An Autonomous System to Evaluate Technical Interview Process Using Machine Learning

John Joseph Z, Barath Sripal G, Gaurav M, Dr. S. Balaji

Department Of Computer Science and Engineering
Panimalar Engineering College, Chennai

zjohnjoseph@gmail.com
barathsripal@gmail.com
gaurav151999jr@gmail.com
balajiit@gmail.com

***Abstract*— Deep Learning and Natural Language Processing are used to create this autonomous system. This model is used to automate technical interviews during campus recruiting. The candidate should answer all of the questions asked by an autonomous interviewer online, and the applicant's score and outcome are assessed at the end of the process. This system can also be used to pre screen candidates appearing for interview. The recruiting process is much more effective when this model is used, and the interviewer's time is saved. Since the process has been automated, employees no longer have to conduct technical interviews for candidates, allowing them to be more efficient for the organisation.**

***Keywords*— Deep Learning, Natural Language Processing, Speech Recognition, Universal Sentence Encoder, Transfer Learning, Sentence Similarity, Django**

I. INTRODUCTION

The amount of material available on the Internet has recently increased dramatically. On the Internet, a vast amount of natural language data is added every day. A considerable portion of this data has been formatted as natural language. This necessitates the use of natural language processing (NLP) techniques in order to make use of such a large volume of data. The degree of similarity between sentences is expressed by the degree of probability. Furthermore, in many applications, discovering sentence similarity is a critical problem. In other terms, many speech recognition applications rely on sentence similarities, including semantic search, summarization, question and answer, text recognition, and emotion analysis, as well as falsification. Moreover, the precision with which similarity between sentences is measured is a critical problem. As a result, calculating sentence similarity has gotten a lot of coverage. Many methods for determining sentence similarity have been suggested. Even if the sentences have different words, semantic similarity means that they have similar meaning.

Different methods exist for semantic similarity, which is based on the meaning of sentences. Different methods are used in these approaches to evaluate two sentences semantically. The corpus-based approach seeks word similarity based on statistical analysis of a large corpus. Deep learning can also be used to represent semantics of words by analysing a broad corpus. The knowledge-based approach relies on a handcrafted semantic net for words.

Deep learning, a recent emerging approach, has shown promising results in the field of image processing. As a result, deep learning technology is being adopted by other fields. A learning model for learning word representation forms a broad corpus in Natural Language Processing. Word semantic features are captured in the generated vectors. In the semantic space, similar terms should have closed vectors. Many recent approaches to determining sentence similarity rely on deep learning-generated semantic word representations.

This paper investigates one of the deep learning methods for measuring Sentence and applying it to the development of a website through transfer learning. The paper is organized as follows. The next section provides a quick description of the libraries used in this application. The third section delves into the specifics of the Universal Sentence Encoder. The Transfer Learning Module is also seen in section four. The design of this application is explained in section five. Finally, in section six, the proposal's conclusion is presented.

II. LIBRARIES USED

1. TensorFlow: It is a machine learning library that is free to use and open-source. It can be used for a variety of machine learning tasks, but it focuses on Deep Neural Network.

2. NumPy: It is a Python library that is used for scientific computation. It provides support for multi dimensional arrays and matrices, and offers high-level mathematical functions to work on these arrays.

3. Django: It is a Python Web framework that offers fast development and quick, functional design. It's designed to take care of a lot of the headaches of Web development so you can concentrate on writing your app instead of reinventing the wheel. It's open source and free.

4. TensorFlow-Hub: TensorFlow-Hub is a repository of qualified machine learning models that can be deployed anywhere and fine-tuned. It can be used to pass learning of a Universal Sentence Encoder model that has already been learned.

III. UNIVERSAL SENTENCE ENCODER

UST model is used to encode sentences into embedding vectors with the aim of using them in other Natural Language Processing tasks. According to studies, transfer learning using sentence embeddings outperforms transfer learning at the word stage. We found that we get promising results and good performance with minimal amounts of supervised training data for a transfer task using transfer learning through sentence embeddings using UST.

For many NLP activities, there is a scarcity of training data. This poses a dilemma for data-hungry deep learning algorithms. Huge training sets are not available for many of the researches or industry NLP tasks due to the high cost of annotating supervised training data. Many models approach the issue by using pre-trained word embeddings to perform selective transfer learning implicitly. For encoding sentences into word embeddings, this model employs the Deep Averaging Network (DAN) architecture.

