The Thirty-First AAAI Conference on Innovative Applications of Artificial Intelligence (IAAI-19)

# Querying NoSQL with Deep Learning to Answer Natural Language Questions

**Sebastian Blank,**[1] **Florian Wilhelm,**[1] **Hans-Peter Zorn,**[1] **Achim Rettinger**[2]
inovex GmbH,[1] Karlsruhe Institute of Technology,[2]
sblank@inovex.de, fwilhelm@inovex.de, hzorn@inovex.de, rettinger@kit.edu

**Abstract**

Almost all of today's knowledge is stored in databases and thus can only be accessed with the help of domain specific query languages, strongly limiting the number of people which can access the data. In this work, we demonstrate an end-to-end trainable question answering (QA) system that allows a user to query an external NoSQL database by using natural language. A major challenge of such a system is the non-differentiability of database operations which we overcome by applying policy-based reinforcement learning. We evaluate our approach on Facebook's bAbI Movie Dialog dataset and achieve a competitive score of 84.2% compared to several benchmark models. We conclude that our approach excels with regard to real-world scenarios where knowledge resides in external databases and intermediate labels are too costly to gather for non-end-to-end trainable QA systems.

## Introduction

Almost all of today's knowledge is stored and organized in various types of databases but the ability to access this knowledge is limited to the ones who have mastered the corresponding query language. Conversational agents promise to overcome this barrier by allowing humans to query these resources with natural language thus fostering the democratization of knowledge. The broad domain of conversational agents can be further subdivided by looking at the complexity of human-machine interaction. Goal-oriented dialog systems follow the purpose of gaining information through conversation in order to complete a specific task. Hence, they are typically applied in short dialogs that are domain specific. In contrast, non-goal-oriented agents are designed to have extended conversations without any restriction to a specific domain.

In our opinion, both categories represent long-term goals in artificial intelligence, but they depend on the practical feasibility of another type of natural language processing (NLP) task that is much simpler. We define a *KBQueryBot* as an agent that fulfils the task of translating a query, posed in natural language, into the domain-specific query language of an external knowledge base (KB) in order to retrieve factoid answers in a single-shot manner. This task can also be seen as a semantic parsing problem.

To enable the application of a KBQueryBot in real-world use-cases, it needs to satisfy two conditions. Firstly, it needs to be trainable in an end-to-end fashion. Labelled data is always a scarce resource and any kind of additional intermediate labelling to fulfil this task is too costly for practical applications since it requires expert knowledge about the query language and scales poorly (Liang et al. 2016). On the other hand, judging the correctness of a retrieved result can be conducted having only domain knowledge and thus even *active learning*-powered techniques can be applied.

Typically, there are two different ways how to interact with a KB. Agents can either query an external KB with the corresponding query language or make use of an internal memory that allows for more flexible means of lookups, e.g. probabilistic lookup. This requires a conversion of the externally available database into the internal memory rendering the user management and access control capabilities of modern databases useless. However, lacking these capabilities results in severe issues regarding data security and privacy. Therefore, our second requirement is that a KBQueryBot needs to cope with today's prevalent databases like Elasticsearch, MongoDB and PostgreSQL.

This work contributes to engineering efforts to establish a standard machine learning model and training procedure for all kinds of structured database queries via natural language questions. We propose a KBQueryBot which relies on a sequence-to-sequence model (Sutskever, Vinyals, and Le 2014) that is augmented with pointer attention (Bahdanau, Cho, and Bengio 2014; Vinyals, Fortunato, and Jaitly 2015). We apply the *REINFORCE*-algorithm (Williams 1992) which enables us to overcome the problem of non-differentiable database operations and train our model on question-answer pairs. Our approach is closely related to Zhong, Xiong, and Socher (2017) and also uses policy gradient but differs in avoiding the usage of intermediate labels. Our model integrates with Elasticsearch (ES) which is one of the most popular NoSQL databases[1] and dominant in search-related applications. Typically, users query ES with the help of graphical user interfaces designed for the specific use-case. However, we notice an increased interest of business customers to make their KBs accessible by natural language input to provide an improved user experience.

