Proceedings of the 4th International Conference on Signal Processing and Machine Learning
DOI: 10.54254/2755-2721/54/20241465

Analysis of the text matching problem

Dingyuan Hu

Department of Computer Science, Chang'an University, Xi'an 710064, China

2022900358@chd.edu.cn

Abstract. As a matter of fact, with the rapid development of computation ability as well as machine learning scenarios, the natural language model had widely use in society in recent years. To be specific, lots of software and applications have been developed, such as Chat-GPT, machine translation and text generation. Among various applications, text matching constitutes a foundational challenge within the realm of natural language processing (NLP). With this in mind, this study delineates the advancements encapsulated by contemporary models pivotal to text matching methodologies, i.e., LSTM, Transformer as well as BERT. At the same time, this study will make a comprehensive demonstration of them, including the principle and the related works. In the meantime, this study will also provide 10 well-known dataset of Text Matching problem. Finally, this research presents analysis of the limitation for this task as well as gives the future prospect for text matching at the same time.

Keywords: NLP, text matching, LSTM, BERT.

1. Introduction

Text Matching is a broad concept. As long as the objective is to analyse the correlation between two expressions, the problem can essentially be considered a matter of text matching. Definitions for matching can differ depending on the scenario, so text matching cannot be seen as a complete and independent research direction. Nevertheless, a number of NLP tasks could be modelled as text matching problems [1-3]. When analysing the text matching problem, it becomes apparent that there exist slight disparities among the model structure, training techniques, and other aspects, notwithstanding their high similarity [4]. Thus, although this difficulty may seem uncomplicated in principle, obtaining admirable outcomes in particular matching problems proves to be challenging (especially before BERT). The sources of training data in Text Matching are variety. Commonly obtained from e-mail, published research, social media and etc. As long as a pair of text that is logical can be considered as a source of data [5].

Text matching, inherently central to various NLP endeavours, underpins numerous applications such as question answering systems, information retrieval, news categorization, affective text analysis, and text mining. The field has witnessed extensive scholarly contributions, resulting in the inception of robust and seminal architectures, including convolutional neural networks, recurrent neural networks, Long Short-term Memory networks, Transformers, and Bidirectional Encoder Representations from Transformers [6-10]. At first, CNNs represents the most popular model in deep learning area, as researchers attempt to emulate human thought patterns and incorporate them into algorithms. The CNN's local weight sharing structure facilitates parallel learning and enables efficient processing of high-

DOI: 10.54254/2755-2721/54/20241465

dimensional data, resulting in closer simulation of biological neural networks. A prevalent complication arises when excessively deep network layers impede the efficacy of back-propagation, decelerating modifications adjacent to the input layer and utilizing gradient descent for iteration may cause training results to converge to a local optimum instead of a global optimum. More importantly, the pooling layer will ignore the correlation between the integral and the local. Therefore, the semantic meaning of the context of the translated text may lack relevance and logic. In 1997, Sepp Hochreiter and Jurgen Schmid Huber introduced LSTM, a complex neural network block, is used for modelling complex sequential data or time-series data. LSTM outperforms RNN, which can't remember longer contexts due to vanishing and exploding gradients issues in sequential data [6].

However, LSTM did not completely solve the problem of gradient. It has some effect in processing N order of magnitude sequences, but processing 10N or longer sequences will still be exposed. Moreover, within each LSTM unit, four fully connected layers exist, signifying a complex architecture. A lengthy time series or an extensively deep network accentuates computational demands and time expenditure. The Transformer architecture addresses these constraints by employing self-attention mechanisms, assigning distinct "attention scores" to individual words within a context, thereby gauging their respective impacts. This approach enhances parallelization, surpassing the efficiencies of traditional CNNs and RNNs, and facilitates the accommodation of expansive models through advanced GPU computations.

2. Basic descriptions of text matching

Expression strategy is based on the Siamese Network architecture, and the two backbone networks share the weight. At the initial stage, that is, two pieces of text are processed independently. First, in the unified semantic space, the representation (semantic vector) of two pieces of text is calculated based on neural network. Then, based on the obtained text representation, the similarity of the two texts is calculated. The single sentence text representation obtained in this way will not change with the change of sentence pair, so only one calculation is needed to store the text representation offline. Two representative works are DSSM and Sentence BERT.

