Exploring Approaches for a Question Answering System

Laurence Dupont
Nu Echo
June 2022

A fundamental functionality of conversational agents is to be able to answer frequent questions formulated in natural language. Indeed, a powerful question answering system improves the user experience and reduces the number of calls made to the contact center.

In a previous blog post, we described the experiments that were conducted to implement such a system with Dialogflow ES [1]. Since then, Nu Echo has explored new approaches to implement a question answering system that could eventually be integrated into a conversational agent in production. This article aims to summarize the steps that led to the choice of the final solution, namely a system using Google's Universal Sentence Encoder (USE) model combined with a neural network.

1 Problem Definition

There are a myriad of ways to approach the problem of developing a machine learning based question answering system. One of them is the answer selection task. It consists, for a given question, in returning the best answer among a set of candidate answers which includes one or more correct answers [2]. If the returned answer is one of the correct answers to the question, then the prediction is correct, otherwise it is incorrect [2].

Knowing that an answer is associated with a question-answer pair, it seems reasonable to extend this task definition and consider that a candidate can be a question-answer pair, its question or its answer. When a candidate in one of these forms is selected, it is trivial to return the corresponding answer.

The answer selection task is based on the assumption that there is always a correct answer for each question. However, this is not always the case in a real question answering system [3]. Indeed, we want the system not to provide a response if a user:

- asks an in-domain question which is not supported by the question answering system;
- asks an out-of-domain question;
- enters text that is not a question.

The answer triggering task offers this possibility. The different approaches to achieve it have therefore been explored.

2 Potential Solutions

To complete the answer triggering task, the approaches described in Table 1 are possible [4]. These approaches aim to achieve two subtasks: determining whether there is a correct answer and selecting an answer [4].

Approach	Description
Joint	A model jointly optimizes the subtasks of correct answer detection and answer selection.
Sequential with upstream rejection	A first model determines if there is a correct answer among the candidate answers. If so, a second model determines the best answer and returns it.
Sequential with downstream rejection	For a given question, a model gives a score to each candidate answer. If the highest score is greater than or equal to a predetermined threshold, the associated candidate answer is returned.

Table 1: Answer triggering approaches.

The sequential approach with downstream rejection was considered preferable, in particular because the threshold can be changed easily without retraining the model.

To accomplish the answer selection subtask, the approaches listed in Table 2 were considered.

Approach	Description
Learning to rank	A model learns to rank the candidate answers to a question so that the best answers are at the top of the ranking [5].
Information retrieval using semantic similarity	A neural model encodes the vectorial semantic representation of frequently asked questions into documents [6]. These documents are then stored in a database optimized for the nearest neighbors search [6]. When a user asks a question, it is encoded with the neural model and a nearest neighbors search is performed to find the frequently asked question that has the most semantic similarity [6]. The associated answer can then be returned by the system.
Intent or text classification	Intent classification is a text classification problem where classes correspond to different intents. In the context of a question answering system, intents correspond to frequently asked questions. When a user asks a question, it is vectorized using a vectorization model, and then the classifier predicts an intent. The associated answer is returned using a table of correspondence between intents and answers.

Table 2: Answer selection approaches.

Among these approaches, only the last two have been retained. Learning to rank was considered less interesting. Indeed, to train a learning to rank model, it is necessary to have a dataset in which each example is associated with a score/degree of relevance [7], which is not easy to produce. Also, since our conversational agents only display the best answer, there is no real need to optimize the order of the other candidate answers.

3 System Definition

To accomplish the answer triggering task using intent or text classification, the system depicted in Figure 1 can be used.

