**K. K. Wagh Institute of Engineering Education & Research, Nashik**

Department of Information Technology

**(Academic Year:2020-21)**

**Project Title: Text Matching for Question-Answer System**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Project Group No.: 01** | | |  | **Guide Name : Prof. R.M. Bora** |  |  |
|  |  |  |  |  |  |  |
|  | **GROUP MEMBERS:** | |  |  |  |  |
|  | **Roll No./** | **Name of Student** |  | **Project Area** | **Project Platform** |  |
|  | **Seat No.** |  |  |
|  |  |  |  |  |  |
|  | 22 | Gangurde Srushti |  |  |  |  |
|  | 30 | Kanchwala Abdulhussain |  | Machine Learning, | Django Framework |  |
|  | 32 | Karlekar Anuja |  | Deep Learning, Natural Language Processing |  |  |
|  | 35 | Kothawade Yashodhan |  |  |  |  |
|  | 36 | Kulkarni Amogh |  |  |
|  |  |  |  |  |  |  |

1. **Abstract**

For an automated Question-Answer Matching System, most important tasks are: to accurately map keywords in the model answer to user’s answer, to check the accuracy in text semantics, and evaluate the answer. This project of Q&A system implements deep semantic matching and extraction by matching information from different angles. It uses Multiple-Perspective Semantic Crossover (MPSC) model for modeling semantic-based match of questions and answers. Inputs to the MPSC model are text files including: Model Question set, Model Answer set, Marks Distribution provided by Administrator and Answer Sheet provided by Student. The model is responsible for calculating matching degree between questions and probable answers. On the basis of this matching degree, system generates output that is an approximate score based on syntactic, semantic and pragmatic relevance of the answer. The MPSC model generates a compact semantic crossover by referring to the text, eluding the limitations of word vector and vector matrices. It integrates Neural Network Models like LSTM and CNN for semantic analysis and accurate feature extraction from the text. This project also uses Binary Cross Entropy as an objective loss function for identifying duplicate questions or answer sentences.

**Keywords:** Question-Answer Matching, Neural Network, LSTM, CNN, Binary Cross Entropy, Adadelta optimizer.

**B) Objectives**

* To build a working model of question and answer matching.
* To reduce workload on examiners.
* To minimize human error.
* To accelerate the task of assessment of papers.
* To provide unbiased paper evaluation.

1. **Scope**

To implement an optimize Question Answer matching system having a better accuracy and a higher degree of similarity. The aspects to be under consideration are:

* Text semantic analysis
* Text feature extraction
* Matching degree calculation
* Scoring/Grading of answer

**D) Literature Survey**

**B. Hu, Z. Lu, H. Li, and Q. Chen. "Convolutional neural network architectures for matching natural language sentences." In Advances in Neural Information Processing Systems, 2014.**

**ARC-I:**

Architecture-I (ARC-I), takes a conventional approach: It first finds the representation of each sentence, and then compares the representation for the two sentences with a multi-layer perceptron (MLP).It is based on a single-text that is trying to represent the text with a vector and then calculating the similarity of the text vector. Although ARC-I enjoys the flexibility brought by the convolutional sentence model, it suffers from a drawback inherited from the Siamese architecture: it defers the interaction between two sentences, therefore runs at the risk of losing details (e.g., a city name) important for the matching task in representing the sentences. In other words, in the forward phase (prediction), the representation of each sentence is formed without knowledge of each other. This cannot be adequately circumvented in backward phase (learning), when the convolutional model learns to extract structures informative for matching on a population level.

