A Flexible, Efficient and Accurate Framework for Community Question Answering Pipelines

Salvatore Romeo, Giovanni Da San Martino, Alberto Barrón-Cedeño,
Qatar Computing Research Institute, HBKU, Doha, Qatar
{sromeo,gmartino,albarron}@hbku.edu.qa

Alessandro Moschitti*
Amazon, Manhattan Beach, CA, USA
amosch@amazon.com

Abstract

Although deep neural networks have been proving to be excellent tools to deliver state-of-the-art results, when data is scarce and the tackled tasks involve complex semantic inference, deep linguistic processing and traditional structure-based approaches, such as tree kernel methods, are an alternative solution. Community Question Answering is a research area that benefits from deep linguistic analysis to improve the experience of the community of forum users. In this paper, we present a UIMA framework to distribute the computation of cQA tasks over computer clusters such that traditional systems can scale to large datasets and deliver fast processing.

1 Introduction

Web forums have been developed to help users to share their information. Given the natural use of questions and answers in the human communication process, traditional automated Question Answering (QA) techniques have been recently applied to improve the forum user experience.

Community Question Answering (cQA) deals with difficult tasks, including comment re-ranking and question re-ranking. The former task is defined as follows: given a thread of comments related to a user question, re-rank the comments in order of relevance with respect to the question. The latter task comes into play when a user wants to ask a new question. In this case an automatic system can be used to retrieve semantically similar questions, together with their threads of comments, already posted in the forum, and sort them according to their relevance against the freshly-posted question. Solving these tasks is beneficial both for the user, who avoids to manually look for such information, and the forum, since related information is not spread into multiple threads.

Previous cQA challenges, e.g., (Nakov et al., 2015, 2016, 2017) have shown that, to build accurate rankers, structural information and linguistic processing are required. Indeed, the results of the challenges have shown that (i) neural approaches are not enough to deliver the state of the art and (ii) kernel methods applied to syntactic structures often achieve top performance (Barrón-Cedeño et al., 2016; Filice et al., 2016).

Unfortunately, the models above are rather inefficient, as they require among others the syntactic parsing of long texts and kernel machine processing. The latter can be computationally expensive as the classification step requires quadratic time in the number of support vectors. Thus, approaches to speed up computation are very appealing. The classical method in these cases is to distribute and parallelize the computation. However, as the cQA processing pipelines can be very complex, an engineering approach to distributed computing is required.

In this paper, we propose a UIMA framework to manage the computation distribution of the complicated processing pipelines involved in cQA systems. In particular, we make the computation of standard linguistic processing components, feature/structure extractors and classification or learning phases scalable. This way, we both attain (i) the state-of-the-art accuracy of tree kernel-based rerankers and (ii) fast response. This makes our models useful for practical applications. We highlight the fact that our framework is rather flexible and extensible as new linguistic or machine learning components can be easily added. Indeed, we built two different cQA systems for English

*This work was carried out when the author was principal scientist at QCRI.

and Arabic (Barrón-Cedeño et al., 2016) by simply adding basic linguistic modules, e.g., the syntactic parsers, for both languages.

We make our software framework, based on UIMA technology, freely available to the research and industrial community by also providing our toolkit with tutorials and usage options for different degrees of user expertise.

2 Related Work

One of the first approaches to answer ranking relied on metadata (Jeon et al., 2006) (e.g., click counts). Agichtein et al. (2008) explored a graph-based model of contributors relationships together with both content- and usage-based features. Some of the most recent proposals aim at classifying whole threads of answers (Joty et al., 2015; Zhou et al., 2015) rather than each answer in isolation.

Regarding question ranking, Duan et al. (2008) searched for equivalent questions by considering the question's focus. Zhou et al. (2011) used a (monolingual) phrase-based translation model and Wang et al. (2009) computed similarities on syntactic-trees. A different approach using topic modeling for question retrieval was introduced by Ji et al. (2012) and Zhang et al. (2014). dos Santos et al. (2015) applied convolutional neural networks.