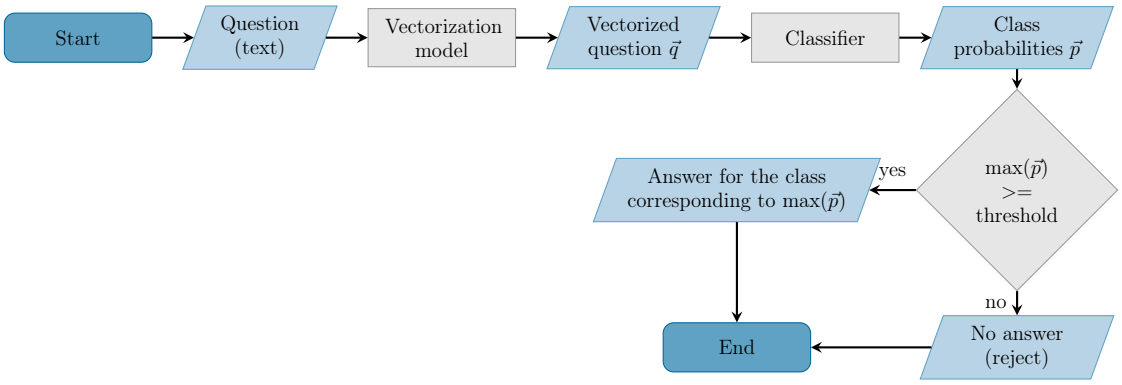


Figure 1: Question answering system pipeline.

In this system, the text of a user's question is first vectorized by a vectorization model. Then, this vector is provided as input to a classifier whose classes correspond to frequently asked questions. This classifier then outputs a probability for each class, and the highest probability is then compared to a threshold. If this probability is greater than or equal to the threshold, the system returns the answer of the class corresponding to this probability. Otherwise, it does not provide an answer.

This system can also be used for the answer selection approach based on semantic similarity. Rather than using a database optimized for nearest neighbors search, one can use a k-nearest neighbors classifier.

4 Dataset

A suitable dataset for the development of this question answering system was then selected. The banking dataset BANKING77 [8], created by the conversational solutions company PolyAI, was chosen. Although it is intended for the task of intent classification rather than answer selection, it contains enough examples formulated as questions and enough intents (77) to be representative of the size of an FAQ.

An interesting feature of this dataset is that it has three variants: *10*, *30* and *full*. Variants *10* and *30* have a training set that contains only 10 and 30 examples per intent, which is a subset of the examples in the *full* variant. All variants have the same test set, which contains 40 examples per intent.

5 Methodology

The question answering system presented earlier contained two models: one for vectorization and one for classification. Experiments were therefore carried out in order to determine the best combination of models.

For each variant of the dataset and each combination of vectorization model and classifier, the following procedure was performed:

- The training data was first vectorized with a vectorization model.
- A classifier was trained with a hyperparameter search procedure with cross-validation.
- The vectorized test data was provided to the classifier so that it could generate predictions from it.
- Finally, these predictions were evaluated with the accuracy metric, which calculates the percentage of correct predictions.

6 Vectorization Models

The traditional and neural vectorization models contained in Table 3 were evaluated.

Model	Description
bag-of-words	Represents a text by the number of occurrences of the words contained in it without taking into account their position [9].
TF-IDF	Like bag-of-words, but also considers the number of occurrences of words in all texts to give less weight to words found in many texts (e.g., determiners) [9].
BERT (average embeddings)	A pretrained BERT model that accepts a text as input and outputs a vector representation that is averaged to obtain a sentence embedding [10].
Sentence-BERT	A pretrained BERT model fine-tuned on several semantic tasks that accepts a text as input and outputs a sentence embedding [10].
Universal Sentence Encoder (USE)	A neural network pretrained simultaneously on several semantic tasks that accepts a text as input and outputs a sentence embedding [11]. For the base variant, the neural network is a Deep Averaging Network [12]. For the large variant, it is the encoder of a Transformer [13].

Table 3: Vectorization models.

The traditional bag-of-words and TF-IDF models were used as baselines for the experiments. They are based only on the training set of the dataset used for a given experiment. As for the neural models, they are pre-trained on external datasets.

For each of these vectorization models, answer selection experiments were performed with different classifiers.

7 Experiments with a KNN Classifier

The semantic similarity approach described earlier encodes a vector representation of frequently asked questions in a database optimized for nearest neighbors search. To evaluate the potential of this approach, experiments were conducted with a k-nearest neighbors classifier, also called KNN (KNeighborsClassifier [9]).

7.1 Cosine distance

First, experiments were performed with a KNN with cosine distance. For each of the variants in the dataset, the USE large vectorization model performed best. It was also observed that the choice of the vectorization model had a considerable impact on the performance. However, even for the best vectorization model, the accuracy still left room for improvement. Indeed, for variant *10* of the dataset, the accuracy was less than 80% (78.44%).

7.2 Learned distance

Then, other experiments were carried out with a learned distance function in order to improve the performance. Metric learning aims to learn a task-specific distance function and is beneficial when combined with a nearest neighbors model.

