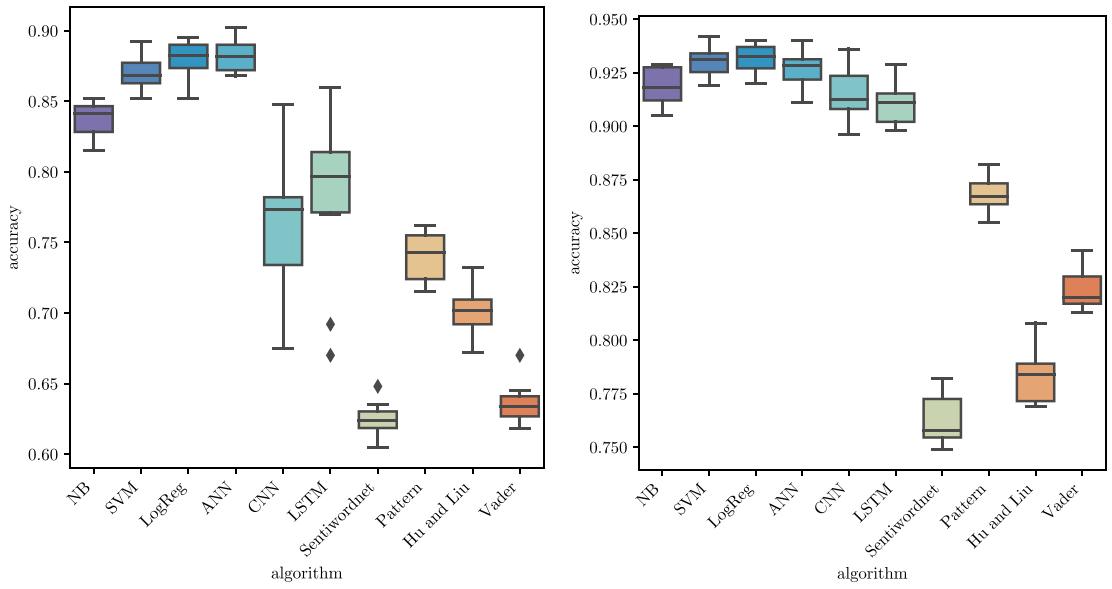
The best-performing models generated by means of the framework consistently achieved competitive results. In the original paper on the PL04 data set [33], Pang and Lee achieved an accuracy score of 87.15% using an SVM classifier. Using the MST selection process outlined by the ECCO framework, the same model achieved scores up to 89.2% during the ten experiments, with a mean accuracy score of 86.96%. The ANN model, on the other hand, achieved a mean accuracy of 88.27%. State-of-the-art results on the data set, as published by Nguyen et al. [39] and Maas et al. [40] range from 86.2–91.6%.

Whilst most researchers employing the Yelp data set predicted star ratings directly, the data set was transformed into a binary sentiment classification problem in this study, as was done by Salinca [34], who achieved a maximum accuracy score of 92.6%, using a subset of 10,000 of the data points and evaluating the model in respect of 20% of this subset. The best model (logistic regression) trained according to the ECCO framework in respect of the extracted subset of the data achieved an accuracy score of up to 94% during the ten experiments, with a mean accuracy score of 93.15%.

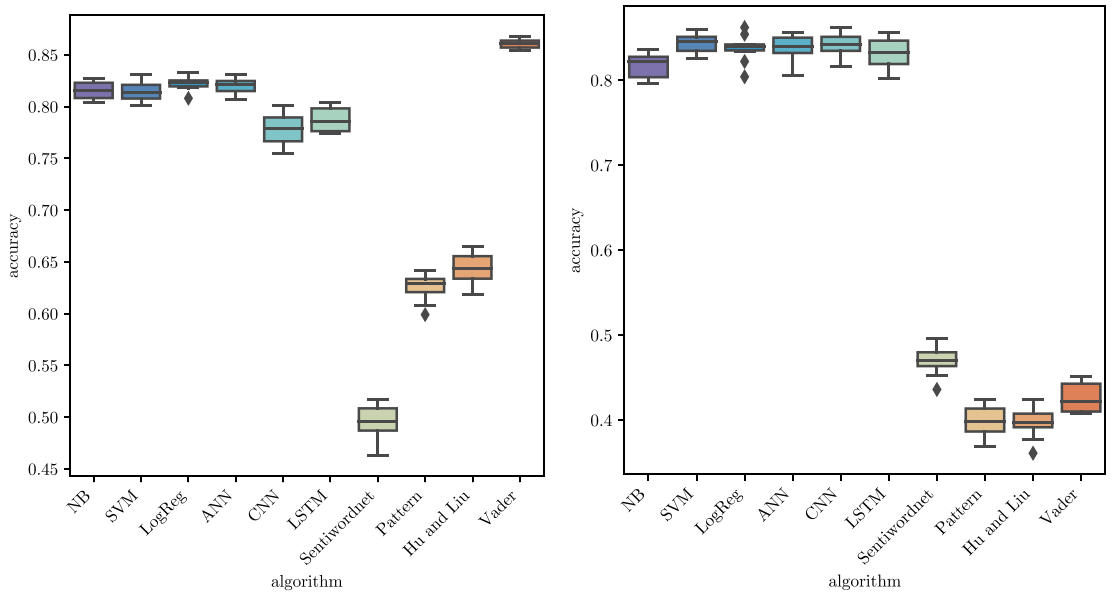
Finally, in respect of the Twitter data set, the best-performing ECCO-trained model was logistic regression with a mean accuracy score of 82.2%. This was significantly lower than the mean score of 86.06% achieved by Vader,[7] which was tailored specifically to this domain and data set. Given this fact, however, the results were still deemed competitive.

The lexicon-based models, on the other hand, achieved highly variable results across domains. Whilst the Vader model dominated in respect of social media data, it achieved the second-poorest results in

[7] In spite of best efforts to imitate the method of implementation, testing conditions and metrics used in the original paper [35], we were unable to replicate the reported F1 score of 96%. Relevant implementation details adopted in [35] may be missing to this end.