Interactive strategy believes that the similarity is calculated based on the text representation at the end of the model, potentially overshadowing inherent linguistic attributes, including morphological and syntactical elements. Therefore, at the initial stage, two pieces of text are interacted based on various levels (such as word level, phrase level, etc.). The structure of the interaction layer is generally based on attention to derive a matching matrix, subsequently employing deep networks for the extraction of advanced matching characteristics, culminating in the computation of a comprehensive similarity score. In essence, the interactive strategy eschews direct learning of text representation, positing instead that global matching efficacy is contingent upon local matching degrees, interacts with the text features as early as possible, captures more basic features, and finally calculates the text similarity based on these basic matching features at the high-level. The work of such methods is basically to optimize the way of interaction or make the model deeper. Typical strategies include MatchPyramid, ESIM and BERT [7].

Text Matching wants to find out the relation between a pair of texts. In this area, there is some of popular issue that researchers most likely to investigate.Text Entailment. Text Entailment necessitates discerning if a hypothesis emerges logically from an initial premise [1]. Question Answering bifurcates into extractive and generative categories. Given a question and a group of answers of candidate, the system evaluates each response as correct or incorrect. Extractive QA involves selecting an answer from a given text, while generative QA necessitates the real-time formulation of responses [2]. Natural Language Inference (NLI). Natural Language Inference (NLI) predominantly involves the encapsulation and computational utilization of these linguistic relationships [3-7]. Textual similarity is a classic application of Text Matching, namely determine whether two text express repeated content. In other words, forms paraphrase. In this task, the common output is in the form of probability, but sometimes it is judged directly by 0 or 1. The well-known datasets are listed as following:

DOI: 10.54254/2755-2721/54/20241465

- GLUE Benchmark (General Language Understanding Evaluation) evaluates the performance of models across diverse NLP tasks, encompassing sentence similarity, answering questions, and sentiment analysis.
- SQuAD (Stanford Question Answering Dataset) is designed to train and assess question-answering models using passages from Wikipedia.
- CoNLL CoNLL refer to Conference on Computational Natural Language Learning contains data for several tasks, incorporating multifaceted data for tasks such as named entity recognition as well as dependency parsing.
- CommonsenseQA is a question answering dataset that requires models to have commonsense knowledge to answer.
- MTurk (Amazon Mechanical Turk) is crowdsourced data for various tasks, often used for generating and validating datasets.
- MultiNLI (MultiNLI Corpus) is a natural language inference dataset used to train models on understanding entailment and contradiction.
- OpenSubtitles contains a collection of translated movie subtitles, often used for machine translation and dialogue system training.
- SNLI (Stanford Natural Language Inference) is another dataset for natural language inference to understand relationships between sentence pairs.
- WikiText is a large-scale dataset derived from Wikipedia used for language modelling.
- SemEval (Semantic Evaluation) provides datasets for various tasks in semantic analysis, like sentiment analysis, semantic role labelling, and more.

These datasets play foundational roles in training, evaluating, and benchmarking NLP models. Some are task-specific, while others, like the GLUE benchmark, encompass a range of tasks to provide a holistic evaluation of model performance. This article will introduce three deep learning models that are currently most widely used and most versatile, i.e., LSTM, Transformer and BERT. In addition, LSTM is constructed upon the RNN architecture, while BERT is grounded on the concept of encoder from the Transformer architecture.

