Question Answer Grader Based On Transformers

Mert Bayraktar
Department of Computer Engineering Akdeniz University Antalya, Turkey 202151075008@ogr.akdeniz.edu.tr

***Abstract*—The transformer is a novel type of neural architecture that uses the attention mechanism to encode the input data as strong characteristics. With transformers, the field of natural language processing is being transformed. The latter is built on a cutting-edge, pre-trained neural network structure. In this paper, transformer models from hugging face are used. The transformer models generate the similarity index between user-provided answer parameters and reference answers from sample text data. Then some similarity measurement techniques are used to compute the similarity scores between sentences.**

***Index Terms*—Answer grading, Transformer, BERT, Deep Learning, Word Embeddings, Sentence Similarity.**

I. Introduction

Two text items were considered similar before if they composed of same words. The text was represented as real value vectors using methods such as Bag of Words (BoW) and Term Frequency-Inverse Document Frequency (TF-IDF) to assist in the estimation of semantic similarity. These approaches, nonetheless, did not account for the fact that words have different meanings and that different words can be used to represent the same notion. These approaches were easy to implement, but they neglected the semantic and syntactic belongings of text. The measure of semantic likeness between two text items is defined as Semantic Textual Similarity (STS) [1]. Semantic similarity is a crucial linguistic technique used in Natural Language Processing (NLP). It is used in various applications such as question answering, sentiment analysis and plagiarism detection. The degree of semantic likeness between two commodities, which can be ideas, sentences, or documents, is determined by semantic similarity. Acquiring similarity is dependent on comparing the similarity of character sequences [2]. The transformer is a well-known architecture that has found widespread application in natural language processing (NLP), computer vision (CV), and speech processing. The transformer was initially suggested as a sequence-to-sequence model for machine translation. Later work demonstrates that Transformer-based pre-trained models (PTMs) can achieve cutting-edge performance on a variety of tasks [3].

The rest of the paper is structured as follows: Related works are presented in Section II. Transformer models and metrics that have been used to evaluate the similarity are presented in Section III. In Section IV the results are shown. Finally in Section V conclusions are given.

II. Related Work

This section provides a detailed survey of semantic similarity approaches that have been used in the NLP area. Figure 1 shows various types of semantic similarity approaches. Knowledge-based methods use the information contained in

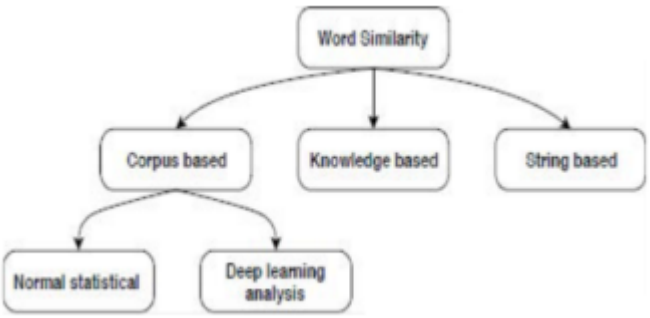

Fig. 1. Semantic Similarity Approaches.

Knowledge graphs to calculate the semantic similarity between items. The similarity of two items can be obtained by estimating the shortest path length between two concepts. The following equation can be used to calculate the similarity between two items.

$$sim_{path}(c_i, c_j) = 1/1 + length(c_i, c_j) \tag{1}$$

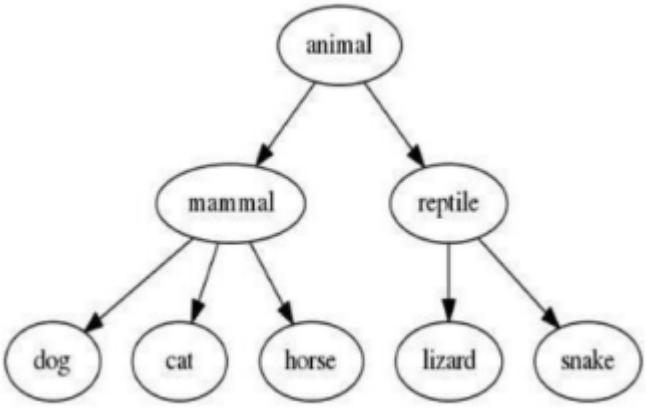

Fig. 2. Knowledge Representation.