(a) PL04 (b) Yelp



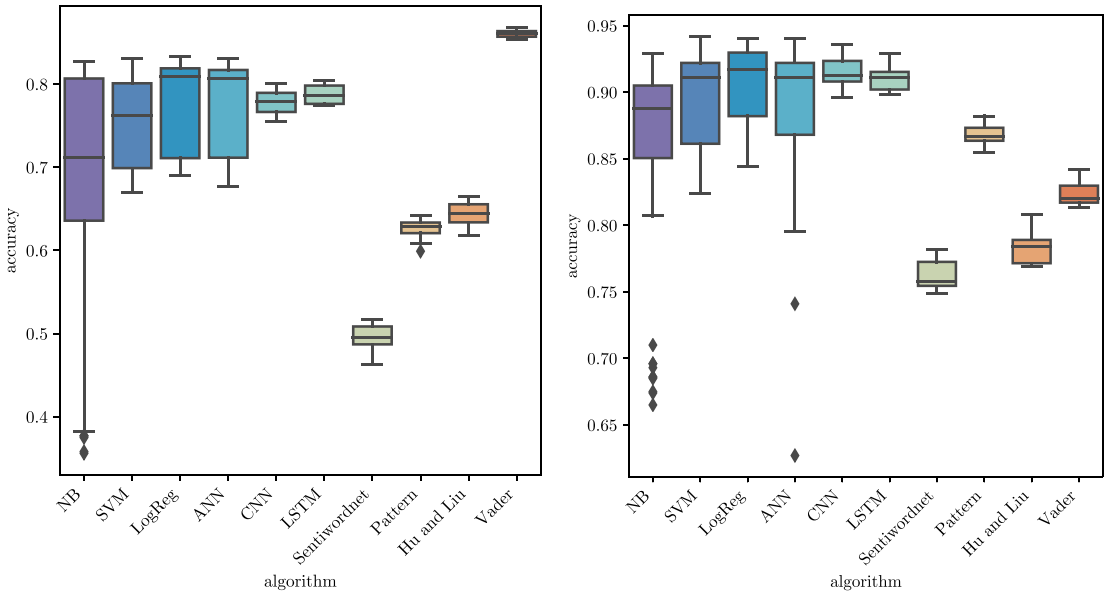
(c) Twitter (d) SMS

Fig. 12. Model performance in terms of accuracy. The accuracy values achieved by the best model-feature pairs during ten experimental runs are shown in the form of a box plot for each of the tested models and each case study data set.

the movie review domain. In fact, in each of the four domains, one of the four lexicon-based models performed better than the remaining three in turn. The difference in performance between machine learning and lexicon-based models is particularly pronounced for the SMS data set, where language is influenced by a non-US context.

These findings support the premise of the ECCO framework, which seeks to facilitate model development rather than apply a specific model in light of the fact that no single model can be guaranteed to outperform all other models in every problem setting. In this manner, a '*good*' sentiment classifier can be constructed for any problem domain.

Simply training a new model in respect of the available data, however, is not sufficient. Although other studies have shown machine learning models to outperform lexicon-based models in respect of certain data sets [41], such results are only achievable with the correct combination of features, model parameters and hyperparameters (the MST). As mentioned in Section 2, however, other existing frameworks do not account for this process and are therefore of little practical use to users who are not well versed in the MST selection processes. To emphasise this point, consider Fig. 13, in which the performance of the machine learning algorithms in respect of two of the data sets is shown for *all* model-feature combinations as opposed to the selected MST in each experimental run, as shown in Fig. 12. Comparing this figure with the previous one highlights the importance of good feature engineering. Whilst the performance of the naïve Bayes algorithm, for example, ranged between 80.4% and 82.7% for the selected feature sets in Fig. 12(c), the same model's performance varied between 35.6% and 82.7% considering all feature sets in Fig. 13(a). More specifically, the model achieved a mean accuracy of 81.31% using the unigram presence



(a) Twitter (b) Yelp

Fig. 13. Performance variation induced by feature engineering. The accuracy values achieved by all model-feature pairs during ten experimental runs are shown in the form of a box plot for each of the tested models.

feature representation, and a mean accuracy of 38.06% using the bigram TF-IDF feature representation. Selecting the correct features is therefore of critical importance to the performance of a machine learning algorithm.

As with the performance of a particular model, however, the relative performance of a particular set of features is not necessarily consistent across all data sets. Consider, for example, the variation in relative performance between the TF-IDF document model and the n-gram range (2,2) (bigrams only) across the PL04 and SMS data sets illustrated in Fig. 14. Whereas the bigrams representation fares particularly poorly in respect of the SMS data set, this difference is not as pronounced for the PL04 data set (and may even be reversed for a different data set). Similarly, whilst the performance of the TF-IDF representation is relatively inconsistent in respect of the PL04 data set, this is not the case for the SMS data set, where this representation, in fact, achieves marginally superior performance.

The examples above relate to the variation in performance caused by feature engineering efforts. The effect of hyperparameter selection is, however, not included in these comparisons, since a grid search was performed for each experiment in the ECCO system. From the outputs of the grid search algorithm, however, observations were made as to the significant effect of this process on model performance. Within the single model-feature combination of the logistic regression model with a unigram presence representation, for example, the validation performance[8] achieved by models employing different hyperparameter combinations varied between 89.24% and 96.91% for the Yelp data set. The final hyperparameter configuration included a value of 1 for the tuning parameter C and the use of the *stochastic average gradient* solver. For the unigram TF-IDF representation, on the other hand, the *Newton-CG* solver with $C = 10$ led to the best performance.

It is therefore imperative to employ a structured approach towards finding suitable sentiment models for each new data set in order to achieve consistent performance across various domains. To the best of our knowledge, the ECCO framework is the only framework which integrates the process of selecting a suitable MST into a sentiment analysis framework, thereby offering a practical guide to researchers and analysts on how to robustly develop competitive models for sentiment analysis across various domains and problem types.

