Journal of Universal Computer Science, vol. 24, no. 11 (2018), 1604-1626
submitted: 28/2/18, accepted: 7/9/18, appeared: 28/11/18 © J.UCS

Mining of Educational Opinions with Deep Learning

Ramón Zatarain Cabada
(Tecnológico Nacional de México/ I.T. Culiacán, Mexico
rzatarain@itculiacan.edu.mx)

María Lucía Barrón Estrada
(Tecnológico Nacional de México/ I.T. Culiacán, Mexico
lbarron@itculiacan.edu.mx)

Raúl Oramas Bustillos
(Tecnológico Nacional de México/ I.T. Culiacán, Mexico
raul.oramas@itculiacan.edu.mx)

Abstract: This paper describes the process of creating an opinion-mining module that uses deep learning techniques to detect the positive or negative polarity of students' opinions regarding the exercises they solve in an intelligent learning environment (ILE) for the Java language, as well as the detection of learning-centered emotions such as engagement, boredom, and frustration. The information serves as the basis for administrators and teachers who use the ILE to analyze the opinions in order to improve the pedagogy of the ILE exercises. To determine the effectiveness of the deep learning model, we carried out experiments with ten different architectures using the Yelp dataset and one of its own named SentiText containing 147,672 and 10,834 balanced sentences, respectively. We obtained encouraging results with a model that combines a Convolutional Neural Network and a Long Short-Term Memory with an accuracy of 84.32% and an error rate of 0.24 for Yelp and 88.26% and an error rate of 0.33% for SentiText.

Keywords: Deep Learning, Sentiment Analysis, Opinion Mining, Intelligent Learning Environments
Categories: I.2.6, I.2.7, L.3

1 Introduction

The growth of Web 2.0 has changed the way young people generate and consume information. For example, Web applications such as Facebook, Instagram, Twitter, to name a few, have changed the way young people communicate with each other. Companies such as Amazon, Netflix, and Uber have modified the consumption habits of young people by guiding their purchasing decisions based on the opinion of other people on these platforms. From the commercial, industrial or academic point of view, the information obtained from the opinions generated in these platforms allows, for example, marketing managers to make decisions to change the strategy of an advertising campaign, or university professors to improve the strategies of teaching-learning of their courses.

Studying and analyzing this information in the traditional way, with data analysts or specialists in the field, is a complicated task due to the large amount of information

which must be processed. With Mining of Opinions (Opinion Mining, OM), it is possible to develop software applications that automatically classify the opinions of users and help data analysts with their tasks. OM is one of the applications of natural language processing (NLP), whose objective is the evaluation and classification of the text from an emotional point of view; for example, classify a sentence with a positive, neutral, or negative polarity. OM in learning environments has been used to automatically assess the opinions of students helping to understand the needs of students to improve the teaching-learning process. Students' opinions may contain significant ideas to help teachers incorporate changes to improve their course material, strategies and other teaching elements. An Intelligent Learning Environment is an educational setting that integrates modules to manage affect, sentiment analysis or other intelligent elements.

In this paper, we describe the development of a module to perform OM using Deep Learning (DL) techniques. The OM module is integrated into an Intelligent Learning Environment (ILE) named Java Sensei [see Zatarain-Cabada et al. 2015], for learning the Java programming language. One of the main features of Java Sensei is that it has a recognizer of facial emotions that provides the ability to use emotions as part of the pedagogical strategies. The purpose of performing OM in Java Sensei is to automatically evaluate the opinions of students regarding the programming exercises that are presented during the interaction with the ILE. With the analysis of opinions, teachers can evaluate the quality of the exercises and incorporate significant improvements to them.

The rest of the document is organized as follows: Section 2 presents a review of the literature on OM and DL. Section 3 describes the methodology used to develop the OM module. Section 4 shows the integration of the OM module within Java Sensei. In section 5 the experiments and results are presented. Finally, Section 6 presents conclusions and future work.

2 Related work

OM, also known as Sentiment Analysis (SA), is the computational study of people's opinions, attitudes, and emotions [Liu and Zhang 2012]. OM is considered a part of NLP, whose task is to find opinions, identify sentiments within a phrase or sentence, and classify them according to a positive or negative polarity or emotions [see Pang and Lee 2004] and [Medhat et al. 2014].

There is a lot of research work as described in [Liu and Zhang 2012], [Medhat et al. 2014], and [Kharde and Sonawe 2016] in the area of Machine Learning (ML) applied to OM tasks. In recent years, as reported in [Singhal and Bhattacharyya, 2016] and [Zhang et al. 2018], DL techniques are applied to OM producing promising results compared to traditional ML techniques. Even though the advances with both techniques are in constant improvement and evolution, in the end it is possible to combine both techniques to produce better results. Perhaps one of the most important reasons why DL techniques are becoming popular has to do with the availability of large amounts of data, the evolution of hardware with accessible prices and the availability of frameworks in different programming languages in order to carry out these tasks. Another reason why DL has become popular is that these techniques have won many victories in important competitions related to ML. However, a big

difference between ML and DL techniques is that with ML you have to analyze the relationships or characteristics between the data you want to work with using statistical techniques, a process known as feature engineering. On the other hand, with DL the algorithms are responsible for automatically discovering the relationships or characteristics between the data.

Below we describe these approaches as well as some research related to OM in educational settings.

2.1 Traditional machine learning in OM

Research conducted regarding OM has produced numerous techniques that include supervised and unsupervised methods for classifying text. Different surveys performed by [Pang and Lee 2004], [Medhat et al. 2014] and [Tsytsarau and Palpanas 2012] have performed an extensive review regarding machine learning techniques, including supervised and unsupervised methods.

The supervised methods use Support Vector Machine (SVM), Maximum Entropy, Naive Bayes, among others. Unsupervised methods include methods that exploit lexicons for sentiment analysis, grammar analysis, and syntactic patterns. The results obtained with these techniques are acceptable although there are specific difficulties related to subjective interpretation and other problems that affect the polarity of the words.

2.2 OM in e-learning environments

In relation to OM techniques in e-learning environments, the use of DL has not been fully explored, to the best of our knowledge. Rather, previous work has been focused towards traditional ML techniques. In [Chaplot et al. 2015] the authors present an algorithm based on an ANN (Artificial Neural Network) to predict student dropout in MOOCs using sentiment analysis and showing the importance of students' feelings in this work. In [Kumar and Jain 2015] student opinions are collected in the form of running text, and sentiment analysis is performed to identify important aspects of the opinions using supervised and semi-supervised ML techniques. In the work of [Dhanalakshmi et al. 2016] the authors use OM with supervised ML algorithms of type SVM (Support Vector Machine), Naïve Bayes, KNN (nearest neighbors) and ANN to find the polarity of the comments of students based on the predefined characteristics of teaching and learning. The study carried out involves the application of a combination of ML and NLP techniques in the student feedback data. The results were compared to find the best performance with respect to several evaluation criteria for the different algorithms. In [Rani and Kumar 2017] they explore NLP and ML techniques based on student comments to help university administrators and teachers address areas of problems in teaching and learning. The system they developed analyzes student comments in online course surveys and sources to identify the polarity of feeling, expressed emotions and student satisfaction. They also made a comparison with the results of the direct evaluation to determine the reliability of the system.

