Automating Question-and-Answer Session Capture using Neural Networks

Zhen Wei
ginny.weizhen@gmail.com
Imperial College London

Yuan Gao
meseta@gmail.com
University of Oxford

Jingqing Zhang
jingqing.zhang15@imperial.ac.uk
Imperial College London

ABSTRACT

Real-time chat conversations and community question-and-answer websites play an increasingly important role in alternative options for higher education and self-directed learning for adults, however without curation of knowledge that published textbooks and syllabuses enjoy, information is difficult to find. In this paper we explore using Seq2seq, a LSTM based encoder-decoder algorithm from deep learning to tag real-time chat conversations that occur in the help section of a programmer community, with the intention of then applying automatic summarizing and indexing. We compare Seq2seq against traditional NB and SVC methods. The result show that Seq2seq/LSTM outperforms NB and SVC, though larger datasets are needed.

CCS CONCEPTS

• **Applied computing → Computer-assisted instruction**; **Collaborative learning**; **E-learning**.

KEYWORDS

lstm, neural networks, nlp, text tagging

1 INTRODUCTION

The internet has been revolutionary as a source of knowledge: a vast amount of knowledge is available to anyone to seek out; whether in the form of scientific papers, or documentation and guides, or instructional videos. The information available on the internet can present a real alternative or supplement to higher education, allowing adult learners to develop the skills they need for a career without going through traditional higher education routes which can incur large student debts.

The wealth of information available on the internet is often likened to a vast library of information accessible at one's fingertips; but, that analogy is now out of date. The internet is no longer just connects learners with a piece of information; it connects learners with real live teachers.

Scientific papers, documentations, and instructional videos are all forms of non-interactive knowledge - a learner accesses this information; and should they have further questions or desire discussion, they go elsewhere. But with the internet being an increasingly large part of our lives; and with people spending longer on the internet, perhaps even constantly connected to it via smartphones and via push-notifications (a way to be notified in real-time about events or messages); it is increasingly possible to exchange information using the internet in an interactive way. Learners can connect and exchange messages with a teacher (or anyone willing to share instruction or knowledge, not necessarily in a formal teaching context), and should the learner have further questions, then clarification could be sought; or should the teacher feel the learner has some incorrect assumptions that would lead them to the wrong conclusion, then they can raise it rather than let the learner walk away with an incorrect understanding. This represents not only access to living knowledge at an unprecedented scales, but also allows interacting with the experience of these teachers and not just their knowledge.

To date, several prominent online community-based question-and-answer (cQA) websites exist, such as Stack Overflow, and Quora. In both, those seeking answers can post a question, and later those who are able provide answers can do so. The original asker (or any others joining the conversation) can then engage in further discussion by adding comments. This provides interaction, but often with an understanding that any discussion will happen with delays of perhaps hours to days. The consequence of these delays is that sometimes the question asker eventually figures it out themselves, or abandons the task; or answerers providing an answer but neglecting to reply to further enquiries. This is a problem in the case of tech support or questions that arise during programming, which moves fast enough that delays of hours or days can be highly disruptive to accomplishing a task.

A more real-time version of this kind of interaction exists, online-chat. A medium that is almost as old as the internet itself, with prominent programs like IRC (Internet Relay Chat) having been around since the 1980s. While IRC has been around for a long time, it's really only recently that real-time text chat has become increasingly used as a medium for technical support and asking and answering question. Its benefits include being a more approachable and natural way of seeking information[26].

Both real-time chat and cQA websites however suffer from a common problem - duplication, where similar questions are posted, and answers that could be re-used to answer other questions, for example Liu et al 2018 [17] found that at least 78% of cQA best answers were reusable. Summarization and de-duplication manually is time-consuming[7], so there have been efforts to use machine learning and automation to apply these to cQA websites[29], however to date little has been done to study the same problem in real-time chat.

DL4Ed, August 04–08, 2019, Anchorage, AK
© 2019 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Real-time chat poses new challenges for indexing, summarizing, and de-duplication: unlike cQA websites where questions, answers, and discussions are naturally separated from each other by the workflow of using these services, in real-time chat there is no structure imposed other than that of our natural languages; furthermore partly due to these challenges, real-time chat discussions are not often indexed by search engines, putting them further out of reach of those seeking information.

This paper will explore the use of deep learning with LSTM as a means to automatically detect the start of a question being asked in an online chat conversation, and thereby overcoming the challenges that real-time chat poses. When combined with other models to detect the end of a conversation, existing summarizing techniques [20], indexing, and search can be used to make question-and-answer sessions conducted over real-time chat to be as easily and reliably searchable as cQA websites.