3. LSTM and LSTM-based model

LSTM is a RNNs architecture. It is tailored to capturing long-term dependencies, and resolving the vanishing gradient issue that RNNs within traditional architectures often encounter. In 1997, Sepp Hochreiter and Jürgen pioneered the Long Short-Term Memory network. A sketch is shown in Fig. 1 [7]. The forked line means copy of vector. And the upper purple point means output of single cell. A key concept within LSTM is the utilization of a memory cell. This component enables the network to selectively recall or discard information over lengthy sequences. This memory cell functions as a central element that facilitates the LSTM in acquiring and retaining information over extended periods. An LSTM cell is engineered with three cardinal components: an input gate, a forget gate, and an output gate, collectively orchestrating the information dynamics within the LSTM cell. The input gate discerns portions of data for retention within the memory cell, employing prior hidden states and current indicators to generate a value spectrum between 0 and 1 for each element of the input. A value of 1 is allocated to indicate that the corresponding element must be fully saved in the memory cell, while a value of 0 means it should be ignored completely. The forget gate ascertains the information to be omitted within the memory cell, utilizing the preceding hidden state and current input to calculate a 'forget factor' for each component. This factor adjudicates the proportion of information preserved or eliminated.

Conversely, the output gate modulates the information transition from the memory cell to the output, determining an 'output factor' for each component based on prior states and current inputs. This factor governs the quantity of information relayed from the memory cell to subsequent network segments. Chen and his group proposed a ESIM (Enhanced sequential inference model) model. They use bidirectional LSTM (biLSTM) as a basic block encoding input (premise and hypothesis) and performing composition inference to construct final prediction. Subsequently, the author analysed alternative

DOI: 10.54254/2755-2721/54/20241465

recurrent memory constructs, notably Gated Recurrent Units (GRUs) [7], which demonstrated sub-optimal performance compared to LSTMs in stand-alone evaluations for Natural Language Inference tasks.

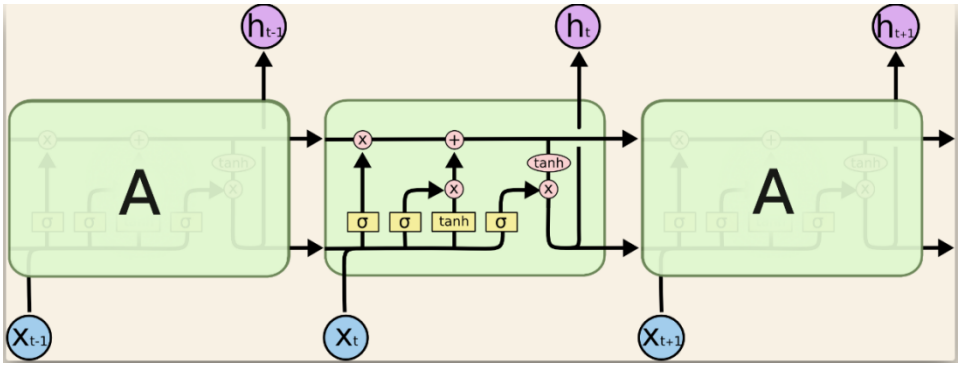

Figure 1. Cell structure of LSTM model. Yellow squares are 4 neural network layers. The direction of arrow is route of vector transformed. And red point is pointwise operation [7].

4. Transformer and Transformers-based model

The Transformer framework, while rooted in encoder-decoder principles, boasts a composition surpassing conventional Attention mechanisms in complexity. Transformers fundamentally operate on an attention mechanism coupled with a FFNN. Constructing a modifiable neural network with stacked Transformers, the study undertook the assembly of six encoder and decoder layers, culminating in a 12-layer Encoder-Decoder structure, thereby reaching an unprecedented threshold in BLEU scores for machine translation. Attention within this domain diversifies into three categories, contingent on the origin of queries and key-value pairs:

- Self-attention. Here, the input sequence segregates into three vectors—query, key, and value—all deriving from an identical input sequences and are used to calculate the attention score between each input element. Therefore, Self-Attention can be used to learn the dependencies between elements in a single sequence, such as context understanding in language modelling.
- Masked Self-attention: Employed within the Transformer's decoder, this constrained self-attention ensures that positional queries solely attend to preceding key-value pairs, inclusive of the current stance. This restriction, customarily executed via a masking operation on the non-normalized attention matrix, permits parallel training, earning the designation of auto-regressive or causal attention [8].
- Cross-attention. This variant projects queries from the antecedent decoder layer's outputs, while keys and values emerge from encoder outputs, accentuating parallel operational capabilities.

