2017 IEEE International Conference on Information Reuse and Integration

Learning Profiles in Duplicate Question Detection

Chakaveh Saedi, João Rodrigues, João Silva, António Branco, Vladislav Maraev
University of Lisbon
Dept. Informática, Univ. Lisboa, Campo Grande, 1749-016 Lisboa, Portugal
{chakaveh.saedi,joao.rodrigues,jsilva,antonio.branco,vlad.maraev}@di.fc.ul.pt

***Abstract*—This paper presents the results of systematic and comparative experimentation with major types of methodologies for automatic duplicate question detection when these are applied to datasets of progressively larger sizes, thus allowing to study the learning profiles of this task under these different approaches and evaluate their merits. This study was made possible by resorting to the recent release for research purposes, by the Quora online question answering engine, of a new dataset with over 400,000 pairs labeled with respect to their elements being duplicate interrogative segments.**

***Keywords*-Natural language processing; semantic similarity; duplicate question detection; machine learning; neural networks; deep learning.**

I. Introduction

Duplicate question detection (DQD) is a natural language processing (NLP) task that has recently become the focus of active research, where two interrogative segments are considered to be semantically equivalent, and thus duplicate, if they can receive the same answer. In line with this surge of interest, the SemEval challenge, devoted to semantic similarity,[1] was the first to include a task specifically on question-question similarity in its last edition of 2016 [1].

Much of the motivation for this research topic is coming from the usefulness of resorting to DQD to support online question answering community forums, and also conversational interfaces, in general.

For instance, when applied to the former, DQD can be used to automatically detect whether a new question entered by a user to a given forum has been asked before in that forum, and to help mark and eventually remove it as a duplicate question, thus mitigating the proliferation of duplicate questions that is a major issue hindering the usability of such forums.

And when embedded in the conversational interfaces, DQD can be used to compare a newly entered question against a database of previous question-answer pairs and if a similar question is found, to reply by delivering the corresponding stored answer, thus possibly avoiding to resort to a human operator.

DQD falls under the broader task of semantic text similarity (STS), which has been the topic of the SemEval challenges since 2012 [2]. These challenges address a variety of STS subtasks, such as plagiarism detection, comparing machine translation output with a post-edited version, paraphrase detection, among several others. What these STS subtasks have in common is that, when given two input segments, the systems must rate their semantic similarity in some scale, ranging from total semantic equivalence to complete semantic dissimilarity.

Paraphrase detection is one such STS subtask, consisting in a binary decision on whether the two input segments are paraphrases of each other. Being an STS subtask, DQD is thus a specific case of paraphrase detection, where the input is restricted to interrogative segments.

From previous research on DQD, a number of lessons have been learned, some of them being pretty much in line with lessons learnt about data-driven approaches in general, while some others being more specific for this task.

Concerning the training data, these lessons seem to indicate that:

- one obtains better performing DQD systems with smaller training datasets on narrow domains than with larger training datasets on a more generic domain [3];
- the gap in terms of performance scores between the best and the worst performing approaches is narrowed down when one moves from specific domains to generic domains [4];
- the accuracy of the systems degrades and comes close to random decision performance when systems are trained on as much data as possible from all sources and different domains and eventually applied over a narrow domain [4];
- for the range of average lengths of interrogative segments, the difference in size of the segments in the training data seem to have little impact on the performance of the systems and including additional training content related to the interrogative segments showed no significant impact on the accuracy of the systems [4];
- the difference in the grammaticality of the segments — that is, segments with sloppy wording and non-standard expressions of suboptimal grammaticality (e.g. from online forums) vs. fully grammatical segments — seem to have little impact on the performance of the systems [4].

Concerning the different methodologies, results from experiments with three major approaches (based on heuristics,

[1] http://alt.qcri.org/semeval2017/

978-1-5386-1562-1/17 $31.00 © 2017 IEEE
DOI 10.1109/IRI.2017.39

support vector machines, and neural networks) seem to indicate that:

- the best variants of these major approaches all deliver competitive results, with accuracy scores all falling within a range of 2 to 3 percentage points, when trained with general domain datasets of size 30k [4].