4.4. Analysis of the model results

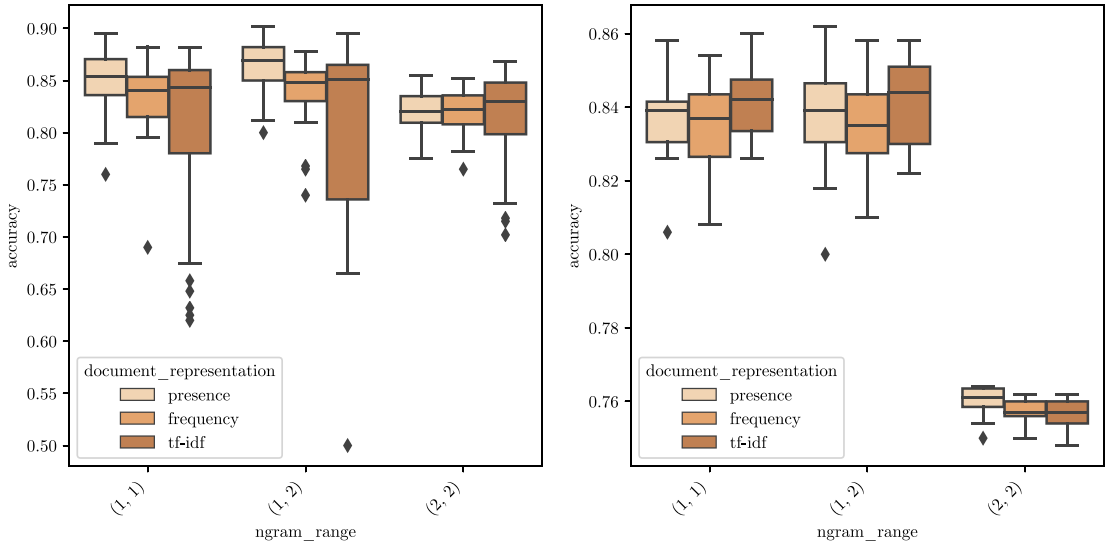
For each of the case study data sets, the best-performing model was selected in order to proceed to the analysis phase of the ECCO framework. More specifically, the ANN, logistic regression, Vader and CNN models were selected to classify the PL04, Yelp, Twitter and SMS data, respectively. The ECCO framework was then employed to analyse the results of these models in order to extract valuable insights from the data that may inform various decision making tasks.

Processes 16.0 and 15.0 of the analysis component of the framework (shown in Fig. 7) were applied to the case study data by means of summaries, including a textual summary (by *template instantiation*), general statistics, and word cloud representations of documents in each sentiment class, as well as by an LDA topic analysis. The word cloud representations of the documents classified into each of the three sentiment classes for the Twitter data set, for example, are shown in Fig. 15. From this representation, it is clear that the terms *bad*, *day* and *RT* are the most common terms mentioned in negative Tweets, whilst the terms *good*, *thank* and *love* are most commonly mentioned in positive Tweets. Interestingly, the expression *RT* or *retweet*, referring to the re-posting or sharing of another person's Tweet, is more prominent in negative rather than positive Tweets in this data set, which may allude to the notion that *'bad news travels fast.'*

The generated topic models were visualised by invoking the package *LDAvis* by Sievert and Shirley [42], as shown in Fig. 16 for the Yelp data set. Based on both the frequency with which each of the most *salient* terms in the corpus were observed in a given topic and the *relevance* of words to a given topic, both of which are visualised in the LDAvis interface, several important keywords could be associated with each topic. For Topic 1, for instance, which is selected in the figure, numerical expressions are most relevant and frequent, along with the terms *order*, *time*, *food*, *get*, *minute* and *wait*. These word co-occurrence patterns suggest that Topic 1 is primarily concerned with reviewers having had to wait long periods for their (food) orders.

After having identified important topical keywords, their importance and relevance to each sentiment class were estimated by

[8] The three-fold cross validated accuracy in respect of the training data.



(a) PL04 (b) SMS

Fig. 14. Performance for various feature sets. The accuracy values achieved during ten experimental runs are shown in the form of a box plot for all models using a particular feature set. The *n*-gram ranges denote the use of unigrams only, unigrams with bigrams, and bigrams only, in order of appearance on the horizontal axis.

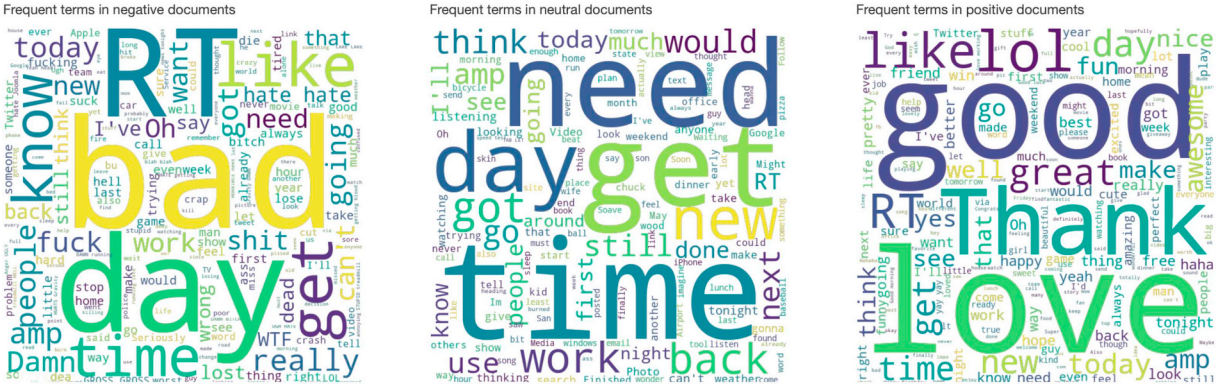


Fig. 15. Word cloud representations of the most frequently used words in each sentiment class for the Twitter data set.

means of further visualisation. Several frequent keywords mentioned in the PL04 data set, for example, are illustrated in Fig. 17. Since the class distribution of the data set was perfectly balanced, keywords which occur significantly more frequently in one sentiment class than another appear to be related to this particular sentiment. From the figure, it is evident, for example, that the *plot* of a reviewed movie is significantly more likely to be mentioned in the criticism of a film rather than its acclamation.

Where supplementary data are available (as for the Yelp and SMS data sets), the relationship between these structured variables and sentiment can be explored using the ECCO framework. Such an analysis can provide valuable further insight into the data and expose hidden patterns useful for decision making. As mentioned in Section 2, the utility of most sentiment analysis frameworks is limited to the sentiment modelling phase or finding an accurate representation of the sentiment distribution (*e.g.* 70% of customers are dissatisfied). The analyses of the review text described above already offer helpful further insights (*e.g.* many dissatisfied customers complained about insufficient *automatic teller machines* (ATMs)). This information is, however, often not yet specific enough to drive decision making. The additional analysis of structured variables can provide a third layer of insight (*e.g.* customers complaining about a lack of ATMs tend to be those in rural areas, and pay monthly fees that are 15% higher than the customer average), rendering the analysis more actionable (*e.g.* address the ATM issue by installing additional machines in rural areas or entering partnerships with local supermarkets for inexpensive cash withdrawals). As with any such analysis, however, care must be taken when interpreting the results. The process outlined in the ECCO framework can only uncover hidden correlations — the assessment of possible causalities lies with decision makers.

