Automated content analysis for modelling student engagement in online discussion forums

Elaine Farrow

Master of Science by Research

Centre for Doctoral Training in Data Science
School of Informatics
University of Edinburgh

2018

Abstract

The widespread use of online discussion forums in educational settings of all kinds provides a rich source of data for researchers interested in how collaboration and interaction can foster effective learning. Online behaviour can be understood through the Community of Inquiry framework, and the *cognitive presence* construct in particular can be used to characterise the depth of a student's critical engagement with the course material. Automated methods have been developed to support this task, but many studies used very small data sets, and there have been few replication studies. Furthermore, some of the classification features that were used in prior work depended on an external knowledge base, limiting their applicability in other domains and language contexts.

In this work, we present findings related to the robustness and generalisability of automated classification methods for detecting cognitive presence in discussion forum transcripts. We closely examined one published state-of-the-art model, comparing different approaches to managing unbalanced classes. We derived new classification features using natural language processing techniques. The explanatory power of the individual features was analysed and compared with prior work.

By demonstrating how commonly-used data preprocessing practices can lead to over-optimistic results, we contribute to the development of the field so that the results of automated content analysis can be used with confidence. We also show that topic modelling can be used to generate new features with explanatory power.

Acknowledgements

I would like to thank my supervisors, Dragan Gašević and Johanna Moore, for their knowledgeable input, help and support.

My grateful thanks also go to Vitomir Kovanović, who generously shared his code, and to Rafael Ferreira and Ed Fincham for useful discussions.

Finally, I want to thank all the members of the CDT in Data Science, both students and staff, who have made this such a stimulating year.

This work was supported in part by the EPSRC Centre for Doctoral Training in Data Science, funded by the UK Engineering and Physical Sciences Research Council (grant EP/L016427/1) and the University of Edinburgh.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Elaine Farrow*)

Table of Contents

Chapter 1

Introduction

1.1 Motivation

Technology use is now a fundamental aspect of the educational experience for many students, and its importance is widely recognised in the research community. The significance of online discussion forums, where students can interact with one another and with their tutors, is of particular note. Some courses operate fully online, with the discussion forum playing a central role. Forums are also a vital component in blended learning courses, which combine face-to-face instruction with rich online interaction. Nowadays, even traditional face-to-face courses with large numbers of students increasingly use text-based forums such as Piazza[1] and online annotation tools like Hypothesis[2] to manage students' questions.

In addition to their primary role in supporting education through interaction and collaboration, discussion forums can also be used to inform research. The messages exchanged in the forum can be exported as a time-stamped record of the discussion. Forum transcripts of this sort encompass social exchanges as well as task-focussed talk and form a rich source of material for researchers interested in studying how participants work together online, and how effective learning takes place through discussion.

The Community of Inquiry (CoI) framework for online education has emerged as a powerful tool for analysing and developing effective learning experiences (Garrison et al., 2000). Since its introduction in 2000, the CoI framework has been used in many studies and found to be both useful and robust (Kovanović et al., 2016). The framework identifies the three main elements (or *presences*) that are important for a

[1] https://piazza.com
[2] https://web.hypothes.is

successful educational experience: a social environment conducive to learning (*social presence*), a well-designed course with ongoing facilitation (*teaching presence*), and the student's own cognitive engagement with the subject (*cognitive presence*).

While CoI has been well received and widely adopted by researchers, its application in practical educational contexts has been limited because of the difficulty in measuring the three presences in a timely manner. Early work relied on manual coding to identify evidence of the presences in discussion transcripts, using a set of coding schemes defined in the CoI framework (Garrison et al., 2001). However, this is time-consuming, expert work, and cannot be deployed in real time. Another approach used self-reported measures collected through surveys (Arbaugh et al., 2008), but these are too intrusive to use for ongoing monitoring.

In response to these limitations, work has been done to develop automated classifiers using features extracted from transcript data. These features include word n-grams and part-of-speech tags, word counts and linguistic complexity scores, context-based metrics like the number of replies a message received, and semantic annotations grounded in an external knowledge base. Corich et al. (2006b) developed an automated content analysis tool and used it to classify forum messages into one of the four levels of cognitive presence. Waters et al. (2015) looked at predicting the level of cognitive presence for entire chains of messages, instead of treating messages in isolation, since cognitive presence is expected to develop over time. Kovanović et al. (2016) developed a model that is able to identify the level of cognitive presence with 70.3% accuracy, compared to gold-standard human annotation. This represents the state-of-the-art for English language data. Neto et al. (2018) followed the same methodology to analyse a corpus of messages written in Portuguese and achieved 83% accuracy.

We observe that many of these models were developed on small data sets, and there have been very few replication studies. It is important for automated classification techniques to be evaluated rigorously in order to understand how well they are likely to perform on new data. In addition, the use of classification features that rely on an external knowledge base limits the usefulness of the model for subject domains and languages where such resources are not readily available. Finally, it is helpful to consider whether there are other features related to the message data that might serve as indicators of cognitive presence.

This work aims to address these concerns. Our goal is to improve the robustness and generalisability of automated classification methods for detecting cognitive presence, so that this work can be used with confidence by researchers in the field.

1.2 Research questions

We addressed three research questions.

Research Question 1

How do the results from the state-of-the-art model (Kovanović et al., 2016) compare with a replication study using current best practice for handling unbalanced classes and splitting data into training and test sets?

Research Question 2

Can we achieve similar predictive performance without relying on an external knowledge base for semantic annotation?

Research Question 3

Can topic modelling give us new insights into cognitive presence?

1.3 Findings

We found that a best-practice replication study was not able to match the published results from Kovanović et al. (2016), and concluded that this is likely to be due to over-fitting of the model to its training data. In the process, we evaluated several variant methods for dealing with unbalanced data sets and identified one method which consistently improves classifier results without introducing data contamination.

Using the lexical and linguistic features from the same data set, we found that adding simple features based on the discussion structure more than compensated for the removal of the externally grounded semantic features. Additional features capturing semantic similarity and sentiment, which we hypothesised might help to distinguish between levels of cognitive presence, did not consistently improve results further.

Finally, we compared topic models of different sizes, along with various ways of using them to add features to the classifier. We found that a relatively simple approach using 15 topics achieved the best results. Our best model predicts the level of cognitive presence with 56.5% accuracy, a macro-averaged F_1 score of 0.55, and a Cohen's κ of 0.40, indicating a 'fair' level of agreement with the gold-standard human coding (Landis and Koch, 1977).

1.4 Structure of this document

The remainder of this document has the following structure. Chapter 2 sets out the background to the project, reviews previous work in this area, and sets the work in context. Chapter 3 describes the experiments we conducted to address our research questions, and Chapter 4 presents the experimental results. In Chapter 5, we analyse the results, set them in the context of related work, and identify how they provide answers to our research questions and contribute to the field. Chapter 6 concludes the report by outlining some unsolved problems and indicating possible future lines of enquiry.

Chapter 2

Background

2.1 The Community of Inquiry model

Garrison et al. (2000) introduced a model of Community of Inquiry (CoI) to describe the necessary aspects of an online educational environment (Figure 2.1). This widely used model has three dimensions, called 'presences': social, teaching, and cognitive. These can be identified in transcripts of online discussions through the presence of particular words and phrases. The presences can be briefly characterised in the following way:

social: the ability to present oneself to other participants as a 'whole person'.

teaching: the design and facilitation of the learning experience.

cognitive: the ability to "construct meaning through sustained communication".

The authors speculate that **social presence** may be essential to motivate continued participation in an online course with no face-to-face component, alleviating the sense of isolation that could otherwise have a strong negative impact and lead to students dropping out. The model includes three categories for social presence: emotional expression (humour and self-disclosure), open communication (mutual awareness, recognition of the contributions of others), and group cohesion.

Teaching presence incorporates both the overall course design and the ongoing facilitation of the learning experience. Its existence is essential if the educational experience is to lead to genuine learning rather than merely a social exchange of uninformed opinions. Three categories are identified for teaching presence: instructional management (planning), building understanding (facilitation), and direct instruction

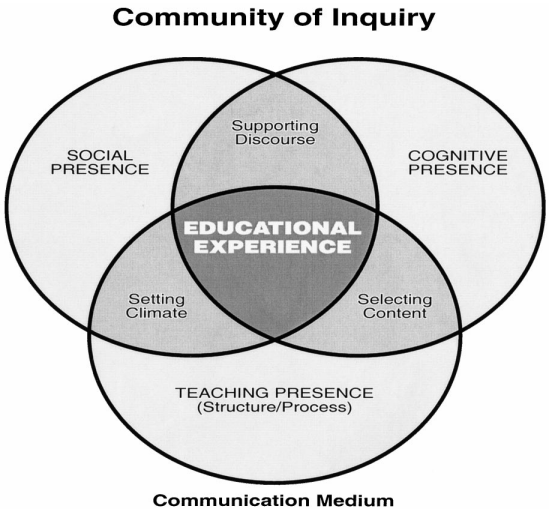

Figure 2.1: Overview of the Community of Inquiry model, taken from Garrison et al. (2000).

(presenting material, assessment, feedback). The students themselves may be expected to take on some responsibility, for example in facilitating discussions or providing peer feedback.

In this work, we will focus on the third of the presences. **Cognitive presence** is further broken down into four levels or phases, representing different stages of the practical inquiry process (Figure 2.2), which dates back to Dewey (1933). These are

Triggering Event: the initial question that sparks a discussion.

Exploration: the phase of the discussion when many new ideas are being considered.

Integration: the phase where ideas begin to coalesce into a more coherent form as connections are identified.

Resolution: the final phase, where a conclusion has been reached, perhaps in the form of a hypothesis that can be tested.

The *triggering event* may come from a teacher or a student; it could be a question based on personal experience. An important aspect of the teacher's role is to focus discussion on those triggers which will lead to learning, discarding out-of-scope distractions.

During the *exploration* phase, students begin to understand the nature of the problem and to discover what information is relevant for solving it. Many different ideas

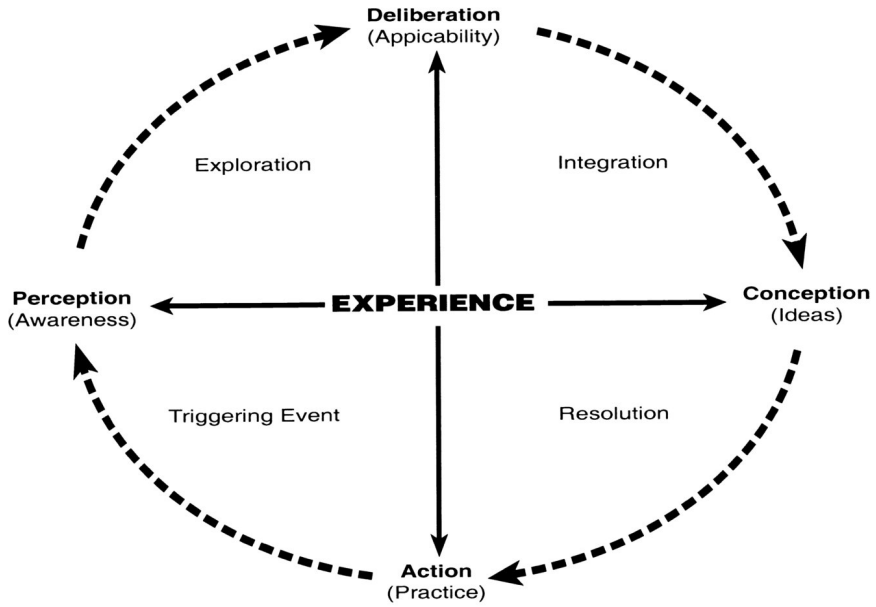

Figure 2.2: The cycle of Practical Inquiry showing the different levels of cognitive presence at each stage, taken from Garrison et al. (2000).

could be explored and then discarded, and the discussion could be wide ranging.

In the *integration* phase, students use their critical thinking skills to synthesise information from the different sources that have been discussed. The authors claim that this phase is the most difficult to detect (for both teachers and researchers) because the "construction of meaning" takes place through reflection and may not be obviously visible in the recorded interaction (Garrison et al., 2001).

The *resolution* phase often involves the participants agreeing on a solution or a course of action. This could take the form of a hypothesis to be tested, and could lead to a new *triggering event*.

Although it is desirable for a discussion to progress through all four phases of cognitive presence and end in *resolution*, not every discussion will do so. Often discussions can become stalled at the *exploration* phase, perhaps due to fear of ideas being rejected (Garrison et al., 2001). In relatively shorter discussions, it becomes less likely that *resolution* will be reached.

Early work looked at manual methods for identifying all three of the CoI presences (teaching, social, and cognitive) in online transcripts using content analysis (Garrison et al., 2000, 2001), and by self-reporting using a 34 item Likert-scale survey (Arbaugh et al., 2008). For content analysis of transcripts, the most appropriate unit of analysis was found to be a single forum message: each message constitutes one conversational

'turn' within a discussion thread. Messages are easy to segment in the data, whereas sentence boundaries are not always clear in informal text. Messages have the additional benefit that the segmentation is generated by the participants themselves and not externally imposed by researchers.

Each message is then coded according to the indicators and socio-cognitive processes listed in Garrison et al. (2001) (Table 2.1). Sometimes a message can show indications of two distinct phases of cognitive presence. The coding scheme indicates that these should be coded with the higher phase (Waters et al., 2015). This is sometimes referred to as *coding up*. In addition, some parts of a discussion will not relate to cognitive presence at all; for example, social greetings and expressions of thanks. These can be left unassigned, or given a distinct label, such as *other*.

Category	Descriptor	Indicators
Triggering Event	evocative	recognizing the problem
		sense of puzzlement
Exploration	inquisitive	divergence—within online community
		divergence—within single message
		information exchange
		suggestions for consideration
		brainstorming
		leaps to conclusions
Integration	tentative	convergence—among group members
		convergence—within a single message
		connecting ideas, synthesis
		creating solutions
Resolution	committed	vicarious application to real world
		testing solutions
		defending solutions
Other		*no indicators of cognitive presence*

Table 2.1: Coding categories, descriptors, and indicators for the CoI model, from Garrison et al. (2001); plus the optional *other* category that indicates an absence of traces of cognitive presence.

The CoI model itself is widely used and has been found to be robust. The transcript-based approach has shown that the coding schemes used to identify the 3 CoI presences, and the 4 phases of cognitive presence in particular, can be applied consistently by different researchers. Only the time and effort required for manual coding has restricted its uptake. For example, to manually code the 1747 messages in the data set used in this study took two experienced coders around 130 hours each (Gašević et al., 2015), and they reached agreement in over 98% of cases (Cohen's $\kappa = 0.97$). The prohibitive time cost means that use of the CoI framework has generally been limited to small research projects rather than wider deployment in educational settings.

Clearly, an automated approach to detecting cognitive presence would allow the CoI framework to be used more widely, perhaps even for real-time monitoring. If we can determine the level of cognitive presence demonstrated in each message, we can use this to track the development of the discussion over time. Ideally every discussion would progress through all four phases, from *triggering event* to *resolution*. If a discussion stalls at the *exploration* phase, an instructor may wish to intervene to encourage students to move on to *integration*. In a similar way, instructors could prompt students to move from *integration* to *resolution* after a suitable interval, or warn if there are too many off-topic (*other*) messages.

2.2 Detecting cognitive presence automatically

Many researchers have proposed methods for automating the coding of cognitive presence. In this subsection, we review some of the methods used and results obtained.

A study using neural networks (McKlin, 2004) to detect the phases of cognitive presence automatically through content analysis used mainly dictionary-based features (182 from the General Inquirer, plus 37 custom categories created for the study), along with 5 features describing the position of the message in the threaded discussion. Human coders each annotated a sample of the data, and a subset of those messages was used to train a neural network. Inter-rater reliability was calculated using the held-out data, both for pairs of human coders, and between the neural network and the consensus value assigned by the humans. The values were found to be comparable, with Cohen's κ in the region of 0.70 and agreement around 81%, indicating that the neural network can code the messages with near-human accuracy. However, the *resolution* class was not used in this study, instead being folded into *integration*, as instances were so rare in the data. Messages that the humans found difficult to code were not

used for training the neural network. It is not stated whether any of the examples in the test data that was used for the reliability comparisons (drawn at random from the same courses) were considered 'difficult'. The most predictive feature was the word count, followed by the number of questions the message contained.

Corich et al. (2006a) presented a proposal for a general-purpose automated content analysis tool (ACAT) that would eliminate the need for manual segmentation and counting in quantitative content analysis studies. Quantitative Content Analysis (QCA) is a general term for the procedure of splitting a transcript into small units (e.g. sentences) and counting how many units fall into each of the categories defined by a formalised coding scheme. A prototype of the tool was described that used word lists to identify category membership and could learn new categories from labelled examples. It was designed to support a range of different coding schemes, including the cognitive presence coding scheme (Garrison et al., 2001). Later work by the same authors (Corich et al., 2006b) used the ACAT tool to label cognitive presence on a small data set that had already been coded manually (Corich et al., 2004). The tool assigned a class label to every unit, whereas human coders had left some units uncategorised to indicate that there were no traces of cognitive presence at any level. If two classes appeared equally probable, the tool simply chose the one that appeared first in the processing loop, rather than using any version of 'coding up'. Although the overall distribution of sentences across the classes was similar, the correlation between the two manual coders was much higher (87%) than the correlation between the manual and automated coding (71%). These results are not directly comparable with other studies of cognitive presence because the data was analysed at the sentence level (484 sentences) rather than labelling the 74 messages directly. Insufficient details of the coding scheme were given to allow for replication.

