Journal of Telecommunication, Electronic and Computer Engineering
ISSN: 2180 – 1843 e-ISSN: 2289-8131 Vol. 15 No. 1
jtec.utem.edu.my

General Words Representation Method for Modern Language Model

Abbas Saliimi Lokman, Mohamed Ariff Ameedeen and Ngahzaifa Ab. Ghani
Faculty of Computing, Universiti Malaysia Pahang, Malaysia.
abbas@ump.edu.my

Article Info

Article history:
Received Oct 24th, 2022
Revised Dec 28th, 2022
Accepted Jan 17th, 2023

Index Terms:
Word representation
Word embedding
Language model
QA system

Abstract

This paper proposes a new word representation method emphasizes general words over specific words. The main motivation for developing this method is to address the weighting bias in modern Language Models (LMs). Based on the Transformer architecture, contemporary LMs tend to naturally emphasize specific words through the Attention mechanism to capture the key semantic concepts in a given text. As a result, general words, including question words are often neglected by LMs, leading to a biased word significance representation (where specific words have heightened weight, while general words have reduced weights). This paper presents a case study, where general words' semantics are as important as specific words' semantics, specifically in the abstractive answer area within the Natural Language Processing (NLP) Question Answering (QA) domain. Based on the selected case study datasets, two experiments are designed to test the hypothesis that "the significance of general words is highly correlated with its Term Frequency (TF) percentage across various document scales". The results from these experiments support this hypothesis, justifying the proposed intention of the method to emphasize general words over specific words in any corpus size. The output of the proposed method is a list of token (word)-weight pairs. These generated weights can be used to leverage the significance of general words over specific words in suitable NLP tasks. An example of such task is the question classification process (classifying question text whether it expects factual or abstractive answer). In this context, general words, particularly the question words are more semantically significant than the specific words. This is because the same specific words in different questions might require different answers based on their question words (e.g. "How many items are on sale?" and "What items are on sale?" questions). By employing the general weight values produced by this method, the weightage of question and specific words can be heightened, making it easier for the classification system to differentiate between these questions. Additionally, the token (word)-weight pair list is made available online at https://www.kaggle.com/datasets/saliimiabbas/genwords-weight.

I. INTRODUCTION

In order for a computer to effectively process natural language, words need to be represented by mathematical values that encapsulate their sematic meaning rather than just their basic atomic value. The process of converting a word's characteristics into these mathematical values, typically in the form of vectors, is called word embedding [1][2]. In the field of Natural Language Processing (NLP), computational Language Models (LM) perform the word embedding process.

To the best of authors' knowledge, the majority of modern Language Models employ the Transformer architecture [3][4][5][6][7][8][9][10]. Although originally proposed for Machine Translation tasks, the Transformer has revolutionized the Machine Learning (ML) field due to its ability to manage multiple parallel information in memory, which is particularly beneficial when handling with sequential data formats, such as a human natural language text. In Language Model (LM) implementations, the Transformer can represent words in a context-sensitive manner, also known as contextualized word embedding or dynamic word representation. The key component that enables this capability is the Attention mechanism [11]. By leveraging Attention, LM can 'remember' and prioritize important words within a given text sentence, learning each word's alignment with others. These crucial words primarily determine the semantical focus or intent of the sentence. Identifying this intent allows for more specialized NLP task, such as Question Answering (QA) and sentence classification, to be accomplished using the Transformer LM.

Although LM is usually used for generating word embeddings, the Transformer LMs can also perform downstream or specialized NLP tasks. This capability is achieved through two distinct but interconnected training processes applied to the Transformer LM: 1) Pre-training and 2) Fine-tuning [12]. During the pre-training phase, the LM is trained to 'understand' language by learning general semantical patterns based on the fundamental linguistic principle that "a word is characterized by the company it keeps" [13]. Once the pre-training process is complete, the LM 'remembers' these patterns through its trained neural network weight. At this stage, the LM can comprehend the meaning of each word in the trained language and these meanings can be extracted for further downstream NLP tasks (usually as 500-dimensional vector values for each word).