Against this background, another result that would be important to have is to know about the impact that different amounts of learning data may have on the performance of systems trained with major approaches. This is all the more important as it is known, from experiments with other NLP tasks, that different machine learning techniques scale differently as the amount of available training data increases, to the point that with different amounts of training data the ranking between the performance of different systems can suffer dramatic changes and even get reversed [5], [6].

Undertaking a study on how the training data size affects different types of approaches to the task of DQD has so far been hampered by the lack of a dataset large enough to plot learning curves over a relevant range of dataset sizes. The recent release, three months ago in January 2017, of the Quora dataset, with over 400k pairs of grammatically well-formed interrogative segments [7], is now enabling such a study.

In the current paper we use the Quora dataset to analyze how different methodologies for the DQD task perform as the dataset size increases. We will be using systems that from previous systematic experimentation with training datasets of size 30k [4], are identified as top-performing representatives of three families of approaches to DQD.

The remainder of this paper is structured as follows: The systems used to support the present study are described in the next Section II. Section III presents the Quora dataset and its preparation, while Sections IV and V report on the results obtained and discuss them. Section VI covers related work and Section VII concludes with a summary of major results obtained and final remarks.

II. Approaches and Systems

The present study was undertaken by resorting to three systems, each being a top-performing representative of a major methodological approach to DQD: one relies on rule-based heuristics, a second is based on a machine-learning classifier that uses lexical and semantic features, and the third one is based on a deep convoluted neural network. These systems were selected because in a previous systematic experimentation on different methodologies to DQD and their variants [4], they achieved the best performance for the type of approach they belong to when developed over a 30k dataset.

A. Rule-based

A first approach uses the Jaccard index over all sets of n-grams of each question, with n ranging from 1 to 4. Being a quite straightforward statistic, this nonetheless supports a very competitive baseline system for DQD [8].

Two questions are considered similar if their Jaccard index is above a certain value. For each dataset size, this similarity threshold is determined by picking the value that achieves the best accuracy on the training data. To find the best threshold, the training set is used in a series of trials whose outcome is applied to the test set.

In the current study, this led to the threshold of 0.1 for all our seven datasets, of size 7k, 15k, 30k, 97k, 165k, 232k and 300k.

Before extracting the n-grams, the questions were tokenized and stemmed, using NLTK [9].

B. Support Vector Machine

Support Vector Machine (SVM) classifiers have been used with great success in many NLP tasks, including DQD [3], [4]. Taking into account these previous studies in the literature, the best-performing system adhering to this methodology relies on a feature vector that is composed as follows for each pair of interrogative segments in the dataset:

1) a vector resulting from the concatenation of the two vectors, for the two segments in the pair, with a one-hot encoding of the n-grams (with n from 1 to 4) with more than 10 occurrences in the training dataset;
2) four scores of the Jaccard index of the pair, respectively when 1, 2, 3 and 4-grams are considered;
3) the two numbers of negative words (e.g. *never*, *nothing*, etc.), for the two segments (after negative words in the text having been normalized: e.g. "n't" $\rightarrow$ "not", etc.);
4) the number of nouns that are common to both questions, provided that they are not already included in the n-grams selected in the steps above;
5) and the cosine similarity score between the distributional semantic vector representations of each segment.

Components 1–4 are lexical features that need only to consider the surface form of each segment and the POS tags assigned during the preprocessing stage.

Being k the total number of 1, 2, 3 and 4-grams with more than 10 occurrences in the training dataset, the one-hot encoding of both interrogative segments in the pair will account for $2k$ features. These are extended with the additional features from the Jaccard scores (4 features), number of negative words in each question (2 features) and number of shared nouns (1 feature).

Component 5, i.e. a single feature with the cosine similarity score between the vector representations of the two questions, is a semantic feature that merits a lengthier description. For each question, its distributional semantic vector (embedding) was obtained by adding up the embeddings of all the nouns and verbs occurring in it (combinations

of words from other POS categories were found to lead to worse performance during experimentation) [4].

