How to Deal with Scarce Annotations in Answer Selection

Emmanuel Vallee
Orange Labs
manu.vallee@gmail.com

Delphine Charlet
Orange Labs
delphine.charlet@orange.com

Gabriel Marzinotto
Aix Marseille Univ, CNRS, LIS
gabriel.marzinotto@lis-lab.fr

Fabrice Clerot
Orange Labs
fabrice.clerot@orange.com

Frank Meyer
Orange Labs
franck.meyer@orange.com

ABSTRACT

Addressing Question Answering (QA) tasks with complex neural networks typically requires a large amount of annotated data to achieve a satisfactory accuracy of the models. In this work, we are interested in simple models that can potentially give good performance on datasets with no or few annotations. First, we propose new unsupervised baselines that leverage distributed word and sentence representations. Second, we compare the ability of our neural network architectures to learn from few annotated samples and we demonstrate how these methods can benefit from a pre-training on an external dataset. With a particular emphasis on the reproducibility of our results, we show that our simple models can approach or reach state-of-the-art performance on four common QA datasets.

CCS CONCEPTS

• **Information systems → Question answering**.

KEYWORDS

Neural Networks; Natural Language Processing; Question Answering; Answer Selection

ACM Reference Format:
Emmanuel Vallee, Delphine Charlet, Gabriel Marzinotto, Fabrice Clerot, and Frank Meyer. 2019. How to Deal with Scarce Annotations in Answer Selection. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '19), July 21–25, 2019, Paris, France.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3331184.3331291

1 INTRODUCTION

Large-scale Question Answering (QA) tasks have recently received a substantial amount of attention. In this paper, we are interested in the QA subtask of Answer Selection (AS), where the goal is to retrieve the correct answers to a question amongst a set of candidate answers.

Most efforts to address the AS task have been directed towards supervised end-to-end training of complex neural architectures. These models are typically based on Convolutional [16] or Recurrent Neural Networks [14], with hundreds of thousands of parameters, and a set of hyperparameters tuned for the target dataset. Faced with the profusion of methods, it becomes difficult to decide which one to use on a given dataset. Here, we question the real added value of Deep Learning methods compared to unsupervised baselines and simple Neural Networks approaches.

Recently, it was proposed to address the AS task with simple models based on word embedding pooling operations and simple neural architecture [11]. A fully unsupervised approach was proposed in [15]. The main contribution of our work is to address the issue of datasets with few or no annotated data. For this, we propose a novel unsupervised method relying on the Hungarian method, as well as distantly supervised neural approaches. Additionally, we explore the capacity of simple models to achieve few-shot learning.

To overcome the scarcity of annotated data, we propose to exploit a pre-trained sentence encoding model, the Universal Sentence Encoder (USE) [2]. Going one step further, we present a domain adaptation approach, where we pre-train our models on an external and larger QA dataset, and then fine-tune the pre-trained models on the target dataset, similar to the work in [8].

Importantly, we make a major effort to achieve fully reproducible results on the initial dataset. Indeed, it is well-known that the deep learning community is facing a reproducibility crisis due to poorly controlled environment settings [3].

We demonstrate that our methods reach near state-of-the-art results on four publicly available datasets. The source code to reproduce our experiments is available at [1].

2 METHODS

2.1 Unsupervised approaches

One of our goals being to assess the performance without any annotated data, we compare a number of unsupervised baselines.

BM25: An IR baseline where the score for each candidate answer is given by the BM25 weights of question words in the answer.

Average word embedding: Each word is represented by a 300 dimensional vector and the word vectors are averaged across all the sentence to obtain the sentence vector. The score is given by the cosine similarity between the question and the answer vectors.

Hungarian method for embedding alignment: We compute a similarity matrix between each vector representing the word of the question and answer. We then adapt the Hungarian method for maximal alignment similarity. The score is given by the Hadamard product between the similarity matrix S and the optimal alignment matrix X:

$$score(Q,A) = S \odot X \tag{1}$$

$$S_{i,j} = Q_i \cdot A_j \tag{2}$$

[1] https://github.com/manuvallee/qa_matching

SIGIR '19, July 21–25, 2019, Paris, France

ACM ISBN 978-1-4503-6172-9/19/07…$15.00
https://doi.org/10.1145/3331184.3331291

where Q_i and A_j are the embedding vectors of the i^{th} question word and the j^{th} answer word.
USE: We encode each question and answer with the Universal Sentence Encoder (USE) [2] into a 512 dimensional vector and compute the score with the cosine similarity between the two vectors.