For certain NLP tasks that can be structured to fit the LM input pattern, fine-tuning the LM may be a more suitable choice. Fine-tuning involves training the pre-trained LM on a specific dataset in a supervised setup. For example, a pre-trained LM model can be fine-tuned using a QA dataset containing Questions, Answers and Contexts (a text corpora containing answers to the questions). In this process, the context and questions serve as input, while the answers function as output. Ultimately, the fine-tuned LM can identify patterns for improving answers based on the inputs of questions and contexts.

Finding an answer in context based on a given question is generally straightforward for a fine-tuned LM. However, challenges arise when reconstructing the answer to align with natural human conversational style. To demonstrate this scenario, consider the following corpus: "Currently shampoo, soap and conditioner are on sale ___" (context), "How many items are on sale?" (question), and "Three" (ground truth answer). Given the question, the QA system can identify "items on sale" as its focus or intent and produce "shampoo, soap and conditioner" as an answer. Compared to the ground truth answer, the generated response is unnatural.

This unnatural scenario is one of several phenomena categorized as abstractive answers (the presented case is part of the 'picking' phenomena [14]). Examining the sample QA corpus, the answer "shampoo, soap and conditioner" could also be the response to the question "What items are on sale?". Comparing the two questions, the words "How many" and "What" serve as the differentiating factors that characterize each query. This observation shows that question words (as well as other general words) are as important as specific words in certain cases, such as abstractive answers within the QA domain.

This paper proposes a novel method for representing general words, with a particular focus on question words. As general words are frequently used in the question text, the generated representation values can also be used to these words. The main contributions of this paper include:

- A proposed new method for calculating the weights of general words
- A collection of token (word) and weight data pairs that can be used in various suitable downstream NLP task

II. RELATED WORK

This section discusses related works on the area of the proposed method’s contributions. The first subsection, entitled “Word Representation Method” discusses the fundamental theory, advancement and existing gaps in the current method. The subsequent subsection, “General Words in Downstream NLP Tasks” further elaborates the previously mentioned gap, in particular when the word representation method (i.e., the LM) has undergone the fine-tuning process (a procedure that prepares the LM for downstream NLP task). In relation to the works in this section, the next section will directly introduce the proposed method, which aims to bridge the identified gap discussed herein. Overall, this section serves as a literature pathway for identifying the problems or gaps that the proposed method seeks to address.

A. Word Representation Method

Representing human language words in high-dimensional vector space has been a common practice for decades [15][16][17]. Based on the fundamental linguistic theory that "a word is characterized by the company it keeps" [13], methods for embedding every word in a dictionary into vector space have been proposed to represent a word's characteristics. With sufficient dimensions, these characteristics can also represent a word’s 'meaning', as demonstrated by Mikolov et al. [1] and Pennington et al. [2]. Two methods, namely word2vec [1] and GloVe [2] have shown that word embedding values can be used to identify words with similar meanings. By measuring the cosine distance between two word embedding vectors, semantically similar words appear close to each other, while semantically different words are distant.

Although highly regarded as a breakthrough in NLP, both word2vec and GloVe generate static word representations. This approach is not entirely ideal because human language is highly dynamic (encompassing synonymy, polysemy, antonym, etc.). Consider the following sentence: "The animal didn't cross the street because it was too ___". In this sentence, the word 'it' refers to 'animal' if the blank is filled with 'tired' and 'street' if the blank is filled with 'wide'. To correctly interpret the word 'it' based on the context of the sentence, dynamic work representation is necessary. To address this challenge, modern Word Representation (WR) methods have been developed to produce Contextualized Word Embedding [3][4][5]. These methods represent words in highly dynamic settings by generating embedding based on the current context. Referring to the previous sentence example, these methods can correctly align the word 'it' with its appropriate reference.

Most modern WR methods capable of producing contextualized embedding utilize the Attention mechanism [11]. With Attention, words that frequently co-occur in the same context receive higher weight (significant in value) than words that co-occur often but in different contexts. For example, consider the sentence "The cat has three kittens ___" in a story entitled "Cat" and the sentence "The apple tree has lots of fruits ___" in a story entitled "Apple tree". Assuming each sentence is from a different context (different stories), the words' cat', 'kittens', 'apple', and 'fruits' will have high weight values since they co-occur frequently in the same context. In contrast, conjunction words such as ‘the’ and 'has' will have low weight values because they co-occur frequently in different contexts. In other words, words with high weightage carry high semantic value, while words with low weightage have low semantical value (conjunction, question words and other general words).