The embeddings are based on WordNet synsets given that preliminary tests showed that synset embeddings performed better than word embeddings [4]. To obtain these synset embeddings we used word2vec [10] followed by Autoextend [11]. The supporting ontology was version 3 of Princeton WordNet, which contains over 120k synsets [12].

The SVM implementation in scikit-learn [13] was used.

C. Deep Convolutional Neural Network

A range of neural network architectures was experimented with, including those reported in the literature as delivering competitive results, and the best performance was achieved with a deep convolutional neural network (DCNN) [4]. This is a novel architecture we developed that combines two of the best architectures for DQD reported in the literature, namely a convolutional neural network inspired by [3] and a deep neural network inspired by [14], which is the top performing system in the "Question-Question" subtask of SemEval 2016 Task 1 [1].

The Keras [15] and Tensorflow [16] were resorted to for the implementation of this DCNN system. A diagram of its architecture is displayed in Figure 1.

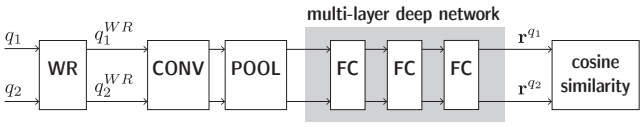

Figure 1. DCNN architecture: word representation (WR), convolution (CONV), pooling (POOL), fully connected layers (FC) and cosine similarity measurement layers

The dataset was prepared with a simple pre-processing stage where the segments were tokenized with NLTK [9], and the tokens were lowercased.

The DCCN is fed with the two segments from an input pair of segments, each going through the same layers and processing procedures all along the neural network, thus resorting to a Siamese architecture.

This neural network starts by obtaining the vectorial representations for the words in each interrogative segment by means of its initial embedding layer. This layer eventually acts as a distributional semantic space by learning a vector for each word during the training of the network. These word vectors are randomly initialized with size 300 and are learned through the training process, where during the back-propagation, the vector for each word gets adjusted and eventually determined given the neighboring words and the task at hand with which the cost function is computed.

The vocabulary of the distributional semantic space represented in this layer is made up of all the word types found in the training dataset. Hence, this first layer uses the same number of neurons as the number of items in the vocabulary by resorting to a one-hot encoding representations of the vocabulary items.

In order to obtain a vectorial representation of the segments, the resulting word vectors are passed to a convolution layer combined with a pooling layer that resorts to a max filter. The convolution layer uses 300 neurons and contains a convolutional window with a kernel size of 15. The weights in the convolution layer are shared for the two segments in the input pair of segments.

The resulting vectorial representations of each segment are then fed through a deep network of three fully-connected layers, each with 50 neurons.

All these layers of the DCNN share the same weights for every one of the interrogative segments in the pair. The two output representations of this Siamese network are then compared by using the cosine similarity between them. If the cosine is above an empirically determined threshold, obtained with the help of a hyperbolic tangent activation function, the segments are classified as being duplicates.

For every one of the experiments undertaken, with training datasets of different sizes, the learning process was performed with a 0.01 learning rate, with hyperbolic tangent as activation function and a stochastic gradient to compute the cost function with a mean squared error loss. In each experiment, the best score was picked from a twenty epoch training.

III. Datasets

Quora[2] is an online moderated question answering site where each query introduced by a user, typically in a grammatical interrogative sentence, receives an answer, often from a volunteer expert.

Three months ago, in January 2017, Quora released a dataset with 404,290 potential duplicate question pairs [7]. Each entry in this dataset contains a pair of questions and a class label indicating whether the questions in the pair are duplicate or non-duplicate, as in the examples in Figure 2.

The segments are written in English and may address any topic, which makes of its collection a dataset of a generic domain.

1	**Q** Why are police lights red and blue? **Q** Why are police lights red and/or blue?
0	**Q** Can anyone predict my birth chart? **Q** What can you tell me regarding this birth chart?
1	**Q** Why are dogs considered omnivores? **Q** Are dogs carnivorous animals?