2.3 OM and DL techniques

In recent years, deep learning has emerged as a trend, which is used in many applications that perform complex operations such as speech recognition systems, facial recognition or objects, to mention a few examples. The term deep learning was used for the first time in [Hinton et al. 2006] and is based on the work of Kunihiko Fukushima who proposed in the 80s an algorithm known as *neocognitron* where several networks were used to emulate the functioning of primary visual cortex neurons.

The limitations in the hardware for many years prevented the popularization of DL methods. However, in recent years, the increase in computational power combined with the maturity of software frameworks (e.g. Keras, TensorFlow, Theano, Caffe, Pytorch, CNTK, MXNet, Torch, deeplearning4j), has generated promising results as reported in [Zhang et al. 2018], and the interest of working with DL has grown significantly.

To work OM with DL, it is necessary to have a corpus or dataset of phrases that are related to a particular domain (in our particular case, opinions about online programming courses), perform some normalization process on these data and convert them to a format usable by the DL algorithms.

In [Wang et al. 2015] the authors used an LSTM (Long Short-Term Memory) model for the prediction of sentiment on Twitter. In their experiments they compare different models such as SVM, Bernoulli Naive Bayes and RNN (Recurrent Neural Networks) in different configurations. The corpus used was the Stanford Twitter Sentiment corpus (STS). In their experiments they obtained 87.2% accuracy with better results than the other models. In [Wang et al. 2016] the authors propose a CNN-LSTM model consisting of two parts: regional CNN (Convolutional Neural Network) and LSTM, to predict evaluations of texts with a valence scale. The regional CNN, unlike a conventional CNN that considers a complete text as input, uses individual sentences as regions so that it can be weighted according to the valence scale. The experimental results show that the proposed method outperforms the lexical, regression-based and ANN-based methods proposed in previous studies. In [Qian et al. 2016] the authors proposed a simple trained model with annotations at the sentence level, and also a model for the linguistic role of the lexical feelings with LSTM, the words of negation and the words of intensity. The results show that the models are able to capture the linguistic role of words with feeling, words of denial and words of intensity in sentimental expression. In [Nguyen et al. 2017] the authors used a lexical-based approach to apply semantic rules and used a DeepCNN with pre-trained vectors at the character level to capture morphological information of each word to determine how those words are formed and their relationship among the others. In this work, they report an accuracy of 86.63%.

2.3.1 Datasets and word embeddings

NLP has been applied with different levels of success in the past. Currently, with the widespread use of social networks, users generate large amounts of information. This information can be used as the basis to create a corpus of data in a specific domain and train DL algorithms to perform automated predictions.

DL algorithms receive a vector of real numbers as input. To perform the representation of the words in a vector space (mapping of words to numbers), techniques such as Word Embeddings are used. Word Embeddings is a technique of translating words into a mathematical domain where numbers that try to capture the semantics of the word represents words. This process is done automatically using millions of phrases. The most common encoding is “one-hot encoding” [Bengfort and Kim 2016] but other forms such as Word2Vec as described in [Mikolov et al. 2013] and GloVe [Pennington et al. 2014] have also been used.

The model One-hot encoding, represents each word with a certain id, with a vector full of zeros except on the id position where it contains a one. It is an easy method to use but it has the problem of high dimensionality, since the dimension of the Vectors is equal to the size of the vocabulary. Usually, this model produces overfitting problems and is computationally expensive. In addition, it does not provide true information about relationships between words.

The Word2Vec and GloVe models (also known as Skip-gram and CBOW), are based on unsupervised ML methods that, based on large volumes of text, achieve syntactic and semantic representations.

2.3.2 Convolutional Neural Network (CNN)

CNN's are a special type of multilayer neural networks that were introduced in 1995 as cited in [Hinton et. al. 2006]. CNN's are similar to neural networks since neurons have weights, biases, receive an input with which they make a scalar product and apply an activation function, in addition to having a loss function. One of the problems with neural networks is that the greater number of hidden layers increases the number of computational resources that are needed and overfitting is easily obtained.

With CNN's it is possible to face this problem since a CNN divide and model the information in smaller parts, then combining this information in the deepest layers of the neural network.

2.3.3 Recurrent Neural Networks (RNN)

RNNs introduce the concept of persistence in memory as described in [Elman 1990]. Traditional neural networks do not have this capability. Using the concept of persistence we can predict (infer) some type of event based on previous events. The RNNs are networks with cycles (or loops) that allow the information to persist. For example, in backward propagation networks, the processing of information is transported through the network from the input layers to the output layers. In the case of an RNN, the connection between the neurons forms cyclical routes.

Back-propagation neural networks accept a fixed size input and produce a fixed size output vector. On the other hand, the RNNs operate on vector sequences and have no restrictions on the size of the sequences. Each sequence that is received as input, updates the hidden states according to the previous calculations. Because an RNN processes input sequences of one element at a time, they maintain in their hidden units a "state vector" that contains the information related to the history of all the elements before the sequence so it acts as a unit of memory. This feature makes

them effective to solve problems such as recognition of discourses, modeling of languages, translations, capturing images, among others.

2.3.4 RNN Long Short-Term Memory (LSTM)

One of the applications of RNNs is to guess the next word that is in a certain context. As information increases, RNNs take longer to learn to connect information. To solve this problem an LSTM network is used.

The influence of a given input on the hidden layer, and therefore on the output of the network, deteriorates or grows exponentially as it propagates through an RNN. This is known as Vanishing Gradient Problem. One way to solve this problem is with an LSTM neural network [Hochreiter and Schmidhuber 1997]. The LSTM network is a special type of RNN architecture, capable of learning long-term dependencies.

2.4 OM in learning environments

Feedback is a very valuable element of information in Virtual Learning Environments such as the ILE Java Sensei, as they can help to understand the needs of students and improve the teaching-learning process as mentioned in [Rowe, 2017], [Poulos et al. 2008], and [Cummins et al. 2010]. This feedback contains open opinions of students and may contain significant ideas for teachers to incorporate changes and improve their course material, learning strategies and other academic elements.

In [Dhanalakshmi et al. 2016] the authors constructed a tool to analyze the publications in community forums of students and teachers, with the aim of systematically studying the opinions expressed there. A similar work [see Altrabsheh et al. 2013] is based on supervised learning algorithms to find the polarity of student comments based on the predefined characteristics of teaching and learning. There are also research works related to massive open online courses (MOOCs) using the discussion forums to monitor the opinions of students towards the contents of the courses as described in [Wen et al. 2014]. An interesting proposal consists of an architecture to analyze student comments using sentiment analysis techniques for Twitter [Abdelrazeq et al. 2016]. SentBuk [Ortigosa et al. 2014] is a Facebook application that retrieves the messages written by users on Facebook and classifies them according to their polarity, showing the results to users through an interactive interface.

