Hungarian Layer: Logics Empowered Neural Architecture

Abstract

Neural architecture is a purely numeric framework, which fits the data as a continuous function. However, lacking of logic flow (e.g. \textit{if, for, while}), traditional algorithms (e.g. \textit{Hungarian algorithm, A* searching, decision tress algorithm}) could not be embedded into this paradigm, which limits the theories and applications. In this paper, we reform the calculus graph as a dynamic process, which is guided by logic flow. Within our novel methodology, traditional algorithms could empower numerical neural network. Specifically, regarding the subject of sentence matching, we reformulate this issue as the form of task-assignment, which is solved by Hungarian algorithm. First, our model applies BiLSTM to parse the sentences. Then Hungarian layer aligns the matching positions. Last, we transform the matching results for soft-max regression by another BiLSTM. Extensive experiments show that our model outperforms other state-of-the-art baselines substantially.

Introduction

Paraphrase identification is an important topic in artificial intelligence and this task justifies whether two sentences expressed in various forms are semantically similar, BIBREF0 . For example, “On Sunday, the boy runs in the yard” and “The child runs outside at the weekend” are identified as paraphrase. This task directly benefits many industrial applications, such as plagiarism identification BIBREF0 , machine translation BIBREF1 and removing redundancy questions in Quora website BIBREF2 . Recently, there emerge many methods, such as ABCNN BIBREF3 , Siamese LSTM BIBREF2 and L.D.C BIBREF4 .

Conventionally, neural methodology aligns the sentence pair and then generates a matching score for paraphrase identification, BIBREF4 , BIBREF2 . Regarding the alignment, we conjecture that the aligned unmatched parts are semantically critical, where we define the corresponded word pairs with low similarity as aligned unmatched parts. For an example: “On Sunday, the boy runs in the yard” and “The child runs inside at the weekend”, the matched parts (i.e. (Sunday, weekend), (boy, child), run) barely make contribution to the semantic sentence similarity, but the unmatched parts (i.e. “yard” and “inside”) determine these two sentences are semantically dissimilar. For another example: “On Sunday, the boy runs in the yard” and “The child runs outside at the weekend”, the aligned unmatched parts (i.e. “yard” and “outside”) are semantically similar, which makes the two sentences paraphrase. In conclusion, if the aligned unmatched parts are semantically consistent, the two sentences are paraphrase, otherwise they are non-paraphrase.

Traditional alignment methods take advantage of attention mechanism BIBREF2 , which is a soft-max weighting technique. Weighting technique could pick out the most similar/dissimilar parts, but is weak in modeling the aligned unmatched parts, which are the crucial evidence to identify paraphrase. For the input sentences in Figure FIGREF1 , the weight between “Sunday” and “run” is lower than the weight between “yard” and “inside”, but the former weight is not the evidence of paraphrase/non-paraphrase, because the former two words that are most dissimilar should not be aligned for an inappropriate comparison.

To extract the aligned unmatched parts, in this paper, we embed Hungarian algorithm BIBREF5 into neural architecture as Hungarian layer (Algorithm SECREF7). Illustrated in Figure FIGREF1 , the alignment in sentence matching could be formulated as the task-assignment problem, which is tackled by Hungarian algorithm. Simply, Hungarian algorithm works out the theoretically optimal alignment relationship in an exclusive manner and the exclusiveness characterizes the aligned unmatched parts. For the example in Figure FIGREF1 , because Hungarian layer allocates the aligned pairs with exclusiveness, the matched parts (i.e (Sunday, weekend), (boy, child), run) are aligned firstly, then the word “yard” would be assigned to the word “inside” with a negative similarity, making a strong evidence for discrimination.

Specifically, our model performs this task in three steps. First, our model applies BiLSTM to parse the input sentences into hidden representations. Then, Hungarian layer leverages the hidden representations to extract the aligned unmatched parts. Last, we apply cosine similarity to metric the aligned unmatched parts for a final discrimination. Regarding the training process of Hungarian layer, we modify the back-propagation algorithm in both directions. In the forward pass, Hungarian layer works out the alignment relationship, according to which, the computational graph is dynamically constructed, as demonstrated in Figure FIGREF13 . Once the computational graph has been dynamically constructed, the backward propagation could be performed as usual in a conventional graph.

