Finding Black Cat in a Coal Cellar - Keyphrase Extraction, Keyphrase Classification & Keyphrase-Rubric Relationship Extraction from Complex Assignments

Manikandan Ravikiran
mravikiran3@gatech.edu

***Abstract*—**Diversity in content and open-ended questions are part and parcel of complex assignments across both classroom and online graduate programs. Inherent scale of these programs poses variety of challenges across both peer and expert feedback including rogue reviews. While identification of important content and relating it to predefined rubrics would simplify and improve the grading process, the research till date is still in a nascent stage. As such in this paper we present a very first work on keyphrase extraction, keyphrase classification and keyphrase-rubric relationship extraction from complex assignments. Through our study, we find that (i) the task of keyphrase classification is ambiguous at a human level with Cohen's kappa of 0.77, (ii) unsupervised approaches are better suited for keyphrase extraction (iii) supervised approaches produce best results for keyphrase-rubric relationship extraction. We finally present extensive analysis and derive practical observations for those interested in these tasks for the future.

1 INTRODUCTION

Graduate programs with MOOC form of delivery (MOOC-Masters) are inherently dependent on peer and expert feedback (Joyner, 2017). While peer feedback focuses on incorporating pedagogical benefits, expert feedback provides improved assessment. MOOC-Masters programs offer inherent benefit of high scaling at lower costs. However, with the scale of these programs comes two major issues first of which is assignment of multiple students to single expert due to lack of large number of experts and high monetary costs to students (Joyner,

Finding Black cat in a Coal Cellar - This is because, we need to find phrases that are needed for purpose of scoring, which is a hard task due its similarity with other contents in the assignments.

2017). On the other side, peer feedback is typically plagued by problem of insufficient reviews (Geigle, Zhai, and Ferguson, 2016) due to dishonesty, retaliation, competition or laziness of the peers (Kulkarni, Bernstein, and Klemmer, 2015).

Automatic grading systems are a key in curbing the effects of the previously mentioned problems across both peer and expert feedback. However current automatic grading systems are typically restricted to addressing one or more of these previously mentioned problems across various stages of the feedback process. More specifically, till date we have automatic scoring systems that compute scores directly from textual essays, grading accuracy improvement tools that adjust scores based on aggregation, modeling, calibration and ranking strategies (Reily, Finnerty, and Terveen, 2009). To avoid bias and retaliation, we have tools that focus better peer and expert allotments (Ardaiz-Villanueva et al., 2011). Finally we also have review analysis tools that focus on understanding and improving the contents of a review comment.

Most of the previously described works address the respective problems positively yet, there are some remaining systematic issues including i) changes in reviewers rating with time and length of assignments leading to lack of effective response by the expert ii) discrepancy between rating expected by the author to that of expert and the peer iii) random fluctuations of scores etc. iv) lack of descriptive reviews owing to scale in both peer and expert feedback. Moreover, with complex assignments - Assignment with open-ended questions commonly seen in MOOC based graduate programs with large and diverse content, the problems further exacerbate.

Considering these problems and nature of complex assignments, we conjecture that identification of relevant contents needed for feedback process would mitigate the previously mentioned issues. For example, in the case of complex long assignments, extracting contents that are needed for rubric alone would reduce time spent on the assignment and in turn lead to fruitful reviews. Hence in this work, we propose to identify important contents useful for feedback. More specifically, we quantify *How AI can be used to identify important contents of complex assignments?* by empirically evaluating following open research questions

- **RQ1:** To what extent can AI extract important phrases from the complex assignments?. *We analyzed supervised and unsupervised keyphrase extraction ap-*

proaches, to find unsupervised ranking approaches to be ideal suit for keyphrase extraction from complex assignment with maximum F1 of 0.63.

- **RQ2:** How does AI fair up in classifying the extracted keyphrases from complex assignments? *We study and compare the effectiveness of traditional classifiers and recent language models. Through this, we found that both pretrained language models and simple TFIDF-SVM classifiers produce similar results with a former producing average of 0.06 F1 higher than the latter*
- **RQ3:** To what extent can AI relate the identified phrases to scoring rubrics? *We compared performance on supervised language models and unsupervised clustering, topic modeling approaches to find supervised approaches to be more effective with maximum F1 of 0.48.*

In the due process, we introduce a theory grounded annotation scheme, novel dataset, and present exhaustive experimental results. Additionally in line with prior works we carry out corpus analysis and introduce briefly the methods used for experimentation. Unlike prior efforts, however, our main objectives are to uncover the impact of content and context diversity on performance (measured primarily by F1 score), and also to study the benefits of various approaches.

The rest of the paper is organized as follows, in section 2 we present literature covering works related to research questions, followed by datasets and methods used in section 3. Section 4 presents experiments and results. Finally in section 5 we conclude overall findings with some possible implications on future work.

2 RELATED WORK

In this section, we present literature on various works that are closely related to the research questions so proposed in section 1.

2.1 Key Phrase Extraction

Key phrase extraction focuses on extracting short important phrases that represent a larger piece of content. Large body of works exists in both general and its application educational domain beginning with (Zhang et al., 2016) which focused on the problem of automatically extracting keyphrases from tweets, followed by (Al-Zaidy, Caragea, and Giles, 2019) that addresses the keyphrase extraction problem as sequence labeling and (Patel and Caragea, 2019) where the authors present an empirical study on keyphrase extraction focusing on formulation of sequence labeling using Bi-LSTM CRF. Multiple other works exists

including that of (Chowdhury, Caragea, and Caragea, 2019), (Wang et al., 2019), and (Zhang and Zhang, 2019). Alternatively, unsupervised approaches which has recently gained a lot of traction with much of the work being formulated as the ranking problem, beginning with work of (Bennani-Smires et al., 2018), which proposed proposes an unsupervised key phrase extract approach from a single document using Embedrank. Then there are (Liu et al., 2009) finds exemplar terms by leveraging clustering techniques, which guarantees the document to be semantically covered by these exemplary terms. Similar works include those on scientific documents (Bhaskar, Nongmeikapam, and Bandyopadhyay, 2012), (Nguyen and Kan, 2007) and few more on educational applications (Badawy et al., 2018) & (Gollapalli, Li, and Yang, 2017).

Our work, while similar to many of the earlier works described in the educational domain with a focus on key phrase extraction from educational data, however, the work differs in three ways. Firstly, we propose to extract key phrases from complex assignments where unlike scientific reports, many of the key phrases are semi-formal. Second, unlike earlier works, we tend to use key phrases for automatic linking to rubrics. Finally, unlike much of keyphrases extraction we focus on complex assignments from multiple courses and multi-domains.

2.2 Minimal Meaningful Proposition

Minimal Meaningful Propositions (MMP)is a decomposition of text, such as a question's answer, into the set of propositions that individually represent single minimal claims or arguments that cannot be further decomposed without losing contextual meaning, and taken as a whole represent the entire meaning of the text(Godea, Bulgarov, and Nielsen, 2016). There are multiple extensions to MMP's notable work on this includes MMP alignment(Bulgarov and Nielsen, 2017) where it focuses on aligning students' response MMP with appropriate actual gold answer MMP's. Also, there are works by (Bulgarov, 2018) which proposes to address the problem of formative assessment by clustering the student responses into groups that suggest similar conceptual beliefs where the author proposes to group responses. Finally, MMP's are also used in improving student-teacher engagement(Bulgarov, 2018), to which works of in addition to MMP's add annotations such that each student response is graded relative to its *engagingness*.

Our work could be argued to be the same as MMP's however MMP's have a predefined set of classes as mentioned in (Godea, Bulgarov, and Nielsen, 2016), which is different from what were propose to extract. Second, MMP's only look for a propositional statement, which is not the case with the proposed idea. However, the approaches for the extraction of key phrases and MMP's are similar.

2.3 Text Classification in Educational Technology

Text classification is a long-standing problem in educational technology, which has gone rapid increase with the advent of large scale MOOCs. The works vary according to their final intended goal itself, resulting in a diverse range of datasets, features, and algorithms. The earliest works use a classification approach on clickstream datasets(Yang et al., 2015). Then there is works include that of Scott et al. (2015) which *analyzed three tools*. Then there are also sentiment related works, by(Ramesh et al., 2014) which included linguistic and behavioral features of MOOC discussion forums. Works on similar lines include(Liu et al., 2016),(Tucker, Dickens, and Divinsky, 2014). On the parallel side, some works use posts and their metadata to detect confusion in the educational contents. Notable work by (Akshay et al., 2015), *emphasizes the capacity of posts to improve content creation*. Additionally, there are works on post urgency classification (Omaima, Aditya, and Huzefa, 2018), speech act prediction (Jaime and Kyle, 2015). Finally, some works focus on using classification towards peer feedback, much of which focuses on scaffolding the review comments themselves (Xiong, Litman, and Schunn, 2010), (Nguyen, Xiong, and Litman, 2016), (Ramachandran, Gehringer, and Yadav, 2016),(Cho, 2008).

Similar to above works, we focus on classification of educational text, however, there is two major difference in our work. First of all being, we focus on assignment text, rather than posts which is predominant. Second, being the criteria and end application of classification is peer feedback. Thirdly, rather than scaffolding peer feedback, we focus on extracting textual content suitable for easy peer feedback.

2.4 Peer Feedback & Grading

Peer Feedback and grading are integral to online education. There are a plethora of works in this line tackling a wide array of problems. Firstly some works focus on improving grading score accuracy. Notable works, in this line, include that of (Reily, Finnerty, and Terveen, 2009) the work focuses on improving the score

through aggregation strategies. Following this, there are also (, 2013) which formulated and evaluated a probabilistic peer grading graphical model (PGM) for estimation of submission grades as well as grader bias and reliability. Multiple other works use PGM's models(He, Hu, and Sun, 2019) which use Markov Chain Monte Carlo approaches and(Wu et al., 2015) which uses a Fuzzy Cognitive Diagnosis Framework (Fuzzy CDF).

Alternatively, rather than improving the accuracy of scores, there is also work on the assignment of correct reviewers by forming effective groups of peers. Some of the works in this line(Ardaiz-Villanueva et al., 2011), (Ounnas, 2010). Similarly, there are (Pollalis and Mavrommatis, 2008) which proposes a method for course construction such that collaboration is easily achievable with a locus on common educational goals. Then we have (Lynda et al., 2017) which proposes to address peer assessment based on a combination of profile-based clustering and peer grading with the treatment of results. There are many other works in a similar line with the usage of automatic algorithms to form groups (Graf and Bekele, 2006), (Fahmi and Nurjanah, 2018), and (Bekele, 2006).

In general much of the work focuses on peer grading either towards accuracy improvement, automatic grading, and reviews. However, in our work we focus towards providing evidence-based grading, where identify and link the important parts of the contents so that peer review is simplified and also evidence is provided along with the scores.

3 DATASET, ANNOTATION AND METHODS

In this section, we present dataset and its annotation scheme, along with algorithms used across experiments in section 4.