3 Methodology

To find the best DL model it is important to establish a series of steps that allow us to explore different configurations, such as fully-conected networks, convolutional, recurrent, and dropout layers, weight initialization and its activation functions, learning rate with optimizers such as stochastic gradient descent, Adam, as well as other hyper-parameters. It is also important to establish the size of the batch, the number of epocs and the evaluation metric for the DL model. [Figure 1] shows the methodology used to explore the different models of DL. The following subsections describe the general tasks that were performed to create, configure, train and evaluate these models.

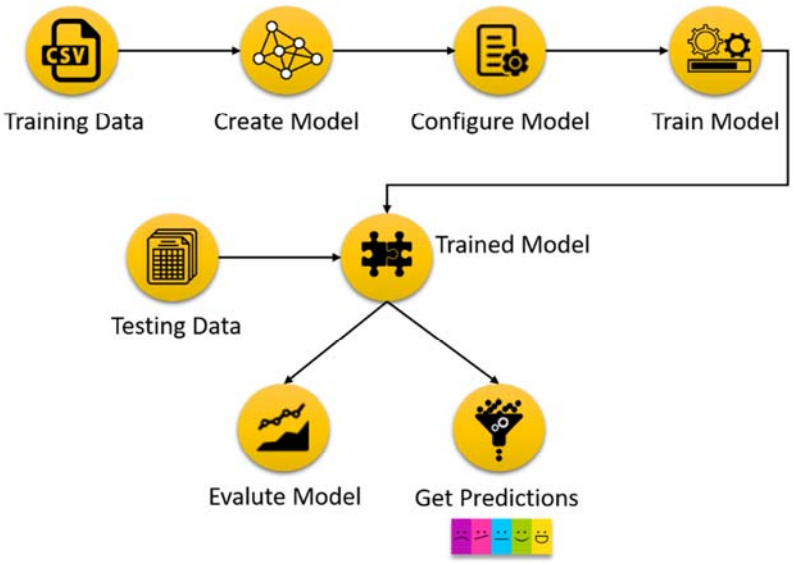

Figure 1: The methodology used to test the DL models.

3.1 The dataset for training

In this research work, we used the Corpus Yelp [yelp.com, 2018]. Yelp is an open data corpus containing more than 158,000 restaurant reviews. Two corpora were also built with opinions related to programming languages. The corpus, called SentiText, is oriented towards the polarity of opinions, classifying them as positive or negative. For the construction of the corpus, we first used an application to extract the opinions of Twitter with a geographical delimitation to the northwest of Mexico with the following keywords: programmer, developer, teacher, student, Java, and Python. A group of professors of programming languages classified these opinions manually. Due to the size of the corpus, it was divided into small sections and each teacher labeled the part assigned to him. The initial size of this corpus was 4164 opinions labeled as positive and 4248 as negative. Since DL models need a big amount of data, the size of this dataset was artificially increased in size using a simple data augmentation technique. This method consisted of adding new words or phrases to a certain opinion, modifying some of them with synonyms of small semantic variations to obtain 13010 phrases: 7593 phrases corresponding to positive labels and 5417 corresponding to negative labels.

We implemented a web application for a corpus oriented towards learning centered opinions [Barrón-Estrada et al. 2017]. The name of the corpus is EduEras (Educational Resources Assessment System). One of the main objectives of the corpus ERAS was to gather opinions from students to produce a new corpus of Spanish opinions. The corpus or dataset has 7386 comments labeled with a learning-centered emotion such as frustrated (2082), bored (1361), neutral (945), excited (1697), and engaged (1301).

It is important to mention that with the traditional techniques of ML, the feature selection is very important, since depending on the selection of these features we can arrive at different results. In the case of DL techniques, the most important thing is to have a reliable and balanced corpus so that the algorithms will discover the patterns among those data. What is more important in the DL techniques is that from these patterns, it is the DL model who is learning the relationships and the rules between these data so that in the end, after training the model, it is able to classify a text with a certain emotion.

3.2 Pre-processing, dataset, and training

The pre-processing of the data is the process of cleaning and preparing the dataset for later use [Haddi et al. 2013]. The pre-processing allows the normalization of the dataset and helps improve the performance of the neural network models. In most OM problems, it is important that the DL model receive as input a clean corpus instead of a noisy one. A noisy corpus refers to text with punctuation marks, numerical values, links and URLs, etc. The elimination of these elements in the text would increase the DL model precision. However, it is essential to keep in mind that these elements, such as the emoticons ":)", ":(", and ":-P", are important for the classification of the sentiment since they indicate an emotional charge for the text [Bao et al. 2014]. The basic pre-processing tasks performed for this research work consisted of:

a) Punctuation marks. It is common for users to omit or abuse some punctuation marks such as the period '.', comma ',', question mark '?', exclamation point '!' among others, for various reasons. All punctuation marks are removed by using regular expressions.
b) Most common abbreviations (slang terms). It is common for users to include abbreviations for their opinions. Given a dictionary of slang terms, they are translated to preserve the grammatical content of an opinion. For example, the term “KO” is replaced with the term “I am dead” and “a2” with “goodbye”.
c) Interjections normalization. Regular expressions are also used to normalize elements such as 'hahaha', 'jajjjaja', 'jijiji' as they cause dispersion problems.
d) Emoticons replacement. As in the case of interjections, users often express their moods using emoticons with different connotations. A dictionary is also used to identify and replace the most frequent emoticons. For example, the emoticon “:)” translates to its equivalent in the text "happy".
e) Link simplification. Often an opinion includes a link to external resources such as images or addresses to other Web pages. Regular expressions are used to detect those links, as they do not represent relevant information.
f) Stopwords. These are words whose presence does not provide information (they do not have an associated emotional charge) to the classification process. Words such as articles, pronouns, prepositions, among others are considered stopwords.
g) Stemming. It consists of taking each word to its root in order to group words with similar meaning. For example, the words 'run' and 'running' are treated as equals because their root is the same.

3.2.1 Dataset tokenization

In order for the dataset to be usable by DL models, it is necessary to divide each phrase into its individual components, words or tokens and assign them a numerical format. This process is known as tokenization. This process consists of two steps. The first step calculates the frequency of each word in the dataset to find the most frequent words and assign to the most common word the value of one. Then the second most common word gets the value of two and so on. In this way, we can omit the words that are not so frequent and that do not help the neural network improve its learning process. The second step is to transform the text from the dataset to tokens with its numerical representation that is known as "fitting". This numerical representation is known as a sequence. Once we have prepared the dataset, we can begin to define the DL model.

3.3 Creation of the DL model

A DL model is built adding several layers. Layers are the basic components of a neural network and their task is to process the input data and produce different outputs, depending on the type of layer. For the DL models of this work, we use a sequential model that is a stack of layers for our neural network.

The first layer that makes up each of the models is the embedding layer. This layer allows the network to expand each token to a larger vector, allowing the network to represent the words in a meaningful way. This layer defines the size of the vocabulary to be used and the size of the vector that will be used to expand each token, as well as the length of each of the text sequences. From this layer, it is possible to combine other types of layers to configure the model like:

- Dense layers: also called fully connected layers, since each node in the input is connected to each node in the output.
- Activation layers: includes activation features such as ReLU, tanh, sigmoid among others.
- Dropout layer: used for regularization during training and it helps with the overfitting, especially on dense layers.
- Flatten and Reshape layers: used for flattens the input and reshapes an output to a certain shape.

