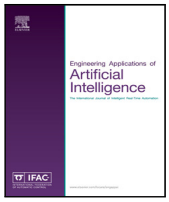
Engineering Applications of Artificial Intelligence 123 (2023) 106137

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai



Improving question answering performance using knowledge distillation and active learning

Yasaman Boreshban [a,*], Seyed Morteza Mirbostani [b], Gholamreza Ghassem-Sani [a,*], Seyed Abolghasem Mirroshandel [b], Shahin Amiriparian [c]

[a] Computer Engineering Department, Sharif University of Technology, Tehran, Iran
[b] Department of Computer Engineering, University of Guilan, Rasht, Iran
[c] Chair of Embedded Intelligence for Health Care & Wellbeing, University of Augsburg, Augsburg, Germany

ARTICLE INFO

Keywords:
Natural language processing
Question answering
Deep learning
Knowledge distillation
Active learning
Performance

ABSTRACT

Contemporary question answering (QA) systems, including Transformer-based architectures, suffer from increasing computational and model complexity which render them inefficient for real-world applications with limited resources. Furthermore, training or even fine-tuning such models requires a vast amount of labeled data which is often not available for the task at hand. In this manuscript, we conduct a comprehensive analysis of the mentioned challenges and introduce suitable countermeasures. We propose a novel knowledge distillation (KD) approach to reduce the parameter and model complexity of a pre-trained bidirectional encoder representations from transformer (BERT) system and utilize multiple active learning (AL) strategies for immense reduction in annotation efforts. We show the efficacy of our approach by comparing it with four state-of-the-art (SOTA) Transformers-based systems, namely KroneckerBERT, EfficientBERT, TinyBERT, and DistilBERT. Specifically, we outperform KroneckerBERT$_{21}$ and EfficientBERT$_{TINY}$ by 4.5 and 0.4 percentage points in EM, despite having 75.0% and 86.2% fewer parameters, respectively. Additionally, our approach achieves comparable performance to 6-layer TinyBERT and DistilBERT while using only 2% of their total trainable parameters. Besides, by the integration of our AL approaches into the BERT framework, we show that SOTA results on the QA datasets can be achieved when we only use 40% of the training data. Overall, all results demonstrate the effectiveness of our approaches in achieving SOTA performance, while extremely reducing the number of parameters and labeling efforts. Finally, we make our code publicly available at https://github.com/mirbostani/QA-KD-AL.

1. Introduction

The development of QA systems is a relatively new challenge in the field of natural language processing (NLP) (Kolomiyets and Moens, 2011). The ultimate goal of creating such systems is to enable machines to comprehend text as well as, or even better than, human beings (Zhang et al., 2019; Dunietz et al., 2020; Boroujeni et al., 2022; Zahedi et al., 2017). Extensive progress has been made in this area over the last few years. In QA models, context paragraphs and their corresponding questions are represented as a series of tokens (Yu et al., 2018). The objective of a QA system is to predict the correct span within a paragraph in which the answer to a given question resides. It is often the case that an attention mechanism is also used to keep the dependency relations between questions and paragraphs. Furthermore, two probability values are computed for each token, which represents the likelihood of the token being the start and end of an answer span. For each query, the system identifies the span with the highest probability value, as the answer to the query (Cheng et al., 2020).

With the increasing interest in deep neural networks (DNNs) (Wang et al., 2021b; Chen et al., 2022a; Afan et al., 2021), extensive progress has been made in many areas of real-world applications (Chen et al., 2022b; Fan et al., 2020; Banan et al., 2020). In NLP and especially in QA systems, DNNs have even reached an accuracy level that is higher than that of humans (Yang et al., 2019; Liu et al., 2019b). Nevertheless, these achievements have been made possible with the cost of building very large and expensive NLP models. Despite all the progress made, there are still several remaining challenges and issues that need to be addressed.

* Corresponding authors.
E-mail addresses: yasaman.boreshban@sharif.edu (Y. Boreshban), m.mirbostani@msc.guilan.ac.ir (S.M. Mirbostani), sani@sharif.edu (G. Ghassem-Sani), mirroshandel@guilan.ac.ir (S.A. Mirroshandel), shahin.amiriparian@uni-a.de (S. Amiriparian).
URLs: https://boreshban.github.io/ (Y. Boreshban), https://mirbostani.github.io/ (S.M. Mirbostani), http://sharif.edu/~sani/ (G. Ghassem-Sani), https://nlp.guilan.ac.ir/mirroshandel/ (S.A. Mirroshandel), https://www.uni-augsburg.de/de/fakultaet/fai/informatik/prof/eihw/team/shahin-amiriparian/ (S. Amiriparian).

https://doi.org/10.1016/j.engappai.2023.106137
Received 8 November 2022; Received in revised form 20 February 2023; Accepted 9 March 2023
Available online 29 March 2023
0952-1976/

The most critical challenges are as follows:

- These models often suffer from high complexity and low robustness problems.
- They normally require a massive amount of labeled data for training.
- The training time of these models is often excessive due to the huge number of trainable parameters.
- They are subject to extensive resource consumption for performant operation and reasonable inference time, which makes them unfit for real-world applications running on devices such as mobiles and embedded systems that have limited resources.

Highly effective deep learning-based approaches can immensely enhance the performance of distributed systems, embedded devices, and field-programmable gate array (FPGA). The use of machine learning technology in virtual and augmented reality on hardware such as smart wearable devices has brought distinct accomplishments in terms of features and capabilities. However, due to the excessive computational complexity imposed by this technology, its implementation on most portable devices is challenging and bounded by their hardware limitations. Accordingly, to address this issue, different model compression techniques have been introduced as a practical solution, which has absorbed a lot of attention over recent years (Cheng et al., 2020).

Current compression techniques can be divided into four general groups: pruning, low-rank factorization, transferred convolutional filters, and knowledge distillation (KD). Pruning removes unimportant DNN weights to produce more compact models. In low-rank factorization, we usually use the singular value decomposition (SVD) approach for decomposing a large matrix into a number of smaller ones. Transferred convolutional filters compress the size of a model using special structural convolution filters. Finally, by the KD technique, we train a small student model by mimicking a large teacher model to reproduce similar outputs. It has been suggested that among these methods, KD can result in a more significant improvement in terms of compression rate, speed-up, and accuracy level (Oguntola et al., 2018). Accordingly, we have decided to study the impact of KD on the QA task.

Another concerning issue entangled with DNNs is the robustness deficiency. Although employing DNNs in NLP models has led to impressive results on multiple downstream tasks, these models are not robust enough and are extremely vulnerable to adversarial examples (Jin et al., 2020; Ebrahimi et al., 2018; Ren et al., 2019; Li et al., 2020a). For QA tasks, it has been demonstrated that an intentional perturbation of a paragraph through including adversarial sentences confuses even the best available QA models, causing a severe reduction of their accuracy (Jia and Liang, 2017). This vulnerability against adversarial examples also makes these models unsuitable for real-world scenarios. Consequently, numerous studies addressing this issue have been conducted to increase the robustness of the proposed models (Le et al., 2022; Ivgi and Berant, 2021; Bao et al., 2021).

Recent accomplishments in DNNs have been heavily dependent on the use of large training datasets; conversely, DNNs are inefficient when trained on small datasets (Dhingra et al., 2018); however, the number of available annotated corpora is inadequate, and manual annotation is a costly procedure. Moreover, for some languages, the required amount of annotated datasets are unavailable. In recent years, there has been a limited number of studies conducted on unsupervised, semi-supervised, and AL for QA systems (Yang et al., 2017; Dhingra et al., 2018; Chung et al., 2018; Lewis et al., 2019). In this study, we introduce a novel combination of a parameter reduction technique and AL for QA systems. We show that the results of this combination are comparable to that of state-of-the-art models for this task.

To overcome the presented challenges of this research, we propose the following solutions. (1) For parameter reduction, we can utilize KD to transfer the knowledge of a large (complex) model to a condensed neural network. In other words, we train a small model in such a way that its accuracy approaches that of the initial large model. In this study, we have used multiple pre-trained models as our initial teacher model and transferred their knowledge to several smaller QA models. It was demonstrated that employing KD significantly improves the robustness and generalization of the models (Papernot et al., 2016). We have specifically investigated the impact of KD on the robustness of QA systems.

(2) We have also utilized AL to reduce the cost of data labeling. Since data annotation is an expensive task, we can employ AL strategies to obtain reasonable results with a small training dataset (Fu et al., 2013). To the best of our knowledge, AL has not yet been applied to the task of QA. We have used several strategies to select informative unlabeled data samples with the ability to transfer more information to the model. As a result, we substantially reduced the required number of samples and their labeling costs for training the model.

(3) By combining KD and AL methods, we can build a model with a significantly reduced number of parameters and required labeled samples. The resultant model is capable of achieving comparable results to that of state-of-the-art models.

The contributions of this paper are as follows:

- We analyze the impact of KD on various student models and QA datasets for reducing computational complexity.
- For the first time, the AL approach is applied to QA to reduce the labeling costs.
- The effects of KD on the robustness of QA models is assessed.
- Based on both KD and AL methods, a new efficient approach is introduced for the QA tasks.

In this paper, we define the theoretical background of QA systems in Section 2 and introduce our related works in Section 3. We describe our proposed approaches in detail in Section 4. We give a brief description of the datasets and models used in this study and present our experimental results in Section 5. Finally, Section 6 includes our conclusions and future works.

2. Theoretical background

In this section, we introduce the architecture of QA systems and the concept of KD as a model compression technique. These are to be discussed in Sections 2.1 and 2.2, respectively. Then, in Section 2.3, we describe the AL method, which can be used to reduce the annotation costs.

2.1. Architectures of QA systems

In QA systems, as shown in Fig. 1, context paragraphs and their corresponding questions are represented as a series of tokens $P = \{p_1, p_2, p_3, \dots, p_n\}$ and $Q = \{q_1, q_2, q_3, \dots, q_n\}$, respectively (Yu et al., 2018). The goal here is to predict the answer in the form of a span within one of the context paragraphs, $A = \{p_j, \dots, p_{j+k}\}$. In such cases, the system is expected to analyze questions and context paragraphs comprehensively to find the best (i.e., the most relevant) answer. Although several different learning methods have been employed in QA systems, deep learning methods, in particular, have achieved a higher accuracy (Lakshmi and Arivuchelvan, 2019).

2.2. Knowledge distillation

KD approaches are divided into three categories: offline, online, and self-distillation. In offline distillation, only the parameters of the student model are updated; however, the parameters of the teacher model are fixed. In online distillation, the parameters of teacher and student models are updated simultaneously. Online distillation is suggested when a high-performance teacher model is unavailable. Self-distillation is another approach in which the student and teacher models are the same (Gou et al., 2021).

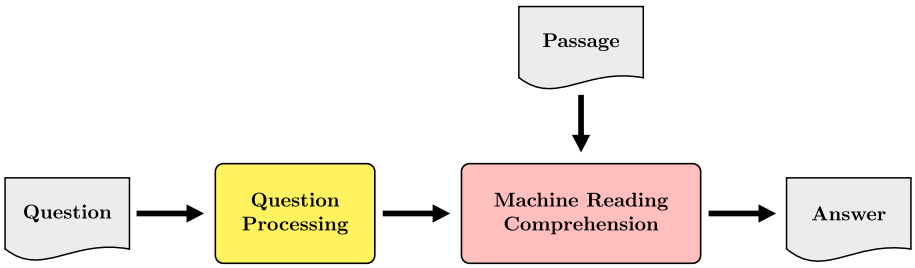


Fig. 1. The architecture of QA systems consists of question processing and machine reading comprehension modules.