Figure 2. Three example pairs of interrogative segments and their labels from the Quora dataset

The sampling method that was used to collect this dataset is reported as having returned an initially imbalanced dataset, with many more duplicate than non-duplicate

[2]https://www.quora.com/

pairs. Non-duplicate examples were subsequently added, including pairs of "related questions". The dataset eventually released has 149,263 (37%) duplicate and 255,027 (63%) non-duplicate pairs [7].

Also, it is indicated that the class labels contain some amount of noise and are "not guaranteed to be perfect", though the estimation of the possible amount of noise is not reported.

For the experiments in the current paper, we use a balanced subset of this Quora release, where there is the same number of duplicate and non-duplicate pairs.

Given this design option for our experiments, the largest dataset size that could be, and was, used is then roughly 300k pairs (298,526 pairs, to be precise), which is twice the minority class. The other six data sets experimented with are subsets of this balanced 300k subset of the Quora dataset released.

One the subsets of this largest balanced dataset are of size 30k, 10 times smaller than it, as that was also the size of the datasets previously used in [3] and [4].

To obtain a few more dataset sizes with which to plot learning curves, we pick the midpoint of this 30k–300k range, and also the midpoints of each of the two resulting parts, giving us three equally spaced points within that range. These three points are, roughly, at sizes 97k, 165k and 232k.

Two other smaller subsets were yet used: one at size 15k, the midpoint in the 0k–30k range, and another at 7k, the midpoint in the 0k–15k range.

The token and vocabulary (word types) counts for each one of these seven datasets are summarized in Table I.

#pairs	#tokens	vocab. size	tokens/sentence
7k	154,755	11,759	22.11
15k	331,067	17,512	22.07
30k	657,317	24,509	21.91
97k	2,129,777	42,292	21.96
165k	3,600,229	53,552	21.82
232k	5,074,653	62,377	21.87
300k	6,552,447	69,774	21.84

Table I
STATISTICS FOR THE DATA SUBSETS USED

The entries in the Quora dataset were randomly shuffled to avoid any possible bias coming from their original position in the released file.

For each dataset size, and in every experiment, 80% of the pairs of interrogative segments, randomly extracted, are used for training, and the other 20% for testing.

IV. RESULTS

In this section we present the performance results obtained in terms of accuracy in duplicate question detection when DQD resolvers are learned under the approaches and by their respective systems described in Section II, with the support of the data subsets presented in Section III.

A. Rule-based

The Jaccard index results from proportions straightforwardly obtained from the interrogative segments in the input pair. Hence the performance of a system learning a baseline DQD resolver that is based on this index is not expected to be too much sensitive to the size of the datasets used if these are large enough, and in our experiments this expectation should be all the stronger as the actual values for the thresholds empirically determined with the help of the seven training datasets happened to be all identical to 0.1.

	7k	15k	30k	97k	165k	232k	300k
JACC	69.00	70.07	70.30	69.55	69.21	69.37	69.50

Table II
ACCURACY (IN PERCENTAGE) OF THE DQD RESOLVER BASED ON JACCARD INDEX OVER THE DIFFERENT DATASETS, ORDERED BY THEIR INCREASING SIZE

The accuracy results obtained, and displayed in Table II, are aligned with this expectation. They fall all in a range of scores of about 1.3 percentage point wide.

B. Support Vector Machine

The time required for training over the 7k, 15k, 30k, 97k and 165k datasets were about 0.01, 0.08, 1.5, 40 and 140 hours respectively. Obtaining these values required approximately 0.8, 1.2, 3.6, 43 and 83.5 GB of RAM respectively for each data set, when running with a single CPU machine as required by the SVM implementation used. This imposed practical limitations to our study, leading us not to train the SVM-based DQD resolver for the remaining, larger datasets.

The size of the feature vector for an input pair grows as the vocabulary size increases with the growth of the training data subset sizes that were lined up to support the experiments in this study. Since the bulk of the feature vector is the one-hot encoding of n-grams (with n from 1 to 4), its size should grow much faster than the growth of the vocabulary, reflecting an increasingly longer tail of low frequency n-grams as we move along the increasingly larger data subsets used for training.