Another exploratory study used support vector machines (SVMs) to classify cognitive presence, based on standard bag-of-words text classification features such as bigrams, part-of-speech tags, and word dependency triplets (Kovanovic et al., 2014). 10-fold cross-validation was used to assess the usefulness of different features. The best model achieved 58.4% accuracy, with Cohen's $\kappa = 0.41$. However, while the feature space was very large, with over 20,000 features, the data set used (the same as in the present study) had only 1747 data points. This mismatch in dimensions greatly increases the likelihood of a model over-fitting to the data used for training, rather than learning a general pattern that will also apply to new data. In fact, the bag-of-words approach itself ties the model to a specific domain-dependent vocabulary. Additionally,

none of the features capture anything about the discussion *context*. From the definitions of the phases of cognitive presence, we can see that it is very unlikely that a discussion will begin with a *integration* message, or that a *triggering event* message would be followed immediately by a *resolution* message. Thus, adding features relating to the context would be expected to improve the model.

One approach to exploiting the message context was explored in Waters et al. (2015) using conditional random fields (CRFs). Again, the data set used was the same as in the present study. The forum discussion data was structured as 84 separate threads, each addressing a single topic. The forum interface allowed hierarchically branching discussion: messages can have replies, and replies can have their own replies. The key feature of the CRF algorithm used here is that it generates a label *sequence* for an entire sequence of messages, rather than considering messages in isolation. However, it can only handle linear sequences without any branching. Linear 'chains' of messages were therefore extracted from the discussion structure, each chain extending from the 'root' message down to a distinct 'leaf' message. As a result, early messages with multiple replies were extracted and analysed multiple times as part of several chains. Classification features included word unigrams, word and sentence counts, and counts of named entities. Six structural features were also extracted to capture information about the position of each message in the chain. These are the chronological position of the message within the thread, the number of replies it received, two flags identifying the first and last message, and similarity scores related to the previous and next message in the chain. The chains were analysed and coded separately, and then recombined. A majority vote was used to label each message with a final label. The resulting model was applied to held-out data and achieved 64.2% accuracy, with Cohen's $\kappa = 0.482$. The approach used here clearly benefited from using information about the structure of the discussion. However, the linear chain approach does not take account of the full richness of the discussion structure. Another approach would be to (additionally) look at the messages in chronological order by time-stamp, independent of their position in the hierarchical message structure.

The current state-of-the-art results for the data used in this present work come from a study using random forests (Kovanović et al., 2016) where the classifier achieved accuracy of 70.3%, with Cohen's $\kappa = 0.63$ on a held-out portion of the data. The goal was not just to build a high-scoring classifier, but also to gain insight into how cognitive presence is manifested in the discussion through feature analysis. For this reason, the features used in the model were selected because they were based on theory and

had potential explanatory power. Around 100 times fewer features were used than in the authors' previous study on the same data (Kovanovic et al., 2014), described above. None of the bigrams, part-of-speech tags, or dependency triplets from the prior study were retained. Instead, two widely-used text analysis tools were used to extract more meaningful features from the textual data. Word counts derived using the LIWC software package (Tausczik and Pennebaker, 2010), which is based on extensive empirical research, make up 91 of the features. These counts indicate affective, cognitive, social, and perceptual psychological processes, among others; plus four higher-level aggregate measures to indicate i) analytical thinking, ii) social status, confidence, and leadership, iii) authenticity, and iv) emotional tone. Next, Coh-Metrix (McNamara et al., 2014) was used to generate 106 metrics related to text coherence, complexity, readability, and lexical category use. In addition to these lexical and linguistic features, two further new features were defined: i) a feature using LSA similarity to represent internal coherence across the sentences within a message, grounded in a semantic space defined using data from Wikipedia; and ii) a count of the number of relevant named entities, based on the set of named entities found in the computer science category of Wikipedia. The 6 structural features from Waters et al. (2015) were also included – although it was not made clear how the 'next' message was defined in cases where the discussion branched into multiple replies – taking the total feature count to 205. The data was preprocessed to redress the class imbalance before splitting it to create a training set and a test set. Results indicated that deeper levels of cognitive presence were associated with longer messages using more complex language, while *triggering event* messages tended to feature more question marks. These results are similar to McKlin (2004). The number of named entities in a message was also highly predictive, and tended to increase with the level of cognitive presence.

Recent work (Ferreira et al., 2018) on the same data set used topic modelling together with graph-based analysis techniques to explore the development of cognitive presence across different course topics. 15 topics were identified in the data using latent Dirichlet allocation (LDA). These were manually aligned with the 14 topics in the course syllabus, with the final topic capturing discussion of course logistics. The relevance of each topic to each of the messages in the data set was calculated and represented as a matrix of 1747×15 values. These were combined with binary indicators for the 4 levels of cognitive presence and grouped by student. However, the details of the graph-based analysis and its results are out of scope for the present study.

The final study we review adopted the same methodology as Kovanović et al.

(2016), and applied it to discussion forum data written in Portuguese (Neto et al., 2018). Text analytics tools are not as readily available for other languages as they are for English. The Portuguese version of Coh-Metrix reports 48 measures (compared to 108 for English) and no version of LIWC exists for Portuguese. Adapted versions of 24 of the LIWC features, including all those which performed well on English-language data, were extracted by the authors. Word similarity scores and counts of the numbers of named entities were also used, along with 6 features relating to the discussion context. These are the same context features as in Waters et al. (2015): the position of the message within the thread, the number of replies it received, two flags identifying the first and last message, and similarity scores related to the previous and next message. In total, 87 features were used. The data was split to create a training set and a test set, using stratified sampling to maintain the same distribution of the levels of cognitive presence in both sets; then the training data was preprocessed to redress the class imbalance. The smallest class (*resolution*) was thereby increased by a factor of 19, from 34 to 646 data points. The classifier achieved 0.83 classification accuracy and Cohen's κ of 0.72 on the held-out test set, higher than the results in Kovanović et al. (2016). The number of question marks was the most predictive feature, followed by average sentence length. Overall, 45% of the top 20 features matched those identified in Kovanović et al. (2016).

2.3 Limitations of prior work

We now examine two aspects of prior work where we see room for improvement. The first is the area of feature engineering. We note that features that rely on an external knowledge base are unlikely to generalise well to new data. We also find the definitions of some features used in prior work to be unclear or ambiguous.

The second aspect relates to best practice in training and validating predictive classifiers, particularly noting the need to avoid over-fitting when using small data sets and a large feature space.

2.3.1 Classification features relying on an external knowledge base

The use of an external knowledge base to create classification features from messages limits the usefulness of the model for subject domains and languages where such resources are not readily available. Even where a knowledge base is available, it can

make the predictive model less likely to be useful for classifying messages generated in a subsequent run of the same course (let alone an unrelated course) by defining the classification space in too narrow a way. For example, if a particular value in the test data was not seen in the training data, the model may not be able to represent it correctly. This issue is most clearly apparent in systems that use word n-grams. All the words seen in the training data define features in the model, and new words that only appear in the test data are simply ignored. The same sort of problem can arise when using concept information extracted from pages in Wikipedia, although it will not be so severe; the coverage there is broader and there will be some overlap in concepts between similar pages.

Two classification features from the 205 used in the state-of-the-art model (Kovanović et al., 2016) were defined in relation to an external knowledge base: i) a feature using LSA similarity to represent internal coherence across the sentences within a message, grounded in a semantic space defined using data from Wikipedia; and ii) a count of the number of relevant named entities, based on the set of named entities found in the computer science category of Wikipedia.

The LSA similarity score measures the average sentence similarity within a message, defined with respect to a separate semantic space for each discussion. Concepts are extracted from the first message of each discussion thread and used to identify related Wikipedia pages. The semantic space is then constructed using concept information extracted from those pages. A discussion thread on another topic would use different Wikipedia pages to define its semantic space. The similarity between sentences depends entirely on the space used, so the same pair of sentences would be scored differently if they appeared together in multiple topics. It is not clear whether any part of the process relies on human judgement, for example to identify the appropriate Wikipedia pages for a concept. If so, that would make it unsuitable for real-time coding and analysis of new data at scale.

The calculation of the number of relevant named entities is more straightforward, relying on a set of entities extracted ahead of time from the computer science category of Wikipedia. While it might require relatively little manual effort to identify a category for an entire course (rather than for each discussion thread), it still has the effect of reducing the generality of the model. A course in a different domain will clearly have different domain concepts – but using this method, it is likely that it will also have a different *number* of domain concepts, so that the numbers and thresholds learned in the predictive model will not be comparable.

2.3.2 Classification features with ambiguous definitions

The 205 classification features used in the state-of-the-art random forest model (Kovanović et al., 2016) included 6 structural features originally used in a CRF model (Waters et al., 2015), and recent work on Portuguese data (Neto et al., 2018) has also used these. In their original context they were well-defined and their limitations were clear: the 'previous' message was always the parent in the (linear) chain, and the 'next' message was the particular child selected for that chain. The number of replies was simply the number of messages appearing lower down that particular chain.

However, in a context where messages are arranged in an unrestricted branching hierarchy, the correct definition of the 'next' message, or the number of replies, is not obvious. We found that the values assigned to these features in the data were defined inconsistently.

2.3.3 Best practice in training and validating models

"*To build models which generalize well to new data (e.g. for use in student early warning systems), it is important to use evaluation methods which provide accurate estimates of how well models will perform on new sessions of a course, when this is the aim of prediction.*" — Gardner et al. (2018).

A review of studies predicting dropout in MOOCs (Whitehill et al., 2017) noted that the practice of evaluating models on data sampled from the same course on which they were trained was widespread. In a study that used the same set of features to train several predictive models, the results were seen to vary systematically with the evaluation conditions. Using a random split to generate training data and test data from the same course(s) led to accuracy estimates that were significantly over-optimistic compared to using a later run of the course. Other work on replication (Gardner et al., 2018) recommends using data from new sessions of the course for validation, rather than taking a random sample from the same course. This practice avoids the possibility of over-fitting to the training data.

None of the studies we reviewed in Section 2.2 validated their models on data from a different run of the course. One reported correlation between automated and human coding on the whole data set (Corich et al., 2006b). Another (Kovanovic et al., 2014) reported the results from cross-validation. The rest held back a random sample of the data to use for validation of the chosen model.

2.3.4 Best practice for avoiding over-fitting

Recent work addressed the issue of over-fitting when working with a large number of features but a small data set (Kuncheva and Rodríguez, 2018). In this case, there is often a need for parameter selection before building the final model. It is important not to use the held-out test data during this step, as it needs to remain unseen to provide an accurate assessment of the model. In the same way, if the training data will be split into smaller subsets, for example using cross-validation (Section 2.6.2), then only the relevant subsets should be used for parameter selection.

A related issue can arise when using over-sampling to address a class imbalance. We look more closely at the details in Section 2.6.4, but note here that both the studies that used SMOTE to redress class imbalance (Kovanović et al., 2016; Neto et al., 2018) are affected.

2.4 Feature engineering

Prior work on automatically identifying the level of cognitive presence in discussion forum messages extracted a large number of abstract features from the data to use as inputs to a classifier (Waters et al., 2015; Kovanović et al., 2016; Neto et al., 2018). These features can be grouped into the following categories:

1. **Structural features** that capture the context of the discussion, such as the number of replies a message received and the position of a message in a conversation;

2. **Lexical features** that capture low-level properties of the text, such as counts of words and parts of speech;

3. **Linguistic features** that capture higher-level properties of the text, such as indicators of tone (measured by the LIWC tool (Tausczik and Pennebaker, 2010)), and measures of text coherence provided by Coh-Metrix (McNamara et al., 2014);

4. **Grounded features** that are dependent on an external knowledge base, such as counting the number of named entities in each message, based on a set of relevant entities grounded in data from Wikipedia about the subject matter of the course.

In Section 2.3, we looked at potential limitations affecting the structural features and grounded features. We now consider three natural language processing (NLP) techniques that could be used to derive additional features from forum messages: semantic similarity, sentiment analysis, and topic modelling.

2.4.1 Semantic similarity

Semantic similarity features have been used in several prior studies (Waters et al., 2015; Kovanović et al., 2016; Neto et al., 2018) and were often found to be predictive.

Semantic similarity can be understood as a way of tracking the development of ideas over time. As a discussion thread develops, we expect to see many different concepts being explored, expanded, and discarded. A *triggering event* may suggest an initial topic for the discussion. Messages in the *exploration* phase will bring in lots of new ideas and suggestions; some of these will go on to be developed, while others are abandoned. In the *integration* phase, students form connections between the ideas that have already been mentioned. The final *resolution* phase may produce a new hypothesis to test. With this in mind, researchers agree that characterising message content and capturing the semantic similarity between messages is likely to prove fruitful.

Semantic similarity metrics need a frame of reference in which to calculate the similarity score. For example, similarity is often operationalised as the cosine distance between two word (or document) vectors. The problem then becomes defining suitable dimensions and normalisation to calibrate those vectors. If the calibration is defined by the messages in one particular data set, the metric is unlikely to give reliable results for new messages that use different vocabulary. In contrast, using an externally defined, general-purpose framework would allow future messages to be compared in a fully equivalent way. For example, word embeddings (multi-dimensional real-valued vectors) trained on a wide selection of written texts (including user comments) using a variant of the `word2vec` algorithm (Mikolov et al., 2013) would provide such a space. We suggest that this approach is essential if semantic similarity features are to be useful features for a predictive classifier.

2.4.2 Sentiment analysis

The relationship between sentiment expressed in forum posts and student engagement is not yet well understood. It might seem reasonable to suppose that a positive attitude will be correlated with deeper engagement, but while MOOC forum posts tagged as

containing subjective expressions of opinion got more responses from other students, neither subjectivity nor sentiment polarity was strongly correlated with engagement in a model that treated engagement as a latent variable underlying learner behaviour (Ramesh et al., 2013).

Wen et al. (2014) also looked at sentiment in MOOC discussion forums, expressed towards the course itself and its various component parts (lectures, assignments, and peer assessment), and found a significant relationship between expressions of sentiment and course dropout rates. When collective sentiment was measured across all students, a higher ratio of positive to negative terms used in posts about the course was found to correlate with lower dropout rates. Next, looking separately at the ratios of positive and negative words to the total word count in individual posts and discussion threads, the results varied between the 3 different MOOC courses in the study. In one case, students who expressed more sentiment than average (both positive and negative) had a 4% to 5% higher chance of dropping out. In another, exposure to negative sentiment through participation in a discussion thread with a higher rate of negative posts increased the likelihood of dropping out by 6%. In the third, negative sentiment was correlated with a *decrease* in the dropout rate of 20% – in this case, many of the negatively coded words actually related to course content (fantasy novels featuring "evil" and "wicked" characters), rather than indicating anything about the students' opinion of the course. The conclusion of this study was that sentiment alone may not be a good predictor of engagement and should be used with caution.

Another study (Chaplot et al., 2015) investigated the use of sentiment analysis to predict student attrition. Each forum post was tagged with a sentiment score derived by summing the individual sentiment scores for the words in the post, in addition to other features. The decision threshold for the binary outcome was adjusted to match the baseline dropout likelihood, as a way of mitigating the effect of unbalanced classes. Models that included the sentiment score features achieved higher Cohen's κ scores than those that ignored sentiment.

While these studies address predictions of attrition and engagement in general, we consider that sentiment analysis could potentially play a role in distinguishing between the levels of cognitive presence. For example, the *exploration* phase of cognitive presence can be identified by evidence of divergence within a single message or between members of the online community, while the *integration* phase is linked to convergence (Table 2.1). We speculate that, in some cases, the difference between these two could be reflected in the sentiment expressed: a divergent statement might be prefaced by a

negative expression (such as "*I disagree*"), while convergent statements might include positive phrases like, "*I think that's a great point*". There will of course be plenty of cases like the third study mentioned in Wen et al. (2014) where the subject matter under discussion will give misleading results, for example, when discussing bug detection and system failures. We conclude that sentiment analysis features might prove useful to a classifier for cognitive presence, in combination with other features.

2.4.3 Topic modelling

Topic modelling using latent Dirichlet allocation (LDA) (Blei et al., 2003) is a widely-used approach to modelling documents through the automatic identification of 'topics', which are probability distributions over the words in the vocabulary. It is an unsupervised approach, so the topics themselves are derived from the data in the corpus during the modelling process without the need for an external knowledge base. Once a topic model has been trained, it can be used to classify new documents.

A formal graphical model of LDA is shown in Figure 2.3.