Using KD, a compact neural network can be trained in such a way that we achieve the same high accuracy of a much larger network (Hinton et al., 2015). The KD architecture is composed of two components, i.e., a *student* model and a *teacher* model. The teacher component is a large model with high accuracy but with heavy computational costs and a large number of parameters. On the other hand, the student component is a compact model with a smaller number of parameters. The student model mimics the teacher's behavior. However, it is more suitable for deployment due to its much lower computational costs. To imitate the behavior of the teacher, the student, along with its own *actual labels* (*hard target*), also employs the teacher's *output logits* (*soft target*). As it follows, the loss function consists of adding hard and soft terms:

$$L = (1-\rho)\cdot C_{\text{hard}}(\boldsymbol{x},\boldsymbol{y}) + \rho\cdot C_{\text{soft}}(\boldsymbol{x},\boldsymbol{q}), \tag{1}$$

$$C_{\text{hard}}(\boldsymbol{x},\boldsymbol{y}) = -\sum_{i=1}^{K} y_i \log p_i(\boldsymbol{x}), \tag{2}$$

$$C_{\text{soft}}(\boldsymbol{x},\boldsymbol{q}) = -\sum_{i=1}^{K} q_i \log p_i(\boldsymbol{x}), \tag{3}$$

where C_{hard} is the cross-entropy (CE) loss function of the student model and C_{soft} is applied to the softmax of the output of both models. ρ is the weight of the hard and soft cross-entropy losses. K is the number of output classes of x. $p_i(x)$ is the softmax output probability of the ith class of the student. The hard target y is a one-hot K-dimensional vector. q is a soft target, which is a K-dimensional vector. q_i is the *tempered softmax probability* for ith class of the teacher model, which is computed as follows (Asami et al., 2017):

$$q_i = \frac{exp(z_i(\boldsymbol{x})/T)}{\sum_{j=1}^{K} exp(z_j(\boldsymbol{x})/T)}, \tag{4}$$

where $z_i(\boldsymbol{x})$ is the pre-softmax output of the teacher model for the ith class. T is the *temperature*. When T is large, the class probability distribution will be uniform. In other words, q is a smooth probability distribution containing the correct class information and between-class similarity. Temperature T controls the importance of the class similarity information during training. When T is greater than 1, small probabilities of non-target classes are emphasized; in that case, the student learns class similarity information more accurately (Hinton et al., 2015).

2.3. Active learning

AL is a learning method that aims at minimizing the annotation costs without sacrificing the accuracy (Fu et al., 2013). The main purpose of this approach is that if the training algorithm is able to choose more informative data during the learning process, the model can reach almost the same accuracy as a supervised method with a much less amount of data. AL approaches are classified into three major categories of *membership query synthesis*, *stream-based selective sampling*, and *pool-based sampling*.

In membership query synthesis, new instances are generated for which an omniscient expert is expected to provide the ground-truth labels. However, those instances may not have a natural distribution, making the annotation difficult even for a human (Settles, 2009). Selective sampling is an alternative approach for synthesizing queries. This approach is also called stream-based (or sequential) AL. Here, unlabeled instances are firstly sampled by the actual distribution. Then it is decided if the samples should be labeled based on their value (Settles, 2009). The pool-based sampling approach is based on the assumption that we have a small set of labeled and an enormous pool of unlabeled data. The best candidates (i.e., the most informative ones) are selected from the pool by different selection criteria, annotated by an oracle, and added to the labeled dataset. The training process is repeated every time that some labeled samples are added to the training set (Settles, 2009; Amiriparian et al., 2017). All AL strategies must measure the usefulness of unlabeled data based on some specified criteria, among which the most popular one is the *uncertainty measure*.

3. Related works

In this section, we first review the conventional and contemporary machine learning methods for QA systems in Section 3.1. Then, we compare various KD and AL approaches in Sections 3.2 and 3.3, respectively.

3.1. Deep learning-based QA models

In recent years, with the popularity of neural networks, an increasing number of end-to-end QA models based on long short-term memory (LSTM) (Wang and Jiang, 2017; Seo et al., 2017; Min et al., 2018; Chen et al., 2017), gated recurrent unit (GRU) (Yu et al., 2016), and reinforcement learning methods (Hu et al., 2018a) have been introduced with promising results.

In 2016, dynamic chunk reader was presented (Yu et al., 2016). It was able to extract varied length answers; whereas, its predecessor models returned one word or a named entity as the answer for each question. bi-directional attention flow (BiDAF) (Seo et al., 2017) and Match-LSTM (Wang and Jiang, 2017) are two widely used models. BiDAF employs LSTM and bidirectional attention flow networks, and Match-LSTM consists of LSTM and Pointer Net (Vinyals et al., 2015). QANet is a model which uses convolutional neural network (CNN) instead of a recurrent architecture (Yu et al., 2018). It was proposed in 2018. The encoder structure in QANet consists of a *convolution*, a *self-attention*, and a *feedforward* layer.

With the introduction of *Transformers* in 2017 (Vaswani et al., 2017), significant improvements have emerged in deep learning-based models. Instead of using recurrent neural networks (RNNs) or CNNs, a self-attention mechanism has been used to increase parallelism. Transformers are encoder–decoder-based models that heavily rely on the self-attention mechanism. bidirectional encoder representations from transformer (BERT) is an extremely popular model, initially released in late 2018 (Devlin et al., 2019). Using bidirectional Transformer

encoders, BERT was unsupervised and pre-trained on the tasks of masked language modeling (MLM) and next next sentence prediction (NSP). It has the capability of being fine-tuned on a variety of downstream tasks. Thereafter, various BERT-based models such as a lite bert (ALBERT) (Lan et al., 2020), robustly optimized BERT approach (RoBERTa) (Liu et al., 2019b), SpanBERT (Joshi et al., 2020), and LinkBERT (Yasunaga et al., 2022) have been proposed, which improved the accuracy and performance of QA models.

3.2. Knowledge distillation

It was shown that KD can improve the models' generalization and robustness (Papernot et al., 2016). For instance, using this technique in a QA system, the knowledge was transferred from an ensemble teacher to a single student model (Hu et al., 2018b). The reinforced mnemonic reader (RMR) is a base model in which attention and reinforcement learning have been integrated (Hu et al., 2018a). A two-stage KD strategy with multiple teachers was used for web QA systems (Yang et al., 2020). These two stages are *pre-training* and *fine-tuning*. The results of this study showed that this method is performant in generalization. Self-knowledge distillation (SKD) was used in Hahn and Choi (2019). As it was mentioned before, in KD, the knowledge is normally transferred from a large (teacher) model to a small (student) model. However, in SKD, the source of the knowledge is the student model itself. The results of applying KD methods in a study conducted on dialogue systems (Arora et al., 2019) with a dataset named Holl-E demonstrate that imitating the behavior of the teacher model has a significant impact on the student's performance.

Recently, some studies have focused on KD using the BERT model as the teacher. The main objective is to create a compact pre-trained (student) model with much fewer parameters and much less inference time than that of the BERT model, but at the same time with competitive accuracy. DistilBERT was presented in 2019 (Sanh et al., 2019). It was demonstrated that using the DistilBERT model, the BERT's size can be reduced by 40% while preserving 97% of its language comprehension capabilities (Sanh et al., 2019). TinyBERT is another $\text{BERT}_{\text{BASE}}$ model created by KD (Jiao et al., 2020). It has gained 96.8% performance of $\text{BERT}_{\text{BASE}}$ applied to the general language understanding evaluation (GLUE) benchmark. In another study (Sun et al., 2019a), KD was used to transfer knowledge from the BERT model as the teacher to a student model. In this work, intermediate layers and the output of the last layer were used as the medium of transferring knowledge.

Sun et al., 2020 proposed MobileBERT, which distills $\text{BERT}_{\text{LARGE}}$ with a bottleneck structure into a 24-layer slimmed version. Wang et al., 2021a proposed a new self-attention distillation. They investigated the effect of a teacher assistant between a large teacher model and a slimmed student model. In another research, multiple intermediate layers have been used besides the output layer for knowledge distillation (Sun et al., 2019b).

Tahaei et al., 2022 employed the combination of KD and Kronecker decomposition to obtain improved results by compressing the embedding, multi-head attention, and the feed-forward network layers. Besides, the combination of quantization and KD was used as a model compression approach. The resultant model achieved a significant compression ratio with only slightly decreased accuracy using (Li et al., 2022b). In another research, a compact BERT model called EfficientBERT was built by joining neural architecture search (NAS) and KD (Dong et al., 2021). In LadaBERT (Mao et al., 2020), three different compression methods, matrix factorization, weight pruning, and knowledge distillation, were combined. Rashid et al., 2021 proposed zero-shot KD that learns the output distribution of the teacher by out-of-domain data and adversarial training. Furthermore, KD has also achieved promising results in some other concepts such as multi-task learning (Clark et al., 2019; Liu et al., 2019a).

We summarize the related works on applying KD to QA in Table 1. We have categorized these works based on the employed teacher and student models, the number of parameters of the resultant model, the speed-up level, distillation types, QA datasets, knowledge sources, and distillation losses. It is worth mentioning that L_2, L_{CE}, and L_{KL} indicate L_2-norm, cross-entropy loss function, and Kullback–Leibler divergence error function, respectively.

3.3. Active learning

AL has been widely used in different subtasks of NLP. For example, in some research studies focused on named entity recognition (NER), AL was applied to a deep learning structure (Shen et al., 2017b; Li et al., 2022a). In this research (Shen et al., 2017b), the model used two CNNs for encoding characters and words, in addition to an LSTM network as a decoder. The results showed, with the aid of AL and merely one-fourth of the training dataset, the model achieved 99% accuracy of the best deep learning models trained on the whole dataset. In Liu et al. (2020), using the BERT-CRF model, an uncertainty-based AL strategy was applied to NER and achieved satisfactory results. In another research, active learning has been applied to NER in Persian (Jalali Farahani and Ghassem-Sani, 2021).

In the past few years, some researchers have employed unsupervised (Lewis et al., 2019; Chung et al., 2018; Momtazi, 2018) and semi-supervised learning (Yang et al., 2017; Dhingra et al., 2018) to tackle QA tasks. However, the combination of AL and deep learning has not been tried on such tasks before. Whereas, this combination has been successfully used for other text processing problems such as coreference resolution (Li et al., 2020b), entity resolution (Kasai et al., 2019), machine translation (Liu et al., 2018), dependency parsing (Mirroshandel and Nasr, 2011), sentiment analysis (Kranjc et al., 2015), search engines (Chen et al., 2020), unbalanced datasets (Aggarwal et al., 2020), and black box attack (Li et al., 2018).

4. Proposed approaches

We propose an interpolated KD method to transfer knowledge to the model and reduce its complexity, and AL strategies to minimize the labeled data requirement. We combine these two approaches to build a small model that gains the high accuracy of a complex model trained on a large corpus, using only a small training dataset. Our approaches are explained in detail in Sections 4.1 and 4.2.

4.1. Knowledge distillation for QA

Pre-trained models such as BERT have achieved outstanding results in several NLP tasks (Devlin et al., 2019). However, available DNNs are computationally regarded as being extremely complex, which makes them unfit for practical applications. Our proposed approach to tackle this issue in QA systems is to apply offline KD methods to such networks. The proposed model structure is depicted in Fig. 2. The formulations are as follows:

$$L = (1-\rho)\cdot C_{\text{hard}} + \rho \cdot C_{\text{soft}} \tag{5}$$

$$C_{\text{hard}} = \sum_{i=1,2} \text{CE}(\text{softmax}(\boldsymbol{\beta}^i), \boldsymbol{y}^i) \tag{6}$$

$$C_{\text{soft}} = T^2 \sum_{i=1,2} \text{KL}(\boldsymbol{p}^i, \boldsymbol{q}^i) \tag{7}$$

$$\boldsymbol{q}^i = \text{softmax}(\boldsymbol{\alpha}^i / T) \tag{8}$$

$$\boldsymbol{p}^i = \text{softmax}(\boldsymbol{\beta}^i / T) \tag{9}$$

The employment of KD in this work has been done as follows:

In standard QA models, the cross-entropy loss function is based on Eq. (10). This term is shown as C_{hard} in Fig. 2.

$$L_{\text{CE}} = -\sum_{k=1}^{m}\sum_{l=1}^{m} \boldsymbol{y}_k^1 \cdot \log p^1(k) + \boldsymbol{y}_l^2 \cdot \log p^2(l|k) \tag{10}$$

Table 1
A summary of related works on applying KD to QA. L_2, L_{CE}, and L_{KL} indicate L_2-norm, cross-entropy loss function, and Kullback–Leibler divergence error function, respectively. The letters 'E' and 'C' stand for ensemble and compact models, respectively.

