This is a post-peer-review, pre-copyedit version of a paper published in Elloumi M, Granitzer M, Hameurlain A, Seifert C, Stein B, Tjoa A & Wagner R (eds.) *Database and Expert Systems Applications. DEXA 2018*. Communications in Computer and Information Science, 903. The final authenticated version is available online at: https://doi.org/10.1007/978-3-319-99133-7_18

A Case Study of Closed-Domain Response Suggestion with Limited Training Data

Lukas Galke[1][0000−0001−6124−1092], Gunnar Gerstenkorn[2][0000−0002−4889−511X], and Ansgar Scherp[3][0000−0002−2653−9245]

[1] University of Kiel, Germany
lga@informatik.uni-kiel.de
[2] University of Potsdam, Germany
gerstenkorn@uni-potsdam.de
[3] University of Stirling, Scotland UK
ansgar.scherp@stir.ac.uk

Abstract. We analyze the problem of response suggestion in a closed domain along a real-world scenario of a digital library. We present a text-processing pipeline to generate question-answer pairs from chat transcripts. On this limited amount of training data, we compare retrieval-based, conditioned-generation, and dedicated representation learning approaches for response suggestion. Our results show that retrieval-based methods that strive to find similar, known contexts are preferable over parametric approaches from the conditioned-generation family, when the training data is limited. We, however, identify a specific representation learning approach that is competitive to the retrieval-based approaches despite the training data limitation.

1 Introduction

Natural language processing digital assistants and tools provide helpful information or automatically suggest responses in a conversation. Most of these assisting tools operate in an open domain without any assumptions. We investigate a similar setting in a closed domain: users of a digital library asking librarians for support regarding the search for literature. Since libraries become more and more digital, these support requests are often made in a digital context, i. e., a chat system. However, many of these requests are repetitive and responding may become tedious. We envision that librarians can benefit from a tool that suggests appropriate responses given the specific request. This way, the repetitive work can be reduced to a minimum of selecting one of few suggested responses. We assume that when operating in a closed domain, finding a well-suited response candidate is easier than in an open domain. However, the restriction to a closed domain also limits the amounts of available training data. That motivates us to re-evaluate different techniques for response suggestion in a setting with limited training data. The amount of training data is a crucial factor for machine learning methods. Thus, we strive to identify those methods that specifically perform well with small amounts of available training data.

We operate on real-world chat transcripts from ZBW — Leibniz Information Centre for Economics[4] obtained through QuestionPoint[5]. QuestionPoint is a platform, that many digital libraries use to process support requests. The transformation from chat transcripts to structured training data involves multiple steps, difficulties, and design decisions that we describe in this paper. Subsequently, we compare retrieval-based, representation learning, and sequence-to-sequence approaches for response suggestion [16, 19]. The core task in response suggestion is to find the most probable response to a given question, which can be formalized as: $r^* = \arg\max_{r \in R} P(r \mid q)$ for a question q and a set of possible responses R [16].

Evaluating different approaches for response suggestion on real-world data is challenging because the responses of training and test data are different. We therefore chose to evaluate the different approaches with the translation metric BLEU [15]. The overlap of word n-grams is used to assess the quality of a suggested response compared to the true response in the data.

In summary, the contributions of this paper are the following:

- A text-processing pipeline that generates question-answer pairs from raw (XML) chat transcripts. We highlight the difficulties and design decisions in constructing such pipelines.
- An empirical evaluation of retrieval-based, conditioned-generation, and representation learning approaches for response suggestion in a closed-domain scenario with small amounts of training data.
- Evidence, that retrieval-based approaches are preferable in scenarios with limited amount of training data. We show that representation learning approaches are less prone to lack of training data than sequence-to-sequence architectures.

After giving a brief overview on the related work in Section 2, we describe the generic text-processing pipeline in Section 3. In Section 4, we describe the employed models for response suggestions, before we depict our experiments in Section 5. The results are presented in Section 6 and discussed in Section 7, before we conclude in Section 8.

2 Related Work

In the following, we briefly review the related work on conditioned-generation and representation learning approaches that are relevant for response suggestion scenarios.

