EasyChair Preprint

№ 15704

Grading Write-Ups Using Deep Learning

Siladitya Mukherjee, Avijit Chakraborty and Anal Acharya

January 13, 2025

Grading Write-Ups using Deep Learning

Siladitya Mukherjee[1], Avijit Chakraborty[1], and Dr. Anal Acharya [1]

[1] St. Xavier's College (Autonomous), 30, Park St, Mullick Bazar, Park Street area, Kolkata, West Bengal 700016

siladitya.mukherjee@sxccal.edu,
avijitchakraborty1998@gmail.com, anal.acharya@sxccal.edu

Abstract. In most of the countries, the quality of education in the urban areas and the rural areas have huge gaps. As a result, the evaluation of a piece of text, specifically handwritten text is very subjective and at times can be biased. So, this project aims to build an unbiased grading system using Deep Learning and Transfer Learning. The aim of the paper is to show - how Transfer Learning in the form of pre-trained word embeddings can be introduced with Deep Learning, a regression model to predict a score accurately on a scale of 1 – 10 and to compare the performance of the model on real world data. This paper shows an approach to Automated Essay Grading using Deep Learning and Transfer Learning. The aim of the paper is to show how Transfer Learning in the form of pre-trained word embeddings can be introduced with Deep Learning; A regression model to predict a score accurately on a scale of 1 – 10; and to compare the performance of the model on real world data.

Keywords: Deep Learning, Transfer Learning, Automatic Essay Grading, Project Essay Grade, Feature Extraction, Exploratory Data Analysis, Data Cleaning, ReLU

1 Introduction

Evaluating a write-up using a rule-based approach is a painful task. Essays play an important role in today's world – from academic examinations to job applications. Essays are an integral part of various application processes. Hence a real time evaluation system of Essays will be of great use. Human language is diverse, with different dialects, language understanding, meaning etc. To get a chance to solve this task, a model is needed to be trained about syntax, semantics, including certain facts about the world. Given enough data, many parameters, and enough computation, a model can do a reasonable job. Hence, this paper attempts to show how Deep Learning can be used to solve this task and create a grading system. This will reduce the time spent on evaluation of essays and the user can spend time in improving the wordings and the writeup rather than spending time on evaluating how well written the essay is.

Natural language processing is a powerful tool and has a wide range of applications. From Chat-bots to analyzing social Media Posts. While applying NLP in everyday we frequently come across tasks which endure from data shortfall and inadequate model generalisation. Transfer learning resolved this problem by letting us to take a pre-trained model of a task and employ it elsewhere. This paper shows how pre-trained embeddings can be used in creating efficient solutions in the field of Natural Language Processing.

2 Related Literature Survey

Automated Essay Grading (AES) is a very important problem statement and has been studied and researched for years. The first attempt was made by Ellis Batten Page in 1968 when his work was published with a program defined as Project Essay Grade (PEG)

[1] that could possibly grade essays. PEG was eventually sold to Measurement Incorporated as computerized text scoring would have been costly at that point.

In 1982, with the advancement of computers a UNIX program called Writer's Workbench offered spelling, punctuation, and grammar recommendation.[2] Intelligent Essay Assessor (IES) created by Peter Foltz and Thomas Landauer in 1997 was benefited to attain essays in undergraduate courses. [3] Numerous such rule-based evaluation systems were used in the coming years.

In 2012, Automated Student Assessment Prize (ASAP), a competition sponsored by the Hewlett Foundation sponsored a competition on Kaggle. The challenge was to predict using automated essay scoring and demonstrate its reliability. Over 201 challenge participants tried to predict, and the team that won retrieved a Kappa Score of 0.81407. [4]

The above works are based on models built on predefined features. Two Researchers at Stanford, Shihui Song and Jason Zhao has worked on a Machine Learning approach on AES.[5] Another paper from Stanford by Huyen Nguyen and Lucio Dery shows a Deep learning approach on AES and have achieved a highest Kappa Score of 0.94.[6]

This paper shows an approach to Automated Essay Grading using Deep Learning and Transfer Learning. The aim of the paper is:

- To show how Transfer Learning in the form of pre-trained word embeddings can be introduced with Deep Learning.
- A regression model to predict a score accurately on a scale of 1 - 10.
- To compare the performance of the model on real world data.

3 Materials and Methods

3.1 DEEP LEARNING OVER TRADITIONAL MACHINE LEARNING

A. Feature Extraction

Grading a write-up depends on many features which can be impossible to extract manually and feed into a machine learning model. Deep learning takes the step of feature extraction away and learns the features implicitly while training based on the data fed.