We conduct our experiments on the public benchmark dataset of “Quora Question Pairs” for the task of paraphrase identification. Experimental results demonstrate that our model outperforms other baselines extensively and significantly, which verifies our theory about the aligned unmatched parts and illustrates the effectiveness of our methodology.

Contributions. (1.) We offer a new perspective for paraphrase identification, which focuses on the aligned unmatched parts of two sentences. Accordingly, we propose the Hungarian layer to extract the aligned unmatched parts. The proposed method can achieve hard and exclusive alignments between two sequences, while we can learn parameters by end-to-end back-propagation. (2.) Our model outperforms other baselines extensively, verifying the effectiveness of our theory and method.

Organization. In Section 2, we survey the related work of paraphrase identification and dynamic differentiable computational graphs. In Section 3, we introduce our neural architecture. In Section 4, we conduct the experiments. In Section 5, we conclude our paper and publish our codes.

Related Work

We have surveyed this task and categorized related papers into three lines.

Non-Neural Architecture for Paraphrase Identification

The topic of paraphrase identification raises in the last decade. The development has been through four stages before neural architectures: word specific, syntactic tree specific, semantic matching and probabilistic graph modeling.

Firstly, BIBREF6 focuses on simple surface-form matching between bag-of-words, which produces poor accuracy, because of word ambiguities and syntactic complexity. Therefore, syntactic analysis is introduced into this task for semantic understanding, such as deeper semantic analysis BIBREF7 , quasi-synchronous grammars BIBREF8 and tree edit distance BIBREF9 . Notably, most of these methods compare the grammar tree (e.g. syntactic tree, dependency tree, etc.) of sentence pair. Further, semantic information such as negation, hypernym, synonym and antonym is integrated into this task for a better prediction precision, BIBREF10 . Finally, BIBREF11 leverages a semi-Markov CRF to align phrases rather than words, which consumes too many resources for industrial applications.

In summary, the advantage of this branch, which roots the foundation in linguistics, is semantically interpretable, while the disadvantage is too simple to understand complex language phenomenon.

Neural Architecture for Paraphrase Identification: Independent Sentence Encoder

With the popularity of deep neural network, some neural architectures are proposed to analyze the complex language phenomenon in a data-fitting way, which promotes the performance. First of all, the neural network extracts the abstracted features from each sentence independently, then measures the similarity of the abstracted feature pair. There list two frameworks: CNN-based and RAE-based.

Commonly, CNN could be treated as n-gram method, which corresponds to language model. Specifically, BIBREF12 applies a bi-gram CNN to jointly model source and target sequences. BIBREF13 achieves a better performance by following this work. BIBREF14 has proposed a RAE based model to characterize phrase-level representation, which promotes simple pooling method, BIBREF15 . Multi-perspective methods BIBREF2 take the advantage of multiple metric aspects to boost the accuracy.

In summary, the advantage of this branch is to model complex and ambiguous linguistic phenomenon in a black-box style. However, the disadvantage is that the encoder could not adjust the abstracted representations according to the correlation of sentence pair, making an imperfect matching process.

Neural Architecture for Paraphrase Identification: Interdependent Sentence Encoder

To emphasize the correlation of sentence pair in encoder, the researchers propose the attention-based neural architectures, which guide the encoding process according to the corresponding part. There introduce the representative methods: ABCNN BIBREF3 and L.D.C BIBREF2 .

ABCNN is a CNN-based model. In a single stage, this model computes the attention similarity matrix for the convolution layer, then sums out each row and column as the weighs of pooling layer. The output of convolution layer is weighted by pooling layer in an average manner as the output of this stage. ABCNN could stack at most three stages. This method achieves satisfactory performance in many tasks, because of modeling correlation in sentence encoder. L.D.C model BIBREF4 is an attention-based method, which decomposes the hidden representations into similar and dissimilar parts, then respectively processes each parts to generate the final result. Notably, L.D.C is the state-of-the-art method.

In summary, the advantage of this branch is to model alignment or correlation in the encoding process. However, the disadvantage is to focus on the matched parts, rather than the unmatched parts, which are critical in this task as previously discussed.

