CS 247: Project Proposal

Anton Lykov
University of California, Los Angeles
Los Angeles, California

Allen Miyazawa
University of California, Los Angeles
Los Angeles, California

Kai Wong
University of California, Los Angeles
Los Angeles, California

Derek Xu
University of California, Los Angeles
Los Angeles, California

1 INTRODUCTION

1.1 Problem setting

The problem we decided to tackle is “Keyword Extraction and Tagging in Community-Based Question Answering”. Scientific forums and QA-resources, such as “StackOverflow”, have become an important part of our everyday life: people use them for work, research, self-development and many other purposes. Recently, the amounts of data stored on such sites has increased drastically, and continues to grow. Thus, it is important to develop techniques to classify, search, and provide fast access to this data upon user request. One of the natural ways to cluster questions and answers in groups is by tagging posts, for example with the most relevant keywords. However, tagging must be done smartly, so that future retrieval of data is easy and straightforward.

1.2 Goal

Our goal is to develop a novel algorithm to solve this problem. To this day, several approaches have been taken already (see [1]). We will be using techniques from the course, published papers on this and similar topics, as well as other resources and libraries. In particular, we are researching such techniques as TextRank, TF-IDF scoring and possibly other methods. We will be working in Python; besides the basic scientific tools for data mining, we will most likely be using Python module **nltk** for natural language processing.

The high-level description of the algorithm is the following: a short text (a question, perhaps) is given as input, and few (tentatively 1-5) keywords are given as output. We will be using supervised learning, namely a collection of tagged texts. We expect our algorithm to perform well on texts that come from the similar sources as training data. We are also planning to use two datasets from past Kaggle competitions: [3] [2]. Possibly, we will also find other sources for testing our algorithm.

2 SOLUTION PLAN

The detailed pipeline would depend on the algorithm we use. Fortunately, the availability of standardized datasets from Kaggle allows us to focus on model selection and development. An approach that we will be focusing on is based on Deep Keyphrase Generation [5] due to the strong empirical performance reported. The authors proposed a generative model for key phrase prediction using an RNN-based encoder-decoder framework trained on scientific metadata in the computer science domain to learn universal language features. Their model was tested on several scientific publication datasets such as SemEval-2010, and we intend to compare the paper’s reported test results with our tests using the StackOverflow dataset. Because the content of the model’s training data and the StackOverflow dataset is relatively similar, we expect comparable performance. As a baseline, we will first test the author’s original model without any modifications.

Since the training dataset has a significantly richer corpus and dimensionality in comparison to the StackOverflow datasets, we suspect that better domain-specific performance can be achieved via transfer learning. While studies in general are still inconclusive regarding the efficacy of transfer learning techniques for deep NLP models unlike the success enjoyed by convolutional nets [6], suggest that when the semantics of the target and original datasets are similar, freezing the embeddings and hidden layers before retraining on the target dataset can fine-tune the output layers for the new domain. Additionally, we will also consider other new approaches we come across if they are promising in order to compare with this model.

3 DATA PLAN

In order to solve this problem, we need a dataset that contains questions, answers for each question, and tags on the question and answers. Three years ago, Stack Overflow released a dataset with the text of 10% of questions and answers from the Stack Overflow programming Q&A. The dataset is organized into three tables:

(1) Question: contains the title, body, creation date, closed date, score and owner ID
(2) Answer: contains the body, creation data, score, and owner ID

(3) Tag: contains the tags on each of the questions

First, we can clean the dataset by getting rid of the questions (and their answers) with no tag and potentially the questions with no answers. Next, we may reduce the dataset by getting rid of unnecessary attributes and be left with the title, body and score (possibly) for Question and the body and score for the Answers. In addition, in order to eliminate faulty data, we can eliminate some answers for certain questions. Potentially, we select the top 3 answers (answers with the most scores) for each question to be included in the dataset.

