Asymmetry Sensitive Architecture for Neural Text Matching

Thiziri Belkacem, Jose G. Moreno, Taoufiq Dkaki, Mohand Boughanem

Asymmetry Sensitive Architecture for Neural Text Matching

Thiziri Belkacem(✉), Jose G. Moreno, Taoufiq Dkaki, and Mohand Boughanem

IRIT UMR 5505 CNRS, University of Toulouse, Toulouse, France
{thiziri.belkacem,jose.moreno,taoufiq.dkaki,mohand.boughanem}@irit.fr

Abstract. Question-answer matching can be viewed as a puzzle where missing pieces of information are provided by the answer. To solve this puzzle, one must understand the question to find out a correct answer. Semantic-based matching models rely mainly in semantic relatedness the input text words. We show that beyond the semantic similarities, matching models must focus on the most important words to find the correct answer. We use attention-based models to take into account the word saliency and propose an asymmetric architecture that focuses on the most important words of the question or the possible answers. We extended several state-of-the-art models with an attention-based layer. Experimental results, carried out on two QA datasets, show that our asymmetric architecture improves the performances of well-known neural matching algorithms.

Keywords: Asymmetric · Attention models · Relevance matching

1 Introduction

Short text matching in general and Question Answering (QA) in particular include several problems that can be grouped into two main classes. The first one consists in identifying whether two texts are semantically similar. Current solutions are based on syntactic and semantic relatedness of the inputs. We refer to this as the *symmetric matching problem*, it includes tasks such as sentence completion [14,19] and question pairs identification [1,2]. The second is whether an input text provides the information sought in another text. In this case, the nature of input texts is not the same and their association is determined not only by the semantic relationship but also by complementarity. We refer to this as *the asymmetric matching problem* and it mainly includes question-answer matching [11,15], where the question contains some of the requested information but not the information itself. Several matching models based on convolutional neural networks (CNN) [7,9] and Long Short-Term Memory Models (LSTM) [13,14] have recently been proposed. In these models, the input sentences are first mapped to a set of word vectors, then processed in a symmetric architecture through different layers. To take into account the asymmetric aspect of the question-answer matching task, we believe that existing QA models must go

https://doi.org/10.1007/978-3-030-15719-7_8

beyond the classical symmetric text matching architecture. An ideal model must focus more on the most important words in the question to better address it. Attention-based models [3,18] can provide a way to fit this requirement. Based on the importance of words in the text, these models learn *attention* coefficients that allow subsequent processes to focus on the most important words of an input text. In this paper, we propose an attention-based architecture that allows to better handle the asymmetric aspect of the question-answer matching, such that the model gives more focus to the most important words of the question. Our main contributions are as follow:

1. We propose an *asymmetric* matching architecture to handle *asymmetric* matching problems.
2. We extend several state-of-the-art models using the proposed architecture.
3. We conduct a comparative experimental study of existing models against extended ones using our architecture.

2 Related Work

Deep neural architectures in recent text matching models are based on a siamese-like architecture [5], where both inputs undergo the same[1] type of processing. This architecture is adopted in several existing models. In [12], Shen et al. propose a C-DSSM model using a convolutional network. The C-DSSM architecture uses a *word hashing* layer, as a common function to construct embedded representations of the inputs, then stacked layers map the representations to low dimensional vectors for the matching process. In [7], the authors proposed two convolutional architectures: ARC-I and ARC-II. The first constructs a sentence representation using a sequence of convolution and pooling layers, then computes the matching score of the input sentences. ARC-II applies a series of convolution and pooling layers to a matching matrix of input word vectors. Experimental results show that ARC-II outperforms ARC-I. In [9], Pang et al. proposed the MatchPyramid model. The architecture of MatchPyramid is also symmetric: first, inputs are represented using embedded vectors then a matching tensor is computed and fed to a sequence of convolution and pooling layers, in order to extract high level interaction signals. In [14], Wan et al. propose the MV-LSTM model, a position-based model for question answering. The symmetry of MV-LSTM consists in a bidirectional LSTM (bi-LSTM) layer that constructs a position-aware representation for both input sentences. Interaction matrices are then computed and passed through a pooling and fully connected layers to compute the final matching score. Mitra et al. [8] proposed a duet architecture to match documents and queries. The duet architecture is composed of the local model that uses the interaction matrix of query and document words, and the distributed model that learns embeddings of the query and the document text before matching. The parameters of both models are optimized jointly during

[1] Some differences may exist but they are only related to the input size which is considered as a non-architectural difference.

training and the final matching score is provided by the duet architecture. Both the local and the distributed models are based on a symmetric architecture, since both the inputs are dealt with in the same way.