Most of the methods in the text similarity field are corpus-based methods. This type of method extracts useful information from a large corpus. A large corpus aids in accurately calculating text similarity by checking word co-occurence. Furthermore, there are two methods for analyzing a corpus. The first approach is to use traditional statistical knowledge,

Latent Semantic Analysis (LSA), and the second approach is to use deep learning. An important goal in corpus analysis is to calculate the term frequency-inverse document frequency (TF-IDF), which is used as a word weighting. LSA is one of the most widely-used methods for calculating the semantic similarity of texts. In LSA, each word is represented by a word co-occurence matrix where the rows represent the words and, columns represent the paragraphs. Based on the constructed word co-occurence vector, the similarity is calculated using cosine similarity. Word embedding models are also used in

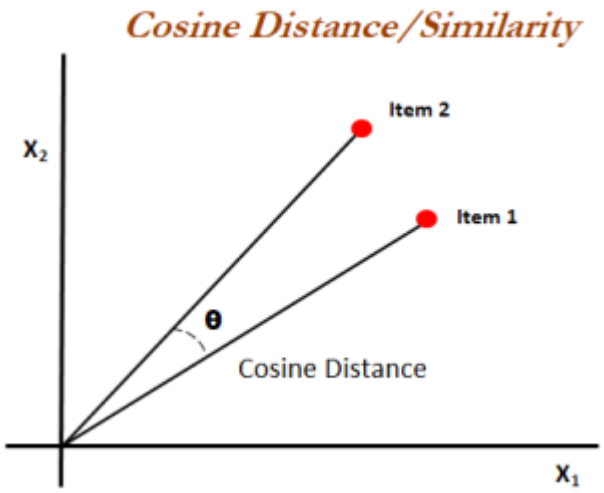

Fig. 3. Cosine Similarity.

text similarity. Word embeddings are vector representations of words that maintain the underlying linguistic affinity between the words. These vectors are computed using different approaches, including neural networks, word co-occurrence matrices [5]. Some of the widespread word embedding models for detecting semantic similarity are, word2vec [6], Glove [7], fastText [8 and BERT [9].

Atish Pawar and colleagues [10] By incorporating corpora-based statistics into a standardized semantic similarity algorithm, this paper proposes a methodology that can be applied in a variety of fields. The proposed method calculates the semantic similarity between words and sentences using an edge-based approach and a lexical database. It has been tested on benchmark standards as well as the mean human similarity dataset, and the methodology demonstrates a high correlation value for Rubenstein's word and sentence similarity. It outperforms other unsupervised models and performs admirably on the SICK dataset.

Taihua Shao [11] propose a Transformer-based neural network for answer selection in this paper. It installed a bidirectional long short-term memory (BiLSTM) behind the Transformer to collect global and sequential information in the question or answer. This Transformer-based network, unlike the original Transformer, focuses on sentence embedding rather than the seq2seq task. Furthermore, BiLSTM is being used to incorporate sequential features. This method is tested on the popular QA dataset WikiQA, and the experimental results show that this Transformer-based answer selection model can outperform several competitive methods. Experiments show that this model outperforms other cutting-edge methods in terms of MAP, MRR, and accuracy.

P J Sijimol et al. [12] Handwritten Short Answer Evaluation System (HSAES) is a paper that describes an automated short answer evaluation system that can identify texts in answer papers and evaluates marks for each short answer based on knowledge acquired by the model through training. OCR tools are used in the proposed system to extract handwritten texts. The human-evaluated sample dataset of handwritten answer papers and answer key is used to extract keywords using NLP. The proposed model uses cosine similarity measures to evaluate scores. Each sentence in the evaluated answer paper will receive a mark.

Alla Defallah Alrehily and others [13] The purpose of this paper is to propose an automated assessment system for subjective questions. It computes the keyword matching ratio between original answers and student answers. The points are assigned based on the semantic and document similarity. The evaluation system includes four modules: preprocessing, keyword expansion, matching, and grading. Various tests are carried out and prospective results are obtained. For the testing parameters recall, precision, accuracy, and f measure, moderate values are obtained. This system contributes to the examination system's increased efficiency and productivity. It also helps to save time, cut costs, and use resources more efficiently.