Dynamic Differentiable Computational Graphs

Neural Turing Machine (NTM) BIBREF16 , BIBREF17 is a seminal work to implement instrument-based algorithm in the neural architecture, which attempts to express algorithms by simulating memory and controller. However, NTM leverages the weighting technique, which involves too much noise and makes the learned algorithm fuzzy. Thus, we propose a hard way to embed algorithms into neural architectures.

There also exist some papers for dynamical computational graph construction. At the lower level, pointer-switch networks BIBREF18 are a kind of dynamic differentiable neural model. At the higher level, some architecture search models BIBREF19 , BIBREF20 construct new differentiable computational graphs dynamically at every iteration.

Methodology

First, we introduce the basic components of our neural architecture. Then, we analyze the training process of Hungarian layer, that how to dynamically construct the computational graph.

Neural Architecture

Our neural architecture is illustrated in Figure FIGREF6 . Basically our model is composed by four components, namely, word embedding, bi-directional LSTM (BiLSTM), Hungarian layer and cosine similarity.

Word Embedding. The goal of this layer is to represent each word INLINEFORM0 in every sentence INLINEFORM1 with INLINEFORM2 -dimensional semantic vectors. The word representations, which are pre-trained by GloVe BIBREF21 , are unmodified within the learning procedure. The inputs of this layer are a pair of sentences as word sequences INLINEFORM3 and INLINEFORM4 , while the outputs are corresponding embedding matrices as INLINEFORM5 and INLINEFORM6 .

Bi-Directional LSTM (BiLSTM). The purpose of this layer is to transform lexical representations to hidden contextual representations. For hidden contextual encoding, we employ a parameter-shared bi-directional LSTM (BiLSTM) BIBREF22 to parse the word embeddings into hidden representations, mathematically as: DISPLAYFORM0

where INLINEFORM0 is the INLINEFORM1 -th hidden representation and INLINEFORM2 corresponds to the INLINEFORM3 -th word embedding in the source/target sentence or INLINEFORM4 / INLINEFORM5 .

Hungarian Layer. This layer, which is the matching component of our model, extracts the aligned unmatched parts from the source and target sentences. This layer is composed by two sequential stages.

Algorithm SECREF7 demonstrates the first stage. The objective of this stage is to align the source and target hidden representations. The inputs of this stage are INLINEFORM0 source hidden representation vectors INLINEFORM1 and INLINEFORM2 target hidden representation vectors INLINEFORM3 , while the outputs of this stage are INLINEFORM4 aligned hidden representation vector pairs INLINEFORM5 , assuming INLINEFORM6 , where INLINEFORM7 corresponds to the INLINEFORM8 -th aligned source/target hidden representation vector, respectively.

Specifically in this stage, there are totally three steps. First, the input hidden representations are crossly dotted to generate the pairwise similarity matrix INLINEFORM0 . Then, Hungarian algorithm works out the aligned source-target position pairs INLINEFORM1 with this similarity matrix. For example in Figure FIGREF1 , assuming the left/top sentence indicates the source/target sequence, the aligned source-target position pairs are listed as INLINEFORM2 . Last, the input hidden representation vectors INLINEFORM3 are re-organized into the aligned source-target hidden representation vector pairs INLINEFORM4 , according to the aligned source-target position pairs INLINEFORM5 .

The second stage attempts to extract the aligned unmatched parts by weighting the aligned hidden representations INLINEFORM0 from the first stage. Required by extracting the unmatched parts, if two aligned representations are matched, the weight for them should be small, otherwise, large dissimilarity leads to large weight. For this reason, we introduce cosine dissimilarity, mathematically as: DISPLAYFORM0

where INLINEFORM0 is the INLINEFORM1 -th aligned cosine dissimilarity and INLINEFORM2 is the INLINEFORM3 -th aligned cosine similarity from the first stage. Thus, the aligned hidden representations are concatenated and then weighted by cosine dissimilarity: DISPLAYFORM0

where INLINEFORM0 is the INLINEFORM1 -th output of Hungarian layer, INLINEFORM2 is the INLINEFORM3 -th aligned source/target hidden representation generated by Algorithm SECREF7 and INLINEFORM4 is the scalar-vector multiplication. Actually in the practical setting, most of cosine dissimilarity approach 0 and the remaining hidden representations indicate the aligned unmatched parts.