In addition to these central layers, other important layers:

- Convolution layers: used to perform convolution.
- Pooling layers: used for descending sampling.
- Recurrent layers: connections between units form a directed graph along a sequence.

3.4 Configuration of the DL model

The best practices were taken into account to configure the DL model [deep-learning-nlp-best-practices 2018], although in the field of DL there is still much to discover in relation to this topic. One must consider that one of the vital components of any

neural network are the activation functions. Activations introduce the concept of non-linearity. For many years, sigmoid activation functions were the preferable option. Currently, the activation function tanh, ReLU, and others are also used. The ReLU function, for example, helps with the Vanishing Gradient problem. We used the ReLU function for this project.

To configure the model, the number of neurons must be taken into account. Maintaining a larger number of hidden units is generally a safe bet since any regularization method will deal with superfluous units, at least up to a point. By increasing the number of hidden units, the model will have the flexibility required to filter the most appropriate information.

Another important aspect that was taken into account for the configuration of the model was the learning rate. There are different optimization methods such as Momentum, Adagrad, Adam, RMSProp. For this project, we decided to use the Adam method since it saves us the manual choice of the initial learning rate. The [Table 1] summarizes the configuration of the layers of the ten DL models used. In [Table 1] the input and output layers are not represented. All the models receive as input a vector called one-hot finding where the sentences of the dataset are encoded to a number format so that the algorithms of machine learning do a better job in the prediction. In our models, we use a MAX value to indicate the size of the vocabulary. We will measure the performance of the neural network, the dimension of the feature vector that by default is 128, as well as the length of each sentence, which is 300 characters for experiment 1 and 160 characters for experiment 2.

Model Name	**Layer Type**					
	Dense	**Conv.**	**Pooling**	**Flatten**	**DropOut**	**LSTM**
MLP	1	---	---	1	---	---
CNN	1	1	1*	1	2	---
CNN 2	1	3	3*	1	4	---
CNN 3	1	1	1**	---	1	---
LSTM	---	---	---	---	1	1
CNN + LSTM	---	1	1*	---	2	1
CNN 4	1	3	---	1	3	---
LSTM_2	---	---	---	---	1	1
CNN + LSTM 2	1	4	4*	---	5	1
CNN 5	1	1	4*	1	5	---

** Max Pooling, ** Global Max Pooling*

Table 1: Deep Learning Models Summary.

Model 1 (MLP) aims to explore the operation of a single-layer neural network. The simple multilayer Perceptron is suitable for classification problems such as in OM. For this model, we first apply a flatten layer that converts the input of any dimensionality to an input of 1 x n. Subsequently, a fully connected hidden layer (Dense) with 512 neurons is used and the activation function known as a rectified

linear unit (ReLU) is established. The ReLU rectifier is a common activation function in deep neural networks that allows estimating non-linear relationships in the data. In the output layer, a single neuron is used with a sigmoid-type activation function that will determine the positive or negative polarity of a given text as input.

Model 2 (CNN) configures a neural network of CNN type. CNN networks work by dividing and modeling information into small parts and combining this information into the deepest layers of the network. CNN's have proven successful in several text classification tasks [Kim, 2014]. A regularization layer (Dropout) is added to this model before entering the convolution layer by 50%. The goal of regularization is to omit randomly each hidden neuron with a certain probability, for example 50% (half of the neurons of the hidden layers are not used for each case of the training set) and hidden neurons are prevented from depending (trust) in the work of the other neurons of the same layer. In this way, it is possible to avoid over-training (overfitting). Following, a convolution layer is added. One of the characteristics of a CNN is its convolution operation that receives as input, in this case a phrase and then applies a filter or kernel that returns a map of the features of the original text. The Conv1D layer creates the convolution kernel. The size of the output filter in the convolution is 128, the size of the kernel is 5 and the activation type is ReLU. For the displacement (stride), we use the value that is defined by default. Then we add the reduction layer (Pooling). The objective of this layer is to reduce the spatial dimensions of the input data for the next layer. The operation that was used for this layer is max-pooling, which divides the input phrase into a set of rectangles and, with respect to each sub-region, it is left with the maximum value. The size of that rectangle or window is equal to five. At the end of the convolution and pooling layers, fully connected layers (dense) are generally used. A dense layer with 128 neurons and with the activation function of type ReLU was used. The output layer is the same as in the previous model.

Model 3 (CNN 2) is a variation of model 2 with the difference that in this model three convolutional layers are used instead of one. The size of the filters is smaller and the size of the window increases in each convolutional layer.

Model 4 (CNN 3) is a CNN network that uses a layer GlobalMaxPooling1D instead of MaxPooling1D. The GlobalMaxPooling layer tries to minimize overfitting. The difference between the two layers is that in the GlobalMaxPooling1D layer the size of the filter is equal to the size of the input and this consumes many computational resources at the time of training the model.

Model 5 (LSTM) defines an RNN network of type LSTM with a simple configuration. The goal of this model is to explore the operation of this type of network with simple parameters. It also uses a DropOut layer to turn off and turn on 20% of the neurons to prevent the neural network from overfiting.

Model 6 (CNN + LSTM) combines a CNN and a LSTM network. The goal is to explore the operation of this type of architecture with a simple configuration. In the model, two Dropout layers are used to turn off and turn on 50% of the neurons, where the convolutional layer is configured with a filter equal to 64, a kernel size of 5, and an activation function of type ReLU. The pooling layer is MaxPooling type with a pool of size 4. The LSTM layer has 128 memory units, which represent the size of the vector of embedded words.

The goal of **model 7 (CNN 4)** is to explore a CNN model with multiple layers but without using MaxPooling1D. We wanted to observe the behavior of the three convolutional layers with different sizes in their filters (300, 150, and 75 respectively) to experience how the network learns from those specific sizes. After this process, the data produced in the previous layers are moved to a fully-connected neural network (named *Dense* layer) with a size of 150 neurons.

The aim of **model 8 (LSTM_2)** is to configure a model to explore an RNN neural network with an LSTM type to explore a feature called "regularizers" in addition to another type of optimizer such as ada-delta. The goal is to observe the behavior of the model with 256 learning units for the LSTM layer with an activation function and learning rate different from the other models.

Model 9 (CNN + LSTM 2) is a variation of model 6 but using more convolutional layers with their respective pooling layers. For each convolutional layer the kernel size is increased by one unit (3,4,5). The size of the filters (32) is the same for the three convolutional layers. After going through the CNN layers, the data will be trained with an LSTM layer with 256 memory units and finally the data will move to a fully-conected or dense neural network with 512 neurons.

Model 10 (CNN 5) is a variation of model 9 but without using LSTM. The goal is to observe if there are differences in the behavior of this model with or without an LSTM layer.

3.5 Training and evaluation of models.

For this research project, three corpora were used: Yelp, SentiText and EduSere. First, the Yelp corpus was used to train and test the ten models, verifying that the different configurations used produce consistent results. That is, a model with more layers, with more neurons should produce better results than a simpler model. Secondly, the same tests were carried out with the SentiText corpus in order to verify that the results are consistent in relation to the tests carried out with the corpus Yelp, although SentiText has a lower number of sentences. The EduSere corpus was not used for this validation task but was trained with the best model that was obtained. [Table 2] shows the total number of positive and negative classes used for the first two corpus.

