What Defines a Good Stack Overflow Answer Post: An Automated Post Rater

Yanhao Jiang, Yanpei Tian, Chunyue Wei

jiangyh@stanford.edu, yanpeit@stanford.edu, weicy18@stanford.edu

Stanford Computer Science

Abstract

Question and Answer websites, like Stack Overflow, are extremely important online resources that programmers rely on to seek inspiration, find solution and help each other. However, many of the question owners do not select their preferred answer, which makes it very inconvenient for other viewers to quickly identify useful information.

This project attempts to address this issue by building an automated agent that identifies “good” answers by taking input from the raw post archive retrieved from Stack Overflow public database. With the implementation, about 70% prediction accuracy can be achieved.

Data Collection

Raw data of answer posts with ‘Python’ tag were generated using SQL query from online Stack Overflow database, including answer and question post bodies, comments, and user profiles. Labels are identified by whether the answer was accepted by question owner. The question posts with no accepted answer are excluded to ensure all data is labeled.

Training dataset (46287 samples)	Test dataset (486 samples)
01/01/2018 to 05/01/2018	06/01/2018

Feature Extraction

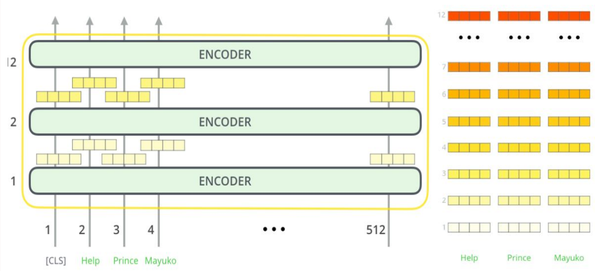

Figure 1: Bidirectional Encoder Representations from Transformers (BERT) Model

- **Language features** (e.g. fluency, representation) are extracted using BERT and N-gram model.
- **Technical features** (e.g. code, URLs) are extracted by identifying specific HTML tags.

Features (13 + 1536)

Language (3 + 1536)
- ❑ Length: # words
- ❑ Text Fluency (2)
  - Unigram/ Bigram
- ❑ BERT (1536)
  - Answer/ Question

Technical (4)
- ❑ Inline Code
- ❑ Block Code (2)
  - Count/ Lines
- ❑ Hyperlink

Non-texual (2)
- ❑ *Comment count*
- ❑ *Edit: binary*

Note: Italic features are the raw input data and other features are derived.

User (4)
- ❑ *Reputation*
- ❑ *User views*
- ❑ *Upvotes*
- ❑ *Downvotes*

Machine Learning Model

- ❑ Multiple models have been implemented including neural network, logistic regression, and SVM.
- ❑ First 2 layers of Neural Network reduces vast Bert embeddings into 70 features.
- ❑ Concatenated compressed bert output with 13 original features to increase original features’ impact to the final classification.
- ❑ Backward Propagation:

$output\ layer: \delta^{[N]} = \nabla_{z^{[N]}} L(y', y),\ other\ layers: \delta^{[l]} = \left(W^{[l+1]T}\delta^{[l+1]}\right) \cdot g'(z^{[l]})$

gradients for layer l: $\nabla_{W^{[l]}} J(W,b) = \delta^{[l]} a^{[l-1]T};\quad \nabla_{b^{[l]}} J(W,b) = \delta^{[l]}$

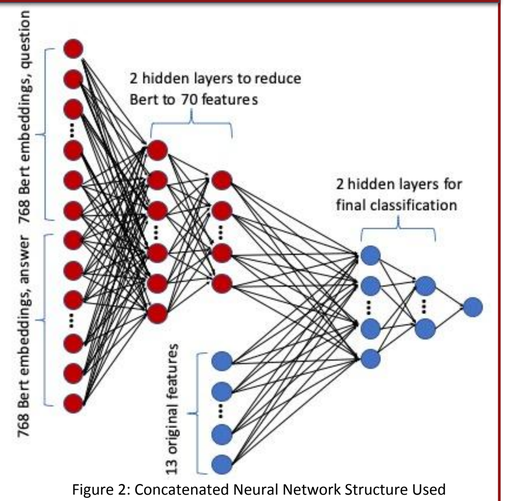

Figure 2: Concatenated Neural Network Structure Used

Results

Training and Testing accuracy for different models:

Model	Training Accuracy	Test Accuracy
Logistic Regression w/o BERT	0.67	0.65
Logistic Regression w/ BERT	0.67	0.65
SVM w/o BERT	0.65	0.67
SVM w/ BERT	0.79	0.68
Simple NN w/o BERT	0.75	0.70
Simple NN w/ BERT	0.98	0.61
Hierarchical NN w/ BERT	0.67	0.70

Discussion and Future Work

With Bert model, N-gram, and 11 other engineered features, the highest prediction accuracy for “good” post can be achieved is around 70%. This result meets the initial expectation. Since the models do not have any contextual prior knowledge about the topics, inferring correctness of the answers from the question itself would be difficult. For future work, other models, such as various decision trees could be tested. In addition, more useful features need to be explored to incorporate the prior domain knowledge into the model.

References

[1] Omondiagbe O. P. , Licorish S. A., and MacDonell S. G. (2019). Features that predict the acceptability of Java and Javascript answers on stack overflow, ACM Int. Conf. Proceeding Ser., pp. 101–110.
[2] Alammar, J. (2019). *The Illustrated BERT, ELMo, and co. (How NLP Cracked Transfer Learning)*. [online] Jalammar.github.io. Available at: http://jalammar.github.io/illustrated-bert/ [Accessed 11 Dec. 2019].
[3] Mccormickml.com. (2019). *BERT Word Embeddings Tutorial · Chris McCormick*. [online] Available at: https://mccormickml.com/2019/05/14/BERT-word-embeddings-tutorial/ [Accessed 11 Dec. 2019].
[4] Mccormickml.com. (2019). *BERT Fine-Tuning Tutorial with PyTorch · Chris McCormick*. [online] Available at: https://mccormickml.com/2019/07/22/BERT-fine-tuning/ [Accessed 11 Dec. 2019].