The three editions of the SemEval Task 3 on cQA (Nakov et al., 2015, 2016, 2017) have triggered a manifold of approaches. The challenges of 2016 and 2017 included Task 3-A on comment re-ranking and Task 3-B on question re-ranking. For task 3-A, Tran et al. (2015) applied machine translation, topic models, embeddings, and similarities. Hou et al. (2015) and Nicosia et al. (2015) applied supervised models with lexical, syntactic and meta-data features.

For task 3-B The top-three participants applied SVMs as learning models (Franco-Salvador et al., 2016; Barrón-Cedeño et al., 2016; Filice et al., 2016). Franco-Salvador et al. (2016) relied heavily on distributed representations and semantic information sources, such as Babelnet and Framenet. Both Barrón-Cedeño et al. (2016) and Filice et al. (2016) use lexical similarities and tree kernels on parse trees. No statistically-significant differences were observed in the performance of these three systems.

In summary, the results for both tasks show that SVM systems based on a combination of vectorial features and tree kernels perform consistently well on the different editions of the competition (Barrón-Cedeño et al., 2016; Filice et al., 2016, 2017): the systems described in those papers won Task 3-A both years, placed second and first on Task 3-B in years 2016 and 2017, respectively.

The most related demonstration papers to ours are (Uryupina et al., 2016; Rücklé and Gurevych, 2017). As ours, the system of Uryupina et al. (2016) is a UIMA-based pipeline. Yet in their case the input is a single text and the output is the result of different levels of textual annotation (e.g., tokens, syntactic information, or wikification). Rücklé and Gurevych (2017) developed an architecture to perform question and answer re-ranking in cQA based on deep learning. Their main focus is the analysis of attention models in these tasks.

3 Structural Linguistic Models for cQA

In this section, we describe the components of the two learning systems.

The ranking function for both tasks can be implemented by the scoring function of an SVM, $r : X \times X \to \mathbb{R}$, where X is either a set of comments (Task 3-A) or a set of questions (Task 3-B). For example, r can be a linear function, $r(x, x') = \vec{w} \cdot \phi(x, x')$, where $\vec{w}$ is the model and $\phi()$ provides a feature vector representation of the pair, (x, x'). The vectors ϕ used by Barrón-Cedeño et al. (2016); Filice et al. (2016) are a combination of tree kernel similarity functions and features derived from similarity measures between the two comments/questions constituting one learning example, as well as features extracting information from the forum threads the comments/questions belong to.

3.1 Tree Kernel

We use the kernel function defined by Filice et al. (2015):

$$\begin{aligned} K((x_1, x'_1), (x_2, x'_2)) = \mathrm{TK}\big(t_{x'_1}(x_1), t_{x'_2}(x_2)\big) \\ + \mathrm{TK}\big(t_{x_1}(x'_1), t_{x_2}(x'_2)\big) \end{aligned}$$

where TK is the Partial Tree Kernel by Moschitti (2006) and $t_y(x)$ is a function which enriches the tree x with information derived from its structural similarity with the tree y (see (Severyn and Moschitti, 2012; Filice et al., 2016) for details).

3.2 Feature Vectors

Various sets of features have been developed in (Barrón-Cedeño et al., 2016; Filice et al., 2016). A number of features which include similarities between the two texts constituting one learning example are computed. Such features include greedy string tiling, longest common subsequence, Jaccard coefficient, word containment, and cosine similarity, which can be computed on n-grams or bag-of-word representations. Another similarity can be obtained by comparing syntactic trees with the Partial Tree Kernel, i.e., $\text{TK}\big(t_{x_1}(x'_1), t_{x'_1}(x_1)\big)$. Note that, different from the model in Section 3.1, the Partial Tree Kernel here is applied to the members of the same pair and thus only produces one feature. In the case of question re-ranking, the SemEval datasets include information about the ranking of the question, as generated by the Google search engine. Such information is exploited in two ways: "as-is", by using directly the position, pos, as a feature of the question, or its inverse, pos^{-1}.