Corpus	Positive (+)	Negative (-)	Total
Yelp	73836	73836	147672
SentiText	5417	5417	10834

Table 2: Corpus Distribution.

To evaluate the effectiveness of the models, we use the measures of loss (Loss) and precision (Accuracy). The lower the loss, the better the model will be (unless the model has been adjusted too much to the training data). The loss is not in percentage compared to the accuracy and is a sum of the errors made for each epoch in the training or validation sets. The accuracy of a DL model is usually determined after the parameters of the model are learned and corrected, and nothing is being learned. Accuracy measures the misclassification percentage of the model. For example, if the number of samples of the test is 1000 and the model classifies 952 of them correctly,

then the model's accuracy is reported as 95.2%. To obtain the loss and precision values, 67% was used for training data and 33% for the test data of the total dataset. The selection of these values depends, most of the time, on the available amount of data. Usually the training and test data are divided into a range of proportions (percentages) of 75:25, 80:20 or 90:10, but there is no rule that indicates which proportion produces the best results, since frequently the best choice is determined based on trial and error. All experiments were executed using a balanced dataset.

One of the problems of DL is overfitting, where the model fits very well with existing data but has the poor performance to predict new results. To avoid this problem and obtain the best DL model, the early-stopping technique was used, which evaluates the error of the network after each epoch and stops training once the loss of validation has not decreased during a fixed number of epochs (epochs). In this way, we avoid overfitting and make sure we get the best model.

3.5.1 Training and evaluation 1.

To train and evaluate the models, we first decided to work with the Yelp dataset, with variations in the vocabulary size of 20000, 40000 and 50000 out of a total of 103842 words, setting the learning process (gradient descent) to a batch size of 32 and 64 respectively. [Table 3] shows the results obtained for a batch size equal to 32.

	Vocabulary					
Model	**20000**		**40000**		**50000**	
	Loss	**Acc**	**Loss**	**Acc**	**Loss**	**Acc**
Model 1	0.68	90.75	0.71	91.05	0.60	90.42
Model 2	0.36	89.58	0.28	89.96	0.28	89.89
Model 3	0.30	89.13	0.33	89.51	0.30	88.58
Model 4	0.35	92.50	0.65	91.40	0.49	92.46
Model 5	0.41	91.66	0.31	91.53	0.32	90.92
Model 6	**0.27**	**90.56**	**0.24**	**91.77**	**0.24**	**92.15**
Model 7	0.46	81.82	0.42	84.07	0.39	85.12
Model 8	8.01	50	8.01	50.0	8.01	50
Model 9	0.27	91.01	0.28	90.46	0.25	90.21
Model 10	0.29	89.84	0.27	89.84	0.31	89.76

Table 3: Results obtained with different size of vocabulary and batch size equal to 32

From the results of [Table 3], it is observed that model 4 and model 6 obtain better results with the different variations in the vocabulary but model 6, which corresponds to a combination of CNN with LSTM, has a lower loss value, indicating that it is the best model evaluated.

	Vocabulary					
Model	**20000**		**40000**		**50000**	
	Loss	**Acc**	**Loss**	**Acc**	**Loss**	**Acc**
Model 1	0.60	91.09	0.72	90.39	0.61	90.48
Model 2	0.30	89.42	0.32	89.71	0.29	89.89
Model 3	0.30	90.49	0.37	86.66	0.30	87.82
Model 4	0.37	93.29	0.55	91.71	0.41	92.46
Model 5	0.39	91.41	0.33	89.29	0.35	90.92
Model 6	**0.26**	**92.40**	**0.24**	**91.70**	**0.24**	**92.20**
Model 7	0.36	85.02	0.42	84.99	0.36	86.66
Model 8	8	50	8.0	50	8.0	50
Model 9	0.28	89.54	0.27	89.83	0.24	91.21
Model 10	0.26	90.63	0.28	89.39	0.27	90.67

Table 4: Results obtained with different size of vocabulary and batch size equal to 64

[Table 4] shows the results obtained with a batch size of 64. In this case, model 4 and model 6 also show identical results, so we can determine that the results obtained are consistent. [Figure 2] shows the architecture of model 6, which illustrates the following: the text "i like this exercise" is transformed into a vector of words that will be the input for the DL model. In the first stages, a convolutional layer processes the word vector and its features are extracted in the pooling layer (Max-pooling); after this, a one-dimensional vector is obtained. A LSTM layer processes this one-dimensional vector and finally a layer of fully connected neurons returns a positive or negative polarity or an emotion.

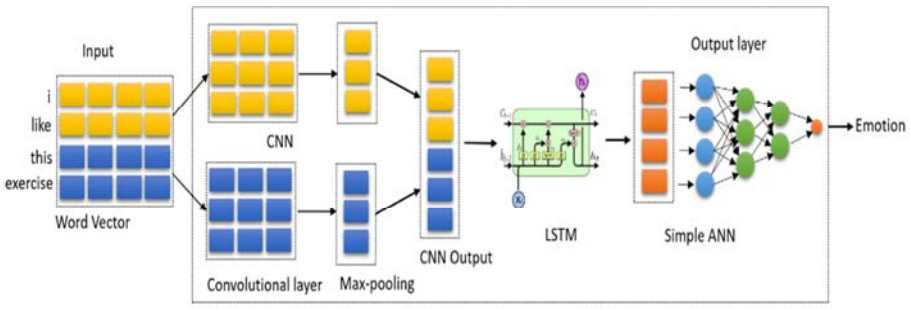

Figure 2: Model 6: DL with CNN and LSTM.

3.5.2 Training and evaluation 2.

Once we found the model that produced the best results (model 6 CNN + LSTM) for the Corpus Yelp and SentiText, another type of tests were performed using the SentiText and EduSere corpus. The goal was to find the best model but eliminating stopwords from phrases and performing stemming to try to determine if these

elements affected the evaluation of model 6. We used 75% of the vocabulary (7086 words) out of a total of 9284 for dataset SentiText y 4000 words out of a total of 4300 for EduERAS with a batch size equal to 32. [Table 5] shows the results obtained.

Model 6	Remove Stopwords	Remove Stemming	Loss	Acc
SentiText	**False**	**False**	**0.33**	**88.26**
	True	False	0.48	85.30
	True	True	0.42	84.32
EduERAS	**False**	**False**	**0.49**	**90.30**
	True	False	0.53	89.47
	True	True	0.86	73.88

Table 5: Results obtained with model 6 and batch size equal to 32

The results presented in [Table 5] show that eliminating stopwords and performing a stemming process on the dataset of phrases does not improve the evaluation of the model since the DL techniques try to recognize patterns. By eliminating those characteristics, the context of each phrase is reduced, causing the DL model to be difficult to learn. It is important to mention that the evaluation of model 6 with the SentiText dataset (10834 sentences) gets lower values compared with the Yelp dataset (147,672 sentences) because the dataset is smaller, but it can be inferred that if we increase the size of the dataset, the evaluation of the model should improve, as shown in [Table 5]. For the particular case of the EduERAS dataset, the results obtained are not consistent because the size of the dataset is not large enough.