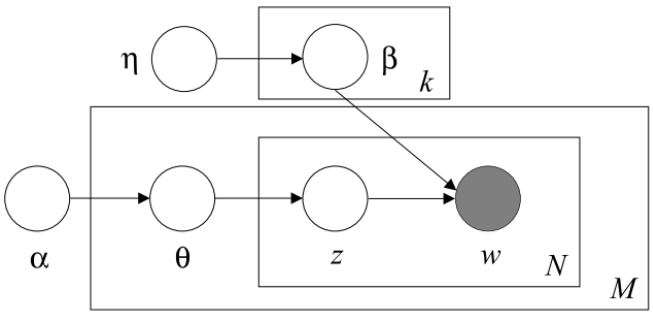

Figure 2.3: Graphical model of LDA with smoothing, from Blei et al. (2003). A set of k topics is defined by the probability distribution matrix β. The hyperparameter η adds a prior on β, ensuring that unseen words do not have zero probability (a form of smoothing). A corpus of M documents is generated by first assigning each document its own probability distribution over topics, θ, sampled from a Dirichlet distribution parametrised by α. For each word in the document, a topic z is chosen from θ and a word w is chosen from the vocabulary, given z and β.

Topics do not form a partition over the words: that is, a word can appear in multiple topics, and the same word can even appear with high probability in more than one topic. A topic model treats a document (in our case, a forum message) as an unordered

collection of words and assigns it a score for each topic. These scores form a probability distribution, so all the scores will lie in the range $[0, 1]$ and the scores across all topics for a given message must add to 1.

Recent work (Ferreira et al., 2018) used topic modelling with the same data set as the present study, with the goal of automatically tagging the 14 topics from the course syllabus for further investigation. Although many of the automatically identified topics will undoubtedly relate to specific details of course content in the training data, it is possible that some topics will capture word distributions that might correlate with the phases of cognitive presence – or with its absence (the *other* class). These 'non-syllabus' topics might be expected to be present in new data, whereas topics relating more closely to the syllabus would be less likely to appear again elsewhere. Knowing which messages contain non-syllabus topics with high probability would then be a useful predictive feature in a classifier.

2.5 Text classification techniques

Classification methods used in prior work include neural networks (McKlin, 2004), support vector machines (Kovanovic et al., 2014), conditional random fields (Waters et al., 2015), and random forests (Kovanović et al., 2016; Neto et al., 2018).

Neural networks are very powerful models that have seen a resurgence in recent years. They are now used successfully in many commercial text processing applications such as machine translation. They are a 'black-box' method with little explanatory power, and they work best on large data sets.

Support vector machines (SVMs) are general-purpose machine learning algorithms for supervised learning problems (that is, problems where there is already some labelled data). They work particularly well on high-dimensional, sparse data. SVMs are often used in text classification, for applications such as identifying spam emails (Sculley and Wachman, 2007).

Conditional random fields (CRFs) are a more specialised class of statistical model used for sequence modelling and pattern recognition. They consider an entire sequence of inputs together, producing a sequence of outputs, and have been used for tasks including part-of-speech tagging and document summarisation (Waters et al., 2015).

The two most recent works we examined in Section 2.2 both use random forests, citing their excellent performance, particularly on small data sets, and their ability to produce interpretable results, allowing us to evaluate the contribution each feature

makes to the classifier's performance.

Random forests (Breiman, 2001) are a type of ensemble method using decision trees. Random forests are well suited to problems where the number of features is large but the size of the training data is relatively small (Kuncheva and Rodríguez, 2018), since irrelevant features are simply ignored.

A decision tree is a simple machine learning model which begins with all the data at the 'root node' and recursively splits each node into two 'child nodes' based on the value of a single feature, in order to produce the best separation between the output classes. A node that only contains data points from a single output class cannot be split further and is considered to be 'pure'. The Gini impurity index tracks the relative purity of nodes, and the goal is to choose the best split to minimise impurity. At inference time, new data can be classified by simply reading off the decision threshold at each split and following the path down the tree until a leaf node is reached. The predicted outcome for that data point is then the class assigned to the leaf node by the training data.

Random forests average the results over a large number of decision trees to reduce variance and prevent over-fitting to the training data. In a random forest model, each decision tree is built using a different random sample of the rows in the training data. In addition, every time a tree node splits to form child nodes, a different random subset of the available features is made available as candidate features for the split. This means that even a feature that is highly predictive will not be used in every tree, so the predictive power of the other features will also be explored. When the model is applied to new data, the predicted output for each data point is the mode value of the output from all of the trees in the forest.

There are two free parameters in a random forest: the number of trees to build, and the number of features to use at each split. The `caret` package in R provides support for tuning the second of these, the `mtry` parameter, by trying various values inside a cross-validation loop and then building a final random forest using the best value.

One measure of the relative importance of different classification features in a random forest is *Mean Decrease Gini (MDG)*. It is calculated as the mean of the decrease in the Gini impurity measure for all tree nodes where the feature is used.

2.6 Evaluation methods

2.6.1 Outcome metrics

Several outcome metrics can be used to measure the power of a predictive classifier. We will define them with reference to a binary classification example (Table 2.2), and data for a worked example with 3 outcome classes (Table 2.3).

	Predicted	
Actual	Positive	Negative
Positive	True positives (TP)	False negatives (FN)
Negative	False positives (FP)	True negatives (TN)

Table 2.2: Defining the true positives, false positives, true negatives, and false negatives in a binary classification task. For classes with more than two outcomes, each class must be considered separately. All classes other than the target class are grouped together as 'false' values.

	Predicted			
Actual	Class A	Class B	Class C	Total
Class A	3	0	1	4
Class B	0	3	1	4
Class C	1	0	1	2
Total	4	3	3	10

Table 2.3: Example data with three outcome classes.

Accuracy is one of the commonest measures: it is simply the proportion of the data points that are classified correctly (Equation 2.2). Using the example data from Table 2.3, the accuracy is the sum of the values on the diagonal, divided by the total number of values: $(3+3+1)/10 = 0.7$, or 70%.

The choice of outcome metric can completely alter the perceived success or failure of an experiment (Hofman et al., 2017). For example, an accuracy of 77% in predicting dropout from a MOOC may sound good, but it is much less impressive if only 23%

of students actually complete the course. In that case, the same accuracy score can be achieved by a simple majority-class baseline that predicts that *every* student will drop out (Chaplot et al., 2015). When dealing with unbalanced classes, as we are here, two alternative measures are often used which are more informative than accuracy: Cohen's κ, and the macro-averaged F_1 score.

Cohen's κ measures agreement between pairs of annotators on a task involving assigning data points to mutually exclusive categories or classes (Equation 2.1). It discounts the potential for agreements due to chance and produces a robust estimate of inter-rater agreement, and can thus be used to assess the overall reliability of the coding scheme. Tasks involving human judges usually aim to achieve Cohen's κ scores above 0.70 (Landis and Koch, 1977).

$$\text{Cohen's } \kappa = \frac{P(A) - P(E)}{1 - P(E)} \tag{2.1}$$

To calculate Cohen's κ, we first calculate the proportional agreement, $P(A)$, which is the same as accuracy. Next, we calculate $P(E)$, the probability that the judges agreed by chance, using the pooled marginals. A worked example of calculating Cohen's κ is shown in Figure 2.4.

Macro-averaged F_1 is found by calculating the F_1 score (the harmonic mean of precision P and recall R, Equations 2.3 – 2.5) separately for each of the classes in turn, and then taking the average (Equation 2.6). In this way, good performance across all classes is rewarded more than high performance only on the larger classes (Table 2.4). It is an appropriate metric to use when the correct identification of instances of all classes is equally important.

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN} \tag{2.2}$$

$$P = \frac{TP}{TP + FP} \tag{2.3}$$

$$R = \frac{TP}{TP + FN} \tag{2.4}$$

$$F_1 = \frac{2PR}{P + R} \tag{2.5}$$

$$\text{macro-averaged } F_1 \text{ score} = \frac{1}{|C|} \sum_{c \in C} F_{1c} \tag{2.6}$$

Sample data:

	Judge 2		
Judge 1	yes	no	Total
yes	300	20	320
no	10	70	80
Total	310	90	400

Calculations:

$$
\begin{aligned}
P(A) &= (300+70)/400 &&= 0.925 \\
P(yes) &= (320+310)/(400+400) &&= 0.788 \\
P(no) &= (80+90)/(400+400) &&= 0.213 \\
P(E) &= P(yes)^2 + P(no)^2 &&= 0.665 \\
\text{Cohen's } \kappa &= (0.925-0.665)/(1-0.665) &&= 0.776
\end{aligned}
$$

Figure 2.4: Worked example of calculating Cohen's κ using judgements from two independent coders on a binary classification problem, adapted from Manning et al. (2008).

	P	R	F_1
Class A	0.75	0.75	0.75
Class B	1.00	0.75	0.50
Class C	0.33	0.50	0.40
macro-averaged F_1			0.67

Table 2.4: Precision, recall, F_1, and macro-averaged F_1 for the data in Table 2.3.

2.6.2 Cross-validation

It is common practice in machine learning to split the data set into a training set, used to build the model, and a held-out test set that is used to estimate how well the model will classify new data. Often, the training data is subdivided further, with some of the data used as a validation set to allow comparison of different values for model parameters, before the best model is tested on the held-out data.

When the data set is small, a method called cross-validation can be used instead of creating a validation set. Here, the training data is divided at random into smaller subsets, called *folds*. 10 folds is a common choice. One of these folds is kept back; the model is trained on the rest and evaluated on the final one. This step is repeated until all of the folds have been used for evaluation. The process can be run again using a different random split into folds. The average across the different trials is a good estimate of the expected score for a model created using the whole of the training data. The R library `caret` provides extensive support for this process.

Cross-validation is useful for comparing different models, but a final evaluation on held-out test data is still needed to estimate how well the best model will perform on completely new data.

2.6.3 Dealing with unbalanced classes

Unbalanced classes often lead to poor classifier results. If a model has seen very few examples of one of the classes during training, it will be harder for it to identify that class in new data. One way to rebalance the classes is to down-sample larger classes by discarding data points until they are closer to the size of the smallest class. However, as the data sets we are considering in this study are already small, we feel that down-sampling would discard an unacceptably large proportion. Another approach involves

up-sampling the smaller classes, for example by duplicating elements, until the number of instances reaches the size of the largest class. When classes are very unbalanced, as is usual with cognitive presence, such duplication would give us many identical copies of each element in the smallest class.

A better alternative is to create synthetic data points with values that correspond to the distribution of the existing instances in the smaller classes, without being exactly identical to any of them. A popular method is SMOTE: Synthetic Minority Over-sampling TEchnique (Chawla et al., 2002).

For each original data point in turn, SMOTE selects one of its k nearest neighbours and creates a new data point whose values lie between the original point and the neighbour (Figure 2.5). The value of k can be configured: 5 is a typical default. This process is repeated for the required number of iterations. Thus, the number of new points created is always a multiple of the size of the original data set.

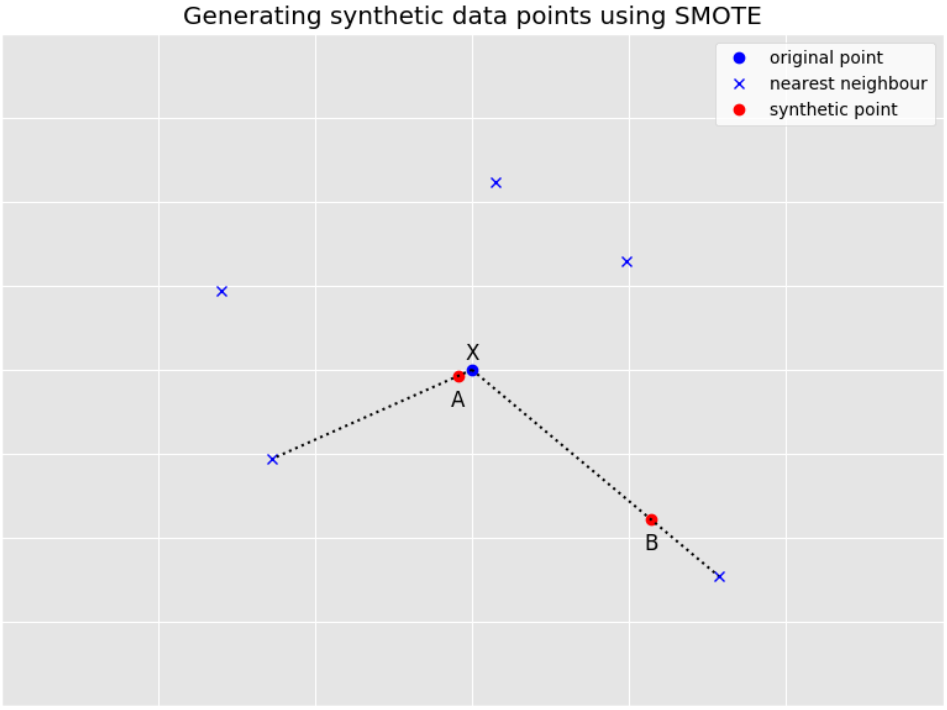

Figure 2.5: Example of synthetic data point creation in 2 dimensions. Each new point is created somewhere between an existing point X and one of its 5 nearest neighbours. Here, point A is very similar to the original data point, while point B is closer to the chosen neighbour.

The original method (Chawla et al., 2002) is only designed for a 2-class problem,

but it can be extended to multiple classes using a one-versus-many approach, where each class in turn is contrasted with the largest class (Figure 2.6). We developed two alternative versions of multi-class rebalancing as wrappers around the existing R implementation of 2-class SMOTE in the `DMwR` library. The implementation of these is described in detail in Section 3.2.

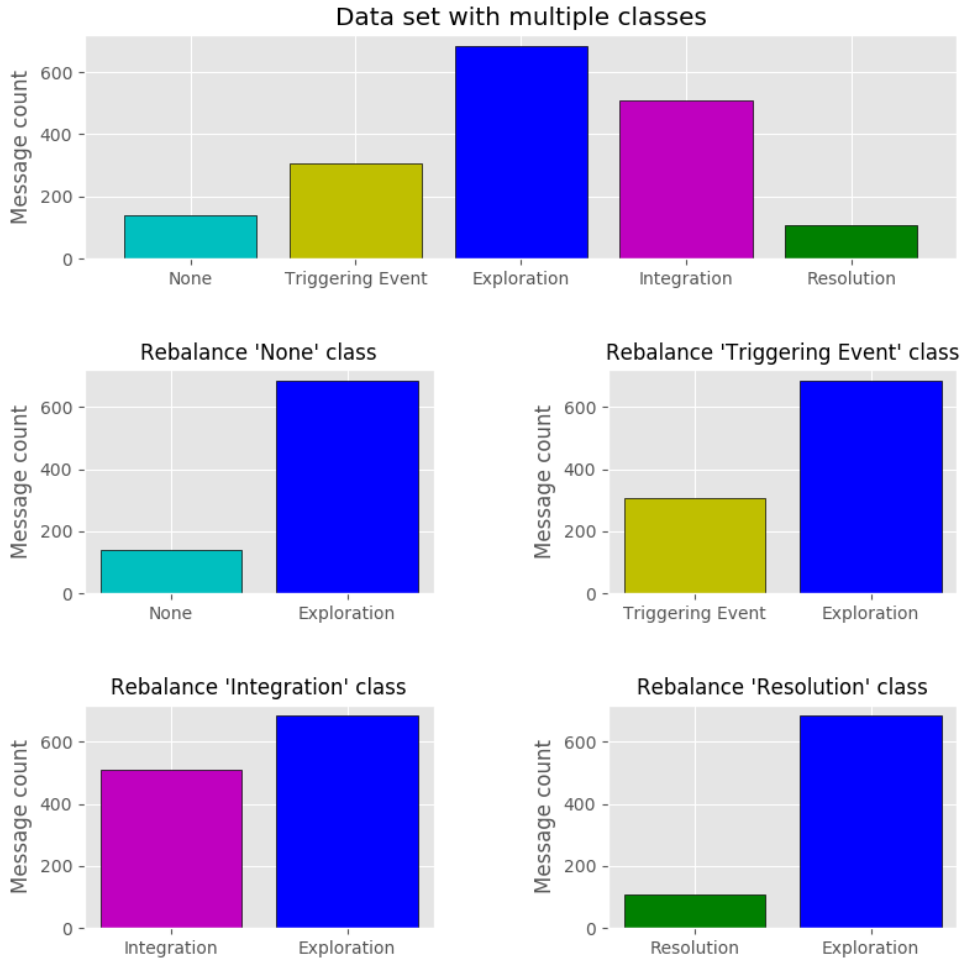

Figure 2.6: To rebalance a data set with more than two classes, we can run SMOTE repeatedly on pairs of classes to generate new synthetic data points in the smaller class, then recombine the outputs.

2.6.4 Class rebalancing and data contamination

In the case where up-sampling uses simple duplication of minority class data points, it is clear that there is a risk of data contamination if the split into training and test sets is not done carefully. If the up-sampling step takes place before the initial split, some

of the data points in the test set, which are meant to be unseen data, may actually be duplicates of elements in the training set. This will lead to artificially high estimates of the power of a model: it is easy for a model to predict the correct values for the test data when those data points have already been seen during training. On genuine held-out data, prediction performance can be expected to be worse.

The same fundamental issue arises when synthetic data points are created. If the test data contains synthetic data points that were constructed from elements in the training set, then it is again likely that estimates of the model's power will be artificially high. For example, point *A* in Figure 2.5 is very similar to the original data point from which it was created. If the original point was part of the training data, and point *A* was in the test data, even a poor model would have a good chance of predicting it correctly.