Within the Transformer's architecture, the encoder segment transmutes input sequences into a continuum of vector representations, subsequently decoded into respective output sequences. Both the encoder and decoder consist of several uniform layers, each featuring two subdivisions: multi-head self-attention and a feed-forward neural network. Explicitly, within the encoder's multi-head self-attention mechanism, each constituent word in the input sequence engages in similarity calculations with its counterparts, determining specific weights. Subsequent vector representations of these words undergo a weighted summation based on these metrics, crafting a novel vector representation for each term. This process ensures holistic input sequence information integration into each word's representation. The feed-forward neural network simply performs a nonlinear transformation on the vector at each position.

The difference involving the multi-head self-attention in the decoder as well as the encoder is that in addition to calculating the similarity of each position in the input sequence, the similarity between the information output by the encoder and the current position also needs to be calculated. This allows the decoder to consider not only the information of the input sequence, but also the previously generated

DOI: 10.54254/2755-2721/54/20241465

context information when generating the output. The feed-forward neural network in decoder is the same as that in encoder.

5. BERT and BERT based model

BERT's methodology pivots on two central phases: pre-training and fine-tuning. In its pre-training stage, the model undergoes conditioning on unlabelled data across various tasks. Conversely, during fine-tuning, the model, initialized with pre-trained parameters, undergoes comprehensive refinement on libelled data specific to downstream tasks. Despite a uniform foundation in pre-trained parameters, each downstream task necessitates distinct fine-tuned models. As depicted in Fig. 2, the foundational elements comprise input embeddings and positional encodings, forming the structural base. The multi-head attention, coupled with Add & Norm functionalities, constitutes the attention mechanism, while the subsequent feed-forward component, also utilizing Add & Norm, characterizes the feed-forward neural network segment.

There are 4 steps to access BERT model. The first step is input representation. BERT takes a text sequence as input, which is represented as word embeddings. Each word is mapped to a d-dimensional vector. BERT supports various word embedding methods, such as Word Piece and BPE. Subsequently, the model employs the Transformer Encoder as a second step, wherein BERT integrates the Transformer framework as its encoding mechanism. This encoder features numerous homogeneous layers, each partitioned into two segments: multi-head self-attention and the feed-forward neural network. The self-attention apparatus meticulously models interrelations among words within the input sequence, enhancing the model's proficiency in managing extensive dependencies. Proceeding to the third phase, pre-training, the process bifurcates into Masked Language Modelling (MLM) and Next Sentence Prediction (NSP). MLM involves the arbitrary masking of 15% of the input sequence words, propelling the model to predict these occlusions, thereby fortifying its contextual comprehension. Concurrently, NSP tasks the model with discerning the coherence or adjacency between sentence pairs, refining its grasp of semantic interrelations. The culmination of this process, fine-tuning, sees the pre-conditioned BERT model tailored to downstream applications, ranging from sentiment analysis to named entity recognition. This stage involves meticulous parameter adjustments, optimizing the model for task-specific exigencies, thereby augmenting its operational efficacy.

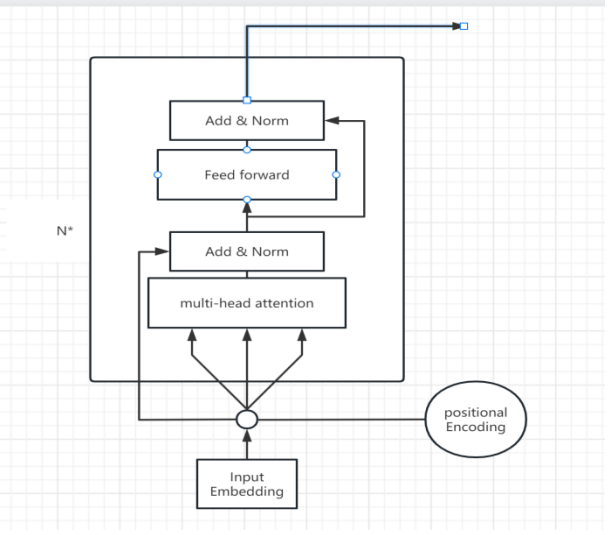

Figure 2. Architectural Overview of the BERT Model (Photo/Picture credit: Original)