The current word weightage setup in language model is generally effective, as it aligns with the natural structure of language. However, in certain cases, such as question-answering task, general or question words play a crucial role generating precise output. For example, consider the question “How many items are on sale?" and the context passage "Currently shampoo, soap and conditioner are on sale ___". According to the attention design, words like "items" and "on-sale" receive higher weightage, leading to the system to produce an answer like "shampoo, soap, and conditioner are on sale". While this answer is partially correct, humans would typically provide a more accurate response, such as “three items are on sale" or simply just "three" (one of the abstractive answers phenomenon [18]. This demonstrates the importance of the question phrase "how many" in generating the desired answer "three". Based on this observation, it can be argued that the current QA-LM, fined tuned with QA dataset, exhibit a bias towards significant words, undervaluing general words that are essential for producing precise results.

B. General Words in Downstream NLP Tasks

Modern language models do not entirely disregard general words; in fact, they dynamically represent such words based on the given context. For example, the meaning of the word 'it' in the sentence "The animal didn't cross the street because it was too ___" can represent either 'animal' or 'street' depending on the word filling the blank, either 'tired' or 'wide'. However, many general words are still overlooked by LM, which is particularly evident in the field of QA, where models are fine-tuned with the QA datasets. As the primary objective of a fine-tuned QA-LM is to provide contextually relevant answers, it tends to focus on specific words in the question while neglecting general and even question words. This happens because, during the fine-tuning process, the LM model's attention mechanism is trained to align semantically similar words among the question, answer and context text. The model prioritizes finding answers within the context text, leading to an increased focus on word alignments between the answer and context while lessening the attention given to dissimilar words. Consequently, question words receive less attention since they primarily appear in the question text, and not in the context paragraph or answer text.

III. GENERAL WORDS REPRESENTATION METHOD

This paper proposes a new method for generating weight value for word representation, emphasizing general and question words. The outcome of this method is a list of token (word)-weight pairs designed to leverage the significance of general words against specific words. Two sets of experiments have been conducted to evaluate this method. The following subsections will discuss both the experiment setups and the results.

A. Experiment Setup

The experiments were designed to test our initial hypothesis: "The significance of general words is highly correlated with its Term Frequency (TF) percentage across various document scales". To validate this hypothesis, two separate but interconnected experiments were conducted. Both experiments employed the same algorithm; the first experiment used only the CoQA [19] dataset (experiment A), while the second experiment used CoQA [19], SQUAD [20] and QuAC [21] datasets (experiment B). Notably, all used datasets are highly regarded in the QA research domain. For both experiments, only question texts from each dataset were used (a combination of Question texts from both Train and Dev/Val datasets). Different scale datasets were used to demonstrate the scalability for the proposed method. Experiment A processed tokens 662,607 tokens containing 17,404 unique tokens, while experiment B processed 2,833,785 unique tokens, a difference of 7,433 unique tokens and 2,171,178 total processed tokens. For both experiments, the WordPiece tokenizer [3] was used to segregate token from the original input text.

B. Algorithm

As mentioned earlier, this method aims to produce a list of token-weight pairs. While the tokens are generated using a standard NLP tokenisers (WordPiece tokeniser [3]), the weights are calculated directly using the following algorithm. The full algorithm to produce token-weight pair data for the chosen QA dataset is presented below:

Algorithm 1 Algorithm to calculate word's weight
Input: All question text from the dataset
Output: token-weight pair

```
    Initialisation: Retrieve all question text from raw QA
    dataset
 1: Combine all question text into one long word sequence
 2: Tokenise the long sequnece using WordPiece tokenizer
 3: while token do
 4:     if token = ‘?’ or ‘’’ or ‘,’ or ‘.’ then
 5:         remove token
 6:     else if token ≠ alphabets (i.e. number only) then
 7:         remove token
 8:     else if token = previous tokens (duplicate) then
 9:         remove token
10:     else
11:         token += Unique Token (UT)
12:     end if
13: end while
14: while UT do
15:     Term Frequency (TF) = UT counts in long sequence
16:     weight (i.e. TF percentage) = (TF / UT) × 100
17: end while
18: return list of token-weight pair
```

IV. RESULT ANALYSIS

Using the previously discussed algorithm, the Term Frequency (TF) percentage value for each unique token (UT) is calculated. This value, defined as the weight value (line 16 in the algorithm), constitutes the token-weight pair data, which serves as the output of the algorithm. To justify the use of the TF percentage as the weight value for general words, this section examines the correlation between the TF percentage and usage distribution of every UT in the experimented datasets. The results from both experiments A and B are analyzed to provide insights into the logic of the proposed algorithm.