Ritter et al. [16] proposed to use statistical machine translation approaches for conditioned response generation. The source sequence is encoded into a latent representation, which is in turn used to generate the target sequence. The models were trained on Twitter utterances and responses. Vinyals et al. [21] extended this

[4] `http://zbw.eu`
[5] `https://www.oclc.org/en/questionpoint.html`

sequence-to-sequence based approach by taking the chat history into account. Al-Rfou et al. [2] explored a prediction-based approach that exploits input, context, and author information on a Reddit dataset of 2.1 billion utterances. Wu et al. [23] investigated the use of dynamic per-question vocabularies. Xu et al. [24] jointly trained an adversarial discriminator to penalize non-informative answers.

Dedicated representation learning approaches for response suggestion aim to learn (joint or separate) representations for the questions and the responses. The training objective comprises learning a representation that leads to a high similarity score for the correct response and lower scores for other responses, given a certain question. The similarity is typically computed by dot-product or cosine similarity of latent representations such as in *StarSpace* [22] and *DSSM* [9]. Both *StarSpace* and *DSSM* are a general representation learning approaches that use cosine similarity as scoring function. All variants have in common that the training is performed by negative sampling in favor of (hierarchical) softmax [14]. Google Research has conducted two studies considering a Smart Reply mechanism for Gmail [8,10]. The first work relied on a sequence-to-sequence (i. e., conditioned-generation) approach [10], while the second work uses an approach that is purely based on representation learning [8].

From the related work, we observe that conditioned-generation and representation learning for response suggestion are well-studied topics. However, most approaches consider open-domain applications, e. g., Twitter, Reddit, or Gmail, where large amounts of training data is available. In this paper, we are instead targeting closed-domain scenarios such as a support system of a library, in which considerably less training data is available.

3 Text-Processing Pipeline

In this section, we describe our text-processing pipeline starting at raw XML query log data. After extracting the transcripts (Section 3.1), we join consecutive utterances (Section 3.2), and then generate question-answer pairs (Section 3.3). Finally, we conduct a language detection step (Section 3.4), before the pairs are supplied to the response suggestion models.

3.1 Transcript Extraction

We operate on XML exports from QuestionPoint[6]. QuestionPoint is a dedicated platform, on which librarians answer support requests of the library's patrons (the user). The XML transcripts consist of an ordered sequence of utterances. Each utterance is associated with an agent. The agent is either a librarian or the user. The transcript opens with the user's initial question. It also comprises status messages, such as "Patron's screen name: X", "Library ended chat session.", "Set Resolution: Y". These status messages are filtered via regular expressions. We further clean all utterances from HTML tags such as `<br>` and `<p>`. Our data

[6] `https://www.oclc.org/en/questionpoint.html`

consists of 3,246 XML transcripts of chat sessions regarding the literature search service EconBiz of ZBW.

3.2 Joining Consecutive Utterances

It is possible that two consecutive utterances were made by the same responsible agent. From inspecting the data, we learn that this is often the case when a domain expert first replies with a rather short answer such as "Hello, let me first read the question" or "I will look it up in our catalogue". Afterwards, the domain expert provides the actual answer. We assume that it is beneficial, when the actual answer is pursued as soon as possible. Otherwise, all considered approaches would tend to yield rather generic yet uninformative suggestions. We therefore decide to consistently concatenate consecutive utterances from the same agent.

3.3 Question-Answer Pair Generation

At the current point, we have a set of transcripts of alternating (library and patron) utterances. The goal is to create a set of paired training data from these utterances. We considered multiple approaches for generating paired training data from the sequences of alternating utterances. The generation of training data is not trivial because each librarian's response may not only refer back to the preceding utterance of the user, but also the context as a whole or the user's initial question.

We continue with generating exactly one question-answer pair for each librarian answer, in which the question is the previous utterance of the user. This approach, in some cases, suffers from a loss of context due to singular "Ok."-fashioned utterances. For instance, the librarian initially response with a greeting message and asks for time, then the user says "Ok.", which is then used as context for the next answer. Still, this approach lead to most reasonable response suggestion models in our pre-experiments. Using this approach, we generated 14,059 question-answer pairs from the 3,246 chat transcripts.

3.4 Language Detection

In the final data preparation step, we apply language detection. The majority of chat transcripts consisted of German utterances. Still a considerable amount of transcripts also held English utterances. To detect the language of a question-answer pair, we employed Naive Bayes classifier operating on character n-grams. We used the implementation of PyCLD2[7], which in turn utilizes the Compact Language Detector 2[8] by Google. Among the 14,059 generated question-answer pairs, 11,511 were recognized as German and 2,795 were recognized as English language. The language detection is, however, not necessarily exclusive: 246 pairs were classified as both German and English.