This is a major reason for the hurdle represented by the very fast growing time required for the training of the DQD resolvers. And its impact is showing up also in the decrease that it is observed in terms of the level of performance of the DQD resolvers, as their accuracy scores, displayed in Table III, fall with the increasing of the size of the training data subsets.

In this connection, it is also worth noting that to a certain extent this is in line with the expected behavior of SVMs with RBF kernels under these circumstances, as "the fit time complexity is more than quadratic with the number of samples which makes it hard to scale to dataset with more than a couple of 10000 samples" [17], [13].

	7k	15k	30k	97k	165k
SVM	67.64	68.83	68.56	67.97	66.55

Table III
ACCURACY (IN PERCENTAGE) OF THE DQD RESOLVER BASED ON SVM OVER THE DIFFERENT DATASETS, ORDERED BY THEIR INCREASING SIZE

C. Deep Convolutional Neural Network

The accuracy scores for the DQD resolvers based on DCNN that were trained and tested over the different data subsets are displayed in Table IV. The performance of the resolvers increases as the number of learning examples increases, in line with what is the expected behavior of deep neural networks [18], [19].

	7k	15k	30k	97k	165k	232k	300k
DCNN	62.29	66.07	70.33	73.43	74.13	76.39	77.64

Table IV
ACCURACY (IN PERCENTAGE) OF THE DQD RESOLVER BASED ON DCNN OVER THE DIFFERENT DATASETS, ORDERED BY THEIR INCREASING SIZE

The time required for training over the experimental data subsets also grows here as the size of these datasets grows but in a less steep pace than for SVMs. The times required for training over the 7k, 15k, 30k, 97k, 165k, 232k, 300k were approximately 0.5, 1, 3, 12, 24, 34 and 44 hours respectively. These values were obtained with a Tesla K40c GPU dedicating all of its 12 GB memory size board to each experiment.

As it can be observed in Table I, after 97k, for every step with an increase of approximately 70k pairs of interrogative segments, there is an increase of approximately 7 to 9k items in the respective vocabularies.

The first layer of the DCNN, devoted to getting the vectorial semantic representation of words of input segments, uses the same number of neurons as the number of items in the vocabulary. The size of the subsequent layers, in turn, maintains their number of neurons irrespective of the different sizes of the training datasets and of their vocabularies.

V. DISCUSSION

The experimental results described in Section IV are plotted together in Figure 3.

These results empirically support the plausibility of a number of insights and conclusions of different scope, ranging from specific results on the effectiveness of the technical approaches to DQD, to broad interest issues concerning the profile of machine learning methods to NLP tasks, and including lessons on the research methodology commonly adopted for DQD and other NLP topics.

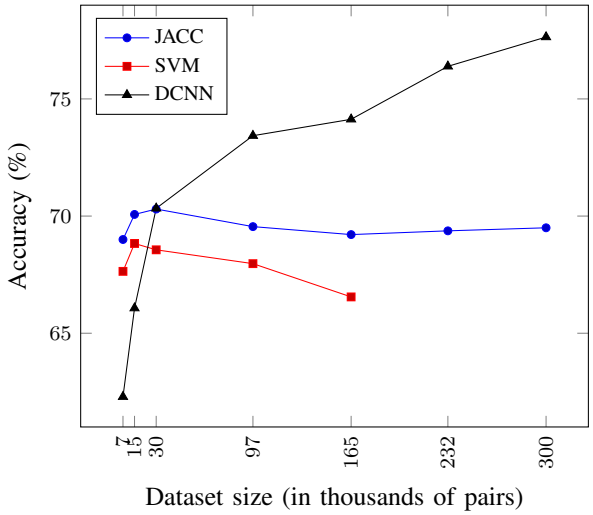

Figure 3. Learning curves of the three approaches over the experimental datasets

On duplicate question detection specifically:

- The best DQD performance results depend not only on the type of approach but also on the size of the dataset: There is not an "absolute" best methodology to DQD.
- For small training datasets, of size inferior to 30k, the technically leaner approach — supported by rule-based heuristics — delivers highly competitive results: given its agile setup and top accuracy scores, in order to support applications that cannot resort to larger training datasets, it is the superior methodology.
- For larger training datasets, of a size larger than 30k, the technically more sophisticated approach — based on deep learning — delivers highly competitive results: to support applications that can resort to such larger training datasets, it is the superior methodology.
- Training datasets of size around 300k-500k seem to support pretty good performance (for the best methodology with training datasets of size larger 30k) and further improvements will require much larger datasets for increasingly smaller deltas of improvement: this is a conjecture whose plausibility is empirically supported by the apparently asymptotic shape of the DCNN curve, and that naturally calls to be further researched with larger training datasets.