[t] Hungarian Layer: First Stage [1] Source and target sentence hidden representations: INLINEFORM0 and INLINEFORM1 . INLINEFORM2 , where INLINEFORM3 and INLINEFORM4 mean the INLINEFORM5 -th aligned hidden representations for source and target respectively, and INLINEFORM6 means the corresponding similarity. Generate the pairwise similarity matrix: INLINEFORM7

where INLINEFORM0 is the dot product and INLINEFORM1 is the length of vector. Perform Hungarian algorithm BIBREF5 to assign the aligned position pairs INLINEFORM2 , where INLINEFORM3 is INLINEFORM4 -th aligned source/target position of the sentence pair. INLINEFORM5 , where INLINEFORM6 is the length of source sentence. Compute INLINEFORM7 , where INLINEFORM8 is the pairwise similarity for INLINEFORM9 -th matched position. return INLINEFORM10 , where INLINEFORM11 corresponds to the INLINEFORM12 -th aligned source/target hidden representation, while INLINEFORM13 is the INLINEFORM14 -th aligned source-target position pair, INLINEFORM15 are the input source/target hidden representation vectors and INLINEFORM16 is the INLINEFORM17 -th aligned cosine similarity.

Cosine Similarity. Last, we average the concatenated hidden representations as the final sentence representation INLINEFORM0 , which is a conventional procedure in neural natural language processing, BIBREF4 . Then, we employ a cosine similarity as the output: DISPLAYFORM0

where INLINEFORM0 is the matching score, INLINEFORM1 is the length of vector and INLINEFORM2 / INLINEFORM3 is the corresponding source/target part of the final sentence representation INLINEFORM4 . Thus, our output ranges in INLINEFORM5 , where INLINEFORM6 means the two sentences are similar/paraphrase, and INLINEFORM7 means otherwise. For further evaluation of accuracy, we also apply a threshold learned in the development dataset to binary the cosine similarity as paraphrase/non-paraphrase. Notably, the introduction of concatenation layer facilitates the inference and training of Hungarian layer.

Training Hungarian Layer

Previously discussed, Hungarian algorithm is embedded into neural architecture, making a challenge for learning process. We tackle this issue by modifying the back-propagation algorithm in a dynamically graph-constructing manner. In the forward pass, we dynamically construct the links between Hungarian layer and the next layer, according to the aligned position pairs, while in the backward process, the back-propagation is performed through the dynamically constructed links. Next, we illustratively exemplify how the computational graph is dynamically constructed in Hungarian layer as Figure FIGREF13 shows.

As Figure FIGREF13 shows, in the forward propagation, Hungarian algorithm works out the aligned position pairs, according to which, neural components are dynamically connected to the next layer. For the example of Figure FIGREF13 , the 1st source and 2nd target word representations are jointly linked to the 1st aligned position of concatenation layer. Once the computational graph has been dynamically constructed in the forward pass, the backward process could propagate through the dynamically constructed links between layers, without any branching and non-differentiated issues. For the example in Figure FIGREF13 , the backward pass firstly propagates to the 1st aligned position of concatenation layer, then respectively propagates to 1st source and 2nd target word representations. In this way, the optimization framework could still adjust the parameters of neural architectures in an end-to-end manner.

Experiment

In this section, we verify our model performance on the famous public benchmark dataset of “Quora Question Pairs”. First, we introduce the experimental settings, in Section 4.1. Then, in Section 4.2, we conduct the performance evaluation. Last, in order to further test our assumptions, that the aligned unmatched parts are semantically critical, we conduct a case study for illustration in Section 4.3.

Experimental Setting

We initialize the word embedding with 300-dimensional GloVe BIBREF21 word vectors pre-trained in the 840B Common Crawl corpus BIBREF21 . For the out-of-vocabulary (OOV) words, we directly apply zero vector as word representation. Regarding the hyper-parameters, we set the hidden dimension as 150 for each BiLSTM. To train the model, we leverage AdaDelta BIBREF23 as our optimizer, with hyper-parameters as moment factor INLINEFORM0 and INLINEFORM1 . We train the model until convergence, but at most 30 rounds. We apply the batch size as 512.