Process 18.0 of the ECCO framework refers to type-specific visualisations of model-feature relationships. Box plots may, for example, be employed to visualise the relationship between quantitative variables and sentiment. Examples from the Yelp data set are shown in Fig. 18. From this figure, it may be deduced that sentiment expressed in reviews correlated almost as strongly with the average star rating given by a user across establishments on the platform as with the average star rating of an establishment. Both the experience and the attitude of a visitor may, therefore, influence the nature of a review.

Similarly, histograms and *bubble maps* may be used to illustrate the relationship between sentiment and respectively qualitative and location information. From Fig. 19(a), for example, it is clear that most of the reviews in the Yelp data set (which were sampled randomly from the larger data set) were on establishments located in Nevada or Arizona, with the proportion of negative reviews exhibited in Arizona significantly larger than that of Nevada. By employing the bubble map view shown in Fig. 19(b), the largest concentrated area of reviews was identified as the Las Vegas strip, with *Gordon Ramsay Hell's Kitchen*

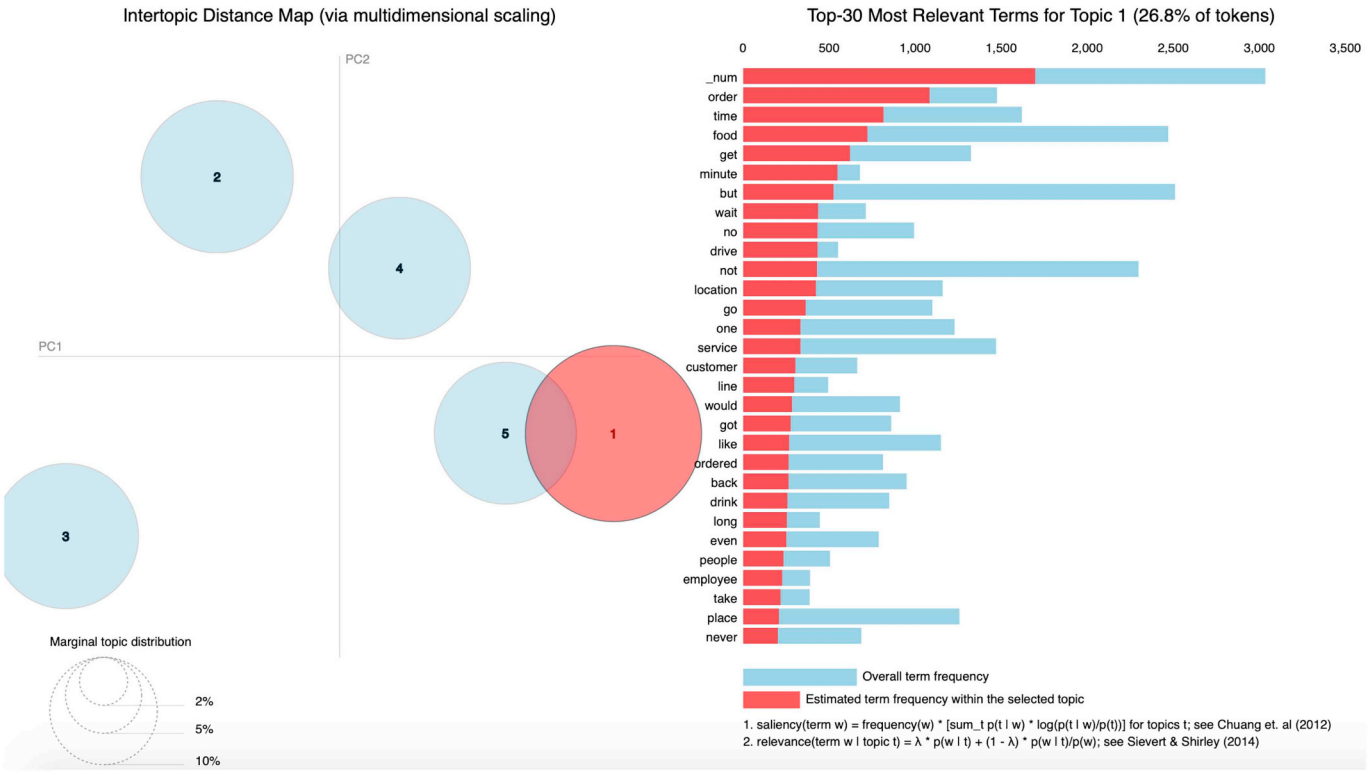


Fig. 16. *LDAvis* visualisation of the LDA topic model for the Yelp data set based on five topics and five passes through the data. Topic 1 is currently selected and the most relevant terms shown on the right-hand side of the figure.

receiving the largest number of reviews for an individual establishment. Compared to the establishment to its left (*Bacchanal Buffet*), the restaurant received a significantly larger proportion of positive reviews.

With the use of the filtering function of Process 14.0, these visualisations can be made increasingly fine-grained. The relationship between the initial rating of a customer and the sentiment of the following free-form response in the SMS data set is, for example, analysed in Fig. 20. From Fig. 20(a), it is clear that most customers who complained about loans rated the bank with a 3 (the worst possible rating). *Loan*, being among the three most frequently mentioned keywords, therefore also appears to be a considerable source of dissatisfaction for customers. The keyword *ATM*, on the other hand, whilst frequently mentioned by customers, did not appear to result in as negative an overall rating. In fact, as shown in Fig. 20(b), most customers who mentioned ATMs in their subsequent reviews evaluated the bank with a rating of 2 out of 3.

Finally, latent relationships between structured variables and the sentiment class can be investigated by means of a multivariate model, as outlined in Process 17.0 of Fig. 7. The classification tree illustrated in Fig. 21 was, for example, fit on the subset of reviews from the SMS data set which contain the keyword *service*. This model achieved a validation accuracy of 72.5% after some tuning of the available hyperparameters during a manual search facilitated *via* a simple interface. The top split of the decision tree is the *Q01 value*, or the initial rating of the customer. The left-hand branch of the root of the tree (observations for which the initial rating was not equal to 3) is classified as positive, whereas the right-hand branch of the tree (observations for which the initial rating was equal to 3) is classified as negative. The decision tree further splits on the attributes describing the average monthly fee paid by customers and the customers' age for the positive and negative branches, respectively. More specifically, the model is *'more certain'* that a given customer submitted a positive review after giving the bank a rating score of 2 out of 3 if the customer's average monthly banking fee is larger than −R 72.24 (*i.e.* less than ZAR 72.24 ≈ US$ 5 per month). Furthermore, the model is *'more certain'* that a review following a rating of 3 is negative in sentiment if the customer is at most 36 years old. Such an analysis can be helpful when constructing certain *profiles* in terms of structured attributes that were typically observed in a particular sentiment category.