4 Integration of the DL model with the ILE Java Sensei

In this section, we describe the integration of the DL module to the ERAS environment as well as ILE Java Sensei.

4.1 The DL for Opinion Mining

The [Figure 3] shows the general scheme for the module that performs opinion mining to detect emotions in the text. Recognized emotions are learning oriented (boring, frustrated, neutral, excited, and engaged) and the module is part of an Intelligent Learning Environment named Java Sensei to learn the Java language.

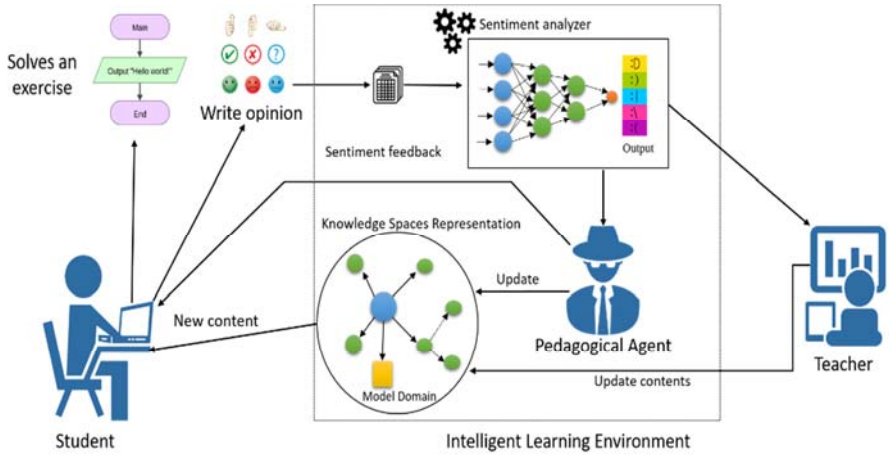

Figure 3: Opinion mining module.

The flow of information is as follows: the student solves a programming exercise provided by the ILE Java Sensei and issues an opinion on that exercise. This opinion is the input to the module that conducts the opinion mining (sentiment analyzer). For the detection of emotions a DL model is used (described in the previous section) with its own corpus in the domain of programming languages. A pedagogical agent uses the detected emotion to perform two activities; first, send a personalized feedback to the student based on the emotion detected; second, make decisions and present the contents of the student's domain model based on those emotions. The teacher will also use the information provided by the opinion mining module to make decisions regarding the educational content that can be improved. The domain model contains expert knowledge that the student wants to learn and practice. Knowledge representation is implemented with knowledge space theory [Doignon and Falmagne 2012]. The expert model represents six basic skills that students must master. The knowledge is modeled by a graph representing the knowledge space. The basic skills are introduction to Java, variables and calculations, selection, iteration, methods, and arrays.

4.2 Integration of the DL module to the ERAS environment

ERAS was designed to interact with students in order to allow them to express their opinions and comments freely about the educational resources or learning objects of the subjects of a course.

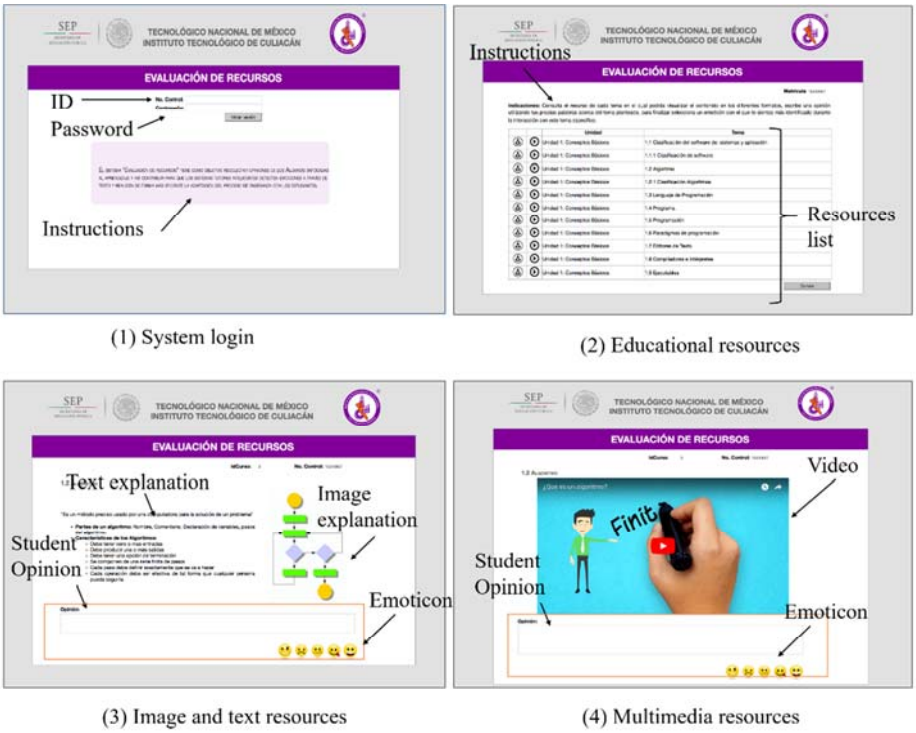

Figure 4: Main ERAS interfaces.

ERAS contains several interfaces, but there are four main GUIs shown in [Figure 4]. The login interface shows a brief introduction of the objective of the system and allows users to login [see Figure 4 (1)]. In [Figure 4 (2)] the second interface presents the topics of the subject selected by the student; there are two different formats (Image-Text and Video) to show the subject of study. [Figure 4 (3)] shows the learning object selected in format text, image or both, and finally, in [Figure 4 (4)], the theme in video format is shown. In both formats, the student is requested to enter a sentence to express an opinion about the learning object and tag it with an emotion related to learning (frustrated, bored, neutral, excited and engaged) using an emoticon provided in the interface.

4.3 Integration of OM module to Java Sensei

Once we find the best DL model to perform educational OM, the next step is its integration into the learning environment Java Sensei, which, as we mentioned before, is an environment for learning the Java programming language. The learning environment was built almost entirely with Java technology.

On the other hand, the DL model was developed or implemented as a module with the Keras Python library, so to achieve the integration of this module with Java

Sensei, we use the Pyro library (Python remote object) which is a middleware for remote communication of objects.