4 Distributed Framework using UIMA

4.1 UIMA introduction

The Unstructured Information Management Architecture (UIMA) is a software framework for creating, combining, and deploying an arbitrary number of language analytics. UIMA Asynchronous Scaleout (UIMA-AS) is a set of functionalities integrated in UIMA to enable scalability using a distributed environment. In UIMA, each document is contained in a Common Analysis Structure (CAS), annotated by processing units called *Analysis Engines*, or *Annotators*.

In UIMA-AS, there are basically three actors: (i) client applications, (ii) brokers, and (iii) UIMA pipelines. The latter is connected to the broker and listens for requests on a queue managed by the broker. The annotation steps are as follows: (a) the client sends the CAS to the broker; (b) the broker, in turn, sends it to the pipeline, which is listening to a specific queue; (c) the pipeline annotates the CAS and send it back to the broker; and finally, (d) the broker send the annotated CAS back to the client. Our pipeline is designed and developed in this framework.

4.2 Our Pipeline

There are three main modules: (i) feature extraction, (ii) learning, and (iii) classification. Each module is designed to be replicated in multiple instances to achieve scalability. Each of these modules is a pipeline deployed as UIMA-AS service that listens to a queue of processing requests (registered on the broker). Each module can interact with others by means of the broker if necessary. For instance, both learning and classification use the feature extraction to extract the features for the input instances.

The entire framework offers two levels of scalability. The first one deploys the same pipeline in different UIMA-AS services but listens to the same queue. In this case, the broker distributes the processing requests to the different CPUs. The second one replicates the pipeline a number of times internally to a UIMA-AS service. In this case, UIMA-AS internally handles the parallel processing of multiple requests.

In UIMA each annotator and pipeline is described and configured with XML descriptors. The descriptors of an annotator include information related to the implementation class, configuration parameters and binding of resources (if any). The descriptor of a pipeline includes information regarding the annotators it is composed of and their order. Furthermore, the deployment descriptors for the pipelines include information about the location of the broker and the queue's name, where the pipeline listen to the processing requests. This configuration process is fully automated in our pipeline and all the required descriptors are generated automatically. Finally, the pipeline can also be deployed on a single machine for either local or parallel computation.

Feature Extraction. Each CAS received by this module contains an instance of one of the aforementioned tasks, i.e., a question along with a set of comments or a set of other questions. The first step of the feature extraction is a sequence of standard preprocessing steps, e.g., segmentation, POS-tagging, lemmatization, syntactic parsing. The questions and comments of each instance of the specific task can be processed in parallel. The input CASes are hence split in a way that each of the output CASes contains either a single question or a single comment and it is asynchronously processed in the *Preprocessing sub-pipeline*. The preprocessed CASes are then aggregated back to form the input task instances. In its current status, our pipeline is meant to work with pairs. Therefore, the aggregated CASes are split in order to form question–comment (for comment re-ranking) or question–question (for question re-ranking) pairs. These are instances to be used in the learning or classification phase.

The output CASes are fed into the *Feature Computation sub-pipeline*. This sub-component computes a set of features for each CAS using the annotation previously created. It is composed of a sequence of *Vectorial Features Annotators* and each of them makes use of a *Feature Computer* (FC) to compute each feature. A FC is implemented as a resource that, given a CAS representing a pair, computes a feature based on the first, the second, or both members of the pair. This allows for sharing FC among different annotators. After the computation of all features is completed, it is possible to add additional information and/or representations of the CAS contents, e.g., syntactic parse trees. This is done by means of the *Decoration sub-pipeline* that is a sequence of *Decorators*. Finally, a *Representation Extractor* (RE) produces the features' representation that the learner and the classifier expect to process. The RE uses a *serializer*, which is a resource in charge of serializing the generic pair in the target representation format. The serializer is plugged into the RE at deployment time to allow the integration of any learning and classification component.

Learning. The learning module is composed of a single annotator that makes use of a *Learner*. A learner is a resource plugged at deployment time. This allows to plug any algorithm by wrapping it in a learner. Furthermore, a single instance can be shared among multiple instances of the learning module. Note that all representation instances are collected before the learning process can start. The resulting trained model is stored and the local file path is sent back to the client in an output CAS. At publication time, the pipeline implements SVM-based models, but it can be extended with others.