[7] `https://github.com/aboSamoor/pycld2`
[8] `https://github.com/CLD2Owners/cld2`

4 Response Suggestion

In the following, we introduce the employed approaches for response suggestion from three different families: information retrieval in Section 4.1, condition-generation in Section 4.2, and pure representation learning in Section 4.3.

4.1 Retrieval Models

We consider employing retrieval models for response suggestion. After indexing the paired training data, we use the patrons' questions as query to the known questions. We employ two retrieval models: TF-IDF [17], a well-known baseline from Information Retrieval [13], and word centroid distance (WCD) [6, 12] which uses word embeddings [14] to compute a similarity score. As investigated in our prior work [6], we use inverse document frequency (as in TF-IDF) to re-weight term counts for computing the word centroid distance.

Using these building blocks, we evaluate multiple combinations as retrieval models: TF, TF-IDF, WCD, and WCD-IDF. In addition, we evaluate the aforementioned retrieval models with a disjunctive term matching operation (abbreviated with the prefix M) that reduces the candidates to those that have at least one term in common with the query, which is in our case the question. For all retrieval models, we use cosine similarity to retrieve the most similar context to the given question and return the known answer to this context.

To compute the word vector centroids, we use German *fastText* [4] word vectors trained on CommonCrawl and Wikipedia[9] [7]. For the experiment on English utterances, we use *Word2Vec* word vectors trained on Google News [14].

We further experiment with employing k-nearest neighbors (k-NN) with $k \in \{1, 3, 5, 7\}$ to predict the response to a question. The k nearest question-response pairs are retrieved via cosine similarity. Each neighbor is associated with a specific response. The neighbors then cast a similarity-based vote on the response to return. It is possible that two or more neighbors point to the same (tokenized) response. In this case, the predicted response may differ from the ones of the retrieval-based approach, which is comparable to k-NN using a single nearest neighbor.

4.2 Conditioned-Generation

Generating an answer based on a question can be regarded as a sequence transduction task [16]. Common sequence-to-sequence models [19] use one recurrent neural network to encode the question into a hidden state. Then, another recurrent neural network decodes the hidden state into the target (known) answer. Sequence-to-sequence models can also be applied as conditioned response generation [16]. For all following experiments, we use the Tensorflow [1] sequence-to-sequence

[9] `https://github.com/facebookresearch/fastText/blob/master/docs/crawl-vectors.md`

implementation [10] with its default hyperparameters. Those were two hidden layers of 128 hidden units with a 0.2 dropout and a learning rate of 1. The vocabulary for the sequence-to-sequence model was generated on the training data only.

4.3 Representation Learning

We report the results of two approaches for response suggestion from the related work [8]. The authors label the two approaches as *joint* and *dotproduct*, respectively. Both of them rely on producing a score, given word n-grams of the question and the reponse. The score determines how appropriate the specific response is to the question. In the *joint* approach, the bag-of-ngram representations of the question and the response are concatenated and then fed to three hidden layers with ReLU activations and a final layer outputs the score. In the *dotproduct* approach, the questions and responses are separately encoded into vector representations before using cosine similarity for scoring. In this case, we use three hidden layers with Tanh activations as in the original work [8]. The computed score is optimized to be high for matching question-answer pairs and low for negative samples. We chose to use soft-margin loss as objective to rank the correct response higher than the responses of negative samples.

We use unigrams and bigrams as initial representation. We chose a hidden layer size of 100 and apply dropout [18] with a drop probability of 0.2 on each intermediate hidden layer. We further experimented with employing pre-trained word vectors for the initial conversion between words and vectors. This, however, did not lead to an improvement. All variants are trained for 50 epochs with Adam [11] optimizer using an initial learning rate of 0.001. We experimented with using between one and five negative samples.

5 Experiments

In the following, we describe our experiments conducted on both the German and the English subset of generated question-answer pairs. After briefly introducing the general experimental procedure in Section 5.1, we provide details on word count statistics in Section 5.2, before we describe our evaluation metrics in Section 5.3.

5.1 Experimental Procedure

For conducting our experiments, we split the data into 90% training pairs and 10% test pairs. The sequence-to-sequence approach internally separates another 10% from the training data for validation. After the training step, we supply each question of the test set to the respective method and compare its top suggested response against the real response.