Model	Teacher	Student	#Params (M)	Speed-up	Distill.	Datasets	Knowledge Source	Losses
DistilBERT Sanh et al. (2019)	$BERT_{BASE}$	$BERT_C$	66.0 52.0	2.0× 3.0×	offline	SQuAD v1.1	• softmax layer • masked language modeling • embedding layer	L_2
TinyBERT Jiao et al. (2020)	$BERT_{BASE}$	$BERT_C$	67.0 14.5	2.0× 9.4×	offline	SQuAD v1.1 SQuAD v2.0	• softmax layer • hidden layer states • attention distillation • embedding layer	L_2 L_{CE}
MobileBERT Sun et al. (2020)	$BERT_{LARGE}$	$BERT_C$	25.3 15.1	–	offline	SQuAD v1.1 SQuAD v2.0	• intermediate layer • masked language modeling	L_2 L_{KL}
Hu et al. (2018b)	RMR_E	RMR	82.8	–	online	SQuAD v1.1 SQuAD v2.0 NarrativeQA Adv. SQuAD	• softmax layer • attention distillation	L_2 L_{CE}
MKDM Yang et al. (2020)	$BERT_E$	$BERT_C$	137.7	–	offline	DeepQA	• softmax layer	L_{CE}
MiniLMv2 Wang et al. (2021a)	$BERT_{BASE}$ $BERT_{LARGE}$ $RoBERTa_{LARGE}$	$BERT_C$	66.0 66.0 81.0	2.0× 2.0× 2.0×	offline	SQuAD v2.0	• attention distillation	L_{KL}
BERT-PKD Sun et al. (2019b)	$BERT_{BASE}$	$BERT_C$	67.0 52.2	2.0× 3.0×	offline	RACE	• softmax layer • intermediate layer	L_{CE}
KroneckerBERT Tahaei et al. (2022)	$BERT_{BASE}$	$BERT_C$	14.3 5.2	7.8× 21.0×	offline	SQuAD v1.1 SQuAD v2.0	• attention distillation • intermediate layer • Embedding Layer	L_2
DQ-BART Li et al. (2022b)	$BERT_{BASE}$	$BERT_C$	–	–	offline	ELI5	• softmax layer • intermediate layer • attention distillation	L_2
$EfficientBERT_{TINY}$ Dong et al. (2021)	$BERT_{BASE}$	$BERT_C$	15.7 9.4	4.4×	offline	SQuAD v1.1 SQuAD v2.0	• softmax layer • intermediate layer • attention distillation • embedding layer	L_2 L_{CE}
LadaBERT Mao et al. (2020)	$BERT_{BASE}$	$BERT_C$	14.6 11.0	–	offline	QNLI	• softmax layer • intermediate layer • attention distillation • embedding layer	L_2

y^1 and y^2 are one-hots for the start and end answer tokens. m is the length of the paragraph. To apply KD, Kullback–Leibler (KL) divergence error function is added to the cross-entropy error function, according to Eq. (11). This term is shown as C_{soft} in Fig. 2.

$$L_{KD} = \mathrm{KL}(p \parallel q) = -\sum_{k=1}^{m}\sum_{l=1}^{m} p^1(k)\cdot\log[p^1(k)/q^1(k)] + p^2(l|k)\cdot\log[p^2(l|k)/q^2(l|k)] \quad (11)$$

q is the probability distribution of the start and end of the answer, which is extracted from the teacher model. Additionally, log-of-softmax is used to compute p and q.

In some cases, the tokenizers of the student model and that of the teacher are not the same. For instance, spaCy[1] and WordPiece (Devlin et al., 2019) are the two different tokenizers used by QANet and BERT, respectively. As a result, the tokenized outputs of these tokenizers often do not exactly match. For instance, a tokenizer may return the word "international" as just one token, whereas a different tokenizer may identify this word as two tokens, e.g., for "inter" and "national" sub-words. In general, to apply the KD loss function, the outputs of the two models must have the same number of tokens. To overcome this problem, we perform the following technique. Whenever the two models have different tokenizers, we look for partially matched tokens in an interpolation-based approach. In these cases, either a token that represents a sub-word S of a word W (i.e., usually the very first sub-word of W) in the student model matches with a substring of a token in the teacher model, or vice versa. In both cases, the information (i.e., the logits) of the token S is exactly retained. This information is also interpolated with that of the other sub-words of W to prevent any loss of information. For instance, in the case of having two tokens for "inter" and "national", the logits of the token "inter" is exactly retained and then interpolated with that of "national". For interpolation, an mean squared error (MSE) loss function is added to the main loss function to minimize the distance between the student vector and that of the teacher. The new loss function is as follows:

$$L = (1-\rho)\cdot C_{hard} + \rho\cdot C_{soft} + \mathrm{MSE}(\mathrm{stdt}_{intrpl}, \mathrm{tchr}) \quad (12)$$

4.2. Active learning for QA

AL is an efficient way to reduce the required time for creating a training labeled dataset. As was mentioned in Section 3, all prior related studies on AL were not particularly dedicated to the QA task. Accordingly, in this work, we have focused on the impact of different AL strategies on this particular task and proposed a few new formulations. Due to the advantages of the pool-based methods, versus stream-based selective sampling and membership query synthesis, we have chosen to apply the pool-based AL method. However, one can alternatively apply some other pool-based methods such as term frequency-inverse document frequency (TF-IDF) and bayesian networks (Ren et al., 2021).

As it is shown in Algorithm 1, at first, all samples of the training dataset are considered to be unannotated. Then, one percent of the dataset is selected to be used for training the model. In this experiment, the chosen model is $BERT_{BASE}$ which is trained for two epochs. Then, 10% of the rest of the unlabeled dataset is selected to be added to the current training dataset using the following strategies. This procedure continues until all unlabeled samples are exhausted.

Most data sampling strategies are based on some uncertainty criteria. Next, we describe the strategies that we have used in this work.

[1] https://spacy.io

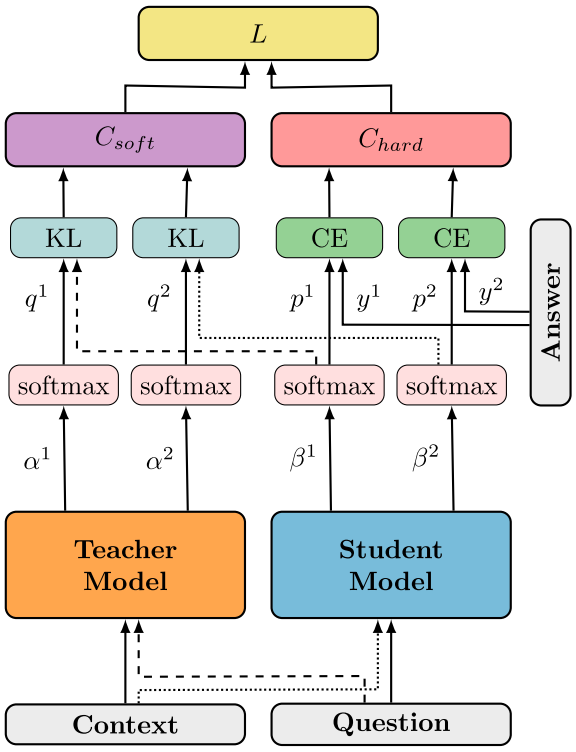


Fig. 2. In our proposed model structure that employs an interpolated KD, C_{soft} is the KL divergence error function that transfers knowledge from teacher to student. C_{hard} is the CE loss function of the student model. L is the weighted combination of C_{soft} and C_{hard}.

Algorithm 1: Pool-based AL approach

Input: Unlabeled data pool $\mathcal{U}$, labeled data set $\mathcal{L}$, most informative unlabeled samples $\boldsymbol{x}^*$, AL sampling $\phi(\cdot,\cdot)$, random sampling $\psi(\cdot,\cdot)$

$\boldsymbol{x}^* \longleftarrow \arg\max_{*x\in\mathcal{U}} \psi(\boldsymbol{x}, 1\%)$;
$\mathcal{L} \longleftarrow label(\boldsymbol{x}^*)$;
$\mathcal{U} \longleftarrow \mathcal{U} \setminus \boldsymbol{x}^*$;
repeat
  $train\_model(\mathcal{L})$;
  $\boldsymbol{x}^* \longleftarrow \arg\max_{*x\in\mathcal{U}} \phi(\boldsymbol{x}, 10\%)$;
  $\mathcal{L} \longleftarrow \mathcal{L} \cup label(\boldsymbol{x}^*)$;
  $\mathcal{U} \longleftarrow \mathcal{U} \setminus \boldsymbol{x}^*$;
until $|\mathcal{U}| = 0$;

4.2.1. Least confidence

The most widely used strategy of AL is the *least confidence sampling* (Settles, 2009). The algorithm selects those instances that have the least confidence (i.e., based on our model) for labeling. This method can be simply employed in probabilistic models. For example, in a probabilistic binary classification model, instances with a probability value around 0.5 are the ones in which the model has the least confidence.

The output of the QA systems that we are interested in is a span extending from the start to the end of the answer tokens. For each question, the model returns multiple answer spans, among which the span with the highest probability value will be selected. In each cycle, a fixed number (e.g., 10%) of questions whose selected answer has the least probability value are selected. The calculations are performed using Eqs. (13) and (14).

$$\boldsymbol{x}^* = \arg\max_{\boldsymbol{x}} \left[1 - p(\hat{A}|\boldsymbol{x})\right] \tag{13}$$

$$\hat{A} = \arg\max_{A} p(A|\boldsymbol{x}) \tag{14}$$

A is the answer set returned by the model for a question. For each instance, $\boldsymbol{x}$, $\hat{A}$ is the answer with the highest probability value given by the model. In this approach, the selected answer with the least probability value is chosen as the least confident instance, denoted by $\boldsymbol{x}^*$. This instance is presumed to contain the highest information content of all.

4.2.2. Margin

Another option that can be used for data sampling is the *margin* criterion. In this method, the probability difference between the two most probable labels is calculated. This difference shows that samples with a larger margin are easier to be classified by the model. That is because the classifier is more confident about those labels. Conversely, the classifier is less confident about those labels that have a smaller margin; therefore, knowing the actual label of such instances helps the model discriminate them more effectively. For applying this criterion to QA systems, the difference between the two most probable answers returned for each question is taken as the margin. This margin is calculated by Eq. (15), in which $\hat{A}_1$ and $\hat{A}_2$ respectively denote the first two most probable answer to question $\boldsymbol{x}$. Here, in each AL cycle, a subset of questions with the highest margin, denoted by $\boldsymbol{x}^*$, are selected to be added to the training dataset.

$$\boldsymbol{x}^* = \arg\max_{\boldsymbol{x}} \left[p(\hat{A}_1|\boldsymbol{x}) - p(\hat{A}_2|\boldsymbol{x})\right] \tag{15}$$

4.2.3. Entropy

When there exist a large number of labels, the margin sampling method practically ignores many labels. In such cases, it only considers the first two labels. In this situation, the sampling method based on *entropy*, which is calculated by Eq. (16), is more suitable for detecting uncertainty. $\hat{A}_i$ denotes the ith most probable answer returned for question $\boldsymbol{x}$.

$$\boldsymbol{x}^* = \arg\max_{\boldsymbol{x}} \left[-\sum_i p(\hat{A}_i|\boldsymbol{x}) \log\left(\hat{A}_i|\boldsymbol{x}\right)\right] \tag{16}$$

For applying this method to QA systems, the first five most probable answers for each question are selected as the candidate answers by the BERT model. After calculating the entropy for these candidates, the samples with the highest entropy are selected to be added to the training dataset.