All the mentioned models use symmetric architectures regardless the nature of the addressed task. In both symmetric [7,8] and asymmetric [9,14] tasks, only symmetric architectures were adopted. The asymmetric aspect of inputs processing is already discussed in [4], where Bordes et al. devised an architecture to learn more than one relation at a time from a knowledge base. However, none of the previous work provide an asymmetric architecture for text matching.

3 Asymmetric Matching Architecture

Motivation

Expressed in a natural language, the question describes a specific user's information need. It is like a puzzle whose missing pieces must be found and put together, in a logical way, to solve the problem. A pattern of each gap in this puzzle describes the corresponding missing part. It is all the same for the question-answer matching: a pattern must be filled by one or more answers, not all answers can be suitable, only those that conform to the pattern (the question) that describes the missing part are able to correctly fill it (give the sought information). The example in Fig. 1 from the WikiQA dataset shows this perception.

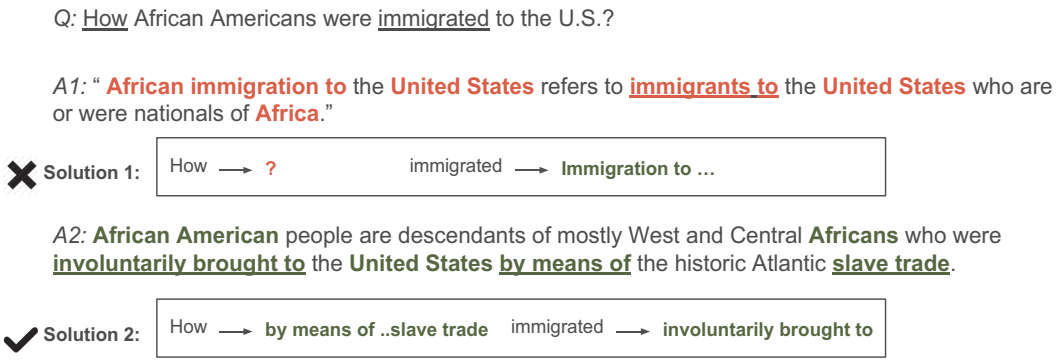

Fig. 1. In the answers *A1* and *A2*, words in **bold** represent semantic and syntactic relatedness. Underlined words are: the key words (pattern clues) in *Q* and corresponding matches in *A1* and *A2*. Solution 2 fills completely the missing part described in *Q*.

In this figure, to correctly answer the question *Q*, we have to focus on the words *"How"* and *"immigrated to"* describing the missing part. *Q* is about how the immigration process was done. We have two different answers *A1* and *A2* with corresponding solutions 1 and 2 respectively, as illustrated in Fig. 1. Based on semantic and syntactic relatedness, both the answers contain several words corresponding to the question. However, one notice that solution 2 better solves

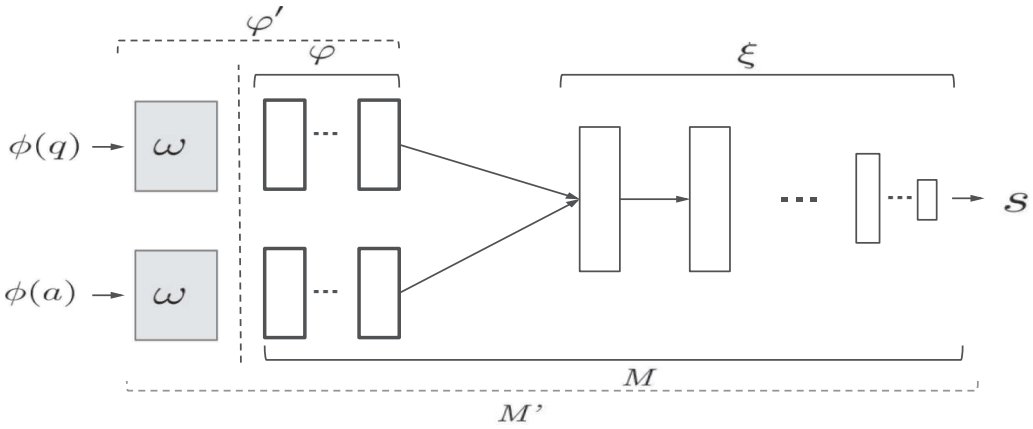

Fig. 2. The asymmetric architecture *M'* extends a model *M*. φ processes the inputs in parallel. ξ a sequence of processing layers to compute the matching score s. The attention-based layers ω are activated according to the configuration of Eq. 1.