**ARC-II:**

In view of the drawback of Architecture-I, we propose Architecture II (ARC-II) that is built directly on the interaction space between two sentences. It has the desirable property of letting two sentences meet before their own high-level representations mature, while still retaining the space for the individual development of abstraction of each sentence. ACR-II performs modelling directly and it considers that the two texts can be interacted earlier, and extracting deep interactive information is more conductive to solve text matching problems. Basically, in Layer-1, we take sliding windows on both sentences, and model all the possible combinations of them through “one-dimensional” (1D) convolutions. Clearly the 1D convolution preserves the location information about both segments. After that in Layer-2, it performs a 2D max-pooling in non-overlapping 2 X 2 windows. In Layer-3, we perform a 2D convolution on k3 X k3 windows of output from Layer-2. This could go on for more layers of 2D convolution and 2D max-pooling, analogous to that of convolutional architecture for image input.

**Liang Pang Yanyan Lan Jiafeng Guo Jun Xu Xueqi Cheng. Manning. 2018.Stanza: "A Study of MatchPyramid Models on Ad-hoc Retrieval"**

**MatchPyramid:**

A recently introduced deep matching model, namely MatchPyramid, which has shown state-of-the-art performances on several text matching tasks. In MatchPyramid, local interactions between two texts are first built based on some basic representations (e.g., word embeddings). The local interactions represented by a matching matrix is then viewed as an image, and a convolutional neural network (CNN) is employed to learn hierarchical matching patterns. Finally, the high-level matching patterns are fed into a multilayer perceptron to produce the matching score between the two texts. The three parts MatchPyramid are as follows:

1. Matching Matrix:

Matching Matrix is a two-dimension structure where each element Mij denotes the similarity between the i-th word wi in the first piece of text and the j-th.

2. Hierarchical Convolution:

Based on the Matching Matrix, MatchPyramid conducts hierarchical convolution to extract matching patterns. Hierarchical convolution consists of convolutional layers and dynamic pooling layers, which are commonly used in CNN (such as AlexNet, GoogLeNet) for image recognition tasks.

3. Matching Score Aggregation:

After hierarchical convolution, 6 two additional fully connected layers are used to aggregate the information into a single matching score.

**D. Peng, S. Wu and C. Liu, "MPSC: A Multiple-Perspective Semantics-Crossover Model for Matching Sentences," in IEEE Access, vol. 7, 2019**

**Text Matching:**

Text matching often starts with embedding a text as a vector in a vector space, called embedding representation. The vector is formed by a sequence of numbers, each of which representing a characteristic distribution of the text in a certain dimension. The most primitive embedding representation is the One-Hot encoding, which is simple but has two major shortages: 1. Supposition that the words are independent from each other both semantically and grammatically 2. As the lexicon grows, the dimensions of the vector increase drastically so that a dimensional disaster will burst. To overcome these drawbacks, Hinton proposed a distributed representation of word vectors, exploiting a fixed-length vector to represent words. This idea has been realized in Word2Vec. In 2015, Shengxian Wan, et al. proposed the MV-LSTM model to fuse context information into word vectors and further capture the contextual information of the text. Using embedding representation, sentence is vectored to generate the word vectors at first, and then add or average them to form the sentence vector. Although this approach is simple, it is effective only for short texts and not long texts. In another representation method, after the words are segmented, they are represented by vectors and combined into a matrix also called as interaction tensor to represent the sentences. Researchers designed deep neural network models to acquire the vector representation of sentences, such as sentence modeling by RNN and convolutional neural network (CNN). MPSC model has fused convolutional neural network (CNN) and bidirectional long short-term memory (Bi-LSTM) network which performs efficient semantic analysis on the question pairs to extract more effective features of the text. LSTM is a special type of RNN, combining two text matching methods: Multi-Semantic and Direct Modeling.