3.1 Annotation Scheme

Human writing is a hierarchical process with the multilevel organization of content, with a focus on continually changing goals, where the content changes and improves as it progress. While the (Flower and Hayes, 1981), was developed in the context of general writing, we hypothesize such an observation should be true for complex assignments. Additionally works of (Hattie and Timperley, 2007) typically the feedback is expected to provide answers to student's questions which includes *Where am I going? (What are the goals?), How am I going? (What progress is being made toward the goal?), and Where to next? (What activities*

need to be undertaken to make better progress?) (Hattie and Timperley, 2007). Based on these characteristics we propose to decompose a large complex assignment into a series of small important phrases, where each phrase expresses a meaningful part of the assignments. More specifically we use the annotation scheme in Table 1 which consists of four different categories of important phrases that could be extracted and useful for grading complex assignments. The four different types of phrases include

- **Task** - Representing activity done by the author.
- **Findings** - Indicating the output of activity.
- **Reasons** - Showing reasons behind the findings.
- **Intuition** - Showing background on why the task was executed.

Table 1—Annotation Schema for Identification of Important Phrases from Complex Assignment

Class	Description	Example
Task	Task indicates an activity explored by the student	We use using pixel based visual representations for images and develop an production system with series of rules
Findings	Findings indicates results of a Task	The agent only solved 2CP's in both sets, which adheres to existing rules, suggesting better rules and analysis are needed to handle CP's
Reason	Reason indicates the rationale behind the finding	Approximate similarity property can be seen in BP-E 9, but the result was erroneous
Intuition	Intuition represents the reason behind the Task	These problems satisfy simple relationships such as XOR, Overlay, Identity etc.

Table 2 shows relationship the between annotation (coding) scheme, works of (Hattie and Timperley, 2007) and (Flower and Hayes, 1981). As we can see the classes in coding scheme we developed accomodates what is needed as part of human writing and peer feedback.

Table 2—Relationship between annotation scheme, works of (Hattie and Timperley, 2007) and (Flower and Hayes, 1981).

Coding Scheme	Relationship to (Hattie and Timperley, 2007)	Relationship to (Flower and Hayes, 1981)
Task	What are the goals?	Hierarchical, Goal Oriented
Findings	Where am I going?	Goal Oriented
Reason	How am I going?	Goal Oriented
Intuition	What activities need to be undertaken to make better progress?	Hierarchical, Goal Oriented, Distinctive thinking

3.2 Dataset

The dataset in this work was developed using the four KBAI report of Fall 2019. The overall dataset statistics are as shown in Table 3 below. The data consists of 791 phrases. Out of this we have 331 phrases that doesn't fit into any of the four classes. We call this as *Other*. Rest of 460 phrases are divided into train and test sets respectively[1]. Further, the dataset consists of 2443 unique words with minimal occurrence of 1 and a maximum occurrence of 800.

Considering the various research questions, the data is split into two folds used for overall cross-validation. Each of the phrase was subject to two rounds of annotation resulting in **Cohen's Kappa(κ) of 0.77** showing that the resultant task is hard and may require more complex semantically relevant features. In the due process, we can see that the dataset is imbalanced across the four classes with *Task* and *Finding* categories dominating the corpus and *Reason* being the least seen sentences. This behavior is because of the semi-formal nature of writing where the majority of sentences focus on inferences drawn from results.

Table 3—Dataset statistics and annotation statistics created for this work.

Splits	**Train**				**Test**				**Other**
	T	**F**	**R**	**I**	**T**	**F**	**R**	**I**	
Fold-1	69	140	25	32	58	90	12	24	
Fold-2	71	140	22	43	56	90	23	15	
Total	276				184				331

Also, same dataset for evaluation of all the research questions. The relative mapping of coding scheme from Table 1 to annotations across keyphrase extraction, keyphrase classification and keyphrase-rubric relationship extraction is as shown in Table 4. According to Table 4, for example, *These problems satisfy simple relationships such as XOR, Overlay, Identity, etc.* will be considered as keyphrase during keyphrase identification, intuition during keyphrase classification and will be related to Agent Reasoning class during keyphrase-rubric relationship extraction.

1 Due to data privacy, the overall data that was used in this work is very limited. We plan to revisit this again in future.

Table 4—Mapping of coding schemes to classes in each tasks. We use same data across answering three different RQ's.

Coding Scheme	Keyphrase Extraction	Keyphrase Classification	Keyphrase Rubric Relationship	Examples
Task	Keyphrase	Task	Project Overview	We use using pixel based visual representations for images and develop an production system with series of rules
Finding		Finding	Cognitive Connection	The agent only solved 2 CP's in both sets, which adheres to existing rules, suggesting better rules and analysis are needed to handle CP's
Reason		Reason	Relation to KBAI Class	Approximate similarity property can be seen in BP-E9, but the result was erroneous
Intuition		Intuition	Agent Reasoning	These problems satisfy simple relationships such as XOR, Overlay, Identity etc.
Other	Non-Keyphrase			The agent was submitted on 11:59 GMT

3.3 Metrics

We use the following evaluation metrics in this work. Respective metrics for analyzing each RQ will be briefed in section 4.

- **Precision (P), Recall (R) and F1-Score (F1):** Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. The recall is the ratio of correctly predicted positive observations to all observations in the actual class. F1 Score is the harmonic mean of Precision and Recall.
- **Cluster Homogeneity (HOMO):** Homogeneity is computed by assigning each cluster to the class which is most frequent in the cluster, and then the accuracy of this assignment is measured by counting the number of correctly assigned phrases and dividing by N.

$$\text{Homogeneity}(\Omega, \mathbb{C}) = \frac{1}{N} \sum_{k} \max_{j} |\omega_k \cap c_j| \tag{1}$$

  where $\Omega = \{\omega_1, \omega_2, \ldots, \omega_K\}$ is the set of clusters and $\mathbb{C} = \{c_1, c_2, \ldots, c_J\}$ is the set of classes. We interpret ω_k as the set of phrases and c_j as the set of input phrases which are classified.
- **Rand Index (RI):** An alternative to this *purity of clusters* one can view clustering as a series of decisions, one for each of the $N(N-1)/2$ pairs of phrases in the collection. We want to assign two phrases to the same cluster if and only if they are similar. A true positive (TP) decision assigns two similar phrases to the same cluster, a true negative (TN) decision assigns two dissimilar phrases to different clusters. There are two types of errors we can commit. An (FP) decision assigns two dissimilar phrases to the same cluster. An (FN) decision

assigns two similar phrases to different clusters. The Rand index (RI) measures the percentage of correct decisions. That is, it is simply accuracy calculated as

$$\mathrm{RI} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{FP} + \mathrm{FN} + \mathrm{TN}} \tag{2}$$

In this section, we shall present the feature extraction and supervised/unsupervised algorithms used for classification.

3.4 Algorithms

In this section, we shall present in brief various algorithms used for the experiments in section 4. A detailed presentation is beyond the scope of the current paper and we invite the readers to look at the respective original works.

3.4.1 *Feature Extraction*

In this work, we use three different approaches for feature extraction namely BERT, TF-IDF and Latent Dirichlet Allocation(Blei, Ng, and Jordan, 2003) coupled with the former two.

1. **BERT**: BERT (Devlin et al., 2019) is a trained Transformer Encoder stack, with 12 encoders as part of the base models and 24 in the large version. BERT encoders typically have larger feed-forward networks (768 and 1024 nodes in the base and large respectively) and more attention heads (12 and 16 respectively). BERT was originally trained on Wikipedia and Book Corpus, a dataset containing +10,000 books of different genres. BERT works like a transformer encoder stack, by taking a sequence of words as input which keeps flowing up the stack from one encoder to the next, while new sequences are coming in. The final output for each sequence is a vector of size 728. We will use such vectors for our phrase classification problem without fine-tuning.
2. **XLNet:** XLNet (Yang et al., 2019) is a large bidirectional transformer that uses improved training methodology, larger data and more computational power to achieve better than BERT prediction metrics on 20 language tasks. To improve the training, XLNet introduces permutation language modeling, where all tokens are predicted but in random order. This contrasts with BERT's masked language model where only the masked (15%) tokens are predicted. This is also in contrast to the traditional language models, where all tokens were predicted in sequential order instead of random order. This helps the model to learn bidirectional relationships and therefore better handles depen-

dencies and relations between words. Besides, Transformer XL was used as the base architecture, which showed good performance even in the absence of permutation-based training. XLNet was trained with over 130 GB of textual data and 512 TPU chips running for 2.5 days, both of which are much larger than BERT.

3. **RoBERTa**: RoBERTa (Liu et al., 2019) from Facebook, robustly optimized BERT approach (RoBERTa), is a retraining of BERT with improved training methodology, 1000% more data and compute power. To improve the training procedure, RoBERTa removes the Next Sentence Prediction (NSP) task from BERT's pre-training and introduces dynamic masking so that the masked token changes during the training epochs. Larger batch-training sizes were also found to be more useful in the training procedure. Importantly, RoBERTa uses 160 GB of text for pre-training, including 16GB of Books Corpus and English Wikipedia used in BERT. The additional data included Common Crawl News dataset (63 million articles, 76 GB), Web text corpus (38 GB) and Stories from Common Crawl (31 GB). This coupled with 1024 V100 Tesla GPU's running for a day, led to the pre-training of RoBERTa. As a result, RoBERTa outperforms both BERT and XLNet on GLUE benchmark results.
4. **XLM:** Though BERT was trained on over 100 languages, it wasn't optimized for multi-lingual models — most of the vocabulary isn't shared between languages and therefore the shared knowledge is limited. To overcome that, XLM (Lample and Conneau, 2019) modifies BERT in the following ways.
   - First, instead of using word or characters as the input of the model, it uses Byte-Pair Encoding (BPE) that splits the input into the most common subwords across all languages, thereby increasing the shared vocabulary between languages. This is a common pre-processing algorithm and a summary of it can be found here.
   - Second, it upgrades the BERT architecture in two manners: Each training sample consists of the same text in two languages, whereas in BERT each sample is built from a single language. As in BERT, the goal of the model is to predict the masked tokens, however, with the new architecture, the model can use the context from one language to predict tokens in the other, as different words are masked words in each language (they are chosen randomly). The model also receives the language ID and the order of the tokens in each language, i.e. the Positional Encoding, separately. The new metadata helps the model learn the relationship between related tokens in

different languages.

5. **Term Frequency (TF):** Term Frequency a.k.a BOW or Term Count measures how frequently a term occurs in a document. Since every document is different in length, multiple tokens would appear much more in long documents than shorter ones. Thus, the term frequency is often divided by the document length (aka. the total number of terms in the document) as a way of normalization.

$$TF(t) = \frac{\text{Number of times term t appears in a document}}{\text{Total number of terms in the document}} \tag{3}$$

6. **Term Frequency Inverse document frequency (TFIDF):** The TFIDF algorithm builds on the following representation of sentences/documents. Each sentence d is represented as a vector $d = (d_1, d_2, ..., d_F)$ so that documents with similar content have similar vectors (according to a fixed similarity metric). Each element d_i represents a distinct word w_i. d_i for a sentence d is calculated as a combination of the statistics $TF(w_i; d)$ and $DF(w_i)$. The term frequency $TF(w_i; d)$ is the number of times word w_i occurs in document d and the document frequency $DF(w_i)$ is the number of documents in which word w_i occurs at least once. The inverse document frequency IDF (w_i) can be calculated from the document frequency.

$$IDF(w_i) = \log\left(\frac{D}{DF(w_{(i)})}\right) \tag{4}$$

Here, D is the total number of sentences. Intuitively, the inverse document frequency of a word is low if it occurs in many documents and is highest if the word occurs in only one. The so-called weight d(i) of word w_i in sentence d^i is then given as

$$d^{(i)} = TF(w_i; d)IDF(w_i) \tag{5}$$

This word weighting heuristic says that a word w_i is an important indexing term for document d if it occurs frequently in it (the term frequency is high). On the other hand, words that occur in many documents are rated less important indexing terms due to their low inverse document frequency. In this work, we use TF-IDF with Bi-Grams of words.