A more subtle case of data contamination due to over-sampling is explored by Kuncheva and Rodríguez (2018) and is highly relevant to our situation. When training classifiers for high-dimensional data with few instances, a common error is to use the same data for selecting features (or tuning other model parameters) as for evaluating the final model. Again, this leads to over-optimistic, heavily biased results. Instead, the class rebalancing step needs to take place separately for each fold of the cross-validation, using only the training data for that fold[1]. Thus, the solution is to do both tuning and model building *inside* the cross-validation loop, so that the same data is used for both steps. Once the parameter tuning is complete, a final model can be built using the full training partition as usual.

The effect of this second error is different from the first. Whereas data contamination between the training data and the 'held-out' test data leads to overly positive evaluation results for the final model, contamination introduced at the parameter tuning step will lead to a sub-optimal model being selected.

[1] As less data is available at this stage, it may be better to avoid down-sampling larger classes and simply create new instances of the smaller classes until they match the size of the largest class.

Chapter 3

Methodology

In the Kovanović et al. (2016) model, the full data set was rebalanced using SMOTE before being split into training and test sets using a stratified random sample. Over-sampling before splitting can lead to data contamination, over-fitting and misleading results (Sections 2.3.4 and 2.6.4). Best practice is to perform the class rebalancing step inside the cross-validation loop. Using a random split can also lead to overly-optimistic results, and best practice is to test the model on a later run of the course (Section 2.3.3). We addressed these points with our first research question:

Research Question 1
How do the results from the state-of-the-art model (Kovanović et al., 2016) compare with a replication study using current best practice for handling unbalanced classes and splitting data into training and test sets?

The discussion in Section 2.3.1, showing that using an external knowledge base to define classification features limits the usefulness of the model, informed our second research question:

Research Question 2
Can we achieve similar predictive performance without relying on an external knowledge base for semantic annotation?

Topic modelling, a technique which allows us to extract latent 'topics' from a transcript, was examined in Section 2.4.3. We speculate that these topics might provide a powerful way of generating new features for the classifier that relate to indicators of cognitive presence – leading to our final research question:

Research Question 3

Can topic modelling give us new insights into cognitive presence?

Work to address our research questions was structured around three experiments. **Experiment 1** was a replication study designed to address Research Question 1. We recreated the state-of-the-art predictive model from Kovanović et al. (2016) using the original data and methodology, then applied insights from best practice when working with small data sets (Kuncheva and Rodríguez, 2018) to compare different algorithms for dealing with the unbalanced classes in the outcome variable. Building on these results, we explored the effect of splitting the data by course session instead of using a random split, in line with best-practice recommendations for replication studies in an educational context (Gardner et al., 2018).

In **Experiment 2**, we addressed Research Question 2. We omitted the classification features that relied on an external knowledge base, recalculated the structural features to avoid prior inconsistencies (Section 2.3.2), and added new features based on context, semantic similarity, and sentiment analysis.

Experiment 3 addressed Research Question 3. We generated three different topic models with different numbers of topics. We used these to derive several sets of new features for the classifier and explored the predictive power of different combinations of topics and features.

We begin by describing the data set used in all 3 experiments.

3.1 Description of the data set

The work presented here makes use of the same data that was used in the Kovanović et al. (2016) study that achieved state-of-the-art results, allowing us to compare our results directly. The data was collected from a Masters-level software engineering course that ran at a Canadian university between 2008 and 2011. It was a fully online distance-learning course. The total number of students across all 6 offerings of the course was 85, with a median of 14 students per session (Table 3.1).

The data consisted of 1747 messages posted on a class discussion forum during weeks 3–5 of each 13-week course offering. Each message was annotated with 205 classification features, as described in Section 2.3. There was a large variation in the number of messages generated per session (mean = 291.2, SD = 192.4, median = 240). All the messages were manually annotated with the level of cognitive

Session	Student count	Message count
Winter 2008	16	212
Fall 2008	24	633
Spring 2009	12	243
Fall 2009	9	63
Winter 2010	15	359
Winter 2011	13	237
Average (SD)	14.8 (5.1)	291.2 (192.4)
Median	14	240
Total	85	1747

Table 3.1: Statistics for the 6 offerings of the course.

presence by expert human annotators (98.1% agreement, Cohen's $\kappa = 0.974$). Examples of messages corresponding to each of the four levels, plus the *other* class used for messages showing no evidence for any phase of cognitive presence, are shown in Table 3.2.

The discussions were structured around individual video presentations that students recorded and uploaded. Each discussion thread began with a message giving the URL to the video along with information about the research paper being presented. This was followed by asynchronous text-based discussion focussed on that presentation. Participation in the discussion counted for 10% of the final course mark. More details about the assignments and the course structure can be found in Gašević et al. (2015).

The number of messages varied significantly across the levels of cognitive presence (Table 3.3). This imbalance is natural, as a single *triggering event* message is expected to lead to multiple messages in the *exploration* and *integration* phases, and a smaller number (or perhaps none at all) in the *resolution* phase. The discussions took place relatively early in the course and were designed to prepare students for their individual research projects, so it is not surprising that relatively few messages indicated that a student had reached the *resolution* phase of cognitive presence. A simple baseline classifier assigning the majority class to every message would achieve 39% accuracy.

Cognitive Presence	Message
Other	Hi [NAME] Thanks for an excellent explanation! [NAME]
Triggering Event	Hi [NAME] Liked your presenation. I have to admit some of it was over my head. I've never looked at neural networks before but sounds quite interesting. In your comments you mentioned that missing some page clones was less of a concern than having false positives. Can you elaborate on that please? I'm not very strong in this topic area and it seems like you know what you are talking about Rgds [NAME]
Exploration	Hi [NAME] Only two experts were used and they were used for both case studies. I think the problem with the manual process is one of volume. There are just too many combinations of pages to examine when tying to do the maintenance in a timely manner. The authors indicate that reverse engineering methods and tools are a must. I have to agree. Regards [NAME]
Integration	I was interested in some work to implement pattern recognition algorithms to develop an anti plagiarism software that is more capable than purely cite paragraphs that have been "copied pasted"...but also to mine for ideas copied from other papers even though they might be written in a "different" way.
Resolution	Hi [NAME] Actually I agree with your remarks there. Being a webdeveloper I would prefer leaving cloned pages rather than investigating non cloned pages that were marked as cloned. The cost of the latter is higher for me as a web developer

Table 3.2: Examples of messages indicating cognitive presence at each of the four levels, and none. All these messages are from the same discussion thread. Students' names were replaced by the `[NAME]` token but spelling errors were not corrected.

Cognitive Presence	Count	Percentage
Other	140	8.01%
Triggering Event	308	17.63%
Exploration	684	39.15%
Integration	508	29.08%
Resolution	107	6.12%
All	1747	100.00%

Table 3.3: Breakdown of messages by level of cognitive presence.

3.2 Methods for rebalancing unbalanced classes

As the number of data points belonging to each outcome class (i.e. each phase of cognitive presence and *other*) is highly unbalanced (Table 3.3) and we know that unbalanced data can cause problems for classification techniques, we used SMOTE (Section 2.6.3) to rebalance the classes in our training data in all our experiments.

We implemented two new R methods for rebalancing our data. These make use of the existing SMOTE implementation in the `DMwR` package for 2-class data and extend it to handle the multi-class case. The first method generates the optimum number of full generations of synthetic points for each of the smaller classes to bring it as close as possible to the size of the original largest class. This means that if the classes are already close in size, no new data points are created. We call this method *SMOTE multi*, since the number of synthetic data points is always a multiple of the original class size (Figure 3.1). It is inspired by prior work (Neto et al., 2018), which used SMOTE to rebalance the training data in this way.

Our second approach starts by creating enough full generations of synthetic data points to match or exceed the target size, then uses random selection within the final generation (only) to make the sizes exactly equal. We call this *SMOTE exact*, because the number of synthetic data points is controlled such that the final class sizes match exactly (Figure 3.2). This version is closer to the approach taken in Kovanović et al. (2016), where all the classes were balanced to be the same size, except that we do not down-sample the larger classes.

We note that this second approach can also be used to generate a data set where the size of each class is set to a predefined target size, for example to allow us to construct

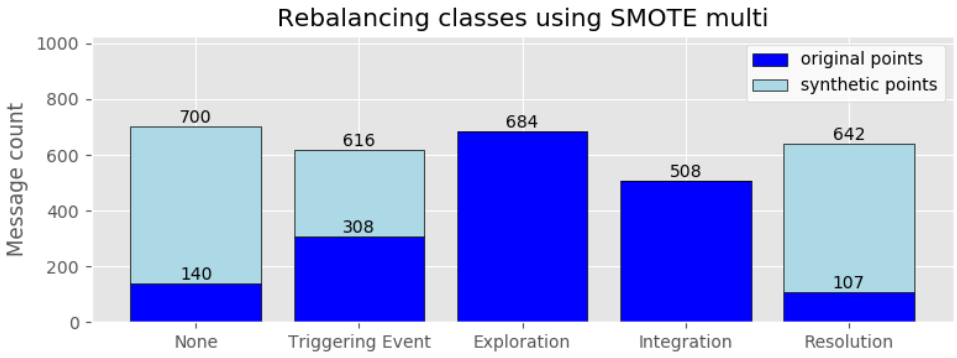

Figure 3.1: The SMOTE multi algorithm generates synthetic data points in multiples of the original class size, to make the final size as close as possible to the majority class. This example demonstrates its behaviour when applied to the full dataset.

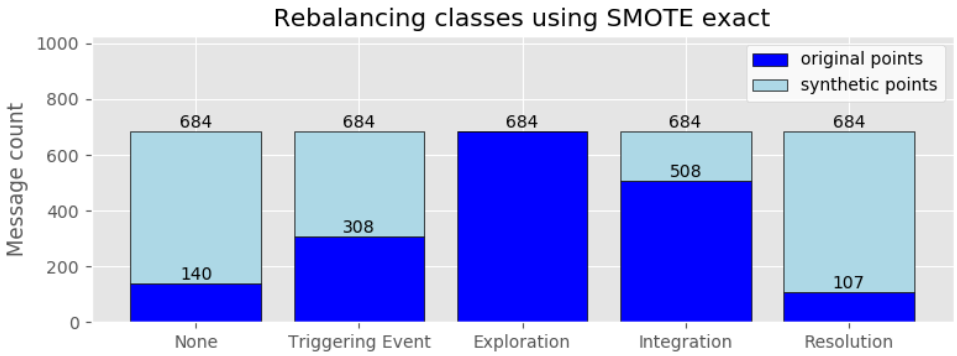

Figure 3.2: The SMOTE exact algorithm generates the exact number of synthetic data points needed to make every class the same size. By default, it will make all the classes the same size as the largest class.

a data set that is the same size as the original but with balanced classes (Kovanović et al., 2016). If any classes are already larger than the target size, they are down-sampled randomly. The algorithm will work correctly even if the target size is greater than all the existing class sizes, so long as it is smaller than the sum of all but the largest class. In the work reported here, we used the size of the largest class as the target.

The SMOTE method as described in Chawla et al. (2002) and implemented in R only works correctly with continuous variables. For categorical variables with multiple classes, the correct approach would be to take a majority vote from the k nearest neighbours. There are no multi-class categorical variables in our data, and we assume that simple rounding will give a reasonable result for binary variables.

Another consideration is the need to preserve the data type of each field. In particular, if the original field is an integer type (for example, a word count), then after generating the new synthetic data points, we round the values and cast the field to integer again to ensure that the data type remains the same.

The final issue to note with SMOTE and similar methods is that where several fields in a data set are related, there is no simple way to maintain that relationship for the newly created data points. For example, in our data, a message cannot be the last in its thread if it has a non-zero number of replies – but a synthetic data point could conceivably be created with inconsistent values for these features. We do not make any attempt to correct for this type of error.

As described in Section 2.6.3, applying the class rebalancing step inside the cross-validation loop is the recommended best-practice approach (Kuncheva and Rodríguez, 2018). We wrote additional R wrappers to allow our methods to be used in this way.

3.3 Experiment 1: Replication study

Experiment 1 was a replication study designed to address Research Question 1. We used exactly the same features as the study we are following and varied the approach used for splitting and rebalancing the data.

3.3.1 Baseline replication

We began by replicating the state-of-the-art model (Kovanović et al., 2016), using the same methodology and the same data. The full data consists of 1747 messages and 205 features. The data was preprocessed to remove the class imbalance, increasing the size of the smaller classes by using SMOTE to create synthetic instances, and down-sampling the larger classes. This produced a new data set of the same size as the original, but with each class equal in size (Figure 3.3).[1]

The balanced data was then split into training and test partitions in the ratio 75 : 25, using stratified sampling to ensure the same distribution of phases in the two partitions. The training data was used to build a series of 20 random forests of 1000 trees, exploring different settings for the `mtry` parameter that controls the number of features available as candidates at each split point. The specific values to be tested are automatically determined by the `caret` library based on the number of features in the model;

[1] The rebalanced data set from the prior study was made available to us, so we used that directly rather than recreating it.

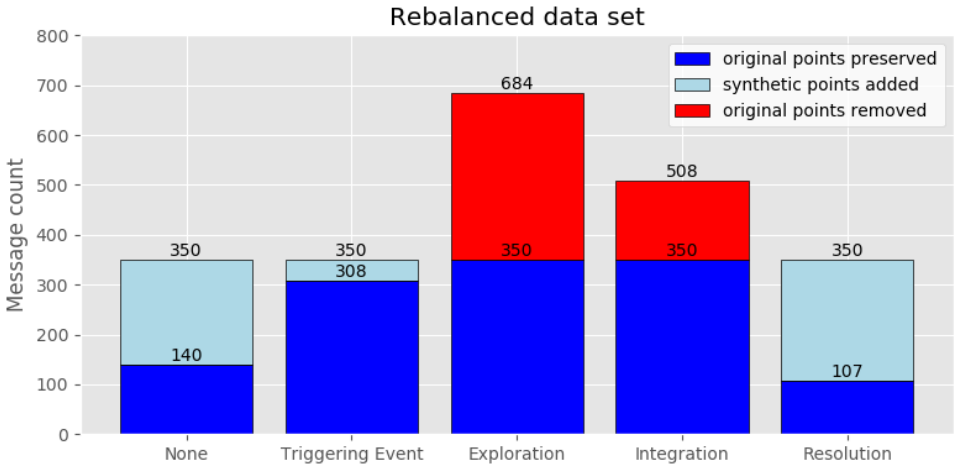

Figure 3.3: In Kovanović et al. (2016), the full data set was rebalanced using SMOTE. Larger classes were down-sampled and smaller classes were up-sampled to create a new data set of the same size as the original, but with all classes equally sized.

here, they were [1, 12, 23, 34, 44, 55, 66, 76, 87, 98, 108, 119, 130, 140, 151, 162, 172, 183, 194, 205]. 10-fold cross-validation, repeated 10 times, was used to select the best performing parameter value. A final random forest model was built using this value on the full training set. The overall accuracy of the final model and the importance of each variable in the model was then assessed using the test data (Table 3.4).

3.3.2 Avoiding data contamination

In Section 2.6.4 we saw how over-sampling can sometimes lead to data contamination, meaning that classifier results are overly-optimistic. Since the data in our baseline replication was split into training and test sets *after* the creation of the additional synthetic data points in the minority classes, this is a real danger.

To address this concern, we split the original unbalanced data into a new training and test set using stratified sampling in the same way as before (75 : 25), and rebalanced the training set only. We used our two SMOTE variants, SMOTE multi and SMOTE exact, to produce two further training data sets (Figure 3.4).

We used the unbalanced training data and the two rebalanced data sets to train three new classifiers using the same procedure as before: 20 random forests of 1000 trees, testing the same set of values for the `mtry` parameter as before, using 10-fold cross-validation, repeated 10 times. In the same way, we trained two further classifiers that performed the class rebalancing step *inside* the cross-validation loop, following

	Training		Test	
Cognitive Presence	Count	Percentage	Count	Percentage
Other	263	20.06%	87	19.95%
Triggering Event	263	20.06%	87	19.95%
Exploration	263	20.06%	87	19.95%
Integration	263	20.06%	87	19.95%
Resolution	263	20.06%	87	19.95%
All	1315	100.00%	435	100.00%

Table 3.4: Over-sampling the data before using a stratified random sample to split it into training and test gives an equal number of examples in each of the phases of cognitive presence in both sets.

best-practice (Kuncheva and Rodríguez, 2018).

We took care to ensure that the same random seed was used for initialising SMOTE every time the same cross-validation fold sub-sample was used, to allow a fair comparison of the different values of the `mtry` model parameter. All five models were evaluated on the same held-out test data (Table 3.5).

	Training		Test	
Cognitive Presence	Count	Percentage	Count	Percentage
Other	105	8.01%	35	8.01%
Triggering Event	231	17.62%	77	17.63%
Exploration	513	39.13%	171	39.15%
Integration	381	29.06%	127	29.08%
Resolution	81	6.18%	26	6.12%
All	1311	100.00%	436	100.00%

Table 3.5: Using a stratified random sample to split the data into training and test gave the same distribution of phases of cognitive presence in both cases, matching the distribution in the original data.

