Information Retrieval and Extraction

Assignment 3 - Detecting Well-Formed Search Queries

Archit Jain (2019101053)

Question 1

What is the problem that the authors are trying to solve and why is it significant?

- **Problem :-**
  Author wants to label a search query whether its a well-formed natural language questions or not and training a model exiplity for this is an expensive task, as well as trained model can give a unsual predication if any rules changed in querry domain. So to solve both these problems, author uses transfer learning technique by fine-tuning on a pretrained language model(BERT) for identifing whether a search query is a well-formed natural language question or not.
- **Significant:-**
  It is important to solve this problem since nowadays of voice based search, verbose queries have become quite popular and we use deep learning algorithms to process such verbose queries efectively. But if a querry is not well formed it might adversely impact the downstream pipeline which processes these queries. Classifation of query can aids heavily in reducing errors in downstream tasks and further helps in improved query understanding.

Question 2

What is transfer learning?

Transfer lerainning is a process of using knowledge of a model trained on one task and applting it on a different yet similar task. The weights in one or more layers from a pre-trained network model are reused in a new model and either keeping the weights fixed, fine-tuning them, or adapting the weights depends entirely when training the model.
This approach takes comparatively less time as well as reduces the resources and amount of labelled data required to train new models .It can be used for tasks where the datasets have too little data to train a full-scale model from scratch due to its rapid progress or improved performance when modelling for the second task.

Question 3

Explain the architecture of the paper in your own words.

The proposed Inductive Transfer Learning architecture constitutes 3 phases

1. LM Pre-training:
   It involves the pretrained language model trained on a large dataset such as Wikitext-103 dataset consisting 103 Million unique words and 28,595 preprocessed Wikipedia articles that will help our model to learn the general language dependencies and is the first step before fine-tuning which targets task-specific data.

2. LM Fine-tuning:
   Here they have used task-specific dataset to fintune the pretrained model since, it is essential information for the language model no matter how diverse the general domain data in the earlier pretraining step is. They also used variale learning rate such as discriminative finetuning and slanted triangular learning rates to combat the catastrophic forgetting language models since, it is essential information for the language model no matter how diverse the general domain data in the earlier pretraining step is.

3. Classifer Fine-tuning:
   The weights we obtained from the second phase are finetuned by keeping the same upstream architecture and also appended 2 fully connected layers for final classification with the last layer predicting the well-formedness rating.
   Along with they had used gradual unfreezing where the last layer is first unfrozen and finetuned for one epoch and subsequently, the next frozen layer is unfrozen and all unfrozen layers are fine-tuned. This is repeated until all layers are fine-tuned until convergence is reached.

   Discriminative Fine-tuning (DFT): a different learning rate is used for tuning the three different layers, since each of the layers may represent a different kind of information.

   Slanted Triangular Learning Rates (STLR): it uses slanted triangular learning rate which first increases the learning rate and then linearly decays it as the number of training samples increases.

   Gradual Unfreezing (GU): all layers are not fine-tuned at the same time, instead the model is gradually unfrozen starting from the last layer, as it contains the least general knowledge

Question 4

Discuss and describe the dataset

Dataset is a set of 25,100 queries from the Paralex corpus (Fader et al., 2013) annotated with human ratings of whether they are well-formed natural language questions. Every query was annotated by five raters each with 1/0 rating of whether or not the query is well-formed and for each query it has the average of the 5 binary judgements as the wellformedness score(rated as 0, 0.2, 0.4, 0.6, 0.8, 1). In this paper a query is considered well-formed if rating if greater than equal to 0.8. It further splitted into train, dev and test which consists of 17500 training, 3750 development and 3850 test queries.

Question 5

Describe the results and takeaways from the paper.

- **Results:-**
  The model formed in this paper has accuracy of 75.03% which a significant improvement form its baseline model State-of-the-art. More broadly the model without Language model finetuning performs bad than the model fintuned using

Discriminative Fine-tuning (DFT) and Slanted Triangular Learning Rates (STLR), and gradual unfreezing helps too improving accuracy.

- **Takeaways:-**
  The idea of using inductive transfer learning by fine-tuning language models aids in identifying whether search queries are wellformed natural language questions.

Question 6

- Max padding length used = 25
- LM finetuning by using “query-wellformedness”
- Used BERT-uncased
- Epoch = 4
- Learning Rate = 1e-5
- Batch Size = 32
- MCC Score: 0.635
- F1 Score: 0.784
- Accuracy Score: 0.811

Question 7

Now, how can the same architecture be *extended to predict the rating of the query*? The rating has to be predicted in steps of 0.2 from 0 to 1, with 0 indicating a bad query and 1 indicating a perfect query. (i.e. the BERT model has to predict query well formedness rating instead of classifying for the same in a binary fashion).

- **Using Bert for Classfier**
  - Different Label are 0.0, 0.2, 0.4, 0.6, 0.8, 1.0
  - For 6 label we can change the argument value in of parameter `num_labels`
    ```
    model = BertForSequenceClassification.from_pretrained(
        "bert-base-uncased",
        num_labels = 6,
        output_attentions = False,
        output_hidden_states = False,
    )
    ```
  - We can enter the inputs with 6 different labels as 0.0, 0.2, 0.4, 0.6, 0.8, 1.0 instead of binary labels and can get the output for the same.
- **Using Bert for Regression**
  - We can use “BertRegresser” for getting output in range 0 to 1 and then later can quntaize the output accordingly like [0-0.10), [0.10-0.30), [0.30-0.60), [0.6-0.90), [0.90-1] as different label for 0, 0.2, 0.4, 0.6, 0.8 and 1.