Revista Romana de Interactiune Om-Calculator 11(2) 2018, 145-162

Improving multi-choice question answering by identifying essential terms in questions

George-Sebastian Pirtoaca, Stefan Ruseti, and Traian Rebedea

University Politehnica of Bucharest
Splaiul Independentei nr. 313, sector 6, 060042, Bucuresti
E-mail: george.pirtoaca@stud.acs.upb.ro, {stefan.ruseti, traian.rebedea}@cs.pub.ro

Abstract. This paper describes a method for identifying the most relevant or essential terms in a question: the words that define the meaning of the question and whose removal makes it impossible to be answered correctly even by human agents. We use an artificial neural network architecture built upon semantic and syntactic features extracted from the question. The neural network prediction represents the degree to which the given term is essential to the question. The model has been trained and validated on a dataset consisting of 2233 questions and about human 18000 labeled terms with essentialness information. It achieves an F1 score of 0.80 which is similar to other state-of-the-art approaches but requires fewer features (15 compared to 120 features used by similar works) and is much easier to train. We further show how using only essential term information can improve the accuracy of a multi-choice question answering system, based on standard information retrieval (IR) techniques, by up to 4%.

Keywords: Natural Language Processing, Question Answering, Information Retrieval; Essential Terms, Dependency Parsing, Neural Networks.

1. Introduction

Learning what is essential in a question is a fundamental problem when dealing with question answering (QA) systems. Our proposed hypothesis suggests that by using relevant information from questions (words that are essential to a question compared to words that can be actually removed from the question without altering its core meaning) the accuracy of a QA system that relies only on information retrieval (IR) techniques can be significantly improved. Naïve IR approaches can be adjusted by filtering out terms from the question that are not relevant. For example, consider the question: “When carbon and oxygen combine chemically, the mass of the product is”. One can argue that the essential terms are: “carbon”, “oxygen”, “mass”, and “product” with “mass” probably being the most important of them. We can think of a

term being essential if and only if by dropping it from the question the global meaning is completely altered and a human agent cannot possibly answer the question correctly (because vital pieces from the question are missing). If we filter out “mass”, “carbon”, and “oxygen”, then the question remains: “When and combine chemically, the of the product is”. It is clear that we cannot answer this question anymore because its core meaning has been altered.

Following recent work in this area (Khashabi et al., 2017), we propose an artificial neural network architecture that is able to identify essential terms in a question with state-of-the-art accuracy. We use the dataset collected by Khashabi et al. (2017) consisting of 2233 questions where each term is labeled with a number between 0 and 5 (from not essential to very important to the question at hand). The output of the neural network is a probability distribution over the classes from 0 to 5 with the following interpretation: what is the probability that the term has importance level L, where L ranges from 0 to 5. In order to predict the essentialness of a term, one can compute the expected value of the network output. Considering the size of the dataset, a recurrent neural network alone – like a Long Short-Term Memory (LSTM) described by Hochreiter and Schmidhuber (1997) – is not able to extract relevant information. The model is very likely to overfit on the training dataset and to perform poorly on new, unseen questions due to the small size of the training set. It turns out that in order to achieve high performance, the network needs to be fed with semantic and syntactic features computed *a priori* (before the network is trained) from the question. Ideally, these features could have been learned by the neural network supposing we would had enough data, but this is not the case: only 2223 questions with about 15 terms per question is way less than what a neural network requires in order to learn complex hypothesis.

The paper continues with an overview of the most representative work in question answering systems using multi-choice science questions and the importance of essential terms in this context. In Section 3 we present the proposed neural based classifier using syntactic and semantic information to extract essential terms from a question. Section 4 offers an overview of the results, both in terms of accuracy for computing the essentialness score for a term and in the improvement this method offers to a baseline IR question answering system. Future research directions are briefly discussed at the end.

2. Related work

The main purpose of this paper is to provide a way for question answering (QA) systems to identify the core concepts in a question and then to employ this information to improve the performance of the QA system. Question answering is an important and difficult task for the artificial intelligence world, especially when we talk about difficult questions which require a form of inference to detect the correct answer. A lot of work has been done in order to improve the information retrieval baseline and to give a better sense of what an agent could achieve with proper, structured knowledge. For example, several Markov Logic Network (MLN) systems with text-derived knowledge were built by Khot et al. (2015): a first-order logic MLN, an entity resolution MLN, and a custom approach called *Praline*. However, the best reasoning-based configuration still does not outperform a simple bag-of-words approach over Wikipedia. The two key issues are that translating from natural language to first-order logic in an efficient and accurate way is hard and that a lot of relevant rules cannot be applied because their preconditions are not fully satisfied.