The Local Fisher Discriminant Analysis (LFDA [14] [15]) supervised learning algorithm was used. It aims to bring together the examples of the same class and move away the examples belonging to different classes. With this distance function, the USE large model obtained the best performance for the *30* and *full* variants. However, this represents a very slight relative improvement over the performance obtained with the cosine distance function.

In a PolyAI blog post that was published after the experiments described here were performed, better results were reported with a metric learning method based on a neural network [16]. The main advantage of the neural network over the LFDA algorithm is that it can learn a non-linear transformation function. However, PolyAI's experiments on different datasets showed that a neural

network performed slightly better than a KNN with deep metric learning [16]. It is therefore not clear that the benefits of KNN with learned distance are large enough to justify the additional implementation complexity.

8 Experiments with a Linear SVM Classifier and a Multilayer Perceptron Classifier (MLP)

Since the SVM classifier has outperformed a KNN with learned distance in another text classification task [17], experiments were carried out with a linear SVM (LinearSVC [9]). For all variants of the dataset and all vectorization models, the linear SVM actually performed better than the KNN (with cosine distance or learned distance). As for the vectorization models, it was once again USE large that performed the best.

Despite the very good performance of the SVM model on the answer selection task, it is not the most appropriate model for the answer triggering task. As the SVM does not directly return probabilities, a calibration module must be used. However, these probabilities are not very well calibrated [9]. For the question answering system to be able to properly reject out-of-domain questions, it was therefore preferable to use a machine learning model that directly returns probabilities, such as a neural network.

Other experiments were thus performed with a multilayer perceptron (MLPClassifier [9]). This resulted in a model with slightly better performance than the SVM, but which is better suited to the answer triggering task, as it returns probabilities.

9 Evaluation on the Answer Selection Task

The answer selection experiments described previously established that the best model was the one combining the Universal Sentence Encoder (USE) large with an MLP. To evaluate this model, a comparison was performed with the NLU intent classification models of Dialogflow ES and Rasa 1.10.12.

The Rasa NLU pipeline that was used is listed below. It is based on the recommended configuration for English.

```
language: "en"
pipeline:
- name: ConveRTTokenizer
- name: ConveRTFeaturizer
- name: CountVectorsFeaturizer
- name: CountVectorsFeaturizer
  analyzer: "char_wb"
  min_ngram: 1
  max_ngram: 4
- name: DIETClassifier
  entity_recognition: False
  epochs: 100
  random_seed: 42
```

The training and test sets of each variant of the BANKING77 dataset were first transformed into the format expected by Dialogflow and Rasa. Each NLU model was then trained on the training set and evaluated on the test set with the accuracy metric. The results obtained are listed in Table 4.

Model	Variant		
	10	**30**	**full**
USE (large) + MLP	86.33	90.03	92.69
Dialogflow ES	74.77	83.86	90.26
Rasa	69.03	86.33	91.17

Table 4: Evaluation results on the BANKING77 dataset.

It can be observed that for all variants of the dataset, the USE model combined with an MLP performed better than Dialogflow ES and Rasa. However, it should be noted that the performance of the Rasa model could surely be improved by optimizing the hyperparameters.

10 Evaluation on the Answer Triggering Task

The previous evaluation on the answer selection task aimed to verify the ability of a model to provide the correct answer to a question supported by the system. For the answer triggering task, an additional objective is to assess the ability of a model to reject an out-of-domain question.

