

Article

# Performance Evaluation of Keyword Extraction Methods and Visualization for Student Online Comments

Feng Liu <sup>1</sup>, Xiaodi Huang <sup>1,\*</sup>, Weidong Huang <sup>2</sup> and Sophia Xiaoxia Duan <sup>3</sup>

<sup>1</sup> School of Computing and Mathematics, Charles Sturt University, Albury 2640, Australia; fliu@csu.edu.au

<sup>2</sup> Faculty of Transdisciplinary Innovation, University of Technology Sydney, Sydney 2007, Australia; weidong.huang@uts.edu.au

<sup>3</sup> School of Accounting, Information Systems and Supply Chain, RMIT University, Melbourne 3000, Australia; sophia.duan2@rmit.edu.au

\* Correspondence: xhuang@csu.edu.au

Received: 18 October 2020; Accepted: 19 November 2020; Published: 22 November 2020



**Abstract:** Topic keyword extraction (as a typical task in information retrieval) refers to extracting the core keywords from document topics. In an online environment, students often post comments in subject forums. The automatic and accurate extraction of keywords from these comments are beneficial to lecturers (particular when it comes to repeatedly delivered subjects). In this paper, we compare the performance of traditional machine learning algorithms and two deep learning methods in extracting topic keywords from student comments posted in subject forums. For this purpose, we collected student comment data from a period of two years, manually tagging part of the raw data for our experiments. Based on this dataset, we comprehensively compared the five typical algorithms of naïve Bayes, logistic regression, support vector machine, convolutional neural networks, and Long Short-Term Memory with Attention (Att-LSTM). The performances were measured by the four evaluation metrics. We further examined the keywords by visualization. From the results of our experiment and visualization, we conclude that the Att-LSTM method is the best approach for topic keyword extraction from student comments. Further, the results from the algorithms and visualization are symmetry, to some degree. In particular, the extracted topics from the comments posted at the same stages of different teaching sessions are, almost, reflection symmetry.

**Keywords:** student comments; metrics; machine learning; topic keywords extraction; Naïve Bayes; LogR; SVM; CNN; Att-LSTM; visualization

---

## 1. Introduction

Nowadays, higher education is considered a service industry, and it is expected to meet the expectations of stakeholders by re-evaluating the education system [1,2]. The importance of student feedback is acknowledged by universities. In particular, student comments are regarded as an effective way to discuss learning issues online. Comments, posted by students on forums, provide valuable information when it comes to teaching and learning. The purpose of this work is to automatically extract keywords from student comments to enhance teaching quality. In fact, topic keywords have been used for text summarization, terms index, classification, filtering, opinion mining, and topic detection [3–8]. Topic keywords are usually extracted by domain experts. It is an extremely time-consuming, complex work. In contrast, automatic extraction is very efficient. Automatic topic keyword extraction refers to the automatic selection of important and topical words from the content [9]. However, different approaches have different performances. These approaches are performed to discover which approach performs best in extracting topic keywords automatically from a particular type of dataset.

A topic keyword, extracted from a document, is a semantic generalization of a paragraph or document, and an accurate description of the document's content [10–12]. In other words, a topic keyword can be regarded as the category label of a generalized paragraph or document by using its keywords as the features. The extraction accuracy of topic keywords affects many tasks in natural language processing (NLP) and information retrieval (IR), such as text classification, text summarization, opinion mining, and text indexing [13]. Different approaches are used to automatically extract the keywords from the text to improve performance.

However, there are challenges concerning how to improve the accuracy of topic keyword extraction. With big data growing, unstructured data are everywhere on the internet. Moreover, there are often many language fragments or documents without topic keywords. This makes it even more difficult to process and analyze them. There are many methods and various applications for keyword extraction. The accuracy of topic keyword extraction must continuously improve.

The problem of this work is to extract the topic keywords from student comments, and predict the topic for a new given comment by using supervised learning. As a category label, each of the topic keywords, such as *assignments*, *online meeting*, and *topic 2* is the summary of semantically related keywords in a group of comments. A feature vector consisting of the keywords, or their embedding in each comment is used to predict its corresponding category label. In this paper, we compare several approaches to extract topic keywords from student comments. Specifically, the compared machine learning algorithms are Naïve Bayes, logistic regression (LogR), support vector machine (SVM), convolutional neural networks (CNN), and Att-LSTM. The performance of the first three algorithms relies heavily on manually selected features, while the last two algorithms are capable of automatically extracting discriminative features from the comments in the training data.

The contributions of this paper are as follows: (1) we compare the performance of different typical algorithms against the subject database in terms of several evaluation metrics; (2) we visualize the keywords and their similarities in the dataset; and (3) we evaluate the performance of the algorithms by using metrics, with visualization of the results of algorithms as further validation.

The paper is the extended version of the conference paper [14]. New content includes a more thorough literature review, and more comprehensive methodology descriptions, algorithm comparisons, and analysis results. The rest of this paper is organized as follows. In the following section, we review related work. Section 3 presents the compared algorithms, followed by the experiments in Section 4. Section 5 describes the visualization results and Section 6 concludes this paper.

## 2. Related Work

Overall, keyword extraction approaches can be classified into simple statistical approaches, linguistic approaches, machine learning approaches, and other approaches [15].

Simple statistical approaches do not require training. They generate topic keywords from candidates based on analysis and statistics of text and calculation of word frequency, probabilities, and other features extracting, such as term frequency inverse document frequency (TF-IDF) [16], word frequency [17], N-Gram [18], and word occurrences [19]. Most statistical approaches are used by unsupervised learning approaches. The mechanism of the methods assigns weights to each word and calculates them by feature detection.

Based on the linguistic features of words, sentences, and entire documents, linguistic approaches include words [20], the syntax [21], and context to analyze [22,23] and find topic keywords in the documents [15]. For linguistic approaches, the analysis is very complex, requiring some linguistic skills.

Machine learning approaches are classified into four main types of learning: supervised learning [24], semi-supervised learning, unsupervised learning [25], and reinforcement learning. With the development of machine learning and NLP, and the various application fields, different machine learning methods for topic keyword extraction have been successfully applied. As supervised learning, these methods use existing datasets to conduct a lot of training on the algorithm model and adjust the parameters in a way that high accuracy of the predictions can be achieved. For topic

keyword extraction, supervised learning is, however, commonly used. For this work, the compared algorithms extract the topic keywords from the text based on the existing algorithms. In particular, we choose Naïve Bayes (NB) [26], logistic regression [27], SVM [28], and two deep learning algorithms as approaches for the keyword extraction.

As machine learning methods, which are based on artificial neural networks, deep learning has achieved high performance in many areas. In this work, two types of deep learning algorithms are used for topic keyword extraction. Deep learning [29] constructs multiple layers of neural networks. It achieved great success recently, in many areas, by automatically extracting keywords from raw text. The domain includes various tasks in natural language processing, such as language modeling, machine translation, and many others. Some researchers used the recurrent neural network (RNN) to extract topic keywords from different scale texts. Zhang et al. used RNN, LSTM, and others to compare the approaches of extracting topic keywords from Twitter datasets. Their deep learning methods achieved good performances.

Long Short-Term Memory (LSTM), a particular type of RNN, achieved a good performance in topic keyword extraction from many domain contents for different purposes. It differs from traditional RNN, and is well suited to classify, process, and predict text problems. Moreover, CNNs perform greatly in image recognition. Recently, CNNs have been applied to text classification—performing very well—but the results are always affected by the quality of the datasets. In this work, these two algorithms will be applied to the student comments and their performances will be compared.

Other keyword extraction approaches use some of the methods above combined with special features to obtain the topic keywords from the text. These features, for example, can include word length, text formatting, and word position [30].

In the following, we review the algorithms used in our experiments.

Keyword extraction can also be thought of as binary text classification. In other words, it determines whether a particular candidate keyword is the exacted topic keyword or not. Based on these binary approaches, topic keywords can also be used for other applications, such as browsing interfaces [31], thesaurus construction [32], document classification, and clustering [33,34].

## 2.1. *Naïve Bayes*

As a simple, effective, and well-known classifier, Naïve Bayes (NB) uses the Bayes probability theory and statistics, with good performance, on high dimensionality of input [35].

Kohavi [36] showed that the performance of NB was not as good as decision trees on the large datasets, so the author combined NB into a decision tree called NB-Tree. In other words, this approach integrated benefits from both the Naïve Bayesian classifier and decision tree classifier. As such, it performed better than each of its components, especially for large datasets. In particular, the decision tree nodes used univariates as regular decision-trees and the leaves as Naïve-Bayes classifiers.

Yasin et al. [37] used a Naïve Bayesian approach as a classifier to extract the topic keywords from the text with supervised learning. In their work, they assumed that the keyword features were normally distributed, independently. The approach extracted topic keywords from the testing set with the knowledge of training gained. The features were used by TF-IDF scores, word distances, paragraph keywords, and sentences from the text.

In the text classification by Kim et al. [38], NB was regarded as the parameter estimation process, resulting in the lower accuracy of classification. They proposed per-document text normalization and feature weighting methods to improve its performance. For the classification task, the NB classifier showed a great performance. The Poisson NB was weight-enhancing, by assuming that the input text was processed by the multivariate Poisson model.