7. **Latent Dirichlet Allocation (LDA):** Originally introduced by works of (Blei, Ng, and Jordan, 2003) is a generative statistical model that allows sets of observations to be explained by unobserved groups that explain why some parts of the data are similar. For example, if observations are words collected into documents, it posits that each document is a mixture of a small number of topics and that each word's presence is attributable to one of the document's topics.

3.4.2 *Clustering Algorithms*

1. **K-Means:** The k-means algorithm is used to partition a given set of observations into a predefined amount of k clusters. The algorithm starts with a random set of k center-points (μ). During each update step, all observations x are assigned to their nearest center-point (see equation 6). In the standard algorithm, only one assignment to one center is possible. If multiple centers have the same distance to the observation, a random one would be chosen.

$$S_i^{(t)} = \left\{ x_p : \left\| x_p - \mu_i^{(t)} \right\|^2 \leqslant \left\| x_p - \mu_j^{(t)} \right\|^2 \forall j, 1 \leqslant j \leqslant k \right\} \tag{6}$$

Afterward, the center-points are re-positioned by calculating the mean of the assigned observations to the respective center-points (see Equation 7).

$$\mu_i^{(t+1)} = \frac{1}{|S_i^{(t)}|} \sum_{x_j \in S_i^{(t)}} x_j \tag{7}$$

The update process reoccurs until all observations remain at the assigned center-points and therefore the center-points would not be updated anymore. This means that the k-means algorithm tries to optimize the objective function 8. As there is only a finite number of possible assignments for the number of centroids and observations available and each iteration has to result in a better solution, the algorithm always ends in a local minimum.

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} ||x_n - \mu_k||^2 \tag{8}$$

$$\text{with } r_{nk} = \begin{cases} 1 & x_n \in S_k \\ 0 & \text{otherwise} \end{cases}$$

The main problem of k-means is its dependency on the initially chosen centroids. The centroids could end up in splitting common data points whilst

other, separated points get grouped if some of the centroids are more attracted by outliers. These points will get pulled to the same group of data points. Hence we test three different types of initialization namely K-Means++, Random and PCA components with maximal variance

2. **Aggolomerative Hierarchical Clustering:** This algorithm works by grouping the data one by one based on the nearest distance measure of all the pairwise distance between the data point. Again distance between the data point is recalculated but which distance to consider when the groups have been formed. For this, there are many available methods. Some of them are *single linkage, complete linkage, average linkage, centroid distance, and ward's distance.* In this work, we use the ward's method and group the data until one cluster is formed. We further use three methods for affinity (distance computation for linkage approaches) namely Euclidean, Cosine & City Block Distance.
3. **Spectral Clustering:** Given an enumerated set of data points, the similarity matrix may be defined as a symmetric matrix A, where $A_{ij} \geqslant 0$ represents a measure of the similarity between data points with indices i and j. The general approach to spectral clustering is to use a standard clustering method such as K-Means on relevant eigenvectors of a Laplacian matrix of A. There are many different ways to define a Laplacian which have different mathematical interpretations, and so the clustering will also have different interpretations. The relevant eigenvectors are the ones that correspond to the smallest several eigenvalues of the Laplacian except for the smallest eigenvalue which will have a value of 0. For computational efficiency, these eigenvectors are often computed as the eigenvectors corresponding to the largest several eigenvalues of a function of the Laplacian.

3.4.3 *Supervised Keyphrase Extraction Algorithms*

1. **KEA:** KEA (Witten et al., 1999) keyphrase extraction algorithm is a supervised approach for retrieval of important phrases from the text. Originally, KEA uses the Naive Bayes Machine Learning Algorithm for training a binary classifier where the phrases that are used with a dataset that is cleaned linguistically by removing stop words, proper names, case-folding, stemming, etc. Each of these sentences is then converted into a TF-IDF matrix as shown earlier. KEA doesn't use any controlled vocabulary instead uses keyphrases from the input text itself. Additionally, the usage of TF-IDF shows that KEA indeed uses lexical information to extract and characterize the phrases from

the document.

2. **WINGNUS:** WINGNUS (Nguyen and Luong, 2010), is similar to KEA except it uses document structure to mine the required keyphrases. WINGUS similar to KEA uses Naive Bayes classifier with TF-IDF, word-offset, phrase length, typeface attribute, title information, title overlap, features indicating phrase appearance and appearence frequencies in Header, Abstract, Introduction, and other sections of documents.

3.4.4 *Unsupervised Keyphrase Extraction Algorithms*

1. **KPMINER:** KPMINER (El-Beltagy and Rafea, 2010) uses a three-step process involving keyphrase selection, weight calculation, and refinement respectively. Candidate keyphrase selection is done using series of rules predominantly seen in textual documents including separation by punctuation mark without any stop words within a given candidate, this is followed by minimal phrase appearence characteristic represented by least allowable seen frequency factor (n) and finally, a so a cutoff constant (*CutOff*) is defined in terms of several words after which if a phrase appears for the first time, it is filtered out and ignored. This will return a series of candidates which are weighed using TF-IDF supplemented with boosting factors. Finally, the weighed candidate phrases are refined with an input keyphrase selection parameters. Overall KPMINER uses document-related information for elucidating the keyphrases.
2. **YAKE:** YAKE (Campos et al., 2020) is a light-weight unsupervised automatic keyword extraction method which rests on statistical text features extracted from single documents to select the most relevant keywords of a text. YAKE is domain-independent, data-independent and free of TF-IDF usually seen in supervised methods. YAKE consists of five major steps namely (1) text preprocessing and candidate term identification - similar to supervised keyphrase extraction, it preprocesses documents (2) feature extraction - uses a set of statistical features to represent the candidates. Typically used features include TF, TF - upper case letters, co-occurrence matrix, sentence offsets, etc. (3) computing term score - these features are heuristically combined into a single score likely to reflect the importance of the term (4) n-gram generation and computing candidate keyword score - generates n-grams and assigns importance scores and (5) data de-duplication and ranking - Finally, terms are ranked based on importance through de-duplication distance similarity mea-

sure.

3.4.5 *Graph based Ranking Algorithms*

1. **TextRank:** TextRank (Mihalcea and Tarau, 2004) constructs graph representations for the input text data following which a graph-based ranking algorithm is then applied to extract the important lexical units (words) in the text. TextRank as part of its implementation uses words as nodes with lexical information such as art-of-speech (nouns and adjectives) and edges are co-occurrence relations, which are managed through word occurrence distances. Following this, the nodes are ranked using an unweighted graph-based ranking algorithm where the iteration of the graph-based ranking algorithm is done until convergence and vertices are sorted based on final scores. Finally, the values attached to each vertex for ranking/selection decisions.
2. **SingleRank:** SingleRank (Wan and Xiao, 2008) originally employs the clustering algorithm to group the documents into a few clusters. The documents within each cluster are expected to be topic-related and each cluster can be considered as a context for any document in the cluster. For each of the cluster, it executes two major steps, first, it does cluster level word evaluation where it builds an affinity graph similar to TextRank followed by cluster-level saliency score is calculated for each word using graph ranking. Following this candidate phrases in the document based on the scores of the words contained in the phrases are evaluated to choose final keyphrases.
3. **TopicRank:** TopicRank (Bougouin, Boudin, and Daille, 2013) is similar to SingleRank where it uses clustering for selecting representative candidates where the document is pre-processed (sentence segmentation, word tokenization, and Part-of-Speech tagging) and keyphrase candidates are clustered into topics. Then, topics are ranked according to their importance in the document and keyphrases are extracted by selecting one keyphrase candidate for each of the most important topics.
4. **PositionRank:** PositionRank (Florescu and Caragea, 2017) algorithm involves three essential steps: (1) the graph construction at word level similar to TextRank (2) the design of Position-Biased PageRank - where unlike page rank PositionRank is to assign larger weights (or probabilities) to words that are found early in a document and are frequent. (3) the formation of candidate phrases where Candidate words that have contiguous positions in a document are concatenated into phrases.

5. **MultipartiteRank:** MultipartiteRank (Boudin, 2018) extends PositionRank, where Keyphrase candidates are selected from the sequences of adjacent nouns with one or more preceding adjectives (/Adj*Noun+/). They are then grouped into topics based on the stem forms of the words they share using hierarchical agglomerative clustering with average linkage.

4 RESULTS, EXPERIMENTS & DISCUSSIONS

In this section, we present each of the research questions in brief along with their results.

4.1 RQ 1: Keyphrase Extraction

Keyphrase extraction, depending on algorithms used is done in two different formulations. In case of supervised approach, keyphrase extraction is treated as a classification of sentences into keyphrases and non-keyphrases. In case of unsupervised approaches, keyphrase approaches extracts small parts of sentences (2-3 words), rather than entire sentence. In such cases we use fuzzy matching between the original phrases and extracted small parts of the sentences to generate final list of keyphrases (See 1).

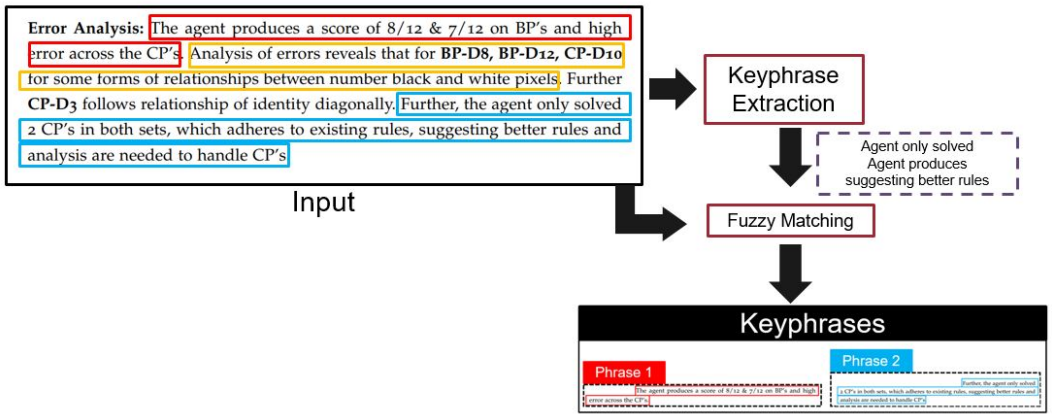

Figure 1—Keyphrase extraction process with Fuzzy Matching.

