AutoCorrect

# Too many papers!

Correction of papers is a very strenuous and physically tiring task. It requires constant concentration and also a clear conscience and fair hand to mark answers and award a candidate their deserved marks. During Examinations, over thousands of papers are sometimes corrected, with some teachers putting in the effort to correct almost a hundred papers every day. This ardent effort sometimes leads to mistakes creeping in, or a non-standardised method of evaluation, wherein benefits may be given to a student, based on his scores in previous answers. We need to find a way to not only reduce the pressure on teachers during exam sessions but also speed up the process of correction.

# Why AutoCorrect?

The future lies in automation and computers aided with machine learning can solve this very issue quite comfortably. If answer script evaluation can be automated, it will not only lead to a stricter and more standardised as well as a fair method of correction but will also lead to lesser instances of mistakes, or any form of malpractice that is possible on the teacher’s end. It will also help increase the rate of paper correction and lessen the burden teachers have during the exam season. Although this method might not completely replace a teacher’s correction, it will certainly go a long way ahead to digitalise the entire process and may even be a possibility in the near future.

# How does it work?

The core of this application lies in Image Recognition and Natural Language Processing (NLP).

A database for every question paper is stored which contains each question’s model answer, stored along with its comparison model which contains key-value pairs of keywords of the answer mapped to sentences containing these values. This is performed beforehand to construct a database to make this process of correction faster.

## *Image Processing*

The pages of the answer script are scanned and saved as images or PDF. One of the most important parts of the project is to extract the handwritten text from the image or PDF format with as much accuracy as possible. The challenges faced in the whole implementation would be taking care of so many types of handwriting. As the recognition part is very crucial for the answer script correction we need to make sure that it produces as little error as possible. The result given by the model could make a considerable change in the marks received.

The objective here is to allow a student to answer the paper in a conventional way and from the scanned version of the answer script a handwritten character recognition will be followed to capture the data. Since the accuracy rate of the state-of-the-art handwritten character reorganization is not still up to the acceptable level, we propose to apply an error correction mechanism to reduce the errors. The solution does not oppose the age-old convention and affordable as it is mostly a software solution with a minimum hardware requirement.

**Procedure:**

* **Model-1**: Image pre-processing: To deal with the low-quality image, noise in the scan, binarization, alignment, etc. This step will produce an acceptable and processable image form.
* **Model-2**: Image segmentation: In this step, we have to segment into a number of blocks identifying different regions such as computer-printed parts, sketches, computer printed images, and hand-written texts, etc. The blocks containing the hand-written texts are the regions of interest (ROIs). This step will return all the ROIs in the answer script.
* **Model-3**: Hand-written text recognition: The ROIs involving the hand-written answer parts are the input in this step. For each such RoI, we have to extract words in them. Then, in each word, we have to identify the characters in it. Thus, the output in this step will be the character images in each word in each RoI.
* **Model- 4**: Character level recognition: Each character image in the last step, will be processed optically to recognize the character and finally it can be stored in the form of ASCII character. The outcome of this step is the ASCII form of each word in each RoI.
* **Model- 5**: Sloppy handwriting correction: The words are to be processed to check if there are any spelling mistakes or predicting the correct words in the hand-written texts.

## *Natural Language Processing*

Once the text content in each image has been extracted and separated into questions and stored, we need to now score them. To perform this we use concepts of similarity, keyword extraction and information retrieval using various custom models as well as trained word embedding models. Multiple models can be used to extract keywords which may not be picked up by a particular model, hence the models chosen are all based on different algorithms such that every “main key point” in the answer is picked up.

Once this is done, we group together all keywords and use simple pattern matching and string manipulations to group together similar keywords and remove redundancies. These final keywords are then stored in a dictionary, and the entire answer is scanned and the sentences related to these keywords are stored in key-value pairs after checking similarity and distance measures.

Once the model for the test answer is ready, we compare the corresponding key-value pairs in the model answer key along with the submitted answer and perform simple contextual and then semantic matching to find whether both convey the same meaning. Based on the number of matched keywords and the content covered within each keyword match we return a score between 0 to 1 for each key, which is used as a multiplier based on the number of keywords to return the final score of the answer out of the allotted marks for that question. This is performed for all answers and the final score of the entire paper is displayed along with other results.

# Technology Stack

* Python
* Machine Learning - Natural Language Processing and Image Processing
* nltk , spaCy and gensim
* RAKE, PyTextRankA
* Word Embedding models
* Keras
* Tensorflow
* OCR
* OpenCV
* Numpy
* Pandas
* Matplotlib
* Glob
* pytesseract