A 20% Jump in Duplicate Question Detection Accuracy ? Replicating IBM team's experiment and finding problems in its data preparation

João Silva, João Rodrigues, Vladislav Maraev, Chakaveh Saedi and António Branco
University of Lisbon
NLX-Natural Language and Speech Group, Department of Informatics
Departamento de Informática, Faculdade de Ciências, Universidade de Lisboa
Campo Grande, 1749-016 Lisboa, Portugal
{jsilva, joao.rodrigues, vlad.maraev, chakaveh.saedi, antonio.branco}@di.fc.ul.pt

Abstract

Validation of experimental results through their replication is central to the scientific progress, in particular in cases that may represent important breakthroughs with respect to the state of the art. In the present paper we report on the exercise we undertook to replicate the central result of the experiment reported in the Bogdanova et al. (2015) paper, *Detecting Semantically Equivalent Questions in Online User Forums*, which achieved results far surpassing the state-of-the-art for the task of duplicate question detection. In particular, we report on how our exercise allowed to find a flaw in the preparation of the data used in that paper that casts justified doubt on the validity of the breakthrough results reported there.

Keywords: replication, duplicate question detection, convolutional neural network

1. Introduction

This paper reports on the replication of the research results reported in *Detecting Semantically Equivalent Questions in Online User Forums* (Bogdanova et al., 2015), a paper published in the proceedings of the 19th Conference on Computational Natural Language Learning (CoNLL) and henceforth referred to as DSEQ.

The DSEQ paper caught our attention — and the attention of everyone doing research on Duplicate Question Detection, we guess — by reporting an accuracy of over 92% on the detection of semantically equivalent questions. This is an accuracy score that was more than 15 points above the results achieved in the related literature for that task (Nakov et al., 2016, Task B), representing a notable progress of 20% with respect to the state of the art.

This result placed DSEQ at the forefront of the research on Duplicate Question Detection (DQD). As such, while driven to advance our understanding of DQD and to improve its application, we considered the replication of DSEQ as being essential to our research on this topic.

The DQD task consists of classifying two input interrogatives sentences on whether they are a duplicate of each other, as in the following example:

(A) Can I install Ubuntu and Windows side by side?

(B) How do I dual boot Windows along side Ubuntu?

Two questions are semantically equivalent if they can be adequately answered by the same answer.

DQD belongs to the family of Semantic Text Similarity (STS) tasks, which assess the degree to which two textual segments are semantically similar. While the DQD task deals with the semantic equivalence of interrogative sentences, it is implicitly understood that the broader STS tasks concern only declarative sentences.

STS and DQD are language processing tasks of the utmost importance, both having been addressed in SemEval competitive shared tasks: STS since 2012 (Agirre et al., 2012) and DQD since 2016 (Nakov et al., 2016). Both tasks have been useful to support conversational interfaces and chatbots, in general, and online question & answering (Q&A) community forums, in particular.

One of the challenges faced by Q&A communities (online user forums) is that different users at different times frequently post duplicate questions that have already been answered before, rendering the forum potentially inefficient and demanding time-consuming human moderation to flag such duplicates. Adopting an automated system that can detect duplicate questions provides a computational technique to mitigate or solve this issue.

Section 2 provides an overview of the original DSEQ paper. The replication effort, described in Section 3, turned out to be a challenging task, requiring full attention to details in the implementation of the classifiers to work out unreported assumptions. We conclude that the replication can be successfully achieved but only if the segments in the pairs being classified already contain information as to their status as reciprocal duplicates, thus affecting the validity of the results reported in DSEQ.

Additionally, we report on experiments we undertook after we cleaned the data from these indications of the status of the segments in the pairs, which obtained results in the range of the state of the art results reported in the literature for other approaches and methods.

We address the resulting implications along with other considerations in Section 4.

2. The DSEQ paper

This section provides an overview of DSEQ. For the full details, we direct the reader to the original paper.

The DSEQ paper "aims to detect semantically equivalent questions in online user forums". The authors follow the usual definition used in the field of DQD according to which two questions are considered to be semantically equivalent if they can be adequately answered by the same answer.