## 2.2. *Logistic Regression*

As a statistical model, LogR is the core method used by a logistic function. It is also called a sigmoid function, the shape of which is like an s-shape, especially performing best in binary classification.

However, it is easy to extend for a multi-class task. It uses probabilities for the classification problem with two outcomes, such as diagnosis of spam emails. It predicts whether or not an email is a spam by using the different features of the email. LogR is used for relationship analysis in dichotomous issues. For text classification, LogR is widely applied to the binary task because the output of LogR probability is between 0.0 and 1.0 [39]. LogR performs well on these tasks, which predict the resultant presence or absence of a feature of the outcome by the Logistic function.

Tsien et al. [40] used a classification tree and LogR to diagnose myocardial infarction. They compared the performance of the Kennedy LogR against the Edinburgh and Sheffield dataset, with improving ROC up to 94.3% and 91.25%, respectively.

Using LogR, Padmavathi supports clinicians in diagnostic, therapeutic, or monitoring tasks. The model predicted the presence or absence of heart attack and coronary heart disease classification. Padmavathi [41] provided criteria that could affect the model building of the regression model in different ways and stages.

### 2.3. Support Vector Machine

In machine learning algorithms, SVM is a supervised learning model that has great performance in classification [42]. In a simple case, SVM attempts to find out a hyperplane, to separate it into two categories, where a data record represents a point in space. It aims to find a hyperplane by a dataset, which divides the results into two categories with a maximum margin. The extensions of SVM can also be used for processing non-learner classification, from an unlabeled dataset. It can be learned by the unsupervised learning approach. This work will focus on using the linear classification of SVM. SVM is used for classification and regression tasks with great performance. Support vector machines can handle numerical data so that the input data are transformed into numbers. The kernel types of SVM are polynomial, neural, Epanechnikov, Gaussian combination, and multiquadric [43].

Zhang et al. [44] used SVM to extract a subset of topic keywords from documents to describe the “meaning” of the text. They utilized global context information and local context information for topic keyword extraction. The methods were based on the SVM for performing the tasks. The results showed that the methods performed better than baseline methods, and the accuracy was significantly improved for the keyword extraction. Isa et al. [45] used a hybrid approach with NB and SVM approaches to predict the topic of the text. Bayesian algorithms vectored text through probability distributions. The probability of each category of the document overcame the effect of dimensionality reduction by using SVM. The combined method can work with any dataset and is compared with other traditional approaches. It reduces training time with greatly improved accuracy. Krapivin et al. [46] used natural language processing technologies to enhance the approaches of machine learning, such as SVM and Local SVM, to extract topic keywords from the documents. Their research showed that the performance of SVMs had better results on the same dataset than KEA, which was based on Bayesian learning.

### 2.4. Convolutional Neural Networks

As a type of a neural network, CNN is a forward feed deep neural network. “Convolution” is a mathematical operation, a specialized linear operation [47]. The typical CNN structure includes convolutional layers, pooling layer, fully connected layers, and dropout layers [48]. CNN is applied widely in different areas, achieving great performance. It is also effective for NLP tasks.

Kim et al. [49] first used CNNs to classify text classification. They trained the CNN on the data processed by word vectors for the sentence-level classification. The simple CNN model achieved great results on multi-benchmarks with a small number of hyper-parameters and static vectors. Moreover, they proposed the model that could use static vectors without significantly modifying the structure. This CNN model improved the performance on 4 out of 7 tasks, which included the question classification and sentiment analysis.

Vu et al. [50] investigated RNN and CNN. The two different ANN models were for the relation classification task and the performance of different architectures. They gave a new context expression

for CNN on the classification task. Moreover, they presented Bi-Directional RNN and optimized the ranking loss. Their research showed that a voting scheme could improve the accuracy of the combined model of CNN and RNN on the task of SemEval relation classification. Their approaches had a great performance.

Deep learning approaches have revolutionized many NLP tasks. They perform much better than traditional machine learning methods so that the approaches widely explore various tasks. Wang et al. [51] compared CNN with RNN on natural language processing tasks, such as relation classification, textual entailment, answer selection, question relation match, path query answering, and art-of-speech tagging [52]. According to their results, they suggested RNN had good performance concerning a range of NLP tasks. CNN performed better on topic keyword recognition in sentiment detection and question and answer matching tasks. Furthermore, they thought the hidden size and batch size could affect the performance of DNN approaches. This is the first work on comparing CNN with RNN for NLP tasks, and exploring some guidance for DNN approaches selection.

Hughes et al. presented an approach to classify clinical text automatically by sentences. They used CNN to learn complex feature representations. The approach was trained by a wide health information dataset based on the emergent semantics, extracted from a corpus of medical text. They compared the other three methods—sentence embedding, mean word embedding, and word embedding—with bags of words [53]. The research showed that their model, based on CNN, outperformed other approaches, with improved accuracy of more than 15% in the classification task.

## 2.5. Long Short-Term Memory

As a type of special RNN, long short-term memory (LSTM) can learn long-term dependencies on time series data [54]. Sepp Hochreiter and Jürgen Schmidhuber proposed LSTM to overcome the errors of back-propagated problems in recurrent neural networks [55]. RNN connects neural nodes to form a directed graph, which generates an internal state of the network that allows them to exhibit dynamic behavior. RNN saves the recent event state as activation of the feedback connections [55].

Wang et al. [56] used word embedding and LSTM for sentiment classification to explore the deeper semantics of words from short texts in social media. The word embedding model was used for learning word usages in different contexts. They used the word-embedding model to convert a short text into a vector, and then input it to the LSTM model for exploring the dependency between contexts. The experimental results showed that the LSTM algorithm is effective in the word usage of context for social media data. When the model is trained enough, it also identifies that the accuracy is affected by the quality and quantity of training data. Wang et al. [57] used an LSTM based attention model for aspect-level sentiment classification. This model takes advantage of the benefits of the embedding model and a deep learning approach. An attention model can change the focus on the different parts of the sentences when the mode is received in different aspects. With experiments on the SemEval 2014 dataset, the results showed that their model had a good performance on aspect-level sentiment classification. The advantage of this strategy is in learning aspect embeddings, to compute the attention weights. The idea of this approach is in aspect embedding, to join computing attention weights. For different aspects, the model focuses on different parts of sentences. The results showed that the models of AE-LSTM and ATAE-LSTM performed better than the baseline models. They constructed datasets by data crawling from “jd.com” to collect product reviews. This approach attempts to provide useful reviews for potential customers and helps reduce manual annotation topic keywords in e-commerce. The experiments showed that the bi-directional LSTM recurrent neural network approach has a high accuracy for keyword extraction.

## 2.6. Attention Mechanism

In recent years, the attention mechanism has frequently appeared in NLP of literature or blog posts, which shows that it has become a fairly popular concept, and has played a significant role in the field of NLP.

The attention mechanism was initially applied in computer vision. The Google Mind team [58] used an attention mechanism on RNN for image classification. Attention is a kind of a vector—outputs of a dense layer via the softmax function. ANN, with an attention mechanism, can understand the meaning of the text. The attention-based ANN model can ignore the noise of text, focus on the keywords of the text, and know which words can be related. Zhou et al. [59] used Att-BLSTM to capture the important semantic information from sentences. By experimenting on the SemEval-2010, the results showed its great performance for the relation classification task. Vaswani et al. [60] proposed a base solely on the attention network model, named Transformer, without recurrence and convolution. This model was trained on two machine translation tasks, with the best performances on them. The model also has some benefits. For example, it is more parallelizable and it saves time when it comes to training. For the task of translating WMT 2014 English into German, it delivered better results than other existing models. Dichao Hu [61] introduced and compared the different attention mechanisms in various types of NLP tasks to explore attention mathematical justification and its application.

## 2.7. Word Embedding

For NLP tasks, word representation is a necessary and fundamental technique for neural network algorithms. Word-vector embedding refers to feature learning techniques in which a word or phrase is mapped to a vector of real numbers. As one of the word embedding methods, Word2vec has widely been used in NLP.

Word embedding, being used as the common input representation, increases the performance of NLP tasks [62]. It also contains word relationships and plenty of semantic information. In this research, the word embedding technique is used as input data for CNN and Att-LSTM.

Word embedding, as the most popular word vectorization method, can capture the word context and word relationship. It has been researched for several years. It can map words into vectors from the vocabulary. Thus, it is important in natural language processing. The word embedding technique converts the word feature from a higher-dimensional to a lower-dimensional vector space. For producing the map by the neural network, dimensionality reduction can be applied from a word matrix. By using an input representation, word embedding improves the performance of many NLP tasks.

Levy and Goldberg [63] analyzed skip-gram with negative-sampling. They found the NCE embedding method could factorize a matrix, with each cell being the log conditional probability of words in the context. Their work showed that words improved the results of the two-word similarity by the sparse shifted positive PMI word-context matrix.

Ganguly et al. [64] improved retrieval by using word embedding. They constructed a language model to gain the conversion probabilities between words. The model captured terms to fit into the context and solved lexical mismatch problems by considering other related terms in the collection. The model experimented on TREC 6–8; robust tasks and the results showed better performance on language models and LDA-smoothed LM baselines.