A method that uses a structured inference system in which the question answering problem is formulated as an Integer Linear Program (ILP) was also proposed and tested by Khashabi et al. (2016). It works by representing knowledge in tables where each row is a predicate of arity k (number of columns) over strings, where each string is a short natural language sentence. The QA problem is viewed as an optimal sub-graph selection problem, where the algorithm tries to find the pair *(question, answer)* that best fits the knowledge base. For evaluation, an easier dataset has been used, consisting of multiple-choice questions from 12 years of the NY Regents 4th Grade science exams. However, even in this case of a much easier dataset, the ILP approach outperformed the simple IR by only 3%.

Ensemble models trying to solving the QA problem have also been proposed. Clark et al. (2016) described a model which combines all of the following: an information retrieval solver based on Lucene, statistical information using Pointwise Mutual Information (PMI), text similarity using word embeddings and a simple Support Vector Machine (SVM), and structured knowledge solvers. The model has been tested on the NY Regents 4th Grade Science exams and clearly outperforms any other candidate so far (on that particular dataset).

One of the proposed solutions that makes use of machine learning techniques is the one by Jansen et al. (2017). They have used potential answer justifications for each answer candidate not only to improve the accuracy of the system, but also to provide simple, natural language explanations for the chosen answer. These are useful when one needs very high confidence in the answer selected by the system as the correct one. For example, in the medical domain, the answer and its justification may be reviewed by a human in order to validate that it is indeed correct.

Another deep learning approach proposed by Nicula et al. (2018) is using relevant candidate contexts extracted from Wikipedia in order to answer the questions correctly. More recently, some work has been invested in learning essential terms in questions. These approaches are using either machine learning (Khashabi et al., 2017) or cognitive psychology and rule-based systems (Jansen et al., 2017).

3. Proposed solution

We propose a neural network architecture that receives a question, along with syntactic and semantic information extracted from it, and a term from the question and predicts whether the term is essential or not for the meaning of the question (a score from 0 to 5). The syntactic and semantic information is very important in improving model's accuracy, as we will see in the evaluation section. The architecture of the neural network has been designed around the following features:

a. *The question itself* which has been tokenized and then embedded into a 50-dimensional space using pre-trained GloVe (Pennington et al., 2014) word vectors. The question is then passed through a Long Short-Term Memory (Hochreiter and Schmidhuber, 1997) recurrent neural network.

b. *The concatenation of all the answers* (each multi-choice question in the dataset has 4 possible answers), which are also embedded into a 50-dimensional space using pre-trained GloVe word vectors and then passed through an LSTM as above, with independent weights.

c. *The term*, which is projected into a 50-dimensional space using pre-trained GloVe (same embeddings as the question and the answer).

d. *Is science term?* – We use a list of 9144 science terms as provided by Khashabi et al. (2016). The dataset we use for training and validation (Khashabi et al., 2017) contains science-related questions, therefore, not

surprisingly, this indicator turns out to be very useful as many relevant terms are, indeed, science terms (atom, gravity, etc.).

e. *Is the term a stop word?* – We use the SMART stop words list (Salton, 1971) along with the stop words identified by spaCy[12] after sentence parsing.

f. *Concreteness rating* – A number between 1 (highly abstract) and 5 (highly concrete) following the work by Brysbaert et al. (2014). As indicated by previous analysis (Jansen et al., 2017), words that are approximately 50% to 80% concrete tend to be used for reasoning about science questions and are often the core concepts of a question.