4.2.4. Clustering method

Clustering is another approach used in our study for data sampling. For this purpose, first, some samples are selected from the unlabeled dataset pool by the least confidence approach. If k instances are to be selected for labeling, we initially choose $3 \times k$ instances based on the least confidence criterion as our candidates. Then, for clustering, questions are encoded with the universal sentence encoder (USE) (Cer et al., 2018), and using the k-means algorithm and based on the Euclidean distance measure, those candidates will be grouped into 10 clusters. To select final k samples, each cluster is sampled proportional to the number of its members. Selected instances are annotated and added to the current labeled dataset. Then the model is re-trained on the resulting dataset. This procedure continues until our unlabeled data are exhausted.

4.3. Joint knowledge distillation and active learning for QA

In this work, our main objective has been improving the performance of QA models. To achieve this goal, we have tried to both reduce the model complexity and decrease the cost of labeling. Accordingly, we have jointly used both KD and AL techniques. The details of the proposed approach are shown in Fig. 3.

5. Experiments

In this section, to assess the performance of our proposed approaches, we explain the experiments we have conducted and analyze their results in detail.

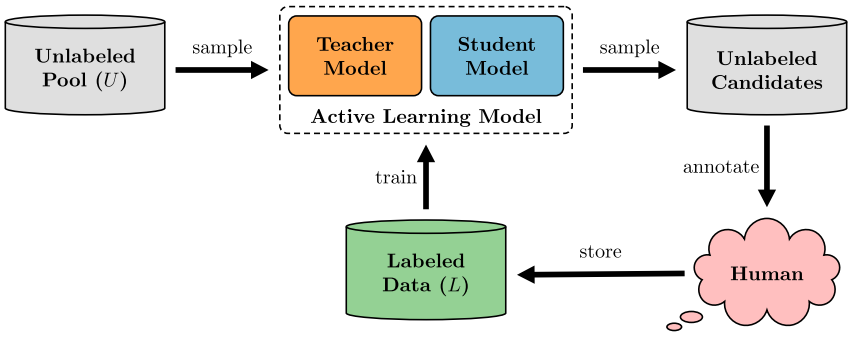


Fig. 3. The joint KD and AL model.

Table 2
Statistics of the SQuAD v1.1 v1.1, NewsQA, TriviaQA, and Adversarial SQuAD datasets.

Dataset	Train	Validation	Test
SQuAD v1.1	87,599	10,570	–
NewsQA	92,549	5,166	5,126
TriviaQA	87,622	11,313	10,832
Adversarial SQuAD: ADDSENT	–	3,560	–
Adversarial SQUAD: ADDONESENT	–	1,787	–

5.1. Datasets

There exist several datasets that can be used for the purpose of experimenting on QA models. Some of these datasets are NarrativeQA (Kočiskỳ et al., 2018), HotpotQA (Yang et al., 2018), WikiQA (Yang et al., 2015), Stanford question answering dataset (SQuAD) v1.1, NewsQA, TriviaQA, and Adversarial SQuAD. Among these, SQuAD v1.1, NewsQA, TriviaQA, and Adversarial SQuAD, which are depicted in Table 2, have been more frequently used. Accordingly, to be able to compare with more related works, we have chosen to use these four datasets in our research. Next, we explain the detailed characteristics of these datasets.

SQuAD. The SQuAD v1.1, released in 2016 (Rajpurkar et al., 2016), contains 107,785 question–answer pairs on 536 articles extracted from Wikipedia. In SQuAD v1.1, the answer to each question is a span of the text from the corresponding reading passage. This dataset has provided the ground for significant progress in building more accurate QA systems in recent years.

NewsQA. This dataset was introduced in 2016 (Trischler et al., 2017). It contains 10K articles and 100K questions and answers extracted from CNN articles.

TriviaQA. This dataset was created in 2017 based on Wikipedia documents (Joshi et al., 2017) and contains 95K questions and answers. These questions are more complex than that of SQuAD.

Adversarial SQuAD. In 2017, an adversarial dataset was built on top of SQuAD v1.1 (Jia and Liang, 2017). Its training set has remained unchanged; however, some paragraphs of its validation set have been extended by some adversarial sentences. This dataset provides three different methods for generating adversarial sentences. The first method is called ADDSENT, in which a few adversarial sentences analogous to the given question are generated and appended to the paragraph that contains the answer to that question. In the second method, called ADDONESENT, a fixed sentence is added to all paragraphs. In the last approach, named ADDANY, adversarial sentences are generated and appended to all paragraphs, regardless of grammatical considerations. Therefore, using this dataset, the robustness of QA models can be evaluated.

5.2. Models

In many previous works on KD, $\text{BERT}_{\text{LARGE}}$ and $\text{BERT}_{\text{BASE}}$ (Devlin et al., 2019) were used as the teacher model. Accordingly, we have used the uncased versions of these two models as our teacher model. On the other hand, there exist several smaller models that can be used as the student model. Some of these include QANet (Yu et al., 2018), BiDAF (Seo et al., 2017), Match-LSTM (Wang and Jiang, 2017), R-Net (Wang et al., 2017), reinforced mnemonic reader (Hu et al., 2017), ReasoNet (Shen et al., 2017a), and dynamic coattention network (Xiong et al., 2016). Among these alternatives, we chose to use the first three models as our student model because they have mainly different architectures. For instance, QANet does not employ any RNN architecture, whereas both BiDAF and Match-LSTM are RNN-based models. All the chosen student models are smaller than the mentioned teacher models in terms of complexity. Due to our hardware resource limitations, we confined our choice of student models to the mentioned three models.

5.3. Experimental setup

It has been suggested that for training the BERT model, the hyperparameters of the model can be set to one of the following learning rates: 2×10^{-4}, 3×10^{-4}, and 5×10^{-4} (Devlin et al., 2019). In our experiment, we set the rate to 5×10^{-4}. The maximum tokens length, which is the maximum length of the input to the model after tokenization, was set to 384. We defined the hyperparameters of the BERT model, such as the number of epochs, learning rate, and maximum token length, based on the suggestions of the original paper. We utilized the Pytorch framework for implementation. The model was trained for 30 epochs in a batch size of 14 samples. ρ parameter, the coefficient of the soft loss function, was set to 0.7, and the temperature T was set to 10.

5.4. Evaluation metrics

There exist several metrics that are often used in the NLP tasks. These include *precision*, *recall*, *F-score*, *exact match (EM)*, and *accuracy*. However, *F1-score* and *EM* are the two most common evaluation metrics that are used in QA tasks. In other words, these two metrics provide more insights into this specific task. Accordingly, we have used *F1-score* and *EM* as our evaluation measure so that we can compare our work with more related works.

To evaluate the performance of the system, its predictions and the ground truth answers are treated as a bag of tokens. The F1-score measures the average overlap between the predicted and the ground truth answers. It is computed for each given question and in the end, is averaged over all the questions. EM measures the percentage of those predictions that exactly match the ground truth answers. Here, when a prediction matches its ground truth answer, EM is set to 1; otherwise, it is set to 0 (Rajpurkar et al., 2016).

5.5. Applying knowledge distillation

To evaluate KD, we have employed three different models (i.e., QANet, BiDAF, and Match-LSTM) as our student model. As the teacher models, $\text{BERT}_{\text{BASE}}$ and $\text{BERT}_{\text{LARGE}}$ have been chosen. By adding *+KD*

Table 3
Performance of various models on SQuAD v1.1. $BERT_{BASE}$ and $BERT_{LARGE}$ are the teacher models. QANet, BiDAF, and Match-LSTM are the student models. Model names with a +KD suffix are the proposed models after applying KD. The best results, specified in bold, against QANet belong to the QANet+KD model with $BERT_{LARGE}$ as the teacher model, which outperforms both $DistilBERT_4$ and $TinyBERT_4$ and is comparable with these models with six layers. Additionally, the proposed approach outperforms $KroneckerBERT_{21}$ and $EfficientBERT_{TINY}$ while is comparable to $KroneckerBERT_8$ and EfficientBERT.

Model	Teacher Model	F1	EM
$BERT_{LARGE}$	–	93.15	86.91
$BERT_{BASE}$	–	88.34	81.00
$DistilBERT_6$	–	86.90	79.10
$DistilBERT_4$	–	81.20	71.80
$TinyBERT_6$	–	87.50	79.70
$TinyBERT_4$	–	82.10	72.70
$KroneckerBERT_8$	–	86.30	78.10
$KroneckerBERT_{21}$	–	80.50	70.70
EfficientBERT	–	85.30	77.00
$EfficientBERT_{TINY}$	–	83.60	74.80
QANet	–	80.09	71.16
QANet+KD (ours)	$BERT_{BASE}$	82.54	74.12
QANet+KD (ours)	$BERT_{LARGE}$	**83.51**	**75.20**
BiDAF	–	78.69	69.47
BiDAF+KD (ours)	$BERT_{BASE}$	78.83	69.52
BiDAF+KD (ours)	$BERT_{LARGE}$	81.08	72.06
MatchLSTM	–	72.80	63.31
MatchLSTM+KD (ours)	$BERT_{BASE}$	73.48	63.96
MatchLSTM+KD (ours)	$BERT_{LARGE}$	73.61	64.11

to the name of a model, we mean in that model, the knowledge of the student has been distilled from its teacher. For example, QANet is one of the base models used in our study, and QANet+KD is the model to which KD has been applied, by adding the KL loss function to the model. Table 3 shows the performance of various combinations of our proposed model on SQuAD v1.1 in comparison with other related models, using F1-score and EM measure. The results in all cases show an improvement in the performance of the student models after using KD over their prior state. As it is demonstrated, by applying KD to the QANet using $BERT_{BASE}$ and $BERT_{LARGE}$, F1-score has increased by 2.45 and 3.42 percentage points, and EM has increased by 2.96 and 4.04 percentage points, respectively. Applying the same approach to BiDAF improved the performance by 0.14 and 2.39 percentage points in F1-score and by 0.05 and 2.59 percentage points in EM, respectively. On Match-LSTM, F1-score has increased by 0.68 and 0.81 percentage points, and EM increased by 0.65 and 0.8, respectively.

Multiple experiments have been conducted to determine the optimum values for the temperature T and the coefficient of the soft loss function ρ. We evaluated the model performance with T set to 1, 5, 10, or 20. Additionally, we fixed the value of T while setting ρ to 0.1, 0.3, 0.7, or 0.9. The empirical optimum values of T and ρ are 10 and 0.7, respectively. Divergence of the T and ρ values from their optimum values resulted in lower performance of the model in terms of F1-score and EM.

We have used the cross-entropy loss function, which is commonly used in QA, to calculate the C_{hard} and C_{soft}. Cross-entropy loss function is normally used when the output is a probability value (i.e., the case in our task). We have also tested other available loss functions, such as L_1, L_2, and L_{KL}. However, according to the model output, those functions did not seem to suit our task (Asami et al., 2017).

In order to reduce the distance between the student vector and that of the teacher, we employed an MSE loss function, which is added to the main loss function. This function requires two vectors of the same dimension. In all cases, the total number of tokens generated by the teacher model's tokenizer is greater than or equal to the total number of tokens for the student model. An alternative is to reduce the dimension of the teacher vector by eliminating the values corresponding to the sub-words. Another method is calculating the mean or maximum value of a token containing a sub-word. However, in our case, these solutions were not suitable because of the information elimination problem. We have used three interpolation methods, i.e., linear, cubic, and quadratic, on the student vector to retain the information. Based on our experiments, in most cases, the cubic interpolation outperformed other methods.

Table 4
Number of parameters and speed comparison between student models and other distilled models on SQuAD v1.1 dataset.

Model	#Params (Millions)	Speed-up (Batches/Second)
$BERT_{BASE}$	110.0	1.0×
$DistilBERT_6$	66.0	2.0×
$DistilBERT_4$	52.2	3.0×
$TinyBERT_6$	67.0	2.0×
$TinyBERT_4$	14.5	9.4×
$KroneckerBERT_8$	14.3	7.8×
$KroneckerBERT_{21}$	5.2	21×
EfficientBERT	15.7	4.4×
$EfficientBERT_{TINY}$	9.4	–
QANet	1.3	2.0×
BiDAF	14.6	4.0×
MatchLSTM	3.0	1.4×

One of the main problems with large pre-trained language models is their intrinsic computational complexity. Accordingly, employing a smaller model with an accuracy comparable to a large fine-tuned model is more practical for real-world applications. To further investigate this issue, we compared the number of parameters and the inference time of our models with other related models. As it is shown in Table 4, not only do our student models have much fewer parameters than that of the associated teacher models, but also they have much less inference time.