As was illustrated in the examples above, the ECCO framework provides a unique approach for analysing and sythesising sentiment

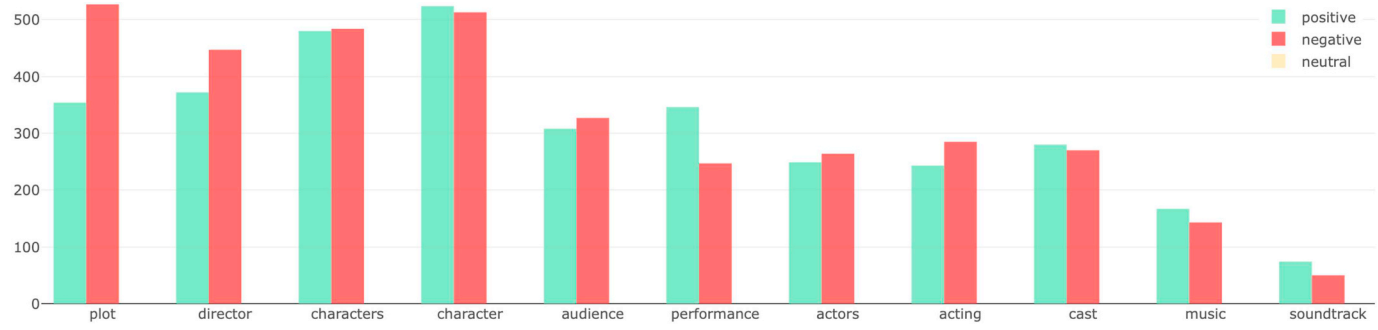


Fig. 17. The frequency of various keywords in documents of each sentiment class for the PL04 data set.

(a) Average stars (b) Average stars user

Fig. 18. Box plots illustrating the relationship between sentiment and (a) the average star rating received by an establishment or (b) the average star rating given by a user.

information that does not rely on assumptions about the type of data being analysed (as many other frameworks do by analysing sentiment according to *product* features) and which facilitates an important and novel '*third layer*' of analysis by incorporating structured variables. This is achieved in an original manner by taking a *decision support* approach to data mining rather than compiling reports *automatically* according to some predefined pattern.

5. Discussion and conclusions

In this paper, a generic framework for evaluating unstructured data by means of sentiment analysis, the ECCO framework, was proposed. This framework addresses two major shortcomings of existing frameworks in the literature, namely the non-existence of a framework that is *comprehensive* enough to be readily applied by analysts, yet *generic* enough to be applicable across various domains, as well as the lack of depth and flexibility afforded by existing frameworks to analyse model results with a view to inform decision making in response to sentiment information.

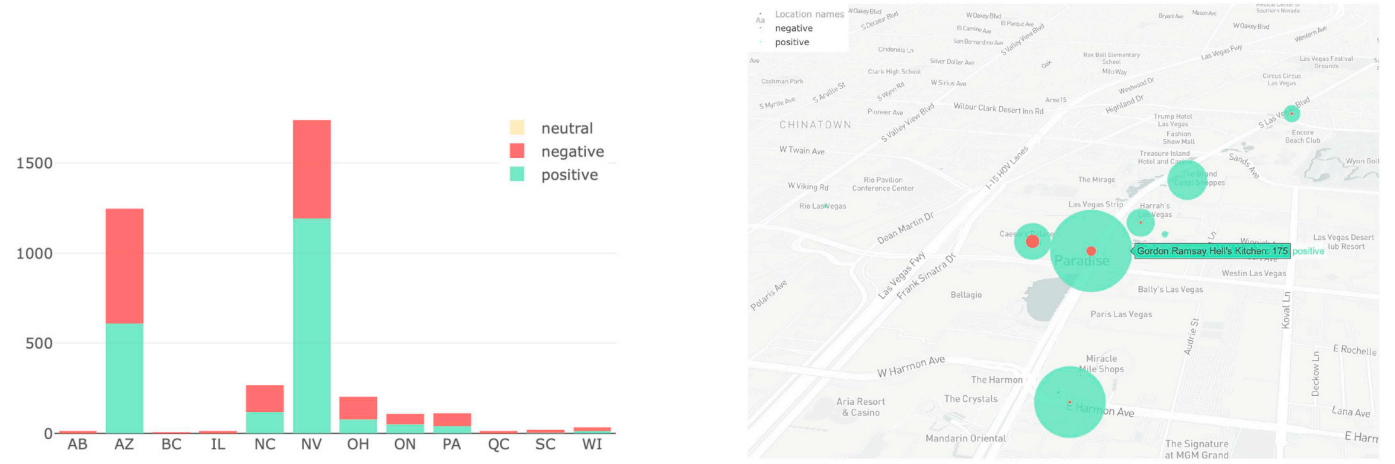
5.1. Research contributions

Before presenting the ECCO framework, a generic data science paradigm was proposed within which the ECCO framework was developed. This paradigm integrates the data science process proposed by O'Neill and Schutt [30] with the widely adopted architectural guidelines for a DSS, and reflects the stages into which a model-driven DSS is typically partitioned (the formulation stage, the solution stage and the analysis stage). Although this paradigm is simple, it may provide a good starting point for the development of future frameworks aimed at facilitating the transformation of raw data into information by adopting a decision support approach. Furthermore, it may provide a structure for comparing, as well as a shared terminology for describing, similar frameworks.

The ECCO framework itself combines the preprocessing, sentiment modelling and analysis steps of a sentiment analysis problem into one comprehensive framework that is generally applicable. The lack of such a framework in the literature was elucidated in Section 2. The TOM framework [7], for example, combined preprocessing and modelling activities, but these steps are tailored specifically to the modelling of sentiment in Twitter data and the analysis of the results is not addressed. The Opinion Observer [12], on the other hand, focused on the analysis and synthesis of product reviews, but did not incorporate sentiment preprocessing or sentiment modelling activities.