Table 1
General Word Weighting Overall Result

	Experiment A	Experiment B
Dataset	CoQA	CoQA, SQUAD and QuAC
Total Token (TT)	662,607	2,833,785
Total Unique Token (UT)	17,404	24,455
Maximum %	5.92%	5.55%
Minimum %	0.00015%	0.00018%

Table 1 presents the overall key parameters for experiments A and B, while Table 2, Table 3, Figure 1 and Figure 2 presents more detailed analyses. Referring to Table 1, the total tokens for experiments A and B are 662,607 and 2,833,785, respectively, with experiment B being more than 300% larger in scale compared to experiment A. The difference in total UTs is 7,051 tokens, with experiment A's UT count comprising 71.17% of experiment B's. Only slight differences are recorded for maximum and minimum percentages, with only 0.37% for the maximum and 0.00003% for the minimum.

For a detailed analysis, the TF percentage values for each token are divided into ten clusters. It is worth noting that a higher percentage value indicates that the token is more general or common, justified by its frequent usage in the given dataset. Conversely, a lower percentage value implies that the token is more specific or uncommon, supported by its lack of usage in the given dataset. Referring to Figure 1 for experiment A and Figure 2 for experiment B, a significant portion of each dataset is dominated by the lowest cluster of TF percentage (53% for experiment A and 61% for experiment B). These results suggest that there is a considerable usage gap between general and specific words in the given datasets.

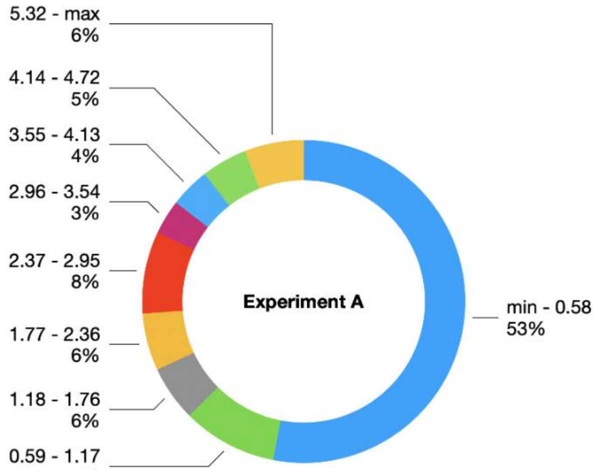

Figure 1. Distribution of Unique Token (UT) TF Percentage for Experiment A

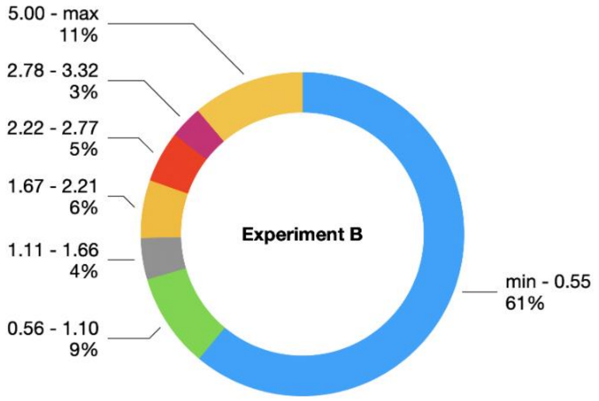

Figure 1. Distribution of Unique Token (UT) TF Percentage for Experiment B

For a more into detailed view of the ten segmented clusters, Table 2 and Table 3 show the TF percentage range, total TF count and its percentage, as well as the total UT count for each cluster. Comparing both tables, it is evident that the UT count for the nine highest clusters is the same (25 UT count), although the TF count differs significantly (a difference of 806,124 TF count, with 312,088 in experiment A and 1,118,212 in experiment B). This result indicated that although the dataset is scaled up (with more texts), the number of unique general words used in the text is immensely similar. Examining those 25 UT more closely, the following list presents the words that each token represents, ordered from the highest to lowest TF percentage:

- Experiment A: [what] [the] [did] [was] [he] [who] [is] [to] [how] [it] [of] [in] [they] [where] [a] [does] [she] [do] [when] [his] [for] [many] [that] [s] [were]
- Experiment B: [the] [what] [did] [of] [was] [in] [is] [to] [he] [who] [how] [a] [when] [for] [s] [are] [do] [it] [where] [they] [does] [were] [any] [and] [that]

Table 2
Detail TF Percentage Distribution over Ten Clusters (Experiment A)

Cluster	Total % in cluster	TF count	UT count	Sum
Min – 0.58	52.90	350,519	17,379	N/A
0.59 – 1.17	9.45	62,616	11	11
1.18 – 1.76	5.53	36,642	4	15
1.77 – 2.36	5.72	37,901	3	18
2.37 – 2.95	8.19	54,268	3	21
2.96 – 3.54	3.48	23,059	1	22
3.55 – 4.13	3.99	26,438	1	23
4.14 – 4.72	4.56	30,215	1	24
4.73 – 5.31	0.00	0	0	24
5.32 – Max	5.92	39,226	1	25
Total	**100**	**662,607**	**17,404**	**25**

Table 3
Detail TF Percentage Distribution over Ten Clusters (Experiment B)

Cluster	Total % in cluster	TF count	UT count	Sum
Min – 0.55	60.54	1,715,573	24,430	N/A
0.56 – 1.10	9.36	265,242	14	14
1.11 – 1.66	4.11	116,469	3	17
1.67 – 2.21	5.74	162,659	3	20
2.22 – 2.77	5.08	143,956	2	22
2.78 – 3.32	3.26	92,381	1	23
3.33 – 3.88	0.00	0	0	23
3.89 – 4.43	0.00	0	0	23
4.44 – 4.99	0.00	0	0	23
5.00 – Max	11.11	314,834	2	25
Total	**100**	**2,833,785**	**24,455**	**25**

From the listed 25 words in both experiments, only three words in each set are not similar between the two (denoted by underlines). Although note entirely identical, this result shows that common general word usage is highly consistent, even when considered across different dictionary sizes. This finding supports our hypothesis that " the significance of general words is highly correlated with its Term Frequency (TF) percentage across various document scales". Furthermore, this result justifies the rationale behind using the TF percentage value as the general word representation value, as a higher percentage value indicates that the word or token is more general or common within the dataset.

V. FUTURE WORK

This paper presents a new word representation method that emphasizes general and question words. The output of this method is a token-weight pair data, which can be used in any suitable downstream NLP tasks. For future work, the aforementioned case study (classifying question text based on whether it expects factual or abstractive answers) could be explored. A small-scale manual experiment could be conducted to further validate the merit of the generated words' weights derived from the proposed method. An ideal setup would involve selecting five to ten different question types, tokenizing each question text, generating its respective vector word representations, multiplying the vector by the scalar value of general weight, concatenating all vectors words representations into a vector sentence representation, and finally comparing the produced vectors (before and after general weight multiplication) to analyze the changes in vector representation. In theory, similar question types should have vector values close to each other, while different question types would exhibit divergent vector values. Once the general weight value is justified through this small-scale experiment, a much larger experimental setup would be an ideal subsequent future work.

VI. CONCLUSION

This paper proposes a new word representation method emphasizes general and question words. The proposed method is considered new in the following aspects: 1) A method to counterbalance the semantic value of general words in current LMs, and 2) Producing token-weight values that can be implemented in any Attention-based LM. Two experiments were designed to test the hypothesis that "the significance of general words is highly correlated with its Term Frequency (TF) percentage in any scale of documents". The results from these experimental setups support the acceptance of this hypothesis. In addition to the future work discussed in the previous section, another direction for this research is to identify more cases where the significance of general words can be increased using the generated general word weights. As the weights are stored in token-weight pairs, the leverage of general word significance can be implemented in pre-LM processing, such as after tokenizing the input, or in post-LM processing, when an additional classifier is needed. Ultimately, it is hoped that the output generated by this method will prove beneficial and find widespread adoption among researchers in the field.

ACKNOWLEDGMENT

The authors would like to thank Universiti Malaysia Pahang for providing financial support under Fundamental Research Grant (RDU) No. UMP.05/26.10/03/RDU220315.