the problem than solution 1 does. Literally, the answer *A2* contains the corresponding information, solution 2 represents the missing piece of this puzzle. This example, highlights the asymmetry aspect of the question-answer matching task. Notice that syntactic and semantic based similarities are not sufficient to solve the problem completely. The question has some words that require a particular attention to retrieve the correct answer. Attention-based models, used in machine translation [3], sentiment classification [10,17] and paraphrase identification [18], enable to identify the kernel information to be considered in a given sequence and focus on discriminating elements. In the previous example, words *"How"* and *"immigrated to"* of the question Q, require more focus. Formally, given a word sequence S, the attention-based model learns a coefficient vector α that determines how much attention should be given to each element of S according to the task to be performed. We propose an asymmetric model architecture using attention-based layers, in order to focus on most important words of the different inputs. Let us consider a question q, an answer a, a matching model M and the embedding function ϕ. Most of state-of-the-art neural-based models can be summarized in Fig. 2. Where $\phi(q)$ and $\phi(a)$ are the embedded representations of q and a respectively. M can be viewed as a sequence of two main blocks of different layers: φ is a sequence of layers used to learn input representations simultaneously. The block ξ is another sequence of different processing layers, used to compute the final matching score s. We define the asymmetric model *M'* that extends the model M by adding layer ω that can be applied differently for asymmetric inputs, as highlighted in the Fig. 2. We define a function φ' as in Eq. 1 to handle asymmetric inputs in the model *M'*. For a sequence S of l words with $S \in \{q, a\}$, we define a parameter $e_S \in \{0, 1\}$ to set up the asymmetric processing as follows: $e_S = 1$ activates the shaded layer ω in Fig. 2 for the corresponding input. If $e_S = 0$ for both the inputs then *M'* = *M*.

$$\varphi'(\phi(S)) = \begin{cases} \varphi \circ \omega(\phi(S)) & if \quad e_S = 1 \\ \varphi(\phi(S)) & otherwise \end{cases} \tag{1}$$

where ω is the extension attention-based layer as mentioned in Fig. 2. We define ω using a gating function [15], as represented in Eq. 2.

$$\omega(\phi(S)) = [w_1 \times \alpha_1, w_2 \times \alpha_2, ..., w_t \times \alpha_t, ..., w_l \times \alpha_l] \tag{2}$$

with $\alpha_t = \frac{exp(\boldsymbol{V}^T.w_t)}{\sum_{j=1}^{l} exp(\boldsymbol{V}^T.w_j)}$, where $\boldsymbol{V}$ is a model parameter. It is the attention coefficients vector of the input sentence S. w_t is a word at position t of the sentence. The layer ω is used to allow an asymmetric processing of the inputs and focusing on their most important words thanks to the attention coefficients.

4 Experiments and Results

4.1 Experimental Protocol

Experiments were performed[2] using the MatchZoo [6] framework for neural text matching models. We used two datasets. First, WikiQA Corpus [16] composed of 3047 questions from Bing query logs and 1473 candidate answers from Wikipedia. Second, QuoraQP dataset composed of 404351 question pairs. We adopted a cross-validation with 80% to train, 10% to test and 10% to validate the different models. We used a public pre-trained 300-dimensional word vectors of GloVe[3], which are trained in a Common crawl dataset. Existing and proposed models were trained using ranking hinge loss function during 400 epochs, on the WikiQA dataset and categorical cross entropy as loss function during 500 epochs, on the QuoraQP[4] dataset. We reported performances at the end of all training epochs. In both *Symmetric* and *Asymmetric* architectures, we opted by the recommended hyper-parameters configuration, either on the corresponding paper or in Matchzoo. For the C-DSSM model, we used embedded word vectors rather than the tri-letter hashing method [12] in order to compare the symmetric and asymmetric version.

4.2 Results and Discussion

Table 1 shows the performance results, in WikiQA and QuoraQP datasets, of the different models with two architecture configurations: the symmetric configuration includes the *Original* architecture of the corresponding models and their respective architecture *(Q+A)* where the attention layer ω is applied at both inputs simultaneously. The asymmetric architecture refers to the extended model where layer ω is added to one input at a time: question input *(Q)* or answer input *(A)*. Superscripts $^{\blacktriangle}$ and $^{\blacktriangledown}$ show respectively the significance[5] of

[2] The corresponding code will be available on MatchZoo and public to allow the reproducibility of the results we show in this paper.

[3] http://nlp.stanford.edu/data/glove.840B.300d.zip.

[4] The loss values of some of the models converged after more than 400 epochs in QuoraQP dataset.