DQD systems usually resort to supervised machine learning methods, which require labeled data to train a model. For DQD, these data consist of several pairs of text segments with each pair (i.e. the two questions) being labeled as either containing duplicate or non-duplicate segments.

DSEQ implemented different supervised machine learning systems as classifiers in a DQD task by resorting to pairs of questions annotated as duplicates or not duplicates. The aim of machine learning algorithms is to generalize over the training data, in our case, a semantic generalization that is aimed to classify testing data — unseen pairs of questions — and correctly assesses if they are semantically equivalent, i.e. duplicates.

This is a challenging task given that all questions can be rephrased in multiple ways. However, the recent boost in the amount of available data and computational power supported the application of machine learning techniques, including neural network models. This motivated DSEQ to compare standard machine learning methods and a convolutional neural network on a DQD task.

In the next subsections, we briefly describe the data used for the training of the machine learning models, the machine learning methods resorted to, the experiments performed, and the results reported.

2.1. Data sets used

Stack Exchange[1] is one of the largest Q&A online communities, with over 100 million monthly unique visitors. Like in all Q&A online communities, users can ask questions, get answers from other members of the community, and use a Q&A search engine to find existing questions. Stack Exchange is organized in such a way that each question consists of a *title* (usually a short, one-sentence formulation of the question) and a *body* that provides further details; these are followed by a *thread* of possible answers, ranked by the community.

Stack Exchange allows its users to tag posted questions as duplicates of previously posted questions. These tagged questions are later manually verified by moderators and definitely labeled as duplicates or not. If a question is marked as a duplicate of an already existing one, the moderators may choose to keep it as a duplicate and link it to that pre-existing question. In this way, duplicate questions (i.e. the different ways of asking the same question) end up linked to one and only one canonical formulation for that question.

DSEQ used the data from two Stack Exchange subcommunities: Ask Ubuntu[2], for users and developers of the Ubuntu operating system, and META Stack Exchange[3], for meta-discussion on issues regarding the Stack Exchange network itself.

[1] https://stackexchange.com/

[2] https://askubuntu.com/

[3] https://meta.stackexchange.com/

The Stack Exchange network provides the user-contributed content from all its Q&A sub-communities by means of publicly available periodic data dumps.[4] The data dumps include the questions (title and body), the answer thread, and meta data regarding each question, in particular its status as a duplicate. Figure 1 shows an example of a duplicate entry.

DSEQ used the Ask Ubuntu data dump from May 2014 and the META Stack exchange dump from September 2014. The instances were randomly selected and class-balanced, resulting in a training/testing sets of 24k/6k pairs for Ask Ubuntu and 20k/4k for META. The validation set is 1k pairs for both. Table 1 summarizes this information.

Data set	Training	Testing	Validation
Ask Ubuntu	24k	6k	1k
META	20k	4k	1k

Table 1: Data sets used in DSEQ for the training, testing and hyper-parameterization optimization (validation set) of the machine learning models. Each instance consists of a pair of questions with a corresponding label (duplicate or non-duplicate).

Regarding data preprocessing, the authors specify that NLTK (Bird et al., 2009) was used for tokenization and that all links were replaced with a unique string.

Each question is taken as a whole, that is, as the concatenation of the title and body parts.

2.2. Methods used for DQD

DSEQ compares (i) a rule-based and traditional similarity measure based on word overlap with shingling (n-grams) and a Jaccard coefficient; (ii) a standard machine learning method, namely Support Vector Machine (SVM); and (iii) a neural network architecture with convolutional layers (CNN).

The **Jaccard coefficient** is computed as a rule-based system. First, a set of n-grams (with n ranging from 1–4) is created from the training data. Second, a Jaccard coefficient for the pairs of questions is computed as

$$J(d_1, d_2) = \frac{S(d_1) \bigcap S(d_2)}{S(d_1) \bigcup S(d_2)},$$

where $S(d_1)$ is the set of n-grams extracted from the first segment (d_1) and $S(d_2)$ the set of n-grams extracted from the second segment (d_2). Segments d_1 and d_2 are deemed to be duplicate if the Jaccard coefficient is above a threshold that is empirically determined by measuring the coefficient of all pairs of questions in the training set.