2.2 Neural approaches

Amongst the broad family of neural architectures and training approaches available in the AS litterature [6], we propose to evaluate two basic neural architectures coupled with two learning approaches.
The **siamese** architecture learns a *projection* of the input vectors and consists of one hidden layer with shared weights between the question and the answer, followed by a cosine similarity that gives the matching score for the Q/A pair. For this architecture the question and the answer are encoded independently.
The **concat** architecture learns the *similarity* between the question and the candidate answer. It concatenates the question and answer vectors, as well as their elementwise product and their absolute difference, to obtain the QA pair representation. It is followed by a single hidden layer, and by a sigmoid output layer that gives the matching score for the Q/A pair.
Additionally, we compare two training approaches for our AS task.
The **pointwise** approach that, given a question, learns to *classify* the candidate answers according to their binary label, and the loss function is the binary cross-entropy.
The **pairwise** approach learns to *minimize the distance* between the positive answer and the question while *maximizing the distance* between the negative answer and the question. In this case, the loss function is the triplet-loss: $\mathcal{L} = max\left(-d(Q,A^+)+d(Q,A^-)+m,0\right)$, where d is the distance between the pair given by the output layer of a given network, Q is the question, A^+ is a positive answer, A^- is a negative answer, and m is the margin.
Finally, we use two different vector representations for the input sequence: the **word2vec** (w2v) word embedding averaged across all the sentence words, or the sentence embedding computed with the **USE**.

Domain adaptation. Following a recent study on domain adaptation for AS task [8], we pre-train our models on an external dataset and fine-tune them on the target dataset. Importantly, pre-training a model on an external dataset allows deploying the model as such, without further fine-tuning on the target dataset. This approach could be of major interest when no annotated data is available.

3 EXPERIMENTS

3.1 Experimental data

We evaluate our methods on four publicly available datasets:
WikiQA: Open domain questions extracted from Bing requests. The answers are sentences from a Wikipedia paragraph. Following recent work, we remove all questions with no correct answers.
TrecQA: Dataset created from the TREC Question Answering tracks [13]. We use the CLEAN version, where all the questions with only positive or only negative answers are removed.
Semeval - Task 3 - English subtask: Questions and comments from a life forum in Qatar. There are potentially several sentences per question and per answer [9].
Yahoo: 10K "How" questions, each with a community chosen best answer [4].
Importantly, Semeval and Yahoo datasets are Community Question Answering (CQA) with user generated content and therefore present a domain shift with WikiQA and TrecQA, which are open domain QA with encyclopedia sourced answers.

3.2 External resources and pre-trained models

The first external resource is the training data used to compute the word embedding. The generic w2v embedding that is trained on the Google News dataset (about 100 billion words) and publicly available [2]. The word2vec embedding can alternatively be trained on a large set of unannotated data from the same source as the target corpus. These **specific embeddings** are built from Wikipedia for WikiQA and TrecQA datasets, and from the target corpus for Yahoo answers and SemEval. We used the skip-gram w2v embedding approach with a window of 8 tokens and a minimum word count of 10.
Second, we propose to employ the pre-trained USE model to compute an embedding for the question and for the answer [2]. It produces a 512 dimensional vector for a given sentence.
Finally, we leverage the large SQuAD QA dataset [10] modified for sentence level AS and denoted SQuAD-T [8]. We note that SQuAD-T presents a domain shift with the CQA datasets, i.e., Semeval and Yahoo.

3.3 Experimental setup

Preprocessing. For the USE representation, no preprocessing is necessary as the model was trained on raw text. For the other methods, we apply the following: switching to lower case, deleting diacritics (glyph added to a letter, or basic glyph), adding spaces around the punctuation, removing confusing signs (coma and dots in numbers, smileys), and replacing URLs by a "_url_" tag. We also remove stopwords as defined in the NLTK english list. [3]

Networks parameters. The hyperparameters of our neural networks are fixed a priori and kept constant across datasets. For both architectures, the hidden layer has a number of units equal to the dimensionality of the preceding layer. For example, with the siamese architecture and a USE representation (d=512), the hidden layer has 512 units. Each hidden layer is followed by a dropout layer with a rate of 0.3. The only parameter optimized across datasets is the margin for the pairwise approach. The optimal values for the development data splits are 0.05 for SQuAD-T, 0.01 for WikiQA, 0.02 for SemEval, 0.1 for TrecQA and 0.5 for Yahoo. The optimization is conducted with Adam optimizer and its default parameters are as proposed in the original paper [5]. The batch size is fixed to 32 and we use an early stopping scheme to avoid overfitting.