B. Transfer Learning

“Transfer Learning is a research problem in ML that focuses on storing knowledge gained while solving one problem and applying it to a different but related problem." NLP is a potent tool, however

in real-life we frequently derive tasks that tolerate data shortfall and inadequate model generalisation. Transfer learning resolved this challenge by permitting us to take a pretrained model of a task and use it elsewhere utilizing Deep Learning.

3.2 PREPARING THE DATA

The Dataset used is an open-source dataset released by Hewlett Foundation. The dataset contains columns viz:

- Essay_set - defining the set of essays it belongs to
- Essay - the essays that were written by students.
- Rater1_domain1 - grading from a rater
- Domain1_score - the cumulative score of that essay

A. Exploratory Data Analysis

The dataset has 12977 essays hand-graded by two raters and a combined score is being taken as the domain1_score which is the final score.

The dataset has 8 types of essay set, and each is being graded on a different scale. Some of the essays have a low score of 2,5 and some even have a score of 40. So, there is no uniform grading scale, and this is since each set has a different grading scale. The plot below shows the high variation in the scores present in this dataset. Maximum essays are being graded in the 0-10 range.

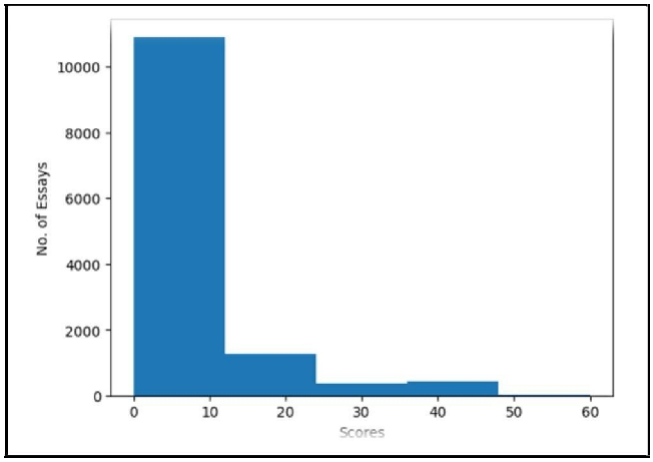

Fig. 1. Plot of actual scores vs number of essays

The maximum score of each essay set have been calculated, using the “do- main1_score” column. Each essay score was divided with the maximum for their respective essay set. In this way all the scores were scaled down to a per- centile and by multiplying this with 10 a uniform grading scale of 0-10 respectively can be constructed. The marks distribution after scaling them to 0-10 looks much more uniform, this marks distribution is being used forward.

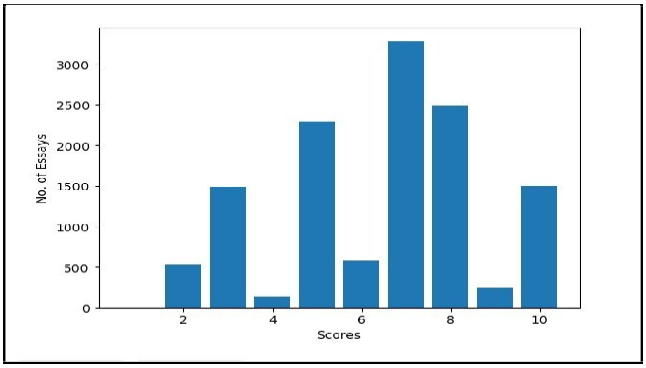

Fig. 2. Plot of normalized scores vs number of essays

A. *Data Cleaning*

Data Cleaning is an essential step in Machine Learning since the value of data and the beneficial evidence that can be obtained from it openly marks the capability of our prototypical to learn; thus, it is awfully significant that we clean our data before feeding it into our model. The steps taken to clean the data is mentioned below:

- Certain words are found in the format like - “donÕt , andÉ”. The Unicode data is normalised in Python to remove umlauts, accents etc.
- Converted to lowercase letters to maintain uniformity of the data. This will re- duce any unnecessary features of the dataset, the model will see all the words in lower case thus every word will be treated with the same weightage in terms of font size, case. Following this all sorts of non-alphabetic characters have been removed so that the words can be matched to their proper embeddings.
- The essays that are present in the dataset has been structured in a way that proper nouns like names, places, organizations have been replaced with placeholders like “@ORGANIZATION1”, “@ORGANIZATION2”, “@NAME1” etc.
  While pre-processing the data these placeholders have been replaced with words like “names”, “organization” and placeholders like “@NUM” have been replaced with “number”.

B. *Splitting the Dataset*

An important step of the data pre-processing step is splitting the dataset into three parts - Training, Testing and Validation.