The **SVM** is a machine learning algorithm that finds a hyperplane that optimizes the division of a data set into two classes. In an SVM, the data set instances are transformed into feature vectors, which are data points in a shared space. Then, a hyperplane is iteratively computed aiming at the best separation of the vectors regarding their classes.

[4] https://archive.org/details/stackexchange

```
1  <row
2      Id="1208"
3      PostTypeId="1"
4      CreationDate="2009-06-30T16:15:07.673"
5      Score="5"
6      ViewCount="152"
7      Body="*"
8      OwnerUserId="130090"
9      LastEditorUserId="-1"
10     LastEditDate="2017-03-20T10:30:01.953"
11     LastActivityDate="2009-07-31T07:48:12.030"
12     Title="Improper pagination of user search"
13     AnswerCount="1"
14     CommentCount="2"
15     ClosedDate="2009-07-19T23:42:02.707"
16 />
17
18 <body>
19     <blockquote>
20     <p>
21         <strong>Possible Duplicate:</strong>
22         <br>
23         <a href='https://meta.stackexchange.com/questions/469/
           page-navigation-on-users-page-doesnt-work-if-using-the-input-box'>
           Page navigation on Users page doesn't work if using the input box
           </a>
24     </p>
25     </blockquote>
26     <p>When searching for a user without knowing the exact name, it shows
       paginated results of all matches. However if you try and navigate
       through these results they are automatically defaulted back to the
       reputation based sort. It's impossible to see any more than 35
       matches for any search.</p>
27     <p>Suggestion:</p>
28     <ol>
29         <li>Either remove the paginated results for user searches </li>
30         <li>Make them work :)</li>
31     </ol>
32 </body>
```

Figure 1: A question data and meta data from the META Stack Exchange dump. For the sake of readability, the HTML entities were normalized and the content of the **Body** attribute (line 7) is shown separately under its own tag `<body>` (lines 18–32).

Resorting to the set of existing n-grams (n ranging from 1–4), for each pair of questions, DSEQ uses a vector with the following features:

1. The one-hot encoding of the n-grams in the first question; that is, for each n-gram, a boolean value indicating its occurrence in the first question.
2. The one-hot encoding of the n-grams in the second question.
3. The overall normalized count of each of the n-grams in both questions.

A radial basis function kernel is used to measure the similarity between feature vectors. The DSEQ's authors mention that a grid search was used to optimize the values of the hyper-parameters C and γ, and a frequency threshold was applied to reduce the features dimension.

A **combination of Jaccard coefficient and SVM** machine learning algorithm was also used. To this purpose, the SVM feature vectors were created as previously described and extended to include an extra field: for each pair, the Jaccard coefficient for that pair was considered in the corresponding feature vector.

A **Convolutional Neural Network** (CNN) is one of many neural network architectures that map an input to an output (class) resorting to layers of connected neurons. Typically, in a neural network each neuron receives input values that are used to output a computed value according to an activation function, such as a binary step or a hyperbolic tangent function. The input values of neurons are usually connections from other neurons. Each connection has an intrinsic weight, used to increase or decrease the values sent through them across neurons, strengthening or weakening the signal.

In the CNN used in DSEQ, each layer is connected consecutively (feedforward), with the output of each layer being sent to the next layer. The neural network receives each of the questions in the pair, with both inputs sharing the same neural network layers, in an architecture called Siamese neural network.

- In a word representation layer, each word of the sentence (question) is transformed into a vectorial representation, also known as a word embedding or distributional semantic vector.
- A convolution layer then computes a new vectorial representation by applying a dimension reduction technique to a matrix populated by all the word vectors from the previous layer. The computation can be observed as a compositional compression, encoding the semantic knowledge of the sentence.
- A final layer compares the representation obtained from both questions, using a cosine similarity function. This value is passed on to an activation function that determines if it is, or not, a duplicate pair.

The neural network learns with the training set to generalize the DQD task by iteratively changing the weights of the neural connections while aiming to output the correct class for each training instance.

2.3. Experiments

Four types of experiments are reported in DSEQ: (i) a comparison of the different DQD methods described above; (ii) an assessment of the impact of using domain-specific distributional semantic vectors; (iii) an assessment of the impact of varying the size of the training set; and (iv) an assessment of performing domain adaptation.