As Tables 3 and 4 show, among different student architectures, QANet+KD with $BERT_{LARGE}$ teacher has had the best performance of all. Although the total number of parameters of this model is about 9% of that of the 4-Layer TinyBERT, it has improved the F1-score and EM of the 4-Layer TinyBERT by about 1.41 and 2.50 percentage points, respectively. Additionally, this model has outperformed the 4-Layer DistilBERT by 2.31 and 3.40 percentage points in F1 and EM, respectively, while using only 2.5% of the total number of parameters of DistilBERT. Our model has also achieved around 95% performance of the 6-Layer TinyBERT and DistilBERT models, using only 2% of their total number of parameters.

QANet+KD also outperforms $KroneckerBERT_{21}$ by 3.01 and 4.50 percentage points in F1 and EM, respectively while having 75% fewer number of parameters. The performance of QANet+KD, while using only 13.8% of the total number of parameters of $EfficientBERT_{TINY}$, is comparable to that of $EfficientBERT_{TINY}$ in terms of F1 and outperforms it by 0.4 percentage points in terms of EM.

Among *bootstrap*, *permutation*, *t-test*, and *z-test* methods, which are suitable for the QA task, we have used the *bootstrap resampling* technique to validate the results of our best student–teacher model (i.e., QANet+KD with $BERT_{LARGE}$). This technique is a non-parametric statistical hypothesis testing method that uses random sampling with replacement to emulate the sampling process. In non-parametric tests, it is assumed that the sampled dataset is representative of the whole population (Dror et al., 2018). We have chosen this test method because, in our experiments, we assume exactly the same assumption. The results of this validation showed a significant difference between the means of the predictions of the two models (i.e., QANet and QANet+KD). Firstly, as our sample set, 10% of the evaluation dataset, represented as X, was randomly selected and fed to both models. Considering EM as our evaluation metric, the difference between the performance of QANet before and after applying KD on X was calculated as $\delta(X)$. To determine whether the null hypothesis, $H_0 : \delta(X) \leq 0$, should be rejected, we must check whether or not $p-\text{value} < \alpha$, where

Table 5
Performance of QANet+KD on NewsQA and TriviaQA datasets. $BERT_{BASE}$ and QANet are the teacher and the student model, respectively. QANet+KD is our proposed model after applying KD.

Model	NewsQA		TriviaQA	
	F1	EM	F1	EM
$BERT_{BASE}$	73.29	60.19	61.37	55.57
QANet	69.27	55.41	52.45	47.23
QANet+KD (ours)	72.46	59.44	55.01	50.96

α is the *significance level*, and $p-$value is a *conditional probability*, based on the null hypothesis H_0. For calculating $p-$value, $\delta(X)$ should be resampled with replacement B times to create numerous $k-$sized sets, where k is the size of $\delta(X)$. Assigning α to 0.05 and B to 100000, our calculated $p-$value is 0.035 which rejects the null hypothesis and shows that the new model performance is statistically significant.

As Table 5 shows, to further investigate the utility of KD, we have also applied our proposed method to both NewsQA and TriviaQA datasets. In this set of experiments, again QANet was chosen as the student model, and $BERT_{BASE}$ was selected as our teacher model. The results show applying KD to QANet increases F1-score by 3.19 and 2.56, and EM by 4.03 and 3.73 percentage points on NewsQA and TriviaQA, respectively. Although the total number of parameters of QANet is only 1% of that of the $BERT_{BASE}$ model, QANet with the aid of KD has achieved a notable performance of 99% and 90% of that of $BERT_{BASE}$ on NewsQA and TriviaQA, respectively.

The results of our experiments demonstrate that using KD on small models increases their accuracy and performance, making them comparable to the large fine-tuned models. Moreover, the inference time of these models outperforms that of the large models, making them suitable and cost-efficient candidates for real-world applications. The extent of the improvement of the base student model in terms of its accuracy highly depends on the performance of the chosen teacher model and that of the initial student model. Besides, the larger and more accurate teacher model, the more information we get by using knowledge distillation. For instance, when we use $BERT_{LARGE}$ as the teacher model, the impact of KD on the accuracy of the student model is greater when $BERT_{BASE}$ is used as the teacher. Moreover, our experiments show that a student model with much fewer parameters than its teacher will make it incapable of extracting much information from the teacher model by KD, resulting in little accuracy improvement.

5.6. Applying active learning

We have also applied AL to the $BERT_{BASE}$ model to evaluate the impact of this technique on the volume of required labeled data and the performance of this model. The chosen values for hyperparameters of the model are as follows. The base model of our study was $BERT_{BASE}$ (uncased version), the learning rate was set to 5×10^{-4}, and the maximum token length was set to 384. Based on the suggestion in the original paper, the $BERT_{BASE}$ model was initially finetuned for only two epochs. That is because increasing the number of epochs reduces the accuracy of the model on the validation dataset (Devlin et al., 2019). In this experiment, the Pytorch framework was used for implementation. Initially, 1% of the training dataset was randomly chosen for fine-tuning the $BERT_{BASE}$ model; the remaining 99% of the training data was assumed to be unlabeled. Then, in each step, according to the sampling strategies proposed in Section 4.2, in each cycle, 10% of the remaining samples was added to the current labeled samples used for training. In each cycle, the model was again fine-tuned on the newly compiled dataset. This process was repeated until the model was fully trained on the whole dataset.

In Table 6, the impact of various selection strategies on the EM measure is demonstrated. RAND denotes the random sampling strategy, LC stands for the least confidence, EN is entropy, M denotes the margin sampling, and LC-CL is our proposed clustering method. The results of our experiments indicate that all the sampling methods that we have used outperform the random sampling strategy. Moreover, among these sampling methods, the least confidence strategy has achieved the best results. Using the least confidence strategy and only 20% of the training dataset, the model can achieve 93.83% EM of the case in which we employ the supervised method and the whole dataset. Additionally, the model can achieve 98.08% EM with only 40% of the training dataset.

Table 6
EM measure of different AL strategies on SQuAD v1.1 dataset. The best performance result, specified in bold, belongs to the LC strategy on 70% of the dataset, which outperforms the supervised method. RAND: random sampling strategy. LC: least confidence. EN: entropy. M: margin sampling. LC-CL: our proposed clustering method.

Dataset	RAND	LC	M	EN	LC-CL
1%	39.83	39.83	39.83	39.83	39.83
10%	71.98	72.04	71.32	71.65	71.74
20%	73.83	76.01	75.61	75.87	75.43
30%	76.07	77.95	77.58	77.86	77.87
40%	78.42	79.45	79.08	79.69	79.50
50%	79.16	79.82	79.04	80.02	79.70
60%	80.00	80.39	80.01	80.29	79.91
70%	80.27	**81.50**	80.55	81.09	80.77
80%	80.34	81.10	80.95	81.11	80.95
90%	80.93	81.40	81.07	81.02	81.02
100%	81.00	81.00	81.00	81.00	81.00

Table 7
F1-score measure of different AL strategies on SQuAD v1.1 dataset. The LC strategy on 70% of the dataset, specified in bold, performs better than the supervised method. RAND: random sampling strategy. LC: least confidence. EN: entropy. M: margin sampling. LC-CL: our proposed clustering method.

Dataset	RAND	LC	M	EN	LC-CL
1%	50.64	50.64	50.64	50.64	50.64
10%	79.26	81.66	81.01	80.94	80.43
20%	82.81	84.83	84.31	84.67	84.40
30%	84.73	86.36	85.83	85.97	86.01
40%	86.51	87.50	86.79	87.44	87.21
50%	86.97	87.56	86.96	87.77	87.23
60%	87.69	88.05	87.75	87.89	87.88
70%	87.92	**88.60**	88.00	88.39	88.12
80%	87.96	88.38	88.13	88.23	88.20
90%	88.12	88.56	88.25	88.27	88.24
100%	88.34	88.34	88.34	88.34	88.34

As it is shown in Table 7, using the least confidence strategy and only 20% and 40% of the training dataset, the model can respectively achieve 96.02% and 99.04% F1-score of the case in which we employ the supervised method and the whole dataset. As it can be seen in Tables 6 and 7, using 70% of the training dataset and the least confidence strategy, the model can even outperform the supervised method by 0.50 and 0.26 in terms of the EM measure and F1-score, respectively. We think this is because AL is effectively using more informative samples for training and ignoring some noisy data. To the best of our knowledge, our work is the first application of AL to the QA task.

To investigate the effectiveness of AL in the task of QA applied to NewsQA and TriviaQA, we have selected the least confidence strategy, which has achieved the best results on SQuAD v1.1. As it is shown in Table 8, similar to the case of SQuAD, we started with 1% of the data, selected 10% of the data on each iteration using the least confidence method, and added them to the training datasets. As it is shown, with only 40% of the NewsQA and TriviaQA training datasets, we have reached 93.86% and 90.27% of the supervised learning F1-score, respectively.

Based on the experiment results, AL approach with 40% of the training data can train a model to reach 90% of the same model trained using the supervised learning approach. AL can reduce the cost of labeling and can be used effectively in scenarios where the labeling process is costly. Additionally, the AL approach has not been used in

Table 8
Impact of AL on $BERT_{BASE}$ on NewsQA and TriviaQA datasets.

Model	Dataset	NewsQA		TriviaQA	
		F1	EM	F1	EM
$BERT_{BASE}$	100%	73.29	60.19	61.37	55.57
$BERT_{BASE}$	40%	68.79	53.85	55.40	49.27

Table 9
Applying both KD and AL on a single model trained on SQuAD v1.1, NewsQA, and TriviaQA datasets. The best performance results, specified in bold, belong to our joint QANet+KD+AL model that is trained on 40% of the datasets. It has reached to 98%, 91%, and 79% performance of the QANet model that is trained on 100% of the SQuAD v1.1, NewsQA, and TriviaQA datasets, respectively.

Model	Dataset	SQuAD		NewsQA		TriviaQA	
		F1	EM	F1	EM	F1	EM
$BERT_{BASE}$	100%	88.34	81.00	73.29	60.19	61.37	55.57
$BERT_{BASE}$+AL (ours)	40%	87.51	79.45	68.79	53.85	55.40	49.27
QANet	100%	80.09	71.19	69.27	55.41	52.45	47.23
QANet+AL (ours)	40%	74.77	63.68	55.31	40.23	32.03	25.99
QANet+KD+AL (ours)	40%	**79.51**	**69.92**	**63.81**	**49.87**	**41.85**	**37.06**

Table 10
Performance of our proposed models trained on SQuAD v1.1 dataset and evaluated on AddSent adversarial dataset. The evaluation results, specified in bold, show that our QANet+KD+AL which is trained on 40% of the dataset has achieved 98.40% F1-score and 96.70% EM of the QANet model that is trained on 100% of the dataset.

Model	Dataset	F1	EM
$BERT_{BASE}$	100%	53.79	48.20
$BERT_{LARGE}$	100%	67.81	63.10
QANet	100%	41.91	36.20
QANet+KD (ours)	100%	45.84	40.30
QANet+AL (ours)	40%	37.29	31.00
QANet+KD+AL (ours)	40%	**41.24**	**35.00**

QA task before. We have applied this approach to QA task for the first time.

5.7. Joint application of knowledge distillation and active learning

To examine the joint application of KD and AL to a single model (i.e., QANet), for each examined dataset, we first picked out 40% of the training dataset using the least confidence sampling method. Then, as our teacher model, $BERT_{BASE}$ was fine-tuned on the training set at hand. Next, QANet was trained on the same dataset while its knowledge was distilled using the fine-tuned $BERT_{BASE}$. The results of these experiments are shown in Table 9. As it can be seen, QANet+KD+AL has outperformed the QANet+AL model after it was trained on only 40% of the dataset. Besides, the joint QANet+KD+AL model has reached 98%, 91%, and 79% performance of QANet that is trained on 100% of the SQuAD v1.1, NewsQA, and TriviaQA datasets, respectively.