4.1.1 *RQ 1.1: Supervised Keyphrase Extraction*

Our first research question focuses on the effectiveness of supervised keyphrase extraction approaches in the complex assignment. In this work, we test two approaches, namely KEA and WINGNUS for supervised keyphrase extraction (Section 3.4.3) by analyzing Precision (P), Recall (R) and F1 metrics under the

setting of both weighed average and macro average (See Table 5). The weighted average is to make results comparable across methods due to an imbalance in data. Both supervised approaches of KEA and WINGNUS internally use TF-IDF as feature extractor (See sections 3.4.1) with the Naive Bayes classifier. Additionally, the latter approach uses document related features (See section 3.4.3).

First look at the results in Table 5 we can see the WINGNUS overcomes KEA by 2% F1 and 3% in F1 with similar P & R. Besides, with WINGNUS, the performance is relatively higher because of use of document and phrase-level features.

Table 5—Results of phrase classification using supervised, unsupervised and graph-based approaches.

	Fold 1							Fold 2						
		Macro Avg			Weighed Average				Macro Avg			Weighed Average		
	Accuracy	P	R	F	P	R	F	Accuracy	P	R	F	P	R	F
KEA	0.58	0.61	0.62	0.58	0.68	0.58	0.59	0.50	0.5	0.25	0.33	1.00	0.50	0.67
WINGNUS	0.61	0.61	0.62	0.60	0.67	0.61	0.63	0.58	0.5	0.29	0.37	1.00	0.58	0.73
KP MINER	0.51	0.54	0.55	0.51	0.60	0.51	0.52	0.83	0.5	0.41	0.45	1.00	0.83	0.90
YAKE	0.62	0.57	0.58	0.57	0.62	0.62	0.62	0.94	0.5	0.47	0.48	1.00	0.94	0.97
TOPIC RANK	0.64	0.58	0.58	0.58	0.63	0.64	0.63	0.75	0.5	0.38	0.43	1.00	0.75	0.86
TEXT RANK	0.35	0.61	0.51	0.29	0.70	0.35	0.22	0.85	0.5	0.42	0.46	1.00	0.85	0.92
SINGLE RANK	0.35	0.58	0.51	0.28	0.67	0.35	0.20	0.57	0.5	0.28	0.36	1.00	0.57	0.72
POSITION RANK	0.45	0.56	0.55	0.45	0.62	0.45	0.43	0.90	0.5	0.45	0.47	1.00	0.90	0.95
MULTIPARTITE RANK	0.67	0.63	0.64	0.64	0.68	0.67	0.67	0.94	0.5	0.48	0.48	1.00	0.94	0.97

Comparing results across folds, we can see that compared to fold-1 results are lower in fold-2. We can attribute this to the fold-2 creation process where we ensured that fold-2 has a higher overlap of vocabulary compared to fold-1. (Section 3). However, more detailed linguistic analysis is warranted to understand the impact of specific words on keyphrases and non-keyphrases. We leave such a study for our future work. In fold-1 we can see balanced precision and recall, meanwhile, in fold-2 the precision is always 0.5, this is because in fold-2 both the algorithms cannot identify the non-keyphrases.

Table 6—Results of KEA algorithm with class-wise separation.

		Fold 1			Fold 2		
		P	R	F1	P	R	F1
Classes	**Non-keyphrases**	0.42	0.76	0.54	0	0	0
	Keyphrases	0.8	0.49	0.58	1	0.5	0.66
Metrics	**Macro Avg**	0.61	0.62	0.58	0.5	0.25	0.33
	Weighed Average	0.68	0.58	0.59	1	0.5	0.67

Detailed results of KEA across folds 1 and 2 are as shown in Table 6. As we

can see in Table 6, the fold-2 the results drop severely for identification of non-keyphrases as mentioned earlier, we observe similar behavior for all the algorithms (both supervised and unsupervised). As such, fold-2 warrants a more detailed study with both the algorithms to understand the reason for such behavior. However, we plan to visit this in our future work.

To summarise, our findings are:

- F1 is highest for WINGNUS compared to KEA, with former producing F1 0.58 and 0.33 across Folds 1 and 2, respectively. WINGNUS offers better performance owing to its usage of the document and phrase-level features besides TF-IDF.
- Across the folds, the performance is lower in fold-2, with everything identified as a keyphrase. We need more investigation to analyze the root cause. We face this issue again in section 4.1.2.
- Overall, both KEA and WINGNUS serve as promising baselines for keyphrase extraction from complex assignments.

4.1.2 *RQ 1.2: Unsupervised Keyphrase Extraction*

Unsupervised approaches for keyphrase extraction typically include graph-based ranking and frequency factoring. We comprehensively study an exhaustive set of approaches covering both these categories. Table 6 shows the results of KP-MINER, YAKE, and series of graph-based ranking approaches. The results vary across the methods ranging from 0.29 SingleRank to 0.64 on multipartite ranking. F1 is lower in SingleRank and TextRank approaches, meanwhile across the rest of the approaches we see the result to be well balanced. We can see that the results are higher than the supervised approaches.

To begin with, let's consider YAKE and KPMINER approaches. From Table 5 we can see that YAKE produces better results than KPMINER with 0.06 and 0.03 higher F1 than KPMINER approaches. This may be because KPMINER uses simple features such as word frequency factors while YAKE uses affinity graph and graph-based ranking. Further in line with results from section 4.1, we can see P of 0.5 and comparatively lower recall.

Meanwhile, the results of the graph-based ranking approach cover the entire spectrum, with multipartite rank producing the best results for both fold-1 and fold-2. Again compared to fold-2, fold-1 produces better results. Also across the

graph-based approaches, we can see that SingleRank performs the worst, despite both SingleRank and MultipartiteRank using clustering and graph-based modeling approach. We believe this is because of the candidate selection strategy where unlike SingleRank multipartite rank considers candidates with sequences of nouns and adjectives. We can draw similar intuition for results of SingleRank in fold-1. Overall, we can see that there is no clear favorite in ranking based approaches across both folds 1 and 2. Finally, we can also see that multipartite produces the best accuracy score across all the methods including supervised algorithms.

Coming to the results of fold-2 using graph-based approaches, we can see fixed P values and lower R and F1 values. An in-depth investigation highlights the same problem encountered earlier in case of supervised approaches, where again the unsupervised approaches lack identification of non-keyphrases (See Table 7).

Table 7—Results of Multipartite algorithm with class-wise separation.

		Fold 1			Fold 2		
		P	**R**	**F**	**P**	**R**	**F**
Classes	**Non-keyphrases**	0.50	0.54	0.52	0	0	0
	Keyphrases	0.70	0.95	0.81	1	0.94	0.96
Metrics	**Macro Avg**	0.68	0.64	0.64	0.5	0.47	0.48
	Weighed Average	0.68	0.67	0.67	1	0.94	0.96

To summarize we find

- MultipartiteRank produces best results across all the methods both supervised/other unsupervised approaches with F1 of 0.64 and 0.48 across folds 1 and 2.
- KPMINER and YAKE produce comparable performances, with YAKE producing superior F1 of 0.57 and 0.48 across the two folds.
- Similar to supervised keyphrase extraction, we see that in fold-2 unsupervised approaches again cannot identify any of the non-keyphrases. Hence deeper analysis of the fold-2 dataset is warranted. We plan to revisit this in the future.

4.2 RQ 2: Keyphrase Classification

Previously in section 1, we saw supervised and unsupervised approaches for extracting keyphrases. Here we will focus keyphrase classification (See Figure 2).

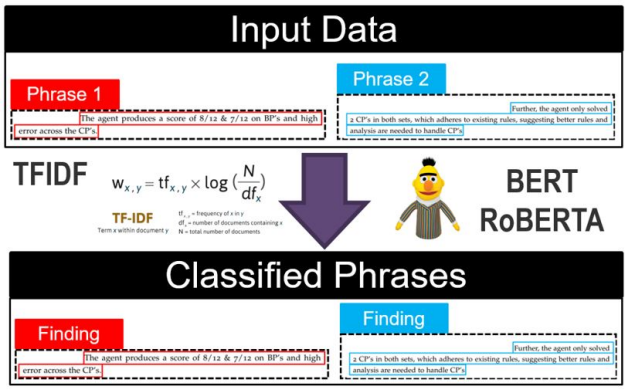

Figure 2—Keyphrase classification process.

4.2.1 *RQ 2.1: Phrase Classification with Traditional Approaches*

Our first research question for keyphrase classification will focus on analysis of performance through traditional supervised approaches. In this work, we restrict ourselves to SVM, due to its usage in benchmarks of short text classification (Zeng et al., 2018). To answer the RQ, Precision (P), Recall (R) and F1 metrics are reported on both as weighed average level and original macro-level (See Table 8). Weighted average in keyphrase classification ensures comparable results by taking data imbalance and its sensitivity on SVM. For weighted average prediction, we weigh the dataset predictions based on the ratio of the sample of a category in the dataset.

Also inline with section **??** we will analyze both the folds of the dataset. We grid search all the hyperparameters for SVM. Also, first look on the dataset suggest the classes of a task, finding and intuition typically encompass words in pairs (predominantly verbs), which can help in classification. Hence we also search for optimal n-gram parameters as part of the grid search resulting in n-gram value of (1,2).

Besides, we implement SVM using Gradient Descent Classifier with Hinge loss, rather than native SVM formation to reduce over fitting and randomness in initialization. This further ensures the overall method doesn't overfit with less data. To ensure fair comparison, we also create a baseline classifier that predicts class labels proportional to their distribution in the training set. We again show the results of the baseline classifier in Table 8.

Table 8 shows results with Precision (P), recall (R), and F1- Score (F1). We present

results for all the folds with an accuracy score. Comparing the different methods, we achieved the highest F1 results among traditional methods with SVM with TFIDF, followed by SVM with BOW and Baseline. SVM with TF-IDF shows high precision in Fold-1 and has a balanced P and R in Fold-2, this is unlike what we saw in keyphrase extraction, where the results were higher for fold-1 and in fold-2 we had issues of identifying non-keyphrases. For SVM with BOW, the results are very close to that of TF-IDF with a small difference of 0.2 in the F1 score. These findings are in line with other work reporting the usefulness of SVM in short text classification (Zeng et al., 2018).

Concerning results on each of the folds, SVM with BOW produces similar results in fold-1 but again dominated by SVM with TF-IDF in fold-2 by a large margin (0.06 F1). However, the overall performance is lower compared to that of typical text classification benchmarks, mostly because of the semi-formal nature of the text.