Evaluation. The evaluation is performed with the official SemEval scorer to compute the metrics [4]. Note that we evaluate all the question sets, including when there is no correct answer for a given question. To ensure reproducible results, each experiment is repeated 10 times with different question subsets, and we report the mean of each metric. To be consistent with the litterature, we report the Mean Average Precision (MAP) for WikiQA, Semeval and TrecQA and the precision at one (P@1) for the Yahoo dataset. We consider that

[2] https://code.google.com/archive/p/word2vec/
[3] https://www.nltk.org/
[4] http://alt.qcri.org/semeval2016/task3/index.php?id=data-and-tools

two results are significantly different when the p-value computed with the Wilcoxon test is smaller than 0.05. For conciseness, we only present the results obtained with the best architecture and approach for each of our use case.

4 RESULTS

4.1 No labels on the target dataset

Unsupervised methods. In Table 1 (top and middle rows), we show the results obtained with the IR baseline and the unsupervised methods on the four datasets. We can reach acceptable performance with fully unsupervised baselines, and two methods stand out from the others. The Hungarian approach with a specific word embedding performs the best on 3 datasets, with a significant difference with the second best (Hungarian with generic embedding) on Semeval and TrecQA datasets. On the Yahoo dataset, there is no significant difference between generic and specific embedding, neither with the Hungarian nor the average embedding approach. We note that the USE representation performs the best only on the WikiQA dataset, with a significant difference with the Hungarian method.

Table 1: Results obtained with fully unsupervised and distantly supervised methods. Pre-trained refers to the pointwise/concat neural architecture pre-trained on SQuAD-T, without fine-tuning on the target dataset. * indicates that the best result is significantly different from the second best result.

Method	Specific w2v	WikiQA MAP	TrecQA MAP	SemEval MAP	Yahoo P@1
BM25	N/A	58.52	67.77	60.95	20.04
w2v cos	No	60.22	56.61	62.91	23.76
	Yes	61.83	67.66	64.15	24.16
Hungarian	No	64.40	71.47	64.52	**29.48**
	Yes	65.98	**76.16***	66.15	29.08
USE cos	No	69.66	72.23	64.16	22.40
Pre-trained w2v	No	61.48	69.12	64.33	21.84
Pre-trained w2v	Yes	62.95	70.71	63.92	22.92
Pre-trained USE	No	**75.70***	72.05	**70.16***	27.72

Distantly supervised neural approaches. At the bottom of Table 1, we report the results obtained with the best neural model (pointwise/concat) pre-trained on the SQuAD-T dataset and deployed on the target dataset without fine-tuning (distantly supervised). Our first observation is that the USE representation always outperforms the w2v representation. The obtained performance is superior to that of fully unsupervised approaches on WikiQA and Semeval, while the Hungarian method is better for TrecQA and Yahoo.

4.2 Few labels in the target dataset

In Figure 1 and 2, we show the learning curve of our methods for increasing amount of training data, from 4 questions up to the full training dataset. The traced line represents the mean MAP across 10 runs, while the shaded area represents the standard deviation.

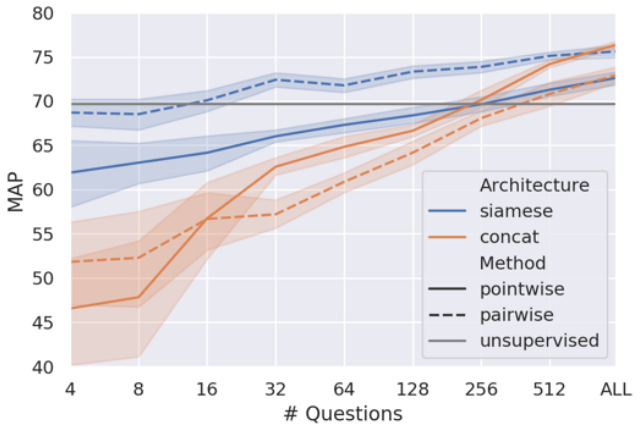

Figure 1: MAP obtained on the WikiQA dataset when training the model on an increasing number of questions. We compare two architectures, siamese and concat, as well as the two training approaches, pointwise and pairwise. The grey line represents the MAP obtained with the unsupervised USE method (cosine).