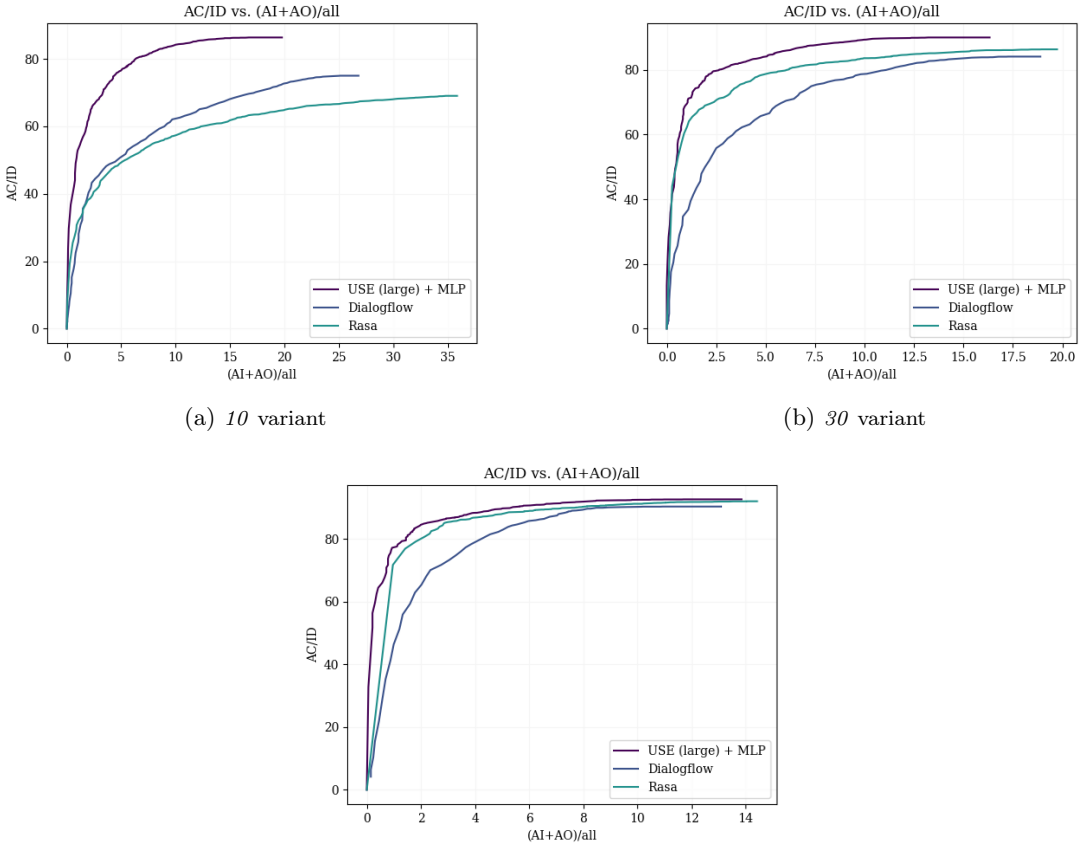
To do this, a new "out_of_scope" intent containing 234 out-of-scope examples has been added to the test set of the three variants of the BANKING77 dataset. More specifically, these examples are questions related to COVID-19 drawn from this dataset [18].

To compare the performance of the models on the test set, a Receiver Operating Characteristic (ROC) curve was used. This graph of one metric versus another (here *AC/ID* versus *(AI+AO)/all*) is generated by varying a threshold. The definitions contained in Table 5 were used.

	Definition
AC	Number of correct predictions whose confidence score is greater than or equal to the threshold
AI	Number of incorrect predictions whose confidence score is greater than or equal to the threshold
AO	Number of out-of-domain predictions whose confidence score is greater than or equal to the threshold
ID	Number of in-domain examples
all	Total number of examples

Table 5: Definitions used for the computation of metrics.

The ROC curves obtained are shown in Figure 2. A model whose curve is above another is better. A perfect result would be to reach the top left corner of the graph, but this does not happen in practice.



(a) *10* variant

(b) *30* variant

(c) *full* variant

Figure 2: ROC curves for the modified BANKING77 dataset with out-of-domain data.

It can be seen that, as in the previous evaluation, the USE model combined with an MLP obtained the best performance, especially for variant *10*. Additionally, it can be observed that if the models had a similar performance on the answer selection task for the *full* variant, it is not the case for the answering triggering task. This demonstrates the importance of this task to compare different models.

For all models, the task of rejecting out-of-domain examples proved difficult, and confidence scores were sometimes higher than expected. For example, Rasa's model returned the intent "age_limit" with a confidence score of over 95% for the following out-of-domain examples:

- "Will educational childcare centres provide meals and snacks?"
- "Should an employee who is 70 years of age continue to work?"
- "Can children accompany their parents in grocery stores, drugstores and other public spaces?"

This can be explained by the fact that the "age_limit" intent was the only one to have examples related to age and children. A potential solution to this kind of problem would be to add similar examples to an "out_of_scope" intent in the training set [19].

11 Conclusion

In conclusion, these experiments made it possible to deliver a proof of concept of a powerful question answering system based on a machine learning model. A good balance between performance and

efficiency could be achieved by using the USE large pretrained vectorization model combined with an MLP.