4 EVALUATION

Keyword extraction introduces several interesting problems from the evaluation perspective. First, each document may contain multiple keywords inside. Each set of keywords related to a document is unordered. Thus, a direct accuracy score may be difficult unless our model can both predict which keywords are most likely and how many keywords match the document. Our goal is similar to the tasks such as machine translation, in the sense that we have outputs of arbitrary length. But unlike machine translation, there is no order. In cases like these, the evaluation metric is even less obvious. We must consider the goal of our model: to predict a fixed set of correct keywords, to predict which keywords are most likely, or to predict whether individual words are keywords. We must also consider the cases where our model is partially correct, for instance, predicting 4 out of 5 keywords correctly may be better than predicting 2 out of 3. If our model generates probabilities for each keyword, we must consider whether we should penalize wrong guesses with high certainty more than wrong guesses with low certainty. Finally, our goal is to argue that our model is better than alternatives. Thus to accurately compare our model with baselines, we must consider the current evaluation metrics being used in literature, whether the our model solves the same issues as the baselines', and whether those evaluation metrics are statistically sound.

As we are still currently designing our model, it is unclear what specific evaluation metric we should use. Some possible candidates are accuracy, precision, and recall. A possible evaluation method is to, instead of treating the outputs as a variable size set of keywords, to treat it as whether we could properly classify individual keywords. This way we can check our model outputs individually and apply more standard statistic measures for model performance. Once our model architecture becomes more clear, we may be able to draw some inspiration for metrics from other domains which measure variable length output.

There have been 2 previous papers on the topic of keyword-extraction: [5], [4]. These papers utilize machine learning and data mining models, making them candidates for our baselines. The first paper utilizes KNN and SVM to complete the task. The second uses some sort of attention mechanism. Both related works used the same evaluation metric. The model predicts a list of words with certain probabilities. From this list the words with the *n* highest probabilities are chosen. The F1 score of these *n* words appearing in the given keywords set is the metric used. As discussed in CS145, the F1 score is a statistical measure incorporating precision and recall.

5 SCHEDULE

Schedule	
Week	Tasks
1-3	NA
4	Read more literature on the topic. Begin coming up with candidate models. Begin cleaning the dataset Begin writing the baselines.
5	Continue cleaning the dataset. Continue writing the baselines.
6	Finish cleaning the dataset. Begin writing the model. Continue writing the baselines.
7	Finish writing the model. Finish writing the baselines.
8	Begin writing the report. Begin making presentation. Running experiments. Hyperparameter tuning.
9	Continue writing the report. Continue making presentation. Running experiments.
10	Continue writing the report. Finish making presentation.
Finals	Finish writing the report.

REFERENCES

[1] Slobodan Beliga. 2014. Keyword extraction: a review of methods and approaches. *University of Rijeka, Department of Informatics* (2014), 1–9.

[2] Kaggle Competition. 2016. https://www.kaggle.com/stackoverflow/stacksample

[3] Kaggle Competition. 2017. https://www.kaggle.com/stackoverflow/stacklite

[4] James Hong and Michael Fang. 2013. Keyword extraction and semantic tag prediction. *unpublished(http://cs229. stanford. edu/proj2013/FangHong-Keyword% 20Extraction% 20and% 20Semantic% 20Tag% 20Prediction. pd f)* (2013).

[5] Rui Meng, Sanqiang Zhao, Shuguang Han, Daqing He, Peter Brusilovsky, and Yu Chi. 2017. Deep Keyphrase Generation. *CoRR* abs/1704.06879 (2017). arXiv:1704.06879 http://arxiv.org/abs/1704.06879

[6] Lili Mou, Zhao Meng, Rui Yan, Ge Li, Yan Xu, Lu Zhang, and Zhi Jin. 2016. How Transferable are Neural Networks in NLP Applications? *CoRR* abs/1603.06111 (2016). arXiv:1603.06111 http://arxiv.org/abs/1603.06111