Table 8—Consolidated Results for Phrase Classification. Check results of bert and others

		Fold 1							Fold 2						
			Macro Average			Weighed Average				Macro Average			Weighed Average		
Feature Extractor	**Classifier**	**Accuracy**	**P**	**R**	**F**	**P**	**R**	**F**	**Accuracy**	**P**	**R**	**F**	**P**	**R**	**F**
-	**Baseline[3]**	0.4	0.29	0.3	0.29	0.39	0.4	0.38	0.34	0.25	0.24	0.24	0.36	0.34	0.35
BOW	**SVM**	0.54	0.46	0.41	0.42	0.53	0.54	0.52	0.54	0.38	0.37	0.37	0.51	0.54	0.52
TFIDF		0.57	0.44	0.40	0.41	0.53	0.56	0.54	0.59	0.45	0.41	0.42	0.64	0.59	0.61
BERT		0.56	0.47	0.40	0.41	0.57	0.56	0.52	0.11	0.17	0.29	0.10	0.22	0.11	0.07
ROBERTA		0.28	0.16	0.29	0.16	0.27	0.28	0.26	0.11	0.05	0.23	0.07	0.02	0.11	0.03
DISTILLBERT		0.60	0.46	0.39	0.39	0.58	0.50	0.56	0.53	0.41	0.39	0.37	0.55	0.53	0.51
XLM		0.63	0.53	0.48	0.49	0.61	0.63	0.62	0.55	0.40	0.38	0.39	0.53	0.55	0.53
XLNET		0.55	0.45	0.42	0.43	0.55	0.55	0.55	0.57	0.52	0.40	0.41	0.56	0.57	0.54
XLMROBERTA		0.31	0.07	0.25	0.11	0.09	0.31	0.15	0.48	0.12	0.25	0.16	0.23	0.48	0.32
ALBERT		0.48	0.40	0.40	0.37	0.52	0.48	0.47	0.59	0.55	0.43	0.44	0.58	0.59	0.56
BERT	**XGB**	0.56	0.49	0.37	0.38	0.54	0.56	0.52	0.55	0.44	0.35	0.35	0.52	0.55	0.50
ROBERTA		0.46	0.14	0.27	0.17	0.24	0.46	0.32	0.18	0.20	0.24	0.23	0.32	0.18	0.16
DISTILLBERT		0.59	0.38	0.35	0.34	0.59	0.54	0.54	0.63	0.47	0.41	0.41	0.58	0.63	0.58
XLM		0.58	0.42	0.34	0.33	0.54	0.58	0.51	0.61	0.40	0.41	0.40	0.56	0.61	0.58
XLNET		0.55	0.39	0.33	0.32	0.51	0.55	0.49	0.58	0.42	0.35	0.34	0.53	0.58	0.51
XLMROBERTA		0.55	0.41	0.33	0.32	0.51	0.55	0.48	0.53	0.37	0.32	0.31	0.48	0.53	0.47
ALBERT		0.54	0.43	0.35	0.35	0.51	0.54	0.49	0.55	0.61	0.36	0.36	0.58	0.55	0.50

Our overall analysis of the traditional method shows that it performs fairly well on the complex assignment phrases, however, we believe owing to sparsity in vocabulary and size of dataset the performance is not so high like usual text classification benchmarks. Hence, with larger datasets and different domains, the performance is expected to be consistent. To summarise, our findings are:

- F1 is highest with SVM with TF-IDF, followed by SVM with BOW.

- SVM with TF-IDF outperforms baseline classifier[2] by a large margin (>= 16 points in F1).
- Among both the folds, fold-2 produces better results then fold-1. This is unlike what we saw in keyphrase extraction (See Table 5)
- Existing hypothesis of SVM with TF-IDF serves as a competitive baseline for short text classification still holds.

4.2.2 *RQ 2.2: Phrase Classification with Language Models*

While SVM with TF-IDF produces significant results, the error and ambiguity are high. This is visible from results in which F1 <0.59. Recently, language models offer a significant advantage where they help to build on existing knowledge gathered from large datasets. Since much of the dataset used in this work is derived from semi-formal reports, we hypothesize such pretrained networks would help in results. Further, since the dataset is very small, the training language model would lead to significant overfitting. Hence to ease this we propose to use the pretrained language models only as feature extractors with SVM and XGBoost (Chen and Guestrin, 2016) classifier. We use XGBoost because of its previous success with language models.

Table 8 shows the results on the multiple language models including *BERT, RoBERTa, DistillBert, Albert, XLM, XLNET, XLMROBERTa* transformer models used as feature extractors. The features are extracted from the second encoder of the transformer[3]. To simplify the analysis we shall consider the only performance of BERT and RoBERTa, due to their widespread usage and generalized results. However similar interpretations could be drawn for others as well.

Firstly, comparing results of all the language models, we can see that BERT offers a balanced F1 of 0.45 an 0.48 across the folds. We can also see that while in fold-1 XLM produces the highest results, in fold-2 the results are significantly lower. Inline to RQ 2.1 we again see that results of fold-2 is lesser than fold-1. Besides this, we can also see that results are lowest for RoBERTa in phase-2. We tested this result of RoBERTa multiple times and found neither programming nor correlation level issues.To further check the impact of training, we froze all the encoders and trained only the final layers, yet the results were significantly lower than the baseline. Also, we can see from Table 8 results using XGB, the

2 We use scikit-learn DummyClassifier with *stratified* strategy.

3 This was based on our experimentation with different encoders

performance is pretty much similar across all models and unlike SVM the results of P and R are well balanced.

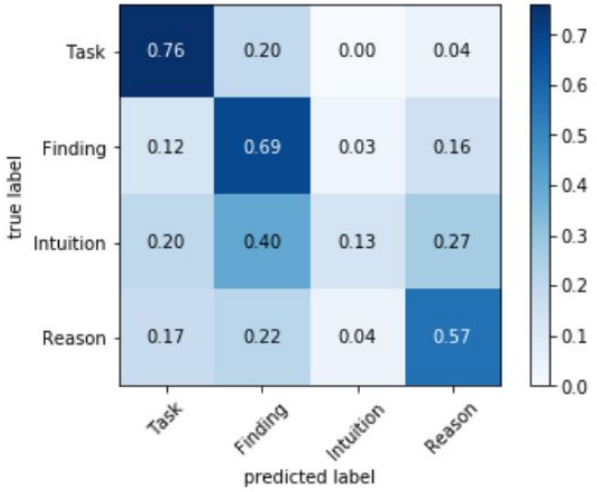

Figure 3—Confusion Matrix of the best performing BERT-SVM model

Regarding RoBERTa's performance on SVM in Fold-2 we suspect following, RoBERTa originally uses a special training procedure with dataset larger than BERT, while we expect the pretrained representations to be well distributed, however with training on large sets the representations form tight clusters results in need of greater weighting of data points. However, more experiments are needed for validating the same. The experimental setup of the RoBERTa tokenizer needs a revisit.

Finally, coming to errors, we can see that the classes of *Task, Finding, and Intuition* are highly ambiguous. Figure 3 shows the confusion matrix, where we can see much of the error is concentrated among the three classes. This is in line with Cohen's kappa (κ) of 0.77 earlier mentioned in the dataset.

To summarise, our findings are:

- Performance on BERT is higher than that of RoBERTa and best-performing SVM across folds. Meanwhile, XGB produces similar results across the different language model representations.
- XLM while produces the highest results in fold-1, the results are lower with fold-2.
- Despite the sensitivity of imbalanced dataset, BERT with SVM produces F1 of

0.45 on fold-1, however for RoBERTa we see otherwise. We believe this low result of RoBERTa is because of the inherent architecture. However, such a detail requires more in depth architectural exploration

- With XGB, both BERT and RoBERTa perform similarly on Fold-1 and Fold-2.
- However, there are still significant differences in the classification of phrases and human annotation. This shows that capturing context is a hard issue, especially with phrases of complex assignments and current annotation schemes. Thus we made need more granular annotation schemes, which we plan to revisit in our future works.

4.2.3 *RQ 2.3: Reliability study of traditional and language models*

Previously in Table 8, we saw the results of both traditional and language models. We could see that among traditional models the performance is closer, with TF-IDF producing higher performance on fold-2. Meanwhile, BERT language models perform significantly higher than the rest. So to further test the performance in-depth, we focused on the reliability aspect of the model where we define reliability *as the models' ability to focus on the required information for classification true from a human's perspective* i.e. the model should look on some parts of the text which a human would see during annotation. To do this, we use Local Interpretable Model-agnostic Explanations (LIME) (Ribeiro, Singh, and Guestrin, 2016).

Table 9—Results of Interpretability on Traditional and BERT.

	Model -1	**Model-2**	**Precision**
1	BERT (SVM)	SVM (TF-IDF)	0.82
2	BERT (SVM)	SVM (BOW)	0.81

Table 10—Example of Interpretability of Traditional (right) and Language models (left) with similar word contributions.

TF-IDF (SVM)		**BERT (SVM)**	
Feature	**Contribution**	**Feature**	**Contribution**
which	+0.350	agent	+0.13
still can	+0.190	this	+0.09
problems involving	+0.171	still	+0.07
have shading	+0.170	which	+0.04
handle problems	+0.145	handle	+0.04
involving shapes	+0.137	problems	+0.03
shapes which	+0.119	have	+0.01
this agent	+0.079	can	+0.01
which have	+0.065	involving	+0.01

Table 11—Example of Interpretability of Traditional (left) and Language models (right) with dissimilar word contributions.

TF-IDF (SVM)		BERT (SVM)	
Feature	**Contribution**	**Feature**	**Contribution**
the	+1.154	a	+0.14
performance	+0.264	the	+0.13
agent	+0.220	the	+0.08
performance of	+0.205	increased	-0.15
a little	+0.074	has	-0.04
increased	+0.071	little	-0.03
increased a	+0.055	of	-0.02
a	-0.051	performance	-0.02
the agent	-0.089	agent	-0.01

We use the **precision** as the evaluation metric, where we compute precision as the fraction of words that are contributing to both traditional and language models performance[4]. The results so obtained are as shown in Table 9. From Table 9, we find that in both the cases the models look around 80% of times on similar word lists. Further, we also see mixed interpretability results.

We also manually analyze the results to find interesting characteristics. Table 10 shows results where both traditional and language models look on similar words, while Table 11 shows the BERT model to focus on words that make little sense in terms of contributions. [5]

From Table 10 we can see that both traditional and BERT look for similar words for predicting the correct results, however, from Table 11 we can see that sometimes traditional model has better interpretability. We believe this is an issue of using the BERT model as a feature extractor, rather than training it from scratch.

Overall, traditional models focus on the right set of words, in most cases. Hence we conclude that traditional models better interpret the cues related to class from the educational text. However, we argue that this needs to be revisited in more depth with an exhaustive comparison on a relationship with features used in the traditional model and language model training process.

To summarise, our findings are:

- Both traditional and language models look on similar word cues around 80%

4 We calculated precision by manually counting the words.

5 We don't use XGB and RoBERTa models as they produce lower performance.

of times, despite dissimilar contributions.

- Traditional models are more reliable in terms of interpretability, this has more to do with the usage of simple n-gram features unlike language models, complex encoders with no training.

4.3 RQ 3: Keyphrase-Rubric Relationship Extraction

In this section, we will focus on the analysis of unsupervised and supervised approaches for keyphrase-rubric relationship extraction. In this work we formulate keyphrase-rubric relationship extraction as a classification problem (See Figure 4).

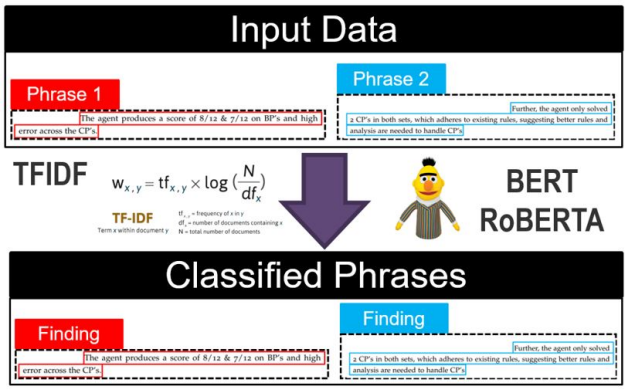

Figure 4—Keyphrase-rubric relationship extraction.