[1]https://db-engines.com/en/ranking

## Related Work

Since the task of a KBQueryBot is to reformulate a single user question into a corresponding structured query, it can be qualified as a specialised QA system which does not take into account any dialog history. As the focus of our work lies on the retrieval of data stored in an external KB, it can clearly be separated from QA systems that only leverage dialog history to answer a question (Eric and Manning 2017).

Recent QA systems that incorporate external knowledge can be distinguished by the way they perform the actual lookup. Dhingra et al. (2016) proposed the term *soft-KB lookup* for using an attention mechanism as introduced by Bahdanau, Cho, and Bengio (2014) to compute a probability distribution over the index of the KB. The index with the highest probability is then retrieved and the corresponding data presented as result. In contrast, generating a structured query which is then executed is known as *hard-KB lookup*.

**Soft-KB lookup.** Memory Networks (MemNN) represent a neural architecture that is extended with an internal memory component (Weston, Chopra, and Bordes 2014; Sukhbaatar et al. 2015). Bordes et al. (2015) proposed their application for soft-KB lookup by ingesting the external KB into the internal memory in a pre-processing step. They showed promising performance against multiple benchmark models, second only to a subgraph embedding based QA system by Bordes, Chopra, and Weston (2014), on the large-scale and domain specific Movie Dialog dataset (Dodge et al. 2015). Miller et al. (2016) confirmed the qualities of MemNN in another benchmark on a modified Movie Dialog dataset where MemNN even outperformed the QA system by Bordes, Chopra, and Weston (2014). Due to these promising results, we use MemNNs as a benchmark for our hard-KB approach on the same dataset.

Dhingra et al. (2016) introduced KB-InfoBot which also performs a soft-KB lookup, but its lookup policy is trained with reinforcement learning similar to our work. They observed that their model tends to fail from random initialization and overcame this problem by using imitation learning to mimic hand-crafted agents. This required human interaction diminishes the advantage of end-to-end trainability.

**Hard-KB lookup.** Non-differentiable database operations are the major shortcoming of hard-KB lookup, since they complicate the end-to-end trainability of the corresponding models. Wen et al. (2016) leveraged intermediate labels to train different components of their modular dialog system separately. In the same manner, recent implementations of MemNNs (Bordes, Boureau, and Weston 2017) and other neural architectures (Dong and Lapata 2016; Xu, Liu, and Song 2017) trained their models on question-query pairs. Thereby, they resolved the broken differentiability but encountered an increased cost caused by additional data annotation. Especially for large-scale datasets this represents an important limitation. Li et al. (2017) also make use of intermediate labels between the language understanding and dialog management to overcome the non-differentiability in their goal-oriented dialog system.

In the same manner, Zhong, Xiong, and Socher (2017) assume the availability of the query ground truth (intermediate labels) and the database response. They propose Seq2SQL, which is a modular approach to translate natural language questions into SQL queries and generalizes across different table schemas. Seq2SQL consists of three modules. The first module is responsible to identify an aggregator, e.g. count or max, while the second module identifies the column that will be used in the select operator. Both modules are trained on question-query pairs. The third module is designed to produce the where-clause. Since arguments in the where-clause can be swapped, this ambiguity is handled by policy-based reinforcement learning trained on question-answer pairs directly. Our work resembles their architecture but uses only reinforcement learning to achieve end-to-end trainability.

The work of Williams and Zweig (2016) applies policy-based reinforcement without intermediate labels in task-oriented dialog system. It translates human commands into API calls also taken into account the dialog history. Our work differs in that no separately trained entity extraction is needed and the resulting query is solely based on the user's input without needing any additional predefined business logic.