## 3. Compared Algorithms

In this section, we briefly describe our compared algorithms in this paper. For the problem of extracting topic keywords, the keywords included in a comment are converted into a feature vector denoted as  $x$ , and the corresponding label of the comment as  $y$ . The question is to predict its  $y$  label for a given comment of  $x$ .

### 3.1. Naïve Bayes

NB assumes that all features are independent of each other. This is why it is called the “Naïve” Bayes’ theorem formula; it is as follows

$$p(y|x) = p(x, y)/p(x) \quad (1)$$

where  $p(y|x)$  is the class posterior to discriminate  $y$  into different classes with features vector  $x$ ,  $p(y)$  class priors, and  $p(x)$  the probability of instance  $x$  occurring. The algorithm has its input as a set of feature vectors  $x \in X$ , where  $X$  is the features space, and its output as the labels  $y \in \{1, \dots, C\}$ . It is a conditional probability model that classifies the data by maximum likelihood.

### 3.2. Logistic Regression

LogR uses probabilities for a classification problem with two outcomes. For a classification problem, the prediction model returns a value scoring between **0** and **1**. If the value is more than the threshold, the observation will classify into class one, otherwise, it will be classified into class two.

The equation of Logistic function is as follows

$$y = f(x) = \frac{L}{1 + e^{-k(x-x_0)}} \quad (2)$$

where  $x_0$  is the  $x$  of the midpoint,  $k$  the logistic steep of the curve,  $x$  the vector of input features, and  $y$  the classified variable.

### 3.3. Support Vector Machine

SVM aims to find a hyperplane by a dataset that is ordered to divide the results into two categories with a maximum margin. This research will focus on the linear classification of SVM.

Linear SVM needs to input a labelled data with paired, the mechanism is as follows: for dataset  $D = (x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$ ,  $D$  include number  $n$  couple of elements, where  $y \in \{-1, 1\}$ ,  $x$  is the features. SVM uses the formulation to find out the hyperplane to separate the dataset into two classes.

The equation:  $y = \omega x + b$  with the constraints:  $-\omega x_i - b \leq \varepsilon$  and  $\omega x_i + b - y_i \leq \varepsilon$ .

SVM algorithms are a cluster of kernel functions that can be used for many types of classification problems; the functions include polynomial, Gaussian, Gaussian radial basis function, Laplace RBF kernel, and sigmoid kernel.

### 3.4. Convolutional Neural Networks

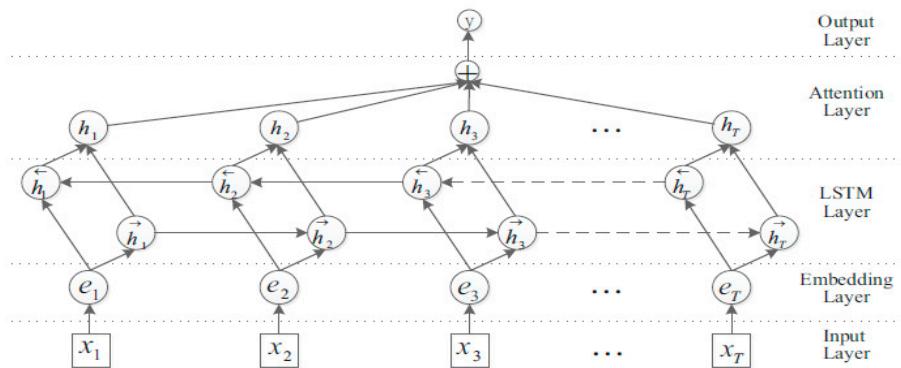
CNN is composed of four main steps: convolution, sub-sampling, activation, and complete connection. It is different from conventional neural networks in terms of signal flow between neurons [65]. For a text classification problem, CNN includes three layers and predicts in the task of text classification. The difference between computer vision and text classification is the kind of data for the input layer. For images, it is pixels of the picture, while for text, a matrix of the word vectors for inputting.

The CNN for text classification was proposed for sentence classification by Yoon Kim in 2014. The input data represented as  $k$ -dimensional words, corresponding each word of the  $n$ -length sentence. Fillets play convolution to extract feature maps on the numeric of text vertically. Pooling works in pooling layer, it performs on each map, recording the max number from each feature map. The 9 univariate vectors concatenated together to generate a feature vector for the single feature vector as input to the next layer; then the softmax layer classify text as a result. For binary classification, the model classifies the result into two states, such as “true” or “false”.

### 3.5. Attention-Based Long Short-Term Memory

For learning sequences from the data, LSTM makes use of input, output, and forget gates to converge on meaningful representation, as shown in Figure 1. However, the length of the sequences is limited. AT-LSTM can avoid the long-term dependence problem with better interpretability. Specifically, the attention learns the weighting of the input sequence and averages the sequence to obtain the relevant information. AT-LSTM model consists of five layers. Each layer is briefly described as follows: Attention-based input layers: accepts the sentences to the model. Embedding layer: converts words in

the sentences to the number of embedding vectors, where each word is mapped to a high-dimensional vector. LSTM layers: captures the higher feature from embedding layers output. Attention layer: a weight vector is generated, which is multiplied by this weight vector to merge the features of the lexical level in each iteration, into the features of the sentence level. Output layers: targets classification of feature vectors at the sentence level.



**Figure 1.** Attention-based Long Short-Term Memory (LSTM) framework with five layers [60].

#### 4. Experiment and Results

In this section, we describe our experiments and report the results.

##### 4.1. Dataset Description

This section describes the datasets used in our experiments. To compare the performance of several machine learning algorithms on the topic keyword prediction, we construct our dataset from two datasets called FR-III-KB and ITC-114.

FR-III-KB dataset was downloaded from Kaggle (<https://www.kaggle.com/c/facebook-recruiting-iii-keyword-extraction/data>). The topics of student comments in these datasets are about their studies on database knowledge topics, such as database design, database management, assignments, and online meetings. The dataset consists of four columns, called ID, title, body, and tags. Each of these columns is used for describing one attribute of the questions, such as an ID for a unique identifier, title for the title, body for description, and tags for the keywords. FR-III-KB dataset includes 1,189,290 recordings, with the longest words of 89 and the shortest of 7. There are 19.07% recordings about database knowledge in this dataset. All of the initial data on comments are collected from websites. The formats contain some HTML tags, which are cleaned by using a regular expression. The algorithms are implemented by using Natural Language Toolkit (NLTK) and TensorFlow Keras on Python 3.6.

ITC-114 is a subject on database systems offered at Charles Sturt University in Australia. The subject has a website with a subject forum for students and lecturers where the subject issues are commented on and discussed. The dataset of student comments was collected for this work. All personal information in the dataset was removed and a few items of each comment, such as a forum, thread, topic, and comment post were retained. In particular, the content of this dataset is about the questions and discussions of teaching and learning of this subject, which include greetings, textbook, database design, assessments, SQL, and the final exam. For two years from Session 3 of 2015 to Session 3 of 2017, this dataset consists of 793 comments posted by 169 students over 344 topics in total, with the longest comment of 92 words and the shortest of 2 words. Among them, 450 comment recordings were labelled manually for training.

For the final dataset in our experiments, we constructed 1,189,290 records about the database knowledge text from Facebook Recruiting III-Keyword Extraction called FR-III-KB, integrated with the ITC-114 dataset.

After being trained by FR-III-KB and ITC-114 student comments, the compared algorithms predicted part of the topic keywords of the ITC-114 dataset. During our training, the dataset was

separated into three parts: the training set, validation set, and testing set. The training set is used for training models and the validation set for the evaluation of models after training. The test set is used for predicting the topic keywords.

As shown in Table 1, the data partitioning method uses a common strategy in the literature on machine learning. Specifically, the ratio for each part is 70% of the entire data for training, 10% for validation, and 20% for testing.

**Table 1.** The entire dataset.

Category	Training	Validation	Test
FR-III-KB	830,503	118,929	237,858
ITC-114	405	45	345
Our built dataset	830,908	118,974	238,203

All of the experiments were constructed on a computer with Ubuntu Linux 18.04, Python3, TensorFlow v1.14, TensorFlow Keras, NLTK v3.4, Scikit-learn v0.21, Regular Expression, Pandas, Numpy, and other third parties of python libraries installed.

#### 4.2. Text Pre-Processing

As an initial and necessary step, data preprocessing cleans the noises to ensure the high quality of data to enhance the performance of models. Data cleaning, the first critical task for any data relational project, also called data cleansing [66], removes the format tags and errors for the next step in data analysis. The specific preprocessing in this work includes data cleaning, and selection of the word features for NB, LogR, and SVM. For our two datasets, the text paragraphs are collected from websites with some HTML tags and other format problems. We use the regular expression and NLTK to clean the format tags and other data noises, followed by removing the stop words and punctuation, converting all capital letters, and restoring abbreviations.

We select some features to produce the candidate topic keywords for NB, LogR, and SVM. The features are word frequency, word position, and word probability, length, part of speech, the occurrence, line position, posterior position, and standard deviation. The details of these features are shown in Table 2. For deep learning, the text can be tokenized into words. Each of these words is then represented as a vector. Deep learning algorithms accept the number of word vectors of the input data for training.