4.3.1 *RQ 3.1: Unsupervised Phrase-Rubric Relationship Classification*

Originally, clustering makes use of labeled phrases to capture the context of phrases and adapts unlabeled phrases to its centroids. Thus the category of each phrase cluster is labeled by the majority labels of texts in it. Hence here clustering is used to generate the classification.

Our first research question in keyphrase-rubric relationship classification focuses on the analysis of the effectiveness of clustering approaches for phrases-rubric relationship classification. To answer this, Precision (P), Recall (R) and F1 metrics are reported on both as weighed average level and original macro-level (See Tables 12-14). Further, we present additional metrics such as *cluster purity (HOMO), Rand Index (RI), Cluster Silhouette (SIL)* to understand the nature of clusters. The results obtained for both the folds across multiple unsupervised methods are as shown in Tables 12-14.

Table 12—Consolidated results of K-means clustering

				Macro Average			Weighed Average			Clustering Metrics			
Split	**Initialization**	**Feature Extractor**	**Accuracy**	**P**	**R**	**F1**	**P**	**R**	**F1**	**HOMO**	**RI**	**AMI**	**SIL**
Fold 1	**K-Means++**	**TF-IDF**	0.3697	0.3502	0.34701	0.31755	0.47341	0.36957	0.39493	0.019	-0.005	0.005	0.002
		BERT	0.38043	0.32827	0.33661	0.28713	0.5001	0.38043	0.40216	0.027	0.028	0.013	0.226
	Random	**TF-IDF**	0.27174	0.21436	0.19521	0.19884	0.31124	0.27174	0.28683	0.027	-0.006	0.012	0.002
		BERT	0.17391	0.37754	0.30881	0.16963	0.59236	0.17391	0.1274	0.023	0.011	0.009	0.205
	PCA	**TF-IDF**	0.20652	0.20872	0.20747	0.17234	0.3022	0.20652	0.20179	0.019	0.011	0.005	0.003
		BERT	0.11413	0.11156	0.20486	0.09994	0.15941	0.11413	0.1024	0.024	0.026	0.01	0.234
Fold 2	**K-Means++**	**TF-IDF**	0.36413	0.32046	0.33986	0.31261	0.43442	0.36413	0.3811	0.042	0.027	0.026	0.003
		BERT	0.35326	0.32407	0.29857	0.27175	0.39	0.35326	0.34836	0.023	0.03	0.008	0.247
	Random	**TF-IDF**	0.2663	0.28733	0.26625	0.24597	0.37864	0.2663	0.29987	0.035	0.012	0.02	0.003
		BERT	0.33696	0.2485	0.25248	0.23594	0.38003	0.33696	0.34856	0.034	0.035	0.02	0.232
	PCA	**TF-IDF**	0.2663	0.19071	0.1972	0.1871	0.28021	0.2663	0.26773	0.044	0.009	0.029	0.004
		BERT	0.20652	0.1636	0.20035	0.15224	0.254147	0.20652	0.21426	0.023	0.007	0.008	0.969

From Table 12, comparing all the results we can see that K-Means produces the best result, followed by Spectral clustering and then Agglomerative. More specifically, while K-Means produces the best result of 0.32 F1-Score Agglomerative clustering produces high precision and recall of 0.48 the macro F1 is lower than 0.2. Further, the behavior is consistent across both the folds. Also, we see high precision in Fold-2 and a balanced P and R in Fold-1. Further for Fold-1, we see high accuracy with Spectral clustering with a accuracy score of 0.45 with F1 of Agglomerative & Spectral approaches very far from those of K-Means. Coming to feature extractors we can see that while TF-IDF produces the top results across the majority of the experiments.

Regarding K-Means, we tested with three different initialization namely *K-Means++*, *Random & PCA* based initialization. We hypothesize that with multiple random tests, the random initialization would produce results similar to K-Means++ and principal components with maximum variance indeed produces useful results. Results so obtained across both the folds are again shown in Table 12. We can see that K-means++ produces results higher than the Random and PCA based initialization. Additionally, we can also see that random initialization, produces results very close to that of K-Means++. Meanwhile, the maximal variance components are indeed weak and biased on the identification of the phrase level classes.

In the case of Agglomerative Clustering, we used ward distance for linkage and multiple affinity measure namely cosine, euclidean and city block distance metrics. Table 13 shows the consolidated results so obtained. From Table 13 we can

Table 13—Consolidated results of Agglomerative Clustering

				Macro Average			Weighed Average			Clustering Metrics			
Split	**Affinity**	**Feature Extractor**	**Accuracy**	**P**	**R**	**F1**	**P**	**R**	**F1**	**HOMO**	**RI**	**AMI**	**SIL**
Fold 1	**Cosine**	**TF-IDF**	0.26087	0.22599	0.22488	0.19884	0.2601	0.26087	0.2254	0.041	0.093	0.02	0
		BERT	0.25	0.18224	0.21951	0.15287	0.25389	0.25	0.1738	0.019	0.008	-0.006	0.002
	Euclidean	**TF-IDF**	0.2663	0.19418	0.20472	0.18799	0.28933	0.2663	0.26072	0.034	0.013	0.011	0.002
		BERT	0.28261	0.20072	0.19133	0.18597	0.30854	0.28261	0.28316	0.035	0.015	0.013	0.004
	Cityblock	**TF-IDF**	0.33152	0.49767	0.26597	0.15312	0.55866	0.33152	0.18576	0.021	0.03	0.004	-0.023
		BERT	0.30435	0.57714	0.25472	0.14321	0.70804	0.30435	0.16346	0.026	0.021	0.011	-0.023
Fold 2	**Cosine**	**TF-IDF**	0.26087	0.1803	0.24574	0.16725	0.26031	0.26087	0.1971	0.014	0.034	-0.008	0.222
		BERT	0.29348	0.29348	0.24579	0.13554	0.26948	0.29348	0.15777	0.016	0.003	-0.008	0.204
	Euclidean	**TF-IDF**	0.33152	0.20736	0.23688	0.18022	0.35104	0.33152	0.2753	0.019	0.013	-0.002	0.169
		BERT	0.23913	0.48054	0.48054	0.18138	0.4694	0.23913	0.16654	0.022	0.011	0.002	0.223
	Cityblock	**TF-IDF**	0.42935	0.26665	0.28221	0.27278	0.41755	0.42935	0.42126	0.033	0.032	0.014	0.182
		BERT	0.21196	0.26006	0.2676	0.20384	0.3732	0.21196	0.23477	0.031	-0.008	0.009	0.353

see that Euclidean Distance produces the best results (similar to original wards computation). Cosine distance produces similar results like that of Euclidean under TF-IDF feature extractor but in rest of the cases, euclidean outperforms. City block distance produces the worst performance. Overall agglomerative clustering produces best results of 0.18 F1 using Euclidean affinity on BERT features.

Table 14—Consolidated Results of Spectral Clustering

				Macro Average			Weighed Average			Clustering Metrics			
Split	**Sampling Strategy**	**Feature Extractor**	**Accuracy**	**P**	**R**	**F1**	**P**	**R**	**F1**	**HOMO**	**RI**	**AMI**	**SIL**
Fold 1	**K-Means**	**TF-IDF**	0.41304	0.35258	0.27234	0.2713	0.41887	0.41304	0.39264	0.036	0.002	0.016	0.01
		BERT	0.45109	0.34087	0.30922	0.28039	0.42579	0.45109	0.40373	0.038	-0.034	0.02	0.01
	Discrete	**TF-IDF**	0.21196	0.36059	0.2215	0.16441	0.60518	0.21196	0.18973	0.04	0	0.019	0.008
		BERT	0.2663	0.181	0.23257	0.15539	0.25327	0.2663	0.17572	0.016	-0.023	-0.007	0.009
Fold 2	**K-Means**	**TF-IDF**	0.28804	0.29388	0.23026	0.16994	0.50693	0.28804	0.23666	0.036	0.002	0.016	0.01
		BERT	0.29348	0.075	0.24107	0.11441	0.0913	0.29348	0.13928	0.024	0.007	0.012	0.393
	Discrete	**TF-IDF**	0.28261	0.31993	0.31344	0.24516	0.46287	0.28261	0.28912	0.04	0	0.019	0.008
		BERT	0.45652	0.20798	0.24683	0.20448	0.34264	0.45652	0.36264	0.026	-0.02	0.009	0.257

Finally, coming to spectral clustering (See Table 14) the behavior is similar to that of K-Means, most because internally spectral clustering, in turn, uses K-means. The best performing spectral clustering approach gives an F1-Score of 0.28 with BERT features similar results could be seen even with TF-IDF feature extraction. K-Means sample selection produces an accuracy score of 0.451 F1 while discretized hierarchical clustering produces the best result of 0.16 F1.

The results of cluster analysis are as shown in the cluster metrics including homogeneity, rand index, etc. We can see that across all the metrics the clusters

are not so good. Much of the cluster doesn't satisfy homogeneity indicating that clusters don't contain only data points which are members of a single class. This is also visible through low P, R and F1 measures. Meanwhile, we can further see the average mutual information is very low indicating the clusters are completely mixed. Additionally, we can see that the silhouette is very low for all the experiments this shows that clusters are extremely overlapping. To summarise, our findings are:

- F1 is highest for K-Means with K-Means++ initialization. The findings also hold for spectral clustering.
- Tf-IDF representation produces the highest results majority of the approaches compared to BERT.
- Agglomerative clustering produces the least results. The same is true with spectral clustering.
- Compared to both the Folds, fold-1 shows better performance than fold-2.
- From cluster analysis, we can see that clusters are overlapping with near-zero silhouette score and homogeneity.

4.3.2 *RQ 3.2: Supervised & Topic Modeling Approaches for Phrase-Rubric Relationship Classification*

Previously in section 4.3.1 we saw a performance of unsupervised models on phrase-rubric relationship extraction. We could see that among the results K-Means dominated the performance visible by F1 in K-Means with K-Means++ initialization. However typically textual content is expected to contain multiple topics which in our case is *Task, Reasoning, Intuition, Findings*. Hence in this research question, we will exploit topic modeling to see if these methods are indeed useful for the process of relationship classification. Generally, a topic model is a type of statistical model for discovering the abstract "topics" that occur in a collection of documents. In this work, we use latent Dirichlet's allocation which dominated NLP tasks involving topic modeling. More specifically we use version specified by the works of Blei, Ng, and Jordan, 2003. We further compare performance to supervised classification models involving SVM with TF-IDF and BERT. Precision (P), Recall (R) and F1 metrics are reported again on both as weighed average level and original macro-level. Tables 15 and 16 shows results of LDA and supervised approaches.