2 DATASET

The chat application Discord[1] is a growing and increasingly popular choice for programmer communities, with one of the larger Python communities[2] boasting over 17,000 members at the time of writing, with several thousand members typically online at any given time. The community has an area dedicated for asking questions about python, and for members to help each other to resolve their programming questions and programming problems.

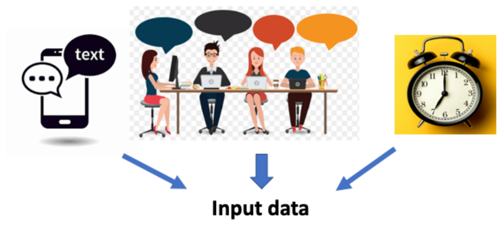
A dataset from this Python Discord server is collected using the software DiscordChatExporter[3]. Two datasets were collected for use in this paper, one spanning a day, which includes 708 messages; and another spanning five days, which includes 8043 messages.

The dataset is exported as CSV, and includes the following data (Figure 1):

- Author (who wrote the message)
- Timestamp (with accuracy in minutes)
- Message (the text content of the user's message)

Each message was labelled by hand with a binary outcome of whether the message could be considered the start of a new question.

Figure 1: Input dataset options



[1]https://discordapp.com/
[2]https://pythondiscord.com/
[3]https://github.com/Tyrrrz/DiscordChatExporter

3 METHODOLOGY

3.1 Problem definition

The dataset contains the logs for a real-time chat service in which many people can freely chat, the topic of the chat is for users to ask questions about Python programming, and for others to answer the questions if they are willing and able to do so. The chat is in English and does not have any structure imposed on it - participants can write a message at any time.

Two methods have been applied to the data; the first is an embedding of the text in the deep neuron layers, then using NLTK[18, 24] and Gensim[21] to further sort the text. In the second method, a hash function[4] [27] is applied to the text to create an input vector for each message. All rows are normalized to the maximum row's size of 254 words; and with 708 rows in the one-day dataset, the resulting final matrix is of dimensions 254×708. With the 8043 rows in five-day dataset, the resulting final matrix is of dimensions 254×8043.

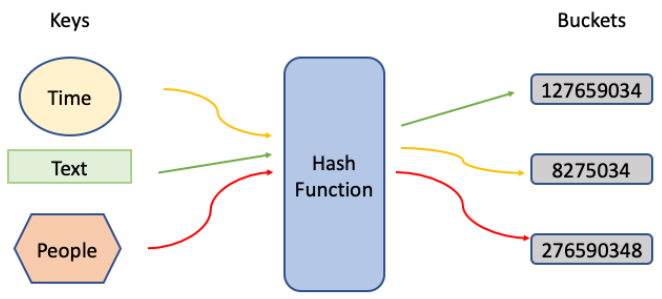
Applying the hash function[27] to the username allows the name to also be another dimension. The timestamp serves as another dimension. Therefore the input dimension are:

- 254 input dimensions: only the text

  (1) text $\xrightarrow{Hash}$ number

  (2) text $\xrightarrow{embedding}$ NLTK/Gensim $\xrightarrow{Hash}$ number
- 255 input dimensions: text+people/text+time
- 256 input dimensions: text+people+time

Figure 2: Hash function



The one-day dataset is divided into three cases of training and testing datasets split by time; for example the first 30-minute case consists of data from the first 23hour 30 minutes used for training; and data from the last 30 minutes used for testing. Note that the time and length do not vary linearly as it depends on how many people were writing messages during that time.

- 30-min case: the length of the training dataset is 687 (97.0%) and the length of the testing dataset is 21 (3.0%)
- 60-min case: the length of the training dataset is 654 (92.4%) and the length of the testing dataset is 54 (7.6%)

[4]https://www.programiz.com/python-programming/methods/built-in/hash

- 90-min case: the length of the training dataset is 650 (91.8%) and the output length of the output sequence is 58 (8.2%)

Time	30-min	60-min	90-min
Training size	687	654	650
Testing size	21	54	58

The five-day dataset is divided into training, validation, and testing datasets, similarly split by time: 3 days worth of data are used for training, 1 day for validation, and the final day for testing. Again, note that the time and length do not vary linearly as it depends on how many people were writing messages during that time.