## Model

The overall architecture of our *SeqPolicyNet* model which solves the introduced KBQueryBot task is illustrated in Figure 1. The core component is an attention-based pointer network (Sutskever, Vinyals, and Le 2014) which fills slots of a predefined ES query template. The fixed-length output sequence is composed of elements which either select a column name of the KB or point to a token of the question for entity extraction. Thus, we will refer to them as *selection* or *extraction* outputs. With the help two extraction outputs acting as start and end pointer, it is thus possible to extract an entity composed by a span of tokens. For instance, given the question "Who was the director of Gone With the Wind?" the start pointer would be *Gone* and the end pointer *Wind*. Our model can thus be seen as a pointer network for entity extraction (Zhong, Xiong, and Socher 2017) with reduced complexity.

### Pointer Network

As input SeqPolicyNet receives the natural language questions $x^q$ and the columns $x^c$ of the KB which are lowercased and concatenated to $x = [x^c, x^q]$. This input is used to produce an output sequence $y = [y_1, ..., y_T]$ which is then used to fill the query template $query(\cdot)$. Executing the final query returns a response set $\omega$. Furthermore, the set $\psi$ denotes the ground truth answer to the question $x^q$.

The sequence-to-sequence network leverages an encoder-decoder approach with pointer attention (Vinyals, Fortunato, and Jaitly 2015). The attention $a_t^s$ at the $t$-th output token over $s$ potential classes is formally defined as

$$\begin{aligned} u_t^s &= v \tanh\left(W_s \bar{h}_s + W_t h_t\right) \\ a_t^s &= \operatorname{softmax}(u_t^s), \end{aligned} \tag{1}$$

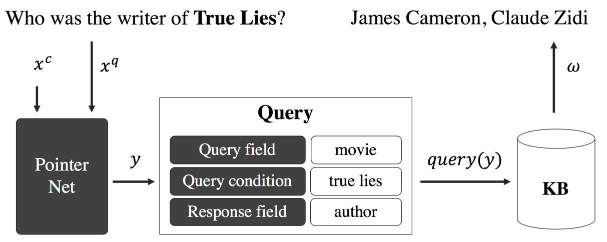

Figure 1: System architecture of SeqPolicyNet: The tokens $x^q$ of the user's question are fed into a pointer attention network (PointerNet) together with the column names $x^c$ of the KB. The model extracts entities from the question and chooses the corresponding column names in order to fill a query template $query(\cdot)$. The model executes the final query and presents the results to the user.

where $v, W_s$ and $W_t$ are trainable parameters and a decoder hidden state $h_t$ is scored against an encoder hidden state $\bar{h}_s$. Pointer attention significantly decreases the output space and therefore reduces the computational effort of the attention mechanism (Vinyals, Fortunato, and Jaitly 2015).

Denoting with $\theta$ the trainable parameters of our model, the attention weights are then used to formulate a stochastic policy

$$\pi\big(y_t \mid y_1, ..., y_{t-1}, x, \theta\big) = a_t, \tag{2}$$

which defines the probability of choosing the next token of the output sequence $y_t$. Sampling $y_t \sim \pi$ from this policy corresponds to either the selection of a column name $x^c$ or pointing to the index of a word in the input question $x^q$.

## Training with *REINFORCE*

Since the database operation $query(y)$ is non-differentiable, we train the pointer network with the policy-based *REINFORCE*-algorithm (Williams 1992). The network weights are updated according to

$$\theta \leftarrow \theta + \alpha R \nabla_\theta \sum_{t \in T} \log \pi\Big(y_t \mid y_1, ..., y_{t-1}, x, \theta\Big), \tag{3}$$

where $\alpha$ is the learning rate and $R$ the reward of our episodic task. In order to escape local minima Strehl and Littman (2005) induce exploration to rarely visited state-action pairs by providing count-based reward boni. We also introduce an exploration bonus $R^+$ motivated by upper confidence bounds (Lai and Robbins 1985) which is defined as

$$R^+ = \begin{cases} \sqrt{\frac{2*\log(n)}{count(y_t)}}, & \text{if boni are active} \\ 0, & \text{else} \end{cases}, \tag{4}$$

where $n$ is the mini-batch sample size and $count(y_t)$ is the number of times that $y_t$ was chosen at index $t$ of the mini-batch iteration. The exploration boni is only applied to the selection outputs since they suffer the most from imbalanced data with predominant column names.