5.8. Robustness against adversarial datasets

The primary goal of our research has been to introduce an approach based on KD and AL to improve the performance of a QA baseline model. According to the given results in Table 3 regarding the impact of KD on QANet, BiDAF, and Match-LSTM models, the best performance improvement belongs to the QANet model. Therefore, we have chosen this model, i.e., QANet, for our robustness analysis experiments.

For analyzing the impact of KD on the robustness of QA models, QANet was trained and assessed on the Adversarial SQuAD dataset before and after applying KD. The best results belong to our proposed models trained on 100% and 40% of the dataset, which demonstrates the substantial impact of KD on the robustness of the models.

The best experiment results of our models on this type of adversarial dataset exhibit the strong impact of KD and AL least confidence strategy on the robustness of the models.

Table 11
Performance of our proposed models trained on SQuAD v1.1 dataset and evaluated on AddOneSent adversarial dataset. The evaluation results, specified in bold, show that our QANet+KD+AL that is trained on 40% of the dataset has achieved 99.96% F1-score and 98.85% EM of the QANet model that is trained on 100% of the dataset.

Model	Dataset	F1	EM
$BERT_{BASE}$	100%	64.80	58.00
$BERT_{LARGE}$	100%	76.92	71.70
QANet	100%	50.74	43.50
QANet+KD (ours)	100%	55.90	49.40
QANet+AL (ours)	40%	46.58	38.30
QANet+KD+AL (ours)	40%	**50.72**	**43.00**

The results of our experiments in Tables 10 and 11 show that using KD increases both F1-score and EM of the base model that is trained on 100% of SQuAD v1.1 by around 4.00 and 5.00 percentage points when it is tested on the AddSent and AddOneSent datasets, respectively. We also evaluated the performance of the model on the adversarial datasets when the model is equipped with both KD and AL. The QANet+KD+AL model has been trained on 40% of SQuAD v1.1 and sampled by the least confidence strategy. On the AddSent adversarial dataset, our model has outperformed the QANet (Base) model, trained on 40% of SQuAD v1.1, by around 4.00 percentage points in F1-score and EM. It has also achieved 98.40% F1-score of the base model that is trained on 100% of the training dataset. The evaluation of this model on the AddOneSent adversarial dataset shows that using only 40% of SQuAD v1.1, it can almost reach the same F1-score and EM as the base model that is trained on the whole training dataset.

6. Conclusions

In this paper, we have proposed a novel combination of an interpolated KD and AL for QA systems, which is comparable to state-of-the-art models in this task. We used $BERT_{LARGE}$ and $BERT_{BASE}$ as our teacher models and QANet, Match-LSTM, and BiDAF as our student models. We evaluated the models on SQuAD v1.1, NewsQA, TriviaQA, and Adversarial SQuAD databases.

- Our experiments showed that after applying KD with $BERT_{LARGE}$ to the QANet model, it outperformed the 4-layer DistillBERT by 2.31 and 3.40 percentage points in F1 and EM, respectively, while using only 2.5% of the total number of parameters of the DistilBERT model. Additionally, it reached 95% performance of the 6-Layer TinyBERT and DistilBERT models, while using only 2% of their total number of parameters. Our approach with 75% fewer number of parameters outperformed the $KroneckerBERT_{21}$ model by 3.01 and 4.50 percentage points in F1 and EM, respectively. The proposed model was also shown to be comparable with $EfficientBERT_{TINY}$ in terms of F1 and outperforms it by 0.4 percentage points in terms of EM with only 13.8% of the total number of parameters of $EfficientBERT_{TINY}$.
- With AL and using only 40% of the training data, we achieved a 99.04% F1-score of the supervised model trained on the whole dataset. Furthermore, we showed that our proposed approach further boosts the performance of QA models by reducing both the complexity of the model and the required training data at the same time.
- Additionally, by testing the model on the Adversarial SQuAD dataset, we showed that using KD can also increase the robustness of the model.

One of our main limitations, especially in the case of experimenting on the AL approach, has been the limited available graphics processing unit (GPU) resource. Although this restriction generally exists in all areas of deep learning, due to the nature of the AL method, this problem was more hindering. That is because in AL, every time a training set of

labeled data is fed to the model, the model must be retrained. Besides, we have evaluated the proposed approach on four extractive question-answering datasets. However, it can be evaluated on more such datasets to further illuminate the advantages and weaknesses of the approach.

For our future work, one interesting direction would be to further improve the effectiveness of KD by connecting the intermediate layers of the teacher and student models to transfer the knowledge between those layers. Recently, pre-trained models such as ALBERT (Lan et al., 2020), XLNet (Yang et al., 2019), and RoBERTa (Liu et al., 2019b) have been introduced that have managed to improve the performance in some downstream tasks. It is interesting to investigate the usage of these models as the teacher model to improve the performance in the QA task, too. Also, it may be beneficial if a combination of multiple teacher models would be used as an ensemble model.

CRediT authorship contribution statement

Yasaman Boreshban: Conceptualization, Methodology, Investigation, Validation, Writing – original draft. **Seyed Morteza Mirbostani:** Software, Validation, Data curation, Writing – original draft. **Gholamreza Ghassem-Sani:** Supervision, Conceptualization,Validation, Writing – review & editing. **Seyed Abolghasem Mirroshandel:** Supervision, Conceptualization, Validation, Writing – review & editing. **Shahin Amiriparian:** Conceptualization, Validation, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request

Appendix A. Implementation details

In this research, all the experiments have been performed on a system with an Intel Core i7-8700K CPU 3.70 GHz 6-Core, 32 GB of RAM, and a GeForce GTX 1080 with 8 GB of VRAM. Additionally, we have used PyTorch, NumPy, Pandas, Matplotlib, SpaCy, and HuggingFace Transformers software packages to implement the proposed approach.

The best hyperparameter values of the experiments are given in Table A.1. We conducted multiple experiments with different setups. Regarding the KD method, different values for the T and ρ parameters with a combination of different interpolation methods were tested. The specified values in the table are the optimum ones. For the AL cycle sampling, various methods such as random sampling, least confidence, maximum entropy, and the least confidence combined with clustering were employed. Moreover, we used the F1-Score and EM metrics to analyze the results of the experiments.

Appendix B. Variables

All the variables used in the formulations, algorithms, and text of this article are shown in Table B.2.

Appendix C. Abbreviations

Table C.3 describes various abbreviations and acronyms used throughout the paper.

Table A.1
The best hyperparameter values of the experiments are listed in this table. We experimented on some hyperparameters with multiple values and determined the optimum values, displayed with a bold typeface. Other hyperparameters were set to their default values. The *Tables* column refers to the corresponding tables regarding the experiments in which these hyperparameters were used.

Parameter	Tables	Type	Value
Teacher Model	3, 5, 9, 10, 11	default	BERT Base
Teacher Batch Size	3, 5, 9, 10, 11	default	32
BERT Truncation Length	3, 5, 9, 10, 11	default	400, **512**
Student Model	3, 5, 9, 10, 11	optimized	QANet
Student Batch Size	3, 5, 9, 10, 11	default	14
QANet Hidden Size	3, 5, 9, 10, 11	default	128
QANet Attention Heads	3, 5, 9, 10, 11	default	8
Word Embedding	3, 5, 9, 10, 11	default	GloVe
Word Embedding Size	3, 5, 9, 10, 11	default	2.2e6
Word Embedding Dimension	3, 5, 9, 10, 11	default	300
Character Embedding	3, 5, 9, 10, 11	default	GloVe
Character Embedding Size	3, 5, 9, 10, 11	default	94
Character Embedding Dimension	3, 5, 9, 10, 11	default	64
Context's Character Limit	3, 5, 9, 10, 11	optimized	400, **512**
Question's Character Limit	3, 5, 9, 10, 11	optimized	50
Answer's Character Limit	3, 5, 9, 10, 11	optimized	30
Word's Character Limit	3, 5, 9, 10, 11	optimized	16
KD Temperature (T)	3, 5, 9, 10, 11	optimized	1, 5, **10**, 20
KD Soft Loss Coefficient (ρ)	3, 5, 9, 10, 11	optimized	0.1, 0.3, 0.5, **0.7**, 0.9
KD Interpolation Method	3, 5, 9, 10, 11	optimized	Linear, Cubic, Quadratic
AL Initial Samples	6, 7, 8, 9, 10, 11	optimized	**1%**, 10%
AL Cycle Samples	6, 7, 8, 9, 10, 11	optimized	10%
AL Target Samples	6, 7, 8, 9, 10, 11	optimized	20%, 30%, **40%**, 50%, 60%, **70%**, 80%, 90%
AL Tuning Samples	6, 7, 8, 9, 10, 11	optimized	500
AL Sampling Strategy	6, 7, 8, 9, 10, 11	optimized	RAND, LC, M, EN, LC-CL
Student Model Optimizer	3, 5, 9, 10, 11	optimized	Adam
Learning Rate	3, 5, 9, 10, 11	default	0.001
Warm-up Steps	3, 5, 9, 10, 11	default	1000
Betas	3, 5, 9, 10, 11	default	0.8-0.999
Epsilon	3, 5, 9, 10, 11	default	1e-8
Weight Decay	3, 5, 9, 10, 11	default	3e-7

(*continued on next page*)

Table A.1 (*continued*).

EMA Decay	3, 5, 9, 10, 11	default	0.9999
Early Stop	3, 5, 9, 10, 11	default	10
Gradient Clipping Value	3, 5, 9, 10, 11	default	5.0
Epochs	3, 5, 9, 10, 11	optimized	20, **30**, 50

Table B.2
The list of variables used in the article.

Variable	Explanation
P	context paragraph
m	paragraph length
Q	question
A	answer
W	word
S	sub-word
C_{hard}	student model's actual labels (hard target)
C_{soft}	teacher model's output logits (soft target)
L_{CE}	cross-entropy loss function
L_{KD}	Kullback–Leibler divergence error function
L	weighted combination of C_{soft} and C_{hard}
y^1	one-hot for the start answer token
y^2	one-hot for the end answer token
ρ	weight of the hard and soft cross-entropy losses
p_i	softmax probability of the ith class of the student model
q_i	tempered softmax probability for the ith class of the teacher model
T	temperature
$\mathcal{U}$	unlabeled data pool
$\mathcal{L}$	labeled data set
$* \; x^*$	unlabeled candidate
$\hat{A}$	answer with the highest probability value
$\hat{A}_i$	the ith most probable answer
H_0	null hypothesis
α	significance level
p-value	conditional probability
B	resampling with replacement count

Table C.3
Various abbreviations and acronyms used in the paper.

Abbreviation	Explanation
AL	active learning
ALBERT	a lite bert
BERT	bidirectional encoder representations from transformer
BiDAF	bi-directional attention flow
CE	cross-entropy
CNN	convolutional neural network
DNN	deep neural network
EM	exact match
FPGA	field-programmable gate array
GLUE	general language understanding evaluation
GPU	graphics processing unit
GRU	gated recurrent unit
KD	knowledge distillation
KL	Kullback–Leibler
LSTM	long short-term memory
MLM	masked language modeling
MSE	mean squared error
NAS	neural architecture search
NER	named entity recognition
NLP	natural language processing
NSP	next sentence prediction
QA	question answering
QANet	question answering network
RMR	reinforced mnemonic reader
RNN	recurrent neural network
RoBERTa	robustly optimized BERT approach
SKD	self-knowledge distillation
SOTA	state-of-the-art
SQuAD	Stanford question answering dataset
SVD	singular value decomposition
TF-IDF	term frequency-inverse document frequency
USE	universal sentence encoder

References

Afan, H.A., Ibrahem Ahmed Osman, A., Essam, Y., Ahmed, A.N., Huang, Y.F., Kisi, O., Sherif, M., Sefelnasr, A., Chau, K.-w., El-Shafie, A., 2021. Modeling the fluctuations of groundwater level by employing ensemble deep learning techniques. Eng. Appl. Comput. Fluid Mech. 15 (1), 1420–1439.