In this paper, it was shown that the preprocessing approach prescribed by the ECCO framework was applicable across four distinct domains. The introduction of user feedback and an iterative approach, moreover, enabled the selection of a good preprocessing configuration in each particular case and gave initial insight into the data.

To the best of our knowledge, the ECCO framework is the first to incorporate the entire MST selection process for modelling sentiment by means of machine learning into a sentiment analysis framework,



(a) Sentiment distribution per state (b) Map view of sentiment per establishment

Fig. 19. Visualisation of the relationship between (a) qualitative variables and (b) location data and sentiment for the Yelp data set.

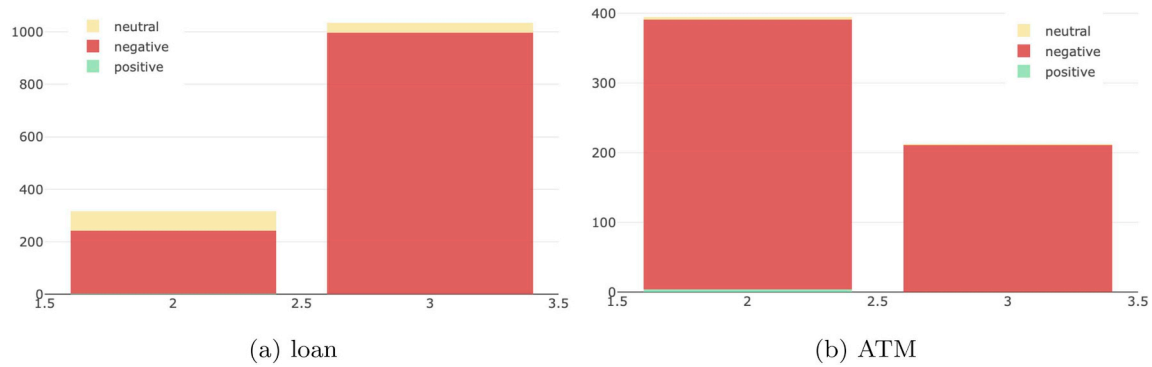


Fig. 20. Producing a fine-grained analysis of the relationship between the qualitative variable Q01 Value (the customer rating) and sentiment by means of filtering on the keywords (a) *loan* and (b) *ATM*.

thereby affording a user the ability to construct robust sentiment models regardless of domain and problem type. In their paper on the Heracles framework referred to in Section 2, Schouten et al. [27] also highlighted the importance of model evaluation and comparison in text mining applications. In their framework, however, the target of this evaluation were the machine learning models themselves, as opposed to the MSTs (incorporating model-feature pairs and their hyperparameters) addressed in the ECCO framework. Feature engineering and hyperparameter tuning efforts are left to automated functions in external packages in the Heracles framework, whose suitability and performance were not evaluated. In this paper, it was illustrated that, by following the process outlined in the ECCO framework, machine learning models could be developed that achieved performance competitive with benchmark results across four distinct domains in both binary and ternary problem classes, and that these models generally outperformed specific pre-trained lexicon-based models by a significant margin.

Our framework is also the first to explicitly explore the relationship between sentiment and supplementary structured variables, facilitating a deeper analysis of sentiment data and allowing decision makers to extract actionable insight from data in order to drive organisational improvement. Furthermore, the exploratory approach adopted in the analysis component allows for the analysis of data from various domains without prerequisite knowledge. In comparison, the popular Opinion Observer framework [12] described in Section 2, for example, facilitates the analysis of sentiment data for product reviews by segmenting sentiment according to product features, whilst the BESAHOT framework [14] by Kasper and Vela (also described in Section 2) facilitates the analysis of hotel reviews by displaying reviews related to various topics discussed by guests (*e.g.* room cleanliness) and summary statistics on overall reviewer demographics. The ECCO framework is applicable in both domains, and the summaries generated by each of the existing frameworks are implicitly incorporated into the framework by means of the histograms for qualitative data comparison (akin to the product feature-sentiment graphs of the Opinion Observer), the topic modelling functionality paired with the filtering function (akin to the summary by topic of BESAHOT) and the type-specific visualisations of supplementary data (akin to the summary statistics of reviewer demographics of BESAHOT). Furthermore, the ECCO framework would also facilitate an investigation of the relationship between sentiment and supplementary data describing guests (*e.g.* do single business travellers complain more than families?), which is not supported by the BESAHOT framework. The ECCO framework therefore offers both versatility and depth of analysis.

Applications of the framework include, for example, the evaluation of customer reviews on a business to inform actions in product development or service delivery aimed at addressing problem areas and the identification of profiles of high-risk companies subjected to

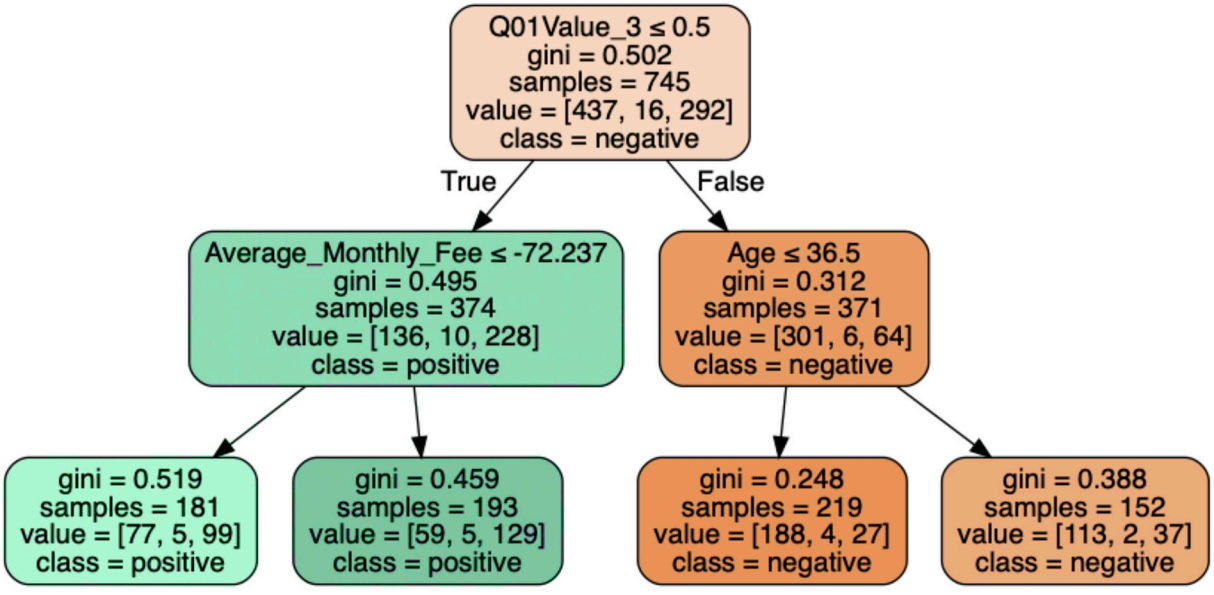


Fig. 21. A decision tree fit to the subset of reviews from the SMS data set that mention the keyword *service*.