In Figure 1, we illustrate the performance of two architectures (siamese and concat) combined with the two training approaches (pointwise and pairwise) on the WikiQA dataset. We notice that the concat architecture gives a low performance with few questions, whereas the siamese one appears to learn efficiently, most likely thanks to its lower number of parameters and fixed similarity function. We note that with the full training set, the pairwise/siamese and pointwise/concat give similar performances. We observe the same trend on the other datasets, except on Semeval. For the latter, the concat/pointwise architecture performs better, and all the approaches and architectures are above the unsupervised baseline for any number of questions.

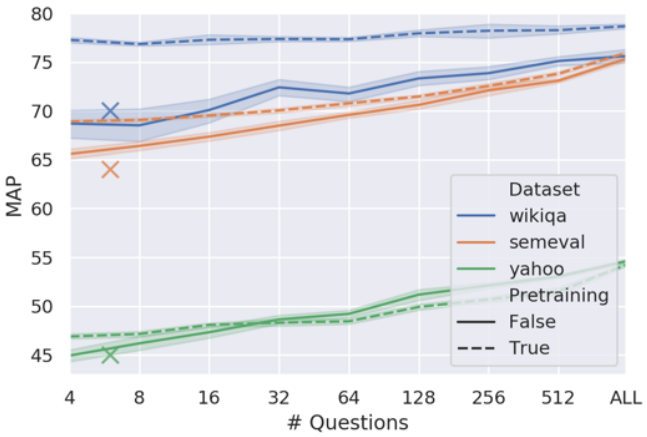

Figure 2: MAP obtained on the WikiQA, Semeval, and Yahoo datasets when training the pairwise siamese model with the USE representation on an increasing number of questions. We asses the gain obtained when pre-training the model on SQuAD-T. The crosses indicate the MAP with the unsupervised USE method (cosine).

In Figure 2, we show the effect of the pre-training on the performance on the WikiQA, Semeval, and Yahoo dataset. The results shown are obtained with the pairwise/siamese approach, which proves to be the best combination for few-shot learning.

Pre-training is highly effective for the WikiQA (and TrecQA) datasets when training is performed on a few questions. In the case of Semeval and Yahoo, pre-training improves the performance when the size of the training set is small. For the Yahoo dataset, pre-training is only effective for a low number of training question (up to 16) and is detrimental when more questions are used for training.

Importantly, we note that the siamese architecture is very effective at learning a representation that outperforms the unsupervised baseline (shown with a cross). Indeed, the model can learn a better representation than the original baseline with as little as four questions.

4.3 All labels in the target dataset

In Table 2, we report the results obtained with the full training set. In this case, the best performing architecture is the concat one, with a pointwise training approach. We only display the latter as it significantly outperforms the other methods and approaches.

Table 2: Results obtained using the full training set with the concat/pointwise architecture, with and without pre-training on SQuAD-T and with the w2v or USE representation. † indicates that the method used SQuAD-T dataset for pre-training the model. * indicates that the best result is significantly different from the second best result.

Method	WikiQA MAP	TrecQA MAP	SemEval MAP	Yahoo P@1
concat - w2v	60.11	73.23	72.10	27.50
concat - w2v †	66.14	**76.60***	73.43	27.30
concat - USE	76.36	65.37	**80.57***	**36.65***
concat - USE †	**78.30***	72.52	80.45	34.87
SOTA	**79.90**† [8]	**86.57** [12]	80.36 [7]	**38.74** [1]

On the WikiQA dataset, we obtain very good results, with a MAP close to the state-of-the-art which is an attention-based RNN architecture also pre-trained on SQuAD-T [8]. Without pre-training, we obtain a MAP of 76.36, which is superior to the state-of-the art without pre-training (74.50 MAP [17]).

Results on the TrecQA dataset exhibit a different pattern, as the w2v representation performs better than the USE one. However, the performance reached by our approach is 10 points below the state-of-the-art. The MAP is slightly, yet significantly, higher than with the unsupervised Hungarian method.

On the Semeval dataset, the pre-training has a small, yet significant, effect on the performance, and our results are superior to the current state-of-the-art.

For the Yahoo dataset, we reach good performance with the USE representation, only 2 points below the state-of-the-art. For this dataset, the pre-training is also detrimental (-1.8 points, $p < 0.05$).

5 CONCLUSION

In this paper, we explored several approaches for the Answer Selection task on 4 datasets and for varying amount of labeled samples. We provided guidelines on the choice of the method depending on the availability of annotations and external resources.