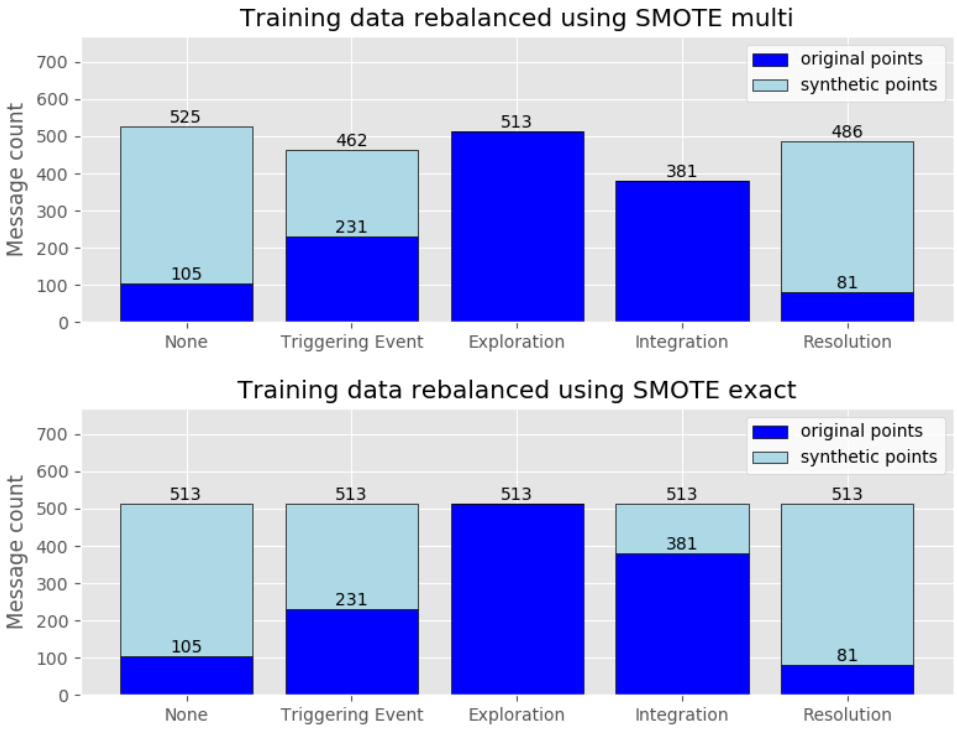

Figure 3.4: The training data was selected using stratified sampling, then rebalanced using two variants of SMOTE. The class balance in the test data was not altered.

3.3.3 Splitting by course offering

Using stratified sampling to generate the training-test split allowed us to directly compare our results to prior work (Kovanović et al., 2016) when using a different approach to rebalancing the unbalanced classes. However, recent work on replicability of results on MOOCs (Gardner et al., 2018) indicates that it is better to hold out the most recent offering of a course for testing. Even when the instructors are the same, a course changes every time it runs. A useful model needs to be general enough to make predictions on future runs of the course.

In addition, when adding features based on topic modelling (Experiment 3), it is important that the data used for testing the final model is not also used to create the topics themselves. We assume that many of the same terms and phrases that appear in messages will be reused in replies to those messages. By using a random split to divide the data into training and test sets, the generated topics would effectively be based on (some) data from the test set.

Therefore, we used the final course offering (Winter 2011) as test data, and the five earlier offerings for training (Table 3.6). We assessed the effect of using this

best-practice data split instead of the stratified sample by training and evaluating three classifiers: the first used the unbalanced training data directly, while the other two performed the class rebalancing step inside the cross-validation loop using each of our SMOTE variants.

The same settings were used as in the previous experiments: 20 random forests of 1000 trees, testing the same set of values for the `mtry` parameter as before, using 10-fold cross-validation, repeated 10 times.

	Training		Test	
Cognitive Presence	Count	Percentage	Count	Percentage
Other	112	7.42%	28	11.81%
Triggering Event	280	18.54%	28	11.81%
Exploration	608	40.26%	76	32.07%
Integration	425	28.15%	83	35.02%
Resolution	85	5.63%	22	9.28%
All	1510	100.00%	237	100.00%

Table 3.6: Using a session-based split and keeping back the final course offering for testing, the distribution of phases of cognitive presence across the training and test set differed. The test set was not as unbalanced as the training data.

3.4 Experiment 2: No external knowledge base

Experiment 2 addressed Research Question 2. We omitted the features from the original data set that relied on an external knowledge base and derived new features to replace them.

3.4.1 Features omitted

We removed the two grounded features (named entity count and message coherence) that depend on data from Wikipedia (Table 3.7), because our goal was to remove reliance on external data sources in order to improve the generalisability of the model (Section 2.3.1).

Grounded feature	Description
named entity count	The number of named entities in each message, based on a set of named entities related to the subject matter of the course identified in Wikipedia.
message coherence	The relative coherence of the sentences in a message, calculated using latent semantic analysis (LSA), grounded in a semantic space defined with reference to an external knowledge base.

Table 3.7: Grounded features used in the baseline replication but discarded from the data used in further experiments.

Six structural features (number of replies, message depth, similarity to previous message, similarity to next message, first message, and last message) that had been carried over from an earlier study (Waters et al., 2015) were also removed because they seem to have been calculated inconsistently in this data set (Section 2.3.2), and we felt we could not rely on them (Table 3.8).

3.4.2 Structural features added

We extracted nine structural features from the data to use as features in the classifier (Table 3.9). We wanted to capture the structure of the discussion in a richer way, beyond message depth and the binary indicators for the first and last message.

The threaded nature of the forum means that every message can receive multiple replies, and replies can themselves receive replies. A new reply can be added at any level in the chain at any time. Without detailed analytics to tell us which messages a student has actually read, we need to make some assumptions. A message posted as a reply to another message can be expected to relate to that message in a meaningful way. Similarly, the impact of a message on the discussion can be measured, not only by the number of replies it gets, but perhaps also by the total count of replies-to-replies: that is, counting all the descendant messages.

We expected that chronological order would also be a valuable indicator, so we redefined the ‘previous’ and ‘next’ message by time-stamp order within the discussion thread, and used the position within that overall order, from both beginning and end, as additional features – richer versions of the existing first and last message indicators.

Structural feature	Description
number of replies	The number of replies a message received – not always consistent with other information about the discussion structure.
message depth	The depth of the message within the nested discussion structure – not always consistent with other structural information.
similarity to next/ previous	The cosine similarity between a message and the next/ previous message, based on a linear sequence of messages from root to leaf. With multiple replies, the same message appears in more than one linear sequence, and it is unclear how these values were calculated.
first/ last message	binary indicators for the first and last message in a discussion. These flags did not always correspond with other information about the discussion structure.

Table 3.8: Structural features used in the baseline replication but removed or redefined before further experiments.

Thus, we redefined three features (message depth, first message, and last message), added two features for replies (number of direct replies and total number of replies), and another three features to capture the chronological order of messages within a discussion by time-stamp (position from start, position from end, and fractional position). A final feature (discussion size) captures the total number of messages in the thread, allowing the classifier to distinguish between longer and shorter discussions.

3.4.3 Semantic similarity features added

In Section 2.4.1, we looked at the potential benefits of using semantic similarity between messages to characterise forum discussion data. The original data set included features for the similarity between a message and the messages immediately before and after it, but we removed these due to inconsistencies in the definition of 'previous' and 'next' (Section 3.4.1). In the study examined in Section 2.2 where linear 'chains' of messages were extracted (Waters et al., 2015), the definition of 'previous' and 'next' messages was unambiguous; but this is not the case in general. A new message might be posted as a 'child' of the most recent message, or as a 'sibling' (reply to the same parent) – or at another level entirely in the message tree.

Structural feature	Description
discussion size	The total number of messages in the current discussion. We speculate that a short discussion is less likely to progress to the deeper levels of cognitive presence than a longer one.
position from start	The index of the message in chronological order from the beginning of the discussion. Messages early in the discussion are intuitively more likely to indicate a *triggering event*.
position from end	The index of the message in chronological order from the end of the discussion. Later messages may relate to deeper levels, such as *resolution*, or to farewell greetings which do not indicate cognitive presence and will be labelled as *other*.
fractional position	The position of the message chronologically within the discussion, as a fraction of the total discussion size. This feature seeks to allow for natural variations in discussion length.
number of direct replies	The number of direct replies to the message. Messages relating to *triggering event*s and *exploration* are expected to generate more replies than those at deeper levels (Waters et al., 2015).
total number of replies	The cumulative number of direct and indirect replies (replies to replies). We are interested to see whether this feature captures the role of *triggering event*s and *exploration* better than direct replies (McKlin, 2004).
depth in thread	The depth of the message within the threaded view of the discussion. A deeper message might be more likely to build on the previous messages and thus relate to a deeper level of cognitive presence.
first/ last message	binary indicators for the first and last message in a discussion, defined chronologically.

Table 3.9: Structural features we added to the data. Some of these were direct replacements for features that we removed from the original data, while others provided additional metrics to characterise the discussion structure.

We have argued that chronological order of posting is potentially just as important as position in the tree structure (Section 3.4.2). We therefore calculated three similarity measures for each message: comparing a message to its parent, and to the messages added to the same thread immediately before and after it chronologically.[2]

We calculated semantic similarity scores using the spaCy tool[3] (Honnibal and Montani, 2017) to estimate document similarity using the pre-trained word vectors provided. These 300-dimensional vectors were trained on Common Crawl with GloVe (Pennington et al., 2014). Unlike the data extracted from Wikipedia that was used in prior work, the word vectors are not domain specific and can be used with content from any domain. The estimated value of the similarity between two messages is the cosine similarity using an average of word vectors. A higher score indicates that the messages are more similar. Three similarity features (similarity to parent, similarity to previous, and similarity to next) were generated directly from these scores (Table 3.10).

Similarity feature	Description
similarity to parent	The similarity between a message and its (unique) parent message.
similarity to previous/ next	The similarity between a message and the message immediately before and after it in chronological order within the discussion thread.

Table 3.10: Semantic similarity features created using spaCy. These replaced the similarity features we removed from the original data.

3.4.4 Sentiment features added

None of the original classification features explicitly captured message sentiment, although the linguistic features extracted by LIWC do include a representation of emotional tone. To investigate whether adding sentiment features adds further value, we measured sentiment using SentiStrength[4], a freely available tool for estimating sentiment strength in short informal texts (Thelwall et al., 2010). By default, SentiStrength reports two scores for each input text, the maximum strength of positive and negative

[2] A more nuanced version of this approach might take into account the elapsed time before deciding whether temporal adjacency was relevant.

[3] https://spacy.io

[4] http://sentistrength.wlv.ac.uk/

sentiment seen at any point in the input. We use these two numeric values directly as classification features (positive sentiment and negative sentiment, Table 3.11).

Sentiment feature	Description
positive sentiment	The strongest expression of positive sentiment, from 1 to 5.
negative sentiment	The strongest expression of negative sentiment, from -1 to -5.

Table 3.11: Sentiment analysis features generated by SentiStrength.

3.4.5 Feature combinations used in evaluation

All the 197 lexical and linguistic features in the original data that were extracted using the LIWC (Tausczik and Pennebaker, 2010) and Coh-Metrix (McNamara et al., 2014) tools were retained. We tested four combinations of additional features:

- the structural features (206 features in total);
- the structural features plus the semantic similarity features (209 features in total);
- the structural features plus the sentiment features (208 features in total); and
- the structural features plus both the semantic similarity features and the sentiment features (211 features in total).

We trained predictive classifiers on the first 5 course offerings, and evaluated them on the final one (Table 3.6). In each case, 20 random forests of 1000 trees were built, using 10-fold cross-validation, repeated 10 times. The values of `mtry` that were tested, based on the number of features in the model, are shown in Table A.1 in Appendix A. Classes were rebalanced using the method that demonstrated the best results in Experiment 1.

3.5 Experiment 3: Using topic modelling

Experiment 3 addressed Research Question 3. Topic modelling allows us to identify common features in discussion forum messages and to classify the types of messages students post, without the need for manual content analysis (Section 2.4.3). In this

experiment, we tested different numbers of topics and features to determine whether using topic modelling could further improve the predictive power of the model from Experiment 2; and examined the predictive power of the topic-related features to see whether they provided any new insights into cognitive presence.

We used the popular MALLET[5] (MAchine Learning for LanguagE Toolkit) tool to do topic modelling on our data (McCallum, 2002). It includes a sampling-based implementation of latent Dirichlet allocation (LDA). MALLET can be used to infer the topic distributions in unseen messages if they are prepared in the same way as the training data.

3.5.1 Generating topics with MALLET

We used MALLET to generate topics from our training data, then used those generated topics to infer values for both training and test data to use as features in the classifier.

The number of topics to generate is a free parameter. We speculated that generating a relatively large number of features could help to identify commonalities across the messages beyond the obvious content-specific ones; and that using a much smaller number of topics could allow us to avoid specificities and find more generally-applicable topics in the text. Recent work (Ferreira et al., 2018) used topic modelling with the same data set (Section 2.2) and concluded that 15 topics was optimal for their purposes. We used three different values for the number of topics: 5, 15, and 30.

We did a small amount of pre-processing on the message text before processing it with MALLET, as is commonly done to improve topic detection. Students' names were replaced with the token `[NAME]`, and URLs were replaced with the token `[URL]`; and MALLET was instructed to ignore these. Mark-up indicating that an answer had been edited by a tutor was also removed. White space was normalised and the text was converted to lower case.

The cleaned data was imported into MALLET format. Standard command-line options were used to ignore the two special tokens, and to remove 'stop words' (common words with little predictive value)[6]. After the training data was successfully imported, we imported the full dataset ready to use for inference, using the built-in option to ensure data compatibility.

We trained 3 separate topic models, using 5, 15, and 30 topics respectively, for 1000 iterations each, optimising the hyperparameters every 10 iterations.

[5] http://mallet.cs.umass.edu

[6] MALLET comes with a general-purpose list of stop words for English text.

In the model with **5 topics** (Table 3.12), Topic 1 seems to relate to general comments about the student presentations and the research they present. Only the word 'code' is domain-specific. We would expect to find this topic appearing with high probability in messages across all of the message threads in our data set, in future runs of the same course, and in courses with a similar structure. In contrast, the remaining topics are more closely tied to the specific research areas chosen by the students represented in the training data and are less likely to be of general value beyond this particular course domain.

Topic 1	Topic 2	Topic 3	Topic 4	Topic 5
presentation	architecture	programming	software	software
paper	system	time	url	development
authors	web	study	engineering	project
good	model	code	presentation	requirements
interesting	application	authors	reverse	process
code	component	maintenance	web	question
work	framework	pair	link	agile
information	data	documentation	paper	approach
research	components	developers	doi	model
mentioned	systems	programmers	computer	paper

Table 3.12: The top words that characterised each topic in a 5-topic model.

When we generate **15 topics** (Table 3.13), we find that Topic 11 is very similar to Topic 1 in the 5-topic model, capturing comments about the student presentations themselves. Topic 13 might capture high-level discussion of the research that was presented by the students. Topic 14 is specific to the course domain ("software", "engineering", "computer", "science") but not to any particular sub-topic in the course, so it might be expected to generalise well to subsequent runs and related courses. Topic 2 in this set looks like it might cover some research areas in computer science and also technical issues with the presentations (with the words "video" and "audio" appearing high in the list). The other topics in the model relate to the research areas the students chose to present. Finally, we note that within the lists of most probable words for a topic, some are closely related (e.g. "defect" and "defects" in Topic 2, "system" and "systems" in Topic 3). Further pre-processing could use stemming to combine these related words before topics are generated; we chose not to do so in this study.

Topic 2	Topic 3	Topic 11	Topic 13	Topic 14
data	system	presentation	software	url
algorithm	systems	paper	authors	software
video	tdd	good	question	engineering
authors	test	interesting	research	presentation
system	code	slides	paper	link
defects	tests	topic	time	doi
bugs	environment	clear	questions	paper
accuracy	trust	authors	presentation	computer
defect	design	mentioned	study	science
audio	researchers	great	good	ieee

Table 3.13: The top words from 5 topics selected from a 15-topic model.

The topic model with **30 topics** (Table 3.14) did capture much finer-grained topic distinctions, as expected. Many of the topics relate to very specific research areas addressed by the students (e.g. Topic 6), but others look as though they could be more widely applicable. Topic 10 relates to technical aspects of the student presentations, rather than their content. Topic 12 is a little more domain-specific but captures something about the research underlying the presentations, with words including "link", "doi", and "reference". Topic 20 seems to capture student peer feedback about the presentations, which is likely to be given early in the discussion and to relate to the earlier phases of cognitive presence (if any). Finally, Topic 22 has a lot of very informal words and might relate to social chat, which does not indicate cognitive presence.

We used each topic model in turn to infer the predicted values for each topic across the whole data set, and then turned these probability distributions into features for the classifier as described next.

3.5.2 Defining topic features

To create features for use in the random forest classifier, we first used the proportion of each topic in a message directly as a feature. For the model with 5 topics, this adds 5 new features to the data (relevance of topic 1, ..., relevance of topic 5). Next, we added a feature capturing the rank order of each topic for that message, with the idea that messages which related to the same topics in the same relative order might be

Topic 6	Topic 10	Topic 12	Topic 20	Topic 22
agile	video	url	presentation	work
development	audio	software	paper	i'm
project	presentation	presentation	interesting	it's
approach	tags	link	good	don't
projects	screen	engineering	authors	lot
methods	text	doi	slides	things
waterfall	file	ieee	topic	time
methodology	quality	reference	clear	i've
requirements	comments	conference	research	make
government	tagviewer	paper	mention	guess

Table 3.14: The top words from 5 topics selected from a 30-topic model.

similar even though the actual proportions would differ. For the 5-topic model, this adds another 5 features (rank of topic 1, ..., rank of topic 5).