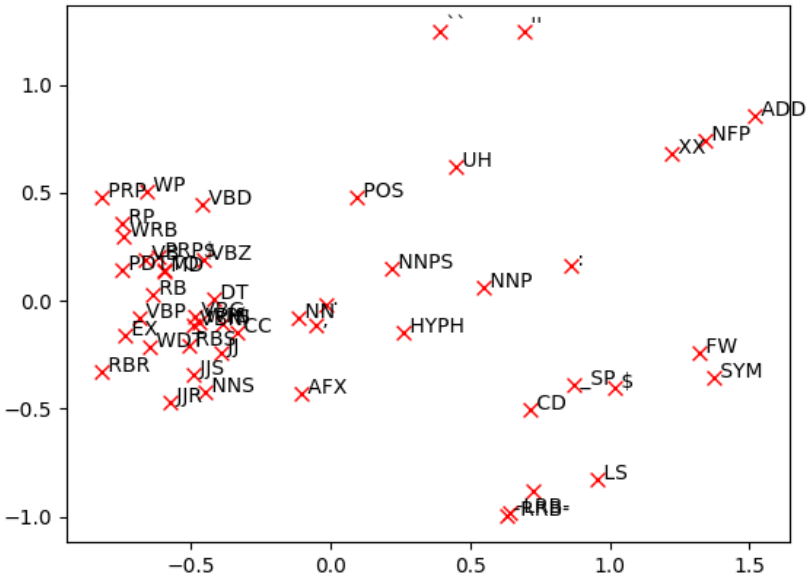

Figure 1. POS embeddings projected on a 2D space using PCA

g. *Part of speech (POS)* – Each term is tagged with a part of speech (using the Penn Treebank II tag set with about 60 parts of speech types). In order to include this feature in the neural network, we chose to embed each POS tag into a 5-dimensional real space. This is due to the fact that a 60-dimensional one-hot vector would introduce too many parameters in the network and the dataset is not that large to ensure a proper training. We want to use as few parameters as possible while keeping the expressive power as

[12] https://spacy.io/ (last accessed November 2018)

high as possible. The POS embeddings have been computed using the skip-gram model (Mikolov et al., 2013) over the question answering ARC Corpus (Clark et al., 2018). The intuition behind this approach is that similar POS tags (like verb past tense and verb past participle) tend to behave similarly in closely related contexts – one can easily swap a past tense form verb with the past participle form of the same verb when it comes to reasoning (e.g. "I went to school" and "I have been to school"). We can examine the similarities the model has learned by projecting the 5-dimensional space into a 2-dimensional space using principal component analysis (PCA) – see Figure 1.

h. *Dependency relation (DEP)* connecting the term to its parent in the dependency parse tree. We used spaCy to assign dependency relations to each term in the question. The English dependency labels follow the CLEAR Style by ClearNLP (Choi and Palmer, 2012) which is closely related to the Universal Stanford Dependencies (De Marneffe et al., 2014). Because the number of dependency labels is quite large (about 65) we used the same strategy as for POS tagging. In Figure 2, we can observe that similar dependencies are grouped in the same region (e.g. *csubjpass* and *nsubjpas*, *nummod* and *nmod*). This is exactly what we want to capture in the dependency embeddings.

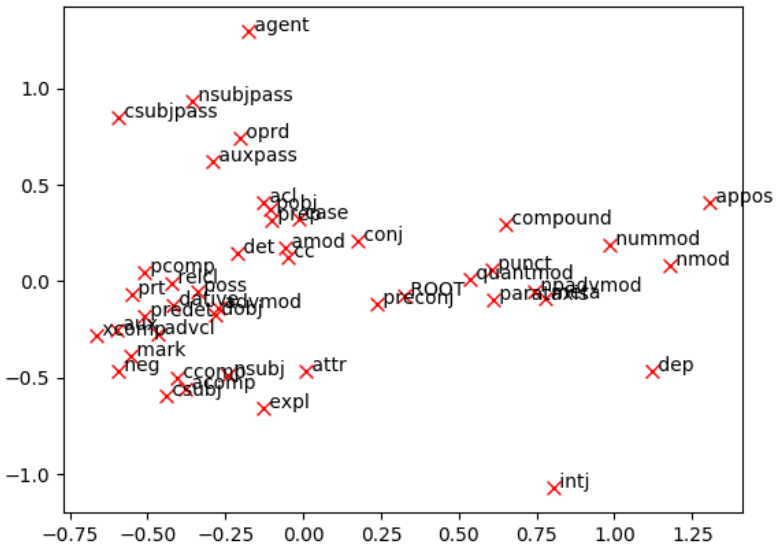

Figure 2. DEP embeddings projected on a 2D space using PCA

i. *Degree centrality* – For each vertex (corresponding to a term) in the dependency graph, we count the number of edges connected to that vertex (the degree). We then normalize all the degrees in the range [0, 1] using the softmax function. This makes possible to compare degree centrality measures from different dependency graphs. See Figure 3 and Table 1 for a complete example on the way degree centrality is computed.

j. *Closeness centrality* – Defined as the reciprocal of the sum of the lengths of the shortest paths between a node and any other node in the dependency graph. The cost of every edge is 1, independent of the dependency relation on that edge. Formally, the equation is:

$$C(x) = \frac{N}{\sum_y d(x,y)} \quad (1)$$

where N is the number of terms in the question, x, y are arbitrary terms and $d(x,y)$ is the shortest distance between x and y in the dependency graph. A term that is "close" to all other terms in the question has a closeness centrality score greater than a term which is far away from the other terms in the question. This feature is helpful for capturing the core terms in the question, which are the terms closely related to many other terms.