Innovative endeavours have emerged, amalgamating the auto-regressive and auto-encoding faculties of Pretrained Language Models. XLNet, for instance, assimilates the auto-regressive essence of models akin to OpenGPT with BERT's bidirectional context interpretation. During pre-training, XLNet implements a permutation strategy, permitting the inclusion of tokens from both preceding and

DOI: 10.54254/2755-2721/54/20241465

succeeding contexts, thereby establishing itself as a sequence-aware auto-regressive language model. This permutation is facilitated through a distinctive attention mask within the Transformer framework. Furthermore, XLNet inaugurates a dual-stream self-attention mechanism, fostering position-conscious word prognostication. This innovation stems from recognizing the substantial disparities in word distributions relative to their positional occurrences. Notably, sentence commencements exhibit distinct distributional characteristics compared to subsequent segments. This method involves constructing a content stream encompassing the position and token embeddings of antecedent words (3, 2, 4) and a subsequent query stream comprising the content stream alongside the position embedding of the targeted predictive word (position 1) [8].

6. Conclusion

To sum up, text matching, pivotal in natural language processing, strives to identify semantic similarity or equivalence between multiple texts. Its applications include information retrieval, text classification, question answering, and dialogue systems. Despite advances in this area, it remains a daunting task due to the complexity and diversity of natural language. Text matching, pivotal in natural language processing, strives to identify semantic similarity or equivalence between multiple texts. The semantic gap between texts poses as a primary challenge. Even if two pieces of text share the same words, they can have different meanings or convey different information. For instance, "apple juice" and "apple company" contain the word "apple," but they have different meanings. Recently, research has concentrated on creating neural network models that can identify the semantic similarity between texts. These models utilize pre-trained language models, such as BERT as well as GPT for encoding textual information into dense vectors that are comparable using various similarity metrics. One of the challenges in text matching is the scarcity of annotated data. Extensive labelled data is necessary to train and assess the models for text matching. Nonetheless, procuring high-quality labelled data is typically costly and time-intensive. To address this challenge, scholars have investigated different learning methods, including unsupervised and transfer learning. Unsupervised techniques can learn from unlabelled data without the need for annotations, while transfer learning can leverage knowledge acquired from a related task to curtail the dependency on voluminous labelled datasets when adapting to novel tasks.

References

[1] Bowman S R, Angeli G, Potts C and Manning C D 2015 arxiv:1508.05326.
[2] Minaee S, Kalchbrenner N, Cambria E, Nikzad N, Chenaghlu M and Gao J 2021 ACM computing surveys (CSUR) vol 54(3) pp 1-40.
[3] Condoravdi C, Crouch D, De Paiva V, Stolle R and Bobrow D 2003 Proceedings of the HLT-NAACL 2003 workshop on Text meaning pp 38-45.
[4] Ido Dagan O G and Bernardo M 2006 Machine learning challenges Evaluating predictive uncertainty visual object classification and recognizing textual entailment pp 177– 190.
[5] Johan B and Katja M 2005 Proc EMNLP vol 6 p 12
[6] Yaroslav F, Yoad W and Nissim F2000 Proc of the 2nd Workshop on Inference in Computational Semantics p 7.
[7] Bill M and Christopher D M 2009 Proc of the Eighth International Conference on Computational Semantics p 13.
[8] Minaee S, Kalchbrenner N, Cambria E, Nikzad N, Chenaghlu M A and Gao J 2020 ACM Computing Surveys (CSUR) vol 54 pp 1-40
[9] Lin T, Wang Y, Liu X and Qiu X 2021 AI Open vol 3 pp 111-132.
[10] Khan S, Naseer M, Hayat M, Zamir S W, Khan F S and Shah M 2022 ACM computing surveys (CSUR) vol 54(10) pp 1-41.