**Table 2.** Selection of Word Features.

Features	Comment
position	“Post” “Both” “Tag”
frequency	Word frequency
posterior probability	Posterior Probability
length	Candidate the length of topic keywords
part of speech	Word class, such as NN, IN, and JJ.
the occurrence	The ordinal number from the candidate words list.
line position	Line position number.
parabolic position	Parabola position.
standard deviation	The average position to all received text.
frequency	Candidate word frequency.

#### 4.3. Result and Discussion

In this section, we first compare the performance of the five algorithms in terms of the metrics of precision, recall, F1 score, and accuracy against this project dataset. We then discuss the experimental results.

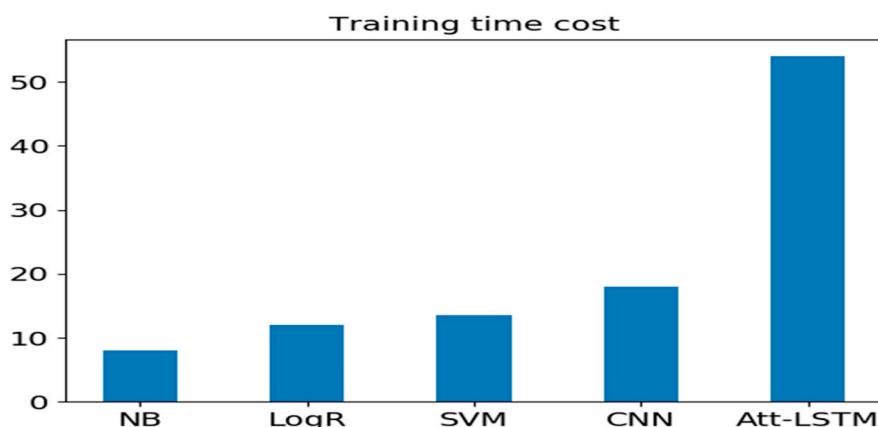
Our experimental results are reported in Table 3, where their performance is scored by the four metrics. Note that all algorithms were trained by supervised learning.

**Table 3.** Comparisons of the five algorithms.

Approaches	Precision	Recall	F1-Score	Accuracy (%)
NB	0.1076	0.04	0.0583	73.8647
LogR	0.269	0.8143	0.4344	75.478
SVM	0.297	0.819	0.4359	77.6895
CNN	0.632	0.5	0.6343	81.22
Att-LSTM	0.8523	0.83	0.8395	84.01

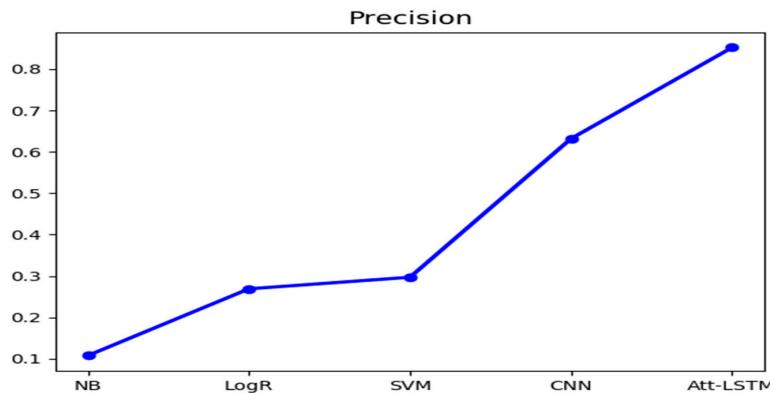
Preparing stage: NB, LogR, and SVM are classical machine learning algorithms, which select some features before the model training. For the NB algorithm, the features are required to be independent of each other. In LogR and SVM, the selected features can capture the characteristics of the text. For CNN, the model selects features automatically.

Training stage: as shown in Figure 2, NB, LogR, and SVM were trained fast. In particular, NB run the fastest among the compared algorithms. LogR and SVM cost little more time than NB, but they saved a lot of time than neural networks. The attention-based LSTM took much time than the other four algorithms.



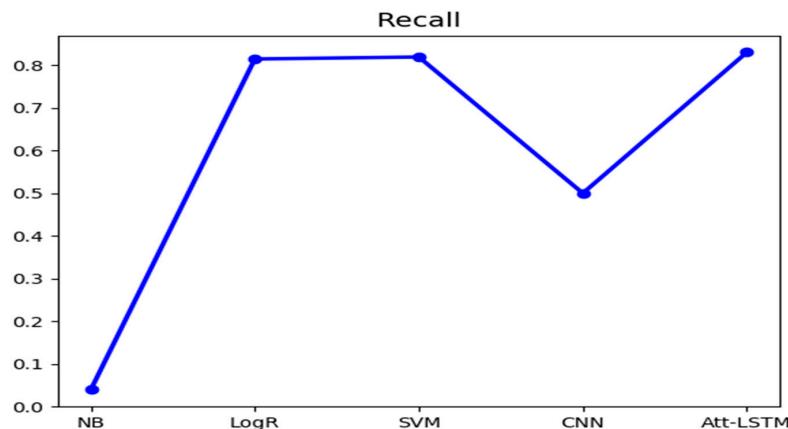
**Figure 2.** Comparisons of training time (minutes) for the five algorithms.

Precision: the precision metric measures the capabilities of the models in the positive prediction. As shown in Figure 3, for the student comment dataset, deep learning approaches performed better than the traditional machine learning algorithm. Att-LSTM performed best, with its precision of up to 85.23%, while NB had 10.76%. The linear models, LogR, and SVM achieved similar scores in precision. Note that these models were trained by using the same feature selection method.



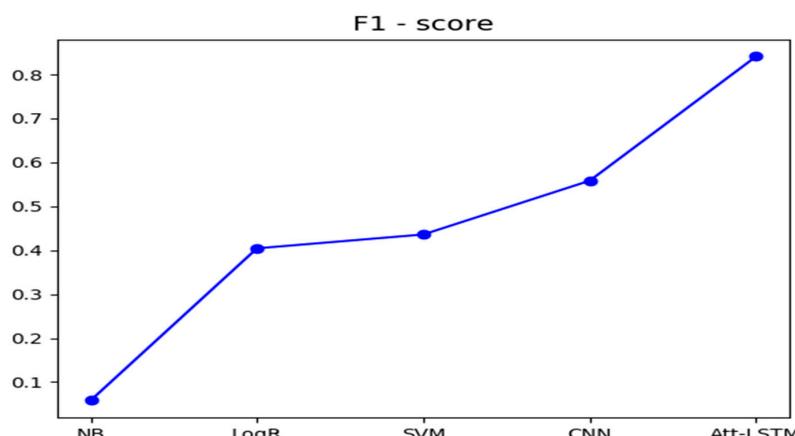
**Figure 3.** Precision of the five algorithms on the database.

Recall: a common metric used in machine learning. As shown in Figure 4, the three algorithms of Att-LSTM, LogR, and SVM performed better than NB and CNN. For NB that achieved the lowest score of recall. This metric score can be improved if the training dataset is used. The recall of the CNN had a better recall score. This improvement can further be made by data quality.



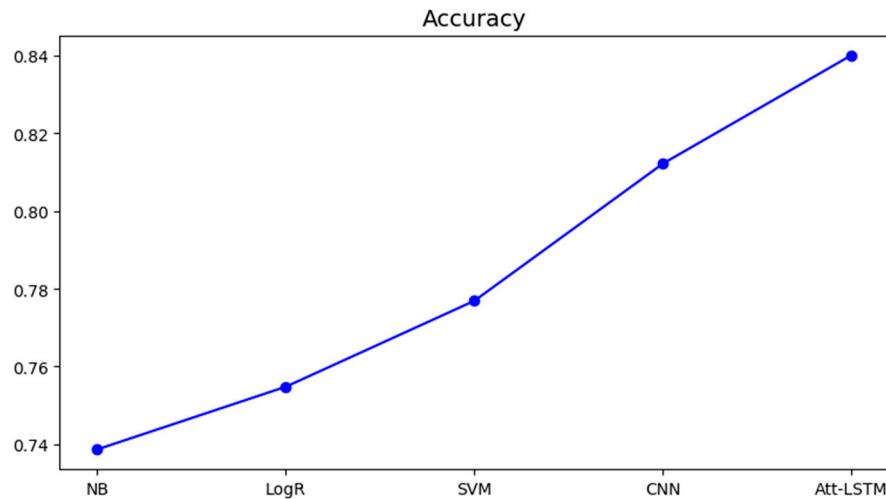
**Figure 4.** Recall of the five algorithms.

F1-score: the achieved metric F1-scores of the algorithms are shown in Figure 5. CNN and Att-LSTM had higher scores than NB, LogR, and SVM. All the models were evaluated by a combination of the accuracy and recall of the metrics. The traditional machine learning algorithms had a similar performance, while neural networks had also close results.