- Testing - The part of the dataset that is used to train the model
- Validation - The part of the dataset that is used to check whether the training process is overfitting or not.
- Testing - The part of the dataset that is used to evaluate the model post the training process.

Here, 80% was used for Training and 10% + 10% was used for validation and testing.

C. *Tokenization and Padding*

To convert each sentence to computer understandable format “tokenization” is being done where every word is being associated with a number. The first step of tokenization is to create a vocabulary of words from the training dataset or the section of the dataset that will be used for training the model. Each word is being replaced with its respective index and an array of numbers is

obtained.

Tokenization is followed by Padding where all the essays of shorter length are being padded with 0 and longer texts are truncated to make all training input of equal size. A Tokenized and Padded sequence looks like this:

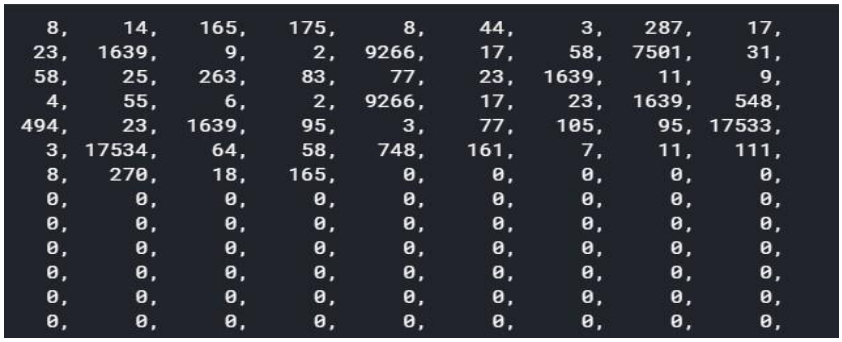

Fig. 3. Padded and Tokenized essay

3.3 WORD EMBEDDINGS & TRANSFER LEARNING

Converting the sentences into a sequence of integers is not enough. Machine learning or deep learning algorithms perform matrix operations within it so it's important that every input data is being presented in the vectorized format. Hence it demands the string/text need to be transformed into a set of real numbers (a vector) — Word Embeddings.

Thus, Word Embeddings or Word Vectorization is an attempt in Natural Language Processing to represent vocabulary or words from lexis to a matching vector of real numbers that are managed to recover word predictions, word similarities/semantics. The process of transforming words into numbers is called Vectorization.

In a word embedding every word can be represented in a N dimensional vector space. Thus, each word embedding is a data point in that vector space and words of similar sentiment are close to each other i.e. the Euclidean distance between those vectors is very less. Thus, word embeddings map human language in the geometric space. There are various ways to create word embeddings. One such popular and most widely used way is by using an "Embedding Layer".

Transfer Learning and Pre-trained embeddings

The outstanding success of pretrained language prototypes is astonishing. One motive for the success of language modelling could be that it is an incredibly challenging task, yet for individuals. To have any probability at resolving this task, a model is expected to be taught about syntax, semantics, including certain facts about the globe. Given enough data, many parameters, and enough calculations, a model can do a sensible job. Here, the pretrained GloVe word embeddings have been used.

According to the Stanford website [6] – "GloVe is an unsupervised learning algorithm for obtaining vector representations for words. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space."

These word embeddings are better than the trained word embeddings because these embeddings are trained on millions and millions of texts. Hence, they can capture the context of a particular text better than trained embeddings. This will be a helpful while grading any write-up.

4 THE MODEL

The Machine Learning Algorithm that has been used is a Neural Network since neural networks is the best suited model for Deep Learning. A sequential mod- el consisting of LSTM (Long Short-Term Memory) and Dense Layers have been used. The model takes in input of (1,300) dimensions in the Embedding Layer and returns a single value from a single neuron Dense Layer.

The activation function used for the dense layers is “ReLu” - a dropout rate of 40% has been used to reduce overfitting. The model was trained for twenty (20) epochs for a batch size of sixty-four (64). The Mean Squared Error loss curve hence obtained after hyperparameter tuning is as follows:

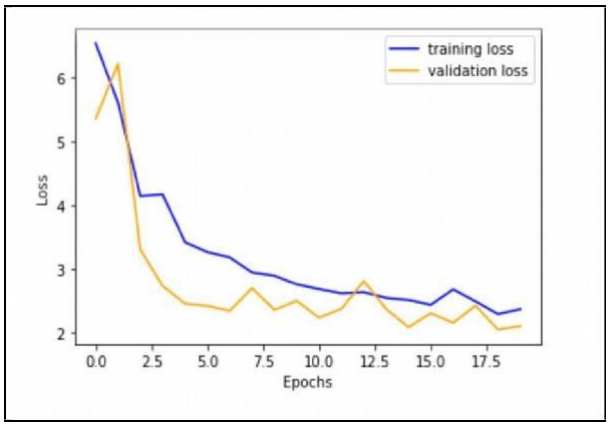

Fig. 4. Plot of Mean Squared Error on validation and training data.