Classification. This module is composed by a single annotator that makes use of a resource plugged at deployment time as well. In this case, the resource is a *Classifier* that uses one of the trained models stored by the learning module. Again, implementing the classifier as a resource allows to plug any type of classification algorithm and to share it among multiple instances of the classification module. Every time the classification annotator receives a new instance, it computes the prediction, updates the input CAS adding the corresponding information and gives it as output.

5 Software Package

Our cQA pipeline is available for download [1] and is distributed under the terms of the Apache 2.0 License. By taking advantage of the Apache Maven project management tool, most dependencies are automatically handled. The only exception is the UIMA framework toolkit. Still, its installation is straightforward. The pipeline is able to process natural language texts and metadata information associated with them and offers three main functionalities:

Feature and **representation extraction** allows to compute features, such as the ones described in Section 3.2. Moreover, the pipeline allows to compute parse trees by using any third-party UIMA parser. Currently we integrated the DKPro (Eckart de Castilho and Gurevych, 2014) wrapper of the Stanford parser.

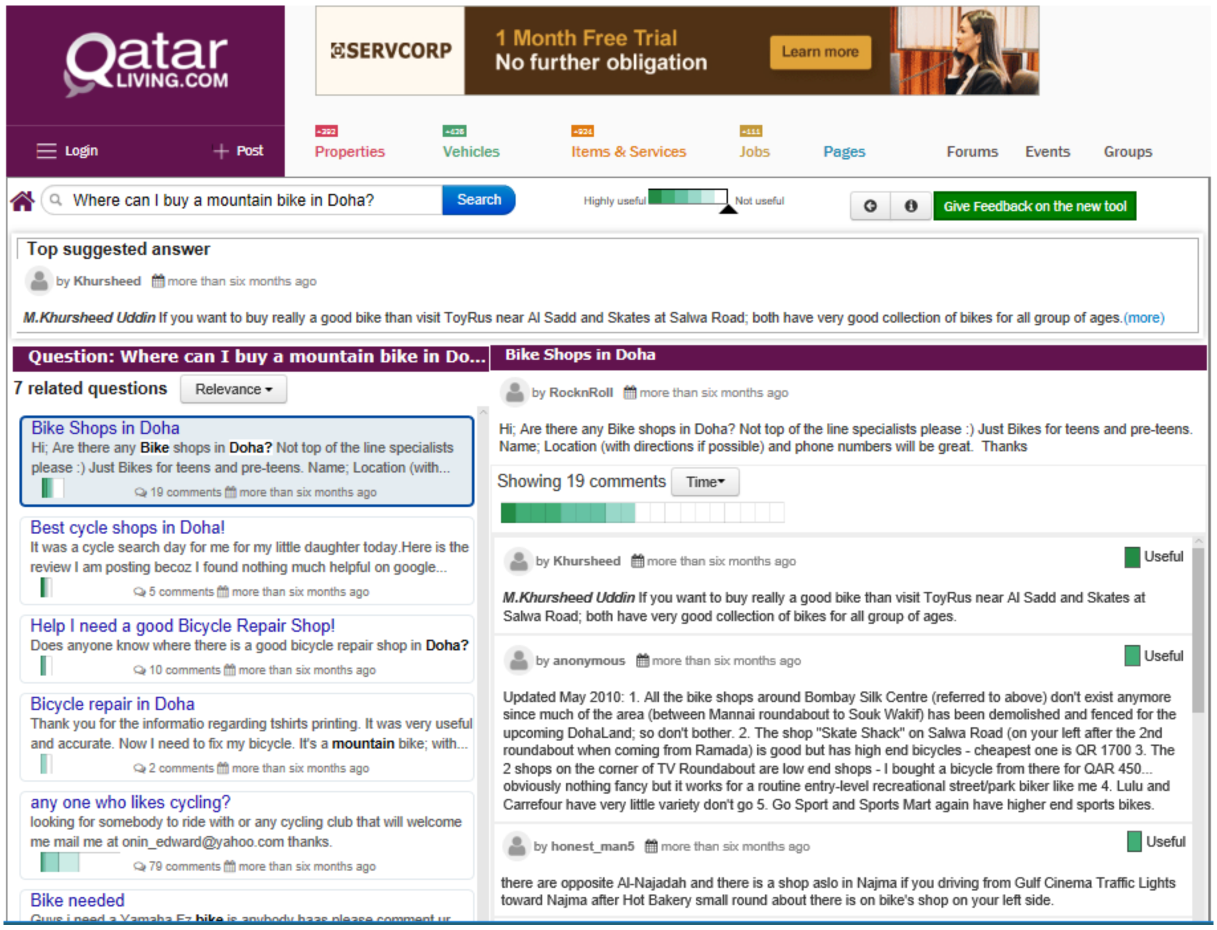
Learning and **classification** allow to apply a variety of learning algorithms on vectorial or structured data. Currently KeLP (Filice et al., 2018) [2] is integrated in the pipeline. KeLP allows to apply a growing number of kernel-based algorithms and kernel functions to perform unsupervised, online and batch supervised kernel methods. We opt for integrating KeLP because the kernel-based cQA systems relying on it perform at state-of-the-art level (see Section 2). Our pipeline is able to reproduce the state-of-the-art models for SemEval cQA tasks 3-A and 3-B.