The reward function $R = [-2, -1, z]$ is defined as

$$R = \begin{cases} -2 + R^+, & \text{if } query(y) \text{ is invalid} \\ -1 + R^+, & \text{if } query(y) \text{ is valid yielding} \\ & \text{an incorrect result} \\ z + R^+, & \text{if } query(y) \text{ is valid yielding} \\ & \text{the correct result} \end{cases}, \tag{5}$$

where $z$ is a positive reward that was varied during our experiments. A query is considered a valid query if its operation yields a non-empty response $\omega$ and causes no error. Furthermore, we consider a response $\omega$ to be correct if it matches with the ground truth answer $\psi$.

## Experiments

For our experiments we chose Elasticsearch (ES) which is one of the most popular NoSQL databases powering the search capabilities of most of today's websites. Since ES is a document-oriented database the columns of a relational database correspond to fields within a document. ES provides a domain specific query language based on JSON which we use to define a query template $query(\cdot)$. To compare our results we consider the accuracy of valid queries Acc$_{vq}$ which is the percentage of questions generating a valid query and the correct results accuracy Acc$_{cr}$ being the percentage of questions for which we retrieve a correct result.

### Movie Dialog Dataset

Since SeqPolicyNet is designed for the task of answering factoid questions without any relation to previous dialog, we chose the QA dataset by Dodge et al. (2015) which focuses on the domain of movies and movie related entities. The dataset comes in a predefined split of 96k question-answer pairs for training and 10k pairs for validation and testing respectively. In addition to the question-answer data, the Movie Dialog dataset also contains metadata of 17,340 movies in 11 categories, e.g. movie title, author, genre, language.

The dataset was ingested into an ES instance and we defined $query(\cdot)$ as a template with three slots. The query field (QF) defines the document field which is used to match the query condition (QC) whereas the response field (RF) defines the field of the document containing the answer. Therefore, we set the pointer network to generate four outputs. Two selection outputs determine QF & RF and two extraction outputs define a start and end pointer in order to extract the entity for QC from the question. Besides the question $x^q$ the network receives the 11 categories as column names $x^c$ for selection. Table 1 illustrates an example from the Movie Dialog dataset.

Furthermore, in order to test the ability to generalize to unseen questions, we additionally define a training set of 81k question-answer pairs by removing two random question formulations for each *class of questions*, i.e. questions sharing the same QF and RF values, from the original 96k dataset. We will use the size of the training set, i.e. 96k or 81k, to refer to the original training and our reduced training set.

| | **Running Example** |
|---|---|
| **Question** | Who was the writer of True Lies? |
| **Query Field** | Movie |
| **Query Condition** | True Lies |
| **Response Field** | Author |
| **Answer** | James Cameron, Claude Zidi |

Table 1: For the given question the query field and condition as well as the response field are extracted in order to generate the corresponding query and retrieve the answer from ES.

## Training Procedure

For our benchmark, we implement PointerNet with a bi-directional two-layer LSTM as encoder and a uni-directional two-layer LSTM as decoder, where all recurrent layers consist of 100 units. We apply a dropout ratio (Srivastava et al. 2014) of 0.3 in the encoder and decoder. The ADAM optimizer (Kingma and Ba 2014) was used to train the network with a learning rate $\alpha$ of 0.0001 and varying mini-batches of size 128 as wells as 256. To embed the words, we opted for the 300-dimensional GloVe embeddings, pre-trained on the 42B corpus (Pennington, Socher, and Manning 2014). All runs were performed for 50 epochs with an early stopping mechanism based on the accuracy of correct results on the validation set. We varied the reward function for queries yielding a correct results by setting $z \in \{1, 5, 25\}$ during our experimentation phase. We found that this is an important parameter with respect to the capability of overcoming local optima.