Finally, we experimented with optionally adding a second set of proportion features relating to some of the top-ranked topics. The values are the same as those in the features for relevance of topic, just in a different order. For example, relevance at rank 1 would be the proportion relating to the most relevant topic for the message, regardless of its topic number. The idea here was to allow the classifier to distinguish between messages that focus narrowly on one or two topics and those with no particular focus. The number of features this adds is k, the number of top-ranked topics used (relevance at rank 1, ..., relevance at rank k). Table 3.15 lists all the topic features.

Topic feature	Description
relevance of topic N	the proportional relevance of topic N.
rank of topic N	the ranked importance of topic N.
relevance at rank k	the proportional relevance of the kth ranked topic.

Table 3.15: Topic features added to the data set, derived from one generated topic model. We used models with $N = 5$, 15, and 30 topics and included the relevance at the top k ranks for $k = 0$, 3, 5, 15, and 30 (up to the number of topics in the model).

We trained classifiers using each of the topic models, with a range of values for the number of top-ranked topics used to create additional features: $k \in \{0, 3, 5, 15, 30\}$ (up to the number of topics in the model), as shown in Table 3.16. In all cases, the data included all 211 features from Experiment 2: lexical, linguistic, and structural features, plus semantic similarity and sentiment. In each case, 20 random forests of 1000 trees were built, using 10-fold cross-validation, repeated 10 times. The values of `mtry` that were tested, based on the number of features in the model, are shown in Table A.2 in Appendix A. We rebalanced the classes based on the best result from Experiment 1, as before. The trained models were evaluated on the data from the final course offering (Table 3.6).

Topics used		Classification features	
Generated	Top-ranked	Topic features	All features
5	0	10	221
	3	13	224
	5	15	226
15	0	30	241
	3	33	244
	5	35	246
	15	45	256
30	0	60	271
	3	63	274
	5	65	276
	15	75	286
	30	90	301

Table 3.16: The number of topics generated by MALLET, the number of top-ranked topics for which we create additional features, the number of topic-related features in the model, and the total number of classification features.

Finally, we used the combination of topic features that achieved the best results and evaluated three further conditions: omitting semantic similarity (3 features), omitting sentiment (2 features), and omitting both semantic similarity and sentiment (5 features). In each case, 20 random forests of 1000 trees were built, using 10-fold cross-validation, repeated 10 times. The values of `mtry` that were tested, based on the

number of features in the model, are shown in Table A.2 in Appendix A. We rebalanced the classes based on the best result from Experiment 1, as usual. The trained models were evaluated on the data from the final course offering (Table 3.6).

3.6 Implementation

The classifier was implemented using the Python[7] and R[8] programming languages. The key software packages and libraries used were:

- LIWC (Tausczik and Pennebaker, 2010) and Coh-Metrix (McNamara et al., 2014) for lexical and linguistic feature extraction;
- spaCy (Honnibal and Montani, 2017) for calculating semantic similarity scores;
- SentiStrength (Thelwall et al., 2010) for sentiment analysis;
- MALLET (McCallum, 2002) for topic modelling;
- the `randomForest` R package (Liaw and Wiener, 2002) for the random forest classifier;
- the `caret` R package (Kuhn et al., 2018) for running repeated cross-validation, and for training and validating the final model; and
- the `DMwR` and `plyr` R packages (Torgo, 2010; Wickham, 2011) for generation of synthetic data points using SMOTE.

[7] https://www.python.org
[8] https://www.R-project.org

Chapter 4

Results

The results of our experiments are reported in this chapter. We analyse them and set them in the context of related work in Chapter 5.

4.1 Experiment 1: Replication study

As expected, our results for the baseline replication closely approximated the previously reported results (Table 4.1). We got the same Cohen's κ of 0.63, and an accuracy of 70.1%, rather than 70.3%. The macro-averaged F_1 score was 0.69. The parameter tuning loop selected 12 as the best value for the `mtry` parameter. The confusion matrix for the replication model is shown in Table 4.4.

Condition	Accuracy	Cohen's κ	Macro F_1
Published baseline	70.3%	0.63	–
Replication of baseline	70.1%	0.63	0.69

Table 4.1: Using the procedure from Kovanović et al. (2016) and the same rebalanced data set, we closely matched the published results.

Using a stratified sample for training, the model that used the unbalanced data achieved an accuracy of 60.6%, with Cohen's $\kappa = 0.43$ and macro-averaged $F_1 = 0.52$ (Table 4.2). The best performing model used SMOTE exact inside the loop and achieved an accuracy of 61.7%, with Cohen's $\kappa = 0.46$ and macro-averaged $F_1 = 0.58$. The parameter tuning loop selected 34 as the best value for the `mtry` parameter. The confusion matrix for this model is shown in Table 4.5.

Condition	Accuracy	Cohen's κ	Macro F_1
Training data without rebalancing	60.6%	0.43	0.52
Training data balanced with SMOTE multi	58.3%	0.41	0.54
Training data balanced with SMOTE exact	59.1%	0.42	0.54
Using SMOTE multi inside the loop	61.2%	0.45	0.57
Using SMOTE exact inside the loop	**61.7%**	**0.46**	**0.58**

Table 4.2: Three approaches to re-sampling and two variants of the SMOTE algorithm, evaluated using a stratified random sample. The best results are in bold.

Splitting the data based on course offering instead of using a random split leads to poorer results (Table 4.3). The model that was trained on the unbalanced data achieved an accuracy of just 52.7%, with Cohen's $\kappa = 0.33$ and macro-averaged $F_1 = 0.47$. The best performing model (using SMOTE exact) achieved an accuracy of 54.9%, with Cohen's $\kappa = 0.38$ and macro-averaged $F_1 = 0.54$. The parameter tuning loop selected 44 as the best value for the `mtry` parameter. The confusion matrix for this model is shown in Table 4.6.

Condition	Accuracy	Cohen's κ	Macro F_1
Training data without rebalancing	52.7%	0.33	0.47
Using SMOTE multi inside the loop	51.9%	0.34	0.52
Using SMOTE exact inside the loop	**54.9%**	**0.38**	**0.54**

Table 4.3: Splitting the data by course offering and comparing the effect of rebalancing the classes using two variants of the SMOTE algorithm inside the cross-validation loop. The best results are in bold.

Looking more closely at the best model trained on the session-based split, we see that a small subset of features have a high degree of explanatory power, evidenced by their high Mean Decrease Gini (MDG) values (Figure 4.1). The median MDG for this model was 7.30 and the mean was 11.86. The top 20 features by importance are listed in Table 4.7 along with their average values in each phase of cognitive presence.

	Predicted				
Actual	Other	Triggering	Exploration	Integration	Resolution
Other	**78**	3	3	1	2
Triggering	4	**67**	9	7	0
Exploration	9	15	**36**	27	0
Integration	4	2	22	**44**	15
Resolution	0	0	4	3	**80**

Table 4.4: Confusion matrix for the baseline replication model in Experiment 1.

	Predicted				
Actual	Other	Triggering	Exploration	Integration	Resolution
Other	**22**	3	8	2	0
Triggering	2	**56**	17	2	0
Exploration	6	17	**110**	36	2
Integration	1	2	43	**75**	6
Resolution	0	0	2	18	**6**

Table 4.5: Confusion matrix for the best performing model trained on a stratified random sample in Experiment 1.

	Predicted				
Actual	Other	Triggering	Exploration	Integration	Resolution
Other	**17**	1	8	2	0
Triggering	1	**23**	4	0	0
Exploration	6	6	**41**	21	2
Integration	2	2	29	**45**	5
Resolution	0	0	5	13	**4**

Table 4.6: Confusion matrix for the best performing model trained using a session-based split in Experiment 1.

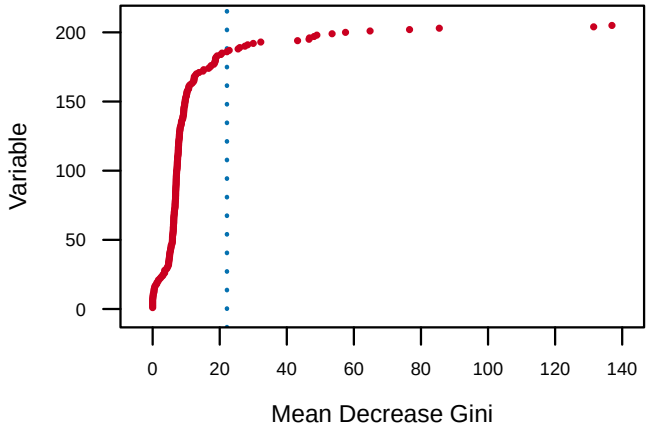

Figure 4.1: Variable importance of the model trained using a session-based split in Experiment 1, ranked by the Mean Decrease Gini measure. The blue dotted line separates the top 20 features.

Variable	Description	MDG	Other	Trig.	Expl.	Integ.	Resol.
ner.entity.cnt	Number of named entities	136.94	13.09	21.30	29.51	44.39	64.56
cm.DESWC	Number of words	131.49	55.16	79.28	122.29	183.32	285.35
message.is.first	First message	85.46	1.00	1.27	1.00	1.00	1.00
message.depth	Message depth in discussion	76.60	2.49	1.00	1.83	1.89	2.03
liwc.SemiC	Number of semi-colons	64.82	0.35	0.18	0.12	0.09	0.20
cm.LDTTRa	Lexical diversity, all words	57.51	0.86	0.77	0.71	0.65	0.58
liwc.discrep	Number of discrepancy words	53.50	1.17	0.98	1.75	1.90	2.53
cm.WRDHYPn	Hypernyms for nouns	48.96	5.03	5.45	6.03	6.17	6.25
liwc.posemo	Number of expressions of +ve emotion	48.14	9.65	4.45	4.00	3.43	3.17
liwc.QMark	Number of question marks	46.77	0.33	1.87	0.94	0.55	0.40
liwc.money	Number of money words	46.63	0.23	0.31	0.37	0.61	0.96
cm.WRDMEAc	Meaningfulness	43.21	364.15	406.56	404.88	405.56	405.72
cm.DESWLltd	SD of word length in letters	32.24	3.11	4.03	2.93	2.90	2.89
message.sim.prev	Semantic similarity to previous	29.99	0.19	0.05	0.23	0.29	0.39
liwc.affect	Number of affective process words	28.33	10.24	5.04	4.73	4.31	4.02
message.reply.cnt	Number of replies	27.51	0.43	1.43	0.81	0.95	0.81
liwc.netspeak	Number of netspeak words	25.99	1.77	0.90	0.25	0.24	0.28
cm.LDVOCD	Lexical diversity, VOCD	25.48	13.77	28.32	54.78	82.44	95.58
cm.LSASSpd	SD of LSA overlap in paragraph	22.83	0.07	0.11	0.13	0.13	0.13
liwc.health	Number of health words	22.15	0.09	0.04	0.12	0.11	0.16

Table 4.7: The 20 most important variables in the model trained using a session-based split in Experiment 1, ranked by Mean Decrease Gini (MDG) and shown with their mean values for messages in different phases of cognitive presence.

4.2 Experiment 2: No external knowledge base

Models using different combinations of classification features were evaluated on the data from the final course offering (Table 3.6) and compared against the best result from Experiment 1. The results are shown in Table 4.8.

The best performing model was trained using 206 lexical, linguistic, and structural features without any additional semantic similarity or sentiment features. It achieved an accuracy of 56.1%, with Cohen's $\kappa = 0.40$ and macro-averaged $F_1 = 0.55$.

The parameter tuning loop selected 23 as the best value for the `mtry` parameter. The confusion matrix for the model with the highest accuracy is shown in Table 4.9.

Condition	Accuracy	Cohen's κ	Macro F_1
Original features (best result from Table 4.3)	54.9%	0.38	0.54
All replacement features	55.7%	0.39	**0.55**
Without semantic similarity features	52.7%	0.35	0.53
Without sentiment features	54.4%	0.38	0.54
Without semantic similarity or sentiment	**56.1%**	**0.40**	**0.55**

Table 4.8: Comparing four combinations of new classification features against the best results with the original features in Experiment 2. The best results are in bold.

	Predicted				
Actual	Other	Triggering	Exploration	Integration	Resolution
Other	**18**	1	7	2	0
Triggering	1	**23**	4	0	0
Exploration	6	6	**41**	21	2
Integration	1	1	31	**47**	3
Resolution	0	0	4	14	**4**

Table 4.9: Confusion matrix for the best performing model in Experiment 2.

The median MDG for this model was 8.16 and the mean was 11.80 (Figure 4.2). The top 20 features by importance are listed in Table 4.10 along with their average values in each phase of cognitive presence.

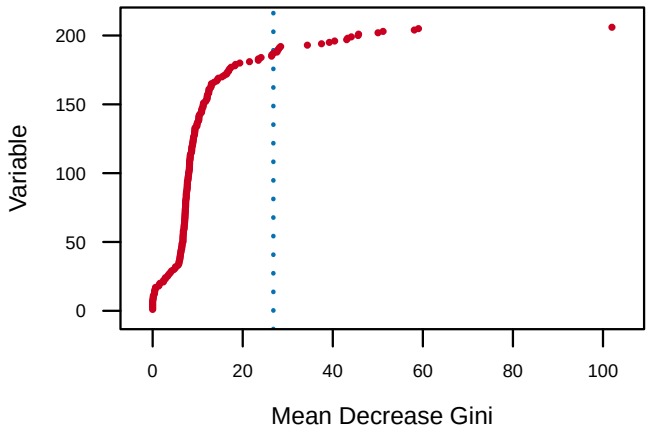

Figure 4.2: Variable importance in Experiment 2, ranked by the Mean Decrease Gini measure. The blue dotted line separates the top 20 features.

Variable	Description	MDG	Other	Trig.	Expl.	Integ.	Resol.
cm.DESWC	Number of words	101.99	55.16	79.28	122.29	183.32	285.35
cm.LDTTRa	Lexical diversity, all words	59.02	0.86	0.77	0.71	0.65	0.58
message.depth	Message depth in discussion	58.11	2.49	1.00	1.84	1.89	2.03
message.is.first	First message	51.16	1.00	1.27	1.00	1.00	1.00
liwc.posemo	Number of expressions of +ve emotion	50.04	9.65	4.45	4.00	3.43	3.17
cm.WRDHYPn	Hypernyms for nouns	45.72	5.03	5.45	6.03	6.17	6.25
message.replies.direct	Number of direct replies	45.63	0.47	2.94	0.55	0.53	0.42
liwc.discrep	Number of discrepancy words	44.11	1.17	0.98	1.75	1.90	2.53
cm.WRDMEAc	Meaningfulness	43.30	364.15	406.56	404.88	405.56	405.72
liwc.SemiC	Number of semi-colons	43.09	0.35	0.18	0.12	0.09	0.20
liwc.QMark	Number of question marks	40.41	0.33	1.87	0.94	0.55	0.40
message.replies.all	Total number of replies	39.23	0.81	6.34	0.80	0.79	0.66
liwc.money	Number of money words	37.50	0.23	0.31	0.37	0.61	0.96
liwc.affect	Number of affective process words	34.36	10.24	5.04	4.73	4.31	4.02
cm.DESSL	Mean sentence length	28.43	11.64	13.55	17.02	19.10	21.99
cm.DESWLltd	SD of word length in letters	28.12	3.11	4.03	2.93	2.90	2.89
cm.LDVOCD	Lexical diversity, VOCD	27.96	13.77	28.32	54.78	82.44	95.58
message.thread.size	Discussion size	27.67	23.13	24.18	24.21	22.92	22.88
cm.LDTTRc	Lexical diversity, content words	27.61	0.95	0.90	0.86	0.82	0.78
cm.LSASSpd	SD of LSA overlap in paragraph	26.81	0.07	0.11	0.13	0.13	0.13

Table 4.10: The 20 most important variables in Experiment 2, ranked by Mean Decrease Gini (MDG) and shown with their mean values for messages in different phases of cognitive presence.

4.3 Experiment 3: Using topic modelling

Models trained using topic features (in addition to the features used in the best model from Experiment 2) were evaluated on the data from the final course offering (Table 3.6), and the results are shown in Table 4.11. The best model, using features from a 15-topic model without any additional features for the top-ranked topics, reached a slightly higher accuracy than the previous best model (56.5%), with Cohen's $\kappa = 0.40$.

The parameter tuning loop selected 64 as the best value for the `mtry` parameter. The confusion matrix for the best model is shown in Table 4.12.