Acknowledgments

This article is based on the work carried out as part of my master's internship, carried out in 2020. I would like to thank Yves Normandin (Nu Echo) and Sherjil Ozair (Mila) for supervising this internship. I would also like to thank Guillaume Voisine (Nu Echo) for editing this text.

References

[1] Y. Normandin. "Question answering experiments with the dialogflow faq knowledge connectors." (2020), [Online]. Available: https://www.nuecho.com/news-events/question-answering-experiments-with-the-dialogflow-faq-knowledge-connectors/.

[2] M. Feng, B. Xiang, M. R. Glass, L. Wang, and B. Zhou, "Applying deep learning to answer selection: A study and an open task," in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2015, pp. 813–820. DOI: 10.1109/ASRU.2015.7404872.

[3] Y. Yang, W.-t. Yih, and C. Meek, "Wikiqa: A challenge dataset for open-domain question answering," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, 2015, pp. 2013–2018. DOI: 10.18653/v1/D15-1237.

[4] J. Zhao, Y. Su, Z. Guan, and H. Sun, "An end-to-end deep framework for answer triggering with a novel group-level objective," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, 2017, pp. 1276–1282. DOI: 10.18653/v1/D17-1131.

[5] T. M. Lai, T. Bui, and S. Li, "A review on deep learning techniques applied to answer selection," in *Proceedings of the 27th International Conference on Computational Linguistics*, Association for Computational Linguistics, 2018, pp. 2132–2144.

[6] Y. Yang and A. Ahmad. "Multilingual universal sentence encoder for semantic retrieval." (2019), [Online]. Available: https://ai.googleblog.com/2019/07/multilingual-universal-sentence-encoder.html.

[7] H. Li, *Learning to Rank for Information Retrieval and Natural Language Processing*. Morgan & Claypool Publishers, 2011, pp. vi, 4. DOI: 10.2200/S00348ED1V01Y201104HLT012.

[8] I. Casanueva, T. Temcinas, D. Gerz, M. Henderson, and I. Vulic, "Efficient intent detection with dual sentence encoders," in *Proceedings of the 2nd Workshop on NLP for Conversational AI*, Association for Computational Linguistics, 2020, pp. 38–45. DOI: 10.18653/v1/2020.nlp4convai-1.5.

[9] F. Pedregosa, G. Varoquaux, A. Gramfort, *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011, documentation et guide utilisateur des modèles (CountVectorizer, TfidfVectorizer, KNeighborsClassifier, LinearSVC, MLPClassifier), documentation des modules ("Probability calibration").

[10] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using Siamese BERT-networks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Association for Computational Linguistics, 2019, pp. 3982–3992. DOI: 10.18653/v1/D19-1410.

[11] D. Cer, Y. Yang, S. Kong, *et al.*, "Universal sentence encoder," *CoRR*, 2018. arXiv: 1803.11175.

[12] M. Iyyer, V. Manjunatha, J. Boyd-Graber, and H. Daumé III, "Deep unordered composition rivals syntactic methods for text classification," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Association for Computational Linguistics, 2015, pp. 1681–1691. DOI: 10.3115/v1/P15-1162.

[13] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, Curran Associates Inc., 2017, pp. 6000–6010.

[14] M. Sugiyama, "Dimensionality reduction of multimodal labeled data by local fisher discriminant analysis," *Journal of Machine Learning Research*, vol. 8, no. 37, pp. 1027–1061, 2007.

[15] W. de Vazelhes, C. Carey, Y. Tang, N. Vauquier, and A. Bellet, "Metric-learn: Metric Learning Algorithms in Python," *Journal of Machine Learning Research*, vol. 21, no. 138, pp. 1–6, 2020.

[16] E. Liberis. "Intent classification with geometrically-friendly embeddings." (2020), [Online]. Available: https://www.polyai.com/intent-classification-geometrically-friendly-embeddings/.

[17] K. Q. Weinberger and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification," *Journal of Machine Learning Research*, vol. 10, pp. 207–244, 2009.

[18] M. Bronzi, J. Pinto, J. Ghosn, *et al.* "A question answering system in response to the covid-19 crisis." (2020), [Online]. Available: https://drive.google.com/file/d/1chJyp6mEhieL13EzbX_Xpf8m3MenDuDE/view.

[19] S. Larson, A. Mahendran, J. J. Peper, *et al.*, "An evaluation dataset for intent classification and out-of-scope prediction," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 1311–1316. DOI: 10.18653/v1/D19-1131.