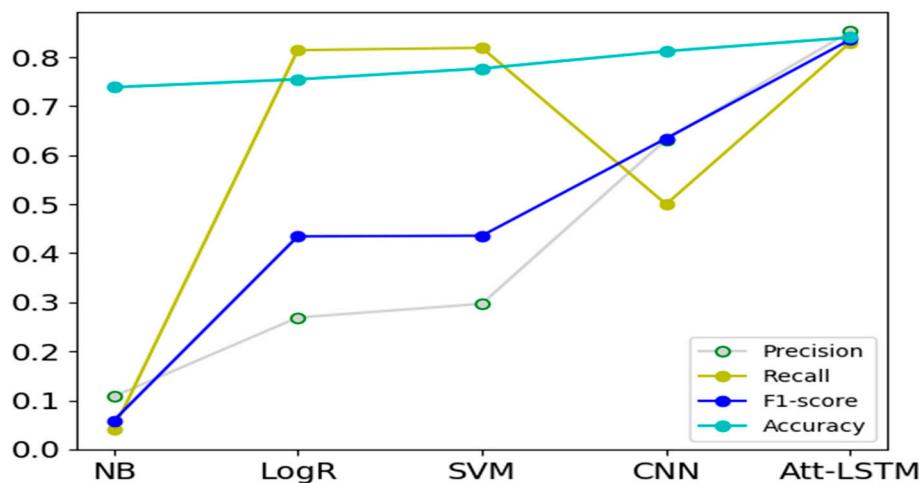
**Figure 5.** F1 scores of the five algorithms.

**Accuracy:** Figure 6 shows the results of the metric of accuracy for the five algorithms. It evaluates the performance of the correct prediction on the student dataset. Att-LSTM and CNN produced higher accuracy than traditional approaches. Note that the accuracy was calculated by regarding the negative cases with one or more correct keywords and incorrect keywords as true.



**Figure 6.** Accuracy of the five algorithms.

For predicting the topic keywords, we compared the five machine learning algorithms against the dataset of student comments in our experiments. Overall, neural networks performed better than traditional machine learning algorithms but took longer running times. In particular, Att-LSTM had achieved the highest performance in terms of the four metrics of precision, recall, F1 score, and accuracy, as shown in Table 3 and Figure 7. CNN performed a little bit lower than Att-LSTM, but higher than NB, LogR, and SVM. Taking less time on their training and predictions, NB, LogR, and SVM relied on the manually selected features for the training. Such features selection could affect their performance.



**Figure 7.** Comparing four metrics of the five algorithms.

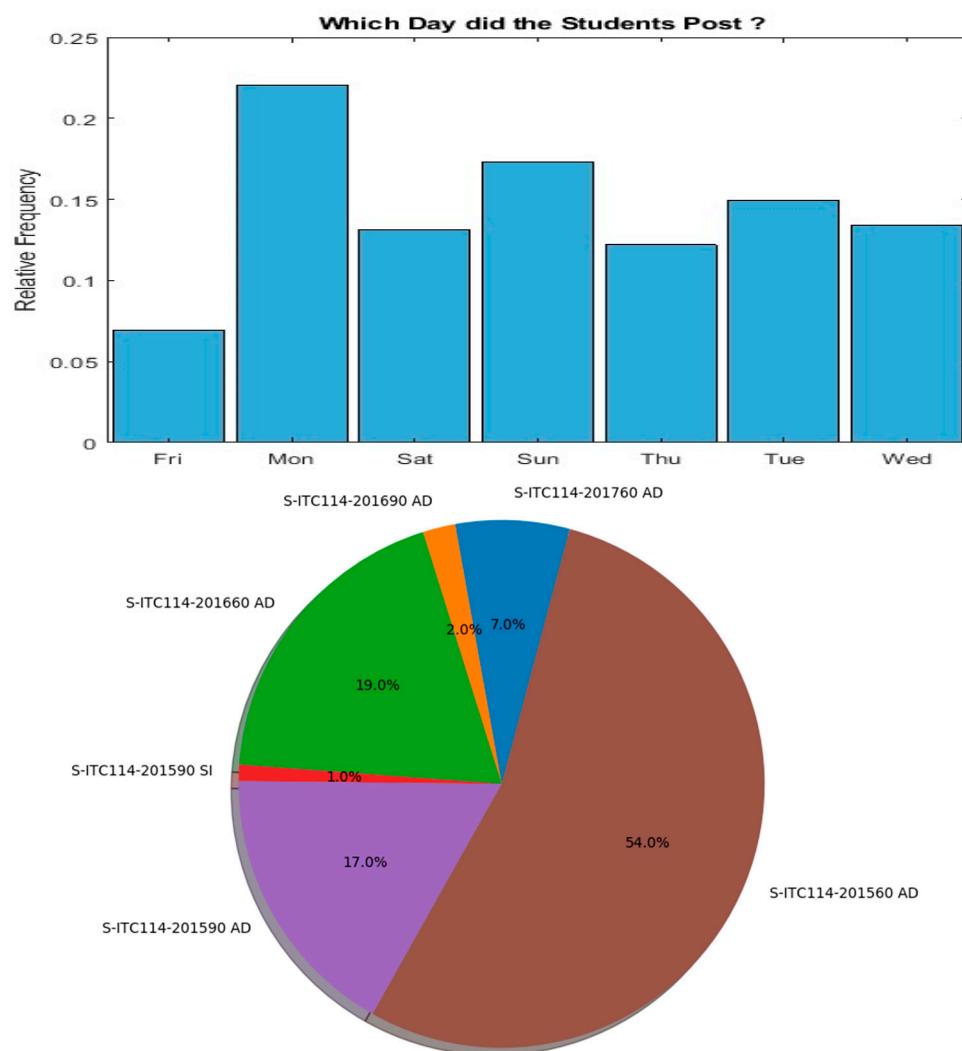
## 5. Visualization of Student Online Comments

So far, we have reported our comparisons of the five algorithms on extracting topic keywords from the student comments. However, the question of why these algorithms are working on this particular type of our dataset remains unanswered. Are there any patterns in student comments? If there are no regularities in these comments, the algorithms cannot detect or predict any categories. In this section, we make use of visual analysis [67–73] to better understand student comments and to compare the

degree of consistency and symmetry among the visualization results against some algorithm results. In particular, we first find some patterns in the student comments of the ITC-114 database and then visualize the keywords in these comments. From the experiments, it has been demonstrated that: (1) the visualization results provide some explanations of why the machine learning algorithms are capable of finding the categories; and (2) the resulting keywords produced by some algorithms are consistent with the visualization results.

### 5.1. Patterns in Student Comments

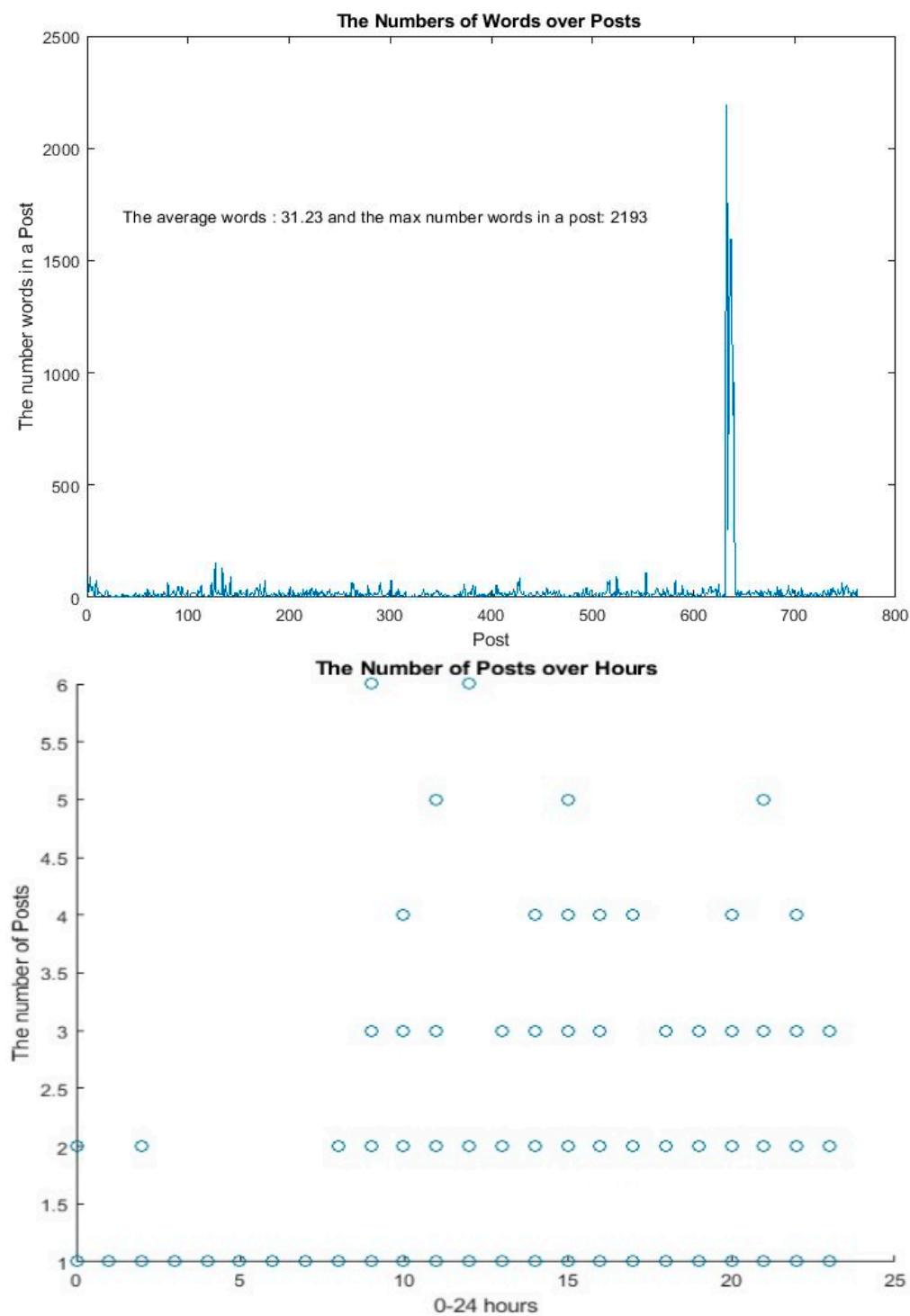
As shown in Figure 8, we examine the number of comments on different days and over different teaching sessions of the ITC-114 database.