- 1st to 3rd day case: the length of the training dataset is 6070 (75.5%)
- 4th day case: the length of the validating dataset is 1024 (12.7%)
- 5th day case: the length of the testing dataset is 949 (11.8%)

Training data size	Validating data size	Testing data size
6070	1024	949

The output dataset is labeled by 6 different people, who decide whether each row of text constitutes the start of a question or not; with numerical value 1 or 0 representing the two cases. The final result is a rounding number of the average of these six people answers: $y = rounding(\frac{1}{n}\sum_{i=1}^{n} p_i)$.

4 EXPERIMENTS

In this section, we describe the compared methods, implementation details and evaluation metrics. We also discuss the results of different models.

4.1 Compared methods

We compared out proposed model with the following methods.

- Naive Bayes classification[15]: NB is a traditional machine learning method used in classification tasks in education topics;
- Support Vector classification[22, 30]: SVC is a version of SVM, also with past applications in classification tasks in education topics;
- Deep Neural networks [19] include the Seq2Seq model, which is an LSTM-based model, used with different combinations of text, author, and time.

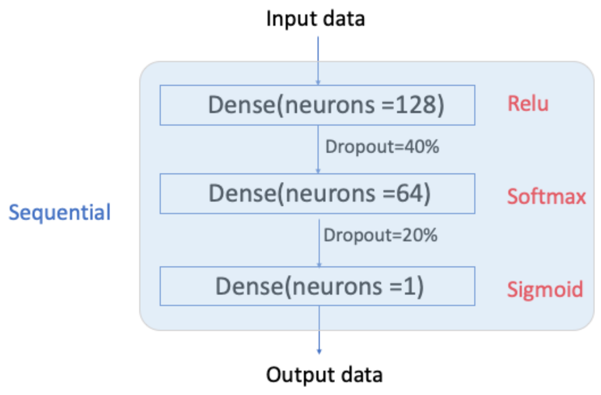
Seq2Seq was chosen as it is a class of neural networks shown to work well with NLP problems, and text-based summarization [25], and fits with future plans for this research to produce not only classifications for the start of a question but also searcheable tags and other indices. For that reason Seq2Seq was desirable to be used despite other methodologies such as RNN being able to produce the same kind of binary classification as explored in this paper.

4.2 Implementation Details

BernoulliNB model is selected from the Naive Bayes, and in the BernoulliNB model, all the parameters are set up to be in default numbers, which means $alpha = 1.0, binarize = 0.0, class_prior = None, fit_prior = True$. In the SVM model, SVC has the paramerters $C = 1.0$, $cache_size = 254$, $class_weight = None$, $coef0 = 0.0$, $decision_function_shape =' ovr'$, $degree = 1$, $gamma = 0.1$, $kernel =' rbf'$,$max_iter = -1, probability = False, random_state = None$, $shrinking = True$,$tol = 0.1$ and $verbose = False$. Note we use scikit-learn[2, 4, 10, 23] to implement the SVC and Naive Bayes.

Keras is used for all deep neural network implementations [5, 6, 12] which is a deep learning library based on Tensorflow [1, 10, 11]. The Stochastic gradient descent using the RMSProp optimizer [5] is applied to update trainable parameters with mini-batch size 10. The sequential model has two hidden layers, and the first hidden state is dense, set with 128 neurons, the first layer activation function is Relu [3, 16], then a 40% dropout followed after the first layer; The second layer is also a dense layer, it has hidden 64 neurons with activation function in Softmax [8, 14], followed by a dropout 0.2. The model compile has binary cross entropy loss[6]. [9] The diagram is shown below in figure 3.

Figure 3: The deep neural network sequential data implementation workflow



4.3 Evaluation metrics

A confusion matrix[13, 28] has been used in the evaluation, with the accuracy formula defined as:

$$Accuracy = \frac{TP + TN}{P + N} \quad (1)$$

Where in this equation, TP represents true positive; TN is true negative; P is the condition positive (the number of real positive cases in the data), N is the condition negative (the number of real negative cases in the data)

[5]http://www.cs.toronto.edu/ tijmen/csc321/slides/lecture$_{slides_lec6.pdf}$
[6]https://keras.io/losses/

Table 1: Results data using the one-day dataset, with best performance marked in bold.