We evaluate three variants of our model. The first variant uses $z = 1$ in the reward function without boni $R^+$ resembling the one of Zhong, Xiong, and Socher (2017). The second variant differs from that by a highly increased reward $z = 25$ for correct results. The third variant additionally activates the exploration boni $R^+$. To quantify the benefits of intermediate labels we also trained our model directly on intermediate labels, i.e. the query ground truth (QF, start pointer, end pointer, RF). This model, denoted as *PointerNet-IL*, runs out of competition according to the goals of this work but gives us a comparison based on equal footing to soft-KB QA systems.

## Baselines

For comparison with the state-of-the-art we provide the results reported by Dodge et al. (2015). They applied a standard LSTM, supervised embeddings and MemNNs, as well as ensembles of the latter ones. Furthermore, they presented the performance of a QA system as described by Bordes, Chopra, and Weston (2014). All models reported by Dodge et al. (2015) perform soft-KB lookup.

## Results

The results of our experiments are presented in Table 2. Training our model with the reward setting of Zhong, Xiong, and Socher (2017) gets trapped in local optima and is not able to outperform the baseline LSTM. We observed that it focuses on the generation of valid queries by choosing the predominant column names for QF and RF.

| **Methods** | **Acc**$_{vq}$ | **Acc**$_{cr}$ |
|---|---|---|
| Baseline LSTM[2] | - | 6.5% |
| Supervised embeddings (ensemble)[2] | - | 43.6% |
| Supervised embeddings[2] | - | 50.9% |
| MemNN[2] | - | 79.3% |
| MemNN (ensemble)[2] | - | 83.5% |
| QA system[2] [3] | - | 90.7% |
| SeqPolicyNet $z = 1$ | 99.0% | 0.0% |
| SeqPolicyNet $z = 25$ | 83.3% | 54.5% |
| SeqPolicyNet $z = 25$ & $R^+$ | 91.3% | 84.2% |
| PointerNet-IL | 93.9% | 90.6% |

Table 2: Comparison of the evaluated models with respect to the share of produced valid queries Acc$_{vq}$ and accuracy Acc$_{cr}$ in retrieving correct results.

Adjusting the reward function to $z = 25$ led to an improved performance and SeqPolicyNet was able to outperform the supervised embeddings reported by Dodge et al. (2015). We observed that the prediction of QC was much improved. The length of the span defined by the start and end pointer to extract QC nearly doubled and the right entities were extracted. The prediction of RF showed an enlarged entropy indicating a balanced distribution over several columns. We did not notice a change in behaviour of predicting QF as the model still predicts a single predominant column name. Overall the share of valid queries dropped while the percentage of correct results increased drastically. We attribute the switch from focusing on valid queries ($z = 1$) to corrects results ($z = 25$) to the weight updates, which depend on the average of the expected return over trajectories in the mini-batch (Arulkumaran et al. 2017). It appears that this averaging causes the vanishing feedback signal in case of low positive rewards which are very sparse. These observations confirm the contemplations by Mou et al. (2016) and Liang et al. (2016) who report that *REINFORCE* gets trapped in local optima easily due to sparse rewards. However, they do not link this to the scale of rewards.

Activating the reward boni that induces exploration for QF and RF boosted the Acc$_{cr}$ performance of our model, now outperforming the ensemble of MemNNs. Figure 2 shows that SeqPolicyNet with reward boni needs more epochs to achieve the same accuracies than without boni but keeps on improving its policy when the performance of SeqPolicyNet without boni already saturates.