[5] We performed Student's test with $P = 0.05$.

Table 1. Comparison of the symmetric and asymmetric architectures using several text matching models, in WikiQA and QuoraQP datasets. The ".ω" refers to application of layer ω with the corresponding model, as described in Fig. 2. Values in **Bold** indicates the best performances. Superscripts ▲ and ▼ refer to the significance of the results improvement and deterioration respectively.

Performance on WikiQA							
Models				MRR	ndcg@3	ndcg@5	MAP
Classical models	LM			0.5981	0.5841	0.6282	0.5932
	BM25			0.5811	0.5668	0.6203	0.5762
Neural Models	Symmetric	*Original*	ARC-II	0.5708	0.5410	0.6095	0.5606
			C-DSSM	0.5586	0.5149	0.5902	0.5451
			DUET	0.6259	0.6016	0.6561	0.6113
			MatchPyramid	0.6529	0.6442	0.6902	0.6436
			MV-LSTM	0.6215	0.6101	0.6549	0.6046
		(Q+A)	ARC-II.ω	0.5814	0.5548	0.6194	0.5743
			C-DSSM.ω	0.5622	0.5266	0.5891	0.5523
			DUET.ω	0.5982	0.5589$^{\blacktriangledown}$	0.6283	0.5801
			MatchPyramid.ω	0.4698$^{\blacktriangledown}$	0.4272$^{\blacktriangledown}$	0.5202$^{\blacktriangledown}$	0.4697$^{\blacktriangledown}$
			MV-LSTM.ω	0.5904	0.5562	0.6145	0.5562
	Asymmetric	*(Q)*	ARC-II.ω	0.5748	0.5117	0.5872	0.5465
			C-DSSM.ω	0.5222	0.4973	0.5530$^{\blacktriangledown}$	0.5134
			DUET.ω	0.6314	0.6116	0.6619	0.6158
			MatchPyramid.ω	**0.6715**	**0.6649**	**0.7068**	**0.6591**
			MV-LSTM.ω	0.6691$^{\blacktriangle}$	0.6519$^{\blacktriangle}$	0.6948$^{\blacktriangle}$	0.6507$^{\blacktriangle}$
		(A)	ARC-II.ω	0.5528	0.5627	0.6151	0.5741
			C-DSSM.ω	0.5886	0.5461	0.6190	0.5763
			DUET.ω	0.6383	0.6113	0.6679	0.6251
			MatchPyramid.ω	0.5575$^{\blacktriangledown}$	0.5360$^{\blacktriangledown}$	0.5952$^{\blacktriangledown}$	0.5502$^{\blacktriangledown}$
			MV-LSTM.ω	0.6174	0.6003	0.6590	0.6165

Accuracy on QuoraQP dataset				
Models	Symmetric		Asymmetric	
	Original	*(Q+A)*	*(Q)*	*(A)*
C-DSSM	0.670969	0.668107	0.751076	0.748548
ARC-II	0.803320	0.785819	0.786159	0.789267
MatchPyramid	**0.818289**	**0.808887**	**0.817010**	**0.815087**
MV-LSTM	0.759707	0.774055	0.790089	0.780036
DUET	0.777509	0.765360	0.767198	0.761784

the improvements and the deteriorations of the models performances. In the WikiQA dataset, the results show that for all models and metrics, at least one of the asymmetric architectures, *(Q)* or *(A)*, outperforms its symmetric counterparts, including *Original* and *(Q+A)*. Indeed, the performances obtained with the asymmetric *(Q)*-MatchPyramid.ω are the best for this dataset. Besides, the *(Q)*-MV-LSTM.ω outperforms significantly the original model MV-LSTM. Note that these results strongly support our claim about the impact of the asymmetric architectures w.r.t. the question or the answer. Even if there are not significant improvements in several models, the asymmetric architecture enable the neural models, such as CDSSM, to reach results of the classical models such as BM25. In the QuoraQP dataset, the symmetric task does not benefit of the asymmetric architecture given the symmetric nature of the question-question matching. There are no significant improvements over the asymmetric architectures compared to the original. We retained from this analysis that the asymmetric aspect of the inputs has an important impact on the matching process. By consequence, matching models must adapt their architectures to the nature of the task. Note

that we carried additional investigations and we figured out that the asymmetric architecture performs differently w.r.t. question type (*what*, *who*, ...). Results are omitted due to paper size limitation.