Besides the functionalities just described, the pipeline has a modular structure. It allows to easily plug in new components, such as alternative natural language preprocessing components (lemmatizers, POS taggers, parsers), features, representations, and learning algorithms. The pipeline can either be run on a stand-alone machine or deployed on a cluster to distribute the computation load. In either case, simple classes are provided to run the pipeline from the command line.

Since this is an ongoing effort, we provide updated information on the wiki page of the GitHub project. The wiki provides instructions on the installation and tutorials to illustrate how to use the three main functionalities: (i) create representations, (ii) learn models, and (iii) classify data.

[1] `https://github.com/QAML/S3QACoreFramework`

[2] `http://www.kelp-ml.org`



Where can I buy a mountain bike in Doha?

Top suggested answer

by Khursheed more than six months ago

M.Khursheed Uddin If you want to buy really a good bike than visit ToyRus near Al Sadd and Skates at Salwa Road; both have very good collection of bikes for all group of ages.(more)

Question: Where can I buy a mountain bike in Do...

7 related questions

Bike Shops in Doha
Hi; Are there any **Bike** shops in **Doha?** Not top of the line specialists please :) Just Bikes for teens and pre-teens. Name; Location (with...
19 comments more than six months ago

Best cycle shops in Doha!
It was a cycle search day for me for my little daughter today.Here is the review I am posting becoz I found nothing much helpful on google...
5 comments more than six months ago

Help I need a good Bicycle Repair Shop!
Does anyone know where there is a good bicycle repair shop in **Doha?**
10 comments more than six months ago

Bicycle repair in Doha
Thank you for the informatio regarding tshirts printing. It was very useful and accurate. Now I need to fix my bicycle. It's a **mountain** bike; with...
2 comments more than six months ago

any one who likes cycling?
looking for somebody to ride with or any cycling club that will welcome me mail me at onin_edward@yahoo.com thanks.
79 comments more than six months ago

Bike needed
Guys i need a Yamaha Fz **bike** is anybody haas please comment ur

Bike Shops in Doha

by RocknRoll more than six months ago

Hi; Are there any Bike shops in Doha? Not top of the line specialists please :) Just Bikes for teens and pre-teens. Name; Location (with directions if possible) and phone numbers will be great. Thanks

Showing 19 comments

by Khursheed more than six months ago — Useful

M.Khursheed Uddin If you want to buy really a good bike than visit ToyRus near Al Sadd and Skates at Salwa Road; both have very good collection of bikes for all group of ages.