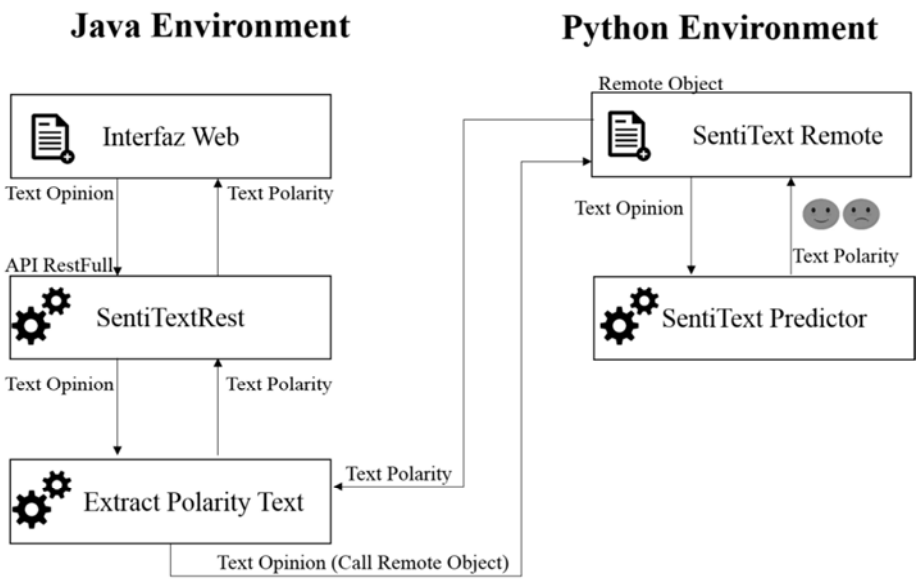

Figure 5: Integration of the DL module into Java Sensei.

[Figure 5] illustrates how this process is completed. First Java Sensei shows an exercise to the student. The student solves the exercise and at the end of this activity, the system asks him to write an opinion of fewer than 160 characters on the exercise. Then, Java Sensei uses the SentiTextRest service to send the student's opinion via a remote call to the SentiTextRemote module. The remote module that is in the Python environment sends the student's opinion to the predictor module SentiText Predictor that evaluates the text and returns the positive or negative polarity of the same. Then, the result is returned by going through the previous modules until it reaches the Web Interface. The result is returned to the call made by the REST service of the ILE and stored in the opinion database for further evaluation by teachers and course administrators.

5 Evaluation of the OM module inside the ILE

In this section, we present the results obtained for the ILE Java Sensei with the SentiText dataset oriented to polarity and the ERAS platform using the EduEras dataset focused on educational emotions.

5.1 OM module evaluation on ERAS

In August of 2017, a preliminary experiment was carried out, with 53 students from the area of computer systems engineering (45 men and 8 women) at the Instituto tecnológico de Culiacán. This complementary activity would be used to weight the final grade of the student.

The content of each sub-topic of the ERAS platform was presented in text, text and images format or in the video (multimedia) and the student is asked to write their opinion about it. The student was also asked to label that opinion with an emoticon that represented an emotional state: bored, frustrated, neutral, excited or engaged. The goal of the student expressing his opinion using an emoticon was to have another measuring parameter in relation to the OM module. At the end of the experiment, 851 opinions were recorded.

Subsequently, in January 2018, we used the OM module to validate the precision of the newly created dataset. There were 458 matches (emotion labeled by student vs OM evaluation), which turned out to be 53.81% from the 851 opinions. [Table 6] shows some results obtained.

Opinion (Spanish)	**Opinion (English)**	**Emotion (labelled by student)**	**OM Evaluation**
Me gusto bastante el vídeo.	I really liked the video.	Engaged	Engaged
No me gusto que las voces fueran de España.	I did not like that the voices were from Spain.	Frustrated	Bored
El video es bueno aunque creo que le falto profundizar más.	The video is good although I think it needs to go deeper.	Excited	Neutral
Vaya, es algo complejo.	Wow, it's something complex.	Engaged	Bored
Quizás con un ejemplo quedaría más claro.	Perhaps with an example, it would be clearer.	Neutral	Excited

Table 6: Example of student opinions.

The results obtained are not the best. Currently, we are working to increase the size of the dataset as well as to obtain a balanced dataset, which will allow obtaining better results.

5.2 OM module evaluation inside an ILE

In a first approach, the OM Module trained with SentiText and model 6 was tested with 43 students in our research lab. Students must solve several exercises to learn and practice the topics of Java language. Each lesson has 15 exercises. At the end of each exercise, the system asks for student opinions. At the end of the test, we

collected 200 students' opinions. [Table 7], shows some examples of collected opinions.

The OM module evaluated the opinions of the students, where we obtained that 169 texts were evaluated and 31 were wrong, which gives us an accuracy of 84.50%. Course administrators and teachers will keep these results for further study and it will help to improve the learning material of the ILE.

Opinion (Spanish)	**Opinion (English)**	**Polarity**	**Evaluation**
No entendí este ejercicio	I did not understand this exercise	Negative	Correct
Me gusto este ejercicio	I liked this exercise	Positive	Correct
El ejercicio es confuso	This exercise is confusing	Negative	Correct
Este ejercicio es sencillo de entender	This exercise is easy to understand	Negative	Incorrect

Table 7: Results obtained in the student evaluation.

6 Conclusions and Future Work

In this work, we have described a general framework to integrate an OM module into an ILE and ERAS environment. Evaluation results give us information to improve learning material and programming exercises in both environments. For this Project, we have created a dataset with opinions related to programming languages labeled with polarity (positive and negative) and another dataset with opinions labeled with emotions centered on learning (bored, frustrated, neutral, excited, and engaged). Once we have created our two datasets, we compare the results of different models applying DL with the Yelp dataset, and with our two datasets (SentiText and EduERAS).

With these datasets, we experimented with 10 different models with variations in the type of layers to configure a multilayer neural network, a CNN, an LSTM, as well as the training of each model with different hyper-parameters. After extensive testing, the best model found was one that combined a CNN network with an LSTM. The results of the tests shown in [Section 3.5.2] in our two datasets were promising (88.26% in SentiText and 90.30% in EduERAS). On the other hand, these results in field tests with students were promising in the case of recognition of polarities (84.50%) and not good in the case of recognition of emotions (53.81%). For this last case (emotions), we are working on increasing the number of opinions of the dataset (EduERAS). This will produce better results since DL requires large datasets to produce good results [Lecun et al. 2015].

For future work, we will carry out the following tasks:

1. Testing the OM module with other learning environments which include exercises for students.
2. Improving the corpus to include more phrases related to programming languages and the different areas of computing science.
3. Implementing an emotion recognition for dialogs in the ILE Java Sensei using the new corpus that includes labels for text emotion.

4. Testing new approaches for feature creation (word2vec, GloVe) that can produce better results.

Acknowledgments

The work described in this paper was fully supported by the TECNM (Tecnológico Nacional de México) and CONACYT (Consejo Nacional de Ciencia y Tecnología) in México.

References

[Abdelrazeq et al. 2016] Abdelrazeq, A., Janßen, D., Tummel, C., Jeschke, S., & Richert, A.: Sentiment Analysis of Social Media for Evaluating Universities. In Automation, Communication and Cybernetics in Science and Engineering 2015/2016 (pp. 233-251). Springer, Cham, 2016.

[Altrabsheh et al. 2013] Altrabsheh, N., Gaber, M., & Cocea, M.: SA-E: sentiment analysis for education. In International Conference on Intelligent Decision Technologies (Vol. 255, pp. 353-362), June, 2013.

[Bao et al. 2014], Bao, Y., Quan, C., Wang, L., & Ren, F. The role of pre-processing in twitter sentiment analysis. In International Conference on Intelligent Computing (pp. 615-624). Springer, Cham.