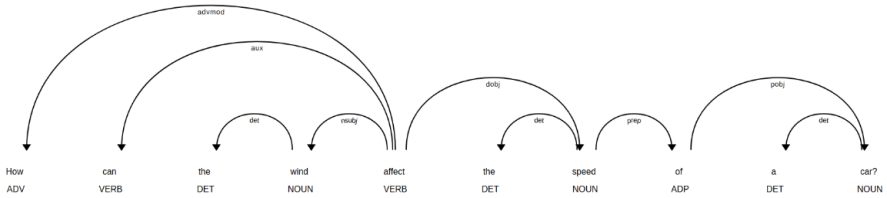

Figure 3. Dependency graph for the question "How can the wind affect the speed of a car?"

k. *Eigenvector centrality* – Which describes another way to capture how important a word is in a sentence. This is similar to the PageRank algorithm (Page et al., 1999) applied to dependency graphs. Given a graph $G = (V, E)$ defined by its adjacency matrix A, the relative score of a vertex $v \in V$ is:

$$x(v) = \frac{1}{\lambda} \sum_{u \in N(v)} x(u) \tag{2}$$

where by $N(v)$ we denote the set of neighbors of vertex v. The equation above can be rewritten as $Ax = \lambda x$ and solved for the largest real eigenvalue which gives the centrality measure (using the Perron –Frobenius theorem).

Table 1. Centrality scores for the question "How can the wind affect the speed of a car?"

Term	Degree centrality	Closeness centrality	Eigenvector centrality
How	0.0133	0.37	0.261
can	0.0133	0.37	0.261
the	0.0133	0.32	0.176
wind	0.0361	0.4	0.315
affect	**0.7265**	**0.55**	**0.632**
speed	0.0983	0.52	0.427
of	0.0361	0.41	0.222
a	0.0133	0.25	0.046
car	0.0361	0.32	0.111
?	0.0133	0.37	0.261

l. *Pointwise Mutual Information* – Given a term, x, for each unigram, bigram and trigram, y, in the corresponding question and all possible answers we compute the normalized pointwise mutual information (NPMI) as described by Church et al. (1990):

$$PMI(x, y) = \log \frac{p(x, y)}{p(x)p(y)} \tag{3}$$

$$NPMI(x, y) = \frac{PMI(x, y)}{-\log\left(p(x, y)\right)} = \frac{\log\left(p(x)p(y)\right)}{\log\left(p(x, y)\right)} - 1 \tag{4}$$

where $p(x, y)$ is the probability that *n-grams* x and y occur together (within some window) in a corpus. For computing the occurrence statistics, we used the ARC Corpus as well as about 65% of the March 2018 Wikipedia dump. A *NPMI score* close to 1 indicates that *n-grams* tend to appear together in similar contexts so they are correlated in some sense. Next we present a sample of computed NPMI scores, while Figure 4 highlights the correlation between NMI and computed essentialness scores:

- $NPMI\left(('helens'), ('mount',' st.')\right) = 0.903$
- $NPMI\left((('amphibians',), ('reptiles',)\right) = 0.733$
- $NPMI\left((('electromagnets',), ('electric',' motors',' generators')\right) = 0.6298$
- $NPMI\left(('car'), ('water')\right) = -0.049$

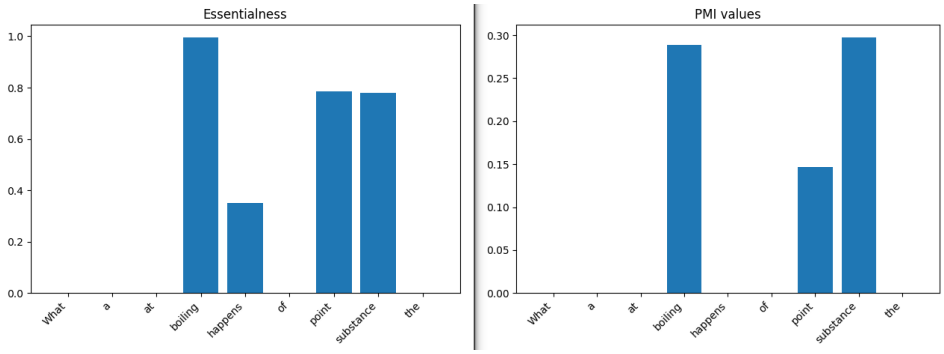

Figure 4. Correlation between term importance and NPMI values for the question "What happens at the boiling point of a substance?"

m. *Named Entity (NER)* – Each term in a question is labeled with a category (e.g. company, location, organization, products, or other). This may be useful for the neural network in order to learn complex hypothesis. It may be the case that entities representing companies or locations are important aspects of a question.

n. *Other Boolean features* such as term is in upper case, term is a currency, term is a number.