**Multi-Perspective Semantics:**

In natural language processing, words are often mapped to vectors of real numbers. This job is accomplished by the technology of word embedding, which is currently used widely and has some existing tools, such as Word2vec and Glove. These tools exploit unsupervised approaches to generate vectors for words by the use of a large amount of texts. Studies have shown that the word vectors obtained by these tools are universal, and to a large extent, can bring more information to the model that is based on them. In this project, word vectors are referred as the original semantics of a sentence. Original semantics are a part of the input to MPSC model. A recurrent neural network (RNN) is able to learn a sequential representation for each text. Long Short-Term Memory (LSTM) network is one of the popular variations of RNN and is widely used in text representation. The average of the hidden states in LSTM cells is used as text representation. A bidirectional LSTM (Bi-LSTM) exploits LSTM cells through both directions to get text representation. Preparation of semantics stage is responsible for computing the interaction information between each sentence. The interaction information includes original semantics, forward semantics, and backward semantics. Of these, original semantic is follow sequence of the given sentence as it is. On the other hand, forward and backward semantics

are computed by the LSTM cells. It is done by scanning given text in both the directions to obtain backward and forward semantics.

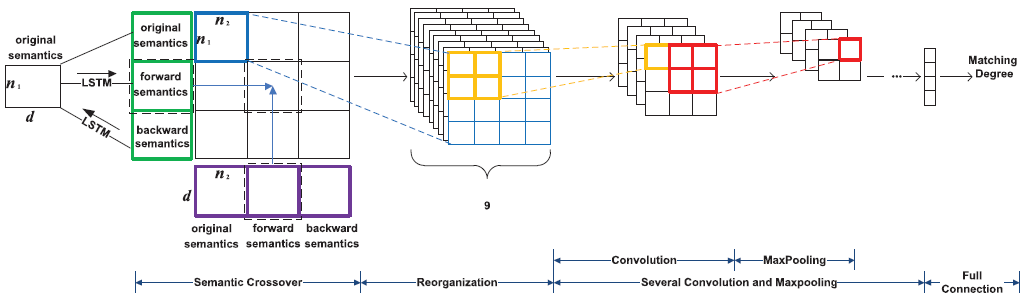


Figure 1: Structure of Multi-Perspective Semantics-Crossover (MPSC) model

**Long Short-Term Memory (LSTM) model:**

Long Short-Term Memory (LSTM) networks are a modified version of recurrent neural networks, which makes it easier to remember past data in memory. The vanishing gradient problem of RNN is resolved here. LSTMs are explicitly designed to avoid the long-term dependency problem as well. The key to LSTMs is the cell state which could be imagined as a conveyor belt running straight down the entire chain. LSTM has the ability to remove or add information to the cell state, carefully regulated by structures called gates. Gates are a way to optionally let information through. There are three gates in LSTM composed of a sigmoid neural net layer and a pointwise multiplication operation: 1. Input gate(it) 2. Forget gate(ft) 3. Output gate(ht).

**Abdalraouf Hassan, Ausif Mahmood, "Convolutional Recurrent Deep Learning Model for Sentence Classification", in IEEE Access, vol. 6, 2018**

**Convolutional Neural Network (CNN):**

The CNN layer is responsible for extracting the most influential semantic aspects from the text. CNN performs feature extraction and classification as one joint task. A convolutional neural network is used to refine the interaction tensor wherein the matrix composed of feature vectors will be processed effectively. Each convolution layer comprise of nonlinear activation function, such as ReLU or Tanh, applied to the result. Output of each layer becomes an input to it’s successive layer. Using the interaction information from the previous stage, multiple Convolution layers as well as multiple Pooling layers are employed to extract the interactive features. In the matching degree calculation, a fully connected layer is built for calculating matching degree. The convolution and the max pooling operation in the convolutional layer will be utilized to capture more meaningful information and discard the rest of the irrelevant information. Single or several rounds of convolution and pooling layers can be employed depending on size of data being considered or accuracy of the output required.

CNN has three layers as follows:

1. Convolution

2. Pooling

3. Fully connected/affine layer

**E) Deliverables**

* Web Application with Admin and Student Login
* Build Machine Learning Model ( .h5 file)

**F) Resource Requirements (Hardware/Software etc.).**

* Hardware
  + GPU
* Software
  + Python3
  + OS: Window 8 and above, Linux
  + Internet connectivity