**Figure 8.** Student post days; 2015–2017 are for the teaching years, 60, 90 for sessions, and AD (distance students), SI (Sydney internal students) for the student cohorts.

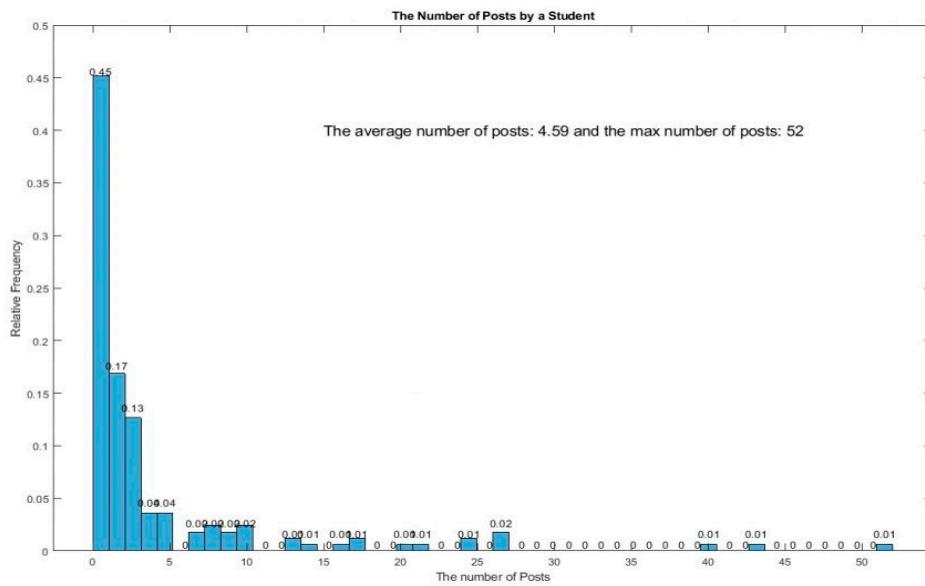
The percentages of comments from the ITC-114 database on different days are 22.05% for Monday, 17.32% for Sunday, 14.96% for Tuesday, 13.39% for Wednesday, 13.12% for Saturday, 12.20% for Tuesday, and 6.96% for Friday.

Next, the number of words in the posts and the number of words posted over hours from the ITC-114 database are shown in Figure 9.



**Figure 9.** The numbers of words over posts and posts over hours.

The distribution of comments by per student in the dataset of ITC-114 is shown in Figure 10.

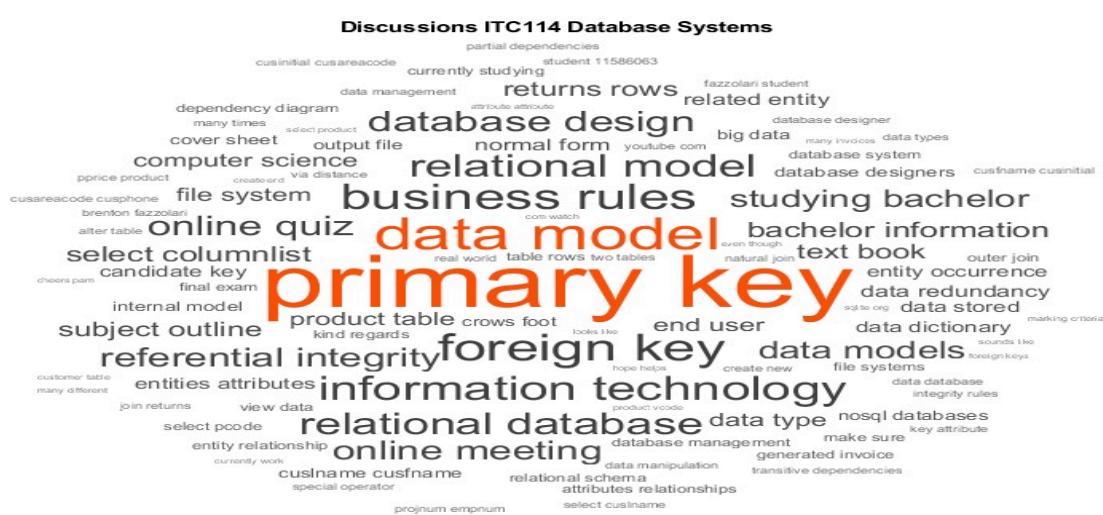


**Figure 10.** Student post days.

Visualization of the patterns in student comments aims to disclose the possible relationships between comments from the time perspective. We can regard the number of comments posted on different weekdays by different students or one student as time-serial data. In other words, these comments contain spatiotemporal information that may be captured by Att-LSTM. This is because the cells in LSTM can remember information over time intervals and use their gates to regulate the flow of information. However, we cannot see the obvious connections between the comments posted based on time and the results of Att-LSTM in our experiments. This may be because the dataset is not big enough or topics of comments do not have the spatiotemporal characteristics that are useful for Att-LSTM.

### 5.2. Keywords in Student Comments

The word cloud of keyword frequencies in the comments is shown in Figure 11.



**Figure 11.** Keyword frequencies.

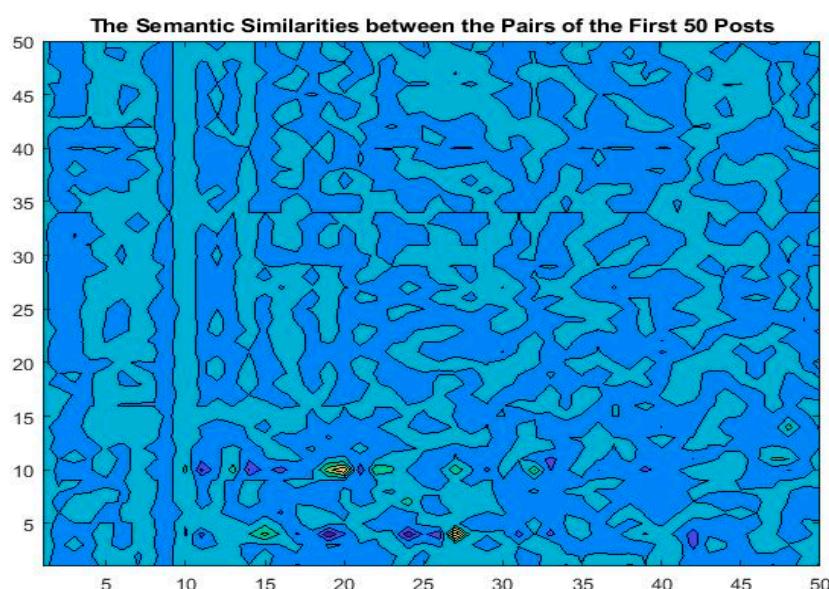
This cloud is based on counting word pairs (bigrams) in comments. The more frequent the pairs, the bigger the sizes of their display in the cloud visualization are. The top 10 bigrams in our dataset are given as follows:

From Table 4, we could see that the most important and difficult concepts related to the subject content have appeared in the student comments. We compare this visualization results with the keywords used in category results from the machine learning algorithms. We find that the word cloud is largely consistent with those by NB but little with Att-LSTM. For example, the category of the *concept* identified by NB shares the highest number of the same keywords in the visualization. This may be because NB uses individual keywords as the feature for clustering labels without considering the semantic relationships between keywords. In this sense, it is similar to the word cloud that is based on the occurrence frequency of keywords in all the comments from a particular category.

**Table 4.** Top 10 word-pairs.

N-Gram		Count
primary	key	66
data	model	33
foreign	key	27
business	rules	25
information	technology	24
relational	model	22
relational	database	21
database	design	20
online	quiz	19
online	meeting	18

Finally, we visualize the semantic similarities between the pairs of the first 50 comments in ITC-114 in Figure 12. For this, we first regard the first 50 comments as a collection and represent it as a matrix where each row is the TF-IDF scores of its keywords of a comment. Second, the similarity scores of all the comments (rows or columns) are calculated by using the pairwise cosine seminaries. As shown in Figure 12, the visualization demonstrates that there are some degrees of similarities among the comments. This fact partially explains why learning algorithms can detect the categories of the comments. Further, we examine the similarity visualization of each category and the results by the compared algorithms. Some keywords in the clustered detected by Att-LSTM appear in the areas with relatively high similarity scores in the visualization. However, other algorithms produce the results that are marginally similar to the visual presentation.