Figure 3 illustrates the attention of SeqPolicyNet when predicting RF for a question example. Until epoch 11 the attention weights over the KB columns are almost uniformly distributed. Starting with epoch 12 SeqPolicyNet attributes an increased importance to the columns *director* and *author* which are thus more often selected. This change in epoch 12

[2]Reported by Dodge et al. (2015)

[3]Based on Bordes, Chopra, and Weston (2014)

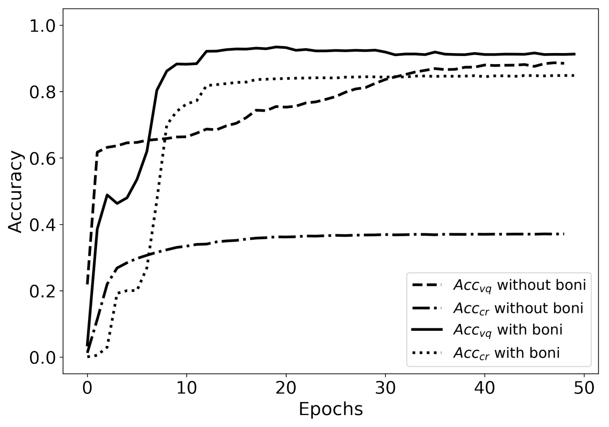

Figure 2: Evaluation of SeqPolicyNet with and without exploration boni $R^+$ regarding the percentage of valid queries $\text{Acc}_{vq}$ and correct results $\text{Acc}_{cr}$ over the number of training epochs on the 96k dataset.

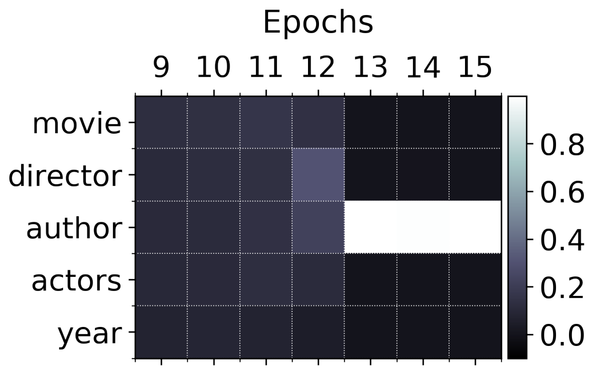

Figure 3: Visualization of SeqPolicyNet's attention over a subset of KB columns when asked to predict the response field for the question: "Who was the author of true lies?".

can also be observed as a small jump in the accuracies of SeqPolicyNet with boni as seen in Figure 2.

Although SeqPolicyNet with reward boni shows good results it is not able to achieve the performance of the soft-KB QA system by Bordes, Chopra, and Weston (2014). Leveraging intermediate labels which moves our end-to-end trainable hard-KB setting closer to the training conditions of soft-KB allows our PointerNet-IL model to match the performance level of Bordes, Chopra, and Weston (2014). This highlights the advantage of having a differentiable cost functional in supervised learning compared to a reinforcement learning approach.

During our evaluation, we also noticed that some ambiguities in the Movie Dialog dataset are the reason that about $4\%$ of all questions have no clear answer. For instance, the KB contains four movies named *the three musketeers* released in 1973, 1993, 1994 and 2011 respectively. Hence, the question "What was the release date of the film the three musketeers?" is highly ambiguous and the correct response is not well-defined in our setup.

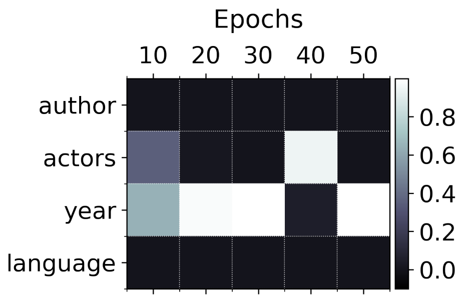

Figure 4: Visualization of SeqPolicyNet's attention over a subset of KB columns when asked to predict the response field for the question: "The film geography club starred which actors?".