Firstly coming to K-means , we used K-Means with TF-IDF features coupled

Table 15—Consolidates results using Latent Dirichlet Allocation with Clustering Algorithms

				Macro Average			Weighed Average			Clustering Metrics			
Split	**Approach**	**Parameter**	**Accuracy**	**P**	**R**	**F**	**P**	**R**	**F**	**HOMO**	**RI**	**AMI**	**SIL**
Fold 1	**K-Means**	**K-Means++**	0.43	0.27	0.26	0.24	0.38	0.43	0.36	0.03	0.04	0.01	0.98
		Random	0.43	0.27	0.26	0.24	0.38	0.43	0.36	0.03	0.04	0.01	0.98
		PCA	0.14	0.17	0.20	0.10	0.28	0.14	0.11	0.03	0.04	0.01	0.98
	Agglomerative	**Cosine**	0.30	0.20	0.25	0.17	0.27	0.30	0.19	0.04	-0.01	0.02	0.48
		Euclidean	0.33	0.33	0.26	0.18	0.51	0.33	0.25	0.04	-0.01	0.02	0.48
		Cityblock	0.32	0.31	0.25	0.17	0.45	0.32	0.23	0.03	-0.01	0.00	0.48
	Spectral	**K-Means**	0.26	0.26	0.22	0.17	0.35	0.26	0.22	0.02	0.01	-0.01	0.47
		Discrete	0.35	0.27	0.28	0.26	0.38	0.35	0.35	0.01	0.01	-0.01	0.46
Fold 2	**K-Means**	**K-Means++**	0.21	0.16	0.20	0.15	0.25	0.21	0.21	0.02	0.01	0.01	0.97
		Random	0.35	0.29	0.32	0.29	0.38	0.35	0.36	0.02	0.01	0.01	0.97
		PCA	0.27	0.28	0.29	0.25	0.38	0.27	0.27	0.02	0.01	0.01	0.97
	Agglomerative	**Cosine**	0.20	0.24	0.17	0.16	0.36	0.20	0.21	0.04	0.01	0.02	0.39
		Euclidean	0.28	0.26	0.29	0.24	0.34	0.28	0.29	0.05	0.02	0.02	0.35
		Cityblock	0.21	0.26	0.27	0.20	0.37	0.21	0.23	0.03	-0.01	0.01	0.35
	Spectral	**K-Means**	0.37	0.30	0.33	0.30	0.40	0.37	0.38	0.05	0.01	0.03	0.41
		Discrete	0.27	0.25	0.29	0.25	0.31	0.27	0.28	0.05	0.01	0.03	0.42

with Latent Dirichlet Allocation. Similar to results from previous section 4.3.1 we can see that K-Means with K-Means++ and Random initialization performs similarly. Also introducing LDA improves results for of randomly initialized k-means. However, the results are significantly lower than the original implementation with only TF-IDF and K-Means++. Overall LDA contributes to improving accuracy by 0.06 with the drop in F1 of 0.08 in fold-1 and 0.05 with drop in F1 by 0.15 in fold-2. In general, we can see an improvement in accuracy score across all the initialization post introduction of LDA. For this

Table 16—Results of supervised phrase-rubric relationship classification

		Fold 1							Fold 2						
		Macro Avg			Weighed Average				Macro Avg			Weighed Average			
Algorithm	**Accuracy**	**P**	**R**	**F**	**P**	**R**	**F**	**Accuracy**	**P**	**R**	**F**	**P**	**R**	**F**
SVM + BOW	0.54	0.46	0.41	0.42	0.53	0.54	0.52	0.54	0.38	0.37	0.37	0.51	0.54	0.52
SVM + TFIDF	0.56	0.44	0.4	0.41	0.53	0.56	0.54	0.592	0.45	0.41	0.42	0.64	0.59	0.61
BERT (SVM)	0.56	0.47	0.4	0.41	0.57	0.56	0.52	0.11	0.17	0.29	0.1	0.22	0.11	0.07
BERT (XGB)	0.56	0.49	0.37	0.38	0.54	0.56	0.52	0.55	0.44	0.35	0.35	0.52	0.55	0.5

In the case of agglomerative clustering, the results are very similar to K-Means, where the accuracy score improved by 0.07 with similar F1-Score as shown in section 4.3.1 for agglomerative clustering. However, we can also see that the city block affinity shows higher sensitivity compared to cosine affinity.

Coming to spectral clustering we see that results are again similar to K-Means, however, there is surprising finding where the results for spectral clustering with K-Means++ sampling are lower than that of Discretized Sample selection process. The difference in results is 0.1 in accuracy and 0.08 F1-Score. Overall with LDA discretized sampling shows the best results for Spectral clustering. Similar observations can be seen in fold-2 of the dataset except spectral clustering with K-Means sample selection producing higher results than that of fold-1.

Overall with LDA, we see two major benefits firstly we can see an improvement in accuracy score. Otherwise overall results are significantly lower. This is because, in this work we used total as four topics during LDA computation however based on results we can conjecture that the model contains topics more than four. The supervised results are as shown in Table 16. We can see best performing supervised approach outperforms the best unsupervised approaches by an average of 0.15 F1 across the folds.

Finally, coming to cluster analysis we can see that the cluster overall is significantly reduced as evident by the silhouette score of 0.9 with K-Means, 0.3 with Hierarchical and 0.4 with spectral clustering. Yet the cluster uniqueness is fairly limited as evident with homogeneity metrics and rand index where the values are close to zero.

To summarise, our findings are:

- Firstly, LDA improves overall accuracy across all the unsupervised algorithms. Further the results are similar to that of section 4.3. The maximum F1 in fold-1 is 0.25 and in fold-2 is 0.29 with both using spectral clustering.
- Further, spectral clustering shows a unique behavior where the results flip between discrete and K-Means sample selection. However, the overall performance is significantly lower compared to section 4.3.
- With LDA we can see the clusters don't overlap as evident by silhouette score. However individual clusters are noisy with data from multiple classes.
- Supervised approaches inherently produce higher results compared to that of unsupervised approaches.

5 CONCLUSION

In this work, we focused on keyphrase extraction, classification and keyphrase-rubric relationship extraction from complex assignments. Firstly, by developing

a new corpus and annotation scheme, we demonstrated that datasets in complex assignments are harder to create: in terms of annotation and size; the balance of classes; the proportion of words; and how often tokens are repeated. The dataset so created is imbalanced with the domination of *Task* and *Finding* classes. This is similar, to the dataset of twitter and web corpora which are traditionally noisy. Also we presented details on how the same annotated dataset is used across all the three tasks along with a brief explanation on various algorithms that was used for the experiments.

Firstly we began our study on keyphrase extraction by evaluating supervised approaches of KEA and WINGNUS where WINGNUS offers better performance owing to its usage of the document and phrase-level features besides TF-IDF. Also we saw that the performance is lower in fold-2, with everything identified as a keyphrase. However both KEA and WINGNUS serve as promising baselines for keyphrase extraction from complex assignments. Meanwhile surprisingly unsupervised approaches offer better results. In unsupervised approaches we tested KPMINER, YAKE and other graph based approaches. We find that MultipartiteRank produces best results across all the methods both supervised/other unsupervised approaches with F1 of 0.64 and 0.48 across folds 1 and 2, meanwhile KPMINER and YAKE produce comparable performances, with YAKE producing superior F1 of 0.57 and 0.48 across the two folds. Similar to supervised keyphrase extraction, we see that in fold-2 unsupervised approaches again cannot identify any of the non-keyphrases.

Second in phrase classification, we saw that traditional approaches like SVM with TF-IDF achieves consistently the highest performance across the folds, and this is the best traditional approach to generalizing from training to testing data. Also BERT trained with SVM as good predictor of F1 in phrase classification, in which the BERT model was used as a feature extractor for each folds of the test corpus. This supported our hypothesis that language models without any training would still be useful owing to their training on large datasets, and training on a small dataset will negatively be correlated with F1. Also our experiments confirmed the issue of reliability where traditional models are more reliable in terms of interpretability while language models without training are sensitive in nature.

Finally, phrase-rubric relationship classification investigated the capacity of both clustering and topic modeling for the task of classification of phrases-rubrics

from complex assignments. we studied three clustering approaches involving K-Means, Agglomerative and Spectral clustering. We find that K-Means dominates the results with both TF-IDF and BERT against the rest followed by spectral clustering with K-Means based sampling. Agglomerative clustering performs the worst across the three. However the best traditional methods results are significantly lower than that of supervised approaches which we can see by comparing Tables 12,13,14 and 16. Regarding topic modeling we find that with LDA the accuracy is significantly higher compared to without LDA especially across the three models. The LDA model improves accuracy however the net improvement is fairly limited. The best F1 of LDA is lower than the best F1 without LDA. Further, we find that K-Means++ dominating the performance throughout. We argue the net improvement of F1 is lower in LDA because of the noisy data and number of topics so present in the dataset. Also our experiments confirmed that supervised models are still reliable for phrase-rubric relationship classification.

Also for we see that for all the three RQ's, by studying performance with the weighted average, it becomes clear that there is also a big difference in performance on corpora with the balanced and imbalanced dataset. This indicates that annotating more training examples for diverse classes would likely lead to a dramatic increase in F1 which in turn is expected to improve performance across all the explored RQ's.

6 REFERENCES

[1] Akshay, A., Jagadish, V., Shane, L., and Andreas, P. (2015). "YouEDU: Addressing Confusion in MOOC Discussion Forums by Recommending Instructional Video Clips". In: *Proceedings of the Eighth International Conference on Educational Data Mining (EDM 2015)*. Madrid, Spain.

[2] Ardaiz-Villanueva, Oscar, Chacón, Xabier Nicuesa, Artazcoz, Oscar Brene, Acedo Lizarraga, María Luisa Sanz de, and Acedo Baquedano, María Teresa Sanz de (2011). "Evaluation of computer tools for idea generation and team formation in project-based learning". In: *Computers Education* 56, pp. 700–711.

[3] Badawy, Mohammed, Mahmood, Mahmood A., El-Aziz, A. A. Abd, and Hefny, Hesham Ahmed (2018). "A Text Mining Approach for Automatic Selection of Academic Course Topics based on Course Specifications". In:

2018 14th International Computer Engineering Conference (ICENCO), pp. 162–167.

[4] Bekele, Rahel (2006). "Computer-Assisted Learner Group Formation Based on Personality Traits". In: *Dissertationsschrift zur Erlangung des Grades eines Doktors der Naturwissenschaften am Fachbereich Informatik der Universität Hamburg*.

[5] El-Beltagy, Samhaa R. and Rafea, Ahmed A. (2010). "KP-Miner: Participation in SemEval-2". In: *SemEval@ACL*.

[6] Bennani-Smires, Kamil, Musat, Claudiu, Hossmann, Andreea, Baeriswyl, Michael, and Jaggi, Martin (2018). "Simple Unsupervised Keyphrase Extraction using Sentence Embeddings". In: *SIGNLL Conference on Computational Natural Language Learning*.

[7] Bhaskar, Pinaki, Nongmeikapam, Kishorjit, and Bandyopadhyay, Sivaji (2012). "Keyphrase Extraction in Scientific Articles: A Supervised Approach". In: *Proceedings of Computational Linguistics (COLING)*.

[8] Blei, David M., Ng, Andrew Y., and Jordan, Michael I. (2003). "Latent Dirichlet Allocation". In: *J. Mach. Learn. Res.* 3, pp. 993–1022.

[9] Boudin, Florian (2018). "Unsupervised Keyphrase Extraction with Multipartite Graphs". In: *NAACL-HLT*.

[10] Bougouin, Adrien, Boudin, Florian, and Daille, Béatrice (2013). "TopicRank: Graph-Based Topic Ranking for Keyphrase Extraction". In: *IJCNLP*.