by anonymous more than six months ago — Useful

Updated May 2010: 1. All the bike shops around Bombay Silk Centre (referred to above) don't exist anymore since much of the area (between Mannai roundabout to Souk Wakif) has been demolished and fenced for the upcoming DohaLand; so don't bother. 2. The shop "Skate Shack" on Salwa Road (on your left after the 2nd roundabout when coming from Ramada) is good but has high end bicycles - cheapest one is QR 1700 3. The 2 shops on the corner of TV Roundabout are low end shops - I bought a bicycle from there for QAR 450... obviously nothing fancy but it works for a routine entry-level recreational street/park biker like me 4. Lulu and Carrefour have very little variety don't go 5. Go Sport and Sports Mart again have higher end sports bikes.

by honest_man5 more than six months ago — Useful

there are opposite Al-Najadah and there is a shop aslo in Najma if you driving from Gulf Cinema Traffic Lights toward Najma after Hot Bakery small round about there is on bike's shop on your left side.

Figure 1: A prototype of the UIMA pipeline applied to a real-world forum.

Our pipeline has been used in a number of prototypes. Qatarliving.com is a forum where expats in Qatar may ask questions on a variety of different topics and comment on them. We implemented the technology described in Section 3 both for question and comment re-ranking [3]. Fig. 1 shows an example of usage: the user asks the question "Where can I buy a bike in Doha?", the systems returns similar questions in the forum together with the best overall comment. By clicking on a question, the right panel shows the corresponding thread of comments with their relevance.

A second example is a cQA demo[4] in Arabic, which retrieves data from multiple medical forums from middle-east. In this case physicians answer to patients' questions: the left panel shows a question from a user and the right panel similar questions with the answers from the expert. In general there is only one (good) answer from the doctor, so this is mostly a question re-ranking task.

[3] `http://www.qatarliving.com/betasearch`

[4] `http://cqa.iyas.qcri.org/cQA-Arabic-Demo`

6 Conclusions

We presented a UIMA framework to distribute the computation of community question answering tasks. As a result, we can scale deep linguistic analysis and kernel technology to large datasets and deliver fast processing. Our toolkit is rather flexible and can be extended with new linguistic components as well as new machine learning components and algorithms. In addition to support state-of-the-art community question answering frameworks, an interesting consequence of the properties above is the fact that our framework also enables multilingual and potentially cross-language pipelines.

References

E. Agichtein, A. Gionis, C. Castillo, G. Mishne, and D. Donato. 2008. Finding High-quality Content in Social Media with an Application to Community-based Question Answering. In *Proceedings of WSDM*.

A. Barrón-Cedeño, G. Da San Martino, S. Joty, A. Moschitti, F. Al-Obaidli, S. Romeo, K. Tymoshenko, and A. Uva. 2016. ConvKN at SemEval-2016 Task 3: Answer and Question Selection for Question Answering on Arabic and English Fora. In *Proceedings of SemEval-2016*.

C. dos Santos, L. Barbosa, D. Bogdanova, and B. Zadrozny. 2015. Learning Hybrid Representations to Retrieve Semantically Equivalent Questions. In *Proceedings of ACL-IJCNLP*.

H. Duan, Y. Cao, C.-Y. Lin, and Y. Yu. 2008. Searching Questions by Identifying Question Topic and Question Focus. In *Proceedings of ACL*.

Richard Eckart de Castilho and Iryna Gurevych. 2014. A Broad-coverage Collection of Portable NLP Components for Building Shareable Analysis Pipelines. In *Proceedings of OIAF4HLT*.

S. Filice, D. Croce, A. Moschitti, and R. Basili. 2016. KeLP at SemEval-2016 Task 3: Learning Semantic Relations between Questions and Answers. In *Proceedings of SemEval-2016*.

S. Filice, G. Da San Martino, and A. Moschitti. 2015. Structural Representations for Learning Relations between Pairs of Texts. In *Proceedings of ACL-IJCNLP*.

S. Filice, G. Da San Martino, and A. Moschitti. 2017. KeLP at SemEval-2017 Task 3: Learning Pairwise Patterns in Community Question Answering. In *Proceedings of SemEval-2017*.

Simone Filice, Giuseppe Castellucci, Giovanni Da San Martino, Danilo Croce, Alessandro Moschitti, and Roberto Basili. 2018. KeLP: A Kernel-based Learning Platform. *To appear in the Journal of Machine Learning Research* .