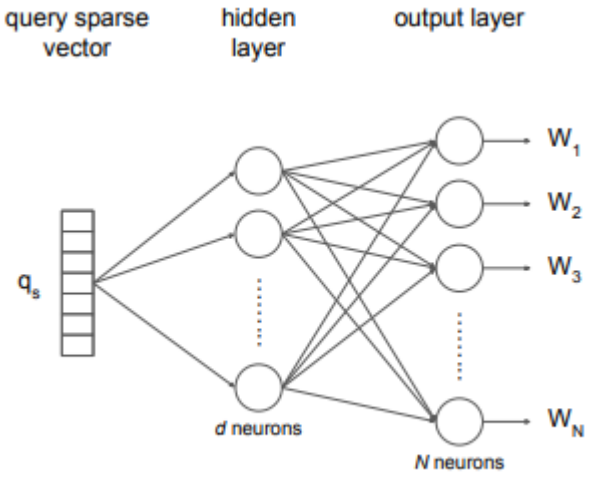

Fig. 4. Word Embedding Architecture.

Vaswani et al. [14] proposed a transformer model that captures the semantic properties of words in embeddings using attention mechanisms. The transformer is divided into two parts: "encoder" and "decoder." Layers of multi-head attention mechanisms are followed by a fully connected feed-forward neural net work in the encoder. The decoder is similar to the

encoder except for one additional layer of multi-head attentic that captures the encoder's attention weights.

A variety of BERT model variations were proposed base on the corpus used to train the model and by optimizir computational resources. Lan et al. [15] proposed ALBER which includes two techniques for reducing BERT's compu tational complexity: "factorized embedding parameterizatior and "cross-layer parameter sharing."

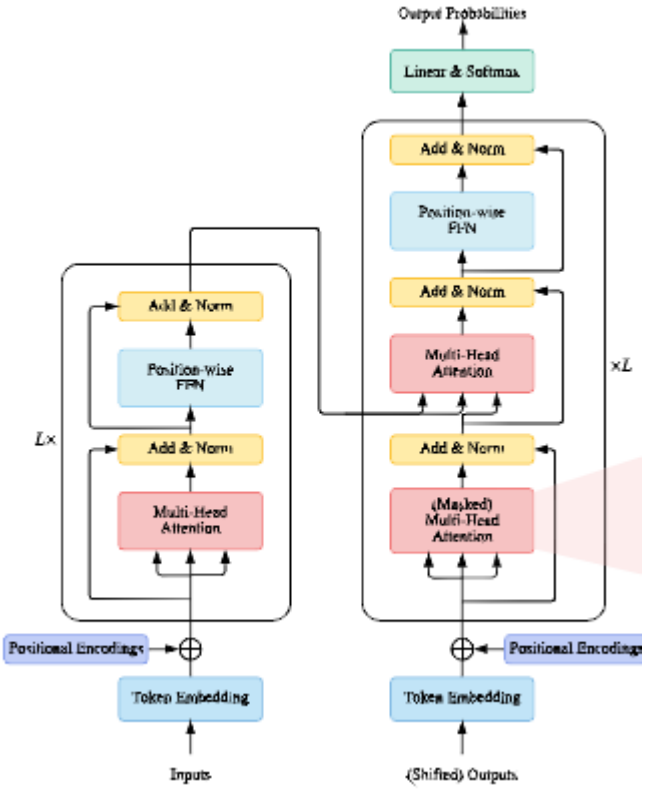

Fig. 5. Vanilla Transformer Architecture.

III. Method

In this subsection, methods that have been used to detect text similarity and similarity metrics that have been used to evaluate the transformer architecture are discussed.

A. Model

Given two text items, the goal of the given task is to measure the similarity between these items. In order to fulfill this task, Sentence-BERT (SBERT) transformer architecture is used. The implementation of the SBERT architecture can be found in Huggingface. SBERT applies a pooling operation to the output of BERT / RoBERTa in order to generate a fixed-sized sentence embedding. In the original paper [16], the authors tested three pooling strategies: using the CLS-token output, computing the mean of all output vectors (MEAN strategy), and computing the maximum-over-time of the output vectors (MAX-strategy). MEAN is the default configuration.