It is important to mention that all inputs whose value are a single real number (like concreteness rating) have been expanded into polynomial features. That is, we insert into the network, not only the input itself but also natural powers of the input (e.g. x^2, x^3) up to some power (which has been determined by fine tuning on the cross validation dataset). Again, this trick will eventually help the neural network to learn a good hypothesis given the small dataset used from training. *Figure 5* describes the entire architecture of the neural network that we have trained for prediction.

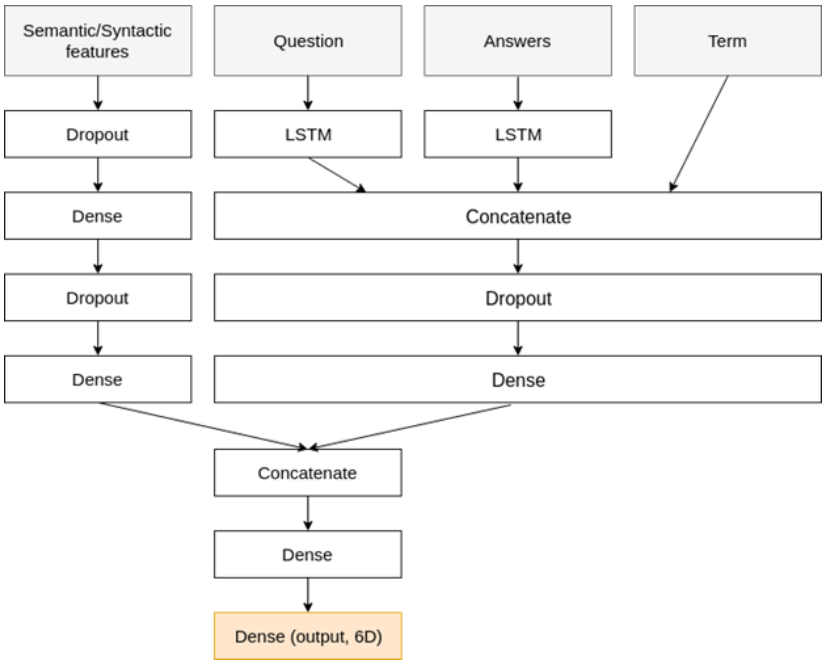

Figure 5. Neural network architecture for computing the essentialness score of a term

The entire dataset has been split into 80% - 10% - 10% batches for training, validation, and testing. The model hyper-parameters have been fine-tuned on the validation dataset and final results are reported (see Results section) on the test set. We varied (using grid search) LSTM hidden units in range 10 to 60, dense layer size from 10 to 100 and we added dropout (including recurrent dropout for the LSTM) in the range from 0.1 to 0.5. The best combination of values has been chosen with respect to the accuracy on the validation set. Dropout layers turned out to be very useful in this particular context because they reduce the network's tendency to memorize important terms. Also, dropout improves the network's capacity to generalize on new, unseen terms (like terms in the test set).

4. Evaluation and results

In order to evaluate the proposed essentialness prediction method, we have used the dataset collected by Khashabi et al. (2017). The dataset contains about 180 duplicate questions which have been completely removed so as to

be sure that the cross-validation and test sets are completely independent of the train set and there is no overlap between any two of them. This operation is very important in order to make sure that the obtained results are correct and meaningful. We report the performance of the model as a 6-way classification problem (each term is classified on a scale from 0 – irrelevant, to 5 – very important) and as a binary classification problem (essential versus not essential). The dataset is skewed, therefore we report *F1* scores when it comes to binary classification. The neural network previously described has 387,446 parameters, from which 97,496 are trainable and 289,950 non-trainable. The neural network has been trained for 400 epochs with a batch size of 4,000 using ADAM optimizer (Ba et al., 2014). The output of the neural network is a probability distribution over the importance of the term (scores from 0 to 5). The threshold above which a term is considered essential affects the accuracy of the model. We decided not to fix it at 0.5 but to choose it such that the *F1* score is maximized as shown in Figure 6. The performance for the binary classification task when the threshold is fixed 0.5 is presented in Table 2.