M. Franco-Salvador, S Kar, T. Solorio, and P. Rosso. 2016. UH-PRHLT at SemEval-2016 Task 3: Combining Lexical and Semantic-based Features for Community Question Answering. In *Proceedings of SemEval-2016*.

Y. Hou, C. Tan, X. Wang, Y. Zhang, J. Xu, and Q. Chen. 2015. HITSZ-ICRC: Exploiting Classification Approach for Answer Selection in Community Question Answering. In *Proceedings of SemEval-2015*.

J. Jeon, W. B. Croft, J. H. Lee, and S. Park. 2006. A Framework to Predict the Quality of Answers with Non-textual Features. In *Proceedings of SIGIR*.

Z. Ji, F. Xu, B. Wang, and B. He. 2012. Question-answer Topic Model for Question Retrieval in Community Question Answering. In *Proceedings of CIKM*.

S. Joty, A. Barrón-Cedeño, G. Da San Martino, S. Filice, L. Màrquez, A. Moschitti, and P. Nakov. 2015. Global Thread-level Inference for Comment Classification in Community Question Answering. In *Proceedings of EMNLP*.

A. Moschitti. 2006. Efficient Convolution Kernels for Dependency and Constituent Syntactic Trees. In *Proceedings of ECML*.

P. Nakov, D. Hoogeveen, L. Màrquez, A. Moschitti, H. Mubarak, T. Baldwin, and K. Verspoor. 2017. SemEval-2017 Task 3: Community Question Answering. In *Proceedings of SemEval-2017*.

P. Nakov, L. Màrquez, W. Magdy, A. Moschitti, J. Glass, and B. Randeree. 2015. SemEval-2015 Task 3: Answer Selection in Community Question Answering. In *Proceedings of SemEval-2015*.

P. Nakov, L. Màrquez, A. Moschitti, W. Magdy, H. Mubarak, A. Freihat, J. Glass, and B. Randeree. 2016. SemEval-2016 Task 3: Community Question Answering. In *Proceedings of SemEval-2016*.

M. Nicosia, S. Filice, A. Barrón-Cedeño, I. Saleh, H. Mubarak, W. Gao, P. Nakov, G. Da San Martino, A. Moschitti, K. Darwish, L. Màrquez, S. Joty, and W. Magdy. 2015. QCRI: Answer Selection for Community Question Answering - Experiments for Arabic and English. In *Proc. of SemEval-2015*.

A. Rücklé and I. Gurevych. 2017. End-to-End Non-Factoid Question Answering with an Interactive Visualization of Neural Attention Weights. In *Proceedings of ACL (System Demonstrations)*.

Aliaksei Severyn and Alessandro Moschitti. 2012. Structural Relationships for Large-scale Learning of Answer Re-ranking. In *Proceedings of SIGIR*.

Q. H. Tran, V. Tran, T. Vu, M. Nguyen, and S. Bao Pham. 2015. JAIST: Combining Multiple Features for Answer Selection in Community Question Answering. In *Proceedings SemEval-2015*.

O. Uryupina, B. Plank, G. Barlacchi, F. J. Valverde-Albacete, M. Tsagkias, Antonio Uva, and Alessandro Moschitti. 2016. LiMoSINe Pipeline: Multilingual UIMA-based NLP Platform. In *Proceedings of ACL (System Demonstrations)*.

K. Wang, Z. Ming, and T. Chua. 2009. A Syntactic Tree Matching Approach to Finding Similar Questions in Community-based QA Services. In *Proceedings of SIGIR*.

K. Zhang, W. Wu, H. Wu, Z. Li, and M. Zhou. 2014. Question Retrieval with High Quality Answers in Community Question Answering. In *Proceedings of CIKM*.

G. Zhou, L. Cai, J. Zhao, and K. Liu. 2011. Phrase-based Translation Model for Question Retrieval in Community Question Answer Archives. In *Proceedings of ACL*.

G. Zhou, T. He, J. Zhao, and P. Hu. 2015. Learning Continuous Word Embedding with Metadata for Question Retrieval in Community Question Answering. In *Proceedings of ACL-IJCNLP*.