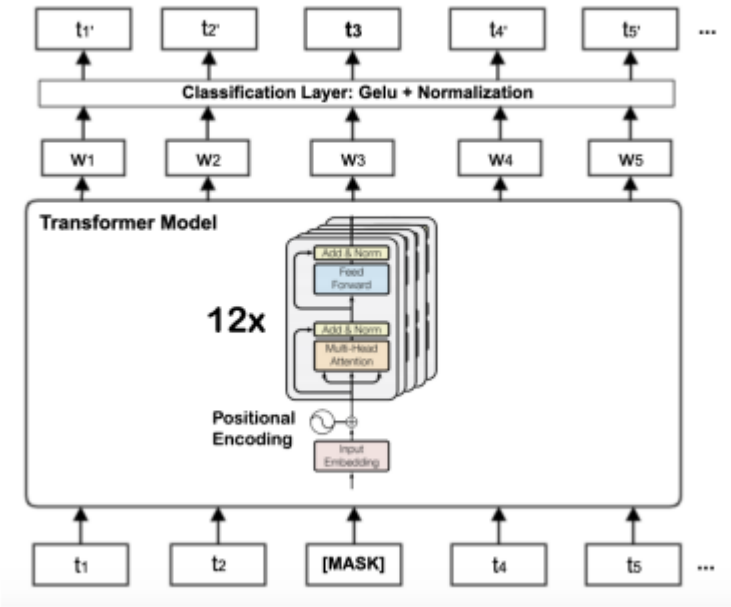

Fig. 6. BERT Architecture.

The authors create siamese and triplet networks in order to fine tune BERT architecture.

The sentence embeddings u and v are concatenated with the element-wise difference —u-v— and multiplied with the trainable weight.

$$o = softmax(W_t(u, v, |u - v|)) \tag{2}$$

where n is the dimension of the sentence embeddings and k is the number of labels. The similarity between the text items are calculated with cosine similarity.

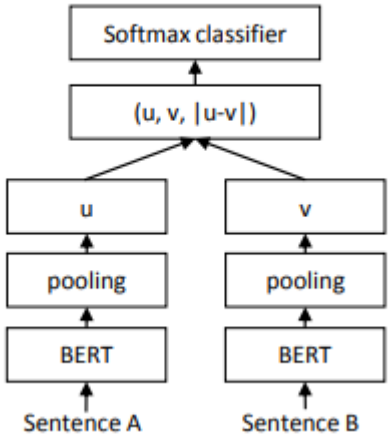

Fig. 7. SBERT Architecture with classification objective function.

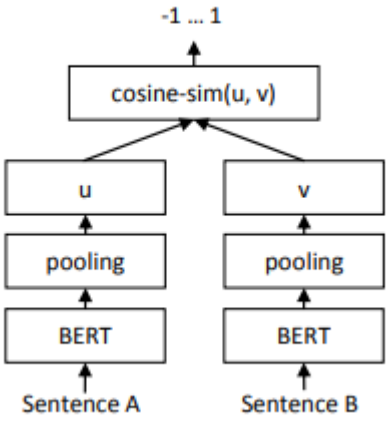

Fig. 8. SBERT Architecture to compute similarity.

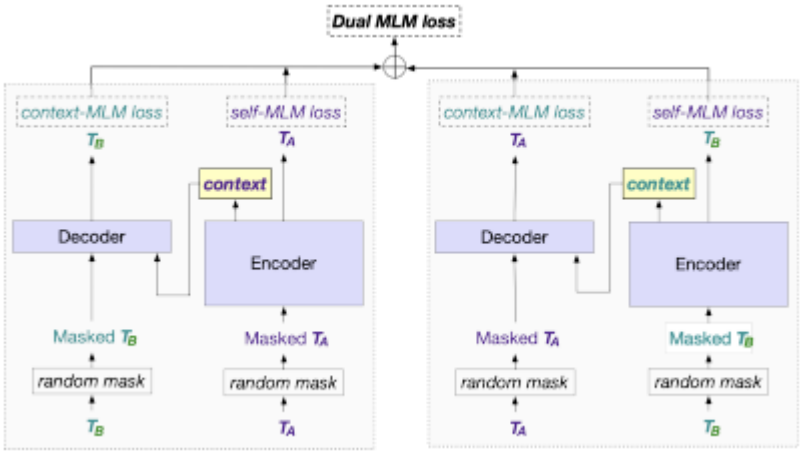

Fig. 9. Cot-MAE: Contextual Masked AutoEncoder Architecture to compute similarity.