The **comparison of the DQD methods** evaluated each method with different parameterizations over the Ask Ubuntu data set. Two experiments were run, the first with a 4k training set and the second with the full 24k training set. The question title and question body were used as inputs in three different ways, namely (i) using the whole title and body;[5] (ii) removing programming language code snippets; and (iii) prefixing programming language code snippets with a special tag. In all cases, the 1k validation set was used to tune the hyper-parameters of the algorithms.

The **assessment of the impact of using domain-specific distributional semantic vectors** was twofold. It (i) evaluated the accuracy of the CNN using already trained distributional semantic vectors with different dimensions (50, 100, 200 and 400); and (ii) evaluated different distributional semantic vector space trained using Wikipedia data as general domain data, and the Ask Ubuntu data as in-domain data.

The **impact of varying the training set size** was assessed by profiling the different systems using different dataset sizes, from only 100 question pairs to the full 24k question pairs.

The **domain adaptation** experiment interchanged the CNN training data. Different corpora were used for training the machine learning algorithm and the distributional semantic vectors. The evaluation was performed with the META test set.

[5]Taking into account the already mentioned NLTK tokenization and links normalization.

2.4. Results

In the first experiment, when comparing the different DQD systems, the combination of SVM with Jaccard performed better than either of its parts individually. The hybrid system obtained a 77.4% accuracy, with the normalized input (removing data related to programming code), $C = 32.0$ and $\gamma \approx 3.05 \times 10^{-5}$. The CNN obtained the best result, 92.4% accuracy, with the normalized input, a 200 vector dimension, $k = 3$, $cl_u = 300$ and $\gamma = 0.005$.
In the second experiment, the study of the impact of domain-specific distributional semantic vectors, by increasing the vectors dimension, the CNN's accuracy improved. Regarding the use of general domain trained distributional semantic vectors from Wikipedia data against the in-domain Ask Ubuntu ones, the in-domain vectors supported a better accuracy: 85.5% accuracy was obtained with the former and 92.4% with the latter.
The third experiment showed that enlarging the training data improved the accuracy of all the systems.
In the fourth experiment, with the META data set, CNN obtained the best score, 92.68% accuracy, when using the META training data in both the training of the CNN and in the training of the distributional semantic vectors.
In Table 2 the best scores obtained in DSEQ with different data sets are reported.

Data set	Accuracy
Ask Ubuntu	92.90%
META	92.68%

Table 2: The best scores reported in DSEQ for the Ask Ubuntu and the META data sets using CNN.

3. Replication of DSEQ

The present Section describes our replication of the experiments reported in DSEQ paper as providing the best results, just indicated in Table 2.
Neither the data sets nor the software with the implementation of the DQD systems used in DSEQ were made publicly available. We attempted to obtain these data sets and more details about the hyper-parameters of the CNN but our emails received no answer.
When **acquiring the data** for the replication exercise, we realized that the Stack Exchange data dumps are frequently updated, with older data dumps being deleted. Thus, at the time of our exercise, we only had access to data dumps from September 2014, given that data dumps from May 2014 had already been removed from the respective distribution page. Table 3 shows the differences between the dumps used in the present work and in DSEQ.
Our **preparation of the data** — for the training of the CNN and the distributional semantic vectors — comprised the following procedures:

- Image removal.
- URL removal.
- Code snippet removal (i.e. <code> blocks).

Data set	Dump date	
	Replication	DSEQ
Ask Ubuntu	Sep. 2014	May 2014
Meta	Sep. 2014	May/Sep. 2014

Table 3: Dump dates for the data sets used in present replication and in DSEQ.

- Text tokenization, using the Stanford Tokenizer (Manning et al., 2014).
- Lowercasing of all tokens.

For the **training of the distributional vectors** we used the DeepLearning4j toolkit[6] with the built-in skip-gram algorithm.
The vectors were trained with a dimension of 200. The values for all the other parameters, which are not described in the DSEQ paper, were taken from the word2vec vanilla parameters.
Table 4 presents the data sets used for the distributional semantic vectors training.

Data set	Vector size	Types	Tokens
Ask Ubuntu	200	68k	38M
META	200	30k	19M

Table 4: Data sets used in replication to train the distributional vectors and respective sizes.