Prediction	30-min	60-min	90-min	Overall
NB	**66.7%**	40.74%	44.8%	50.74%
SVC	38.9%	**62.9%**	55.17%	52.32%
Seq2Seq(Hash text)	61.9%	59.2%	**67.24%**	**62.78%**
Seq2Seq(embedding text)	NA	NA	NA	NA
Seq2Seq(text+people)	NA	NA	60.3%	NA
Seq2Seq(text+time)	NA	NA	NA	NA
Seq2Seq(text+people+time)	NA	NA	NA	NA

Table 2: Results data using the five-day dataset, with best performance marked in bold.

Prediction	Accuracy
NB	72.2%
SVC	68.1%
Seq2Seq(Hash text)	**83.09**%
Seq2Seq(embedding text)	66.2%
Seq2Seq(text+people)	76.9%
Seq2Seq(text+time)	65.4%
Seq2Seq(text+people+time)	NA

4.4 Results and discussion

We have hand-labelled the one-day dataset collected from the programming community chat board, split the dataset into training and tests, and applied several different models, NB, SVC, and Seq2seq to it.

The table 1 shows the accuracy prediction from the different models. The 30-min means the test dataset is the text from 11:30pm till the next day 00:00am, 60-min means the test dataset is from 11:00pm till 00:00am and 90-min means the test dataset is from 10:30pm till 00:00am. NA in the table means not applicable, indicating that the model did not converge.

The results in the Table 1 shows that in the 30-min model, NB has a better accuracy, at 66.7% than SVC and Seq2Seq models; in the 60-min model, SVC can achieve the accuracy up to 62.9%, which is better than all others. Seq2Seq has the highest accuracy up to 67.24% among all models at the 90-min and overall accuracy is about 62.78% in the Seq2Seq model.

Using the larger five-day dataset sees accuracy increase to 83.09% over NB accuracy of 72.2%.

With more data, it is possible that Seq2Seq models are able to achieve higher accuracies, and more likelihood of convergence. The results in Table 2 shows that Seq2Seq(text+people+time) can achieve the highest accuracy at 83.09%, while NB and SVC machine learning methods can only achieve about 70% accuracy.

5 CONCLUSIONS

The ability to classify the start of a question in a real-time chat conversation in a programmer community is the first piece of the puzzle to being able to extract questions and answers, and index them to make them searchable by other learners seeking for similar discussions and topics. An accurate model would be able to run in real-time, automatically collecting and post conversations to a searchable database as they happen, building a larger library of knowledge more rapidly than can be done by traditional community question-and-answer website formats.

We explored the use of Seq2seq, a LSTM-based method from deep learning, showing that even with a relatively straightforward implementation, it is able to achieve better results than traditional NB, and SVC models

Future work include applying the model to larger datasets, and extending the model to also detect the end of a question/topic, and identify sub-classes such as requests for additional information, comments with low relevance to the topic, and interruptions from other users that do not contribute to the topic.

With the growth of popularity of real-time chat as a means for internet users to exchange information, this represents a growing repository of information that remains relatively opaque to those searching for information.

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*. 265–283.

[2] Alexandre Abraham, Fabian Pedregosa, Michael Eickenberg, Philippe Gervais, Andreas Mueller, Jean Kossaifi, Alexandre Gramfort, Bertrand Thirion, and Gaël Varoquaux. 2014. Machine learning for neuroimaging with scikit-learn. *Frontiers in neuroinformatics* 8 (2014), 14.

[3] Forest Agostinelli, Matthew Hoffman, Peter Sadowski, and Pierre Baldi. 2014. Learning activation functions to improve deep neural networks. *arXiv preprint arXiv:1412.6830* (2014).

[4] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, et al. 2013. API design for machine learning software: experiences from the scikit-learn project. *arXiv preprint arXiv:1309.0238* (2013).

[5] Francois Chollet. 2018. *Deep Learning mit Python und Keras: Das Praxis-Handbuch vom Entwickler der Keras-Bibliothek.* MITP-Verlags GmbH & Co. KG.

[6] François Chollet et al. 2015. Keras.

[7] Mihai Dascalu, Traian Rebedea, and Stefan Trausan-Matu. 2010. A Deep Insight in Chat Analysis: Collaboration, Evolution and Evaluation, Summarization and Search. In *Artificial Intelligence: Methodology, Systems, and Applications*, Darina Dicheva and Danail Dochev (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 191–200.

[8] Rob A Dunne and Norm A Campbell. 1997. On the pairing of the softmax activation and cross-entropy penalty functions and the derivation of the softmax activation function. In *Proc. 8th Aust. Conf. on the Neural Networks, Melbourne*, Vol. 181. Citeseer, 185.