[10] `https://www.tensorflow.org/versions/r1.1/tutorials/seq2seq`

Table 1. Word count statistics of questions and answers

Data	Min	Q25	Q50	Q75	Max	Mean	SD
English sources	1	4	9	18	386	14.19	19.20
English targets	1	10	19	34	697	31.34	50.84
German sources	1	4	9	18	386	13.99	16.77
German targets	1	9	17	30	790	24.53	34.14

5.2 Dataset Characteristics

The basic statistics of questions' and answers' word count are provided in Table 1 for both the German and the English subset. The median of the word counts is between 4 and 10. Most of utterances are rather short, the median is lower than the mean length of utterances (between 16 and 50). The longest question is 386 words long while the longest answer consists of 697 words in English and 790 words in German. We manually removed one pair, in which a whole email-conversation of more than 2,000 words has been copied into the chat.

5.3 Evaluation Metrics

Since the responses of our generated datasets are not organized along classes, we use evaluation metrics from machine translation. To evaluate the suggested responses, we employ the BLEU (bilingual evaluation understudy) score, which was proposed by Papineni et al. [15]. The BLEU score computes the precision of word n-grams of the hypothesis against one or more references. In our case, we use a single reference, which is the correct response given by the data, for each question-response pair. We reuse the Corpus-BLEU implementation provided by NLTK[11]. The Corpus-BLEU score aggregates word n-gram counts over the whole corpus before performing a single division. A brevity penalty for the length difference between question and reference answer is imposed. We abbreviate the Corpus-BLEU score in our results by BLEU. We use smoothing function 5 from the work of Chen and Cherry [5], which averages the n-gram precision scores over $n-1, n, n+1$ before computing the BLEU score. This method attained the highest correlation with human judgements without introducing a new parameter for the metric. We also evaluate the sole word choice and fluency part of BLEU: modified precision [15] (p_n) on word uni-, bi-, and trigrams. The modified precision score refers to the fraction of words in the suggested response that also appear in the actual response, bounded by the number of times a word appears in the actual response. This allows a more detailed inspection of the models, whereas a single corpus wide BLEU score would paint a superficial picture.

Table 2. BLEU modified precision scores using uni- (p_1), bi- (p_2), and trigrams (p_3) along with Corpus BLEU score of the response suggestion methods on the **German** question-answer pairs.

Model	p_1	p_2	p_3	BLEU
Traditional retrieval				
TF	28.53	18.65	16.70	23.76
TF-IDF	28.73	19.39	17.54	23.57
M-TF	28.05	18.29	16.45	23.56
M-TF-IDF	28.48	19.03	16.88	23.19
Retrieval with KNN				
1-NN	29.27	19.68	17.60	23.49
3-NN	**29.47**	**20.15**	**18.12**	23.35
5-NN	28.47	19.04	17.02	23.08
7-NN	28.04	18.71	16.70	23.08
Retrieval with word vectors				
WCD	26.87	17.65	16.22	23.70
WCD-IDF	27.55	18.25	16.35	24.17
M-WCD	27.45	17.94	16.31	23.51
M-WCD-IDF	28.16	18.67	16.92	**24.37**
Representation learning				
Dotproduct-n1	11.43	1.18	0.38	8.04
Dotproduct-n3	7.36	0.69	0.20	7.61
Joint-n3	26.06	16.95	15.26	21.19
Joint-n4	27.21	17.25	15.71	17.99
Joint-n5	24.23	16.13	14.77	18.11
Conditioned-generation				
seq2seq	11.51	2.75	0.99	7.42

Table 3. BLEU modified precision scores using uni- (p_1), bi- (p_2), and trigrams (p_3) along with Corpus BLEU score of the response suggestion methods on the **English** question-answer pairs.

Model	p_1	p_2	p_3	BLEU
Traditional retrieval				
TF	27.32	17.33	15.34	19.50
TF-IDF	27.89	17.59	15.85	21.16
M-TF	26.47	17.20	15.19	18.84
M-TF-IDF	27.89	**17.96**	**16.26**	21.67
Retrieval with KNN				
1-NN	27.40	17.26	15.50	23.09
3-NN	26.64	16.93	15.59	23.71
5-NN	26.92	17.29	15.93	**24.00**
7-NN	27.02	17.25	15.87	23.79
Retrieval with word vectors				
WCD	28.40	17.70	15.59	20.72
WCD-IDF	28.09	17.35	15.25	20.57
M-WCD	26.76	17.24	15.57	18.55
M-WCD-IDF	**28.96**	17.77	15.49	19.44
Representation learning				
Dotproduct-n1	13.61	1.55	0.72	8.59
Dotproduct-n3	4.52	0.51	0.04	7.13
Joint-n3	25.34	16.23	14.80	14.07
Joint-n4	25.72	16.73	15.22	16.81
Joint-n5	26.53	16.69	14.83	18.73
Conditioned-generation				
seq2seq	16.93	7.58	4.8	4.79