The **data sets acquired were organized** to approximate the organization of DSEQ data sets by using the same sizes for the training, testing and validation subsets. See Table 5 for a detailed rendering.
The **implementation of the CNN** was done using the Keras Python library (Chollet and others, 2015) with Theano (Team, 2016) as the back-end.
For the CNN hyper-parameters, we used the same values as in the DSEQ, when they were reported. The remaining hyper-parameters, namely batch size and number of epochs, were empirically determined by experimentation.
Table 6 shows the values for the main hyper-parameters used in the CNN replication.
The **evaluation** of the CNN over the Ask Ubuntu data set achieved a 94.1% accuracy, and 94.2% accuracy over the META data set.
The replicated models show a performance that is very similar, or even slightly better, to the one reported in DSEQ. Table 7 collects the relevant scores.

3.1. The problematic clue strings

When preparing the data sets for the replication exercise, we realized that removing the URLs from the data dumps as described in DSEQ was not enough to produce unbiased data sets.
We noticed that duplicate questions contain information that provides explicit *clues* as to their status as a duplicate. In particular, at the start of the `body` content, duplicate questions contain the string `Possible Duplicate:` followed by a link to the canonical question of which the question at stake is a duplicate. This is illustrated in Figure 1, in lines 21–23.
Note that these strings, with this explicit indication of the solution of the DQD task, cannot be left in the data since they provide direct clues for the answer the system should optimally deliver — i.e. whether the questions are duplicate or not.
The replication results we reported above in Table 7 were obtained when such clues were kept in the data.
It is not indicated in the DSEQ paper if these clue strings were kept in or removed from the data sets in the experiment reported therein. But further experiments we undertook provide a strong indication that they were not.
We repeated the same experiments by changing only the way the data sets were prepared, in particular by removing such clue strings from them. The scores obtained in this second round of replication — with data sets cleaned from these clue strings — are in line with the state of the art that existed before the DSEQ paper.
Table 8 presents the comparison of our two replication rounds against DSEQ. When removing the clue strings from all the data sets the accuracy drops in all the experiments; when keeping them, all scores are very close to the ones reported in DSEQ.
This very likely indicates that in the experiments reported in DSEQ the clue strings in to duplicate questions were not removed from the data sets used in its experiments.
The data and models used in the replication exercise reported here are available at this GitHub page.[7]
Due to a couple of implementation details that were left unreported in DSEQ — and we had to figure out by ourselves for the replication exercise (cf. Table 6) — and due to slight differences in the data set dump dates (cf. Table 3), our replication settings are not fully identical to the ones of DSEQ. However, given the results obtained in the different rounds of replication and how they are closely aligned with results from the DSEQ (in the first round) and from the literature (in the second round), these differences are not enough to prevent the main conclusions coming out of the present replication exercise.

4. Conclusions

In the present paper, we describe the exercise we undertook of replicating the experiment described in (Bogdanova et al., 2015), which was reported to outperform by 20% the state of the art in the task of Duplicate Question Detection that was contemporary to the publication of that work.
As in the literature on Duplicate Question Detection the progress reported in different papers typically represent a much smaller delta of progress, this result appeared as an outstanding breakthrough in this area, to which, moreover, none of the subsequent advances reported in the literature

[6] http://deeplearning4j.org/word2vec

[7] `https://github.com/nlx-group/Replication-of-IBM-Team-s-Duplicate-Question-Detection-Experiment`

Data set	Total pairs	Duplicates	Training	Testing	Validation
Ask Ubuntu	167,765	17,115	24k	6k	1k
META	67,746	19,456	20k	4k	1k

Table 5: Splits and sizes of the data sets used in replication to train the CNN.

Parameter	Value	Description
d	200	Size of word representation
k	3	Size of k-gram
cl_u	300	Size of convolutional filter
γ	0.005	Learning rate
batch size	1	Examples per gradient update
epochs	20	Number of Training epochs

Table 6: CNN training hyper-parameters. Only the first four parameters were explicitly provided in DSEQ.