REFERENCES

[1] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space", *ArXiv Preprint*, ArXiv:1301.3781, 2013.

[2] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation", *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, 2014.

[3] J. Devlin, M. W. Chang, K, Lee, K. and Toutanova, "Bert: Pretraining of deep bidirectional transformers for language understanding", *ArXiv Preprint*, ArXiv:1810.04805, 2018.

[4] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le, "XLNet: Generalized Autoregressive Pretraining for Language Understanding", *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pp. 5753–5763, 2019.

[5] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, V. Stoyanov, and P. G. Allen, "Roberta: A robustly optimised bert pre-training approach", *ArXiv Preprint*, ArXiv:1907.11692, 2019.

[6] S. Zhang, S. Roller, N. Goyal, M. Artetxe, M. Chen, S. Chen, C. Dewan, M. Diab, X. Li, X. V. Lin, and T. Mihaylov, "Opt: Open pre-trained transformer language models", *ArXiv Preprint*, ArXiv:2205.01068, 2022.

[7] H. Sajjad, N. Durrani, F. Dalvi, F. Alam, A. R. Khan, and J. Xu, "Analyzing encoded concepts in transformer language models." *ArXiv Preprint,* ArXiv:2206.13289, 2022.

[8] S. Borgeaud, A. Mensch, J. Hoffmann, T. Cai, E. Rutherford, K. Millican, G. B. Van Den Driessche, J. B. Lespiau, B. Damoc, A. Clark, and D. De Las Casas, "Improving language models by retrieving from trillions of tokens." *International conference on machine learning*. PMLR, 2022.

[9] A. Haviv, O. Ram, O. Press, P. Izsak, and O. Levy, "Transformer Language Models without Positional Encodings Still Learn Positional Information." *ArXiv preprint,* ArXiv:2203.16634, 2022.

[10] H. B. Zia, I. Castro, A. Zubiaga, and G. Tyson, "Improving Zero-Shot Cross-Lingual Hate Speech Detection with Pseudo-Label Fine-Tuning of Transformer Language Models." *Proceedings of the International AAAI Conference on Web and Social Media*, Vol. 16. 2022.

[11] A. Vaswani, G. Brain, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need", *Advances in Neural Information Processing Systems*, 30, 2017.

[12] R. Li, W. Xiao, L. Wang, H. Jang, and G. Carenini, "T3-Vis: visual analytic for Training and fine-Tuning Transformers in NLP", *In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 220-230, 2021, November.

[13] J. R. Firth, "A synopsis of linguistic theory", 1930–1955. *Studies in Linguistic Analysis (Special Volume of the Philological Society),* 1952(59), pp. 1–32, 1957.

[14] A. S. Lokman, M. A. Ameedeen, and N.A. Ghani, "A Question-Answering System that Can Count", *Computational Science and Technology*, Lecture Notes in Electrical Engineering, vol. 724, pp. 61–70, 2021.

[15] J. Elman, "Finding Structure in Time", *Cognitive Science*, 14, pp. 179-211, 1990.

[16] G. E. Hinton, J. L. McClelland, D. E. Rumelhart, "Distributed representations", *Parallel distributed processing: explorations in the microstructure of cognition*, vol. 1: foundations, pp. 77–109, 1986.

[17] D. E. Rumelhart, and R. J. Williams, "Learning internal representations by back-propagating errors", *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Eds.*, 1986.

[18] M. Yatskar, "A qualitative comparison of CoQA, SQuAD 2.0 and QuAC", *arXiv preprint*, arXiv:1809.10735, 2018. Chicago

[19] S. Reddy, D. Chen, and C. D. Manning, "CoQA: A Conversational Question Answering Challenge", *Transactions of the Association for Computational Linguistics*, 7, pp. 249–266, https://doi.org/10.1162/TACL A 00266/43511, 2019.

[20] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "Squad: 100,000+ questions for machine comprehension of text", *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2383–2392, https://arxiv.org/abs/1606.05250, 2016.

[21] E. Choi, H. He, M. Iyyer, M. Yatskar, W. Yih, Y. Choi, P. Liang, and L. Zettlemoyer, "QuAC: Question answering in context", *ArXiv Preprint*, ArXiv:1808.07036, 2018.