5 Conclusion

In this paper we proposed an asymmetric architecture for asymmetric matching tasks. We used an attention layer to extend several state-of-the-art models and construct the corresponding asymmetric architectures. Experiments in two different QA datasets showed promising results of the asymmetric architecture as compared to the symmetric one. We conclude that when the model performs an asymmetric matching task, our architecture enables to acknowledge the asymmetric aspect and provide better results. Since there were no significant differences with some experimented models between the original and extended versions, the up coming work will involve the use of additional datasets to confirm the importance of the asymmetric architecture. Our work opens a new perspective for future research and will focus our attention on how to make a neural model automatically adapt to the nature of the task being addressed.

References

1. Abishek, K., Hariharan, B.R., Valliyammai, C.: An enhanced deep learning model for duplicate question pairs recognition. In: Nayak, J., Abraham, A., Krishna, B.M., Chandra Sekhar, G.T., Das, A.K. (eds.) Soft Computing in Data Analytics. AISC, vol. 758, pp. 769–777. Springer, Singapore (2019). https://doi.org/10.1007/978-981-13-0514-6_73
2. Addair, T.: Duplicate question pair detection with deep learning. Stanf. Univ. J. (2017)
3. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473 (2014)
4. Bordes, A., Weston, J., Collobert, R., Bengio, Y.: Learning structured embeddings of knowledge bases. In: Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2011, pp. 301–306. AAAI Press (2011). http://dl.acm.org/citation.cfm?id=2900423.2900470
5. Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., Shah, R.: Signature verification using a "Siamese" time delay neural network. In: Advances in Neural Information Processing Systems, pp. 737–744 (1994)
6. Fan, Y., Pang, L., Hou, J., Guo, J., Lan, Y., Cheng, X.: MatchZoo: a toolkit for deep text matching. arXiv preprint arXiv:1707.07270 (2017)
7. Hu, B., Lu, Z., Li, H., Chen, Q.: Convolutional neural network architectures for matching natural language sentences. In: Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N.D., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems 27, pp. 2042–2050. Curran Associates, Inc. (2014). http://papers.nips.cc/paper/5550-convolutional-neural-network-architectures-for-matching-natural-language-sentences.pdf

8. Mitra, B., Diaz, F., Craswell, N.: Learning to match using local and distributed representations of text for web search. In: Proceedings of the 26th International Conference on World Wide Web, WWW 2017, pp. 1291–1299. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland (2017). https://doi.org/10.1145/3038912.3052579
9. Pang, L., Lan, Y., Guo, J., Xu, J., Wan, S., Cheng, X.: Text matching as image recognition. In: AAAI, pp. 2793–2799 (2016)
10. Parikh, A.P., Täckström, O., Das, D., Uszkoreit, J.: A decomposable attention model for natural language inference. arXiv preprint arXiv:1606.01933 (2016)
11. Peng, Y., Liu, B.: Attention-based neural network for short-text question answering. In: Proceedings of the 2018 2nd International Conference on Deep Learning Technologies, ICDLT 2018, pp. 21–26. ACM, New York (2018). https://doi.org/10.1145/3234804.3234813
12. Shen, Y., He, X., Gao, J., Deng, L., Mesnil, G.: Learning semantic representations using convolutional neural networks for web search. In: Proceedings of the 23rd International Conference on World Wide Web, WWW 2014 Companion, pp. 373–374. ACM, New York (2014). https://doi.org/10.1145/2567948.2577348
13. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N.D., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems 27, pp. 3104–3112. Curran Associates, Inc. (2014). http://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf
14. Wan, S., Lan, Y., Guo, J., Xu, J., Pang, L., Cheng, X.: A deep architecture for semantic matching with multiple positional sentence representations. AAAI **16**, 2835–2841 (2016)
15. Yang, L., Ai, Q., Guo, J., Croft, W.B.: aNMM: ranking short answer texts with attention-based neural matching model. In: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, pp. 287–296. ACM (2016)
16. Yang, Y., Yih, W., Meek, C.: WikiQA: a challenge dataset for open-domain question answering. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP). Association for Computational Linguistics, Lisbon, September 2015
17. Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., Hovy, E.: Hierarchical attention networks for document classification. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 1480–1489 (2016)
18. Yin, W., Schütze, H., Xiang, B., Zhou, B.: ABCNN: attention-based convolutional neural network for modeling sentence pairs. Trans. Assoc. Comput. Linguist. **4**(1), 259–272 (2016)
19. Zweig, G., Platt, J.C., Meek, C., Burges, C.J.C., Yessenalina, A., Liu, Q.: Computational approaches to sentence completion. In: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1, ACL 2012, pp. 601–610. Association for Computational Linguistics, Stroudsburg (2012). http://dl.acm.org/citation.cfm?id=2390524.2390609