[9] Kevin Swersky Geoffrey Hinton, Nitish Srivastava. [n. d.]. Neural Networks for Machine Learning. *http://www.cs.toronto.edu/ tijmen/csc321/slides/lecture*$_s lides_l ec6.pdf([n.d.])$.

[10] Aurélien Géron. 2017. *Hands-on machine learning with Scikit-Learn and TensorFlow: concepts, tools, and techniques to build intelligent systems.* " O'Reilly Media, Inc.".

[11] Sanjay Surendranath Girija. 2016. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *Software available from tensorflow. org* (2016).

[12] Antonio Gulli and Sujit Pal. 2017. *Deep Learning with Keras.* Packt Publishing Ltd.

[13] AM Hay. 1988. The derivation of global estimates from a confusion matrix. *International Journal of Remote Sensing* 9, 8 (1988), 1395–1398.

[14] Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144* (2016).

[15] Sotiris B Kotsiantis, CJ Pierrakeas, and Panayiotis E Pintelas. 2003. Preventing student dropout in distance learning using machine learning techniques. In *International conference on knowledge-based and intelligent information and engineering systems.* Springer, 267–274.

[16] Yuanzhi Li and Yang Yuan. 2017. Convergence analysis of two-layer neural networks with relu activation. In *Advances in Neural Information Processing Systems.* 597–607.

[17] Yuanjie Liu, Shasha Li, Yunbo Cao, Chin-Yew Lin, Dingyi Han, and Yong Yu. 2008. Understanding and Summarizing Answers in Community-based Question Answering Services. In *Proceedings of the 22Nd International Conference on Computational Linguistics - Volume 1 (COLING '08)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 497–504. http://dl.acm.org/citation.cfm?id=1599081.1599144

[18] Edward Loper and Steven Bird. 2002. NLTK: the natural language toolkit. *arXiv preprint cs/0205028* (2002).

[19] Ioanna Lykourentzou, Ioannis Giannoukos, Vassilis Nikolopoulos, George Mpardis, and Vassili Loumos. 2009. Dropout prediction in e-learning courses through the combination of machine learning techniques. *Computers & Education* 53, 3 (2009), 950–965.

[20] Tatsunori Mori, Masanori Nozawa, and Yoshiaki Asada. 2005. Multi-answer-focused Multi-document Summarization Using a Question-answering Engine. 4, 3 (Sept. 2005), 305–320. https://doi.org/10.1145/1111667.1111672

[21] Hajime Okumura, Kazushi Miki, Shunji Misawa, Kunihiro Sakamoto, Tsunenori Sakamoto, and Sadafumi Yoshida. 1989. Observation of Direct Band Gap Properties in GenSim Strained-Layer Superlattices. *Japanese journal of applied physics* 28, 11A (1989), L1893.

[22] Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd annual meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 271.

[23] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in Python. *Journal of machine learning research* 12, Oct (2011), 2825–2830.

[24] Jacob Perkins. 2010. *Python text processing with NLTK 2.0 cookbook.* Packt Publishing Ltd.

[25] Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A Neural Attention Model for Abstractive Sentence Summarization. *arXiv e-prints*, Article arXiv:1509.00685 (Sep 2015), arXiv:1509.00685 pages. arXiv:cs.CL/1509.00685

[26] Arpit Sood, Thanvir P. Mohamed, and Vasudeva Varma. 2013. Topic-Focused Summarization of Chat Conversations. In *Advances in Information Retrieval*, Pavel Serdyukov, Pavel Braslavski, Sergei O. Kuznetsov, Jaap Kamps, Stefan Rüger, Eugene Agichtein, Ilya Segalovich, and Emine Yilmaz (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 800–803.

[27] Douglas R Stinson. 1991. Universal hashing and authentication codes. In *Annual International Cryptology Conference*. Springer, 74–85.

[28] James T Townsend. 1971. Theoretical analysis of an alphabetic confusion matrix. *Perception & Psychophysics* 9, 1 (1971), 40–50.

[29] B. Xu, Z. Xing, X. Xia, and D. Lo. 2017. AnswerBot: Automated generation of answer summary to developers' technical questions. In *2017 32nd IEEE/ACM International Conference on Automated Software Engineering (ASE)*. 706–716. https://doi.org/10.1109/ASE.2017.8115681

[30] Xiaojin Zhu. 2015. Machine teaching: An inverse problem to machine learning and an approach toward optimal education. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*.