Aggarwal, U., Popescu, A., Hudelot, C., 2020. Active learning for imbalanced datasets. In: 2020 IEEE Winter Conference on Applications of Computer Vision. WACV, pp. 1417–1426. http://dx.doi.org/10.1109/WACV45572.2020.9093475.

Amiriparian, S., Pugachevskiy, S., Cummins, N., Hantke, S., Pohjalainen, J., Keren, G., Schuller, B., 2017. CAST a database: Rapid targeted large-scale big data acquisition via small-world modelling of social media platforms. In: 2017 Seventh International Conference on Affective Computing and Intelligent Interaction. ACII, IEEE, pp. 340–345.

Arora, S., Khapra, M.M., Ramaswamy, H.G., 2019. On knowledge distillation from complex networks for response prediction. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). Association for Computational Linguistics, Minneapolis, Minnesota, pp. 3813–3822. http://dx.doi.org/10.18653/v1/N19-1382, URL https://www.aclweb.org/anthology/N19-1382.

Asami, T., Masumura, R., Yamaguchi, Y., Masataki, H., Aono, Y., 2017. Domain adaptation of DNN acoustic models using knowledge distillation. In: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing. ICASSP, pp. 5185–5189. http://dx.doi.org/10.1109/ICASSP.2017.7953145.

Banan, A., Nasiri, A., Taheri-Garavand, A., 2020. Deep learning-based appearance features extraction for automated carp species identification. Aquac. Eng. 89, 102053.

Bao, R., Wang, J., Zhao, H., 2021. Defending pre-trained language models from adversarial word substitution without performance sacrifice. In: Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021. Association for Computational Linguistics, pp. 3248–3258. http://dx.doi.org/10.18653/v1/2021.findings-acl.287, Online, URL https://aclanthology.org/2021.findings-acl.287.

Boroujeni, G.A., Faili, H., Yaghoobzadeh, Y., 2022. Answer selection in community question answering exploiting knowledge graph and context information. Semantic Web (Preprint), 1–18.

Cer, D., Yang, Y., Kong, S.-y., Hua, N., Limtiaco, N., St. John, R., Constant, N., Guajardo-Cespedes, M., Yuan, S., Tar, C., Strope, B., Kurzweil, R., 2018. Universal sentence encoder for english. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations. Association for Computational Linguistics, Brussels, Belgium, pp. 169–174. http://dx.doi.org/10.18653/v1/D18-2029, URL https://www.aclweb.org/anthology/D18-2029.

Chen, L., Fan, A., Shi, H., Chen, G., 2020. Search task success evaluation by exploiting multi-view active semi-supervised learning. Inf. Process. Manag. 57, 102180.

Chen, D., Fisch, A., Weston, J., Bordes, A., 2017. Reading wikipedia to answer open-domain questions. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Association for Computational Linguistics, Vancouver, Canada, pp. 1870–1879. http://dx.doi.org/10.18653/v1/P17-1171, URL https://www.aclweb.org/anthology/P17-1171.

Chen, W., Sharifrazi, D., Liang, G., Band, S.S., Chau, K.W., Mosavi, A., 2022a. Accurate discharge coefficient prediction of streamlined weirs by coupling linear regression and deep convolutional gated recurrent unit. Eng. Appl. Comput. Fluid Mech. 16 (1), 965–976.

Chen, C., Zhang, Q., Kashani, M.H., Jun, C., Bateni, S.M., Band, S.S., Dash, S.S., Chau, K.-W., 2022b. Forecast of rainfall distribution based on fixed sliding window long short-term memory. Eng. Appl. Comput. Fluid Mech. 16 (1), 248–261.

Cheng, Y., Wang, D., Zhou, P., Zhang, T., 2020. A survey of model compression and acceleration for deep neural networks. arXiv arXiv:1710.09282.

Chung, Y.-A., Lee, H.-Y., Glass, J., 2018. Supervised and unsupervised transfer learning for question answering. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers). Association for Computational Linguistics, New Orleans, Louisiana, pp. 1585–1594. http://dx.doi.org/10.18653/v1/N18-1143, URL https://www.aclweb.org/anthology/N18-1143.

Clark, K., Luong, M.-T., Khandelwal, U., Manning, C.D., Le, Q.V., 2019. BAM! Born-again multi-task networks for natural language understanding. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. Association for Computational Linguistics, Florence, Italy, pp. 5931–5937. http://dx.doi.org/10.18653/v1/P19-1595, URL https://aclanthology.org/P19-1595.

Devlin, J., Chang, M.-W., Lee, K., Toutanova, K., 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). Association for Computational Linguistics, Minneapolis, Minnesota, pp.

4171–4186. http://dx.doi.org/10.18653/v1/N19-1423, URL https://www.aclweb.org/anthology/N19-1423.

Dhingra, B., Danish, D., Rajagopal, D., 2018. Simple and effective semi-supervised question answering. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers). Association for Computational Linguistics, New Orleans, Louisiana, pp. 582–587. http://dx.doi.org/10.18653/v1/N18-2092, URL https://www.aclweb.org/anthology/N18-2092.

Dong, C., Wang, G., Xu, H., Peng, J., Ren, X., Liang, X., 2021. EfficientBERT: Progressively searching multilayer perceptron via warm-up knowledge distillation. In: Findings of the Association for Computational Linguistics: EMNLP 2021. Association for Computational Linguistics, Punta Cana, Dominican Republic, pp. 1424–1437. http://dx.doi.org/10.18653/v1/2021.findings-emnlp.123, URL https://aclanthology.org/2021.findings-emnlp.123.

Dror, R., Baumer, G., Shlomov, S., Reichart, R., 2018. The hitchhiker's guide to testing statistical significance in natural language processing. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Vol. 1, pp. 1383–1392.

Dunietz, J., Burnham, G., Bharadwaj, A., Rambow, O., Chu-Carroll, J., Ferrucci, D., 2020. To test machine comprehension, start by defining comprehension. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. Association for Computational Linguistics, pp. 7839–7859. http://dx.doi.org/10.18653/v1/2020.acl-main.701, Online, URL https://aclanthology.org/2020.acl-main.701.

Ebrahimi, J., Rao, A., Lowd, D., Dou, D., 2018. HotFlip: White-box adversarial examples for text classification. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). Association for Computational Linguistics, Melbourne, Australia, pp. 31–36. http://dx.doi.org/10.18653/v1/P18-2006, URL https://aclanthology.org/P18-2006.

Fan, Y., Xu, K., Wu, H., Zheng, Y., Tao, B., 2020. Spatiotemporal modeling for nonlinear distributed thermal processes based on KL decomposition, MLP and LSTM network. IEEE Access 8, 25111–25121.

Fu, Y., Zhu, X., Li, B., 2013. A survey on instance selection for active learning. Knowl. Inf. Syst. 35 (2), 249–283. http://dx.doi.org/10.1007/s10115-012-0507-8.

Gou, J., Yu, B., Maybank, S.J., Tao, D., 2021. Knowledge distillation: A survey. Int. J. Comput. Vis. 129, 1789–1819.

Hahn, S., Choi, H., 2019. Self-knowledge distillation in natural language processing. In: Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019). INCOMA Ltd., Varna, Bulgaria, pp. 423–430. http://dx.doi.org/10.26615/978-954-452-056-4_050, URL https://aclanthology.org/R19-1050.

Hinton, G.E., Vinyals, O., Dean, J., 2015. Distilling the knowledge in a neural network. arXiv arXiv:1503.02531.

Hu, M., Peng, Y., Huang, Z., Qiu, X., Wei, F., Zhou, M., 2018a. Reinforced mnemonic reader for machine reading comprehension. In: Proceedings of the 27th International Joint Conference on Artificial Intelligence. IJCAI '18, AAAI Press, pp. 4099–4106.

Hu, M., Peng, Y., Qiu, X., et al., 2017. Reinforced mnemonic reader for machine comprehension. CoRR, Abs/1705.02798.

Hu, M., Peng, Y., Wei, F., Huang, Z., Li, D., Yang, N., Zhou, M., 2018b. Attention-guided answer distillation for machine reading comprehension. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Brussels, Belgium, pp. 2077–2086. http://dx.doi.org/10.18653/v1/D18-1232, URL https://www.aclweb.org/anthology/D18-1232.

Ivgi, M., Berant, J., 2021. Achieving model robustness through discrete adversarial training. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, pp. 1529–1544. http://dx.doi.org/10.18653/v1/2021.emnlp-main.115, URL https://aclanthology.org/2021.emnlp-main.115.

Jalali Farahani, F., Ghassem-Sani, G., 2021. BERT-persNER: A new model for Persian named entity recognition. In: Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021). INCOMA Ltd., Held Online, pp. 647–654, URL https://aclanthology.org/2021.ranlp-1.73.

Jia, R., Liang, P., 2017. Adversarial examples for evaluating reading comprehension systems. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Copenhagen, Denmark, pp. 2021–2031. http://dx.doi.org/10.18653/v1/D17-1215, URL https://aclanthology.org/D17-1215.

Jiao, X., Yin, Y., Shang, L., Jiang, X., Chen, X., Li, L., Wang, F., Liu, Q., 2020. TinyBERT: Distilling BERT for natural language understanding. In: Findings of the Association for Computational Linguistics: EMNLP 2020. Association for Computational Linguistics, pp. 4163–4174. http://dx.doi.org/10.18653/v1/2020.findings-emnlp.372, Online, URL https://www.aclweb.org/anthology/2020.findings-emnlp.372.

Jin, D., Jin, Z., Zhou, J.T., Szolovits, P., 2020. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. In: Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 34, (05), pp. 8018–8025.

Joshi, M., Chen, D., Liu, Y., Weld, D.S., Zettlemoyer, L., Levy, O., 2020. SpanBERT: Improving pre-training by representing and predicting spans. Trans. Assoc. Comput. Linguist. 8, 64–77. http://dx.doi.org/10.1162/tacl_a_00300, URL https://aclanthology.org/2020.tacl-1.5.

Joshi, M., Choi, E., Weld, D., Zettlemoyer, L., 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Association for Computational Linguistics, Vancouver, Canada, pp. 1601–1611. http://dx.doi.org/10.18653/v1/P17-1147, URL https://aclanthology.org/P17-1147.

Kasai, J., Qian, K., Gurajada, S., Li, Y., Popa, L., 2019. Low-resource deep entity resolution with transfer and active learning. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. Association for Computational Linguistics, Florence, Italy, pp. 5851–5861. http://dx.doi.org/10.18653/v1/P19-1586, URL https://aclanthology.org/P19-1586.

Kočiskỳ, T., Schwarz, J., Blunsom, P., Dyer, C., Hermann, K.M., Melis, G., Grefenstette, E., 2018. The narrativeqa reading comprehension challenge. Trans. Assoc. Comput. Linguist. 6, 317–328.

Kolomiyets, O., Moens, M.-F., 2011. A survey on question answering technology from an information retrieval perspective. Inform. Sci. 181 (24), 5412–5434. http://dx.doi.org/10.1016/j.ins.2011.07.047, URL https://www.sciencedirect.com/science/article/pii/S0020025511003860.

Kranjc, J., Smailovic, J., Podpecan, V., Grcar, M., Žnidaršič, M., Lavrac, N., 2015. Active learning for sentiment analysis on data streams: Methodology and workflow implementation in the ClowdFlows platform. Inf. Process. Manag. 51, 187–203.

Lakshmi, K., Arivuchelvan, K.M., 2019. A survey on datasets for machine reading comprehension. In: Proceedings of International Conference on Recent Trends in Computing, Communication & Networking Technologies. ICRTCCNT, http://dx.doi.org/10.2139/ssrn.3429211, URL https://ssrn.com/abstract=3429211.

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R., 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In: International Conference on Learning Representations. URL https://openreview.net/forum?id=H1eA7AEtvS.