To generate sentence embeddings, the input words are first averaged and then passed through a feed forward deep neural network (DNN). The encoder takes a lowercased Penn Treebank tokenized string as input and produces a 512-dimensional sentence embedding as an output. It employs multitasking learning. The DAN encoder's key advantage is that its compute time is proportional to the length of the activation function.

IV. TRANSFER LEARNING MODULE

The similarity of the sentence embeddings provided by the encoder in the pair wise semantic similarity task can be used to find the degree of likelihood of two sentences. We compute the cosine similarity of the two sentence embeddings first, as seen in Eq. 1

$$\mathbf{Sim(u,v) = (1 - arccos\ (\ (u\ .\ v\ )\ /\ (||u||\ .\ ||v||))/\pi)\ (1)}$$

V. ARCHITECTURE

The interviewer first creates a user account in the application. This account helps to create and manage interviews. The interviewer is greeted with a dashboard after signing in. The interviewer would have the option of scheduling an appointment. He/she is given a collection of well-thought-out, well-defined, well-organized, and to-the-point questions. The interviewer then chooses the questions based on the work requirements. Each question can be selected using a variety of filters available to the interviewer. The interviewer establishes the link for the test after the questions are ready. Candidates who passed the first round of the job interview are given this link. The applicant completes the test and responds to each question. The responses are saved in a database. When the Interviewer goes to the outcome section of the dashboard, he or she will see a list of candidates who attended the interview, as well as their findings and interview remarks. These observations will significantly assist the interviewer in going forward in the process. Django is used to construct the entire application, and transfer learning modules are combined to deliver the output.

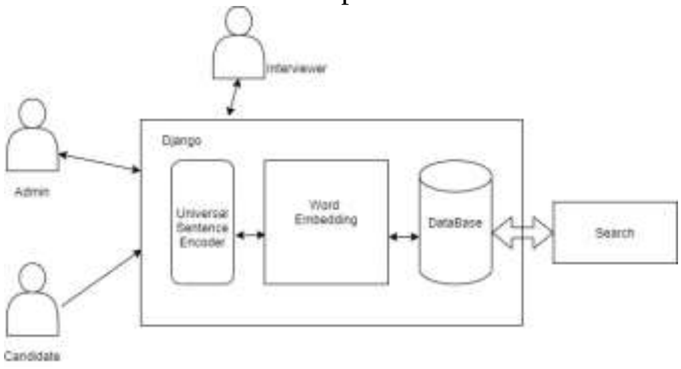

Fig. 1 Simple Architecture Diagram of the application

VI. CONCLUSION

Thus an application is developed using Django and the universal sentence encoder. This paper introduces the first iteration of the application. In the future, it can be further developed to capture live video using computer Vision for facial expressions of the candidate. Moreover several Deep learning methods like sentiment analysis, interpersonal skills can be identified using the answers provided by the candidate.

ACKNOWLEDGEMENT

We thank our teammates from Panimalar, and other people for their feedback and suggestions. Special thanks to our Project guide Dr. S. Balaji for helping us to successfully implementing the first version of the project.

REFERENCES

[1] "Universal Sentence Encoder", CornellUniversity, https://arxiv.org/abs/1803.11175

[2] Mamdouh Farouk, "Measuring Sentences Similarity: A Survey", Cornell University, https://arxiv.org/abs/1910.03940

[3] Adrien Seig, "Text Similarities : Estimate the degree of similarity between two-texts", https://medium.com/@adriensieg/text-similarities-da019229c894

[4] Rupert Thomas, "How to Rank Text Content by Semantic Similarity", https://towardsdatascience.com/how-to-rank-text-content-by-semantic-similarity-4d2419a84c32

[5] Eneko Agirre, "STS (Semantic Textual Similarity) Benchmark", http://ixa2.si.ehu.eus/stswiki/index.php/STSbenchmark

[6] Karl Pearson, "Pearson correlation coefficient", https://en.wikipedia.org/wiki/Pearson_correlation_coefficient

[7] Django Web Framework Documentation, https://docs.djangoproject.com/en/3.1

[8] TensorFlow, https://www.tensorflow.org/