particularly negative reviews on the Internet. The former was illustrated in the SMS case study, where customer feedback on a bank's services was evaluated in order to identify key concerns of clients, such as a lack of ATMs, and to make decisions to address these concerns. Components of the framework can also be applied in isolation. The modelling component may, for example, be applied to processed data simply to find a good MST for a particular classification task.

5.2. Limitations and future work

The case study examples in this paper were limited to a *competitive* selection mechanism between models. As described in Section 3.4, however, the ECCO framework also contains a *complementary* selection mechanism in Process 12.0, or the combination of several models into an ensemble. The benefits of this approach and the comparative performance of ensemble models in respect of the case study data are the subject of continued work.

Furthermore, as mentioned in the paper, the case study examples all exhibited relatively small data sets, all of which were predominantly in English (although the SMS data were gathered from an African context with different linguistic nuances). There was, however, considerable diversity in domain, medium (SMS, social media, review sites), document length and vocabulary size across the case study examples, which was deemed sufficient to support a claim of generality of the framework.

Finally, the study was limited in scope to perform sentiment classification at the document level, since it is less expensive to label overall sentiment than sentiment related to individual aspects. More specifically, the sentiment polarity of a document was determined by means of the modules of the *sentiment modelling* component of the framework, and these results were later summarised in respect of automatically extracted aspects (or *keywords*) using the modules of the *analysis* component of the framework. An alternative approach would be first to extract aspects from each review and then to determine the sentiment polarity of text excerpts related to a given aspect. This may, however, also require such phrases to be labelled individually or necessitate the use of weakly supervised models. Models for aspect extraction include *association rule mining* and attention-based recurrent neural networks, which learn to emphasise aspect-related terms during the training of aspect embeddings. An overview of the current state of the art in this growing field is provided in [43].

CRediT authorship contribution statement

Jacqueline Kazmaier: Conceptualization, Methodology, Software, Validation, Data curation, Formal analysis, Writing - original draft, Writing - review & editing. **Jan H. van Vuuren:** Conceptualization, Supervision, Resources, Writing - review & editing.