Le, T., Park, N., Lee, D., 2022. SHIELD: Defending textual neural networks against multiple black-box adversarial attacks with stochastic multi-expert patcher. In: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Association for Computational Linguistics, Dublin, Ireland, pp. 6661–6674. http://dx.doi.org/10.18653/v1/2022.acl-long.459, URL https://aclanthology.org/2022.acl-long.459.

Lewis, P., Denoyer, L., Riedel, S., 2019. Unsupervised question answering by cloze translation. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. Association for Computational Linguistics, Florence, Italy, pp. 4896–4910. http://dx.doi.org/10.18653/v1/P19-1484, URL https://aclanthology.org/P19-1484.

Li, W., Du, Y., Li, X., Chen, X., Xie, C., Li, H., Li, X., 2022a. UD_BBC: Named entity recognition in social network combined BERT-BiLSTM-CRF with active learning. Eng. Appl. Artif. Intell. 116, 105460.

Li, L., Ma, R., Guo, Q., Xue, X., Qiu, X., 2020a. BERT-ATTACK: Adversarial attack against BERT using BERT. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing. EMNLP, Association for Computational Linguistics, pp. 6193–6202. http://dx.doi.org/10.18653/v1/2020.emnlp-main.500, Online, URL https://aclanthology.org/2020.emnlp-main.500.

Li, B.Z., Stanovsky, G., Zettlemoyer, L., 2020b. Active learning for coreference resolution using discrete annotation. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. Association for Computational Linguistics, pp. 8320–8331. http://dx.doi.org/10.18653/v1/2020.acl-main.738, Online, URL https://www.aclweb.org/anthology/2020.acl-main.738.

Li, Z., Wang, Z., Tan, M., Nallapati, R., Bhatia, P., Arnold, A., Xiang, B., Roth, D., 2022b. DQ-BART: Efficient sequence-to-sequence model via joint distillation and quantization. In: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). Association for Computational Linguistics, Dublin, Ireland, pp. 203–211. http://dx.doi.org/10.18653/v1/2022.acl-short.22, URL https://aclanthology.org/2022.acl-short.22.

Li, P., Yi, J., Zhang, L., 2018. Query-efficient black-box attack by active learning. arXiv arXiv:1809.04913.

Liu, M., Buntine, W., Haffari, G., 2018. Learning to actively learn neural machine translation. In: Proceedings of the 22nd Conference on Computational Natural Language Learning. Association for Computational Linguistics, Brussels, Belgium, pp. 334–344. http://dx.doi.org/10.18653/v1/K18-1033, URL https://www.aclweb.org/anthology/K18-1033.

Liu, X., He, P., Chen, W., Gao, J., 2019a. Improving multi-task deep neural networks via knowledge distillation for natural language understanding. arXiv arXiv:1904.09482.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V., 2019b. RoBERTa: A robustly optimized BERT pretraining approach. arXiv arXiv:1907.11692.

Liu, M., Tu, Z., Wang, Z., Xu, X., 2020. LTP: A new active learning strategy for bert-CRF based named entity recognition. arXiv arXiv:2001.02524.

Mao, Y., Wang, Y., Wu, C., Zhang, C., Wang, Y., Yang, Y., Zhang, Q., Tong, Y., Bai, J., 2020. Ladabert: Lightweight adaptation of bert through hybrid model compression. arXiv preprint arXiv:2004.04124.

Min, S., Zhong, V., Socher, R., Xiong, C., 2018. Efficient and robust question answering from minimal context over documents. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Association for Computational Linguistics, Melbourne, Australia, pp. 1725–1735. http://dx.doi.org/10.18653/v1/P18-1160, URL https://www.aclweb.org/anthology/P18-1160.

Mirroshandel, S.A., Nasr, A., 2011. Active learning for dependency parsing using partially annotated sentences. In: Proceedings of the 12th International Conference on Parsing Technologies. Association for Computational Linguistics, Dublin, Ireland, pp. 140–149, URL https://aclanthology.org/W11-2917.

Momtazi, S., 2018. Unsupervised latent Dirichlet allocation for supervised question classification. Inf. Process. Manage. 54 (3), 380–393.

Oguntola, I., Olubeko, S., Sweeney, C., 2018. Slimnets: An exploration of deep model compression and acceleration. In: 2018 IEEE High Performance Extreme Computing Conference. HPEC, IEEE, pp. 1–6.

Papernot, N., McDaniel, P., Wu, X., Jha, S., Swami, A., 2016. Distillation as a defense to adversarial perturbations against deep neural networks. In: 2016 IEEE Symposium on Security and Privacy. SP, pp. 582–597. http://dx.doi.org/10.1109/SP.2016.41.

Rajpurkar, P., Zhang, J., Lopyrev, K., Liang, P., 2016. SQuAD: 100,000+ questions for machine comprehension of text. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Austin, Texas, pp. 2383–2392. http://dx.doi.org/10.18653/v1/D16-1264, URL https://www.aclweb.org/anthology/D16-1264.

Rashid, A., Lioutas, V., Ghaddar, A., Rezagholizadeh, M., 2021. Towards zero-shot knowledge distillation for natural language processing. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, pp. 6551–6561. http://dx.doi.org/10.18653/v1/2021.emnlp-main.526, URL https://aclanthology.org/2021.emnlp-main.526.

Ren, S., Deng, Y., He, K., Che, W., 2019. Generating natural language adversarial examples through probability weighted word saliency. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. Association for Computational Linguistics, Florence, Italy, pp. 1085–1097. http://dx.doi.org/10.18653/v1/P19-1103, URL https://aclanthology.org/P19-1103.

Ren, P., Xiao, Y., Chang, X., Huang, P.-Y., Li, Z., Gupta, B.B., Chen, X., Wang, X., 2021. A survey of deep active learning. ACM Comput. Surv. 54 (9), 1–40.

Sanh, V., Debut, L., Chaumond, J., Wolf, T., 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. arXiv arXiv:1910.01108.

Seo, M.J., Kembhavi, A., Farhadi, A., Hajishirzi, H., 2017. Bidirectional attention flow for machine comprehension. In: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings. OpenReview.net, URL https://openreview.net/forum?id=HJ0UKP9ge.

Settles, B., 2009. Active learning literature survey. Technical Report 1648, University of Wisconsin–Madison.

Shen, Y., Huang, P.-S., Gao, J., Chen, W., 2017a. Reasonet: Learning to stop reading in machine comprehension. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 1047–1055.

Shen, Y., Yun, H., Lipton, Z., Kronrod, Y., Anandkumar, A., 2017b. Deep active learning for named entity recognition. In: Proceedings of the 2nd Workshop on Representation Learning for NLP. Association for Computational Linguistics, Vancouver, Canada, pp. 252–256. http://dx.doi.org/10.18653/v1/W17-2630, URL https://www.aclweb.org/anthology/W17-2630.

Sun, S., Cheng, Y., Gan, Z., Liu, J., 2019a. Patient knowledge distillation for BERT model compression. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019. Association for Computational Linguistics, pp. 4322–4331. http://dx.doi.org/10.18653/v1/D19-1441.

Sun, S., Cheng, Y., Gan, Z., Liu, J., 2019b. Patient knowledge distillation for BERT model compression. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). Association for Computational Linguistics, Hong Kong, China, pp. 4323–4332. http://dx.doi.org/10.18653/v1/D19-1441, URL https://aclanthology.org/D19-1441.

Sun, Z., Yu, H., Song, X., Liu, R., Yang, Y., Zhou, D., 2020. MobileBERT: a compact task-agnostic BERT for resource-limited devices. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. Association for Computational Linguistics, pp. 2158–2170. http://dx.doi.org/10.18653/v1/2020.acl-main.195, Online, URL https://aclanthology.org/2020.acl-main.195.

Tahaei, M., Charlaix, E., Nia, V., Ghodsi, A., Rezagholizadeh, M., 2022. Kronecker-BERT: Significant compression of pre-trained language models through kronecker decomposition and knowledge distillation. In: Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. pp. 2116–2127.

Trischler, A., Wang, T., Yuan, X., Harris, J., Sordoni, A., Bachman, P., Suleman, K., 2017. NewsQA: A machine comprehension dataset. In: Proceedings of the 2nd Workshop on Representation Learning for NLP. Association for Computational Linguistics, Vancouver, Canada, pp. 191–200. http://dx.doi.org/10.18653/v1/W17-2623, URL https://aclanthology.org/W17-2623.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I., 2017. Attention is all you need. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (Eds.), Advances in Neural Information Processing Systems. Vol. 30, Curran Associates, Inc., URL https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Vinyals, O., Fortunato, M., Jaitly, N., 2015. Pointer networks. In: NIPS.

Wang, W., Bao, H., Huang, S., Dong, L., Wei, F., 2021a. MiniLMv2: Multi-head self-attention relation distillation for compressing pretrained transformers. In: Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021. Association for Computational Linguistics, pp. 2140–2151. http://dx.doi.org/10.18653/v1/2021.findings-acl.188, Online, URL https://aclanthology.org/2021.findings-acl.188.

Wang, W.-c., Du, Y.-j., Chau, K.-w., Xu, D.-m., Liu, C.-j., Ma, Q., 2021b. An ensemble hybrid forecasting model for annual runoff based on sample entropy, secondary decomposition, and long short-term memory neural network. Water Resour. Manag. 35, 4695–4726.

Wang, S., Jiang, J., 2017. Machine comprehension using match-LSTM and answer pointer. arXiv arXiv:1608.07905.

Wang, W., Yang, N., Wei, F., Chang, B., Zhou, M., 2017. Gated self-matching networks for reading comprehension and question answering. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Association for Computational Linguistics, Vancouver, Canada, pp. 189–198. http://dx.doi.org/10.18653/v1/P17-1018, URL https://aclanthology.org/P17-1018.

Xiong, C., Zhong, V., Socher, R., 2016. Dynamic coattention networks for question answering. arXiv preprint arXiv:1611.01604.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., Le, Q.V., 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In: NeurIPS.

Yang, Z., Hu, J., Salakhutdinov, R., Cohen, W., 2017. Semi-supervised QA with generative domain-adaptive nets. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Association for Computational Linguistics, Vancouver, Canada, pp. 1040–1050. http://dx.doi.org/10.18653/v1/P17-1096, URL https://www.aclweb.org/anthology/P17-1096.

Yang, Z., Qi, P., Zhang, S., Bengio, Y., Cohen, W., Salakhutdinov, R., Manning, C.D., 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Brussels, Belgium, pp. 2369–2380. http://dx.doi.org/10.18653/v1/D18-1259, URL https://aclanthology.org/D18-1259.

Yang, Z., Shou, L., Gong, M., Lin, W., Jiang, D., 2020. Model compression with two-stage multi-teacher knowledge distillation for web question answering system. In: Proceedings of the 13th International Conference on Web Search and Data Mining. pp. 690–698.

Yang, Y., Yih, W.-t., Meek, C., 2015. WikiQA: A challenge dataset for open-domain question answering. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Lisbon, Portugal, pp. 2013–2018. http://dx.doi.org/10.18653/v1/D15-1237, URL https://aclanthology.org/D15-1237.

Yasunaga, M., Leskovec, J., Liang, P., 2022. Linkbert: Pretraining language models with document links. In: Association for Computational Linguistics. ACL.

Yu, A.W., Dohan, D., Luong, M.-T., Zhao, R., Chen, K., Norouzi, M., Le, Q.V., 2018. Qanet: Combining local convolution with global self-attention for reading comprehension. arXiv arXiv:1804.09541.

Yu, Y., Zhang, W., Hasan, K., Yu, M., Xiang, B., Zhou, B., 2016. End-to-end answer chunk extraction and ranking for reading comprehension. arXiv arXiv:1610.09996.

Zahedi, M.S., Mansouri, B., Moradkhani, S., Farhoodi, M., Oroumchian, F., 2017. How questions are posed to a search engine? An empiricial analysis of question queries in a large scale Persian search engine log. In: 2017 3th International Conference on Web Research. ICWR, IEEE, pp. 84–89.

Zhang, X., Yang, A., Li, S., Wang, Y., 2019. Machine reading comprehension: a literature review. arXiv arXiv:1907.01686.