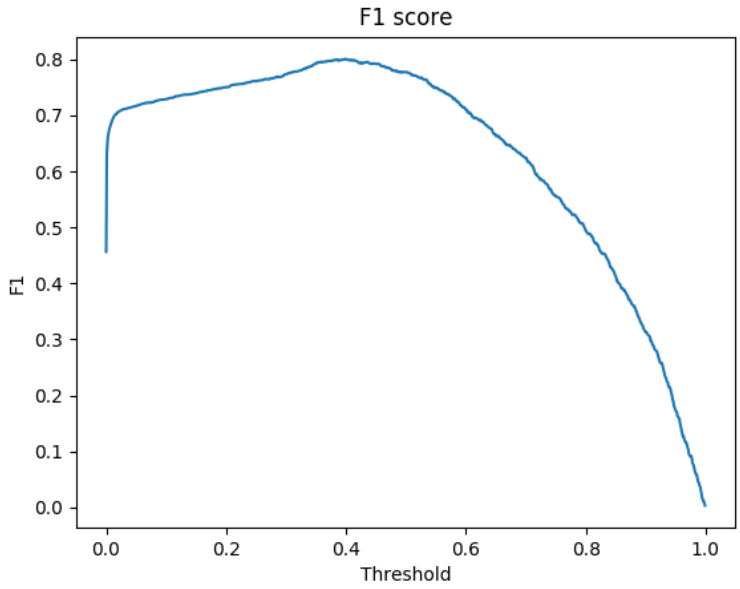

Figure 6. F1 score when varying the lower limit for a term to be essential. Maximum F1 score (80.1%) is obtained when the threshold is 0.4.

Our model is based on the observation that a simple neural network is not able to learn interesting features given the small dataset. The hypothesis is that injecting manually handcrafted features into the network (as inputs computed *a priori* to training) will increase the performance. In order to validate this hypothesis, we have identified how each feature affected the accuracy of the system. See Table 3 for a complete list of ablation tests, where the accuracy is measured for the 6-class classifier.

Table 2. Binary classification performance with the threshold fixed at 0.5

Metric	**Value**
Accuracy	86.57%
Precision	76.31%
Recall	79.17%
F1	0.78

The best model published so far, reporting results on the same dataset we used, achieves an F1 score of roughly 0.81 but uses a SVM classifier and 120 hand-crafted features (Khashabi et al., 2017). Our model performance is similar but the neural network itself is much easier to train (20 features versus about 120). Furthermore, the proposed model will likely benefit from a larger training dataset which might not be the case of the SVM.

Table 3. Feature contribution to model's performance for 6-way classification (validation dataset)

Features	**6-way accuracy**
Is science term only	59.42%
Concreteness rating only	54.05%
POS only	59.40%
All above	60.50%
Is stop term	58.40%
Centrality measures (all 3)	60.80%
All above	62.50%
PMI + all above	63.40%
DEP + NER + all above	64.30%
LSTM only	65.21%
LSTM + all above	**68.29%**

From all the features we described earlier, the following have also been used by Khashabi et al. (2017): is science term, POS tag, NER information, and PMI values. Apart from these, we have added the rest of the features (about 10 of them) at the sentence level using dependency graphs, rather than

the word level. Those features are important because it is often very hard to categorize a term as essential or not based only on the term itself – the surrounding context and the core meaning of the question are very important. Centrality measures are useful especially when combined with other features like: is stop term or POS tag. This is caused by the fact that some words have a large centrality value but they do not represent an important term (usually stop words also have large centrality values). The neural network should be able to combine input data giving rise to more complex hidden features (e.g. if the term has a high centrality measure and the part-of-speech is VERB then is important, if it is an abstract term used as a noun and it is the subject of the sentence then is very important). We should emphasize one more time that input features taken independently are not very useful (see Table 3), but combined can lead to a powerful classification model. By using only 20 features (including LSTM score) we achieved about the same *F1* score as other similar state-of-the-art approaches.

In order to test how this method can improve a simple information retrieval baseline, we used Lucene to index a large collection of documents representative for answering science questions (English Wikipedia dump, a set of 8th-degree science books from the Internet and the ARC Corpus). For each pair *(question, answer)* we search for documents containing all terms from both the question and each candidate answer. That is, we launch a Lucene query with the following structure: *(question AND answer)*. The documents are ranked based on the default Lucene *TF-IDF* score, without any term boosting (by default). The top-scoring document is chosen as the most relevant one and the answer with the biggest score is predicted as being correct.