## Generalization

In order to test the capabilities of our model to generalize on unseen questions we evaluated SeqPolicyNet also on the 81k dataset. Table 3 shows the results for the best performing variant of SeqPolicyNet, i.e. $z = 25$ and activated reward boni.

| Train Size | Questions | $\textbf{Acc}_{vq}$ | $\textbf{Acc}_{cr}$ |
|---|---|---|---|
| | Known patterns | 92.1% | 81.9% |
| 81k | Unknown patterns | 91.4% | 69.0% |
| | Total | 91.6% | 80.1% |
| 96k | Total | 91.3% | 84.2% |

Table 3: Comparison of SeqPolicyNet's accuracy on the 81k dataset for testing generalisation and the original 96k dataset.

We observed that the performance on the total amount of questions in the test set was reduced by a small margin. As to be expected, there was no difference for known question patterns which were already in the training set. However, the $\text{Acc}_{cr}$ performance decreased drastically for unknown question patterns that were not in the training set whereas the share of valid queries remained on a similar level as before.

This was mainly due to failures in predicting the RF. We found that a significant share of the incorrect results was due to questions that needed RF to be the column name *actor*. Further analysis revealed that there are only seven different question patterns of this kind and we removed two of them in the 81k dataset. Figure 4 illustrates the prediction of RF on an example question of this class. While the correct RF *actor* was selected in epoch 40, the policy switches back to *year* in the following epochs. We conclude that for this class of questions the training set is too scarce for the model to learn. To support this conclusion, we also conducted several experiments with our dataset reduced to a training set of only 10k samples. While the performance of PointerNet-IL decreased only by a small margin of $1.5$ percentage points, SeqPolicyNet's accuracy of correct results dropped by $53.2$ points. This is due to the sample inefficiency of reinforcement learning (Sukhbaatar et al. 2017).

## Conclusion

This work integrates an AI-technology, namely attention-based pointer networks, with the well established NoSQL database Elasticsearch. Our end-to-end trainable, hard-KB lookup *SeqPolicyNet* model outperforms several baseline models on the Movie Dialog dataset. SeqPolicyNet generalizes even on question patterns unseen during training given enough samples for each class of question.

Almost all applications with search capabilities have graphical user interfaces that convert user inputs to domain-specific queries. We propose to complement these user interfaces with additional text inputs for the user's inquiry that will be translated by a KBQueryBot like SeqPolicyNet. It is our strong belief that KBQueryBots are an emerging application in the era of mobile devices with voice input-based interfaces. Based on the results of this work we have built a KBQueryBot for an e-commerce customer as proof-of-concept. After further evaluation and improvements, we hope to deploy a KBQueryBot soon in production.

## References

Arulkumaran, K.; Deisenroth, M. P.; Brundage, M.; and Bharath, A. A. 2017. A Brief Survey of Deep Reinforcement Learning. *arXiv preprint arXiv:1708.05866*.

Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv preprint arXiv:1409.0473*.

Bordes, A.; Usunier, N.; Chopra, S.; and Weston, J. 2015. Large-scale Simple Question Answering with Memory Networks. *arXiv preprint arXiv:1506.02075*.

Bordes, A.; Boureau, Y.-L.; and Weston, J. 2017. Learning End-to-End Goal-Oriented Dialog. *arXiv preprint arXiv:1605.07683*.

Bordes, A.; Chopra, S.; and Weston, J. 2014. Question Answering with Subgraph Embeddings. *arXiv preprint arXiv:1406.3676*.

Dhingra, B.; Li, L.; Li, X.; Gao, J.; Chen, Y.-N.; Ahmed, F.; and Deng, L. 2016. Towards End-to-End Reinforcement Learning of Dialogue Agents for Information Access. *arXiv preprint arXiv:1609.00777*.