Condition	Accuracy	Cohen's κ	Macro F_1
Replacement structural features (Table 4.11)	56.1%	**0.40**	0.55
All replacement features (from Table 4.11)	55.7%	0.39	0.55
5 topics, no top-ranked topic features	54.4%	0.37	0.54
5 topics, 3 top-ranked topic features	53.6%	0.36	0.54
5 topics, 5 top-ranked topic features	55.3%	0.38	0.54
15 topics, no top-ranked topic features	**56.5%**	**0.40**	0.55
15 topics, 3 top-ranked topic features	53.1%	0.36	0.53
15 topics, 5 top-ranked topic features	54.9%	0.38	0.54
15 topics, 15 top-ranked topic features	51.9%	0.34	0.51
30 topics, no top-ranked topic features	53.1%	0.36	0.52
30 topics, 3 top-ranked topic features	53.6%	0.36	0.54
30 topics, 5 top-ranked topic features	55.7%	0.39	**0.56**
30 topics, 15 top-ranked topic features	51.9%	0.34	0.50
30 topics, 30 top-ranked topic features	51.9%	0.35	0.52

Table 4.11: Experiment 3 compared the effect of adding features derived from topic models, in addition to the features used in Experiment 2. The best results are in bold.

Three further conditions were evaluated, using the best-performing topic features (15 topics, no additional features for the top-ranked topics) and omitting semantic similarity, sentiment, and both. The results are shown in Table 4.13. None of these new conditions beat the previous best result, using both semantic similarity and sentiment features in conjunction with the topic features.

	Predicted				
Actual	Other	Triggering	Exploration	Integration	Resolution
Other	**19**	1	7	1	0
Triggering	1	**22**	5	0	0
Exploration	7	5	**43**	19	2
Integration	1	5	30	**46**	4
Resolution	0	0	4	14	**4**

Table 4.12: Confusion matrix for the best performing model in Experiment 3.

Condition	Accuracy	Cohen's κ	Macro F_1
Best model from Table 4.11	**56.5%**	**0.40**	**0.55**
Without semantic similarity features	53.2%	0.36	0.53
Without sentiment features	55.3%	0.39	0.54
Without semantic similarity or sentiment	55.3%	0.38	0.54

Table 4.13: Comparing the effect of omitting the semantic similarity and sentiment features from the best model from Table 4.11. The best results are in bold.

The median MDG for the best model was 5.59 and the mean was 10.09 (Figure 4.3). The top 20 features by importance are listed in Table 4.14 along with their average values in each phase of cognitive presence.

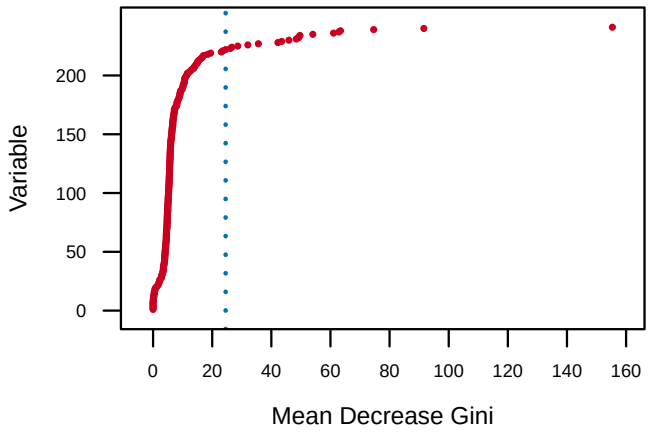

Figure 4.3: Variable importance in the best model in Experiment 3, ranked by Mean Decrease Gini. The blue dotted line separates the top 20 features.

Variable	Description	MDG	Other	Trig.	Expl.	Integ.	Resol.
cm.DESWC	Number of words	155.35	55.16	79.28	122.29	183.32	285.35
message.is.first	First message	91.59	1.00	1.27	1.00	1.00	1.00
liwc.SemiC	Number of semi-colons	74.62	0.35	0.18	0.12	0.09	0.20
message.depth	Message depth	63.38	2.49	1.00	1.84	1.89	2.03
cm.LDTTRa	Lexical diversity, all	62.85	0.86	0.77	0.71	0.65	0.58
liwc.posemo	Number of expressions of +ve emotion	61.05	9.65	4.45	4.00	3.43	3.17
cm.WRDMEAc	Meaningfulness	54.04	364.15	406.56	404.88	405.56	405.72
liwc.QMark	Number of question marks	49.69	0.33	1.87	0.94	0.55	0.40
topics.topic14.percent	Probability of Topic 14	49.42	0.10	0.16	0.03	0.03	0.01
liwc.money	Number of money words	49.18	0.23	0.31	0.37	0.61	0.96
cm.WRDHYPn	Hypernyms for nouns	48.44	5.03	5.45	6.03	6.17	6.25
liwc.discrep	Number of discrepancy words	45.97	1.17	0.98	1.75	1.90	2.53
message.replies.direct	Number of direct replies	43.50	0.47	2.94	0.55	0.53	0.42
similarity.prev	Semantic similarity to previous	42.24	0.88	0.69	0.95	0.96	0.97
similarity.parent	Semantic similarity to parent	35.62	0.87	0.65	0.92	0.93	0.95
topics.topic11.percent	Probability of Topic 11	32.04	0.29	0.27	0.19	0.12	0.10
liwc.affect	Number of affective process words	28.66	10.24	5.04	4.73	4.31	4.02
message.thread.size	Discussion size	26.66	23.13	24.18	24.21	22.92	22.88
message.replies.all	Total number of replies	26.18	0.81	6.34	0.80	0.79	0.66
cm.LDVOCD	Lexical diversity, VOCD	24.55	13.77	28.32	54.78	82.44	95.58

Table 4.14: The 20 most important variables in the best model in Experiment 3 ranked by Mean Decrease Gini (MDG), shown with their mean values for messages in different phases of cognitive presence.

Chapter 5

Analysis

Our overall goal was to improve the robustness and generalisability of automated classification methods for detecting cognitive presence, so that this work can be used with confidence by researchers in the field. We posed three research questions and conducted experiments in order to answer them. We now look at each experiment in turn and analyse the results and their implications.

5.1 Experiment 1: Replication study

Research Question 1

How do the results from the state-of-the-art model (Kovanović et al., 2016) compare with a replication study using current best practice for handling unbalanced classes and splitting data into training and test sets?

We discuss the results of Experiment 1 in terms of i) metrics for predictive performance (accuracy, Cohen's κ, macro-averaged F_1 score) described in Section 2.6.1, and ii) the contribution of individual classification features used in the prediction task.

Predictive performance metrics

As expected, our baseline replication closely approximated the previously reported results (Table 4.1). We got the same Cohen's κ of 0.63, and an accuracy of 70.1%, rather than 70.3%. The macro-averaged F_1 score was 0.69. The small difference in accuracy is assumed to be due to variations in the random seeds used in the stratified sampling step and the initialisation of the random forest.

Using a stratified random sample for training, we saw that rebalancing the training data before tuning the model (as Neto et al. (2018) did) actually decreased the final model's performance compared to using unbalanced data, whereas moving the rebalancing step inside the cross-validation loop improved the results (Table 4.2). The best performing model used SMOTE exact inside the loop and achieved an accuracy of 61.7%, with Cohen's $\kappa = 0.46$ and macro-averaged $F_1 = 0.58$. Compared to the unbalanced case, accuracy was 1.1 percentage points higher, Cohen's κ increased by 0.03, and macro-averaged F_1 by 0.06. This is consistent with prior work on parameter tuning with small data sets (Kuncheva and Rodríguez, 2018).

Using a session-based data split instead, training on the earlier course sessions and evaluating on the final one led to much lower results on every metric than when using a random split (Table 4.3). This is again consistent with recent work on replication in MOOCs (Gardner et al., 2018), which cites several studies where evaluating predictive models using data from students in the same session of the course resulted in higher estimates of model performance, compared with using data from new sessions of the same course. One reason for this behaviour could be that the data points are not independent. Leaving aside any issues to do with over-sampling, the messages themselves are related to one another and form a natural sequence. They will share commonalities, such as vocabulary used, that go beyond cognitive presence. Taking several messages from a discussion thread to use for training, and then testing on another message from the same thread, will not give unbiased results.

This also suggests that where splitting by course offering is not possible, it could be worthwhile to select discussion threads for training and testing so that the same participants do not appear in both sections, in order to achieve more reliable results – although this may not be possible with some very small courses.

Despite attempting to closely replicate the prior work from Kovanović et al. (2016) and using the same data, we were unable to achieve similar results when we split the data into training and test sets before applying class rebalancing: the results were much lower than the state-of-the-art result on every outcome metric. This leads us to believe that the prior results may have been affected by data contamination between the training and test sets, as we explored in Section 2.6.4, leading to an over-estimation of that model's predictive power. Using a session-based split instead of a random split in our replication reduced the accuracy, Cohen's κ, and macro-averaged F_1 further. The SMOTE exact variant of the rebalancing algorithm improved the results over the unbalanced data in every case, by a larger margin than SMOTE multi.

Thus, we answer Research Question 1 by concluding that following a best-practice approach leads to lower results than those reported in Kovanović et al. (2016) because of two types of over-fitting. The first is due to the way the class rebalancing step was carried out in this particular study; while the second relates to the widespread practice of using a stratified random sample to split data into training and test sets, and means that reported results from other studies may also be over-optimistic.

Variable importance analysis

One advantage of random forests over 'black box' methods like neural networks is that they have the capacity to be interpretable – they provide information about which features within the model are most predictive, allowing us to gain deeper insight (Hofman et al., 2017). We will now look at which features in the best model were most predictive according to the Mean Decrease Gini (MDG) metric, and how they relate to prior work on cognitive presence.

The most explanatory variable was the number of named entities, closely followed by the number of words. Higher numbers of named entities in a message, and longer messages, were both associated with deeper levels of cognitive presence. This is in line with previous studies (McKlin, 2004; Kovanović et al., 2016).

Messages displaying deeper levels of cognitive presence tended to occur later in the discussion, used more words from the LIWC categories relating to discrepancies (such as *should* and *would*) and money (such as *owe*), and fewer question marks; and displayed a greater than average similarity with the previous message, indicating that they are building on what has gone before.

Messages in the *other* class tended to be nested deepest in the discussion; to contain more words from LIWC relating to affective processes, particularly expressions of positive emotion, and more semi-colons; and to have a lower Coh-Metrix score for meaningfulness. These correlations match well with our observations of the kind of social messages often coded as *other*, like the example in Table 3.2.

Lower levels of lexical diversity were associated with *other* and *triggering event* messages, in common with prior work (Kovanović et al., 2016). However, the 'first message' indicator, which was not found to be predictive in prior work, was highly predictive here and strongly indicative of *triggering event*s. We speculate that the over-sampling process used in prior work may have had the side-effect of changing it from a boolean indicator to a continuous value, reducing its predictive power.

Overall, only 9 of the top 20 features (45%) were the same as those identified in Kovanović et al. (2016). This low figure is notable, as Experiment 1 used exactly the same set of features as the prior work. We believe that the issues with data contamination and over-fitting identified here may have led the prior model to see some features as more predictive than was really the case, and to disregard others that actually have more discriminative power.

5.2 Experiment 2: No external knowledge base

Research Question 2
Can we achieve similar predictive performance without relying on an external knowledge base for semantic annotation?

Experiment 2 compared the features used in the original published model with a new set of features that did not depend on an external knowledge base. Following best practice on replicability (Gardner et al., 2018), the most recent offering of the course was used as held-out test data. As with Experiment 1, we discuss the results in terms of i) predictive performance metrics and ii) the contribution made by individual classification features.

Predictive performance metrics

The model with the best results was trained using lexical, linguistic, and structural features, without any additional semantic similarity or sentiment features. It achieved an accuracy of 56.1%, with Cohen's $\kappa = 0.40$ and macro-averaged $F_1 = 0.55$ (Table 4.8). Compared to Experiment 1, it correctly identified one additional *other* message and 2 more *exploration* messages (Table 4.9).

The model that was trained using all 211 features (lexical, linguistic, and structural features, plus semantic similarity and sentiment features) had the same macro-averaged F_1 score, similar Cohen's κ, and slightly lower accuracy.

Thus, we can give an affirmative answer to Research Question 2: we have improved performance over our previous results while removing reliance on the external knowledge base.

Variable importance analysis

Looking more closely at the model with the highest accuracy, we consider which of the 206 classification features in the best model were most predictive by MDG score, and compare them to prior work and to Experiment 1.

Once again, longer messages were associated with deeper levels of cognitive presence, and number of words was by far the most predictive variable. Messages displaying deeper levels of cognitive presence again used more words from the LIWC categories relating to discrepancies and money, and fewer question marks. They tended to be more deeply-nested in the discussion, but were not necessarily later in time. The feature we added to capture message depth was highly predictive, but none of the features relating to order within a thread by time-stamp appeared in the top 20 – other than the 'first message' indicator, which is strongly indicative of *triggering event*s.

Likewise, the variables that were most associated with *other* messages were the same as in Experiment 1, with the addition of the Coh-Metrix feature for standard deviation of LSA overlap in paragraph, where *other* messages scored the lowest.

In this experiment, we used two counts for message replies. Both were predictive, with high values indicative of *triggering event* messages. However, the number of direct replies a message received was more predictive than the total number of replies.

The feature for discussion thread size also proved to have some predictive power. It seems that both *triggering event* and *exploration* messages are associated with longer than average discussions. This is an interesting result, suggesting that longer discussions tend to feature more exploration.

Overall, 14 of the top 20[1] most predictive variables were the same as in Experiment 1, including 8 of the 9 that also appeared in Kovanović et al. (2016) (this data set does not include a feature for the number of named entities).

The relative stability of the top 20 features compared to Experiment 1 gives us confidence that after removing the grounded features that relied on an external knowledge base, we are still capturing the same important indicators of cognitive presence.

5.3 Experiment 3: Using topic modelling

Research Question 3

Can topic modelling give us new insights into cognitive presence?

[1] Both features we added for number of replies appear in the top 20 – we count this as a single match to the number of replies feature in Experiment 1.

We were satisfied that the features we added in Experiment 2, as well as those retained from the original data, provided a reliable foundation. Experiment 3 looked at whether adding classification features based on topic modelling could further improve the model's predictive power. As with the previous experiments, we discuss the results of Experiment 3 in terms of i) predictive performance metrics and ii) the contribution made by individual classification features used in the prediction task.

Predictive performance metrics

Topic models were generated with 5, 15, and 30 topics, and features based on the topic models were added to the predictive classifier. We found that a relatively simple approach using 15 topics gave another small improvement in accuracy and Cohen's κ. The best model, using features from a 15-topic model without any additional features for the top-ranked topics, reached a slightly higher accuracy than the previous best model (56.5%), with Cohen's κ of 0.4 indicating a 'fair' level of agreement with the gold-standard coding (Landis and Koch, 1977).

It was a little surprising to find model performance *decreasing* in many conditions when more features were added to the classifier, as one advantage of the decision tree algorithm (and, by extension, random forests) is that it simply ignores irrelevant features. One possible explanation for this behaviour is to note that as the number of features increases, so does the search space for the optimum value of the random forest parameter `mtry`. In order to train our models in a reasonable amount of time, we limited parameter tuning to 20 distinct values. These values were selected automatically by the R `caret` library based on the size of the feature space (Table A.2). Increasing the number of features in the data changed the specific values of `mtry` that were tested, and left many more values untested.

As the results from Experiment 2 showed good performance both with and without the semantic similarity and sentiment features, we tested three further conditions: using the best-performing topic features but omitting semantic similarity, sentiment, or both. Omitting either semantic similarity or sentiment features lowered performance on every metric (Table 4.13).

Variable importance analysis

In all, 16 of the top 20 most predictive features in the best model for this experiment were the same as in Experiment 2, including all of the top 14, and all 8 that also ap-

peared in Kovanović et al. (2016). The 4 additional features that were found to be predictive in this experiment were the similarity between a message and the previous and parent messages; and the probabilities of Topics 11 and 14 from the 15-topic model. None of these 4 features was used in Experiment 2.

The probability of Topic 11 from the 15-topic model (Table 3.13) was highest for messages that lacked signs of cognitive presence, and declined as the level of cognitive presence rose. This topic captures initial peer feedback on the student presentation, which can sometimes focus solely on surface aspects (e.g. nice slides) and lack depth.

Topic 14 seemed to be closely tied to the course domain, but not to any specific part of the syllabus (Section 3.5.1). Its mean was highest for *triggering event* messages and lowest for *resolution* messages, where we would expect students to have moved away from generalities to engage deeply with one particular aspect of the course.

We saw that adding features relating to the top-ranked topics did not improve predictive performance. The features that capture the rank order of topics within a message did not prove to be predictive either, with none appearing in the top 20.

We found that semantic similarity scores comparing a message to its parent message and to the previous message by time-stamp order[2] were both predictive and showed a similar pattern: *triggering event* messages had the lowest average score[3], while the other levels of cognitive presence all scored similarly, and much higher than messages labelled *other*. This contrasts with the findings of prior work (Kovanović et al., 2016), where similarity to the previous message rose with the level of cognitive presence, and similarity to the next message was also predictive. However, the calculation of semantic similarity scores in the prior work was sometimes inconsistent (Section 2.3.2). The methods used for deriving the semantic similarity features also differed: in our data, they were computed by the spaCy tool using a general-purpose word embedding, while prior work used the Stanford CoreNLP Java library.