We integrated essentialness information in this retrieval scheme using two different approaches:

a) When querying Lucene with a pair *(question, answer)* we kept the terms in the question with essentialness greater than or equal to a threshold. We used the entire Wikipedia dump as a set of documents as well as a set of science books collected from various places around the Web. We count how many questions can be answered correctly based on the Lucene score for each answer (the correct answer is the one with the greatest TF-IDF score). See Figures 7 and 8 for the accuracy of the QA system depending on the chosen

threshold. A threshold set to 0 means no term filtering using essentialness scores is used.

b) When querying Lucene with a pair *(question, answer)* we use term boosting with respect to the words in the question, according to the essentialness scores. Therefore, the most important terms are contributing more to the Lucene score (according to their essentialness), but no terms are completely removed from the question. See Tables 4 and 5 for results.

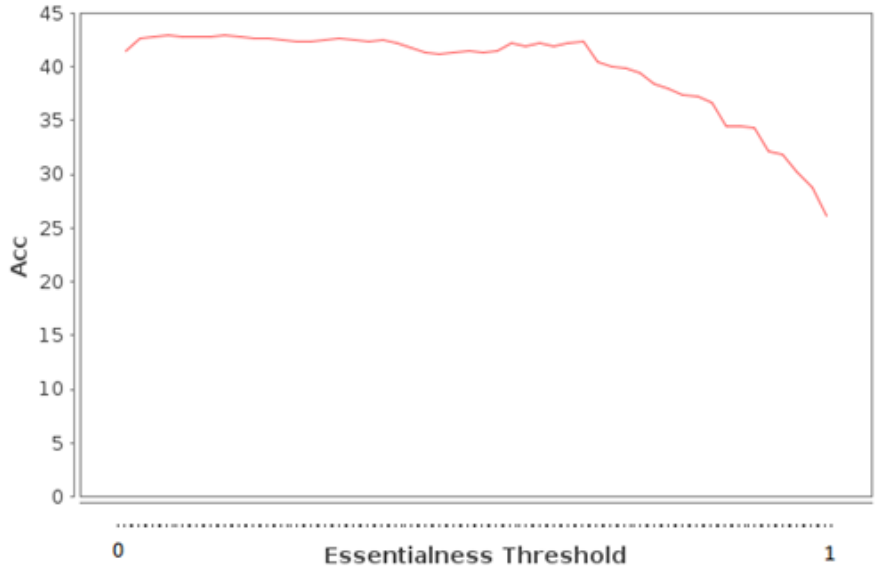

Figure 7. QA accuracy using Lucene backed by Wikipedia and essential terms. Max = 37.10% at threshold = 0.46. Gives +1.98% accuracy improvement (AllenAI dataset)

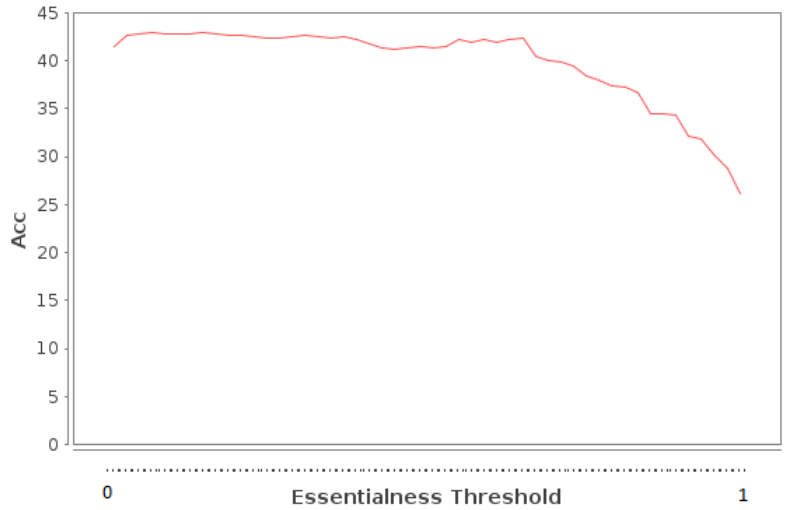

Figure 8. QA accuracy using Lucene backed by science books collection and essential terms. Max = 42.48% at threshold = 0.42. Gives +1.00% accuracy improvement (AllenAI dataset)

For measuring the QA system accuracy, we used the AllenAI QA dataset which has been proposed on Kaggle13 (from this dataset, we only use the publicly available part consisting of 2,500 science questions). The second dataset we have tested the model on is the ARC-Easy dataset (also collected by AllenAI). The ARC-Easy dataset contains easier questions which are more suitable for testing an information retrieval approach.

From Figures 7 and 8 we can observe that the maximum performance is reached when the essentialness threshold is set around 0.4 (in accordance with the value to which the *F1* score is maximized – as expected). When the threshold is set to 1 all terms are removed from the question and the accuracy is close to random.