Concerning machine learning methods to NLP:

- The ranking of different machine learning classes of solutions in terms of their performance for some NLP tasks may change as the size of the training datasets change, even to the point that there may be dataset sizes where the approach that was the best swaps with the approach that was the worst when these were trained over a dataset with a different size. This observation gained its first contributions from [5], [6], and its

empirical support is thus further strengthened with the results obtained in the present study.

And on research methodology commonly adopted for DQD and other NLP topics:

- Studying the virtues of different methodologies and approaches for a given NLP tasks or research topic on the basis of a data "slice" — that is some training data of a given size — supports an insufficient insight on the strengths and weaknesses of the different approaches for addressing that task: this is an observation that receives also further empirical support from the present study.

VI. Related Work

An approach to DQD with a Jaccard coefficient in [8] was used to measure similarities between two interrogative segments in a pair. A DQD dataset was created resorting to the Baidu Zhidao, a question and answer forum in Chinese provided by the Baidu search engine. Training with 3M pairs and testing on 3k pairs, the system scored an f-score of 60.29.

[3] introduced the architecture of convolutional neural network to tackle DQD. This network obtains the vectorial representation of the words in the two input segments, and the next convolutional layer constructs a vectorial representation for each one of the two segments. Finally, the two representations are compared using cosine similarity. This system was reported to score over 92% accuracy, resorting to 30k data taken from the Meta forum in StackExchange and AskUbuntu forum, with an 80%/20% train/test split.[3]

A deep neural network approach to DQD in [14] obtained the best accuracy in the SemEval-2016 Task 1 "Question-Question" subtask, namely 0.73035 in terms of Pearson correlation coefficient, which had the objective of determining the degree of similarity, on a 0–5 scale, between two interrogative sentences.

In what concerns the Quora dataset, released 3 months ago, to the best of our knowledge, there is only one unpublished paper concerning DQD up to now [20]. It proposes a multi-perspective matching (BiMPM) model. The resulting DQD resolver is reported to reach an accuracy of 88.17% when evaluated upon a 96%/2%/2% train/dev/test split.

Other draft results concerning the Quora dataset are available only as blog posts, namely 83% from [21] and 87% from [22], both based on the model for natural language inference proposed by [23].

VII. Conclusions

The study reported in the present paper permits to advance the understanding of methodologies for the automatic duplicate question detection task and of their application.

[3]A replication of this experiment in [4] was found to drop almost 20 percentage points when the indication of the titles of the possible duplicate questions are removed from the questions and their threads in the dataset used.

A major conclusion is that the more technically sophisticated type of approach resorted to, namely deep learning, has clearly a superior performance than other approaches, provided it can be trained over a sufficiently large dataset.

Interestingly, for smaller training datasets, the performance of the more technically lean type of approach, which is based on Jaccard index, shows a clear advantage over the other much more sophisticated approaches.

In future work, it will be interesting to study the apparent asymptotic progress of the learning curve of the approach based on deep convoluted neural networks in order to understand, in case this behavior gets confirmed, what is the order of magnitude for the size of the training dataset that is enough to ensure a pretty good performance of this approach. This will help to design applications where duplicate question detection is embedded, in particular to guide an effective collection of the respective training dataset with a good enough size.

Acknowledgements

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

References

[1] E. Agirre, C. Banea, D. Cer, M. Diab, A. Gonzalez-Agirre, R. Mihalcea, G. Rigau, and J. Wiebe, "Semeval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation," in *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, 2016, pp. 497–511. [Online]. Available: http://www.aclweb.org/anthology/S16-1081

[2] E. Agirre, A. Gonzalez-Agirre, D. Cer, and M. Diab, "Semeval-2012 task 6: A pilot on semantic textual similarity," in *Proceedings of 1st Joint Conference on Lexical and Computational Semantics*, 2012, pp. 385–393.