**Figure 12.** The Similarities between the pair of the first 40 comments in ITC-114 datasets.

After comparing what the algorithms produce, we conclude that results from some algorithms and visualization are consistent (and symmetry) to some extent. Further, we first divided a teaching session into several stages, such as introduction, assignment, online meeting, and exam. By companion, we then find that the extracted topics from the messages posted at the same stages were, almost, a reflection symmetry to those from different sessions. This is also reflected in Figure 12 with some symmetry. This is because the data has not only the regularizes, but also some symmetry.

This work implies that we should consider actionable insight from the results of the algorithms and visualization. Actionable insight is the result of data-driven analytics of patterns that occur in student comments. By analyzing the comments—important data regarding online students—the lecturers can develop an understanding of students' needs and expectations. More importantly, we can make data-informed decisions. On Mondays and Tuesdays, we may, for example, post our questions and read student comments on discussion boards, according to the identified categories by the algorithm. This comprehensive data analysis will shed light into optimal ways of creating meaningful learning experiences for students.

## 6. Conclusions

Online comments from students can provide an effective way of teaching and receiving feedback. How to automatically extract insightful information from these comments are important. In this work, we rely on machine learning algorithms to extract topic keywords from student comments to predict their topics. By doing so, we can summarize the information of online subject posts for improving the quality of learning and teaching in higher education.

In this paper, we presented the results of the compared performances of five algorithms given the same task of extracting topic keywords from student comment dataset. The selected methods ranged from traditional approaches to deep learning algorithms. They were compared against the same dataset and evaluated by the five metrics. From our experiments, our conclusions are as follows. The performance of the compared statistical algorithms, though being trained faster depends on the selected features. Deep learning algorithms achieved great accuracy, but with more training time. Of the two deep learning approaches compared, Att-LSTM performs the best in terms of all the metrics used. A combination method may perform better than any single approach due to overcoming limitations. Moreover, the quality of the dataset affected the results.

During the past decade, machine learning algorithms have demonstrated promising performances in a wide range of application areas. However, they are still limited in educational settings, particularly for higher education. In universities, many text tasks are usually time-consuming. Machine learning algorithms can automatically assist in completing these tasks, as demonstrated in this work.

**Author Contributions:** Conceptualization, X.H., W.H., and S.X.D.; experiments and original draft preparation on machine learning algorithms, F.L.; ITC-114 dataset preparation and draft preparation on visualization, S.X.D. and X.H.; project supervisor, X.H.; review and editing, W.H., X.H., and S.X.D. All authors have read and agreed to the published version of the manuscript.

**Funding:** F.L. was partially supported by the Scientific and Technological Research Program of Chongqing Municipal Education Commission (project number: KJ1600512).

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Nair, C.S. Evaluation of Subject, Teaching and Research. In Proceedings of the Higher Education Research and Development Society of Australia Conference, Peth, WA, Australia, 7–10 July 2002; pp. 481–489.
2. Ilias, A.; Hasan, H.F.A.; Rahman, R.A.; Yasoa, M.R.B. Student Satisfaction and Service Quality: Any Differences in Demographic Factors? *Int. Bus. Res.* **2008**, *1*, 131–143. [[CrossRef](#)]
3. Palshikar, G.K. Keyword Extraction from a Single Document Using Centrality Measures. In Proceedings of the International Conference of Pattern Recognition and Machine Intelligence, Kolkata, India, 18–22 December 2007.

4. You, R.; Huang, X.; Zhu, S. DeepText2Go: Improving large-scale protein function prediction with deep semantic text representation. *Methods* **2018**, *145*, 82–90. [[CrossRef](#)] [[PubMed](#)]
5. Diyanati, A.; Sheykhhahmadloo, B.S.; Fakhrahmad, S.M.; Sadredini, M.H.; Diyanati, M.H. A proposed approach to determining expertise level of StackOverflow programmers based on mining of user comments. *J. Comput. Lang.* **2020**, *61*, 101000. [[CrossRef](#)]
6. Rose, J.D.; Dev, D.D.; Robin, C.R. An Improved Genetic Based Keyword Extraction Technique. In *Nature Inspired Cooperative Strategies for Optimization (NISCO 2013)*; Springer: Cham, Germany, 2014; pp. 153–166.
7. Wang, X.; Tang, X.; Qu, W.; Gu, M. Word sense disambiguation by semantic inference. In Proceedings of the 2017 International Conference on Behavioral, Economic, Socio-cultural Computing (BESC), Krakow, Poland, 16–18 October 2017; pp. 1–6.
8. Gollapudi, S.; Panigrahy, R. Exploiting asymmetry in hierarchical topic extraction. In Proceedings of the 15th ACM International Conference on Information and Knowledge Management, Arlington, VA, USA, 5–11 November 2006; pp. 475–482.
9. Turney, P.D. Learning Algorithms for Keyphrase Extraction. *Inf. Retr.* **2000**, *2*, 303–336. [[CrossRef](#)]
10. Witten, I.H.; Medelyan, O. Thesaurus based automatic keyphrase indexing. In Proceedings of the IEEE 6th ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL’06), Chapel Hill, NC, USA, 11–15 June 2006; pp. 296–297.
11. Kadhim, A.I. Survey on supervised machine learning techniques for automatic text classification. *Artif. Intell. Rev.* **2019**, *52*, 273–292. [[CrossRef](#)]
12. Hasan, K.S.; Ng, V. Automatic keyphrase extraction: A survey of the state of the art. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Baltimore, MD, USA, 22–27 June 2014; Volume 1, pp. 1262–1273.
13. Weerasooriya, T.; Perera, N.; Liyanage, S.R. A method to extract essential keywords from a tweet using NLP tools. In Proceedings of the 2016 Sixteenth International Conference on Advances in ICT for Emerging Regions (ICTer), Negombo, Sri Lanka, 1–3 September 2016; pp. 29–34.
14. Liu, F.; Huang, X.; Huang, W. Comparing Machine Learning Algorithms to Predict Topic Keywords of Student Comments. In *Cooperative Design, Visualization, and Engineering. CDVE 2020. Lecture Notes in Computer Science*; Luo, Y., Ed.; Springer: Cham, Germany, 2020; Volume 12341.
15. Zhang, C. Automatic keyword extraction from documents using conditional random fields. *J. Comput. Inf. Syst.* **2008**, *4*, 1169–1180.
16. Salton, G.; Buckley, C. Term-weighting approaches in automatic text retrieval. *Inf. Process. Manag.* **1988**, *24*, 513–523. [[CrossRef](#)]
17. Luhn, H.P. A statistical approach to mechanized encoding and searching of literary information. *Ibm J. Res. Dev.* **1957**, *1*, 309–317. [[CrossRef](#)]
18. Cohen, J.D. Highlights: Language-and domain-independent automatic indexing terms for abstracting. *J. Am. Soc. Inf. Sci.* **1995**, *46*, 162–174. [[CrossRef](#)]
19. Matsuo, Y.; Ishizuka, M. Keyword extraction from a single document using word co-occurrence statistical information. *Int. J. Artif. Intell. Tools* **2004**, *13*, 157–169. [[CrossRef](#)]
20. Ercan, G.; Cicekli, I. Using lexical chains for keyword extraction. *Inf. Process. Manag.* **2007**, *43*, 1705–1714. [[CrossRef](#)]
21. Hulth, A. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*; Association for Computational Linguistics: Stroudsburg, PA, USA, 2003; pp. 216–223.
22. Dennis, S.F. The Design and Testing of a Fully Automatic Indexing-Searching System for Documents Consisting of Expository Text. In *Information Retrieval: A Critical Review*; Thompson Book Company: Washington, DC, USA, 1967; pp. 67–94.
23. Salton, G.; Buckley, C. *Automatic Text Structuring and Retrieval-Experiments in Automatic Encyclopaedia Searching*; Cornell University: Ithaca, NY, USA, 1991. [[CrossRef](#)]
24. Ardimento, P.; Bilancia, M.; Monopoli, S. Predicting bug-fix time: Using standard versus topic-based text categorization techniques. In *International Conference on Discovery Science*; Springer: Cham, Germany, 2016; pp. 167–182.
25. Casalino, G.; Castiello, C.; Del Buono, N.; Mencar, C. A framework for intelligent Twitter data analysis with non-negative matrix factorization. *Int. J. Web Inf. Syst.* **2018**, *14*, 334–356. [[CrossRef](#)]