B. Evaluation-Text Similarity

Modern approaches frequently learn a (complex) regression function that converts sentence embeddings to a similarity score. However, because of the combinatorial eruption, these regression functions are often not scalable once the group of sentences reaches a certain size. To compare the similarity of two sentence embeddings, cosine similarity is used. The given text file was the answer of the question 'How does the abstraction helps engineering?'. User provided parameters 'Abstraction is a general concept which you can find in the real world as well as in OOP languages. Abstraction is a general concept which you can find in the real world as well as in OOP languages.', and 'Abstraction (from the Latin abs, meaning away from and trahere , meaning to draw) is the process of taking away or removing characteristics from something Abstraction (from the Latin abs, meaning away from and trahere , meaning to draw) is the process of taking away or removing characteristics from something.' are compared with the true answer. The results are shown in the below table for various transformer architectures.

SBERT	BERT	miCSE	CoT-MAE
0.48	0.45	0.46	0.45
0.31	0.30	0.29	0.28

The first row represents the cosine similarity scores for the first user-provided answer. The second row represents the cosine similarity scores for the second user-provided answer.

IV. Conclusion

In this paper, text similarity methods such as knowledge-based methods, corpus-based methods, and deep learning-based methods have been analyzed and various transformer architectures for the given task is implemented. Analysis shows that the SBERT method outperforms the other methods for the given text data.

References

[1] S. P. and A. P. Shaji, 'A Survey on Semantic Similarity', in 2019 International Conference on Advances in Computing, Communication and Control (ICAC3), Dec. 2019, pp. 1–8. doi: 10.1109/ICAC347590.2019.9036843.

[2] D. Chandrasekaran and V. Mago, "Evolution of semantic similarity—A survey," ACM Comput. Surv., vol. 54, no. 2, pp. 1–37, 2021.

[3] T. Lin, Y. Wang, X. Liu, and X. Qiu, "A survey of transformers," arXiv preprint arXiv:2106.04554, 2021.

[4] A. Vaswani et al., "Attention is all you need," in Proc. Adv. Neural Inf. Process. Syst., 2017, pp. 5998–6008.

[5] T. Schnabel, I. Labutov, D. Mimno, and T. Joachims, "Evaluation methods for unsupervised word embeddings," in Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, 2015, pp. 298–307.

[6] T. Mikolov, K. Chen, G. S. Corrado, and J. Dean. Efficient estimation of word representations in vector space. CoRR, abs/1301.3781, 2013

[7] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in Proc. Conf. Empirical Methods Natural Lang. Process., 2014, pp. 1532–1543.

[8] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," Trans. Assoc. Comput. Linguistics, vol. 5, pp. 135–146, Dec. 2017

[9] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, arXiv:1810.04805. [Online]. Available: http://arxiv.org/abs/1810.04805

[10] A. Pawar and V. Mago, "Challenging the Boundaries of Unsupervised Learning for Semantic Similarity," IEEE Access, vol. 7, pp. 16291–16308, 2019, doi: 10.1109/ACCESS.2019.2891692.

[11] T. Shao, Y. Guo, H. Chen, and Z. Hao, "Transformer-Based Neural Network for Answer Selection in Question Answering," IEEE Access, vol. 7, pp. 26146–26156, 2019, doi: 10.1109/ACCESS.2019.2900753.

[12] Sijimol P J, Surekha Mariam Varghese, "Handwritten Short Answer Evaluation System (HSAES)" , IJSRST — Volume 4, 2018

[13] Alrehily, A.D., Siddiqui, M.A. and Buhari, S.M., 2018. "Intelligent Electronic Assessment for Subjective Exams". ACSIT, ICITE, SIPM, pp.47-63.

[14] M. Syamala Devi and Himani Mittal,"Machine Learning techniques with Ontology", International Journal on Natural Language Computing (IJNLC), vol5, no.2, 2016

[15] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut. Albert: A lite bert for self-supervised learning of language representations. arXiv preprint arXiv:1909.11942, 2019.

[16] N. Reimers and I. Gurevych. "Sentence-bert: Sentence embeddings using siamese bert-networks.", arXiv preprint arXiv:1908.10084, 2019.

[17] T. Kei, M. Nabi, 2018. "miCSE: Mutual Information Contrastive Learning for Low-shot Sentence Embeddings", arXiv.

[18] X. Wu, G. Ma, M. Lin, Z. Lin, Z. Wang, S. Hu, 2022, “ConTextual Mask Auto-Encoder for Dense Passage Retrieval”, arXiv.