6 Results

Table 2 shows the results for the experiment on the German question-answer pairs. We report the mean modified precision values p_1, p_2, p_3, and BLEU scores for each of the models described in Section 4. KNN with $k = 3$ is the top-performing approach considering p_1, p_2, and p_3. Considering BLEU score, the M-WCD-IDF retrieval model yields the highest score of 24.37. The *joint* representation learning approach is the closest competitor to the retrieval-based methods and attains BLEU scores of up to 21.19.

Table 3 shows the BLEU and modified-precision scores of the methods on the English question-answer pairs. The approaches M-WCD-IDF and M-TF-IDF yield the highest p_1, p_2, p_3 scores. The KNN-based approaches consistently yield the highest BLEU scores. With 5 nearest neighbors, the BLEU score peaks at 24.00. The representation learning approach *joint* is the strongest competitor for the retrieval-based methods with a modified precision values of 26.53, 16.69, and 14.83, respectively, along with a BLEU score of 18.73.

7 Discussion

The main result is that retrieval models are superior to conditioned-generation approaches in our closed domain case study with limited training data. We hypothesize that this is primarily caused by the lack of training data for the parameter-learning approaches. The *joint* approach becomes competitive to the retrieval methods when tuning the number of negative samples. The *dotproduct* approach did not lead to reasonable results, also not with more negative samples.

A limitation of this work is that we, for now, focussed on comparing the retrieval-based approaches and the representation learning approaches to the basic RNN sequence-to-sequence architecture. Sequence-to-sequence is, however, a general framework, in which many extensions [3, 20, 23, 24] are possible. While our results are a strong indicator that the sequence-to-sequence models lack training data, we cannot exclude that a different configuration of a sequence-to-sequence architecture might yield a higher performance.

Both retrieval models and representation learning approaches yield responses that are also present in the training set. In contrast, the conditioned-generation model generates new sentences and is susceptible to be ungrammatical or semantically questionable. This might be insufficiently addressed by the BLEU score. For this reason, we manually inspected the generated responses. Despite the limited amount of training data, the learned grammar was in most cases appropriate. The generated responses often contained phrases of two to five words that were reminiscent of the training set. As expected, we observe a tendency that the generated responses are rather generic and reflect the most-common responses of the training set. While suggesting such generic responses is in principle not undesirable, the BLEU scores show that the generated response does, in most cases, not reflect what the librarian's actual response was.

[11] `http://www.nltk.org/_modules/nltk/translate/bleu_score.html`

Other works use response normalization [8, 16] for evaluation. The responses of the test set are normalized with respect to the responses of the training set. This would also alleviate the aforementioned issue of *UNK* tokens in conditioned-generation approaches. In our case, however, response normalization would have biased our evaluation. Due to the limited data, more semantically different answers would have been mapped to the same response class. Thus, we preferred the BLEU score for our evaluation.

We considered alternative transformations of chat transcripts into paired training data. One alternative was to solely extract the initial question and answer. This, however, leads to rather uninformative response suggestions such as greeting the user and notifying him/her that his/her question is recognized. Still, in some cases, the initial response did already hold substantial information, such that omitting the first response was also not an option. We also considered aggregating all past utterances within a transcript into a joint context. Whenever an utterance of a librarian is traversed, a pair of context and response is emitted. Both the user's utterances and past librarian utterances are kept in the context until the end of the transcript. The loss of immediate context, however, harmed the performance of the response suggestion models. As future work, we consider extracting contextualized triples holding the immediate context, the long-term contexts, and the response.

We evaluated the different approaches over two languages of a single real-world dataset. Further studies are necessary to assert whether our results generalize to other datasets. For this dataset, we started from raw data to a training set of question-answer pairs. As future work, we consider taking also the context (the chat history) of each pair into account. We envision that more contextualized predictions may help to mitigate the lack of training data.