26. Frank, E.; Paynter, G.W.; Witten, I.H. Domain-Specific Keyphrase Extraction. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence*; Morgan Kaufmann: Stockholm, Sweden, 1999; pp. 668–673.
27. Mitchell, R.S.; Michalski, J.G.; Carbonell, T.M. *An Artificial Intelligence Approach*; Springer: Berlin, Germany, 2013.
28. Zhang, K.; Xu, H.; Tang, J.; Li, J. Keyword extraction using support vector machine. In *The International Conference on Web-Age Information Management*; Springer: Berlin/Heidelberg, Germany; pp. 85–96.
29. Schmidhuber, J. Deep Learning in Neural Networks. *Neural Netw.* **2015**, *61*, 85–117. [CrossRef] [PubMed]
30. Beliga, S. Keyword extraction: A review of methods and approaches. *J. Comput. Sci.* **2014**, *1*–9.
31. Gutwin, C.; Paynter, G.W.; Witten, I.H.; Nevill-Manning, C.G.; Frank, E. Improving browsing in digital libraries with keyphrase indexes. *J. Decis. Support Syst.* **1999**, *27*, 81–104. [CrossRef]
32. Kosovac, B.; Vanier, D.J.; Froese, T.M. Use of keyphrase extraction software for creation of an AEC/FM thesaurus. *J. Inf. Technol. Constr.* **2000**, *5*, 25–36.
33. Jonse, S.; Mahoui, M. Hierarchical document clustering using automatically extracted keyphrase. In Proceedings of the Third International Asian Conference on Digital Libraries, Seoul, Korea, 6–8 December 2000; pp. 113–120.
34. Sarkar, K.; Nasipuri, M.; Ghose, S. Machine Learning Based Keyphrase Extraction: Comparing Decision Trees, Naïve Bayes, and Artificial Neural Networks. *J. Inf. Process. Syst.* **2012**, *8*, 693–712. [CrossRef]
35. Wu, Y.F.; Li, Q.; Bot, R.S.; Chen, X. Domain-specific keyphrase extraction. In Proceedings of the 14th ACM International Conference on Information and Knowledge Management, Bremen, Germany, 31 October–5 November 2005; pp. 283–284.
36. Kohavi, R. Scaling Up the Accuracy of Naïve-Bayes Classifiers: A Decision-Tree Hybrid. *KDD* **1996**, *96*, 202–207.
37. Yasin, U. Keyword Extraction Using Naïve Bayes. In *Bilkent University, Computer Science Dept, Turkey*; 2005; Available online: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.103.2128&rep=rep1&type=pdf> (accessed on 18 October 2020).
38. Kim, S.B.; Han, K.S.; Rim, H.C.; Myaeng, S.H. Some effective techniques for Naïve Bayes text classification. *IEEE Trans. Knowl. Data Eng.* **2006**, *18*, 1457–1466.
39. Tolles, J.; Meurer, W.J. Logistic regression: Relating patient characteristics to outcomes. *JAMA* **2016**, *316*, 533–534. [CrossRef]
40. Tsien, C.L.; Fraser, H.S.; Long, W.J.; Kennedy, R.L. Using classification tree and logistic regression methods to diagnose myocardial infarction. *Medinfo* **1998**, *98*, 493–497.
41. Padmavathi, J. Logistic regression in feature selection in data mining. *Int. J. Sci. Eng. Res.* **2012**, *3*, 1–4.
42. Cortes, C.; Vapnik, V. Support-vector networks. *Mach. Learn.* **1995**, *20*, 273–297. [CrossRef]
43. Igiri, C.P. Support Vector Machine—Based Prediction System for a Football Match Result. *IOSR J. Comput. Eng.* **2015**, *17*, 21–26.
44. Yu, J.X.; Kitsuregawa, M.; Leong, H.V. Advances in web-age information management. In Proceedings of the 7th International Conference, WAIM 2006, Hong Kong, China, 17–19 June 2006; Volume 4016.
45. Isa, D.; Lee, L.H.; Kallimani, V.P.; Rajkumar, R. Text document preprocessing with the Bayes formula for classification using the support vector machine. *IEEE Trans. Knowl. Data Eng.* **2008**, *20*, 1264–1272. [CrossRef]
46. Krapivin, M.; Autayeu, A.; Marchese, M.; Blanzieri, E.; Segata, N. Keyphrases extraction from scientific documents: Improving machine learning approaches with natural language processing. In *International Conference on Asian Digital Libraries*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 2–111.
47. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016; pp. 326–380.
48. O’Shea, K.; Nash, R. An introduction to convolutional neural networks. *arXiv* **2015**, arXiv:1511.08458.
49. Kim, Y. Convolutional neural networks for sentence classification. *arXiv* **2014**, arXiv:1408.5882.
50. Vu, N.T.; Adel, H.; Gupta, P.; Schütze, H. Combining recurrent and convolutional neural networks for relation classification. *arXiv* **2016**, arXiv:1605.07333.
51. Wang, J.H.; Liu, T.W.; Luo, X.; Wang, L. An LSTM Approach to Short Text Sentiment Classification with Word Embeddings. In Proceedings of the 30th Conference on Computational Linguistics and Speech Processing (ROCLING 2018), Hsinchu, Taiwan, 4–5 October 2018; pp. 214–223.
52. Yin, W.; Kann, K.; Yu, M.; Schütze, H. Comparative study of CNN and RNN for natural language processing. *arXiv* **2017**, arXiv:1702.01923.

53. Hughes, M.; Kotoulas, S.; Suzumura, T. Medical Text Classification Using Convolutional Neural Networks. In *Informatics for Health: Connected Citizen-Led Wellness and Population Health*; IOS Press: Amsterdam, The Netherlands, 2017; pp. 246–250.
54. Rodriguez, P.; Cucurull, G.; González, J.; Gonfaus, J.M.; Nasrollahi, K.; Moeslund, T.B.; Roca, F.X. Deep pain: Exploiting long short-term memory networks for facial expression classification. *IEEE Trans. Cybern.* **2017**. [[CrossRef](#)]
55. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [[CrossRef](#)] [[PubMed](#)]
56. Wang, Y.; Huang, M.; Zhao, L. Attention-based LSTM for aspect-level sentiment classification. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Austin, TX, USA, 1–5 November 2016; pp. 606–615.
57. Wang, Y.; Zhang, J. Keyword extraction from online product reviews based on bi-directional LSTM recurrent neural network. In Proceedings of the 2017 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM), Singapore, 10–13 December 2017; pp. 2241–2245.
58. Mnih, V.; Heess, N.; Graves, A. Recurrent models of visual attention. *arXiv* **2014**, arXiv:1406.6247.
59. Zhou, P.; Shi, W.; Tian, J.; Qi, Z.; Li, B.; Hao, H.; Xu, B. Attention-based bidirectional long short-term memory networks for relation classification. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), Berlin, Germany, 7–12 August 2016; pp. 207–212.
60. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Polosukhin, I. Attention is all you need. *arXiv* **2017**, arXiv:1706.03762.
61. Hu, D. An introductory survey on attention mechanisms in NLP problems. In *Proceedings of SAI Intelligent Systems Conference*; Springer: Cham, Switzerland, 2019; pp. 432–448.
62. Cotterell, R.; Schütze, H. Morphological word embeddings. *arXiv* **2019**, arXiv:1907.02423, preprint.
63. Levy, O.; Goldberg, Y. Neural word embedding as implicit matrix factorization. In Proceedings of the 27th International Conference on Neural Information Processing Systems, Montreal, QC, Canada, 8–13 December 2014; pp. 2177–2185.
64. Ganguly, D.; Roy, D.; Mitra, M.; Jones, G.J. Word embedding based generalized language model for information retrieval. In Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, Santiago, Chile, 9–13 August 2015; pp. 795–798.
65. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Commun. ACM* **2017**, *60*, 84–90. [[CrossRef](#)]
66. Helmis, M.; Hollmann, R. *Web-Based Data Integration, Approaches to Measuring and Securing the Quality of Information in Heterogeneous Data Sets Using a Fully Web-Based Tool*, 1; Vieweg Verlag: Wiesbaden, Germany, 2009.
67. Huang, X.; Lai, W. Clustering graphs for visualization via node similarities. *J. Vis. Lang. Comput.* **2006**, *17*, 225–253. [[CrossRef](#)]
68. Huang, X.; Huang, W. GO: A cluster algorithm for graph visualization. *J. Vis. Lang. Comput.* **2015**, *28*, 71–82. [[CrossRef](#)]
69. Liao, X.; Huang, X.; Huang, W. Visualization of farm land use by classifying satellite images. In *International Conference on Cooperative Design, Visualization and Engineering*; Springer: Cham, Germany, 2018; pp. 287–290.
70. Seipp, K.; Gutiérrez, F.; Ochoa, X.; Verbert, K. Towards a visual guide for communicating uncertainty in visual analytics. *J. Comput. Lang.* **2019**, *50*, 1–18. [[CrossRef](#)]
71. Yoo, S.; Ryu, H.R.; Yeon, H.; Kwon, T.; Jang, Y. Visual analytics and visualization for android security risk. *J. Comput. Lang.* **2019**, *53*, 9–21. [[CrossRef](#)]
72. Angelini, M.; Bonomi, S.; Lenti, S.; Santucci, G.; Taggi, S. MAD: A visual analytics solution for Multi-step cyber Attacks Detection. *J. Comput. Lang.* **2019**, *52*, 10–24. [[CrossRef](#)]
73. Du, Y.; Yin, H.; Wang, C.; Li, C. Visual analysis of customer switching behavior pattern mining for takeout service. *J. Comput. Lang.* **2020**, *57*, 100946. [[CrossRef](#)]

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).