A Mean Squared Error loss value of 1.1897 has been obtained which is subject to research for further improvement.

5 RESULTS

A. Experiment with various Models

Experimentation was done on various models and the best model have been taken for deployment.

Table 1. MODEL ARCHITECTURE VS EMBEDDING SIZE.

MODEL ARCHITECTURE	EMBEDDING DIMENSIONS	SEQUENCE LENGTH	KAPPA SCORE
Embedding+LSTM (2) +Dropout+Dense	**200**	**500**	**0.75**
	200	**200**	**0.76**

Embedding + Dropout + LSTM + Dropout + LSTM + Dropout + Dense			
Embedding+Dropout+ Bi- Directional LSTM + Dropout + Bi-Directional LSTM+Dropout+Dense	**200**	**200**	**0.77**
Embedding+Bi-Directional LSTM+Bi-Directional LSTM+Dense	**200**	**200**	**0.73**

Experiment on Real World Data

The success of a machine learning model depends on how it works on real world data i.e., the data outside this dataset. So, random texts have been taken from various sources and those have been graded using the model. A summary of the results is shown in the table below.

Table 2. II. RESULTS ON REAL WORLD DATA

SL. No	SOURCE	LENGTH OF THE TEXT EXTRACT	SCORE (0 - 10)
1	Some random input	033	**2.9248710**
2	"MEDIUM" BLOG	200	**6.8268423**
3	Scholarship Application	170	**5.8814673**
4	Children's Essay	429	**7.6759730**
5	BLOG	416	**8.2230720**
6	**"Kite Runner" - Story book**	**320**	**8.3782790**

From the experiments, the extract from "Kite Runner" gets the maximum score which is obvious because a writer will write better compared to others. It should also be noted that writeup no five is graded lesser than writeup no. four despite the fact five has a lesser number of words. Also, the Children's Essay is graded more than the scholarship application and writeup number one which has some randomly typed letter without any meaning is graded the least. Thus, the model can detect which is a better style of writing and is grading accordingly. So, we have an automated tutor which can rate what a student type. Thus, it's not that the more you type the more you get marks rather it's that the better you write, the better will be your marks.

6 TABLES

Embedding+Bi- Directional LSTM + Bi-Directional LSTM+Dropout+Dense	**300**	**500**	**0.787**

Embedding+Bi- Directional LSTM + Bi-Directional LSTM+Dropout+Dense	**300**	**1100**	**0.799**

7 CONCLUSION

On a current Kappa Score of 0.787 it is clear that the current model is working well on the real-world data. This model can be deployed as an Flask API (Application Programming Interface) [7] to serve in a model app or web app for the users to use.

Natural Language Processing being a vast domain, it is always open for exploration. This implementation can be extended to a lot of features:

- *Regional Language Support* - Currently, it supports only English language. In the future regional languages grading can be used which will extend its use in rural India.
- *Grading Professional Write-ups* - Currently the model is trained on student essays, we would like to extend it grading recommendation letters, statement of purpose, emails etc which will greatly help all sorts of users.

Acknowledgement

I want to thank my guides Prof. Siladitya Mukherjee and Dr. Anal Acharya for their support and guidance.

References

1. Page, E.B. (1968). "The Use of the Computer in Analyzing Student Essays", International Review of Education.
2. MacDonald, N.H., L.T. Frase, P.S. Gingrich, and S.A. Keenan (1982). "The Writers Work-bench: Computer Aids for Text Analysis", IEEE Transactions on Communications.
3. Rudner, Lawrence. "Three prominent writing assessment programs" Archived 9 March 2012 at the Wayback Machine.
4. The Hewlett Foundation Automated Essay Scoring, Kaggle 2012 - https://www.kaggle.com/c/asap-aes
5. "Automated Essay Scoring Using Machine Learning", Stanford - a paper by Shihui Song Jason Zhao.
6. "Neural Networks for Automated Essay Grading"- a paper by Huyen Nguyen, Department of Computer Science Stanford University and Lucio Dery, Department of Computer Science Stanford University.
7. GloVe: Global Vectors for Word Representation - Jeffrey Pennington, Richard Socher, Christopher D. Manning
8. "Create An API To Deploy Machine Learning Models Using Flask and Heroku" - a medium publication by Elizabeth Ter Sahakyan.