8 Conclusion

We have shown that retrieval-based methods yield higher BLEU scores on a response suggestion task than sequence-to-sequence models and representation learning approaches, when the training data is limited. Dedicated representation learning techniques, however, are in some cases competitive to the strong retrieval baselines. More specifically, an architecture trained to learn a joint representation of question and answers using three hidden layers with rectified linear activations did yield competitive scores. Given our results, this approach can be considered the most promising parametric approach for response suggestion in scenarios with limited training data. We presented a text-processing pipeline to extract question-answer pairs from XML data. Such question-answer pairs, regardless of where they come from, can be used to construct a response suggestion model. We make our text-processing pipeline, the response suggestion models, and the evaluation procedure openly available on GitHub[12]. The goal is that other researchers and practitioners may re-use our implementation to verify our results on other datasets or construct question answering systems for a real world applications.

[12] `https://github.com/lgalke/resuggest`

Acknowledgements

This research was co-financed by the EU H2020 project MOVING[13] under contract no 693092. We thank Nicole Krueger from ZBW for providing the chat transcripts and helpful discussions on requirements and possible applications.

References

1. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., et al.: Tensorflow: Large-scale machine learning on heterogeneous distributed systems. CoRR **abs/1603.04467** (2016)
2. Al-Rfou, R., Pickett, M., Snaider, J., Sung, Y., Strope, B., Kurzweil, R.: Conversational contextual cues: The case of personalization and history for response ranking. CoRR **abs/1606.00372** (2016)
3. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. CoRR **abs/1409.0473** (2014)
4. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. TACL **5**, 135–146 (2017)
5. Chen, B., Cherry, C.: A systematic comparison of smoothing techniques for sentence-level BLEU. In: WMT@ACL. The Association for Computer Linguistics (2014)
6. Galke, L., Saleh, A., Scherp, A.: Word embeddings for practical information retrieval. In: GI-Jahrestagung. LNI, vol. P-275. GI (2017)
7. Grave, E., Bojanowski, P., Gupta, P., Joulin, A., Mikolov, T.: Learning word vectors for 157 languages. In: Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018) (2018)
8. Henderson, M., Al-Rfou, R., Strope, B., Sung, Y., Lukács, L., Guo, R., Kumar, S., Miklos, B., Kurzweil, R.: Efficient natural language response suggestion for smart reply. CoRR **abs/1705.00652** (2017)
9. Huang, P., He, X., Gao, J., Deng, L., Acero, A., Heck, L.P.: Learning deep structured semantic models for web search using clickthrough data. In: CIKM. ACM (2013)
10. Kannan, A., Kurach, K., Ravi, S., Kaufmann, T., Tomkins, A., Miklos, B., Corrado, G., Lukács, L., Ganea, M., Young, P., Ramavajjala, V.: Smart reply: Automated response suggestion for email. In: KDD. ACM (2016)
11. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. CoRR **abs/1412.6980** (2014)
12. Kusner, M.J., Sun, Y., Kolkin, N.I., Weinberger, K.Q.: From word embeddings to document distances. In: ICML. JMLR Workshop and Conference Proceedings, vol. 37. JMLR.org (2015)
13. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to information retrieval. Cambridge University Press (2008)
14. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: NIPS (2013)
15. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: Bleu: a method for automatic evaluation of machine translation. ACL-200: 40th Annual meeting of the Association for Computational Linguistics (2002)
16. Ritter, A., Cherry, C., Dolan, W.B.: Data-driven response generation in social media. In: EMNLP. ACL (2011)

[13] `http://www.moving-project.eu/`

17. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Inf. Process. Manage. **24**(5) (1988)
18. Srivastava, N., Hinton, G.E., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. Journal of Machine Learning Research **15**(1) (2014)
19. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: NIPS (2014)
20. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: NIPS (2017)
21. Vinyals, O., Le, Q.V.: A neural conversational model. CoRR **abs/1506.05869** (2015)
22. Wu, L., Fisch, A., Chopra, S., Adams, K., Bordes, A., Weston, J.: Starspace: Embed all the things! CoRR **abs/1709.03856** (2017)
23. Wu, Y., Wu, W., Yang, D., Xu, C., Li, Z., Zhou, M.: Neural response generation with dynamic vocabularies. CoRR **abs/1711.11191** (2017)
24. Xu, Z., Liu, B., Wang, B., Sun, C., Wang, X., Wang, Z., Qi, C.: Neural response generation via GAN with an approximate embedding layer. In: EMNLP. Association for Computational Linguistics (2017)