[Barrón-Estrada et al. 2017] Barrón-Estrada, M.L., Zataraín-Cabada, R., Oramas-Bustillos, R., Ramírez-Ávila,S.L. Building a Corpus of Phrases Related to Learning for Sentiment Analysis. Research in Computing Science 146: 17-26 (2017)

[Bengfort and Kim, 2016] Bengfort, B., & Kim, J. Data Analytics with Hadoop: An Introduction for Data Scientists. " O'Reilly Media, Inc.".

[Chaplot et al. 2015] Chaplot, D. S., Rhim, E., and Kim, J. (2015). Predicting student attrition in moocs using sentiment analysis and neural networks. In Proceedings of AIED 2015 Fourth Workshop on Intelligent Support for Learning in Groups.

[Cummins et al. 2010] Cummins, S., Burd, L., & Hatch, A.: Using feedback tags and sentiment analysis to generate sharable learning resources investigating automated sentiment analysis of feedback tags in a programming course. In Advanced Learning Technologies (ICALT), 2010 IEEE 10th International Conference on (pp. 653-657). IEEE, July, 2010.

[Dhanalakshmi et al. 2016] Dhanalakshmi, V., Bino, D., & Saravanan, A. M. (2016). Opinion mining from student feedback data using supervised learning algorithms. In Big Data and Smart City (ICBDSC), 2016 3rd MEC International Conference on (pp. 1–5).

[deep-learning-nlp-best-practices 2018], Web Page, Available: http://ruder.io/deep-learning-nlp-best-practices/, 2018

[Doignon & Falmagne 2012] Doignon, J.P., Falmagne, J.C., 2012 Knowledge Spaces. Springer Science & Business Media.

[Elman 1990] Elman, J. L.: Finding structure in time. Cognitive science, 14(2), 179-211, 1990.

[Haddi et al. 2013] Haddi, E., Liu, X., & Shi, Y.: The role of text pre-processing in sentiment analysis. Procedia Computer Science, 17, 26-32, 2013.

[Hinton et al. 2006] Hinton, G. E., Osindero, S., & Teh, Y. W.: A fast learning algorithm for deep belief nets. Neural computation, 18(7), 1527-1554, 2006.

[Hochreiter and Schmidhuber 1997] Hochreiter, S., & Schmidhuber, J.: LSTM can solve hard long time lag problems. In Advances in neural information processing systems (pp. 473-479), 1997.

[Kharde and Sonawe 2016] Kharde, V., Sonawane, P.: Sentiment Analysis of Twitter Data: A Survey of Techniques, Int. J. Comput. Appl., vol. 139, no. 11, pp. 975–8887, 2016.

[Kim 2014] Kim, Y.: Convolutional Neural Networks for Sentence Classification, in Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1746–1751, 2014.

[Kumar and Jain 2015] Kumar, A., & Jain, R. (2015). Sentiment analysis and feedback evaluation. In MOOCs, Innovation and Technology in Education (MITE), 2015 IEEE 3rd International Conference on (pp. 433–436).

[LeCun et al. 2015] LeCun, Y., Bengio, Y., & Hinton, G.: Deep learning. Nature, 521(7553), 436, 2015.

[Liu and Zhang 2012] Liu, B., & Zhang, L.: A survey of opinion mining and sentiment analysis. In Mining text data (pp. 415-463). Springer US, 2012.

[Medhat et al. 2014] Medhat, W., Hassan, A., & Korashy, H.: Sentiment analysis algorithms and applications: A survey. Ain Shams Engineering Journal, 5(4), 1093-1113, 2014.

[Mikolov et al. 2013] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J.:Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems, pp. 3111-3119, 2013.

[Nguyen et al. 2017] Nguyen, H., & Nguyen, M.-L. (2017). A Deep Neural Architecture for Sentence-level Sentiment Classification in Twitter Social Networking. https://doi.org/10.1016/B978-0-444-51747-0.50005-6.

[Ortigosa et al. 2014] Ortigosa, A., Martín, J. M., & Carro, R. M.: Sentiment analysis in Facebook and its application to e-learning. Computers in Human Behavior, 31, 527-541, 2014.

[Pang and Lee 2004] Pang, B., & Lee, L.: A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In Proceedings of the 42nd annual meeting on Association for Computational Linguistics (p. 271). Association for Computational Linguistics, July 2014.

[Pennington et al. 2014] Pennington, J., Socher, R., & Manning, C: Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP) (pp. 1532-1543), 2014.

[platzi.com 2018] platzi.com, Web Page, Available: https://platzi.com/fundamentos-programacion/, 2018.

[Poulos et al. 2008] Poulos, A., & Mahony, M. J.: Effectiveness of feedback: The students' perspective. Assessment & Evaluation in Higher Education, 33(2), 143-154, 2008.

[Qian et al. 2016] Qian, Q., Huang, M., Lei, J., & Zhu, X. (2016). Linguistically Regularized LSTMs for Sentiment Classification. https://doi.org/10.18653/v1/P17-1154.

[Rani and Kumar 2017] Rani, S., & Kumar, P. (2017). A sentiment analysis system to improve teaching and learning. Computer, 50(5), 36–43.

[Rowe 2017] Rowe, A. D.: Feelings about Feedback: The Role of Emotions in Assessment for Learning. In Scaling up Assessment for Learning in Higher Education (pp. 159-172). Springer, Singapore, 2017.

[Singhal and Bhattacharyya 2016] Singhal, P., Bhattacharyya, P.: Sentiment Analysis and Deep Learning: A Survey, 2016.

[Tsytsarau and Palpanas 2012]. Tsytsarau, M., & Palpanas, T. Survey on mining subjective data on the web. Data Mining and Knowledge Discovery, 24(3), 478-514.

[Wang et al. 2015] Wang, X., Liu, Y., SUN, C., Wang, B., & Wang, X. (2015). Predicting Polarities of Tweets by Composing Word Embeddings with Long Short-Term Memory. Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), 1, 1343–1353. Retrieved from http://www.aclweb.org/anthology/P15-1130.

[Wang et al. 2016] Wang, J., Yu, L.-C., Lai, K. R., & Zhang, X. (2016). Dimensional Sentiment Analysis Using a Regional CNN-LSTM Model. Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), (January), 225–230. https://doi.org/10.18653/v1/P16-2037

[Wen et al. 2014] Wen, M., Yang, D., & Rose, C.: Sentiment Analysis in MOOC Discussion Forums: What does it tell us?. In Educational data mining 2014.

[Yelp Dataset 2018] yelp.com, Web Page, Available: https://www.yelp.com/dataset/challenge, 2018.

[Zatarain-Cabada et al. 2015], Zatarain-Cabada, R., Barrón-Estrada, M.L., González-Hernández, F., & Oramas-Bustillos,R., "An Affective Learning Environment for Java," 2015 IEEE 15th Int. Conf. Adv. Learn. Technol., pp. 350–354, 2015.

[Zhang et al. 2018] Zhang, L., Wang, S., & Liu, B.: Deep Learning for Sentiment Analysis: A Survey, Jan. 2018.