We designed a new unsupervised approach (Hungarian) that remains the safest choice when no annotation or external dataset are available. When an external dataset is available, the distantly supervised approach is better than the Hungarian method on two datasets out of four. Importantly, we found that the pairwise/siamese method allows for low-shot learning and that pre-training the model can boost the performance further.

Finally, we found that when a larger amount of data is available, our models can compete with much more complex approaches typically relying on convolutional and/or recurrent models with attention mechanisms [14]. We argue that the effectiveness of our methods relies on the use of pre-trained language model (USE) whereas other methods usually perform end-to-end training, which might not be appropriate for the relatively small size of the datasets.

REFERENCES

[1] Bogdanova, D., Foster, J., Dzendzik, D., and Liu, Q. (2017). If You Can't Beat Them Join Them: Handcrafted Features Complement Neural Nets for Non-Factoid Answer Reranking. In *HLT-EACL*, pages 121–131, Valencia, Spain. Association for Computational Linguistics.

[2] Cer, D., Yang, Y., Kong, S.-y., Hua, N., Limtiaco, N., John, R. S., Constant, N., Guajardo-Cespedes, M., Yuan, S., Tar, C., Sung, Y.-H., Strope, B., and Kurzweil, R. (2018). Universal Sentence Encoder. *arXiv:1803.11175 [cs]*. arXiv: 1803.11175.

[3] Crane, M. (2018). Questionable Answers in Question Answering Research: Reproducibility and Variability of Published Results. *Transactions of the Association for Computational Linguistics*, 6(0):241–252.

[4] Jansen, P., Surdeanu, M., and Clark, P. (2014). Discourse Complements Lexical Semantics for Non-factoid Answer Reranking. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 977–986, Baltimore, Maryland. Association for Computational Linguistics.

[5] Kingma, D. P. and Ba, J. (2014). Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 [cs]*. arXiv: 1412.6980.

[6] Lai, T. M., Bui, T., and Li, S. (2018). A Review on Deep Learning Techniques Applied to Answer Selection. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2132–2144, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

[7] Liu, Y., Rong, W., and Xiong, Z. (2018). Improved Text Matching by Enhancing Mutual Information. In *AAAI*.

[8] Min, S., Seo, M., and Hajishirzi, H. (2017). Question Answering through Transfer Learning from Large Fine-grained Supervision Data. *arXiv:1702.02171 [cs]*. arXiv: 1702.02171.

[9] Nakov, P., MÃărquez, 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 10th International Workshop on Semantic Evaluation*, pages 525–545, San Diego, California. Association for Computational Linguistics.

[10] Rajpurkar, P., Zhang, J., Lopyrev, K., and Liang, P. (2016). SQuAD: 100,000+ Questions for Machine Comprehension of Text. *arXiv:1606.05250 [cs]*. arXiv: 1606.05250.

[11] Shen, D., Wang, G., Wang, W., Min, M. R., Su, Q., Zhang, Y., Li, C., Henao, R., and Carin, L. (2018). Baseline Needs More Love: On Simple Word-Embedding-Based Models and Associated Pooling Mechanisms. *arXiv:1805.09843 [cs]*. arXiv: 1805.09843.

[12] Tayyar Madabushi, H., Lee, M., and Barnden, J. (2018). Integrating Question Classification and Deep Learning for improved Answer Selection. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3283–3294, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

[13] Wang, M., Smith, N. A., and Mitamura, T. (2007). What is the Jeopardy Model? A Quasi-Synchronous Grammar for QA. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*.

[14] Wang, S. and Jiang, J. (2016). A Compare-Aggregate Model for Matching Text Sequences. *arXiv:1611.01747 [cs]*. arXiv: 1611.01747.

[15] Yadav, V., Sharp, R., and Surdeanu, M. (2018). Sanity Check: A Strong Alignment and Information Retrieval Baseline for Question Answering. *arXiv:1807.01836 [cs]*. arXiv: 1807.01836.

[16] Yin, W., SchÃijtze, H., Xiang, B., and Zhou, B. (2015). ABCNN: Attention-Based Convolutional Neural Network for Modeling Sentence Pairs. *arXiv:1512.05193 [cs]*. arXiv: 1512.05193.

[17] Zhang, P., Hou, Y., Su, Z., and Su, Y. (2018). Two-Step Multi-factor Attention Neural Network for Answer Selection. In *PRICAI 2018: Trends in Artificial Intelligence*, pages 658–670. Springer, Cham.