Performance Evaluation

Dataset. Actually, to demonstrate the effectiveness of our model, we perform our experiments on the famous public benchmark dataset of “Quora Question Pairs” . For a fair comparison, we follow the splitting rules of BIBREF2 . Specifically, there are over 400,000 question pairs in this dataset, and each question pair is annotated with a binary value indicating whether the two questions are paraphrase of each other or not. We randomly select 5,000 paraphrases and 5,000 non-paraphrases as the development set, and sample another 5,000 paraphrases and 5,000 non-paraphrases as the test set. We keep the remaining instances as the training set. Baselines. To make a sufficient comparison, we choose five state-of-the-art baselines: Siamese CNN, Multi-Perspective CNN, Siamese LSTM, Multi-Perspective LSTM, and L.D.C. Specifically, Siamese CNN and LSTM encode the two input sentences into two sentence vectors by CNN and LSTM, respectively, BIBREF24 . Based on the two sentence vectors, a cosine similarity is leveraged to make the final decision. Multi-Perspective methods leverage different metric aspects to promote the performance, BIBREF2 . L.D.C model BIBREF4 is an attention-based method, which decomposes the hidden representations into similar and dissimilar parts. L.D.C is a powerful model which achieves the state-of-the-art performance.

We have tested L.D.C. and our model five times to evaluate the mean and variance, then perform the test for statistical significance.

INLINEFORM0 We apply t-test and INLINEFORM1 . Thus, the improvement is statistically significant.

Results. Our results are reported in Table TABREF17 . We can conclude that:

Our method outperforms all the baselines, which illustrates the effectiveness of our model.

In order to evaluate the reliability of the comparison between L.D.C and our model, the results are tested for statistical significance using t-test. In this case, we obtain a p-value = 0.003 INLINEFORM0 0.01. Therefore, the null hypothesis that values are drawn from the same population (i.e., the accuracies of two approaches are virtually equivalent) can be rejected, which means that the improvement is statistically significant.

Compared with Siamese LSTM BIBREF24 , which lacks the matching layer, our model could precisely align the input sentences. Thus, our method promotes the performance.

Compared with L.D.C. BIBREF4 , which is an attention-based method and still analyzes the dissimilar part, our model could exactly extract the aligned unmatched parts rather than the fuzzy dissimilar parts. Thus, our performance is better.

Notably, L.D.C. is a very complex model, which is beaten by our simple model within a statistically significant improvement. This comparison illustrates our model is indeed simple but effective. Thus it is very suitable for industrial applications.

Case Study

We have conducted a case study in the practical setting of “Quora Question Pairs” with our model for paraphrase identification. Illustrated in Figure FIGREF18 , the slashed grids correspond to the aligned matched parts, while the crossed ones indicate the aligned unmatched parts. Notably, we mark the pairwise similarity below INLINEFORM0 as unmatched in this case study.

For the example of (a), there exist two input sentences: “What is your review of Hidden Figures -LRB- 2016 movie -RRB-” and “What are your impressions of Hidden Figures -LRB- 2017 movie -RRB-”. From our case analysis, most of the aligned parts are matched, while minor aligned unmatched parts are similar. Thus, our method justifies the two sentences as paraphrase. This is accorded to our assumption.

For the example of (b), there exist two input sentences: “Why is saltwater taffy candy imported in Austria” and “Why is salt water taffy candy unknown in Japan”. There are two unmatched parts that “imported/unknown” and “Austria/Japan”, which are conflicted. Thus, the case is classified as non-paraphrase.

For the example of (c), the two sentences are: “How can I stop being addicted to love” and “How can I stop being so addicted to my phone”. From our case analysis, there is an extreme conflict that “love/phone”, making this case non-paraphrase, according to our assumption.

For the example of (d), the two sentences are: “Is a(n) APK file just a hidden app” and “Where do APK files get stored in Android Studio”. As we know, there are too many conflicts in this case, making a very dissimilar score as non-paraphrase.

In summary, this case study justifies our assumption that “the aligned unmatched parts are semantically critical”.

Conclusion

In this paper, we leverage Hungarian algorithm to design Hungarian layer, which extracts the aligned matched and unmatched parts exclusively from the sentence pair. Then our model is designed by assuming the aligned unmatched parts are semantically critical. Experimental results on benchmark datasets verify our theory and demonstrate the effectiveness of our proposed method.