[11] Bulgarov, Florin Adrian (2018). "Toward Supporting Fine-Grained, Structured, Meaningful and Engaging Feedback in Educational Applications". In: *Dissertation Prepared for the Degree of DOCTOR OF PHILOSOPHY, UNIVERSITY OF NORTH TEXAS*.

[12] Bulgarov, Florin Adrian and Nielsen, Rodney (2017). "Minimal Meaningful Propositions Alignment in Student Response Comparisons". In: *Artificial Intelligence in Education*.

[13] Campos, Ricardo, Mangaravite, Vítor, Pasquali, Arian, Jorge, Alípio Mário, Nunes, Celia, and Jatowt, Adam (2020). "YAKE! Keyword extraction from single documents using multiple local features". In: *Inf. Sci.* 509, pp. 257–289.

[14] Chen, Tianqi and Guestrin, Carlos (2016). "XGBoost: A Scalable Tree Boosting System". In: *ArXiv* abs/1603.02754.

[15] Cho, Kwangsu (2008). "Machine Classification of Peer Comments in Physics". In: *First International Conference on Educational Data Mining*.

[16] Chowdhury, Jishnu Ray, Caragea, Cornelia, and Caragea, Doina (2019). "Keyphrase Extraction from Disaster-related Tweets". In: *ArXiv* abs/1910.07897.

[17] Devlin, J., Chang, M., Lee, K., and Toutanova, K. (2019). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

[18] Fahmi, Fitra Zul and Nurjanah, Dade (2018). "Group Formation Using Multi Objectives Ant Colony System for Collaborative Learning". In: *2018 5th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)*, pp. 696–702.

[19] Florescu, Corina and Caragea, Cornelia (2017). "PositionRank: An Unsupervised Approach to Keyphrase Extraction from Scholarly Documents". In: *ACL*.

[20] Flower, Linda and Hayes, J. R. (1981). "A Cognitive Process Theory of Writing." In: *College Composition and Communication Vol. 32, No. 4, pp. 365-387*.

[21] Geigle, Chase, Zhai, ChengXiang, and Ferguson, Duncan C. (2016). "An Exploration of Automated Grading of Complex Assignments". In: *Proceedings of the Third International Conference of on Learning at Scale (L@S)*. Edinburgh, United Kingdom.

[22] Godea, Andreea, Bulgarov, Florin Adrian, and Nielsen, Rodney (2016). "Automatic Generation and Classification of Minimal Meaningful Propositions in Educational Systems". In: *Proceedings of Computational Lingusitics (COLING)*.

[23] Gollapalli, Sujatha Das, Li, Xiaoli, and Yang, Peng (2017). "Incorporating Expert Knowledge into Keyphrase Extraction". In: *The Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)*.

[24] Graf, Sabine and Bekele, Rahel (2006). "Forming Heterogeneous Groups for Intelligent Collaborative Learning Systems with Ant Colony Optimization". In: *Intelligent Tutoring Systems*.

[25] Hattie, John and Timperley, Helen (2007). "The Power of Feedback". In: *Review of Educational Research* 77.1, pp. 81–112. DOI: 10.3102/003465430298487. eprint: https://doi.org/10.3102/003465430298487. URL: https://doi.org/10.3102/003465430298487.

[26] He, Yu, Hu, Xinying, and Sun, Guangzhong (2019). "A cognitive diagnosis framework based on peer assessment". In: *ACM Turing Celebration Conference*.

[27] Jaime, A. and Kyle, S. (2015). "Predicting Speech Acts in MOOC Forum Posts". In: *International AAAI Conference on Web and Social Media*. Oxford, England. URL: https://www.aaai.org/ocs/index.php/ICWSM/ICWSM15/paper/view/10526.

[28] Joyner, David A. (2017). "Scaling Expert Feedback: Two Case Studies". In: *Proceedings of the Fourth International Conference of on Learning at Scale (L@S)*. Cambridge, MA.

[29] Kulkarni, Chinmay, Bernstein, Michael S., and Klemmer, Scott R. (2015). "PeerStudio: Rapid Peer Feedback Emphasizes Revision and Improves Performance". In: *Proceedings of the Second International Conference of on Learning at Scale (L@S)*. Vancouver, British Columbia.

[30] Lample, Guillaume and Conneau, Alexis (2019). "Cross-lingual Language Model Pretraining". In: *NeurIPS*.

[31] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). "RoBERTa: A Robustly Optimized BERT Pretraining Approach". In: *ArXiv* abs/1907.11692.

[32] Liu, Z., Liu, S., Liu, L., Sun, J., Peng, X., and Wang, T. (2016). "Sentiment recognition of online course reviews using multi-swarm optimization-based selected features". In: *Neurocomputing* 185, pp. 11–20.

[33] Liu, Zhiyuan, Li, Peng, Zheng, Yabin, and Sun, Maosong (2009). "Clustering to Find Exemplar Terms for Keyphrase Extraction". In: *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*.

[34] Lynda, Haddadi, Farida, Bouarab-Dahmani, Tassadit, Berkane, and Samia, Lazib (2017). "Peer assessment in MOOCs based on learners' profiles clustering". In: *2017 8th International Conference on Information Technology (ICIT)*, pp. 532–536.

[35] Mihalcea, Rada and Tarau, Paul (2004). "TextRank: Bringing Order into Text". In: *EMNLP*.

[36] Nguyen, Huy, Xiong, Wenting, and Litman, Diane J. (2016). "Instant Feedback for Increasing the Presence of Solutions in Peer Reviews". In: *Proceedings of the 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

[37] Nguyen, Thuy Dung and Kan, Min-Yen (2007). "Keyphrase Extraction in Scientific Publications". In: *Proceedings of the International Conference on Asian Digital Libraries*.

[38] Nguyen, Thuy Dung and Luong, Minh-Thang (2010). "WINGNUS: Keyphrase Extraction Utilizing Document Logical Structure". In: *SemEval@ACL*.

[39] Omaima, A., Aditya, J., and Huzefa, R. (2018). "Needle in a haystack: Identifying learner posts that require urgent response in MOOC discussion forums". In: *Computers Education* 118, pp. 1–9.

[40] Ounnas, Asma (2010). "Enhancing the automation of forming groups for education with semantics". In: *Doctoral Thesis, School of Electronics and Computer Science,University of Southampton*.

[41] Patel, Krutarth and Caragea, Cornelia (2019). "Exploring Word Embeddings in CRF-based Keyphrase Extraction from Research Papers". In: *Tenth International Conference on Knowledge Capture*.

[42] Pollalis, Yannis A. and Mavrommatis, George (2008). "Using similarity measures for collaborating groups formation: A model for distance learning environments". In: *European Journal of Operational Research* 193, pp. 626–636.

[43] Ramachandran, Lakshmi, Gehringer, Edward F., and Yadav, Ravi (2016). "Automated Assessment of the Quality of Peer Reviews using Natural Language Processing Techniques". In: *International Journal of Artificial Intelligence in Education* 27, pp. 534–581.

[44] Ramesh, Arti, Goldwasser, Dan, Huang, Bert, Daumé, Hal, and Getoor, Lise (2014). "Understanding MOOC Discussion Forums using Seeded LDA". In: *Proceedings of the Ninth Workshop on Innovative Use of NLP for Building Educational Applications*. Baltimore, Maryland.

[45] Reily, Ken, Finnerty, Pam Ludford, and Terveen, Loren G. (2009). "Two peers are better than one: aggregating peer reviews for computing assignments is surprisingly accurate". In: *ACM SIGCHI International Conference on Supporting Group Work, GROUP'09*.

[46] Ribeiro, Marco Tulio, Singh, Sameer, and Guestrin, Carlos (2016). ""Why Should I Trust You?": Explaining the Predictions of Any Classifier". In: *KDD '16*.

[47] Scott, A. C., M., Danielle S., Ryan, S., Yuan, W., Luc, P., Tiffany, B., and Yoav, B. (2015). "Language to Completion: Success in an Educational Data

Mining Massive Open Online Class". In: *Eighth International Conference on Educational Data Mining*. Madrid, Spain.

[48] Tucker, C. S., Dickens, B., and Divinsky, A. (2014). "Knowledge Discovery of Student Sentiments in MOOCs and Their Impact on Course Performance". In: *Proceedings of the ASME Design Engineering Technical Conference*. Buffalo, NY, USA.

[49] "Tuned Models of Peer Assessment in MOOCs" (2013). In: *ArXiv* abs/1307.2579.

[50] Wan, Xiaojun and Xiao, Jianguo (2008). "CollabRank: Towards a Collaborative Approach to Single-Document Keyphrase Extraction". In: *COLING*.

[51] Wang, Yue, Li, Jing, Chan, Hou Pong, King, Irwin, Lyu, Michael R., and Shi, Shuming (2019). "Topic-Aware Neural Keyphrase Generation for Social Media Language". In: *The 57th Annual Meeting of the Association for Computational Linguistics (ACL)*.

[52] Witten, Ian H., Paynter, Gordon W., Frank, Eibe, Gutwin, Carl, and Nevill-Manning, Craig G. (1999). "KEA: practical automatic keyphrase extraction". In: *ArXiv* cs.DL/9902007.

[53] Wu, Run-ze, Liu, Qi, Liu, Yuping, Chen, Enhong, Su, Yu, Chen, Zhigang, and Hu, Guoping (2015). "Cognitive Modelling for Predicting Examinee Performance". In: *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*.

[54] Xiong, Wenting, Litman, Diane J., and Schunn, Christian D. (2010). "Assessing Reviewer's Performance Based on Mining Problem Localization in Peer-Review Data". In: *Proceedings of 3rd International Conference on Educational Data Mining*.

[55] Yang, D., Wen, M., Howley, I. K., Kraut, R. E., and Penstein, C. (2015). "Exploring the Effect of Confusion in Discussion Forums of Massive Open Online Courses". In: *Proceedings of the Second International Conference on Learning at Scale (L@S)*. Vancouver, British Columbia, Canada: ACM Press.

[56] Yang, Zhilin, Dai, Zihang, Yang, Yiming, Carbonell, Jaime G., Salakhutdinov, Ruslan, and Le, Quoc V. (2019). "XLNet: Generalized Autoregressive Pretraining for Language Understanding". In: *NeurIPS*.

[57] Al-Zaidy, Rabah A., Caragea, Cornelia, and Giles, C. Lee (2019). "Bi-LSTM-CRF Sequence Labeling for Keyphrase Extraction from Scholarly Documents". In: *WWW '19*.

[58] Zeng, Jichuan, Li, Jing, Song, Yan, Gao, Cuiyun, Lyu, Michael R., and King, Irwin (2018). "Topic Memory Networks for Short Text Classification". In: *EMNLP*.

[59] Zhang, Qi, Wang, Yuhuai, Gong, Yeyun, and Huang, Xuanjing (2016). "Keyphrase Extraction Using Deep Recurrent Neural Networks on Twitter". In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*.

[60] Zhang, Yingyi and Zhang, Chengzhi (2019). "Using Human Attention to Extract Keyphrase from Microblog Post". In: *The 57th Annual Meeting of the Association for Computational Linguistics (ACL)*.