	Data set	Accuracy
DSEQ	Ask Ubuntu	92.90%
	META	92.68%
Replication	Ask Ubuntu	94.10%
	META	94.20%

Table 7: Performance results of DSEQ and of the present replication exercise, using the CNN model for both data sets.

had come close.[8] That was the major motivation for our replication exercise.

The replication exercise reported here permitted to find out that the best scores described in (Bogdanova et al., 2015) can be replicated only when the data sets are not properly prepared. In particular, they can be replicated only when clue strings in the data — with the explicit indication that questions are duplicates — are not removed.

Our replication exercise permitted also to find out that when the data sets are cleaned from these clues, as they should, the accuracy of those very same models drops sharply to scores in line with the state of the art scores reported in the literature contemporary to that paper.

This casts justified doubts on the validity of the breakthrough result reported, indicating a jump of 20% with respect to the state of the art, that does not hold.

The current study also highlights the importance of replication as a first class citizen in research on language technology. If this replication exercise reported here had not be undertaken, the community would have remained with an incorrect believe about what would be the state-of-the-art for the task of Duplicate Question Detection.

[8] Among several others, see the results of SemEval2017, Task 3, Subtask B, reported in (Nakov et al., 2016), and the recent advances obtained by our team, reported in (Rodrigues et al., 2018), (Rodrigues et al., 2017), (Saedi et al., 2017) and (Maraev et al., 2017)

	Ask Ubuntu			META	
Clues	**4k**	**full val.**	**test**	**val.**	**test**
Removed	71.8	73.8	73.3	57.3	55.7
Kept	91.8	92.3	94.1	96.1	94.2
DSEQ	92.4	93.4	92.9	92.8	92.7

Table 8: Accuracy (%) of CNN models over Ask Ubuntu and on META data sets, with clue strings kept and with clue strings removed in replication, compared to DSEQ.

5. Acknowledgments

The present research was partly supported by the Infrastructure for the Science and Technology of the Portuguese Language (CLARIN Língua Portuguesa), by the National Infrastructure for Distributed Computing (INCD) of Portugal, and by the ANI/3279/2016 grant.

6. References

Agirre, E., Gonzalez-Agirre, A., Cer, D., and Diab, M. (2012). Semeval-2012 task 6: A pilot on semantic textual similarity. In *Proceedings of the 1st Joint Conference on Lexical and Computational Semantics (*SEM2012)*, pages 385–393.

Bird, S., Klein, E., and Loper, E. (2009). *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit*. O'Reilly Media, Inc.

Bogdanova, D., dos Santos, C. N., Barbosa, L., and Zadrozny, B. (2015). Detecting semantically equivalent questions in online user forums. In *Proceedings of the 19th Conference on Computational Natural Language Learning (CoNLL2015)*, pages 123–131. http://aclweb.org/anthology/K/K15/K15-1013.pdf.

Chollet, F. et al. (2015). Keras. GitHub.

Manning, C., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S., and McClosky, D. (2014). The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics (ACL2014)*, pages 55–60.

Maraev, V., Saedi, C., Rodrigues, J., Branco, A., and Silva, J. (2017). Character-level convolutional neural network for paraphrase detection and other experiments. In *Proceedings, 6th Artificial Intelligence and Natural Language Conference*.

Nakov, P., Marquez, L., Moschitti, A., Magdy, W., Mubarak, H., Freihat, A. A., Glass, J., and Randeree, B. (2016). Semeval-2016 task 3: Community question answering. In *Proceedings of the 11th International Conference on Semantic Evaluation (SemEval2016)*, pages 27–48.

Rodrigues, J., Saedi, C., Maraev, V., Silva, J., and Branco, A. (2017). Ways of asking and replying in duplicate question detection. In *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (*SEM2017)*, pages 261–270.

Rodrigues, J., Saedi, C., Branco, A., and Silva, J. (2018). Semantic equivalence detection: Are interrogatives harder than declaratives? In *Proceedings of the 11th International Conference on Language Resources and Evaluation (LREC2018)*. Accepted, to appear.

Saedi, C., Rodrigues, J., Silva, J., Branco, A., and Maraev, V. (2017). Learning profiles in duplicate question detection. In *Proceedings of the IEEE 18th International Conference on Information Reuse and Integration (IEEE-IRI2017)*.

Team, T. D. (2016). Theano: A python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688.