5.4 Summary

Our best performing model overall, while following a best-practice approach to resampling and splitting the data by session, was therefore the one which used lexical, linguistic, and structural features, plus semantic similarity and sentiment, with topic features from a topic model generated with 15 topics. It identified the level of cogni-

[2] In many cases the previous message by time-stamp order will also be the parent message.

[3] The semantic similarity score is defined to be 0 when there is no parent/ previous/ next message.

tive presence in messages taken from an unseen run of the same course with 56.5% accuracy compared to gold-standard annotation, with Cohen's $\kappa = 0.40$. While these values are far below the best previously reported results of 70.3% accuracy and 0.63 Cohen's κ, we have demonstrated that our model can be expected to achieve similar results on future unseen data, whereas the prior results may not generalise well.

5.5 Limitations

Despite our careful attempts at replication, a major limitation of this work is that we only used data from a single course. It would be good to compare our results with data from other courses, to see whether the high-level features that worked well here will be useful in other contexts.

As we are using a very small data set, the topics extracted using topic modelling in Experiment 3 may also lack predictive power.

Chapter 6

Conclusion

Modelling student engagement in online discussion forums is an important topic with the potential to bring benefits to both students and educators. The tools of data science allow us to process large and varied data sets to find patterns in student behaviour and the written content they produce in informal discussion forums that can characterise their engagement with the course. Meanwhile, the Community of Inquiry framework provides a well-developed theoretical grounding for research in this area.

In this work, we used a data set of 1747 discussion forum messages to develop a classifier for predicting a student's level of cognitive presence. We found that we could not reproduce the previous state-of-the-art results while following best-practice guidelines to avoid data contamination. We were able to avoid using features that depend on an external knowledge base while improving overall performance. Topic modelling did not improve the predictive power of the classifier as much as we had hoped, but did provide a way of creating new classification features with some explanatory power.

Future work

We observed that many of the classification features examined in this work were good at distinguishing messages that show traces of cognitive presence from messages that do not (labelled *other*). We suggest that future work could consider using a hierarchical classification framework: first identifying and excluding the *other* messages, then classifying the remaining messages further. Mu et al. (2011) describes a layered domain-independent framework for classification that could provide a good model.

A variation on this idea is to group the four levels of cognitive presence into pairs in multiple ways, train and predict using these pairs, then recombine the results to make

a final prediction. The pairs of levels could be defined in terms of shallow and deep, rare and common (Table 6.1).

Shallow	Deep	Rare	Common
Triggering Event	Integration	Triggering Event	Exploration
Exploration	Resolution	Resolution	Integration

Table 6.1: The levels of cognitive presence can be grouped into pairs in different ways.

Staying with the feature-engineering approach used in this work, we believe that noting the order in which students contribute to a thread and tracking their subsequent contributions could help to predict levels of cognitive presence. In the data we used in this study, the student who posts the first message in each thread (introducing their video) also takes on the role of 'expert' regarding the thread topic (Gašević et al., 2015). Thus, their contributions to the thread are qualitatively different from the other participants. In other courses, it might be reasonable to assume that the first student to respond to a *triggering event* message is the most well-informed about the subject area. It is an open question whether a better-informed student is more likely than others to move the discussion from *exploration* to *integration*, or from *integration* to *resolution*.

Bibliography

Arbaugh, J. B., Cleveland-Innes, M., Diaz, S. R., Garrison, D. R., Ice, P., Richardson, J. C., and Swan, K. P. (2008). Developing a community of inquiry instrument: Testing a measure of the Community of Inquiry framework using a multi-institutional sample. *Internet and Higher Education*, 11(3-4):133–136.

Atapattu, T. and Falkner, K. (2016). A Framework for Topic Generation and Labeling from MOOC Discussions. In *Proceedings of the Third (2016) ACM Conference on Learning @ Scale - L@S '16*, pages 201–204.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022.

Breiman, L. (2001). Random Forests. *Machine Learning*, 45(1):5–32.

Chaplot, D. S., Rhim, E., and Kim, J. (2015). Predicting student attrition in MOOCs using sentiment analysis and neural networks. *Workshops at the 17th International Conference on Artificial Intelligence in Education, AIED-WS 2015*, 1432:7–12.

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, 16:321–357.

Corich, S., Kinshuk, and Hunt, L. M. (2004). Assessing Discussion Forum Participation: In Search of Quality. *International Journal of Instructional Technology & Distance Learning*, 1(12):1–12.

Corich, S., Kinshuk, and Hunt, L. M. (2006a). Computerised Content Analysis for Measuring Critical Thinking within Discussion Forums. *Journal of e-Learning and Knowledge Society*, 2:47–60.

Corich, S., Kinshuk, and Hunt, L. M. (2006b). Measuring critical thinking within discussion forums using a computerised content analysis tool. In *Proceedings of the 5th International Conference on Networked Learning*, pages 1–8.

Corich, S., Kinshuk, and Jeffrey, L. M. (2007). Changing Focus From Group To Individual: Using an Automated Tool To Measure Evidence of Critical Thinking in Discussion Forums. In *IADIS International Conference on Cognition and Exploratory Learning in Digital Age (CELDA 2007)*, pages 163–172.

Crues, R. W., Bosch, N., Perry, M., Angrave, L., Shaik, N., and Bhat, S. (2018). Refocusing the lens on engagement in MOOCs. In *Proceedings of the Fifth Annual ACM Conference on Learning at Scale - L@S '18*, pages 1–10, New York, New York, USA. ACM Press.

Dahl, D. B. (2016). *xtable: Export Tables to LaTeX or HTML*. R package version 1.8-2.

Dewey, J. (1933). *How We Think: A restatement of the relation of reflective thinking to the educative process*. C. Heath, Boston.

Ferreira, R., Kovanović, V., Gašević, D., and Rolim, V. (2018). Towards Combined Network and Text Analytics of Student Discourse in Online Discussions. In *International Conference on Artificial Intelligence in Education*, volume 10331, pages 111–126. Springer International Publishing.

Gagolewski, M. (2018). *R package stringi: Character string processing facilities*.

Gardner, J., Brooks, C., Andres, J. M., and Baker, R. (2018). Replicating MOOC predictive models at scale. In *Proceedings of the Fifth Annual ACM Conference on Learning at Scale - L@S '18*, pages 1–10, New York, New York, USA. ACM Press.

Garrison, D., Anderson, T., and Archer, W. (2000). Critical Inquiry in a Text-Based Environment: Computer Conferencing in Higher Education. *The Internet and Higher Education*, 2(2-3):87–105.

Garrison, D. R., Anderson, T., and Archer, W. (2001). Critical thinking, cognitive presence, and computer conferencing in distance education. *American Journal of Distance Education*, 15(1):7–23.

Gašević, D., Adesope, O., Joksimović, S., and Kovanović, V. (2015). Externally-facilitated regulation scaffolding and role assignment to develop cognitive presence in asynchronous online discussions. *Internet and Higher Education*, 24:53–65.

Hofman, J. M., Sharma, A., and Watts, D. J. (2017). Prediction and explanation in social systems. *Science*, 355(6324):486–488.

Honnibal, M. and Montani, I. (2017). spaCy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing. https://spacy.io.

Jiang, Y., Bosch, N., Baker, R. S., Paquette, L., Ocumpaugh, J., Andres, J. M. A. L., Moore, A. L., and Biswas, G. (2018). Expert Feature-Engineering vs. Deep Neural Networks: Which Is Better for Sensor-Free Affect Detection? In *International Conference on Artificial Intelligence in Education*, volume 1, pages 198–211. Springer International Publishing.

Joksimović, S., Poquet, O., Kovanović, V., Dowell, N., Mills, C., Gašević, D., Dawson, S., Graesser, A. C., and Brooks, C. (2017). How do we Model Learning at Scale? A Systematic Review of Research on MOOCs. *Review of Educational Research*.

Kim, J., Shaw, E., Feng, D., Beal, C., and Hovy, E. (2006). Modeling and assessing student activities in on-line discussions. In *Proc. of the AAAI Workshop on . . . .*

Kovanović, V. (2017). *Assessing Cognitive Presence using automated learning analytics methods*. PhD thesis, University of Edinburgh.

Kovanovic, V., Joksimovic, S., Gasevic, D., and Hatala, M. (2014). Automated cognitive presence detection in online discussion transcripts. *CEUR Workshop Proceedings*, 1137.

Kovanović, V., Joksimović, S., Poquet, O., Hennis, T., Čukić, I., de Vries, P., Hatala, M., Dawson, S., Siemens, G., and Gašević, D. (2018). Exploring communities of inquiry in Massive Open Online Courses. *Computers and Education*, 119(November 2017):44–58.

Kovanović, V., Joksimović, S., Waters, Z., Gašević, D., Kitto, K., Hatala, M., and Siemens, G. (2016). Towards Automated Content Analysis of Discussion Transcripts: A Cognitive Presence Case. In *LAK '16*, pages 15–24. University of Edinburgh.

Kuhn, M., Wing, J., Weston, S., Williams, A., Keefer, C., Engelhardt, A., Cooper, T., Mayer, Z., Kenkel, B., the R Core Team, Benesty, M., Lescarbeau, R., Ziem, A., Scrucca, L., Tang, Y., Candan, C., and Hunt, T. (2018). *caret: Classification and Regression Training*. R package version 6.0-79.

Kuncheva, L. I. and Rodríguez, J. J. (2018). On feature selection protocols for very low-sample-size data. *Pattern Recognition*, 81:660–673.

Landis, J. R. and Koch, G. G. (1977). The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174.

Le, C. V., Pardos, Z. A., Meyer, S. D., and Thorp, R. (2018). Communication at Scale in a MOOC Using Predictive Engagement Analytics. In Penstein Rosé, C., Martínez-Maldonado, R., Hoppe, H. U., Luckin, R., Mavrikis, M., Porayska-Pomsta, K., McLaren, B., and du Boulay, B., editors, *International Conference on Artificial Intelligence in Education*, volume 10947 of *Lecture Notes in Computer Science*, pages 239–252. Springer International Publishing.

Liaw, A. and Wiener, M. (2002). Classification and regression by randomforest. *R News*, 2(3):18–22.

Manning, C. D., Ragahvan, P., and Schutze, H. (2008). *An Introduction to Information Retrieval*. Cambridge University Press.

McCallum, A. K. (2002). MALLET: A machine learning for language toolkit. http://mallet.cs.umass.edu.

McKlin, T., Harmon, S., Evans, W., and Jones, M. (2001). Cognitive presence in web-based learning: A content analysis of students' online discussions. *Annual Proceedings of Selected Research and Development*, pages 272–277.

McKlin, T. E. (2004). *Analyzing Cognitive Presence in Online Courses Using an Artificial Neural Network*. PhD thesis, Georgia State University.

McNamara, D. S., Graesser, A. C., McCarthy, P. M., and Cai, Z. (2014). *Automated Evaluation of Text and Discourse with Coh-Metrix*. Cambridge University Press, New York, NY, USA.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Distributed Representations of Words and Phrases and Their Compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems*, pages 3111—-3119. Curran Associates Inc.

Mu, J., Stegmann, K., Mayfield, E., Rosé, C., and Fischer, F. (2011). ACODEA: A Framework for the Development of Classification Schemes for Automatic Classifications of Online Discussions. In *Connecting Computer-Supported Collaborative Learning to Policy and Practice: CSCL 2011 Conference Proceedings - Long Papers, 9th International Computer-Supported Collaborative Learning Conference*, pages 438–445.

Neto, V., Rolim, V., Ferreira, R., Kovanovi, V., Gaševi, D., Dueire Lins, R., and Lins, R. (2018). Automated Analysis of Cognitive Presence in Online Discussions Written in Portuguese. *In press*.

Pennington, J., Socher, R., and Manning, C. D. (2014). GloVe: Global Vectors for Word Representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Ramesh, A., Goldwasser, D., Huang, B., Daum, H., and Getoor, L. (2013). Modeling Learner Engagement in MOOCs using Probabilistic Soft Logic. *NIPS Workshop on Data Driven Education*, pages 1–7.

Revelle, W. (2018). *psych: Procedures for Psychological, Psychometric, and Personality Research*. Northwestern University, Evanston, Illinois. R package version 1.8.3.

Revolution Analytics and Weston, S. (2017). *doMC: Foreach Parallel Adaptor for 'parallel'*. R package version 1.3.5.

Schwartz, M. et al. (2015). *WriteXLS: Cross-Platform Perl Based R Function to Create Excel 2003 (XLS) and Excel 2007 (XLSX) Files*. R package version 4.0.0.

Sculley, D. and Wachman, G. M. (2007). Relaxed online svms for spam filtering. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 415–422. ACM.

Tausczik, Y. R. and Pennebaker, J. W. (2010). The psychological meaning of words: LIWC and computerized text analysis methods. *Journal of Language and Social Psychology*, 29(1):24–54.

Thelwall, M., Buckley, K., Paltoglou, G., and Cai, D. (2010). Sentiment Strength Detection in Short Informal Text. *Journal of the American Society for Information Science and Technology*, 61(12):2544–2558.

Torgo, L. (2010). *Data Mining with R, learning with case studies*. Chapman and Hall/CRC.

Wang, X., Yang, D., Wen, M., Koedinger, K., and Rosé, C. P. (2015). Investigating how student's cognitive behavior in MOOC discussion forums affect learning gains. In *Proceedings of the 8th International Conference on Educational Data Mining*, pages 226–233.

Waters, Z., Kovanović, V., Kitto, K., and Gašević, D. (2015). Structure Matters: Adoption of Structured Classification Approach in the Context of Cognitive Presence Classification. In *Lecture Notes in Computer Science*, volume 9460, pages 227–238.

Weimer, M., Gurevych, I., and Mühlhäuser, M. (2007). Automatically assessing the post quality in online discussions on software. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions - ACL '07*, pages 125–128, Morristown, NJ, USA. Association for Computational Linguistics.

Wen, M., Yang, D., and Rosé, C. P. (2014). Sentiment Analysis in MOOC Discussion Forums: What does it tell us? In *Educational Data Mining*.

Whitehill, J., Mohan, K., Seaton, D., Rosen, Y., and Tingley, D. (2017). MOOC Dropout Prediction: How to Measure Accuracy? In *Proceedings of the Fourth (2017) ACM Conference on Learning @ Scale - L@S '17*, pages 161–164.

Wickham, H. (2011). The split-apply-combine strategy for data analysis. *Journal of Statistical Software*, 40(1):1–29.

Wise, A. F. and Cui, Y. (2018). Unpacking the relationship between discussion forum participation and learning in MOOCs. In *Proceedings of the 8th International Conference on Learning Analytics and Knowledge - LAK '18*, pages 330–339.

Appendix A

Model tuning parameters

In the random forest model, the `mtry` parameter controls the number of features available as candidates at each split point. The specific values that are tested during model tuning are automatically determined by the `caret` library based on the number of features in the model. The values of `mtry` that were tested in Experiment 2 are shown in Table A.1, while the values tested in Experiment 3 are shown in Table A.2.

Feature Count	Values of `mtry` tested									
206	2	12	23	34	44	55	66	77	87	98
	109	120	130	141	152	163	173	184	195	206
209	2	12	23	34	45	56	67	78	89	100
	110	121	132	143	154	165	176	187	198	209
208	2	12	23	34	45	56	67	77	88	99
	110	121	132	142	153	164	175	186	197	208
211	2	13	24	35	46	57	68	79	90	101
	112	123	134	145	156	167	178	189	200	211

Table A.1: The 20 values of the `mtry` parameter tested in each model in Experiment 2. The specific values to be tested were automatically determined by the `caret` library based on the number of features in the model.

Feature Count	Values of `mtry` tested									
221	2	13	25	36	48	59	71	82	94	105
	117	128	140	151	163	174	186	197	209	221
224	2	13	25	37	48	60	72	83	95	107
	118	130	142	153	165	177	188	200	212	224
226	2	13	25	37	49	60	72	84	96	108
	119	131	143	155	167	178	190	202	214	226
236	2	14	26	38	51	63	75	88	100	112
	125	137	149	162	174	186	199	211	223	236
238	2	14	26	39	51	64	76	88	101	113
	126	138	151	163	175	188	200	213	225	238
239	2	14	26	39	51	64	76	89	101	114
	126	139	151	164	176	189	201	214	226	239
241	2	14	27	39	52	64	77	90	102	115
	127	140	152	165	178	190	203	215	228	241
244	2	14	27	40	52	65	78	91	103	116
	129	142	154	167	180	193	205	218	231	244
246	2	14	27	40	53	66	79	91	104	117
	130	143	156	168	181	194	207	220	233	246
256	2	15	28	42	55	68	82	95	108	122
	135	149	162	175	189	202	215	229	242	256
271	2	16	30	44	58	72	86	101	115	129
	143	157	171	186	200	214	228	242	256	271
274	2	16	30	44	59	73	87	102	116	130
	145	159	173	188	202	216	231	245	259	274
276	2	16	30	45	59	74	88	102	117	131
	146	160	175	189	203	218	232	247	261	276
286	2	16	31	46	61	76	91	106	121	136
	151	166	181	196	211	226	241	256	271	286
301	2	17	33	49	64	80	96	112	127	143
	159	175	190	206	222	238	253	269	285	301

Table A.2: The 20 values of the `mtry` parameter tested in each model in Experiment 3. The specific values to be tested were automatically determined by the `caret` library based on the number of features in the model.