Dodge, J.; Gane, A.; Zhang, X.; Bordes, A.; Chopra, S.; Miller, A. H.; Szlam, A.; and Weston, J. 2015. Evaluating Prerequisite Qualities for Learning End-to-End Dialog Systems. *arXiv preprint arXiv:1511.06931*.

Dong, L., and Lapata, M. 2016. Language to Logical Form with Neural Attention. *arXiv preprint arXiv:1601.01280*.

Eric, M., and Manning, C. D. 2017. A Copy-Augmented Sequence-to-Sequence Architecture Gives Good Performance on Task-Oriented Dialogue. *arXiv preprint arXiv:1701.04024*.

Kingma, D. P., and Ba, J. 2014. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980*.

Lai, T. L., and Robbins, H. 1985. Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics* 6(1):4–22.

Li, X.; Chen, Y.-N.; Li, L.; Gao, J.; and Celikyilmaz, A. 2017. End-to-End Task-Completion Neural Dialogue Systems. *arXiv preprint arXiv:1703.01008*.

Liang, C.; Berant, J.; Le, Q. V.; Forbus, K. D.; and Lao, N. 2016. Neural Symbolic Machines: Learning Semantic Parsers on Freebase with Weak Supervision. *arXiv preprint arXiv:1611.00020*.

Miller, A. H.; Fisch, A.; Dodge, J.; Karimi, A.-h.; Bordes, A.; and Weston, J. 2016. Key-Value Memory Networks for Directly Reading Documents. *arXiv preprint arXiv:1606.03126*.

Mou, L.; Lu, Z.; Li, H.; and Jin, Z. 2016. Coupling Distributed and Symbolic Execution for Natural Language Queries. *arXiv preprint arXiv:1612.02741*.

Pennington, J.; Socher, R.; and Manning, C. D. 2014. Glove: Global Vectors for Word Representation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1532–1543.

Srivastava, N.; Hinton, G. E.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research* 15:1929–1958.

Strehl, A. L., and Littman, M. L. 2005. A theoretical analysis of Model-Based Interval Estimation. In *International Conference on Machine Learning (ICML)*, 856–863.

Sukhbaatar, S.; Szlam, A.; Weston, J.; and Fergus, R. 2015. End-To-End Memory Networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2440–2448.

Sukhbaatar, S.; Lin, Z.; Kostrikov, I.; Synnaeve, G.; Szlam, A.; and Fergus, R. 2017. Intrinsic Motivation and Automatic Curricula via Asymmetric Self-Play. *arXiv preprint arXiv:1703.05407*.

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, 3104–3112.

Vinyals, O.; Fortunato, M.; and Jaitly, N. 2015. Pointer Networks. In *arXiv preprint arXiv:1506.03134*.

Wen, T.-H.; Vandyke, D.; Mrksic, N.; Gasic, M.; Rojas-Barahona, L. M.; Su, P.-H.; Ultes, S.; and Young, S. 2016. A Network-based End-to-End Trainable Task-oriented Dialogue System. *arXiv preprint arXiv:1604.04562*.

Weston, J.; Chopra, S.; and Bordes, A. 2014. Memory Networks. In *arXiv preprint arXiv:1410.3916*.

Williams, J. D., and Zweig, G. 2016. End-to-end LSTM-based dialog control optimized with supervised and reinforcement learning. *arXiv preprint arXiv:1606.01269*.

Williams, R. J. 1992. Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Machine Learning* 9:229–256.

Xu, X.; Liu, C.; and Song, D. 2017. SQLNet: Generating Structured Queries From Natural Language Without Reinforcement Learning. *arXiv preprint arXiv:1711.04436*.

Zhong, V.; Xiong, C.; and Socher, R. 2017. Seq2SQL: Generating Structured Queries from Natural Language using Reinforcement Learning. *arXiv preprint arXiv:1709.00103*.