References

[1] B. Liu, Sentiment analysis and opinion mining, Synthesis Lectures on Human Language Technologies, vol. 1 (5), 2012, pp. 1–168.
[2] K. Bannister, Understanding sentiment analysis: what it is & why it's used, https://www.brandwatch.com/blog/understanding-sentiment-analysis/, [Accessed: February 2018] (2015).
[3] J. A. Vargas, Spring awakening: how an egyptian revolution began on facebook, http://www.nytimes.com/2012/02/19/books/review/how-an-egyptian-revolution-began-on-facebook.html, [Accessed: February 2018] (2012).
[4] A. Sharma, S. Dey, A comparative study of feature selection and machine learning techniques for sentiment analysis, 2012 ACM Research in Applied Computation Symposium, San Antonio (TX), 2012, pp. 1–7.
[5] S. Wang, C. Manning, Baselines and bigrams: Simple, good sentiment and topic classification, 50th Annual Meeting of the Association for Computational Linguistics, Jeju Island, 2012, pp. 90–94.
[6] A. Kumar, R. McCann, J. Naughton, J.M. Patel, Model selection management systems: the next frontier of advanced analytics, SIGMOD Record 44 (4) (2016) 17–22.
[7] F.H. Khan, S. Bashir, U. Qamar, TOM: Twitter opinion mining framework using hybrid classification scheme, Decis. Support. Syst. 57 (2013) 245–257, https://doi.org/10.1016/j.dss.2013.09.004.
[8] M.Z. Asghar, A. Khan, S. Ahmad, M. Qasim, I.A. Khan, Lexicon-enhanced sentiment analysis framework using rule-based classification scheme, PLoS One 12 (2) (2017) 1–22, https://doi.org/10.1371/journal.pone.0171649.
[9] F.H. Khan, U. Qamar, S. Bashir, eSAP: a decision support framework for enhanced sentiment analysis and polarity classification, Inf. Sci. 367 (2016) 862–873.
[10] A.G. Vural, B.B. Cambazoglu, P. Senkul, Z.O. Tokgoz, A framework for sentiment analysis in Turkish: application to polarity detection of movie reviews in Turkish, Comput. Inf. Sci. (2013) 437–445.
[11] J. Yi, T. Nasukawa, R. Bunescu, W. Niblack, Sentiment analyzer: extracting sentiments about a given topic using natural language processing techniques, 3rd IEEE International Conference on Data Mining, Melbourne (FL), 2004, pp. 427–434.
[12] B. Liu, M. Hu, J. Cheng, Opinion observer: Analyzing and comparing opinions on the web, International World Wide Web Conference, ACM, Chiba, 2005, pp. 342–351.
[13] K. Dave, S. Lawrence, D.M. Pennock, Mining the peanut gallery: opinion extraction and semantic classification of product reviews, International World Wide Web Conference, ACM, Budapest, 2003, pp. 519–528.
[14] W. Kasper, M. Vela, Sentiment analysis for hotel reviews, Speech Technol. 4 (11) (2012) 96–109.
[15] G. Ren, T. Hong, Investigating online destination images using a topic-based sentiment analysis approach, Sustainability 9 (10) (2017) 1765:1–1765:18.
[16] C.I.M. de Lucena, Framework for Collaborative Knowledge Management in Organizations, PhD thesis NOVA University Lisbon, Lisbon, 2016.
[17] N. Yussupova, M. Boyko, D. Bogdanova, A decision support approach based on sentiment analysis combined with data mining for customer satisfaction research, Int. J. Adv. Intell. Syst. 1 (1) (2015) 145–158.
[18] S.J. Wu, R.D. Chiang, H.C. Chang, Applying sentiment analysis in social web for smart decision support marketing, J. Ambient. Intell. Humaniz. Comput. (2018) 1–10.
[19] M. Bank, AIM — A Social Media Monitoring System for Quality Engineering, PhD Thesis University of Leipzig, Leipzig, 2013.
[20] K. Ganesan, Opinion driven decision support system, PhD Thesis University of Illinois, Champaign (IL), 2013.
[21] M.R. Yaakub, Y. Li, J. Zhang, Integration of sentiment analysis into customer relational model: the importance of feature ontology and synonym, Proc. Technol. 11 (2013) 495–501, https://doi.org/10.1016/j.protcy.2013.12.220.
[22] T.L. James, E.D. Villacis Calderon, D.F. Cook, Exploring patient perceptions of healthcare service quality through analysis of unstructured feedback, Expert Syst. Appl. 71 (2017) 479–492.
[23] N.-L. Hsueh, A framework for opinion mining system with design pattern, 2016 International Computer Symposium (ICS), IEEE, Chiayi, 2016, pp. 626–631.
[24] E. Lloret, A. Balahur, J.M. Gómez, A. Montoyo, M. Palomar, Towards a unified framework for opinion retrieval, mining and summarization, J. Intell. Inf. Syst. 39 (3) (2012) 711–747.
[25] H. Isah, P. Trundle, D. Neagu, Social media analysis for product safety using text mining and sentiment analysis, 14th UK Workshop on Computational Intelligence, IEEE, Bradford, 2014, pp. 1–7.
[26] S. Liu, I. Lee, A hybrid sentiment analysis framework for large email data, 10th International Conference on Intelligent Systems and Knowledge Engineering, Taipei, 2015, pp. 324–330.
[27] K. Schouten, F. Frasincar, R. Dekker, M. Riezebos, Heracles: a framework for developing and evaluating text mining algorithms, Expert Syst. Appl. 127 (2019) 68–84.
[28] J. Shim, M. Warkentin, J.F. Courtney, D.J. Power, R. Sharda, C. Carlsson, Past, present, and future of decision support technology, Decis. Support. Syst. 33 (2002) 111–126.
[29] R.M. Stair, G.W. Reynolds, J. Hulbert, J.W. Calhoun, L. Shipp, Principles of Information Systems, 10th edition, Cengage Learning, Boston (MA), 2012.
[30] C. O'Neil, R. Schutt, Doing Data Science: Straight Talk from the Frontline, 1st edition, O'Reilly Media Inc, Sebastopol (CA, 2014.
[31] T. Fawcett, An introduction to roc analysis, Pattern Recogn. Lett. 27 (8) (2006) 861–874.
[32] C.M. Bishop, Pattern Recognition and Machine Learning, 29 Springer Science & Business Media, New York (NY), 2006.
[33] B. Pang, L. Lee, A sentimental education: sentiment analysis using subjectivity summarization based on minimum cuts, Annual Meeting of the Association for Computational Linguistics, Barcelona, 2004, pp. 271–278.
[34] A. Salinca, Business reviews classification using sentiment analysis, International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), 2015, pp. 247–250.
[35] C.J. Hutto, E. Gilbert, Vader: a parsimonious rule-based model for sentiment analysis of social media text, 8th International AAAI Conference on Weblogs and Social Media, Ann Arbor (MI), 2014, pp. 216–225.
[36] S. Bird, E. Klein, E. Loper, Natural Language Processing with Python — Analyzing Text With the Natural Language Toolkit, 1st edition, O'Reilly Media, Sebastopol (CA, 2009.
[37] G.A. Miller, Wordnet: a lexical database for english, Commun. ACM 38 (11) (1995) 39–41.
[38] K. Atkinson, Gnu aspell, http://aspell.net, [Accessed: May 2019] (2016).
[39] D.Q. Nguyen, D.Q. Nguyen, T. Vu, S.B. Pham, Sentiment classification on polarity reviews: an empirical study using rating-based features, Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, 2014, pp. 128–135, , https://doi.org/10.3115/v1/W14-2621.
[40] A.L. Maas, R.E. Daly, P.T. Pham, D. Huang, A.Y. Ng, C. Potts, Learning word vectors for sentiment analysis, 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland (OR), 2011, pp. 142–150.

[41] Z. Hailong, G. Wenyan, J. Bo, Machine learning and lexicon based methods for sentiment classification: a survey, Web Information System and Application Conference, IEEE, Tianjin, 2014, pp. 262–265.
[42] C. Sievert, K. Shirley, Ldavis: a method for visualizing and interpreting topics, Workshop on Interactive Language Learning, Visualization, and Interfaces, Baltimore (MD), 2014, pp. 63–70.
[43] K. Schouten, F. Frasincar, Survey on aspect-level sentiment analysis, IEEE Trans. Knowl. Data Eng. 28 (3) (2016) 813–830.

Jacqueline Kazmaier was born in Windhoek, Namibia. She obtained a bachelor's degree in industrial engineering from Stellenbosch University in 2017 and enrolled for a master's degree in the SUnORE research group within the Department of Industrial Engineering at the same institution in 2017. Her master's thesis was upgraded to PhD in November 2019. She was awarded the prestigious Chancellor's medal at Stellenbosch University in 2017 and was also a runner-up for the Gerhard Geldenhuys medal for the best final-year project in South Africa by the Operations Research Society of South Africa in 2018. She has attended and presented her work at several conferences. She has also supervised undergraduate students towards the completion of their studies in areas of operations research. Her research interests include machine learning, natural language processing, decision support systems and optimisation.

Jan van Vuuren was born in Durban, South Africa. He obtained a master's degree in applied mathematics from Stellenbosch University in 1992 and a doctorate in mathematics from the university of Oxford, United Kingdom, in 1995. He has been a member of staff at Stellenbosch University since 1996, first at the department of Applied Mathematics (1996–2007) and then in its Department of Logistics (2007–2013). He is currently professor of operations research and head of the SUnORE research group within the department of Industrial Engineering at Stellenbosch University. He is the author of a number of journal and peer-reviewed conference proceeding papers. He has also supervised many undergraduate and postgraduate students to the successful completion of their studies in areas of operations research. His research interests include graph theory, combinatorial optimisation, and decision support systems.