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

[4] J. Rodrigues, C. Saedi, V. Maraev, J. Silva, and A. Branco, "Ways of asking and replying in duplicate question detection," in *Proceedings of the workshop of ACL2017- The 55th Annual Meeting of the Association for Computational Linguistics, *SEM 2017: 6th Joint Conference on Lexical and Computational Semantics*, 2017.

[5] M. Banko and E. Brill, "Mitigating the paucity of data problem: Exploring the effect of training corpus size on classifier performance for NLP," in *Proceedings of the 1st Human Language Technology (HLT) Conference*, 2001, pp. 1–5.

[6] ——, "Scaling to very very large corpora for natural language disambiguation," in *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL) and 10th Conference of the European Chapter of the ACL (EACL)*, 2001, pp. 26–33.

[7] S. Iyer, N. Dandekar, and K. Csernai, "First Quora dataset release: Question pairs," 2017, https://data.quora.com/First-Quora-Dataset-Release-Question-Pairs.

[8] Y. Wu, Q. Zhang, and X. Huang, "Efficient near-duplicate detection for Q&A forum," in *Proceedings of the 5th International Joint Conference on Natural Language Processing (IJCNLP)*, 2011, pp. 1001–1009. [Online]. Available: http://aclweb.org/anthology/I/I11/I11-1112.pdf

[9] S. Bird, E. Klein, and E. Loper, *Natural Language Processing with Python*. O'Reilly Media, 2009.

[10] T. Mikolov, W. Yih, and G. Zweig, "Linguistic regularities in continuous space word representations," in *Proceedings of the Human Language Technologies (HLT) Conference*, 2013, pp. 746–751. [Online]. Available: http://aclweb.org/anthology/N/N13/N13-1090.pdf

[11] S. Rothe and H. Schütze, "Autoextend: Extending word embeddings to embeddings for synsets and lexemes," *arXiv preprint arXiv:1507.01127*, 2015.

[12] C. Fellbaum, *WordNet*. Wiley Online Library, 1998.

[13] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[14] N. Afzal, Y. Wang, and H. Liu, "MayoNLP at SemEval-2016 task 1: Semantic textual similarity based on lexical semantic net and deep learning semantic model," in *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, 2016, pp. 1258–1263.

[15] F. Chollet, "Keras," 2015, available from https://github.com/fchollet/keras. [Online]. Available: https://github.com/fchollet/keras

[16] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: http://tensorflow.org/

[17] A. Abdiansah and R. Wardoyo, "Time complexity analysis of support vector machines (SVM) in libsvm," *International Journal Computer and Application*, 2015.

[18] X. Zhu, C. Vondrick, C. C. Fowlkes, and D. Ramanan, "Do We Need More Training Data?" *International Journal of Computer Vision*, vol. 119, no. 1, pp. 76–92, 2016. [Online]. Available: http://dx.doi.org/10.1007/s11263-015-0812-2

[19] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[20] Z. Wang, W. Hamza, and R. Florian, "Bilateral multi-perspective matching for natural language sentences," *CoRR*, vol. abs/1702.03814, 2017. [Online]. Available: http://arxiv.org/abs/1702.03814

[21] M. Honnibal, "Deep text-pair classification with quoras 2017 question dataset," 2017, https://explosion.ai/blog/quora-deep-text-pair-classification.

[22] L. Jiang, S. Chang, and N. Dandekar, "Semantic question matching with deep learning," 2017, https://engineering.quora.com/Semantic-Question-Matching-with-Deep-Learning.

[23] A. P. Parikh, O. Täckström, D. Das, and J. Uszkoreit, "A decomposable attention model for natural language inference," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1–4, 2016*, 2016, pp. 2249–2255. [Online]. Available: http://aclweb.org/anthology/D/D16/D16-1244.pdf