Table 4. QA system accuracy on AllenAI dataset using term essentialness

Lucene backend	Without boosting	With boosting	Delta
English Wikipedia	37.6%	41.52%	***+3.92%***
Science books	41.56%	43.6%	***+2.04%***
ARC Corpus	41.00%	43.36%	***+2.36%***

[13] https://www.kaggle.com/c/the-allen-ai-science-challenge/data (last accessed November 2018)

Table 5. QA system accuracy on ARC-Easy test dataset using term essentialness

Lucene backend	Without boosting	With boosting	Delta
English Wikipedia	43.50%	47.18%	***+3.68%***
Science books	50.02%	53.91%	***+3.89%***
ARC Corpus	53.06%	56.99%	***+3.93%***

The boosting strategy gives better results than term filtering strategy mainly due to the following reasons: the TF-IDF score is more relevant when using boosting (is affected a lot by the boosting factors) and sometimes the model predicts wrong importance scores for some words and as a result they are removed completely from the question, although they are very relevant (therefore, the Lucene will give no importance at all for that words). Furthermore, the science book collection and the ARC Corpus appear to be a much better knowledge base choice mainly due to the fact that they present information related to science and, thus, are more focused to the specific QA datasets we have used. Wikipedia contains general information and a large fraction of it is not helpful in answering science-based questions.

5. Conclusions

In this article, we described a method for assigning importance scores to terms in questions. This can be useful in increasing the performance (accuracy) of IR-based question answering systems by up to 4%, as observed in our experiments, depending on the dataset used. Our proposed model archives close to state-of-the-art performance for computing the essentialness score for a term in the question and it is using (a lot) fewer features than other methods (Khashabi et al., 2017). Furthermore, we postulate that essentialness information can help other question answering models achieve better results. Thus, more recent multiple-choice question answering with candidate contexts using deep learning as proposed by Nicula et al.(2018) could also be potentially improved using our essentialness model, similalry to the IR models. Apart from this, the same idea can be easily extended and used to identify relevant terms in general texts (not only questions), which has a lot of practical applications.

References

Brysbaert, M., Warriner, A. B., & Kuperman, V. (2014). Concreteness ratings for 40 thousand generally known English word lemmas. Behavior research methods, 46(3), 904-911.

Choi, J. D., & Palmer, M. (2012). Guidelines for the Clear style constituent to dependency conversion. Technical Report 01–12.

Church, K. W., & Hanks, P. (1990). Word association norms, mutual information, and lexicography. Computational linguistics, 16(1), 22-29.

Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C., & Tafjord, O. (2018). Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge. arXiv preprint arXiv:1803.05457.

Clark, P., Etzioni, O., Khot, T., Sabharwal, A., Tafjord, O., Turney, P.D., Khashabi, D. (2016). Combining retrieval, statistics, and inference to answer elementary science questions. In: AAAI 2016, pp. 2580–2586.

De Marneffe, M. C., Dozat, T., Silveira, N., Haverinen, K., Ginter, F., Nivre, J., & Manning, C. D. (2014). Universal Stanford dependencies: A cross-linguistic typology. In LREC (Vol. 14, pp. 4585-4592).

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780. 3.

Jansen, P., Sharp, R., Surdeanu, M., Clark, P.: Framing qa as building and ranking intersentence answer justifications. Computational Linguistics (2017)

Khashabi, D., Khot, T., Sabharwal, A., Clark, P., Etzioni, O., & Roth, D. (2016). Question answering via integer programming over semi-structured knowledge (extended version). In Proc. 25th Int. Joint Conf. on Artificial Intelligence (IJCAI 2016).

Khashabi, D., Khot, T., Sabharwal, A., & Roth, D. (2017). Learning What is Essential in Questions. In Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017) (pp. 80-89).Edwards, J., Bagozzi, R. (2000) On the nature and direction of relationship between constructs and measures. *Psychological Methods 5(2)*, 155-174.

Khot, T., Balasubramanian, N., Gribkoff, E., Sabharwal, A., Clark, P., & Etzioni, O. (2015). Exploring Markov logic networks for question answering. In Proceedings of EMNLP 2015.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems (pp. 3111-3119).

Nicula, B., Ruseti, S., & Rebedea, T. (2018). Improving Deep Learning for Multiple Choice Question Answering with Candidate Contexts. In European Conference on Information Retrieval (pp. 678-683). Springer, Cham.

Page, L., Brin, S., Motwani, R., & Winograd, T. (1999). The PageRank citation ranking: Bringing order to the web. Stanford InfoLab.

Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP) (pp. 1532-1543).

Salton, G. (1971). The SMART retrieval system -experiments in automatic document processing. Englewood Cliffs.

.